# Integration Cloud

Version 6.6.0

August 2020

WEBMETHODS

This document applies to webMethods Integration Cloud 6.6.0 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

**Document ID: IL-OLH-660-20200929**

# Table of Contents

# webMethods Integration Cloud

Software AG Cloud is the cloud-based, secure and reliable Platform-as-a-Service (PaaS) suite from Software AG and addresses today's business needs with unmatched speed, ease-of-use, and full support for social and mobile collaboration.

webMethods Integration Cloud is the Integration Platform as a Service (iPaaS) offering from Software AG and is a part of the Software AG Cloud family of products. Integration Cloud enables you to integrate your cloud-based Software as a Service (SaaS) applications, with other cloud-based applications. It also integrates your SaaS applications with on-premises applications.

Integration Cloud provides service-based integration for faster development and deployment. It enables cloud-to-any integration and connects cloud-based SaaS applications and on-premises ESB implementations. The multi-tenant environment scales elastically based on demand. Delivered as a service, the solution empowers your subject matter experts and eliminates integration silos. You can integrate applications hosted in public or private clouds, as well as applications hosted on-premises, reduce the dependency on IT and integrate your SaaS applications faster.

Integration Cloud enables:

- Lightweight integration on public and private clouds
- Easy-to-configure cloud-to-cloud integrations
- Secure and reliable hybrid integrations
- Elastic scalability managed automatically based on usage

Integration Cloud also enables cloud deployment of on-premises integrations. You can build integration projects in Software AG Designer, a full-featured Eclipse™-based environment, and deploy new or existing projects directly into the cloud.

With Integration Cloud, you can use the same set of tools to develop, debug and test integrations for the cloud as you do for on-premises integration. When you want to deploy your existing integrations to the cloud, Integration Cloud combines powerful developer tooling with a SaaS platform. You can select from a range of deployment landscapes, from simple to production-ready configurations that include clustered webMethods Integration Servers, Universal Messaging and Terracotta In-Memory Data Management. Your landscape will be provisioned automatically—and your integrations, APIs, and business logic deployed with it.

Develop integrations faster in Integration Cloud with:

- Rich data mapping, a large collection of built-in services, and data and cloud applications
- Easy conversion of Java® or flow services into APIs
- Familiar debugging, version control and test tools, while deploying to a multi-function iPaaS
- Integration to on-premises resources through VPN for secure data transmission
- Dashboards to monitor solution health and flow service behavior

Integration Cloud is intended for you if you have a requirement to integrate and synchronize data between multiple SaaS applications, as well as integrate your existing on-premises applications with cloud-based SaaS applications. This solution is delivered as a service, offered on a subscription basis, and is available in multiple packages and price tiers.

# Document Conventions

| Convention | Description |
|---|---|
| **Bold** | Identifies elements on a screen. |
| Narrowfont | Identifies service names and locations in the format *folder.subfolder.service*, APIs, Java classes, methods, properties. |
| *Italic* | Identifies:<br>Variables for which you must supply values specific to your own situation or environment.<br>New terms the first time they occur in the text.<br>References to other documentation sources. |
| `Monospace font` | Identifies:<br>Text you must type in.<br>Messages displayed by the system.<br>Program code. |
| { } | Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols. |
| \| | Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the \| symbol. |
| [ ] | Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols. |
| ... | Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...). |

# Online Information and Support

## Software AG Documentation Website

You can find documentation on the Software AG Documentation website at http://documentation.softwareag.com. The site requires credentials for Software AG's Product Support site Empower. If you do not have Empower credentials, you must use the TECHcommunity website.

## Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at https://empower.softwareag.com/.

You can find product information on the Software AG Empower Product Support website at https://empower.softwareag.com.

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.asp and give us a call.

## Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

# Data Protection

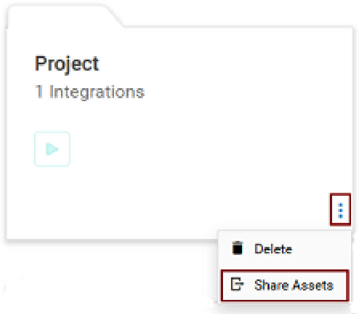
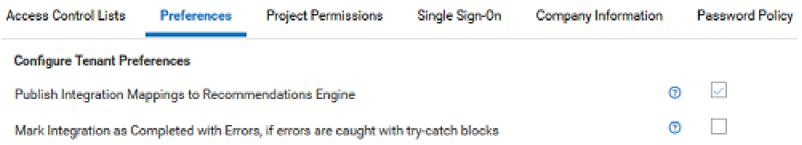
Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

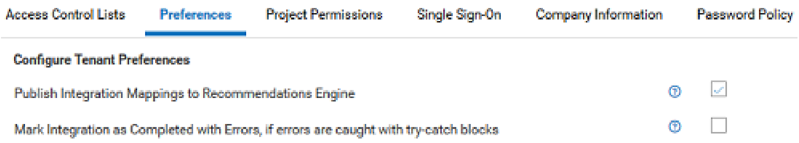
# 1 What's New

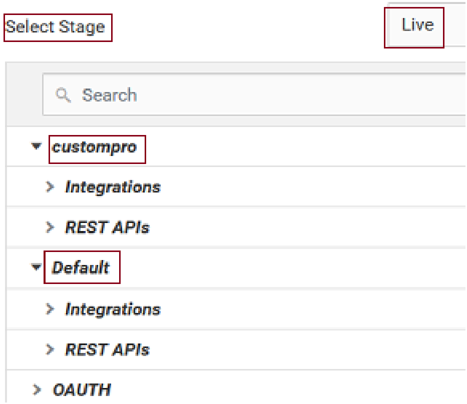
# Integration Cloud and Cloud Deployment

## Version 6.6.0 (August 2020)

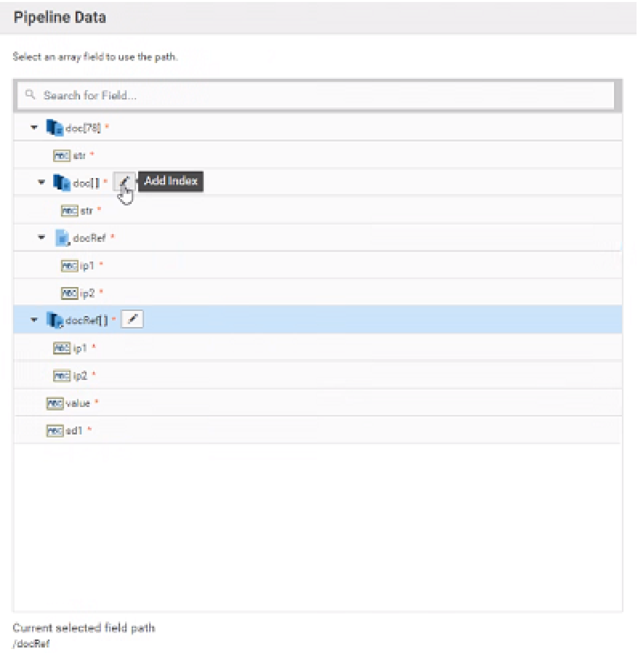
### Integration Cloud

| Item | Description |
|---|---|
| "Support for sharing assets across projects" on page 98 | You can now share integrations and document types across projects by clicking the vertical ellipses icon available on a project and selecting the **Share Assets** option. Asset sharing allows you to use the same assets in other projects.<br><br> <br>You can share assets with other projects only in the **Development** stage and if you are assigned to the **Administrator** Access Profile. |
| "Support for Smart mapping recommendations " on page 406 | Smart mapping provides you with intelligent mapping recommendations regarding which fields should be mapped in the pipeline. You can enable the smart mapping functionality by selecting the **Settings > Preferences** > **Publish Integration Mappings to Recommendations Engine** option.<br><br> Access Control Lists<br>Preferences<br>Project Permissions<br>Single Sign-On<br>Company Information<br>Password Policy<br>Configure Tenant Preferences<br>Publish Integration Mappings to Recommendations Engine ☑<br>Mark Integration as Completed with Errors, if errors are caught with try-catch blocks ☐<br>A Machine Learning (ML) algorithm is applied to provide the suggestions. This algorithm learns from the mappings you create and automatically provides suggestions to map similar fields.<br>If you do not select this option, you will not see the **Recommend Mappings** check box while mapping the |

| Item | Description |
|---|---|
| | fields in the pipeline. For all trial tenants, the mapping data is collected by default.<br>**Note:**<br>For paid tenants, only an Admin user who has access profile **ID 4**, has the permission to enable this feature. For trial tenants and Free Forever Edition, this is always enabled and cannot be changed. |
| "Capability to mark integrations as Completed with Errors, if errors are caught with try-catch blocks" on page 58 | You can enable this option in the new **Settings > Preferences > Configure Tenant Preferences** section.<br><br>Access Control Lists · Preferences · Project Permissions · Single Sign-On · Company Information · Password Policy<br>Configure Tenant Preferences<br>Publish Integration Mappings to Recommendations Engine ☑<br>Mark Integration as Completed with Errors, if errors are caught with try-catch blocks □<br>Select this option to view on-screen messages when an exception occurs while executing an operation or a service within an integration using a Try-Catch block.<br>When this option is enabled, if an exception occurs within a Try block, then the end result of the integration execution is shown as completed with errors. Even if the error is handled in the Catch block, the integration is shown as completed with errors.<br>When this option is not selected, if an exception occurs within a Try block, then the end result of the integration execution is shown as integration completed successfully. |
| "Support for creating a document type with optional fields" on page 637 | In earlier releases, when you loaded an XML or a JSON file to create a document type from scratch, by default, all the fields were marked as mandatory. In this release, after you click the **Load JSON** or **Load XML** options, you can use the **Required** option ◉ Required to mark all the fields as optional or mandatory. By default, all the fields are marked as mandatory. If a field is mandatory, you must pass a value for that field when running an integration. |
| "New Services" on page 524 | The following **MIME services** are now available in Integration Cloud. You can use MIME services to create MIME messages and extract information from MIME messages.<br>■ addBodyPart |

| Item | Description |
|---|---|
| | ■ addMimeHeader<br>■ createMimeData<br>■ getBodyPartContent<br>■ getBodyPartHeader<br>■ getContentType<br>■ getEnvelopeStream<br>■ getMimeHeader<br>■ getNumParts<br>■ getPrimaryContentType<br>■ getSubContentType<br>■ mergeHeaderAndBody |
| “OAuth 2.0 scope management” on page 84 | In earlier releases, while creating or updating a scope, exposed integrations and REST resources available only in the **Default** project were available for selection in the **Services** panel.<br>In this release, the **Services** panel displays the exposed integrations and REST resources available in all projects, that is, in custom projects and in the Default project *in the selected stage*.<br> Select Stage<br>Live<br>Search<br>custompro<br>Integrations<br>REST APIs<br>Default<br>Integrations<br>REST APIs<br>OAUTH |
| “Monitor page enhancements” on page 662 | You can now restart or resume an integration execution from the *Monitor* > *Dashboard* page. You can also restart or resume an integration execution from the *Monitor* > *Execution Results* page and from the *Last 5 Execution Results* page. |

| Item | Description |
|---|---|
| Dedicated infrastructure support for hybrid integration scenarios | Performance, scalability, and availability of on-premises connectivity for hybrid integration scenarios have now been enhanced by having dedicated Software AG Universal Messaging (UM) nodes for each tenant. A tenant can be associated with dedicated UMs based on the license. Contact Software AG Global Support for assistance in setting up the dedicated hybrid infrastructure.<br>If you are using hybrid connectivity and when you update the on-premises settings for the first time after the v6.6.0 upgrade, restart the on-premises webMethods Integration Server to resume hybrid connectivity.<br>If you have whitelisted the Cloud UM hostname or IP in the firewall, then you have to also whitelist the new UM hostname and IP along with the old ones. Click "here" on page 323 for information on the IP addresses. |
| Performance improvement | In this release, a performance improvement of 70 percent is observed on integration execution throughput. |
| Parameters page changes | The *Active* option is now removed from the **Parameters** page while creating an operation for a REST Application to improve the usability. All parameters are now provided as part of the input signature. |
| Support for encoding of URI context parameters in custom REST Applications | Integration Cloud now supports encoding of URI context parameters. For example, space is now encoded with %20: POST /pubapi/v1/fs-content/Shared/test2/Egnyte%20Logo.png and # is encoded with %23: POST /pubapi/v1/fs-content/Shared/test2/Egnyte%23Logo.png |
| Support for executing or debugging integrations generated from APIs | You can now run and debug integrations generated from SOAP APIs using WSDL and REST APIs using Swagger. |
| Support for using the field element inside an array list | While selecting a field for the conditions, that is, while using the If, or Loop, or Switch statements, you can now click on the **Select Field** expression and choose a field in the **Pipeline Data** dialog box to add its path to the condition. If you want to use the field element inside an array list, select the **Add Index** option to add the index and use the indexed field path. |

| Item | Description |
|---|---|
|  | <br>Pipeline Data<br>Select an array field to use the path.<br>Search for Field...<br>doc[78] *<br>str *<br>doc[ ] *<br>Add Index<br>str *<br>docRef *<br>ip1 *<br>ip2 *<br>docRef[ ] *<br>ip1 *<br>ip2 *<br>value *<br>ad1 *<br>Current selected field path<br>/docRef<br>While selecting a field for the iterations, that is, while using the **for-each** statements, you can click now on the **Select Field** expression and choose an array in the **Pipeline Data** dialog box to add its path to the iteration. If you want to use the array element inside the array list, select the **Add Index** option to add the index and use the indexed field path. |
| "Troubleshooting tips on Account configurations" on page 364 | The Help set now contains a new section, which provides troubleshooting tips to some of the most common questions on Account configurations. |

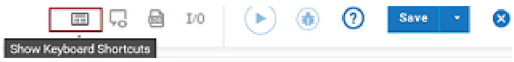
# Version 6.1.0 (April 2020)

## Integration Cloud

| Item | Description |
|---|---|
| Integration workspace enhancements | ■ On the *Integrations* page, you can now click the icon to see the details of an Integration, click the ellipsis ⋮ icon to delete or copy an Integration, point to the icon to view the scheduled status, scheduled type and when is the next scheduled execution date and time, and to the 🌐 icon to view the request URL. |

| Item | Description |
|---|---|
| | ■ The look and feel of the blocks have been enhanced and the size of the blocks reduced to increase the work space.<br>■ You can now copy a block from an integration and paste that block in another integration across projects. Currently, blocks cannot be pasted across different browsers or across domains.<br>■ The ≡ icon has been removed from the blocks. The *Copy*, *Delete*, *Comment*, *Disable*, and *Duplicate* options are now available by right-clicking on a block.<br>■ The *Modify Mapping* ( ) option now opens the pipeline data window for mapping. Further, mapping arrows have been removed from the source and target fields in the mapping window. |
| Monitor page enhancements | ■ On the *Dashboard* page, you can now filter the execution details of Integrations based on the invocation channel, for example, Scheduler, User Interface, HTTP Interface, REST APIs, SOAP APIs, and Listeners. The in-progress executions can also be filtered based on the invocation channels.<br>■ You can now terminate multiple in-progress integration executions from the *Dashboard* page by selecting the in-progress integration executions and clicking *Terminate*.<br>■ On the *Execution Results* page, you can now select the *Invocation* channel for which you want to view the integration execution results. You can select All Invocations, All Projects, and All Integrations, if you want to view the execution details of Integrations based on all the invocation channels, for all integrations, and in all projects in the active stage, for the specified time period.<br>■ The performance and responsiveness of the Monitor screen to high volume of data have been enhanced. |
| Flat File enhancements | Integration Cloud now supports flat files that do not have a record identifier. You can select the *Yes* or *No* options on the Flat File Definition page depending on whether the flat file contains or does not contain a record identifier. The following options appears on the Flat File Definition page if you select *Yes*: |

| Item | Description |
|---|---|
| | ■ *Start at position* - Identifies the character position in the record (counting from zero) where the record identifier is located.<br>■ *Nth field* - Identifies the field in the record (counting from zero) that contains the identifier. |
| Support for changing the Authentication Type | While editing an Account in any stage, you can now select a different *Authentication Type* without impacting any integrations. So if an integration is using an Account with a specified *Authentication Type*, the integration will now run with the changed Account configuration.<br>Further, while editing or creating an Integration, after you select the Operation, the Account field now lists all the Accounts that are supported for the execution of the selected Operation.<br>While testing the Operation, you can select another Account created with a different *Authentication Type*, if the Account is supported for the execution of the selected Operation. |
| Support for clearing storage locks | If the environment goes down while an integration is running, the lock taken on the integration will not be automatically removed immediately, so any scheduled executions for the same integration will be skipped. You can now remove the locks from the *Monitor > Clear Storage Locks* page. |
| SOAP API enhancements | Integration Cloud now allows you to export and import SOAP APIs. |
| Recipes page enhancements | ■ You can now search a recipe by the recipe name.<br>■ If the main integration created out of a recipe does not have Applications but has sub-integrations, and if the sub-integrations have Applications, then the Applications are now pulled from the sub-integrations. The logos of the Applications in the sub-integrations will now appear on the Recipes page.<br>■ You can now see the details of a recipe when you click *Use*. Further, the recipe configuration page now shows only the Applications and not the Operations for each Application. Also, now you do not have to configure each integration and sub-integration individually. |

| Item | Description |
|---|---|
| | ■ If you have already used a recipe, Integration Cloud now prompts you if an integration with the same name already exists in your selected project and also whether you want to overwrite the references and changes made in the existing integration. |
| New Account Configuration fields | You can now set the following fields while configuring an Account:<br>■ *Block Timeout* is the number of milliseconds that Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error.<br>■ *Expire Timeout* is the number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled.<br>■ *Idle Timeout interval* in milliseconds defines the interval for which a connection will be kept alive if it is not in use.<br>■ *Keep Alive Interval* in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. |
| REST Resource Operation enhancements | You can now modify a REST resource even if the resource has an operation with a deleted integration mapped to the operation. |
| Project permission changes | To create projects, you must now have the *Administrator* Access Profile as well as the *Developer* project permission assigned. |
| New Applications | ■ *webMethods Cloud Container*: You can now use this Application to invoke Java and Flow services for any Cloud Container solution. |
| Support for Keyboard shortcuts for workspace and debug operations | You can now click the *Show Keyboard Shortcuts* icon available above the integration workspace area and view the available shortcut keys.<br><br> |
| JSON Web Token support | For some Applications, for example, Salesforce CRM v44, Integration Cloud will now get an Access Token using the JSON Web Token (JWT) flow after you save the Account. You can generate OAuth 2.0 tokens using either the |

| Item | Description |
|---|---|
| | Authorization Code flow or the JSON Web Token (JWT) flow approaches. |

## Version 6.0.0 (October 2019)

### Integration Cloud

| Item | Description |
|---|---|
| Preview integrations | You can now view the pipeline and mapping details for a previous version of an orchestrated integration by clicking the *Show History* option in the Development stage. Pipeline preview is available in all stages of the Integration. To view the pipeline and mapping details for a previous version of an orchestrated integration in any stage other than the Development stage, change the stage, select the integration from the Integrations page, and then click *Preview*. You will not be able to make any modifications to the existing pipeline and mapping data. |
| Usage Reports | You can now view or download the run counts of integration usages for a specific stage and for the specified time frame. The report shows the count for the currently active stage. The data you see on the Reports page depends on when you view the data and when data collection ends, and it displays data collected till the day before yesterday. |
| Copy Integrations, REST APIs, and SOAP APIs across projects | You can now copy Integrations, SOAP APIs, and REST APIs between projects. Ensure that you create any account or reference data associated with the respective asset in the target project. |
| New services | **Storage**<br>■ deleteStore - Deletes a data store and all its contents.<br>**XML**<br>■ xmlNodeToDocument - Converts an XML node to a document.<br>■ xmlStringToXMLNode - Converts an XML document (represented as a String, byte[ ], or InputStream) to an XML node.<br>■ getXMLNodeType - Returns information about an XML node. |

| Item | Description |
|---|---|
| | ■ queryXMLNode - Queries an XML node. |
| New Applications | ■ Salesforce® CRM REST - Integration Cloud connects to Salesforce® CRM REST using the REST API and allows you to manage security for inbound requests, log payloads, and specify run-time performance conditions for consumers for outbound requests. It also supports multiple authentication mechanisms.<br>■ Google Cloud Pub/Sub - Integration Cloud connects to Google Cloud Pub/Sub and allows you to create, get, delete, set policy, and get policy on topics and subscription resources. |
| Locale information | You can now set the locale information according to the user's preference from the *Users > Locale* tab. The value set by you here is the locale applicable for your user profile, irrespective of the value set by a tenant administrator in the *Default Locale* field in *Company Information*. |
| Support for duplicate keys | While running or debugging Integrations and testing Operations, if the input has any duplicate keys, or if the service returns an output with duplicate keys, you can now view those keys. |
| End to End Monitoring | End to End Monitoring is a cloud offering by Software AG to monitor a business transaction from its start to end, as it passes through the various cloud platforms which includes webMethods API Gateway, webMethods Integration Cloud, and webMethods.io B2B. End to End Monitoring allows you to identify any errors that occur during a business transaction. It identifies the application within the cloud platform where the error has occurred and also provides details of the time at which the error has occurred. |

### Cloud Deployment

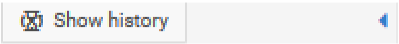
| Item | Description |
|---|---|
| Support upgrade to v10.5 from v10.4 and v10.3 | You can now upgrade runtimes like webMethods Integration Server, Universal Messaging, and Terracotta that are part of a solution from an earlier solution 10.3 or 10.4 version to the latest 10.5 version. |
| Support GraphQL in Cloud Deployment | Cloud Deployment now allows you to deploy GraphQL assets which are developed using Software AG Designer and on-premises webMethods Integration Server. You can |

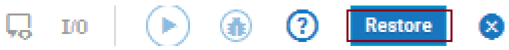
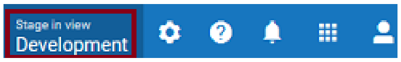
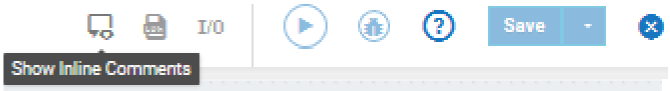
| Item | Description |
|---|---|
|  | choose from predefined solution landscapes to deploy your GraphQL on-premises assets from Software AG Designer. |
| Support for tenant specific image | Hot fixes are now created for specific tenants. The *Hot Fix* list box in the Solution page lists all available fixes that include enhancements to the selected versions. |
| Support for creating solutions in non-development stage | Cloud Deployment now supports creating solutions in any stage, for example, in the development, test, live, and pre-live stage. |
| Support for defining stateless or stateful cluster for a solution in all stages | You can now define that a cluster is stateless or stateful for a solution in all stages. A stateless cluster of webMethods Integration Servers does not use a Terracotta Server Array. Select *Stateful* to add the Terracotta section. The Terracotta icons will be activated. |

## Version 5.6.0 (August 2019)

**Integration Cloud Version 5.6.0**

| Item | Description |
|---|---|
| Projects | A project is an independent entity and corresponds to a folder for organizing your assets. A project holds all the assets created as a part of that project by the logged-in user, along with the configurations associated with the assets. Any asset, for example, Integrations, REST APIs, SOAP APIs, Document Types, and Reference Data, is a part of a project.<br>If you are an existing tenant, your assets will be available in the *Default* project. You cannot delete this default project. If you are a new tenant, the *Default* project is not available and you need to create a new project.<br>**Note:** Only Administrators can create new projects. |
| Project Permissions | Project permissions are used to associate permissions with projects. The new *Project Permissions* page available under the *Settings* > *Project Permissions* tab allows you to associate permissions. Permissions for the *Default* project are assigned on the *Administrative Permissions* page in Access Profiles. |

| Item | Description |
|---|---|
| Assets categorized under Projects | Assets are now categorized under *Projects* on the *Deploy* page. If an asset is pulled and if the associated project is not present in the current stage, the project along with the asset will be available in the current stage. |
| Administrative permissions regrouped | Administrative permissions under Access Profiles have been regrouped under *Global Permissions*, *Functional Controls*, and *Project Permissions for Default Project*.<br>If you are an existing tenant, your user's *Access Profile* controls global permissions as well as permissions for the *Default* project. If you are a new tenant, the *Default* project is not available and your user's *Access Profile* controls only global permissions. |
| Multi Authentication Support | You can now select different authentication types for the same Application while creating an Account. Currently, you can select different authentication types, for example, *Credentials*, *OAuth V2.0 (Authorization Code Flow)*, or *OAuth V2.0 (JWT Flow)* only for Salesforce CRM v44. |
| Flat File enhancements | Integration Cloud now supports Fixed length and Variable length parsers when you use a *sample file* to define the definition and structure of a Flat File Application. |
| Lock and unlock Integrations | Integration Cloud allows you to manage an Integration during the development life cycle by auto locking. When you edit an Integration, it is automatically locked for you. This restricts multiple users editing that Integration at the same time. To unlock an Integration, from the Integrations page, click the Integration link. The Integration Overview page appears. From the Integration Overview page, click *Unlock*.<br>**Note:**<br>Only the user who locked the Integration or an Administrator can unlock the Integration. |
| Create document type from an XML Schema Definition | You can now create and delete a document type built from an XML Schema Definition (XSD). |
| Integration Versioning | While editing an Integration, you can now view the version change history of the Integration and also restore an earlier or previous version of the Integration. Click the *Show history* option available on the tool bar to view the version change history of the Integration.<br> |

| Item | Description |
|---|---|
| | Click on an earlier version to view that Integration version and click the *Restore* option to restore the selected previous version of the Integration.<br><br> <br>If you have reverted to an earlier version and there is a scheduled execution for the Integration, the reverted version of the Integration will be run as per the defined schedule. |
| User Interface enhancements | The following user interface enhancements and changes are made in this release:<br>- When you log in to Integration Cloud, you will now reach the *Projects* page instead of the home page.<br>- The new *Notifications* icon on the Integration Cloud navigation bar allows you to view update notifications, when was the last login, and the list of enhancements and changes in the current release.<br><br> <br>- The Orchestrated Integration workspace has been enhanced to improve the usability. Icons on the Orchestrated Integration workspace navigation bar have been replaced with intuitive icons and the new *Show inline comments* option allows you to view comments entered for the blocks.<br><br> |
| Database Application enhancements | *SSL account in the Database Application*<br>You can now create SSL account to the databases. To create an SSL account, choose the respective certificate from the *Truststore Alias* option available in the *Account* screen.<br>*Pre-loaded driver support*<br>You can now use driver groups like Microsoft JDBC Driver for Microsoft SQL Server and PostgreSQL JDBC Driver for PostgreSQL to create the connection. |
| Custom *From Email Address* for alert emails | Alert emails that come from Integration Cloud have the *From Email Address* as *noreply@webmethodscloud.** by default. |

| Item | Description |
|---|---|
| | You can now request for a custom *From email Address* for your tenant. |
| Custom Domain Name | You can now request for a custom domain name for your tenant. For example, if your company name is XYZ, you can have the domain name as https://subdomain.xyz.com. |
| Support for form encoded parameter | You can now send simple key value parameters embedded in the Request Body for POST or PUT requests. This uses the default web form encoding, which is application/x-www-form-urlencoded. |
| Server Name Indication (SNI) Support | While configuring an Account for an Application, you can select the *Enable SNI* option if a SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SaaS provider to send the host name specified in the Server URL field, as part of the TLS SNI Extension server_name parameter.<br>To explicitly specify a host name to be included as a part of the SNI extension server_name parameter, in case the host name is other than the host name specified in the Server URL field, specify the host name value in the *SNI Server Name* field. |
| Modify alert frequency period | You can now set the alert frequency period from the *Alert Rules* page to send email messages. |
| New Applications | The following new Applications are available in this release:<br>■ Simple Mail Transfer Protocol (SMTP) Application: Integration Cloud allows you to connect to an SMTP server using the Simple Mail Transfer Protocol (SMTP) Application. The SendEmail predefined operation sends an email to the specified recipient using SMTP.<br>■ Shopify: Integration Cloud connects to Shopify using the Shopify REST API and allows you to organize your products, customize your storefront, accept credit card payments, track, and respond to orders.<br>■ CloudStreams Connector for Microsoft Azure Data Lake Store: Integration Cloud connects to CloudStreams Connector for Microsoft Azure Data Lake Store using the REST API and allows you to manage File System resources through the Hadoop Distributed File System (HDFS) API. You create |

| Item | Description |
|---|---|
| | directories, folders, and files in your Azure Data Lake Store instance that can store and retrieve data.<br>■ Zuora REST: Integration Cloud connects to Zuora REST using the Zuora REST API. Zuora allows you to manage Zuora objects in the Zuora Business Object Model, process revenue schedules, and perform other financial operations.<br>■ Salesforce® CRM REST: Integration Cloud connects to Salesforce® CRM REST using the REST API and allows you to manage security for inbound requests, log payloads and specifies run-time performance conditions for consumers for outbound requests. It also supports multiple authentication mechanisms.<br>■ webMethods.io B2B:<br>The webMethods.io B2B application allows you to:<br>■ Interact, accept requests, and build integrations for the webMethods.io B2B product instance.<br>■ Exchange business documents between trading partners.<br>■ Electronic Data Interchange (EDI):<br>This application provides predefined operations to parse, validate, and transform EDI messages received from the webMethods.io B2B application and use these transformed messages to create orchestrations. |

### Cloud Deployment Version 5.6.0

| Item | Description |
|---|---|
| Cloud Deployment Command Line Interface (CLI) | You can now manage a solution, monitor the status of all runtimes in a solution, promote assets from one stage to another, and so on using the Cloud Deployment CLI. The CLI supports *Interactive* and *Normal* modes. |
| Database as a Service | You can now add a MySQL database to your cloud deployment subscription. This enables you to configure, store, and monitor your database directly in the cloud instead of using external systems. The database endpoint can be shared by multiple solutions deployed by the tenant. |

| Item | Description |
|---|---|
| Self-managed update management | You can now update any product in a solution to the available higher version after you create the solution. The *Update Available* option appears if a higher version is available for any of the products in the solution. The latest version will appear in the *Version* drop-down list. |
| Copying solutions | You can now copy solutions in any stage. Copying solutions allows you to have a back up of your solution before you make any changes. You can choose to make a copy of a solution by either using the same configuration and services in the solution landscape or by modifying the configuration and services in the solution landscape. |
| End to end monitoring using AppDynamics | If you are currently using AppDynamics to trace end to end business flows, the new tracing list box option available on the Solution creation page allows you to trace logs after creating or updating a solution for a webMethods Integration Server runtime. |
| Viewing API signature of executable services | After deploying assets, you can go to the Asset explorer page and click the *API Details* option to view the API details of the service such as the HTTP Method, URL, Input structure, and the parameters that are required to invoke the service from an external system, for example, a REST client.<br>**Note:** The *API Details* option appears only for executable services. |
| Monitoring support for JDBC adapter and CloudStreams connectors | You can now view *Connectivity KPIs* for both JDBC adapter and CloudStreams connectors in the runtimes of solutions. For connectors, listeners and connections data are displayed. |
| webMethods Integration Server service execution statistics | webMethods Integration Server service execution for solutions is now displayed by default for all services. There is no need to enable audit logging. |

## Version 5.5.0 (April 2019)

This section describes the enhancements and changes made in Version 5.5.0 for **Integration Cloud**:

| Item | Description |
|---|---|
| Validate Input and Output | You can now select the **Validate input** and **Validate output** options to specify whether you want Integration Cloud to validate the input and output to the Integration, against the service input or output signature. |

| Item | Description |
|---|---|
| Streaming support and replaying Salesforce events | Some Integration Cloud Applications, for example, *Salesforce CRM version 44*, now support connectivity with streaming APIs and processing of streaming API events.<br>You can create a *Salesforce CRM* listener, select a subscription channel, and specify the Integration to be invoked on the incoming events. Additionally, you can configure the headers and parameters as well as enable and disable the listener. Once enabled, the listener receives the streaming API events and processes the received events.<br>The *Salesforce CRM* listener can subscribe and listen to Salesforce events. Salesforce stores standard-volume events for 24 hours, so for versions of Salesforce later than v37.0, you can retrieve the events if they are within the retention window. You can replay the lost events by selecting the following replay options:<br>■ *New* - Receive only new events that are broadcast after subscription.<br>■ *All* - Receive new events including past events (last 24 hrs) that are within the retention window.<br>The Salesforce CRM listener can now subscribe and listen to all event types, for example, Salesforce Push Topic Event, Salesforce Platform Event, Salesforce Change Data Capture Event, and Salesforce Generic Event. |
| Support for consuming and producing Flat Files | You can now create a Flat File Application by defining a flat file structure either manually or from a sample file. You can then convert an inbound flat file to a document by invoking the predefined *convertFlatFileToDocument* operation in an Integration, or convert a document to an outbound flat file by invoking the predefined *convertDocumentToFlatFile* operation. |
| Support for creating SOAP APIs | Integration Cloud allows you to write integration logic to integrate different types of applications. This logic can now be exposed to the external world using SOAP APIs. You can create SOAP APIs by using an existing set of Integrations (from scratch) or by using a WSDL file.<br>A SOAP API exposes one or more Integrations as *operations*, so each operation in a SOAP API corresponds to an Integration.<br>Using a SOAP client, you can invoke the SOAP operation *externally* by using either Basic Authentication or 2-way SSL. |

| Item | Description |
|---|---|
| | When the SOAP operation is invoked, the associated Integration gets executed. |
| Connect to a database using the Database Application | You can now connect to a database using the new *Database* Application and perform database operations with cloud databases. |

This section describes the enhancements and changes made in Version 5.5.0 for **Cloud Deployment**:

| Item | Description |
|---|---|
| Upgrade products to a higher fix version in a solution | You can now upgrade any product in a solution to the available higher fix version after you create the solution. The *Upgrade* option appears if a higher fix version is available for any of the products in the solution. The latest fix version will appear in the *Fix* version drop-down list. |
| Enable packages during solution creation | While creating a solution, after you select a fix version, webMethods Integration Server packages such as WmCloudStreams and WmJDBCAdapter will appear in the *Packages* group box based on the selected fix version. You can select the packages that you want to enable. |
| Enable cloud deployment capability for all tenants | Cloud Deployment capability is now enabled by default for all tenants. As soon as you register, 3 CPU cores and 6 GB memory are allocated for all tenants. *Provisioning* happens if you access Cloud Deployment for the first time using the application launcher. Solutions created using a trial account are deactivated daily. After you log in, you need to reactivate the solutions. All assets will be available after a short delay. |
| Deploy webMethods CloudStreams assets | You can now deploy CloudStreams provider packages, CloudStreams connector services, CloudStreams connection, and CloudStreams connector listeners to a solution in Cloud Deployment and view those assets. This is applicable only if you have selected *WmCloudStreams* as the package option while creating the solution. |
| Promote assets from a solution to another solution | Within a tenant, you can now promote assets from a solution to another solution, from a previous stage to the current stage, for the same runtime type. You can promote assets if the source runtime version is lesser than or same as the target runtime version. |
| Load pipeline data for testing services | When you run a service in Software AG Designer, you can save the pipeline data as an XML document to your local file system. After you deploy the service in Cloud Deployment, you can now click the *Load Data* option in the |

| Item | Description |
|---|---|
| | service editor in Cloud Deployment to load or update the pipeline data and test the service. |
| Download user deployed packages and configurations | You can now download user deployed packages and configurations from the *Assets* page. The assets will be zipped and downloaded to your local storage space.<br>From the *Asset Repository* page, you can either download individual packages or download the whole repository for each product. The assets including ACDL files will be zipped and downloaded to your local storage. |
| Enhanced user interface for Monitoring | ■ The *Alerts* page now displays the *Resolved On* date for all the resolved alerts.<br>■ On the *Runtimes* page, you can now click on the *Adapter KPI* link to display the Adapters details in a pop-up window.<br>■ On the landing page, a help icon is now added in the *Service Executions* card.<br>■ From the *Alerts* card on the landing page, you can now click *Configure* and go to the *Alerts Configuration* page. |

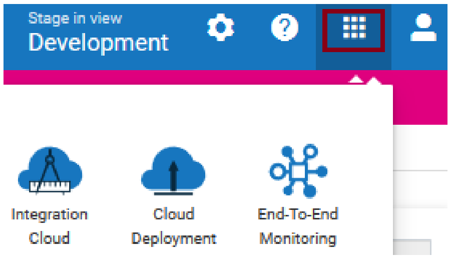
# 2 Create and secure your account

# Registration

**Note:**
The registration page is not applicable if you have created your account using the Software AG Cloud sign-up page.

**Note:**
If you have created your account using the Software AG Cloud sign-up page, you will receive an email which contains the link to log in. After you log in, you can go to the Software AG Cloud portal using the Application Launcher.

The **Cloud Deployment** option is available only if you are an existing tenant and have already provisioned Cloud Deployment.



**Registration** is the process of creating a new Integration Cloud user account. You need to register to create your instance of the platform in the cloud.

Your organization may have multiple members, for example, your organization may be an entire company, an internal department, or just yourself. Similarly, your Integration Cloud account can have multiple internal users who interact with the platform. The very first person to open the Integration Cloud account becomes the first System Administrator for the tenant. The Administrator can then create new users (internal users).

# Creating an Account

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

Creating an account is the first step in the Registration Process.

### To create a new User Account

1. On the **Registration** page, complete the following fields:

   **Note:**
   Required fields are marked with an asterisk on the screen.

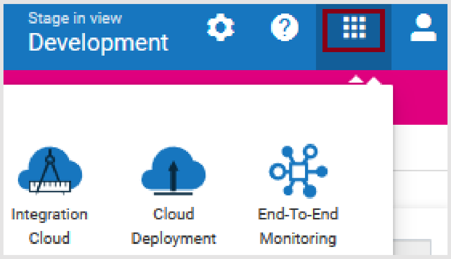
| Field | Description |
|---|---|
| **First Name** | Type your first name.<br>You can change the value after the user is created from the **Settings** ⚙ **> Users** screen. |
| **Last Name** | Type your last name.<br>You can change the value after the user is created from the **Settings** ⚙ **> Users** screen. |
| **Company** | Type your company name.<br>You can change that later from the **Settings** ⚙ **> Company Information** screen. |
| **Country** | Select your country from the drop-down list box.<br>You can change that later from the **Settings** ⚙ **> Users** screen. |
| **State or Province** | Type your State or Province.<br>You can change that later from the **Settings** ⚙ **> Users** screen. |
| **Phone** | Type your phone number.<br>You can change that later from the **Settings** ⚙ **> Users** screen. |
| **Your Role** | Select your role in your current organization from the drop-down list box. |
| **Company Size** | Select the number of employees range in your current organization from the drop-down list box. |
| **Is your interest based on** | Select at what stage is your current project from the drop-down list box. |
| **Sub-Domain** | Provide a unique sub-domain, typically your company name.<br>For example, suppose you are at ABC Company and you decide to use “abc” as your unique sub-domain. With that setting, you will access your instance of the platform at https://abc.webmethodscloud.com.<br>**Note:**<br>You must log in with the correct sub-domain. Some functionalities may not work properly if you log in with a generic sub-domain. |
| **Work Email Address** | Type your work email address.<br>The email field becomes both the user name and the email address for the initial user. You can change the values after the user is created, from the **Settings** ⚙ **> Users** screen. |

| Field | Description |
|---|---|
| **Type the characters shown in the image** | Type the twisted alphanumeric characters in the text input area as shown in the image. You can click the refresh icon to view a new set of characters. |
| **I agree to the Terms of Service** | Select this option to agree to the webMethods Integration Cloud **Terms of Service**. |
| **I confirm that I have acknowledged and accepted the terms of the Data Processing Agreement. I further confirm that I have downloaded and validly countersigned the Data Processing Agreement.** | Select this option to acknowledge and accept the terms and conditions of processing of personally identifiable information as per the General Data Protection Regulation (GDPR). GDPR sets guidelines on how to collect and process personally identifiable information. You must also download and countersign the agreement. |
| **Promo Code** | Enter a valid promotion code if you have one, for availing subscription benefits. |
| **I opt-in to hearing from Software AG** | Select this option if you want to receive information about products, services, and events from Software AG. Software AG may also contact you by phone and may process your personally identifiable information for these purposes. You can unsubscribe and withdraw your consent at any time. |

2. Click **Register** to continue to the next step to activate and secure your account. After you click **Register** and as soon as the registration process is complete, two different emails will be sent to the email address you provided during registration. One email will contain the user ID and the other email will contain the temporary password. Use the temporary password to log in. You will be asked to change your password.

   **Note:**
   Your organization is a tenant in the platform. When you log in to the platform, you log into your organization's tenancy.

   **Note:**
   If you have created your account using the Software AG Cloud sign-up page, you will receive an email which contains the link to log in. After you log in, you can go to the Software AG Cloud portal using the Application Launcher.



**Note:**
If you are not able to login successfully after a few login attempts, Integration Cloud displays twisted alphanumeric characters in the login page. Type the twisted alphanumeric characters that appear in the text box. You can click the refresh icon to view a new set of characters.

If you have already configured SAML based single sign-on (SSO), the **SSO Login** option appears in the login page. If you click the **SSO Login** option, Integration Cloud redirects you to the Identity Provider (IdP) login page. After you provide the IdP login credentials, you will be logged into Integration Cloud.

## Securing your Account

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

Securing your account is the second step in the Registration process. When you login for the first time, you are asked to change your password and also select a security question. The security question and answer is associated with your user name. If you forget your password, this information is used to verify the account ownership.

### To secure your account

1. Type your new password, and then select a security question from the drop down list. Optionally, you can select the option **Write my own security question** to compose a personalized security question.

2. Provide an answer to the security question.

3. Click **Submit**.

**Note:**
If you forget your password, in the login page, click the **Forgot Password?** link, enter your user name, type the distorted alphanumeric characters in the text box, and then click **Change Password**. An email is sent that contains a request to answer the **Security Question** you chose when your account was created. When the email arrives, click the link to open the **Password Reset** page. Provide the answer to your Security Question and enter a new password. After you provide the correct answer, you can log in with your changed password.

# 3 Settings

# Users

You can use the **Users** screen to create and manage administrators and other users. A User has a login identity, password, email address, and other descriptive attributes.

From the main **Users** screen, you can search for users, create a new user, delete an existing user, update existing user information, and reset a user's password. If you have the **User Management** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls**, you can either edit or delete users.

> **Note:**
> You cannot delete your own user profile. If a user is deleted, then the user cannot be recovered and all assets created or modified by the user will appear in the *Created By* and *Modified By* columns as *Unknown User{first two characters of the first name and last name}*.

Click **Reset Password** to reset the user's password. As soon as the password is reset, two different emails will be sent to the email address you provided during registration. One email will contain the user ID and the other email will contain the temporary password. Use the temporary password to log in. You will be asked to change your password.

Users who have the required access privileges under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can edit user information.

## Adding Users

If you have created your account using the Software AG Cloud sign-up page, that is, if you are a Software AG Cloud tenant, you can perform certain user management tasks like adding users, updating users, and resetting passwords only from the Software AG Cloud User Administration page. Further, a new user is created in Integration Cloud when you log in for the first time using the Software AG Cloud login page. The newly created user is associated with the **Regular User** Access Profile if you have selected Integration Cloud-User in the Software AG Cloud User Administration page, or the **Administrator** Access profile if you have selected Cloud-Tenant-Administrator in the Software AG Cloud User Administration page.

You can delete users from the **Users** page in Integration Cloud. If you have created Users U1, U2, and U3 in Software AG Cloud, the first time U1 logs in to Integration Cloud, user U1 will be created in Integration Cloud. Now if U1 is deleted from Software AG Cloud but still exists in Integration Cloud, U1 will not be able to log in to Integration Cloud. If U1 is deleted from Integration Cloud but still exists in Software AG Cloud, U1 will be created in Integration Cloud when you again log in to Integration Cloud.

If you have not created your account using the Software AG Cloud sign-up page, you can add users in Integration Cloud.

**To add a user if you have not created your account using the Software AG Cloud sign-up page**

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> Users**.

2. From the upper right part of the Users screen, click **Add New User**.

3. On the **Basic** tab, complete the following fields. Required fields are marked with an asterisk on the screen.

**Field** — **Description**

**First name**

User's first name as it should appear in the platform.

**Last name**

User's last name as it should appear in the platform.

**Title**

User's professional title.

**Access Profile**

The access profile assigned to the User. Each User is assigned an access profile, which can be shared by other users. An Access Profile specifies the network locations (IP addresses) from where it is possible to login and administrative permissions. Select one of the following Access Profiles:

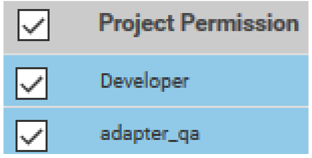
- **Administrator** - Provides permissions needed by the System Administrator.
- **Regular User** - Provides permissions that are more appropriate for normal users.

By default, the system administrator can change the Administrative Permissions associated with each Access Profile (except the above mentioned **Administrator** Access Profile), and can add additional Access Profiles, as needed.

> **Note:**
> By default, the **Administrator** and **Regular User** Access Profiles are associated with the Development Stage. If you have created a new Access Profile, ensure that the Access Profile you have created is associated with the Development Stage. See "Adding or Updating Access Profiles" on page 52 for more information and for information on API Management Access Profiles and permissions.

**Project Permission**

Project permissions are used to associate permissions with projects.



☑ Project Permission
☑ Developer
☑ adapter_qa

Any new project created is automatically associated with the *Developer* project permission profile. If a project permission profile is associated with a user on the user profile page, the

| Field | Description |
|---|---|
| | user can perform only the permitted tasks in the mapped project. If you are an existing tenant, all existing user profiles will be associated with the *Developer* project permission profile. If you are a new tenant, the Administrator will be associated with the *Developer* project permission profile and can assign a project permission profile to a new user. You can add more project permission profiles to the user. See "Project Permissions" on page 59 for more information.<br>**Note:** A user must have the **Administrator** Access Profile and the **Developer** project permission assigned to create projects. |
| **Employee Number** | Optional identification number for each employee. |
| **Email** | Email address of the user. User credentials will be sent to the specified email address.<br>As soon as you add a new user, two different emails will be sent to the email address. One email will contain the user ID and the other email will contain the temporary password. Use the temporary password to log in. You will be asked to change your password. |
| **User Name** | User name is a unique name associated with each user and is required to log in. It can be an email address or an alphanumeric text string.<br>**Note:** If you are a Software AG Cloud user, you will not be able to update the **User Name**. |
| **Federation ID** | Enter the **Federation ID** if your Identity Provider passes the Federation ID for **Single Sign-On**. See the "Single Sign-On" Help page for more information. The Federation ID acts as a user's authentication across multiple IT systems or organizations. A federated identity means linking a person's electronic identity and attributes stored across multiple distinct identity management systems. |
| **Partner** | Select this option if the user is a Partner user. If **Allow User Interface Access** permission available under **Access Profile > Administrative Permissions > Account Controls** is not enabled, a Partner User can still perform on-premises tasks. |
| **Active** | Select this option to indicate that the user account is active. You can use this option to reactivate a locked or disabled user account. |

4. On the **Locale** tab, complete the following fields:

| Field | Description |
|---|---|
| **Time Zone** | Choose a **Time Zone Code** from the drop down list. |
| **Date Format** | Choose a Date Format from the drop down list. “mm” is "Month", “dd” is "Day", and “yyyy” is Year.<br>Dates and Times are used throughout the platform, in Appointments, as Start/End Dates in Tasks, Expected Close Date, Estimated Start/End Date, Date Due, and so on. Default formats are specified under the **Settings** ⚙ **> Company Information > Advanced Information** tab. Administrators and Users can change the default selection in the **Users** screen. |
| **Locale** | This setting determines the language in which you will view the application.<br>The value set by you here is the language applicable for your user profile, irrespective of the value set by the administrator in the **Default Locale** field of **Company Information** settings.<br>For example, if you set the value in the **Locale** field as `Chinese` and the value set by the administrator in the **Default Locale** field of **Company Information** settings is `English`, then you will view all the application labels in the Chinese language. |
| **Time Format** | Select a 12-hour clock (hh:mm a) with AM/PM, or a 24-hour clock (HH:mm). |

5. On the **Address and Contact** tab, complete the following fields:

| Field | Description |
|---|---|
| **Phone** | Primary phone number for the user. |
| **Mobile Phone** | Mobile phone number for the user. |
| **Fax** | Fax number for the user. |
| **Street Address** | Street address for the user. |
| **City** | City for the user. |
| **State/Province** | State or province for the user. |
| **Postal/Zip Code** | Postal or ZIP Code for the user. |
| **Country** | Country for the user. |

6. Click **Add** if you are adding a User or **Apply** if you are editing any User information.

   You can fill the **Address and Contact** section later or the Administrator can fill the details by editing the record after the User has been added. The **Address and Contact** screen is also available under 👤 **> My Profile > My Information** tab.

   > **Note:**
   > A User can log in, and then go to 👤 **> My Profile > Edit** to change the user details. The Administrator who created the User can also edit the User details.

## Updating Users

### To edit or update the user information

> **Note:**
> If you have created your account using the Software AG Cloud sign-up page, that is, if you are a Software AG Cloud tenant, you can perform certain user management tasks like adding users, updating users, and resetting passwords only from the Software AG Cloud User Administration page.

1. From the Integration Cloud navigation bar, click **Settings** ⚙ **> Users**.

2. Select a user from the list, and then click **Edit**.

3. Make necessary modifications. See "Adding Users" on page 42 for information on the relevant fields. You can also enter or update the following information on the **Address and Contact** tab. Required fields are marked with an asterisk on the screen.

   > **Note:**
   > If you are a Software AG Cloud user, you will not be able to update the **User Name**.

| Field | Description |
|---|---|
| **Phone** | Primary phone number for the user. |
| **Mobile Phone** | Mobile phone number for the user. |
| **Fax** | Fax number for the user. |
| **Street Address** | Street address for the user. |
| **City** | City for the user. |
| **State/Province** | State or province for the user. |
| **Postal/Zip Code** | Postal or ZIP Code for the user. |
| **Country** | Country for the user. |

4. Click **Apply**.

   The default initial information comes from the ⚙ > **Company Information** page, but you can modify it here.

   **Note:**
   A user can log in and then go to 👤 > **My Profile** to change the user details. The administrator who created the user can also edit the user details.

   **Note:**
   If you have the **User Management** permission under **Settings** ⚙ > **Access Profiles** > **Administrative Permissions** > **User and Ownership Controls**, you can either update or delete users. You cannot delete your own user profile. If a user is deleted, then the user cannot be recovered and all assets created or modified by the user will appear in the *Created By* and *Modified By* columns as *Unknown User{first two characters of the first name and last name}*.

## Resetting Passwords

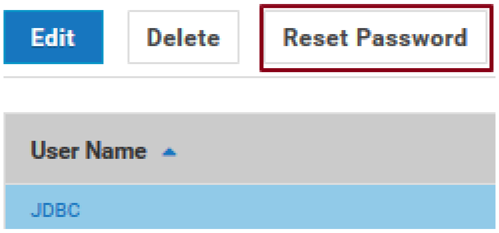
**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

**Note:**
If you have created your account using the Software AG Cloud sign-up page, that is, if you are a Software AG Cloud tenant, you can perform certain user management tasks like adding users, updating users, and resetting passwords only from the Software AG Cloud User Administration page.

**➤ To reset a user's password if you have not created your account using the Software AG Cloud sign-up page**

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ > **Users**.

2. For the user whose password is to be reset, select the user and click **Reset Password**.



   Integration Cloud sends two different emails to the email address you provided during registration. One email will contain the user ID and the other email will contain the temporary password. Use the temporary password to log in. You will be asked to change your password.

> **Note:**
> A User can log in, and then go to 👤 > **My Profile** to change the user details. The administrator who created the User can also edit the User details.

## User Profile

If you are on the **My Information** page 👤 > **My Profile** > **My Information**, the page provides profile information for the logged in user for the Integration Cloud instance.

If you are on any user profile page, (**Settings** ⚙ > **Users** > **Click on the User Name link**), the page provides profile information for the selected user for the Integration Cloud instance.

You can view the **Basic**, **Locale**, and the **Address and Contact** information.

Click **Edit** to update the information.

### Security Question

> **Note:**
> This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

**To update the Security Question and Answer**

1. From the Integration Cloud navigation bar, go to 👤 > **My Profile** > **Security Question**.

2. Select a **Security Question** and type a **Security Answer**. You can change the **Security Question** associated with your Account Login/Password.

3. Click **Submit**.

   > **Note:**
   > The User name and Email address can differ, depending on the settings specified in the 👤 > **My Profile** > **My Information** page.

### Change Password

> **Note:**
> This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

**To change your password**

1. From the Integration Cloud navigation bar, go to 👤 > **My Profile** > **Change Password**.

2. Type your current password in the **Old Password** field, your new password in the **New Password** field, and again retype your new password in the **Retype New Password** field.

3. Click **Submit**. You will receive a confirmation email about your changed password.

### My Certificate

Integration Cloud allows you to store client certificates and associate a certificate with a user account. When a client presents one of these certificates, Integration Cloud logs in the client, as the user *mapped* to the certificate. You can view the client certificate for the logged in user on the **My Certificate** page.

**To view the certificate**

1. From the Integration Cloud navigation bar, click 👤 **> My Profile > My Certificate**.

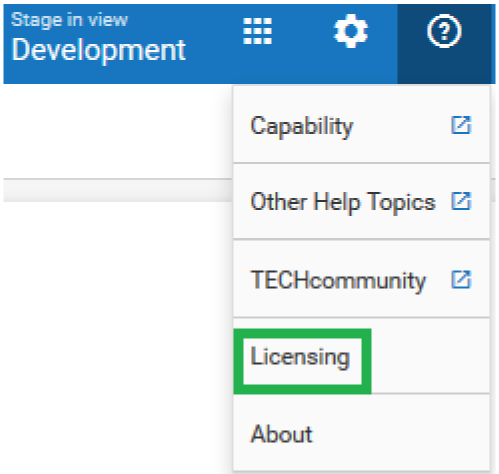
2. If a certificate is configured for the user, the **View Certificate** panel displays the configured certificate. You can click **Download** to download the user certificate or click **Delete** to delete the user certificate. The downloaded file is named as `<username>`.crt.

3. In the **Upload New Certificate** field, click **Browse** to upload a new client certificate signed by a trusted Certificate Authority (CA).

4. In the **Generate Private Key and Certificate** field, click **Generate** if you want Integration Cloud to generate a private key and a new Integration Cloud signed client certificate. Integration Cloud validates it against the issuer of the certificate. The generated certificate is named as `<username>`.txt.

## Capability

The **Capability** (? > Licensing) page allows you to view the status of some of the system capabilities, based on your license offering.



You can view the details of the following capabilities in **Integration Cloud**:

| Field | Description |
|---|---|
| **Allowed application count** | Total number of Applications that can be utilized by the tenant. |
| **On-premises connection** | If **Yes**, then on-premises applications can be uploaded from on-premises systems. |
| **Max allowed users** | Maximum number of active users allowed for the tenant. |
| **Allowed number of stages** | Maximum number of staging environments allowed for the tenant. |
| **Integration restart and resume** | Integrations can be restarted and resumed. |
| **Integration import and export** | Integrations can be imported and exported. |
| **Trial account** | If **Yes**, then the account is a trial account. |
| **Trial end date** | The trial period end date. This field appears only if the account is a trial account. |

You can view the details of the following capabilities in Cloud Deployment:

| Field | Description |
|---|---|
| **Max allowed cores** | Maximum number of CPU cores allowed across all active solutions and all stages for the tenant. You will not be able to create additional solutions if you exceed this capability. |
| **Max allowed memory** | Maximum memory capacity allowed across all active solutions and all stages for the tenant. You will not be able to create additional solutions if you exceed this capability. |
| **Allowed number of stages** | Maximum number of staging environments allowed for the tenant. |
| **Trial account** | If **Yes**, then the account is a trial account. |
| **Trial end date** | The trial period end date. This field appears only if the account is a trial account. |

## Access Profiles

An **Access Profile** specifies a collection of permissions that can be applied to multiple users. Each user is assigned an Access Profile, which can be shared by other users.

Users who have the required access privileges under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can edit the Access Profiles information.

An Access Profile specifies:

- The network locations (IP addresses) from where it is possible to login.
- Administrative permissions.
- Container user groups
- API Management permissions

See this article for information on Access Profiles, Project Permissions, and ACLs.

The default Access Profiles are:

- Administrator, which provides permissions needed by the System Administrator.
- Regular User, which provides permissions that are more appropriate for normal users.

  > **Note:**
  > The Integration Cloud User role in Software AG Cloud maps to the Regular User access profile in Integration Cloud. Users assigned to the Integration Cloud User role have limited permissions that are more appropriate for normal users.

- API Gateway Administrators - By default, all API Management permissions are assigned to the Administrators access profile and these privileges cannot be modified.
- API Gateway Providers - By default, the following permissions are assigned to the API Gateway Providers access profile and these privileges cannot be modified:
  - Manage APIs
  - Manage Applications
  - Manage policy templates
  - Manage packages and plans
  - Publish to API Portal
  - Export assets
  - Execute service result cache APIs
  - Activate/Deactivate APIs/Packages
  - Manage aliases
  - Import assets
- API Portal Administrators - The API Portal Administrator can perform all the functions in API Portal.

- API Portal Providers - The API Portal Provider can manage APIs and packages in API Portal.

**Note:**
You can create and manage API Management Access Profiles provided you have the required API Gateway Cloud and/or API Portal Cloud licenses.

By default, the system administrator can change the **Administrative Permissions** associated with each Access Profile and can add additional Access Profiles, as needed.

To edit an existing Access Profile, select the profile and click **Edit**. To delete an Access Profile, select the profile and click **Delete**. You will not be able to delete an Access Profile if it is used by a user. To create a new Access Profile, click **Add New Access Profile**.

**Note:**
The Access Profile ID is needed while configuring Single Sign-On (SSO). You have to provide the ID while configuring the Identity Provider (IDP), if you want to create a user if the user is not present. The newly created user will be associated with the Access Profile represented by the ID sent by the IDP in the SAML Response. The name of the SAML attribute that designates the user's access profile must contain the ID of the Access Profile.

## Adding or Updating Access Profiles

Use the **Access Profiles** screen to create or edit profiles assigned to users.

### To add or update an Access Profile

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> Access Profiles**.

2. Click **Add New Access Profile** to add a custom access profile or click **Edit** to modify an existing Access Profile.

3. On the **Add New Access Profile** or **Update Access Profile > Access Profile Information** tab, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Name** | Provide a name for the Access Profile. You can reference the profile by name when assigning it to a user. |
| **Description** | Provide a general description for the Access Profile. |

4. On the **Login IP Address Restrictions** page, complete the following fields:

| Field | Description |
|---|---|
| **IP Address Ranges** | For extra security, enter ranges of IP addresses from which users are allowed to access the platform. If a user attempts to login from a |

| Field | Description |
|---|---|
| | computer on a network outside of the specified range, access to the platform is denied.<br>**Note:**<br>A maximum of 25 IP address ranges can be specified. You can add, modify, and delete the entries. Accepted format is xxx.xxx.xxx.xxx - yyy.yyy.yyy.yyy, where xxx and yyy are numbers in the range 0-255 and xxx.xxx.xxx.xxx is less than or equal to yyy.yyy.yyy.yyy. To specify a single IP address, use the same IP address for the start and endpoint of the range: 192.168.1.1 - 192.168.1.1<br>When a user attempts to log in, the IP address of the system the request originated from is checked against the configured settings. If the address is in the allowed range, the user can continue the login process. Otherwise, login is denied. Access violations are recorded in the audit log, identifying both the user and the IP address from where the login attempt originated. Login restrictions do not apply to Customer Support logins. |

5. On the **Administrative Permissions** page, select the operations a user can perform in order to access, view, create, update, upgrade, administer, execute, export, deploy, and delete and to allow the user to customize selected aspects of the platform.

| Field | Description |
|---|---|
| **Global Permissions** | |
| **User and Ownership Controls** | **User Management** - Select this option if you want to add, update, delete users, or assign users to Access Profiles.<br>**Access Control** - Select this option if you want to allow a user to modify Access Profiles, edit ACLs, specify user application access rights, manage Access Profiles, specify the password policy, create, edit, and delete OAuth 2.0 clients and scopes, and delete OAuth 2.0 tokens.<br>**Manage Personal Setup** - Select this option if you want to allow a user to modify the personal information, and generate or edit the user's own certificate. |
| **Account Controls** | **Manage Company Capabilities** - Select this option if you want to allow users to modify the company information.<br>**Allow User Interface Access** - Select this option if you want to allow users to log in to Integration Cloud and access the user interface. Clear this option if you want to deny users to access the user interface. Further, even if you clear this option, all users can still interact with Integration Cloud using REST interface calls. |

| Field | Description |
|---|---|
| | **Note:**<br>If the **Allow User Interface Access** permission is not enabled for a user but if the user is a Partner user, that user will still be able to perform on-premises tasks. |
| **Data Management Controls** | **Manage Audit Log** - Select this option if you want to allow users to view the Audit Log. If this option is enabled, the Audit Log page will be displayed. If not selected, the user will not be able to view the Audit log page. To view the **Audit Log** screen, from the Integration Cloud navigation bar, click **Settings** ⚙ **> Audit Log**. |
| **Functional Controls** | |
| | Select the required options under **Assets**, **Stages**, **Advanced Security**, and **Application**. You must select the required permissions to deploy, export, administer, upgrade, create, update, and delete those functions. |
| **Project Permissions for Default Project** | |
| | Here you will manage the permissions for new and existing assets only inside the **Default** project. See "Project Permissions" on page 59 for information. Select the required options under **Accounts**, **Operations**, **Reference Data**, **Document Type**, **Integrations**, **REST APIs**, **SOAP APIs**, and **Listeners**.<br>**Note:**<br>If you are a new tenant, the *Default* project is not available, so this section is not applicable. Your user's **Access Profile** controls only global permissions. |

6. On the **Container** page, enter the names of the webMethods Integration Server Access Control List (ACL) groups separated by a comma, for example, Administrators, Developers, and so on. Users who are assigned to this Access Profile will be now part of the webMethods Integration Server container user group (s) and can perform tasks allowed for those user groups. If you do not map an Access Profile to an webMethods Integration Server group, you will not be able to invoke webMethods Integration Server services. For information about user groups, see the Managing Users and Groups section in the *webMethods Integration Server Administrator's Guide*.

   **Note:**
   The **Container** tab and Container related Administrative permissions are available only if you have the required license for Containers.

   **Note:**
   Integration Cloud Administrator profiles are *not automatically* assigned to the webMethods Integration Server Administrators ACL group. If you do not enter any user groups in the **Container User Groups** field, but have configured webMethods Integration Server in a way

such that it needs to verify the ACL groups you have entered in the **Container User Groups** field while invoking services, you will not be able to run or invoke webMethods Integration Server services from Integration Cloud.

7. The **API Management** tab displays the API management permissions.

**Note:**
Integration Cloud provides the user management capability for API Gateway. You can create and manage API Management Access Profiles provided you have the required API Gateway Cloud and/or API Portal Cloud licenses.

| Field | Description |
|---|---|
| **User and Ownership Controls** | **User Management** - Select this option if you want to create and manage users. |

Select the following **Functional Controls** based on your requirements:

| Field | Description |
|---|---|
| **Manage APIs** | To create and manage APIs. |
| **Activate/Deactivate APIs** | To activate, deactivate and manage APIs. |
| **Publish to API Portal** | To publish assets to API Portal. |
| **Manage Applications** | To create and manage applications and register applications with the APIs. You cannot modify or delete an application if you are not the owner of the application. |
| **Manage aliases** | To create and manage aliases. |
| **Manage Global Policies** | To apply a global policy to all APIs or the selected set of APIs. |
| **Activate/Deactivate Global Policies** | To activate and deactivate global policies. |
| **Manage Policy Templates** | To apply one or more policy templates to an API. |
| **Manage Threat Protection Policies** | To prevent malicious attacks on applications that typically involve large, recursive payloads, and SQL injections. |
| **Manage Packages and Plans** | To create packages and plans, associate a plan with a package, and associate APIs with a package. In addition, you can view the list of packages, package details, APIs, and plans associated with the package. |
| **Activate/Deactivate Packages** | To activate and deactivate packages. |

| Field | Description |
|---|---|
| **Import Assets** | To import already exported APIs, application, policies, and aliases by selecting *Username* > *Import* in API Gateway. |
| **Export Assets** | To export assets to your local system. |
| **Manage general administration configurations** | To create and manage administration configurations. |
| **View Administration Configurations** | To view administration configurations. |
| **Manage General Configurations** | To manage general configurations. |
| **Manage Security Configurations** | To create and manage security configurations. |
| **Manage Destination Configurations** | To publish events and performance metrics data to the configured destinations. |
| **Manage System Settings** | To create and manage system settings. |
| **Purge/Restore Runtime Events** | To purge and restore events from the API Gateway store by setting the required date or duration in API Gateway. |
| **Manage Service Result Cache** | To manage caching of the results of API invocations depending on the caching criteria defined. |
| **Manage Promotions** | To add, modify, and delete API Gateway stages, or move API Gateway assets from the source stage to one or more target stages, or to rollback an asset promotion that is already available in the target stage at any time. |
| **API Portal Administrator** | To manage all API Portal administrative tasks. |
| **API Portal Provider** | To manage all API Portal provider tasks. |

8. The **Solution Permissions** page displays the webMethods Integration Server User Groups for all the solutions. You can map webMethods Integration Server user groups to an Access Profile. Enter the names of the webMethods Integration Server User Groups separated by a comma, for example, Administrators, Developers, and so on. Integration Cloud users who are assigned to this Access Profile will then be a part of the webMethods Integration Server user group(s) and can perform tasks allowed for those user groups. If you do not map an Access Profile to a webMethods Integration Server user group, you will not be able to view, edit, or run webMethods Integration Server services in a *solution*. For information about user groups, see the Managing Users and Groups section in the *webMethods Integration Server Administrator's Guide*.

> **Note:**
> Integration Cloud Administrator profiles are *automatically assigned* to the webMethods Integration Server Administrators User Group.

> **Note:**
> To view and access webMethods Integration Server packages in Integration Cloud, you must assign any custom user groups created in webMethods Integration Server, which are assigned to Access Profiles in the **Solution Permissions** page, to the following Access Control Lists in webMethods Integration Server: Administrators ACL, Developers ACL, and Replicators ACL.

9. Click **Apply**.

   The new Access Profile appears in the **Access Profiles** page.

10. Click on the Access Profile link in the **Access Profiles** page. In the **Associated Users** page, you can view the active users associated with the selected Access Profile. In the **Associated ACLs** page, you can view the Access Control Lists associated with the selected Access Profile.

## Access Control Lists

You can use Access Control Lists (ACLs) to control the execution permission of an Integration. ACLs provide you with another level of control over who can execute specific Integrations. An ACL can be assigned to an Integration and a user can be associated with the ACL through the Access Profile. Therefore using ACLs, you can control the users who can execute an Integration.

**Example 1**

You have three users U1, U2, and U3. U1 is assigned to Access Profile AP1, U2 is assigned to Access Profile AP2, and U3 is assigned to Access Profile AP3. Each user has the Integration execution permission. There are also four Integrations IN1, IN2, IN3, and IN4 in your tenancy. Initially, U1, U2, and U3 can run all the four Integrations IN1, IN2, IN3, and IN4. Now you want IN1 to be executed only by U1 and *not* by U2 and U3. To do that, create an Access Control List, ACL1. Associate ACL1 to IN1. Then associate ACL1 to AP1. As U1 has already been assigned to AP1, IN1 can be executed by *only* U1. If you want IN1 to be executed also by U2, then associate ACL1 with AP2.

Integration Cloud provides you with a default ACL, *Default*, and this default ACL is associated with all Integrations. You can change the ACL associated with an Integration in the *Integration Details* page.

**Example 2**

You have three users U1, U2, and U3. U1 is assigned to Access Profile AP1, U2 is assigned to Access Profile AP2, and U3 is assigned to Access Profile AP3. Each user has the Integration execution permission. There are also four Integrations IN1, IN2, IN3, and IN4 in your tenancy. Initially, U1, U2, and U3 can run all the four Integrations IN1, IN2, IN3, and IN4. Now you want U1 to run only IN1 and *not* IN2, IN3, and IN4. To do that, create an Access Control List, ACL1. Associate ACL1 to IN1. Then associate ACL1 to AP1, AP2, and AP3. Now disassociate AP1 from the default ACL.

As AP1 is associated with *only* ACL1, U1 will be able to execute only those Integrations associated with ACL1.

Users who have the **Access Control** permission under **Settings > Access Profiles > Administrative Permissions > User and Ownership Controls** can edit the ACL information.

To edit an existing ACL other than the default ACL, select the ACL and click **Edit**. To delete an existing ACL other than the default ACL, select the ACL and click **Delete**. If you delete an ACL, the ACL will be removed from the associated Integration and the Integration will be associated with the default ACL. To create a new ACL, click **Add New Access Control List**.

## Adding or Updating Access Control Lists

Use the **Access Control Lists** page to create, edit, or delete Access Control Lists (ACLs). You can also edit the default ACL, *Default*, but you cannot delete it.

### To add or update an ACL

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> Access Control Lists**.

2. Click **Add New Access Control List** to add an ACL or click **Edit** to modify an existing ACL.

3. On the **Add New Access Control List** or **Update Access Control List** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Name** | Provide a name for the ACL. The name cannot be modified after you save the ACL. |

4. On the **Associate with Access Profiles** tab, complete the following fields:

| Field | Description |
|---|---|
| **Select Access Profiles** | Select the Access Profiles that you want to associate with the ACL. All Access Profiles created in Integration Cloud appear in the panel. The *Administrator* Access Profile will always be associated with all the ACLs, therefore users associated with the Administrator Access Profile will be able to execute all Integrations. |

5. Click **Apply**.

# Preferences

The **Settings** ⚙ **> Preferences** screen allows you to provide the settings for the following:

- Smart Mapping
- Try-catch error display

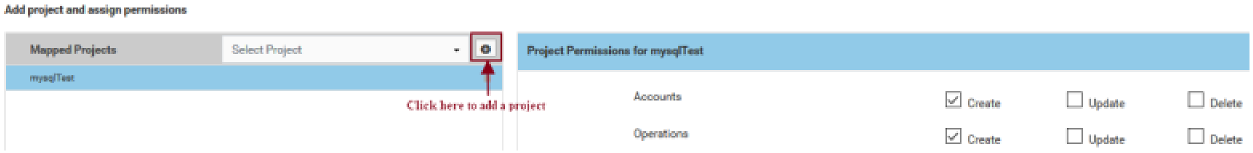
**Note:**
Only the users with Administrator (Access Profile ID 4) privileges can edit the **Preferences** screen.

| Option | Description |
|---|---|
| **Publish Integration Mappings to Recommendations Engine** | Select this check box to enable the smart mapping feature. This provides you mapping recommendations whenever you perform mapping. For more information, see "Smart Mapping" on page 406.<br>**Note:**<br>By enabling this feature, you are also providing us your consent to collect your mapping information. For trial tenants, this feature is enabled by default. |
| **Mark Integration as Completed with Errors, if errors are caught with try-catch blocks** | Select this check box to view on-screen messages when an exception occurs while executing an operation or a service within an integration using a Try-Catch block.<br>When this check box is enabled, if an exception occurs within a Try block, then the end result of the integration execution is shown as completed with errors. Even if the error is handled in the Catch block, the integration is shown as completed with errors.<br>When this check box is not selected, if an exception occurs within a Try block, then the end result of the integration execution is shown as integration completed successfully. It is the responsibility of the integration developer to handle the error inside the Catch block.<br>For example, let us consider an integration that contains a **divideInts** Math service within a Try-Catch block. While doing the division, if we enter a string value instead of an integer and execute this integration and if this check box is not selected, then you will see the message **Integration completed successfully**. If the check box is selected, then you will see the message **Integration completed with errors**. |

## Project Permissions

A project is an independent entity and corresponds to a folder for organizing your assets. A project holds all the assets created as a part of that project by the logged-in user, along with the configurations associated with the assets. Project permissions are used to associate permissions with projects.

You associate permissions with projects from the **Settings** ⚙ **> Project Permissions > Add New Project Permission** page.



Add project and assign permissions

| Mapped Projects | Select Project |
|---|---|
| mysqlTest | |

Project Permissions for mysqlTest

| | | | |
|---|---|---|---|
| Accounts | ☑ Create | □ Update | □ Delete |
| Operations | ☑ Create | □ Update | □ Delete |

Integration Cloud provides a system-generated **Developer** project permission profile in your tenant. Any new project created is automatically associated with the **Developer** project permission profile. This profile has permissions to create, update, delete, and execute all assets. You cannot edit or delete this system-generated project permission profile. If a project permission profile is associated with a user on the user profile page, the user can perform only the permitted tasks in the mapped project.

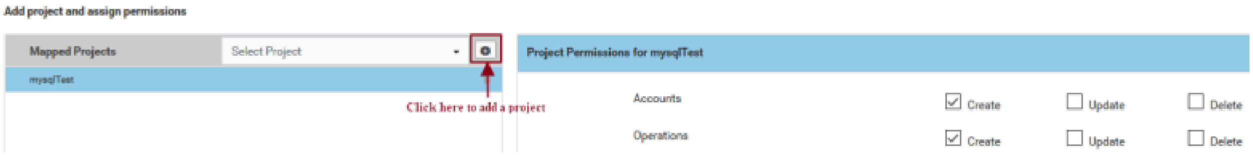
**If you are an existing tenant:**

- Your user's **Access Profile** controls global permissions as well as permissions for the **Default** project.
- Your user's Project Permissions profile control permissions for other projects.
- All existing user profiles will be associated with the **Developer** project permission profile.
- All projects created have the **Developer** project permission profile associated with it.
- All existing assets will be available in the *Default* project.
- Manage the permissions for existing assets inside the **Default** project from the **Project Permissions for Default Project** section under **Settings** > **Access Profiles** > **Administrative Permissions**.
- Manage the permissions for new assets created inside the **Default** project from the **Project Permissions for Default Project** section under **Settings** > **Access Profiles** > **Administrative Permissions**.
- If you create a new project, you have to assign the project permissions from the **Settings** > **Project Permissions** > **Add New Project Permission** page.

**If you are a new tenant:**

- The *Default* project is not available. There are no existing assets. Your user's **Access Profile** controls only global permissions.
- The Administrator will be associated with the **Developer** project permission profile and can assign a project permission profile to a new user.
- All projects created have the **Developer** project permission profile associated with it.
- If you create a new project, you have to assign the project permissions from the **Settings** > **Project Permissions** > **Add New Project Permission** page.

Once you log in, from the Integration Cloud navigation bar, click **Settings** > **Project Permissions** > **Add New Project Permission**. Select a project and then click the add icon ⊕ to add the project in the panel. Then assign the relevant permissions to the selected project . You can select another project and assign the permissions to the selected project. Click **Add**.


Add project and assign permissions
Mapped Projects
Select Project
Click here to add a project

mysqlTest

Project Permissions for mysqlTest

| | | | |
|---|---|---|---|
| Accounts | ☑ Create | □ Update | □ Delete |
| Operations | ☑ Create | □ Update | □ Delete |

**Note:**
Integration **Execute** permission available as part of **Project Permissions** are applicable only for top-level projects. For example, if you have the Integration **Execute** permission for project B but not for project A, and have shared assets of project A with project B, then when you execute an integration of project B which has an integration of project A, the integration execution will be successful.

**Note:**
If you associate the project permission profile to a user on the user profile page, the user can perform only the permitted tasks in each mapped project.

### Example 1

A user is assigned to two project permission profiles A and B in the user profile. Project permission profile A has the Execute Integration permission in Project X but Project permission profile B does not have the Execute Integration permission for the same Project X. The user will still be able to execute the Integration.

### Example 2

In this example, we will see how you will allow an existing user U1 to create, update, delete, and execute Integrations for the Project P1 but not for Project P2.

1. You must have the **Access Control** permission under **Access Profiles > Administrative Permissions**.

2. Create Project P1 and Project P2.

3. Create a new Project Permission profile PP1 under **Settings > Project Permissions > Add New Project Permission**.

4. On the **Add New Project Permission** page, add Project P1 in the mapped projects list. Assign the create, update, delete, and execute Integration permissions to the mapped Project P1.

5. Do not map Project P2.

6. Go to the **Basic** tab of user U1 and select *only* PP1.

So when U1 logs in, U1 will only be able to create, update, delete, and execute Integrations for the Project P1 but not for Project P2.

## Single Sign-On

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

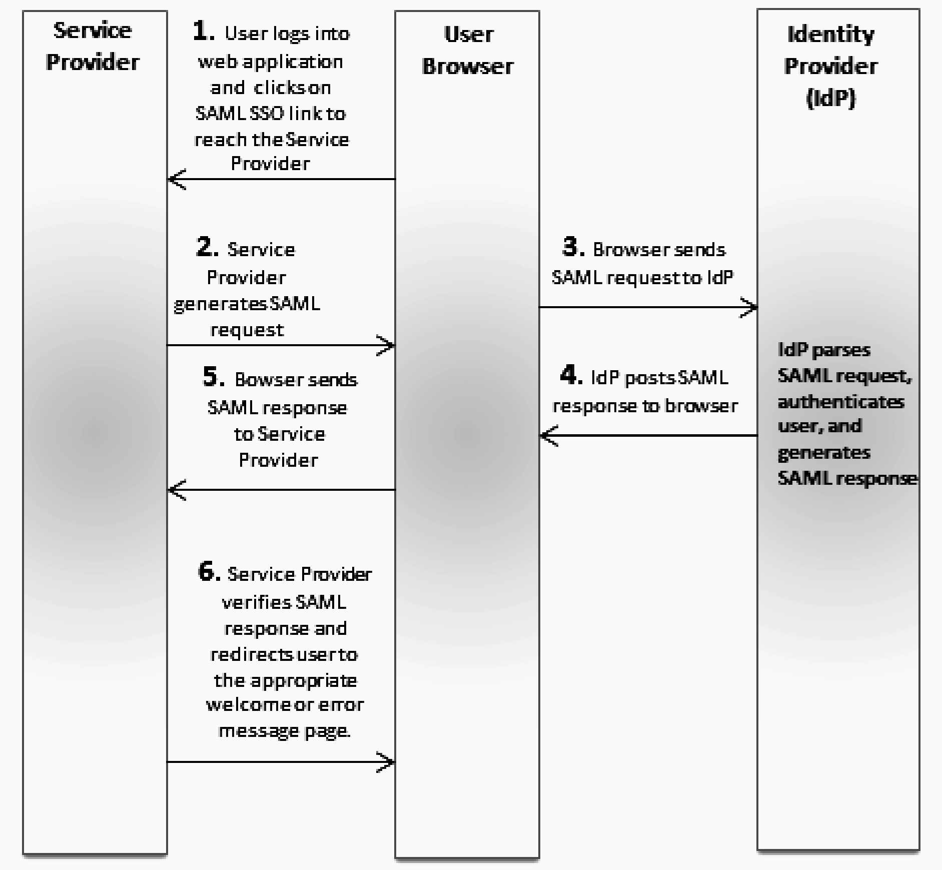
Single sign-on is a process that allows users to access all authorized network resources without having to log in separately to each resource.

Security Assertion Markup Language 2.0 (SAML 2.0) is a standard for exchanging authentication and authorization data between security domains. SAML 2.0 is an XML-based standard that uses security tokens containing assertions to pass information about a principal (usually an end user), between a SAML authority, that is, an identity provider (IdP), and a SAML consumer, that is, a service provider. Using SAML, a service provider can contact an identity provider to authenticate users who are trying to access secure content.

**Note:**
Currently, only SAML 2.0 is supported.

Integration Cloud supports single sign-on (SSO) that allows users to authenticate themselves against an Identity Provider (IdP) rather than obtaining and using a separate username and password. Under the SSO setup, Integration Cloud works as a Service Provider through SAML. You can put the IdP you already trust in charge of authentication, while your users can access Integration Cloud without another password to manage.

The following actions take place while logging into Integration Cloud using SAML 2.0:


Service Provider
1. User logs into web application and clicks on SAML SSO link to reach the Service Provider
User Browser
Identity Provider (IdP)
2. Service Provider generates SAML request
3. Browser sends SAML request to IdP
5. Bowser sends SAML response to Service Provider
4. IdP posts SAML response to browser
IdP parses SAML request, authenticates user, and generates SAML response
6. Service Provider verifies SAML response and redirects user to the appropriate welcome or error message page.

**Service Provider**: Integration Cloud

**Identity Provider (IdP)**: Microsoft Azure, Okta, Oracle Access Manager

1. User logs into a web application and clicks on the SAML SSO link to access Integration Cloud.
2. Integration Cloud generates a SAML authentication request and posts the request to the user's browser.
3. The browser sends the SAML request to the Identity Provider for authentication. The SAML request contains user information, Identity Provider URL, and the assertion response URL.
4. The Identity Provider decodes the SAML request, extracts the URL, authenticates the user, generates a SAML response, and posts the SAML response to the browser.
5. The browser sends the SAML response to Integration Cloud.
6. Integration Cloud checks if the Identity Provider authentication was successful, that is, verifies the SAML response, and redirects the user to the appropriate home page or the error message page.

**Note:**
Integration Cloud SSO capability has been tested to work with Microsoft Azure Active Directory (Azure), Oracle Access Manager (OAM), and Okta as Identity Providers.

You can click **Edit** to configure SAML 2.0 settings for single sign-on or click **Export SAML 2.0 Metadata** if you want to export the Integration Cloud SAML metadata.

See Configuring SAML Settings for Single Sign-On on how to configure SAML settings for single sign-on.

**Note:**
If you have already configured SAML based single sign-on (SSO), the **SSO Login** option appears in the login page. If you click the **SSO Login** option, Integration Cloud redirects you to the Identity Provider (IdP) login page. After you provide the IdP login credentials, you will be logged into Integration Cloud.

**Note:**
You can access or edit the single sign-on configuration page only if you can edit the **Company Information**, that is, have the **Manage Company Capabilities** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Account Controls**.

## Configuring SAML Settings for Single Sign-On

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

The **Single Sign-On Configuration** screen allows you to configure SAML 2.0 settings for single sign-on (SSO). To prevent modifications to the SSO configurations, the SSO settings may not be enabled in your organization.

**Note:**
You can access or edit the single sign-on configuration page only if you can edit the **Company Information**, that is, have the **Manage Company Capabilities** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Account Controls**.

**Note:**
If you have configured SSO, the **SSO Login** option appears in the login page. You can click the **SSO Login** option to log in to Integration Cloud without providing your Username and Password.

➢ **To configure SAML 2.0 settings for single sign-on**

1. From the Integration Cloud navigation bar, click **Settings** ⚙ **> Single Sign-On**.

2. Click **Edit**.

3. On the **Update Single Sign-On Configuration** screen, select **SAML 2.0** in the **Sign-On Using** field and make the necessary modifications. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Choose Single Sign-On Type** | |
| Sign-On Using | Select the sign-on type from the drop-down list. Default is **None**.<br>Security Assertion Markup Language 2.0 (SAML 2.0) is an XML-based standard for exchanging authentication and authorization data between security domains. Integration Cloud (Service Provider) must enroll with an Identity Provider (IdP) and obtain an Identity Provider URL. |
| **Requestor Details** | |
| Authentication Service URL | This URL is the SAML SSO link and is used to trigger the SAML based single sign-on. Use this link to login to Integration Cloud using your Identity Provider.<br>To login to API Gateway Cloud, add `done=apiGatewayUIHome` parameter to the Authentication Service URL.<br>To login to API Portal Cloud, add `done=apiPortalUIHome` parameter to the Authentication Service URL. |
| Assertion Consumer Service URL | This is the URL which consumes the SAML response from the Identity Provider. You need to apply this URL in the relevant field in the Identity Provider SAML configuration page.<br>For Oracle Access Manager (OAM), apply it in the Assertion Consumer Service URL field.<br>For Microsoft Azure, apply it in the Reply URL field.<br>For Okta, apply it in the Single sign on URL field. |
| RelayState for Identity Provider initiated SSO | RelayState is a parameter used by SAML protocol implementations to identify the specific resource at the resource provider, in an Identity Provider initiated single sign-on scenario. In an Identity Provider initiated single sign-on scenario, you must set the RelayState value in the Identity Provider. Test the Identity Provider initiated SSO only after configuring the RelayState.<br>For Oracle Access Manager (OAM), apply the RelayState value as the Return URL in the Identity Provider initiated URL.<br>For Microsoft Azure, send the RelayState value to Microsoft Azure AD to configure the RelayState for your application instance. See Microsoft Azure website for more information.<br>For Okta, apply it in the Default RelayState field. |
| **Identity Provider Configuration** | |

| Field | Description |
|---|---|
| SAML Request Issuer URL | This is the Integration Cloud (Service Provider) URL used to access this tenant. This URL acts as the Service Provider ID.<br>For Oracle Access Manager (OAM), apply it in the Provider ID field.<br>For Microsoft Azure, apply it in the Identifier field.<br>For Okta, apply it in the Audience URI (SP Entity ID) field. |
| Identity Provider Details | Specify how you want to define the Identity Provider details.<br>Select **Enter Manually** if you want to manually enter the URL that uniquely identifies Integration Cloud in your SAML Identity Provider in the **Issuer** field.<br>Select **Load From Identity Provider Metadata** and select the metadata file to upload the IdP details. |
| Issuer | A URL that uniquely identifies Integration Cloud in your SAML Identity Provider. Integration Cloud (Service Provider) must enroll with an Identity Provider and obtain an Issuer URL.<br>If you have selected **Enter Manually** for **Identity Provider Details**, copy the URL provided by the IdP here after setting up Integration Cloud configuration in the IdP.<br>If you have selected **Load From Identity Provider Metadata** for **Identity Provider Details** and uploaded the IdP file, the **Issuer** field will be automatically populated.<br>For Microsoft Azure, copy the URL from the Issuer URL field.<br>For Oracle Access Manager (OAM), copy the URL from the Provider Id field under Federation Settings.<br>For Okta, copy the URL from the Identity Provider Issuer field. |
| Identity Provider Certificate | This is the authentication certificate (a valid x509 issuer certificate) issued by your Identity Provider and is required to sign and verify SAML messages.<br>If you have selected **Enter Manually** for **Identity Provider Details**, select **Browse** and upload a file that contains the Identity Provider's certificate.<br>If you have selected **Load From Identity Provider Metadata** for **Identity Provider Details** and uploaded the IdP file, the IdP certificate will be automatically uploaded. |
| Identity Provider Login URL | This is the URL used to log in to the Identity Provider. |

| Field | Description |
|---|---|
| | If you have selected **Enter Manually** for **Identity Provider Details**, type the URL that will be used to log in to the Identity Provider.<br>If you have selected **Load From Identity Provider Metadata** for **Identity Provider Details** and uploaded the IdP file, the IdP login URL will be automatically populated.<br>For Oracle Access Manager (OAM), the URL is *http://<oamserverhost name>:14100/oamfed/idp/samlv20*.<br>For Microsoft Azure, copy the URL from the Single sign-on service URL field.<br>For Okta, copy the URL from the Identity Provider Single Sign-On URL field. |
| User ID Type | Determines the type of identifier.<br>Assertion contains user's Integration Cloud username - Select this option if your Identity Provider passes the username 👤 > *User Profile* > **Basic** tab) in the SAML assertion to identify the user.<br>Assertion contains the Federation ID from the User Object - The Federation ID acts as a user's authentication across multiple IT systems or organizations. A federated identity means linking a person's electronic identity and attributes stored across multiple distinct identity management systems. Select this option if your Identity Provider passes the Federation ID (👤 > *User Profile* > **Basic** tab), to identify the user. You can add the **Federation ID** (👤 > *User Profile* > **Basic** tab) to each user’s profile after you have configured single sign-on. |
| User ID Location | Specifies an attribute tag that defines the location of the User ID. This is the location in the assertion where a user should be identified.<br>Select **Subject** if the User ID is located in the <Subject> statement of the assertion.<br>Select **Attribute** if the User ID is specified in an <AttributeValue>, located in the <Attribute> of the assertion. If you have selected **Attribute**, specify the attribute that contains the User ID in the **Attribute for User ID** field. If the User ID attribute is empty or does not match an existing user, then either login fails or a new user is created, depending on the **Create Users** setting. |
| Attribute for User ID | This field appears if you have selected **Attribute** in the **User ID Location** field. Specify the attribute that contains the User ID. If the User ID attribute is empty or does not match an existing user, then |

| Field | Description |
|---|---|
| | either login fails or a new user is created, depending on the **Create Users** setting. |
| Create Users | Select this option to create a new user when the User ID is not recognized. When selected, additional options appear where you can specify the attribute to use for the First Name, Last Name, Email, and Access Profile.<br>**Attribute for First Name** - The name of the SAML attribute that designates the user's first name.<br>**Attribute for Last Name** - The name of the SAML attribute that designates the user's last name.<br>**Attribute for Email** - The name of the SAML attribute that designates the user's email address.<br>**Default Access Profile** - This field is used to specify the default Access Profile for the created user.<br>**Attribute for Access Profile** - The name of the SAML attribute that designates the user's access profile. The attribute must contain the **ID** of the Access Profile. You can get the ID of the Access Profile from the **Access Profiles** screen (**Settings** ⚙ **> Access Profiles**). |

**Note:**
You must select Email Address as the NameID Format in the Identity Provider SSO Configuration screen.

## Company Information

This screen displays your company information. Users who have the **Manage Company Capabilities** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Account Controls** can edit the company information.

See "Updating Company Information" on page 68 for information on the fields.

Click **Edit** to update the company information.

### Updating Company Information

You can view and update the company information and use them across all applications in the platform.

**To update the Company Information**

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> Company Information**.

2. Click **Edit**.

3. In the **Basic** section, complete the following fields. Required fields are marked with an asterisk on the screen.

| **Field** | **Description** |
|---|---|
| **Tenant ID** | This is the unique ID assigned to your organization's tenancy on the platform.<br>**Note:**<br>This field cannot be edited and appears in **view only mode** under the **Basic** tab. |
| **Sub Domain** | This is the unique sub domain that you specified during registration. A sub domain is a domain that is part of a main domain. For example, suppose you are at ABC Company and you decide to use “abc” as your unique sub domain. With that setting, you will access your instance of the platform at https://abc.webmethodscloud.com.<br>**Note:**<br>This field cannot be edited and appears in **view only mode** under the **Basic** tab. |
| **Company Name** | The name of the company. This field accepts only alphanumerics, spaces, and hyphens (-). The company name is automatically populated from the **Registration** screen. |
| **Street** | The street address of the company. |
| **City** | The city where the company is located. |
| **State/Province** | The state or Province where the company is located. The state or province name is automatically populated from the **Registration** screen. |
| **Postal/Zip Code** | The postal or zip code for the company. |
| **Country** | The country where the company is located. The country name is automatically populated from the **Registration** screen. |
| **System Notification Email Addresses** | Enter an address or comma-separated email addresses to receive system notifications. Such notifications can occur, for example, when a connection to the system mailbox fails after repeated attempts. This field displays the information from the **Registration** screen but you can change that later using the **Edit** button. |

4. In the **Advanced Information** section, complete the following fields:

| Field | Description |
|---|---|
| **Time Zone** | Choose your time zone from the drop down list. |
| **Time Format** | Choose a time format from the drop down list. You can choose a 12-hour clock with AM/PM or a 24-hour clock.<br>hh:mm a - 12-hour clock - 3:30 AM, 3:30 PM<br>HH:mm - 24-hour clock - 3:30, 15:30 |
| **Date Format** | Choose a date format from the drop down list.<br>mm is "Month", dd is "Day", yyyy is Year and the delimters are:(/) slash or stroke(-) dash or hyphen(.) period, dot, or full stop. |
| **Default Locale** | This setting determines the language for the tenant.<br>**Note:**<br>Users within a tenant can set their preferred language from the **Users > Locale** tab.<br>For example, if you set the value here as `English` and the user for this tenant sets the value as `Chinese` in the **Locale** field, then the user will see all the application labels in the Chinese language.<br>For more information, see "Adding Users" on page 42. |
| **Last Modified** | This field displays the date and time when the company information record was last updated. This field cannot be edited and appears in **view only mode**. |

## Password Policy

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page. Password policies are defined in the Software AG Cloud User **Administration** page.

A Password Policy defines password requirements and login protections. Users who have the **Access Control** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can edit the Password Policy information.

You can view the password policies for the Integration Cloud instance in this screen. See "Updating Password Policy Settings" on page 71 for information on the fields.

Click **Edit** to modify the password policy information.

## Updating Password Policy Settings

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page. Password policies have to be defined in the Software AG Cloud User **Administration** page.

You can set password policies for users on the **Update Password Policy** page.

### To update the Password Policy

1. From the Integration Cloud navigation bar, click **Settings** ⚙ **> Password Policy**.
2. Click **Edit**.
3. On the **Update Password Policy** page, make the necessary modifications.

| Field | Description |
|---|---|
| **Minimum Length** | Select the minimum number of characters in the password. |
| **Required Character types** | This option defines the level of security for passwords, which can be simple and allow any character combination, or very secure, requiring upper and lower case characters, as well as special characters. |
| **Expires in** | Select the number of days the password will remain valid before the user will be prompted to change it.<br>By default, no user is exempt from the Password Policy. You can specify a user to be excluded from the password expiration policy by selecting *Never*. |
| **Password Never Expires for** | Select the users for whom the password will never expire. Only active users appear in the list. You can make an user account active by selecting the **Settings** ⚙ **> Users > Update User > Basic tab > Active** option. |
| **New Password cannot match** | The new password cannot match the number of previous passwords. |
| **Minimum Age** | Select the number of days that must pass before a user can change passwords. |
| **Session Timeout** | Select the length of time the session will remain active without any user activity. The session will end when it reaches the selected timeout. The user will need to log in again. |

| Field | Description |
|---|---|
| **Account Lockout Threshold** | Select the number of login attempts before the account is locked out.<br>The login limit defines the number of failed attempts allowed before a user account is disabled or locked for a specified time. When a user attempts to login and fails (because of an incorrect password), each attempt counts against the login limit. When the login limit is achieved, the account is disabled or locked for a specified time, according to the parameters set in the *Account Lockout Duration* field. The login limit is defined by the *Password Policy*. |
| **Account Lockout Duration** | Select the length of time that an account is locked out. |
| **Record Information** | For audit purposes, the following information is displayed after you save the record:<br>*Last Modified By <username> on {date} <time> Created by System.* |

4. Click **Apply**.

## Client Certificate

Secure Sockets Layer (SSL) is a means of securing communications over a network so that only the sender and receiver have access to the sensitive data.

In a *one-way* SSL connection, an anonymous client authenticates the credentials of a server in preparation for setting up a secure transaction. In most cases, the server knows nothing about the client's identity because verification of its credentials is not required. When desired, the client can be authenticated using basic authentication by providing a username and password. This type of authentication typifies connections where a browser establishes a connection to a server to perform a secure transaction, for example, viewing a savings account, or buying items with a credit card. The client must authenticate the server's credentials before initiating the transaction, but it is not necessary for the server to authenticate and keep a record of every possible client (browser). This type of connection is typically one where a partner application or resource needs to verify the authenticity of the server without itself needing to be authenticated.

Two-way SSL authentication refers to two parties authenticating each other by verifying the provided digital certificate so that both parties are assured of the others' identity. It refers to a client (web browser or client application) authenticating itself to a server and the server authenticating itself to the client by verifying the public key certificate or digital certificate issued by the Certificate Authorities (CAs).

Integration Cloud supports two-way SSL for inbound connections. The request for an SSL connection originates from a client. During the SSL handshake process, the entity acting as the SSL server responds to the request for a connection by presenting its SSL credentials (an X.509 certificate) to the requesting client. If those credentials are authenticated by the client, either:

- An SSL connection is established and information can be exchanged between the client and server.

  - or -

- The next phase of the authentication process occurs, and the server requests the SSL credentials of the client. If the server verifies those credentials, that is, the client's *identity*, an SSL connection is established and information exchange takes place.

**Note:**
When a client or partner application submits a request to Integration Cloud using HTTPS on port 8443, and a two-way SSL connection is established, the client acts as the SSL client and Integration Cloud acts as the SSL server.

The following table provides a high-level roadmap for configuring SSL.

| Task | Activities | Notes |
|---|---|---|
| Create keys and certificates | ■ Generate a public key/private key pair.<br>■ Generate a certificate signing request (CSR) and send it to the certificate authority (CA) for signing.<br>■ Receive validated certificate from the CA. | Two-way SSL connection requires a valid client certificate. |
| Upload client certificate or generate a certificate | Upload the CA signed client certificate for the user in the **Client Certificate** page or generate a private key and a new Integration Cloud signed client certificate. | Required for two-way SSL connections. |
| Connect to Integration Cloud using the client certificate. | Configure the REST client with the private key and certificate. Optionally you can also pass the basic authentication credentials. | Integration Cloud support two-way SSL on port 8443. |

Users who have the **User Management** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can generate or edit any client certificate. Users who have the **Manage Personal Setup** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can generate or edit the user's own certificate.

See "Adding Client Certificates" on page 73 for information on how to add client certificates.

## Adding Client Certificates

Integration Cloud allows you to store client certificates and associate a certificate with a user account. You can add client certificates for users on the **Client Certificate** page. When a client presents one of these certificates, Integration Cloud logs in the client, as the user "mapped" to the certificate.

### To add a client certificate

1. From the Integration Cloud navigation bar, click **Settings** ⚙ > **Client Certificate**.
2. In the **User** field, select a user. Only active users are listed in the **User** field.
3. In the **Upload New Certificate** field, click **Browse** to upload a new client certificate signed by a trusted certificate authority (CA). If a certificate is configured for a user, the **Certificate Details** panel displays the configured certificate. You can click **Download** to download the user certificate or click **Delete** to delete the user certificate. The downloaded file is named as `<username>`.crt.
4. In the **Generate Private Key and Certificate** field, click **Generate** if you want Integration Cloud to generate a private key and a new Integration Cloud-signed client certificate. Integration Cloud validates it against the issuer of the certificate. The generated certificate is named as `<username>`.txt which contains the private key and the client certificate.

## Executing Integrations using Two-way SSL

### Summary

Two-way SSL authentication, also referred to as client or mutual authentication or certificate-based authentication, refers to two parties authenticating each other by verifying the provided digital certificate, so that both the parties are assured of the other's Identity.

Two-way SSL authentication involves the following steps:

1. Client (Postman, SoapUI) requests access to protected resources of server (webMethods Integration Cloud).
2. Server presents its certificate to the client.
3. Client validates the server's certificate.
4. Client sends its certificate to the server.
5. Server verifies the client’s certificate.
6. If successful, the server grants access to the protected resources requested by the client.

In this tutorial, we will create an integration in webMethods Integration Cloud, expose the integration over HTTP (exposing the integration over HTTP allows the integration to be executed from an outside environment), and then execute the integration using two-way SSL authentication by using a REST Client (Postman). You can also use the same technique for SOAP APIs, REST APIs, or any other exposed APIs.

## Actors

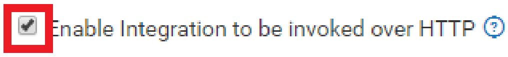
- Integration developers who develop and expose the integrations over HTTPS in Integration Cloud.
- Integration executors who runs integrations.

## Before you begin

- You must have the permissions to create and execute integrations in Integration Cloud under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Integrations**.

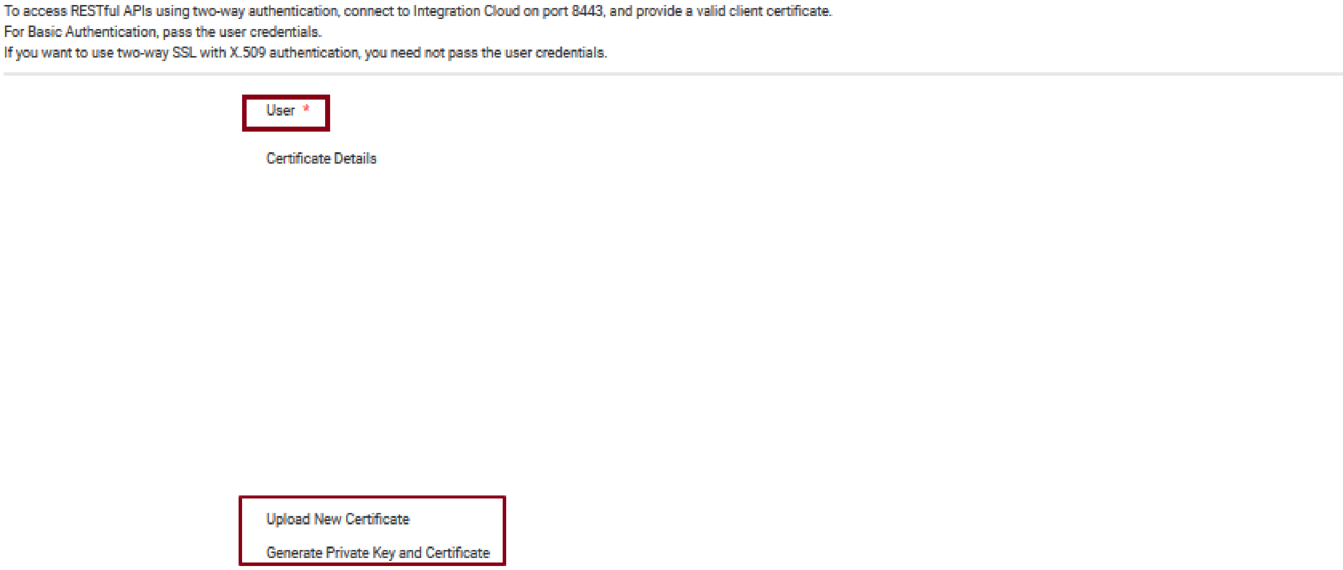
## Basic Flow

1. Log in to Integration Cloud.
2. Create an Integration (testApp) in Integration Cloud.
3. On the Integrations page, click the integration link, and on the integration **Overview** page, select the **Enable Integration to be invoked over HTTP** option.



☑ Enable Integration to be invoked over HTTP ?

4. Click **Settings** ⚙ **> Client Certificate**.

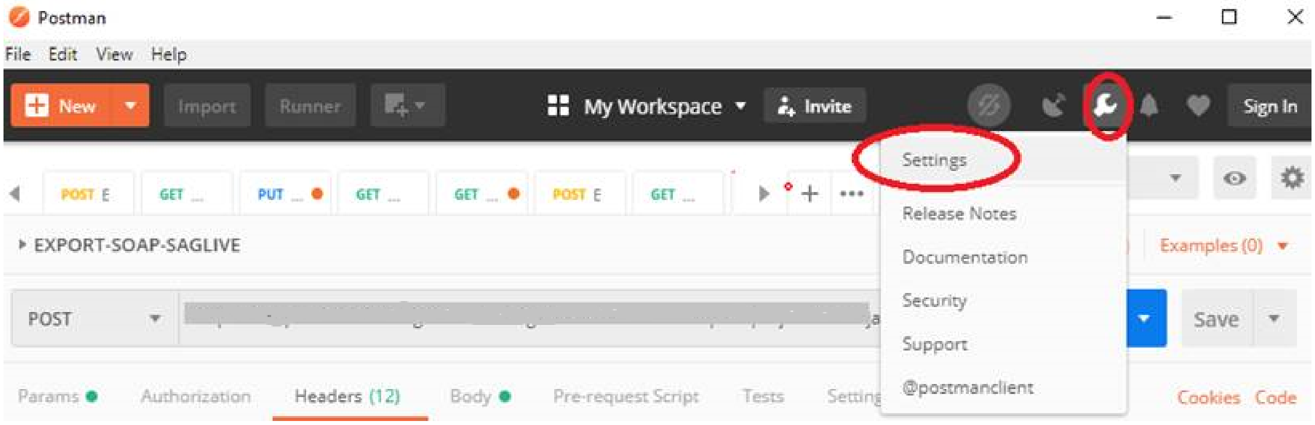


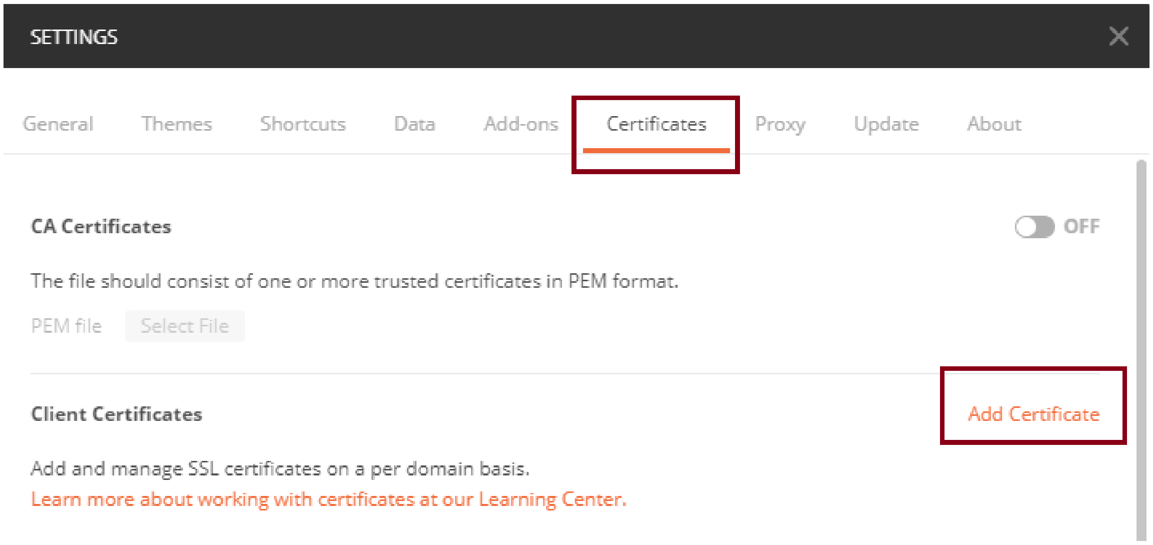
To access RESTful APIs using two-way authentication, connect to Integration Cloud on port 8443, and provide a valid client certificate.
For Basic Authentication, pass the user credentials.
If you want to use two-way SSL with X.509 authentication, you need not pass the user credentials.

5. Select the **User**. You can either upload a certificate to the user if there are any available CA-signed certificates, or you can generate and assign a certificate to the user. Click **Browse** to upload a certificate if you want to use the user's own certificate or click **Generate** to generate and download the private and public key for the user.

After downloading the file, copy the private key to a file and name it as `{privateKeyFileName}.key` and the public key to a file and name it as `{publicKeyFileName}.crt`.

6. Open the Postman REST client and click **Settings > Certificates**. Then click **Add Certificate**.

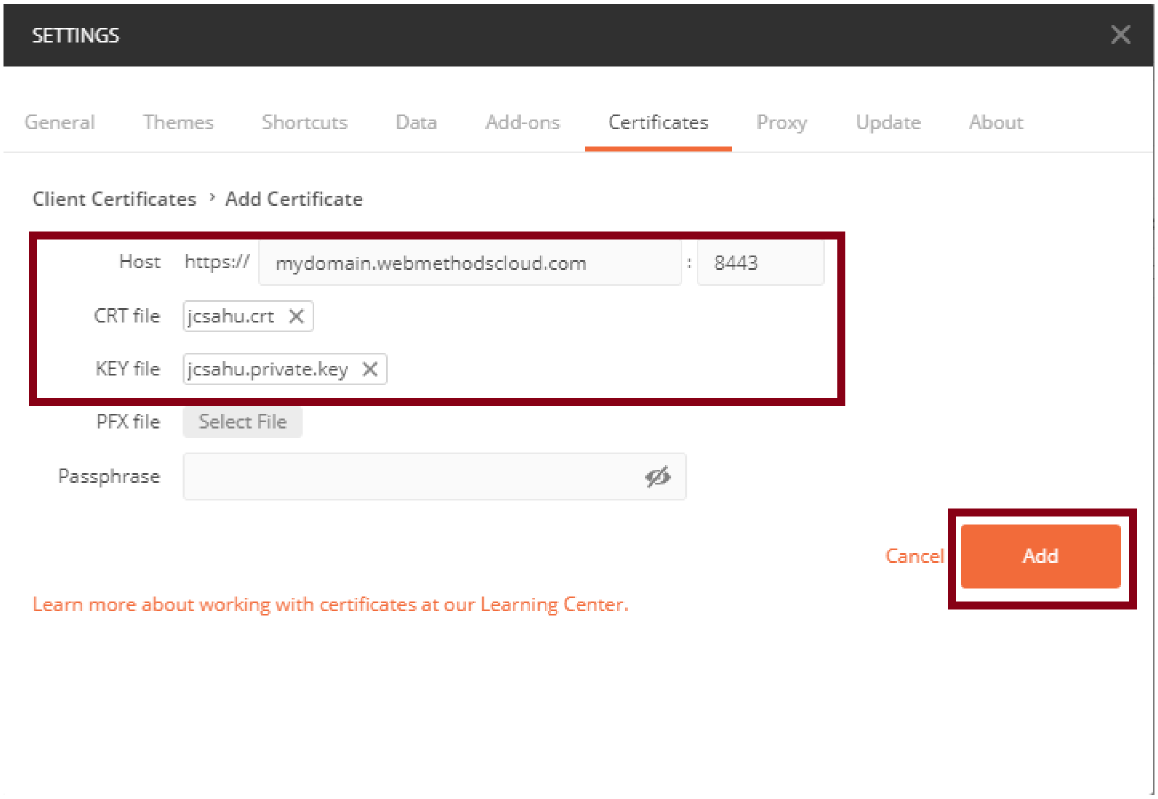




The file should consist of one or more trusted certificates in PEM format.

Add and manage SSL certificates on a per domain basis.
Learn more about working with certificates at our Learning Center.

7. The Add Certificate page appears. Now configure the certificate and private key in Postman.



Host https:// mydomain.webmethodscloud.com : 8443

CRT file jcsahu.crt

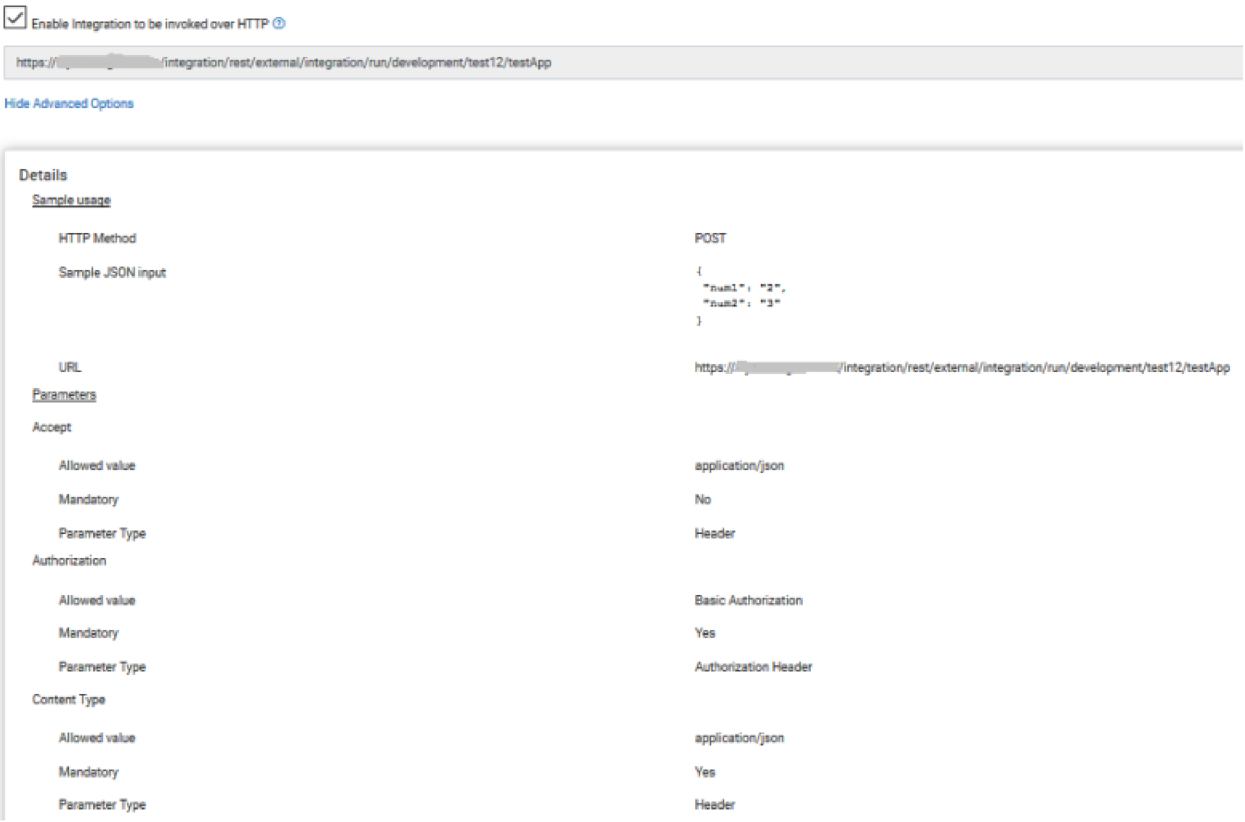
KEY file jcsahu.private.key

Passphrase

Learn more about working with certificates at our Learning Center.

8. As shown in the above figure, specify the **Host** name and the port number as 8443. Specify the location of the key files, that is, the **CRT file** (certificate) and the **KEY file** (private key). Click **Add** to save the two-way SSL configuration.

9. Open a new tab in Postman and add the request details you have obtained from Integration Cloud. The request details are available after you select the **Enable Integration to be invoked over HTTP** option.



☑ Enable Integration to be invoked over HTTP

https://[redacted]/integration/rest/external/integration/run/development/test12/testApp

Hide Advanced Options

Details

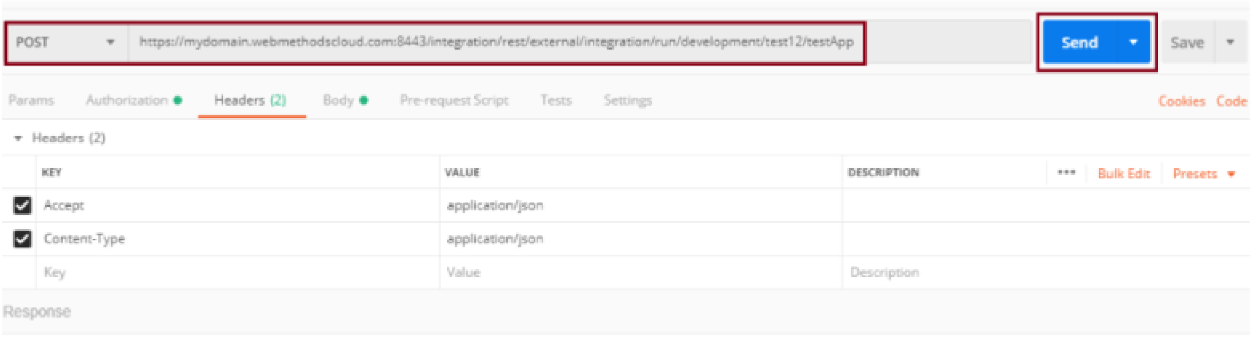
Sample usage

| | |
|---|---|
| HTTP Method | POST |
| Sample JSON input | {<br>"num1": "2",<br>"num2": "3"<br>} |
| URL | https://[redacted]/integration/rest/external/integration/run/development/test12/testApp |

Parameters

| | |
|---|---|
| Accept | |
| Allowed value | application/json |
| Mandatory | No |
| Parameter Type | Header |
| Authorization | |
| Allowed value | Basic Authorization |
| Mandatory | Yes |
| Parameter Type | Authorization Header |
| Content Type | |
| Allowed value | application/json |
| Mandatory | Yes |
| Parameter Type | Header |

10. To execute the Integration, configure a POST request in Postman as shown below and click **Send**. Change the port to 8443 of your service. URL is https://mydomain.webmethodscloud.com:8443/<*Your service URL*>.



POST https://mydomain.webmethodscloud.com:8443/integration/rest/external/integration/run/development/test12/testApp

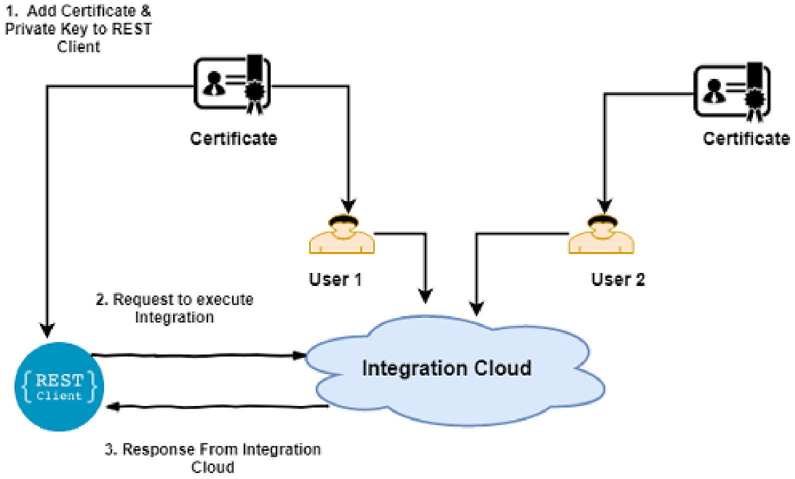
| | KEY | VALUE | DESCRIPTION |
|---|---|---|---|
| ☑ | Accept | application/json | |
| ☑ | Content-Type | application/json | |
| | Key | Value | Description |

## Exceptions

The following errors may occur in the REST Client when there is a certificate mismatch between what is specified in Integration Cloud with what is sent from the REST Client.

- The server could not send a response.
- Self-signed SSL certificates are blocked.

### Visual Model


1. Add Certificate & Private Key to REST Client
Certificate
Certificate
User 1
User 2
2. Request to execute Integration
Integration Cloud
REST Client
3. Response From Integration Cloud

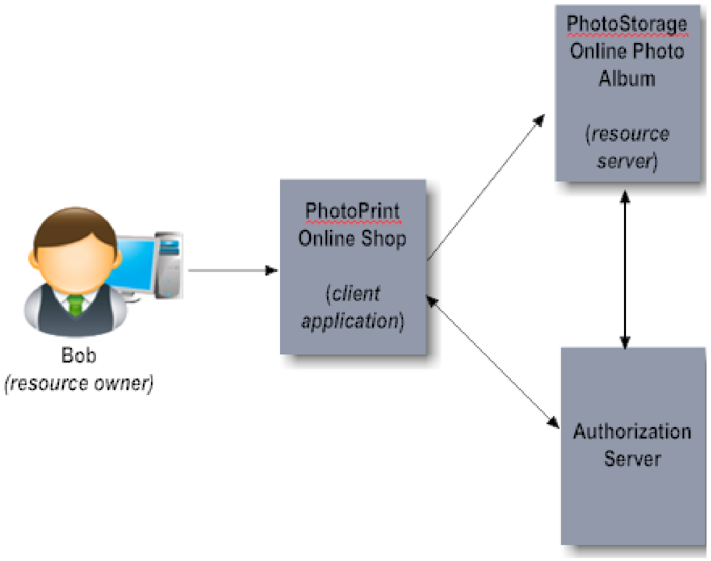
# OAuth 2.0

## About OAuth 2.0

**Note:**
This page is not applicable if you have created your account using the Software AG Cloud sign-up page.

The OAuth 2.0 Authorization Framework facilitates the sharing of private resources (data or services) with a third-party client application (*client*). In an OAuth session, private resources are stored on a *resource server* and the owner of the resources, or *resource owner*, grants the client application permission to access them. The resource owner is typically a person; however, in some cases it could be an application. When a resource owner grants permission, the OAuth *authorization server* issues an *access token* to the client application. When the client application passes the access token to the resource server, the resource server communicates with the authorization server to validate the token and, if valid, provides access to the resources.

The following example illustrates the roles involved with an OAuth session. In the example, Bob is the resource owner who wants to access and print his photos stored on the PhotoStorage website (the resource server) using the PhotoPrint service (the client application). PhotoPrint supplies Bob with an application that runs on his device (phone or laptop). Bob uses that application to initiate the process. PhotoPrint sends a request to the PhotoStorage authorization server. The authorization server requests authorization from Bob and issues a token to PhotoPrint. PhotoPrint can then access Bob's photos on PhotoStorage.



Integration Cloud services can be accessed through REST APIs from any REST client. In OAuth 2.0, the client obtains an access token issued by an authorization server on approval of the resource owner. The client uses the access token to access the protected resources.

**Note:**
Integration Cloud acts both as a Resource server and as an Authorization server.

An in-depth description of OAuth is beyond the scope of this guide but is available elsewhere. For information about the OAuth protocol, see the OAuth 2.0 Authorization Framework.

## Configuring OAuth 2.0

Before you can invoke services using OAuth 2.0 tokens, you must define clients, scopes, associate scopes to clients, and generate OAuth 2.0 tokens. The following table describes how to configure OAuth 2.0.

| Steps | Description |
|---|---|
| “Define Clients” on page 81 | Define the clients that are authorized to invoke services in Integration Cloud. Specify the client name, version number of the client, client type, redirection URLs, allowed grants, expiration interval, and the refresh count. See “Registering Clients” on page 81 for more information. |
| “Define Scopes” on page 84 | A scope defines the services the client can access on behalf of the resource owner. A scope consists of a name and one or more services. If access is granted for a scope, then access is granted for all the services in that scope. |

| Steps | Description |
|---|---|
| "Associate scopes to a client" on page 81 | Associate defined scopes with the registered clients. When you associate scopes, you authorize the scopes that each client can access. |
| "Generate Tokens" on page 86 | Generate tokens (Access Token and Refresh Token) by using a REST Client. See "Generating Tokens" on page 86 for more information. Integration Cloud supports the Authorization Code Grant, Implicit Grant, Client Credentials Grant, and the Resource Owner Password Credentials Grant to generate the access tokens. Clients use the tokens to execute REST URLs for running the Integrations. After you generate the tokens, the tokens are available in the **Token Management** page in Integration Cloud. |

## Registering Clients

Before a client can request access to a protected resource, it should register with Integration Cloud. When you register a client, you identify the client as a confidential client or a public client, select the grant types the client can use, and specify the token expiration and refresh information. The **Client Registration** page lists the clients registered with Integration Cloud.

See "About OAuth 2.0" on page 79 for information on the high-level steps for configuring OAuth 2.0.

**Note:**
When you delete a client, Integration Cloud also deletes all the access tokens and refresh tokens for the client. When you deactivate a client by clearing the **Active** option while updating the client, all the access tokens and refresh tokens for the client become invalid. You can activate a deactivated client.

**Note:**
Users who have the **Access Control** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can create, edit, and delete clients.

### To add a Client

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> OAuth 2.0 > Client Registration > Add New Client**.

2. On the **Add New Client** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Name** | Type the name of the client. You cannot create clients with the same **Name** and **Version** combination. You cannot modify the client name after the client is saved. Client names are not case-sensitive. |
| **Description** | Type a description of the client. |
| **Client ID** | The Client ID field appears only when you update a client. This is a client identifier issued to the client to identify itself to the authorization server, and is used while generating tokens. |
| **Client Secret** | The Client Secret field appears only when you update a client. This is a secret matching to the client identifier and is used while generating tokens. It will not be generated if the Client Type is **Public**. |
| **Authorization Endpoint** | View the authorization URL that has to be provided while generating tokens. See the *Generating Tokens* section for more information. |
| **Token Endpoint** | View the Access Token URL that has to be provided while generating tokens. See the *Generating Tokens* section for more information. |
| **Refresh Token Endpoint** | View the Refresh Token URL that has to be provided while refreshing Access Tokens. See the *Refreshing Access Tokens Using Refresh Tokens* section for more information. |
| **Version** | Type the version number of the client. You cannot create clients with the same **Name** and **Version** combination. |
| **Type** | Select the type of the client according to its ability to communicate with Integration Cloud.<br>**Confidential** - Select **Confidential** when the OAuth session uses the following grants:<br>■ Authorization Code Grant<br>■ Client Credentials Grant<br>■ Resource Owner Password Credentials Grant<br>This client is capable of maintaining secure client authentications. When you select client type as |

| Field | Description |
|---|---|
| | **Confidential**, Integration Cloud generates a client secret. This client secret will be required by Integration Cloud when the client makes requests to the OAuth services.<br>**Public** - Select **Public** when the OAuth session uses the Implicit Grant type. This client is not capable of maintaining secure client authentications. |
| **Redirection URLs** | Specify the URLs that Integration Cloud will use to redirect the resource owner's browser during the grant process.<br>You can add more than one redirection URL.<br>If you select the Authorization Code Grant or the Implicit Grant types, you must enter at least one Redirection URL for the client. |
| **Allowed Grants** | Select the type of grant flow required by the client. |
| **Expiration Interval** | Select the length of time (in seconds) that the access token is valid.<br>**Never Expires** - Indicates that the access token never expires. The Token Management page displays **Lifetime** for that token.<br>**Expires In** - Specify the number of seconds the access token is valid. |
| **Refresh Count** | Select the number of times the access token can be refreshed.<br>**Unlimited** - Refresh the access token an unlimited number of times using the refresh token. The Token Management page displays **Unlimited** for that refresh token.<br>**Limited** - Specify the number of times to refresh the access token. The Token Management page will display the **Refresh Count** for that refresh token. If you specify 0 or leave the field empty, a refresh token will not be issued.<br>**Note:**<br>Tokens can be refreshed only when using the Authorization Code Grant flow. |

| Field | Description |
|---|---|
| **Active** | This option appears only when you update a client. Clear this option to deactivate the client. When you deactivate a client, all the access tokens and refresh tokens for the client become invalid. |

3. Click **Add** to add the client in the **Client Registration** page.

4. On the **Client Registration** page, if you want to associate scopes with a client, select a client and then click **Associate Scopes**. The **Associate Scopes with <ClientName(Version)>** page appears. The **Associate Scopes with <ClientName(Version)>** page displays the already associated scopes with the selected client.

   a. On the **Associate Scopes with <ClientName(Version)>** page, to associate existing scopes with the client, select **Associate Existing Scopes**.

   b. On the **Select Scopes to Associate** dialog box, select the existing scopes to associate with the client and then select **Associate Scopes**. The newly associated scopes will appear in the **Associate Scopes with <ClientName(Version)>** page.

   c. To create a new scope and associate it with the selected client, select **Associate New Scope**. Create the new scope as described in the "Managing Scopes" on page 84 section. The new scope will be associated with the selected client.

   d. To disassociate a scope from a client, select the scope on the **Associate Scopes with <ClientName(Version)>** page and then click **Disassociate**.

## Managing Scopes

A scope defines the services the client can access on behalf of the resource owner. A scope consists of a name and one or more services. If access is granted for a scope, then access is granted for all the services in that scope. When a request is made, Integration Cloud verifies that the scope is defined for a client. The client is allowed to access only the service URLs that are specified for the scope. If the requested scope is not defined, Integration Cloud returns an error indicating that the scope is invalid.

**Note:**
You cannot delete a scope that is used by a client. Also, a scope cannot be deleted if it is associated with an existing token.

**Note:**
Users who have the **Access Control** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can create, edit, and delete scopes.

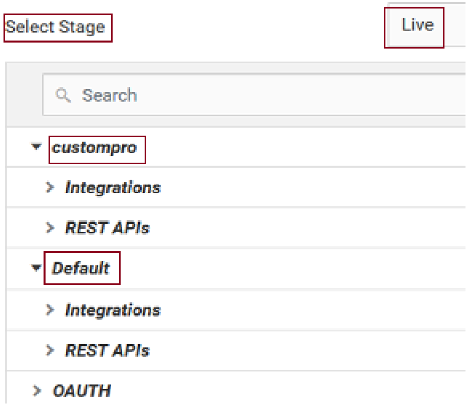
See "About OAuth 2.0" on page 79 for information on the high-level steps for configuring OAuth 2.0.

### To add a scope

1. From the Integration Cloud navigation bar, go to **Settings** ⚙ **> OAuth 2.0 > Scope Management > Add New Scope**.

2. On the **Add New Scope** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Name** | Type a unique name for the scope. You cannot modify the scope name after a scope is saved. Scope names are not case-sensitive. |
| **Description** | Type a description of the scope. |
| **Services** | Specify the list of services that the client can access on behalf of the resource owner for executing the integrations. Click **Select Services** and in the **Services** dialog box, select the exposed Integrations or REST Resources that you want to add as Service URLs from the listed projects *in the selected stage*.<br>The **Services** dialog box displays the exposed integrations and REST Resources available in all projects, that is, in custom projects and in the Default project in the selected stage.<br>**Note:**<br>You can search by the project name, integration name, or by the REST API name. The search function works for integrations and REST APIs only after you have expanded the integrations or REST API nodes in that project. It is recommended to first search for the project and then search for the integrations and REST APIs in that project. |

| Field | Description |
|---|---|
| | <br>Select Stage<br>Live<br>Search<br>custompro<br>Integrations<br>REST APIs<br>Default<br>Integrations<br>REST APIs<br>OAUTH<br>In the **Services** dialog box, select the exposed integrations and REST Resources that you want to add as Service URLs, and then click **Add** or **Update** to add or update the respective service URLs to that scope. |
| **Service URLs** | This field appears once you have added the exposed Integrations and REST Resources and shows the selected services.<br>**Service URL** is a relative URL and it must start with /*integration*. For example, if the absolute URL is https://<subdomain>.webmethodscloud.com/integration/rest/external/integration/run/development/restintegration, then the Service URL is /integration/rest/external/integration/run/development/restintegration. |

3. See the “Registering Clients” on page 81 section on how to associate scopes with a client.

## Generating Tokens

You can generate tokens (Access Token and Refresh Token) by using a REST Client. Integration Cloud supports the Authorization Code Grant, Implicit Grant, Client Credentials Grant, and Resource Owner Password Credentials Grant to generate the access tokens. Clients use the access tokens to invoke REST URLs for running the Integrations.

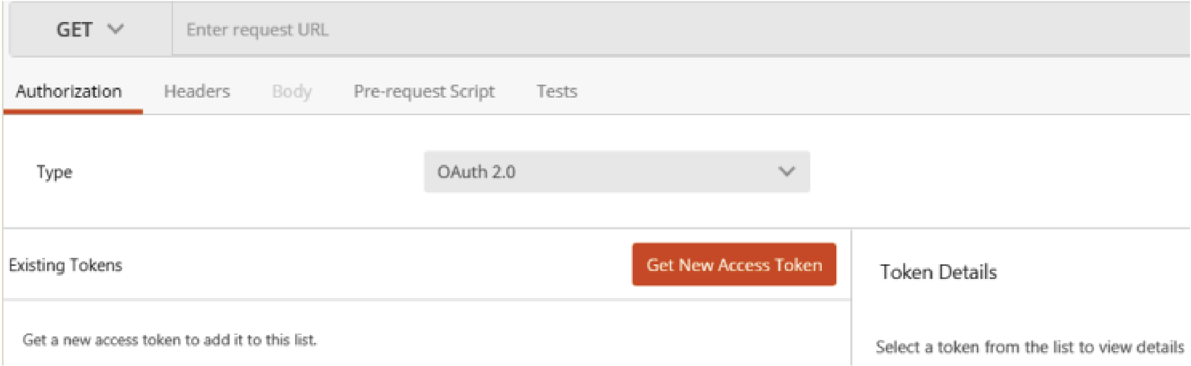
See “About OAuth 2.0” on page 79 for information on the high-level steps for configuring OAuth 2.0.

### The following example shows how to generate tokens using Postman

1. Add the Postman extension to your Google Chrome web browser.

2. Open the Postman application.

3. On the Postman **Authorization** page, select the **Type** as **OAuth 2.0**, and then click **Get New Access Token**. The **Get New Access Token** page appears.



Type: OAuth 2.0

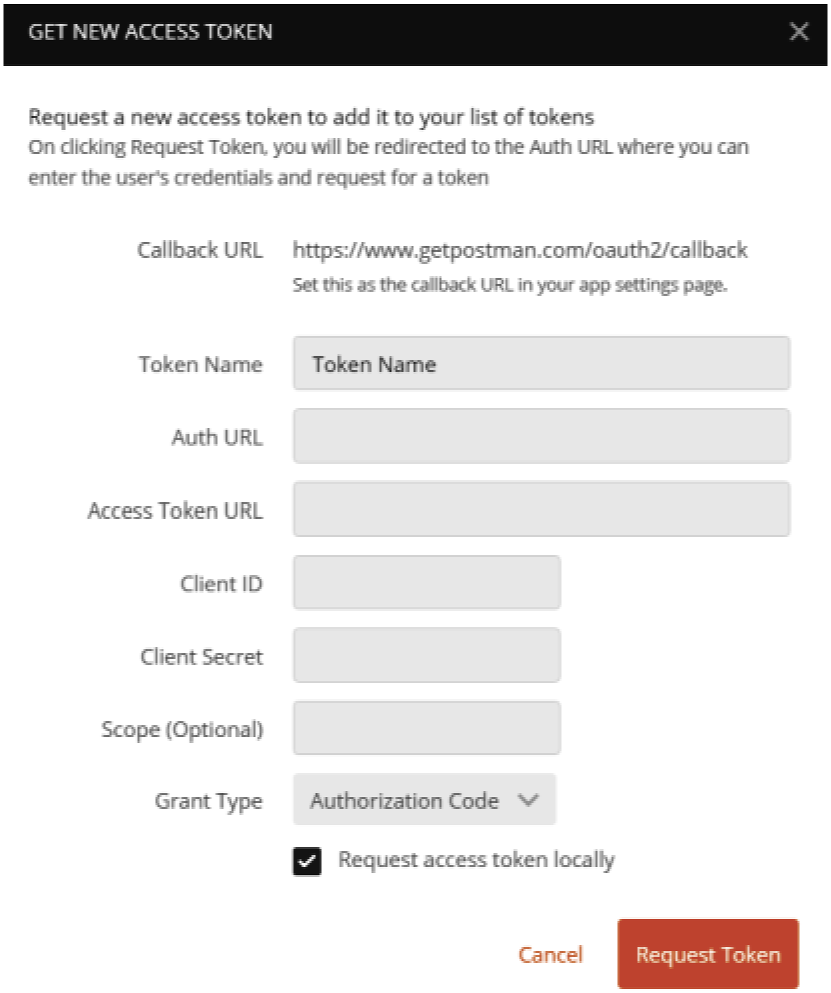
Existing Tokens

Get a new access token to add it to this list.

Token Details

Select a token from the list to view details

4. On the **Get New Access Token** page, complete the following fields to request a new access token.



GET NEW ACCESS TOKEN

Request a new access token to add it to your list of tokens
On clicking Request Token, you will be redirected to the Auth URL where you can enter the user's credentials and request for a token

Callback URL: https://www.getpostman.com/oauth2/callback
Set this as the callback URL in your app settings page.

Token Name: Token Name

Auth URL:

Access Token URL:

Client ID:

Client Secret:

Scope (Optional):

Grant Type: Authorization Code

☑ Request access token locally

| Field | Description |
|---|---|
| **Callback URL** | Specify the redirection URL added during client registration. |
| **Token Name** | Provide a token name. |
| **Auth URL** | Provide the **Authorization Endpoint** URL available on the **Client** page in the following format:<br>https://abc.webmethodscloud.com/integration/rest/oAuth/authorize |
| **Access Token URL** | Provide the Access **Token Endpoint** URL available on the **Client** page in the following format:<br>https://abc.webmethodscloud.com/integration/rest/oAuth/getToken |
| **Client ID** | Specify the client ID available on the **Client** page. |
| **Client Secret** | Specify the client secret available on the **Client** page. |
| **Scope (Optional)** | Specify the scope associated with the client. |
| **Grant Type** | Select **Authorization Code** or **Implicit**. |

5. Click **Request Token**.

   Integration Cloud login page appears.

6. Login to Integration Cloud with your credentials.

   An approval page appears. The approval page is an HTML page Integration Cloud sends to the resource owner, after a client submits a request for access to its private resources. The resource owner uses the page to accept or deny the request.

7. Select the scopes you want to grant access and then click **Approve**.

   On approval, an access token will be generated. A refresh token may also be generated depending on the **Refresh Count** configured for your client, and also if your grant type is **Authorization Code Grant**.

## Refreshing Access Tokens Using Refresh Tokens

You can refresh Access Tokens using Refresh Tokens.

**To refresh access tokens using Postman**

1. Add the Postman extension to your Google Chrome web browser.

2. Open the Postman application.

3. Make a HTTP POST call with the following details:

| Field | Description |
|---|---|
| **Post URL** | Provide the following URL:<br>https://abc.webmethodscloud.com/integration/rest/oAuth/getToken |
| **Query Parameters** | Provide the following query parameters:<br>*grant_type* - The Grant Type value will be refresh_token<br>*refresh_token* is the refresh token obtained while generating the tokens.<br>Example of an HTTP POST request for refreshing an access token:<br>https://abc.webmethodscloud.com/integration/rest/oAuth/getToken?grant_type=refresh_token&refresh_token=<*refresh_token_id*> |

**Note:**
In Postman, select **Basic Auth** as the **Authorization** type and specify Client ID and Client Secret as the **Username** and **Password** while refreshing the token.

4. An access token will be generated which can be used to invoke the service URLs. The Refresh Count value will decrease by 1.

## Managing Tokens

You can use this page to delete the *active tokens* issued by Integration Cloud. Client applications use these tokens to access the resources on Integration Cloud. When you delete the tokens, the client application can no longer access the resources owned by the resource owners. See the "Generating Tokens" on page 86 section on how to generate tokens (Access Token and Refresh Token) by using a REST Client. See "About OAuth 2.0" on page 79 for information on the high-level steps for configuring OAuth 2.0.

**Note:**
Users who have the **Access Control** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > User and Ownership Controls** can delete tokens.

For expired Access Tokens, the **Expiry Time** displays **Expired**. If the Refresh Count is also 0, then the row in the Token Management page is removed.

### To delete a token

1. From the Integration Cloud navigation bar, go to **Settings** > **OAuth 2.0** > **Token Management**.

2. Select a token and click **Delete**.

   When you delete a token from the list of active tokens, Integration Cloud deletes both the access token and the refresh token. To prevent a client from accessing resources, you can delete the client.

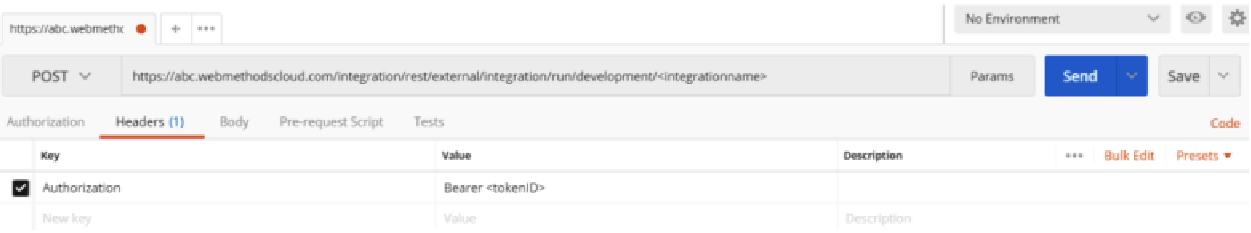
## Running Services Using OAuth 2.0

### To invoke service URLs using Postman

1. Add the Postman extension to your Google Chrome web browser.

2. Open the Postman application.

3. Type the **Request URL** and change the **HTTP method** to **POST**.



POST https://abc.webmethodscloud.com/integration/rest/external/integration/run/development/<integrationname>

| | Key | Value | Description |
|---|---|---|---|
| ☑ | Authorization | Bearer <tokenID> | |
| | New key | Value | Description |

4. On the **Headers** tab, add *Authorization* as the **Key** and *Bearer <token ID>* as the **Value**.

5. Click **Send** to run the Integration.

# Audit Log

**Audit Log** allows you to access logs related to additions, deletions, updations, export, schedule, skip, login, logout, password changes, record access attempts, access violations, deployments, restart Integration executions, resume Integration executions, and so on for a user.

To view the **Audit Log**, from the Integration Cloud navigation bar, click **Settings** > **Audit Log**.

> **Note:**
> The Audit Log page can be viewed only by administrators and users who have the **Manage Audit Log** permission under **Settings** > **Access Profiles** > **Administrative Permissions** > **Data Management Controls**.

By default, the **Audit Log** page displays the current day’s log entries, with the most recent entries listed on top. You can sort the log to view the latest log entries. You can also search the **Audit Log** for **User**, **Type**, or **Operation**.

**Activity Date** refers to the date and time when the event occurred. **User** refers to the name of the logged in user when the event occurred. **Type** refers to the type of log entry, for example, User, Login/Logout, Reference Data, Stage, Account, Application, Integration, License Agreement, Password Policy, Access Profile, Company, and so on. **Operation** refers to the action performed, for example, Export, Execute, Terminate, Add, Delete, Update, Login, Logout, and so on. **Description** refers to a summary of the action performed.

Click **Modify Retention Period** and specify the number of days to retain the Audit Log entries. Logs whose age exceeds the specified retention period are deleted.

Click **Download Audit Log** if you want to download and export log entries for a specified period. You can download Audit logs only up to 30 days.

# 4 Projects, Permissions, Sharing Assets

# Projects

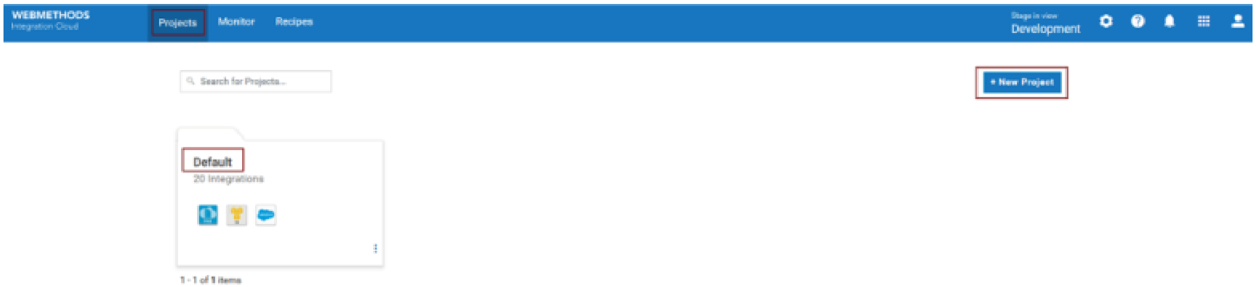
A project is an independent entity and corresponds to a folder for organizing your assets. A project holds all the assets created as a part of that project by the logged-in user, along with the configurations associated with the assets. Any asset, for example, integrations, REST APIs, SOAP APIs, Document Types, and Reference Data, is a part of a project.

If you are an *existing tenant*, your assets will be available in the *Default* project. You cannot delete this default project. If you are a *new tenant*, the *Default* project is not available and you need to create a new project.

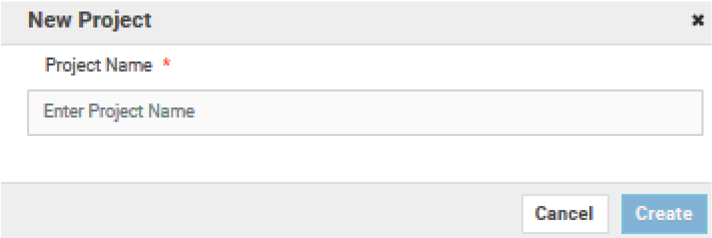
**Note:**
You must have the **Administrator** access profile and the **Developer** project permission assigned to create projects. Project permissions are used to associate permissions with projects. See "Project Permissions" on page 59 for information on how to assign permissions to projects.

### Creating projects

Once you log in, click **Projects > + New Project** to create a new project.



In the **New Project** dialog box that appears, enter a suitable name for the project that you want to create and then click **Create**. This will create a new project. You can now start creating integrations inside this project. You can create as many projects as you want.

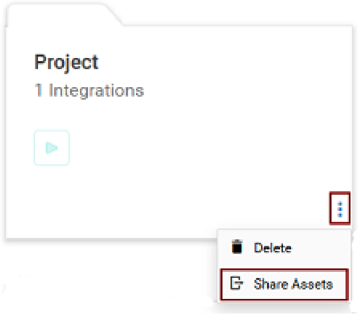


### Deleting projects

To delete a project, locate the project that you want to delete. Then click on the vertical ellipsis icon (or three tiny dots) at the lower-right corner of your project and click **Delete**. Confirm the delete action to permanently delete the project. You cannot delete the **Default** project.

### Sharing assets across projects

You can share integrations and document types across projects by clicking the vertical ellipses icon available on a project and selecting the **Share Assets** option. Click "here" on page 98 for more information.



### Asset Types specific to a project or is shared across all projects

The following table lists the asset types and identifies whether the asset type is specific to a project or is shared across all projects.

**Note:**
On-premises applications are stored globally when uploaded to Integration Cloud. They can be referenced by any project while creating integrations. The same global principle applies to predefined applications, predefined operations, and recipes. REST Applications, SOAP Applications, Flat File Applications and the Accounts and Operations created for the Applications are contained within a project.

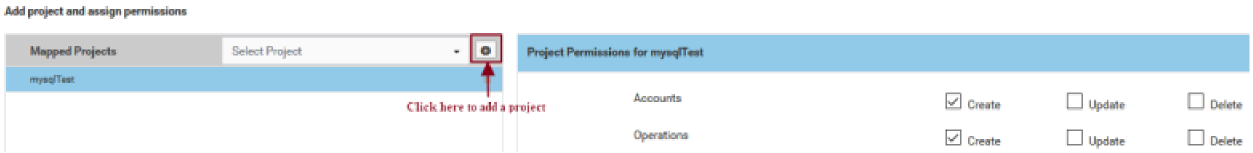
| Asset Type | Location | Remarks |
|---|---|---|
| **Predefined Applications** | Global | Shared across all projects |
| **REST Applications** | Project | Project specific |
| **On-Premises Applications** | Global | Shared across all projects |
| **SOAP Applications** | Project | Project specific |
| **Flat File Applications** | Project | Project specific |
| **Accounts** | Project | Project specific |
| **Operations - Custom** | Project | Project specific |
| **Operations - Predefined** | Global | Shared across all projects |
| **Keys & Certificates** | Project | Project specific |
| **SFTP Application** | Project | Project specific |
| **FTP Application** | Project | Project specific |
| **Integrations** | Project | Project specific |
| **Document types** | Project | Project specific |

| Asset Type | Location | Remarks |
|---|---|---|
| **REST APIs** | Project | Project specific |
| **SOAP APIs** | Project | Project specific |
| **Reference data** | Project | Project specific |
| **Recipes** | Global | Shared across all projects |

## Project Permissions

A project is an independent entity and corresponds to a folder for organizing your assets. A project holds all the assets created as a part of that project by the logged-in user, along with the configurations associated with the assets. Project permissions are used to associate permissions with projects.

You associate permissions with projects from the **Settings** ⚙ **> Project Permissions > Add New Project Permission** page.



Add project and assign permissions

Mapped Projects: Select Project

mysqlTest

Click here to add a project

Project Permissions for mysqlTest

Accounts ☑ Create ☐ Update ☐ Delete

Operations ☑ Create ☐ Update ☐ Delete

Integration Cloud provides a system-generated **Developer** project permission profile in your tenant. Any new project created is automatically associated with the **Developer** project permission profile. This profile has permissions to create, update, delete, and execute all assets. You cannot edit or delete this system-generated project permission profile. If a project permission profile is associated with a user on the user profile page, the user can perform only the permitted tasks in the mapped project.

**If you are an existing tenant:**

- Your user's **Access Profile** controls global permissions as well as permissions for the **Default** project.
- Your user's Project Permissions profile control permissions for other projects.
- All existing user profiles will be associated with the **Developer** project permission profile.
- All projects created have the **Developer** project permission profile associated with it.
- All existing assets will be available in the *Default* project.
- Manage the permissions for existing assets inside the **Default** project from the **Project Permissions for Default Project** section under **Settings** ⚙ **> Access Profiles > Administrative Permissions**.

- Manage the permissions for new assets created inside the **Default** project from the **Project Permissions for Default Project** section under **Settings** ⚙ **> Access Profiles > Administrative Permissions**.

- If you create a new project, you have to assign the project permissions from the **Settings** ⚙ **> Project Permissions > Add New Project Permission** page.

**If you are a new tenant:**

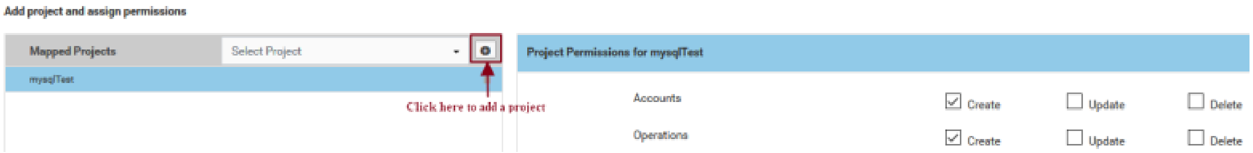
- The *Default* project is not available. There are no existing assets. Your user's **Access Profile** controls only global permissions.

- The Administrator will be associated with the **Developer** project permission profile and can assign a project permission profile to a new user.

- All projects created have the **Developer** project permission profile associated with it.

- If you create a new project, you have to assign the project permissions from the **Settings** ⚙ **> Project Permissions > Add New Project Permission** page.

Once you log in, from the Integration Cloud navigation bar, click **Settings** ⚙ **> Project Permissions > Add New Project Permission**. Select a project and then click the add icon ⊕ to add the project in the panel. Then assign the relevant permissions to the selected project . You can select another project and assign the permissions to the selected project. Click **Add**.



Add project and assign permissions

Mapped Projects | Select Project

mysqlTest

Click here to add a project

Project Permissions for mysqlTest

| | | | |
|---|---|---|---|
| Accounts | ☑ Create | ☐ Update | ☐ Delete |
| Operations | ☑ Create | ☐ Update | ☐ Delete |

**Note:**
Integration **Execute** permission available as part of **Project Permissions** are applicable only for top-level projects. For example, if you have the Integration **Execute** permission for project B but not for project A, and have shared assets of project A with project B, then when you execute an integration of project B which has an integration of project A, the integration execution will be successful.

**Note:**
If you associate the project permission profile to a user on the user profile page, the user can perform only the permitted tasks in each mapped project.

### Example 1

A user is assigned to two project permission profiles A and B in the user profile. Project permission profile A has the Execute Integration permission in Project X but Project permission profile B does not have the Execute Integration permission for the same Project X. The user will still be able to execute the Integration.

### Example 2

In this example, we will see how you will allow an existing user U1 to create, update, delete, and execute Integrations for the Project P1 but not for Project P2.

1. You must have the **Access Control** permission under **Access Profiles > Administrative Permissions**.
2. Create Project P1 and Project P2.
3. Create a new Project Permission profile PP1 under **Settings > Project Permissions > Add New Project Permission**.
4. On the **Add New Project Permission** page, add Project P1 in the mapped projects list. Assign the create, update, delete, and execute Integration permissions to the mapped Project P1.
5. Do not map Project P2.
6. Go to the **Basic** tab of user U1 and select *only* PP1.

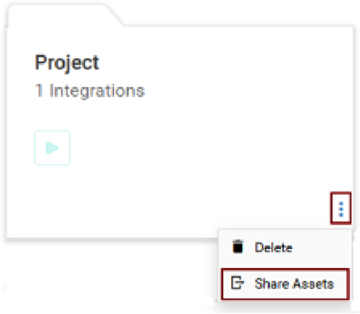
So when U1 logs in, U1 will only be able to create, update, delete, and execute Integrations for the Project P1 but not for Project P2.

## Sharing Assets across Projects

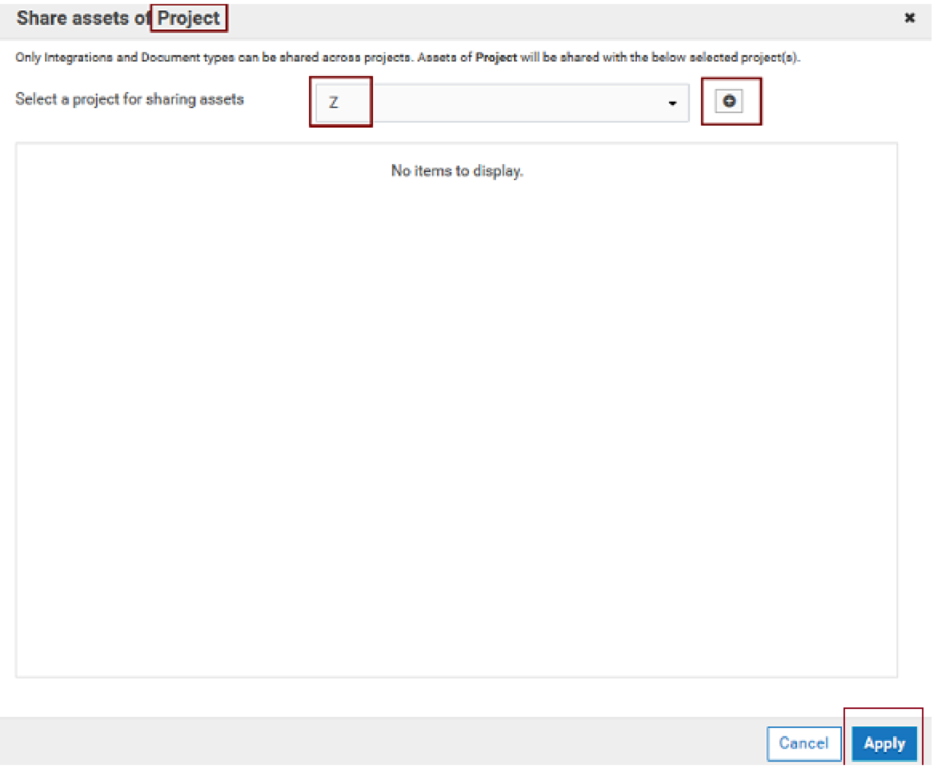
You can share integrations and document types available in one project with other projects from the **Projects** page. Sharing of assets allows you to use the same assets in other projects.

**Note:**
Sharing assets with other projects is allowed only in the **Development** stage. If an integration in a shared project has applications, then you can configure the account only in that shared project. You must have the **Administrator** access profile privileges to share assets.

You can share assets available in a project by clicking the vertical ellipses icon on a project and then clicking the **Share Assets** option as shown.



In the **Share Assets of *project name*** dialog box, after you select the project **Z** and add the project, the assets of **Project** is shared with **Z** after you click **Apply**.



Share assets of Project

Only Integrations and Document types can be shared across projects. Assets of **Project** will be shared with the below selected project(s).

Select a project for sharing assets: Z

No items to display.

- While creating an integration, the **Integrations** category available on the left panel lists all the shared projects and the integrations available in the shared projects along with the integrations available in the same project.

- While creating an integration and defining the input and output signature by clicking the **I/O** icon, you can select the shared project in the **Project** drop-down list in the **Define Input and Output Signature** dialog box. All document types available in the shared project appear in the **Document Reference** drop-down list.

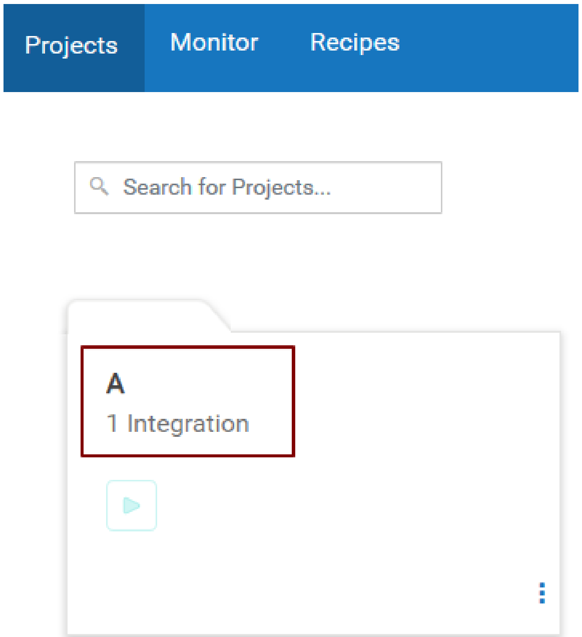
- While adding a new field with **Type** as Document Reference, in the **Document Reference** dialog box, you can select the shared project.

- While creating a new document type from scratch, in the **Field Properties** panel, if you select the **Type** as **Document Reference**, then in the **Document Reference** dialog box you can select the shared project.

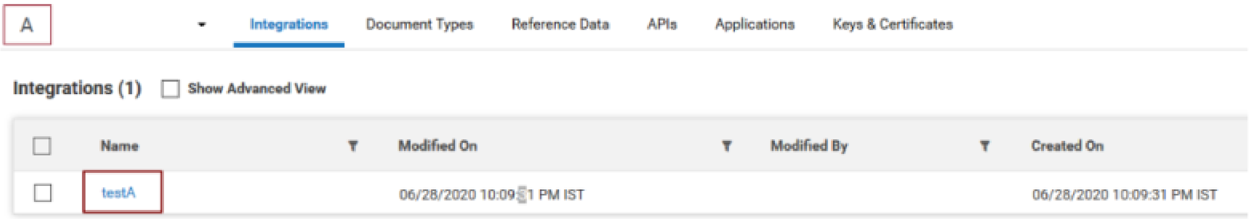
- While mapping fields in the pipeline, in the **Field Properties** panel, if you select the **Type** as **Document Reference**, then in the **Document Reference** dialog box you can select the shared project.

Let us see how asset sharing works and its behavior with the help of an example. In this example, we will *share assets of project A with project B*.

1. In the **Development** stage, go to the **Projects** page. Project **A** has only one integration.

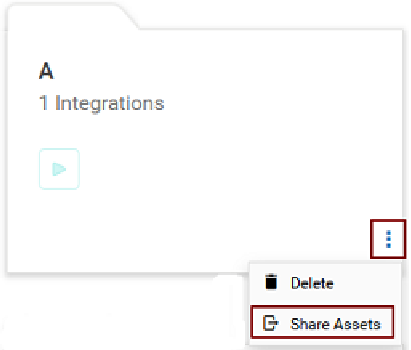


The name of the integration in project **A** is **testA**.

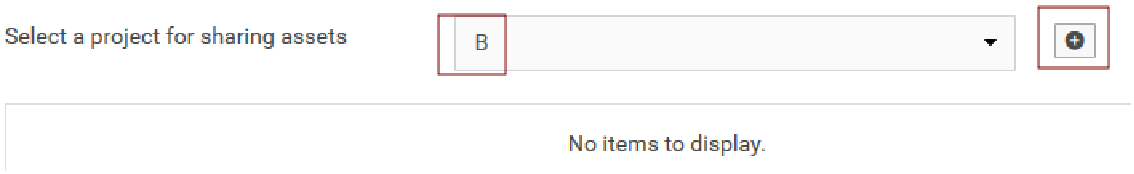


| □ | Name | Modified On | Modified By | Created On |
|---|---|---|---|---|
| □ | testA | 06/28/2020 10:09:31 PM IST | | 06/28/2020 10:09:31 PM IST |

2. If you want to share **testA** with project **B**, click the vertical ellipses icon on project **A**, and then click **Share Assets**.



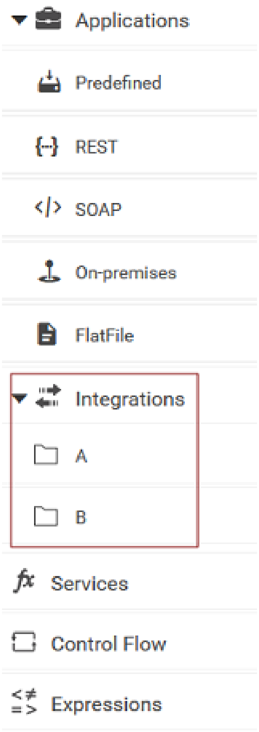
3. On the **Share assets of A** dialog box, select project **B** from the drop-down menu, click the add icon ⊕, and then click **Apply**.



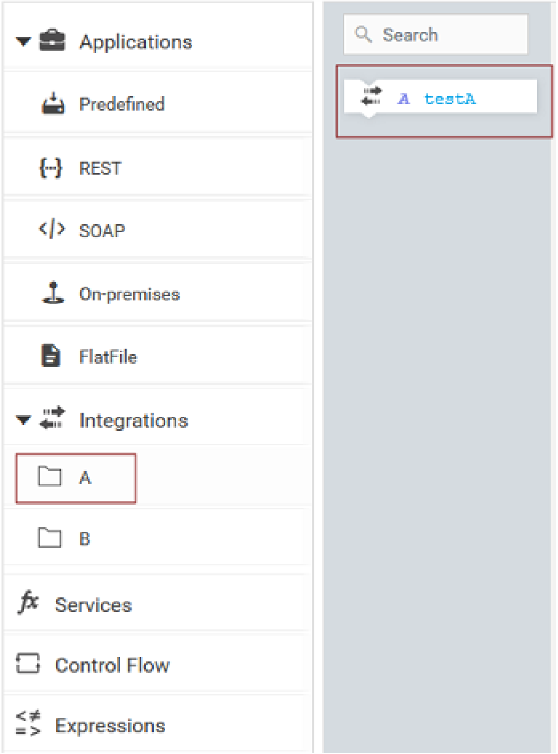
Select a project for sharing assets B

No items to display.

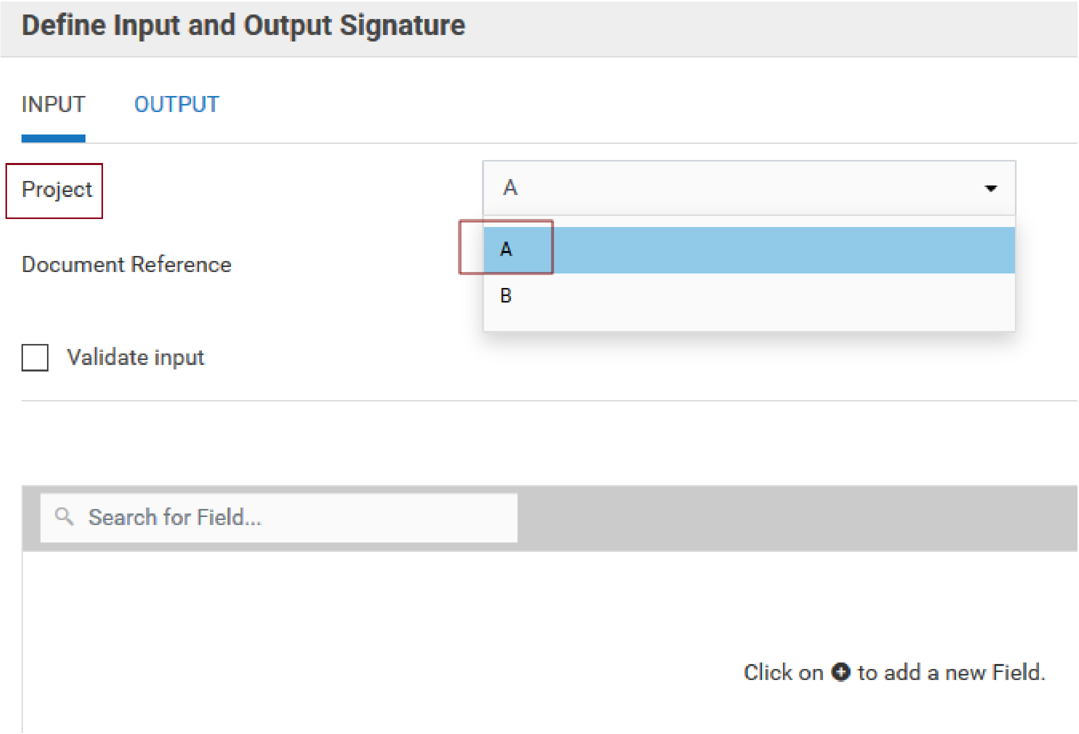
4. After you click **Apply**, go to project **B** and create an integration in project **B**. Project **A** appears under the **Integrations** category on the left panel along with its shared asset **testA**.



The display format is *project name integration name* A testA.



5. While creating an integration in project **B**, to define the input and output signature, click the **I/O** icon and in the **Define Input and Output Signature** dialog box, select project **A**. All document types available in project **A** appear in the **Document Reference** drop-down menu.


Define Input and Output Signature

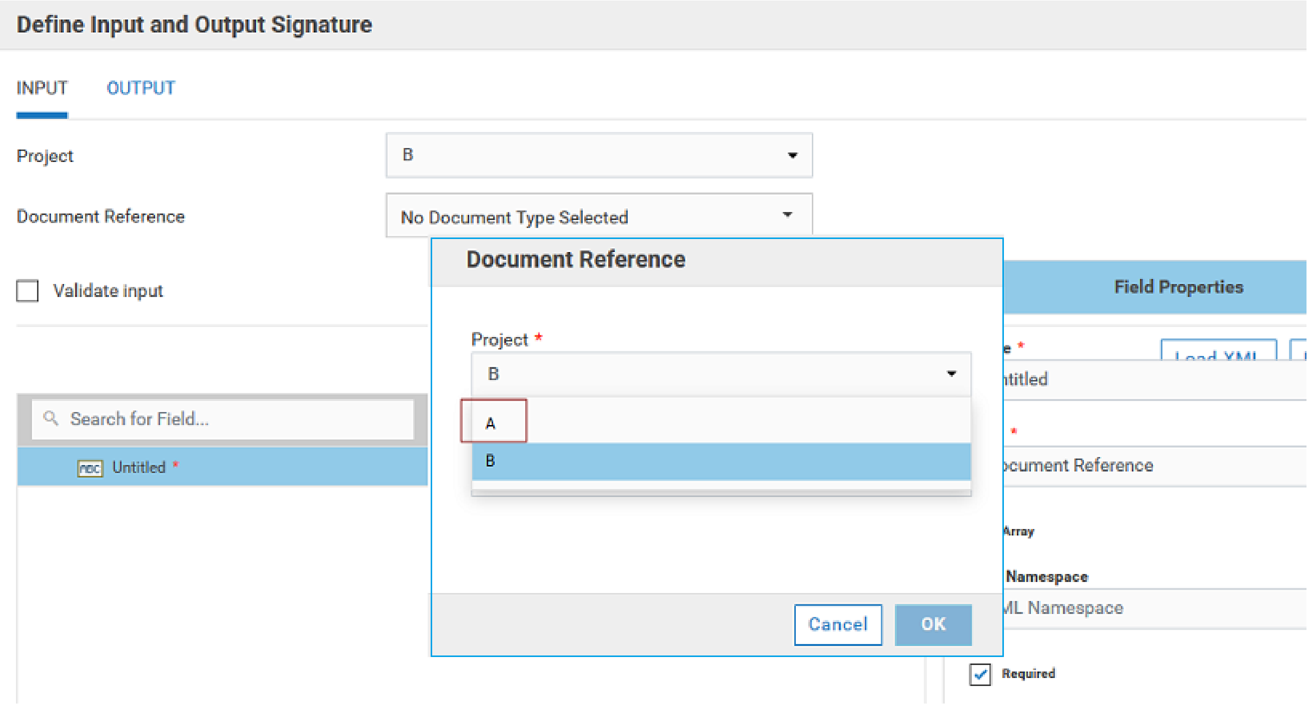
INPUT OUTPUT

Project: A

Document Reference

□ Validate input

6. While adding a new field, select the **Type** as **Document Reference**. Select project **A**. All document types available in project **A** appear in the **Document Reference** drop-down menu.



Define Input and Output Signature

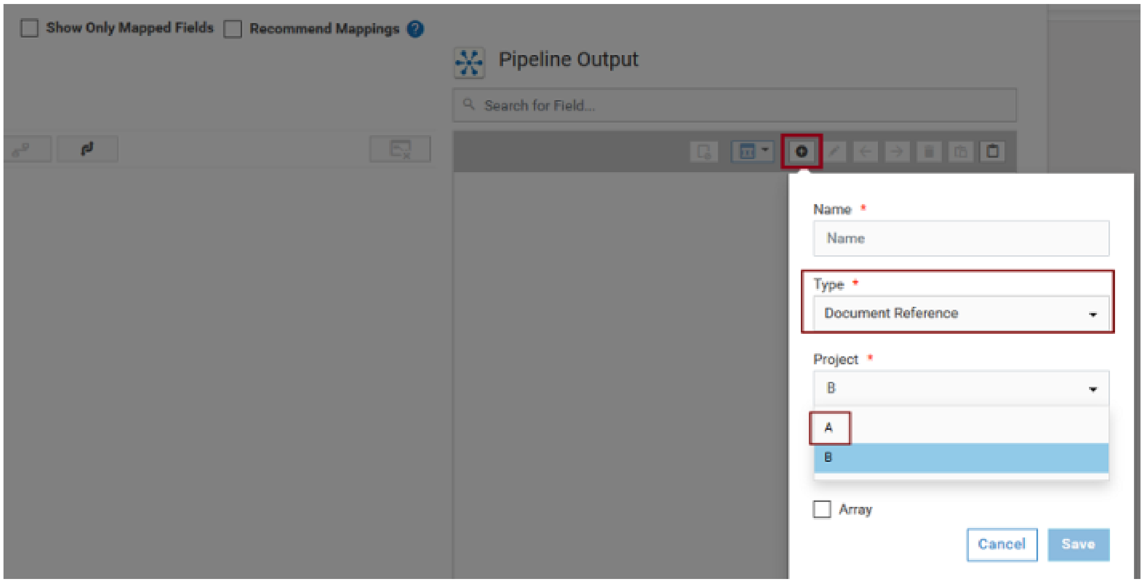
Project: B

Document Reference: No Document Type Selected

Document Reference

Project *: B

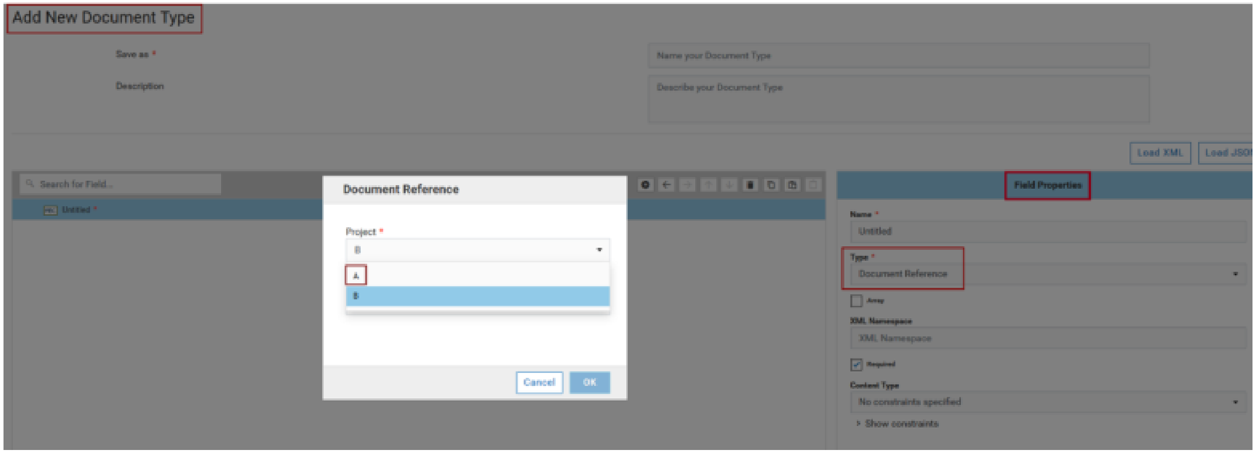
7. While mapping fields in the pipeline, add a new field. In the **Field Properties** panel, select the **Type** as **Document Reference**. Then in the **Document Reference** dialog box, select the shared project **A**.



Type *: Document Reference

Project *: B

8. While creating a *new document type from scratch*, in the **Field Properties** panel, if you select the **Type** as **Document Reference**, then in the **Document Reference** dialog box, select project **A**.



Add New Document Type

Save as * Name your Document Type

Description Describe your Document Type

Document Reference

Project * B

A

B

Field Properties

Name * Untitled

Type * Document Reference

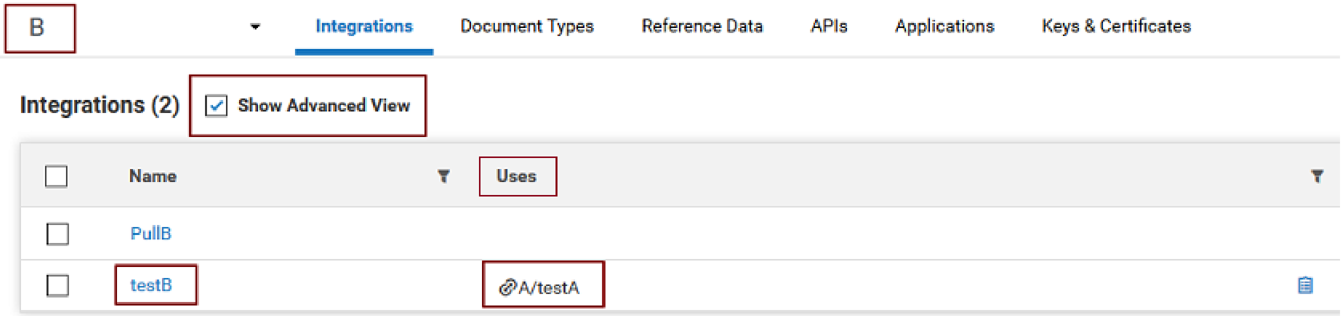
☐ Array

XML Namespace XML Namespace

☑ Required

Content Type No constraints specified

9. Project **B** has an integration **testB**. Click the **Show Advanced View** option to view the **Uses** column. The **Uses** column displays the references in the format *project name/referenced asset name* that are used to create the integration.

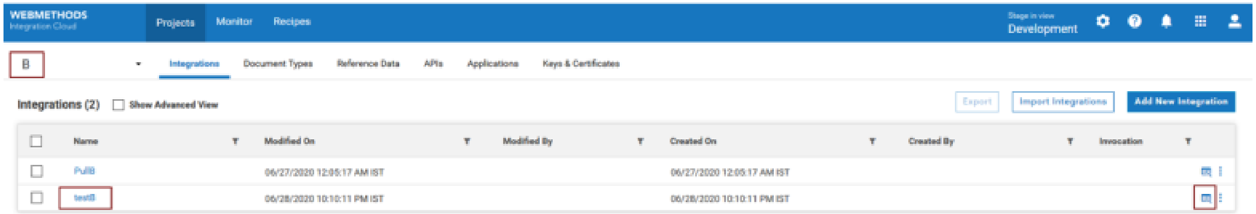


B

Integrations (2) ☑ Show Advanced View

| ☐ | Name | Uses |
|---|---|---|
| ☐ | PullB | |
| ☐ | testB | A/testA |

Click the 🔍 icon to view the **Overview** page.

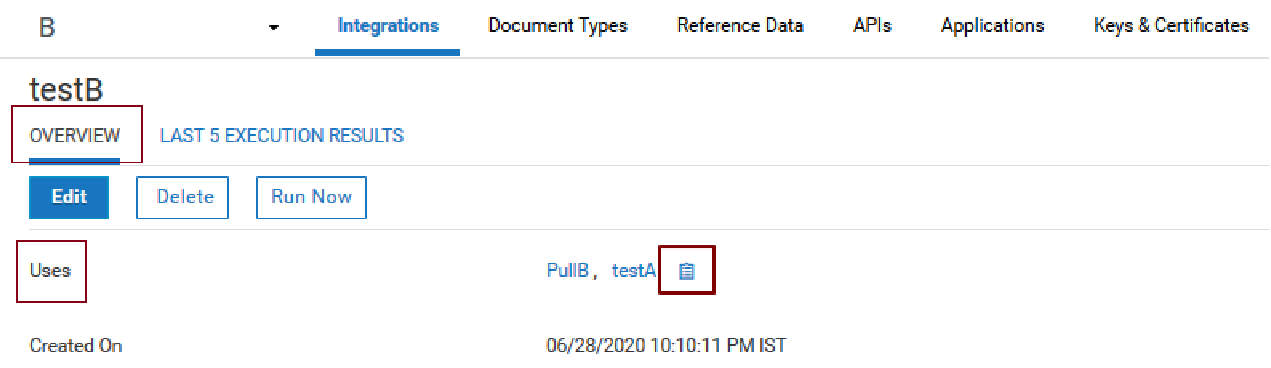


B

Integrations (2) ☐ Show Advanced View

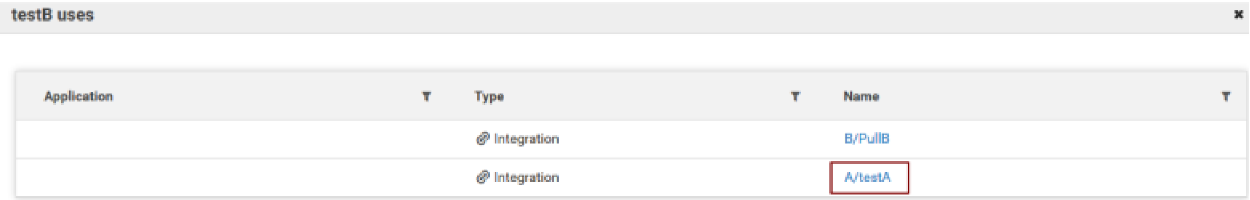
| ☐ | Name | Modified On | Modified By | Created On | Created By | Invocation |
|---|---|---|---|---|---|---|
| ☐ | PullB | 06/27/2020 12:05:17 AM IST | | 06/27/2020 12:05:17 AM IST | | |
| ☐ | testB | 06/28/2020 10:10:11 PM IST | | 06/28/2020 10:10:11 PM IST | | |

On the **Overview** page, click the **Uses** icon 📋.



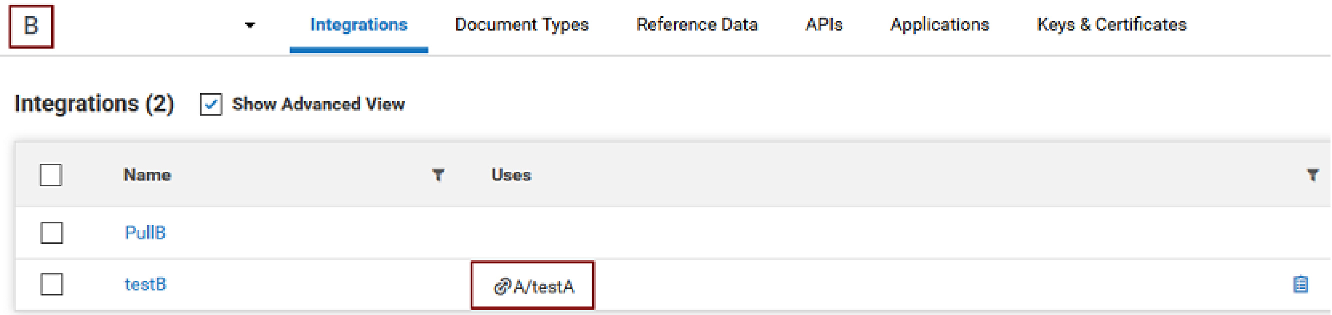
| | |
|---|---|
| Uses | PullB , testA |
| Created On | 06/28/2020 10:10:11 PM IST |

The **Uses** window displays the shared project **A** and the integration **testA** in the *project name/integration name* format.


testB uses

| Application | Type | Name |
|---|---|---|
| | Integration | B/PullB |
| | Integration | A/testA |

Also, on the integrations list page, click **Show Advanced View** to view the **Uses** column, which displays the references that are used to create the integration in the format *project name/referenced asset name*.



| □ | Name | Uses |
|---|---|---|
| □ | PullB | |
| □ | testB | A/testA |

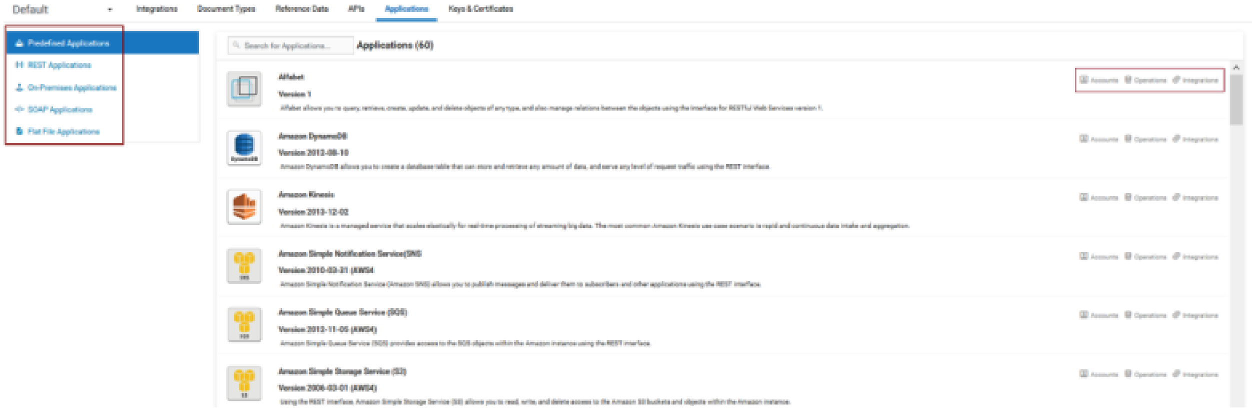
10. If you use integration **testA** of project **A** in the integration **testB** of project **B**, and if you deploy the integration **testB** to another stage, then project **A** along with integration **testA** is deployed to that stage.

> **Note:**
> Integration **Execute** permission available as part of **Project Permissions** (**Settings > Project Permissions**) are applicable only for top-level projects. For example, if you have the integration **Execute** permission for project **B** but not for project **A**, but have shared assets of project **A** with project **B**, then when you execute an integration of project **B** which has an integration of project **A**, the integration execution will still be successful.

# 5 Applications

# Overview

Integration Cloud allows you to create and govern Integrations between Software as a Service (SaaS) or on-premises applications. A set of predefined and configurable Applications are provided, for example, Salesforce, StrikeIron, ServiceNow, and so on. Applications allow you to connect to the particular SaaS providers.



Alfabet
Version 1
Alfabet allows you to query, retrieve, create, update, and delete objects of any type, and also manage relations between the objects using the interface for RESTful Web Services version 1.

Amazon DynamoDB
Version 2012-08-10
Amazon DynamoDB allows you to create a database table that can store and retrieve any amount of data, and serve any level of request traffic using the REST interface.

Amazon Kinesis
Version 2013-12-02
Amazon Kinesis is a managed service that scales elastically for real-time processing of streaming big data. The most common Amazon Kinesis use case scenario is rapid and continuous data intake and aggregation.

Amazon Simple Notification Service(SNS
Version 2010-03-31 (AWS4
Amazon Simple Notification Service (Amazon SNS) allows you to publish messages and deliver them to subscribers and other applications using the REST interface.

Amazon Simple Queue Service (SQS)
Version 2012-11-05 (AWS4)
Amazon Simple Queue Service (SQS) provides access to the SQS objects within the Amazon instance using the REST interface.

Amazon Simple Storage Service (S3)
Version 2006-03-01 (AWS4)
Using the REST interface, Amazon Simple Storage Service (S3) allows you to read, write, and delete access to the Amazon S3 buckets and objects within the Amazon instance.

You create REST, SOAP, Flat File, FTP/FTPS, SFTP, and SMTP Applications from this page. To create a REST Application, click **Projects > <Select a Project> > Applications > REST Applications > Add New Application**.To create a SOAP Application, click **Projects > <Select a Project> > Applications > SOAP Applications > Add New Application**. To create a Flat File Application, click **Projects > <Select a Project> > Applications > Flat File Applications > Add New Application**. FTP/FTPS, SFTP, and SMTP Applications are available under Predefined Applications that allow Integration Cloud to connect to FTP/FTPS, SFTP, and SMTP servers.

On-Premises Applications loaded from on-premises systems are also listed in the **Applications > On-Premises Applications** page but you will not be able to create Accounts or Operations for on-premises Applications. Those can be uploaded only from webMethods Integration Server. Further, when you upload services as part of an Application from on-premises webMethods Integration Server to webMethods Integration Cloud, the comments field of the service is uploaded and displayed in the webMethods Integration Cloud Application. This field will be displayed if present and cannot be edited. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for more information.

From the **Application category** page, you can create Accounts and Operations for an Application and Integrations between different SaaS applications. For an Application, click **Accounts**, **Operations**, or **Integrations** if you want to create or edit them for that Application. For REST Applications, the **Documents Types** link appears and allows you to create new Document Types. Document Types created for a REST Application appear only in the **Document Types** panel for the selected REST Application.

If you have the required access privileges under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Application**, you can upgrade Application assets (Accounts, Operations, and the associated Integrations) from a lower version to a higher version by clicking the **Upgrade** button.

To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

**Note:**
Users who have the required project permissions under **Settings** ⚙ **> Project Permissions** can create, update, delete and execute the Integrations, Accounts, Operations, Reference Data, Document Type, and Listeners information.

## Predefined Applications

A set of predefined and configurable Applications are provided, for example, Salesforce, StrikeIron, ServiceNow, and so on. These Applications allow you to connect to the particular SaaS providers.

**Note:**
It is recommended to use secured protocols such as HTTPS and FTPS for securing the data transmitted over the network.

### Applicability Statement 2 (AS2)

Applicability Statement 2 (AS2) is a communication protocol developed by the Internet Engineering Task Force (IETF) for the exchange of business-to-business (B2B) transactions over the Internet securely. The AS2 application uses the HTTP transport protocol along with Multipurpose Internet Mail Extensions (MIME). The AS2 application governs the means of connection and exchange of data securely and reliably. Besides the advanced security features, the AS2 application offers the following additional benefits:

- Privacy
- Authentication
- Nonrepudiation of origin and receipt of the message
- Data integrity

The AS2 application provides a medium to exchange business data with partners by configuring an account in Integration Cloud. The application supports the AS2 protocol versions 1.1 and 1.2.

| Field | Description |
|---|---|
| **Recipient Endpoint** | The endpoint URL of the recipient. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. You can choose one of the following options:<br>- **none**: No additional authorization scheme will be executed at run time. For example, when you specify a user name and password, but do not specify a value for the authorization type, the user credentials are not inserted into an authorization header. |

| Field | Description |
|---|---|
| | - **basic**: When the application requires or supports HTTP basic authentication for user name and password. |
| **From** | The AS2 ID of the sender. |
| **To** | The AS2 ID of the recipient. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration from the list. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. Select **Add New Truststore** to add a new trust store from this list. |
| **Keystore Alias** | Select the alias for the Integration Cloud keystore configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. Select **Add New Keystore** to add a new keystore from this list. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the **Keystore Alias** field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both, **Keystore Alias** and **Client Key Alias** fields. |
| **Hostname verifier** | Select a hostname verifier implementation for guards against man-in-the-middle (MITM) attacks from the list. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier. This enables hostname verification. Select **org.apache.http.conn.ssl.NoopHostnameVerifier** from the list to disable hostname verification. |

| Field | Description |
|---|---|
| **Username** | The name of the user account that the AS2 connection will use to connect to the AS2 provider. |
| **Password** | The password for the user name provided in the **Username** field. |
| **Compression** | Select this option to compress an outbound AS2 message. |
| **Sign Message** | Select this option to sign an outbound AS2 message. |
| **Signing Algorithm** | The signing algorithm to use for an outbound AS2 message. The available options are:<br>- MD5<br>- SHA-1<br>- SHA-256<br>- SHA-384<br>- SHA-512 |
| **Signing Keystore and Key Aliases** | The keystore aliases and the key aliases in the keystore to use for signing an outbound AS2 message. |
| **Receive Signed Message** | Select this option to receive a signed inbound AS2 message. If you select this option and the incoming AS2 message is not signed, then an `Insufficient message security` error is encountered and shared with the sender if MDN is requested by the sender. |
| **Signature Verification Certificate** | The certificate to use for verifying an inbound signed AS2 message. |
| **Encrypt Message** | Select this option to encrypt an outbound AS2 message. |
| **Encryption Algorithm** | The encryption algorithm to use for an outbound AS2 message. The available options are:<br>- RC2 40<br>- RC2 64<br>- RC2 128<br>- DES<br>- TripleDES<br>- AES 128<br>- AES 192 |

| Field | Description |
|---|---|
| | ■ AES 256 |
| **Encryption Certificate** | The certificate to use for encrypting an outbound AS2 message. |
| **Receive Encrypted Message** | Select this option to receive an encrypted inbound AS2 message. If you select this option and the incoming AS2 message is not encrypted, then an `Insufficient message security` error is encountered and shared with the sender if MDN is requested by the sender. |
| **Decryption Keystore and Key Aliases** | The keystore aliases the key aliases in the keystore to use for decrypting an inbound AS2 message. |
| **Request MDN** | Whether you want the recipient to return an MDN to the sender.<br>You can select one of the following options:<br>■ **None**: The recipient of the AS2 message does not return an MDN to the sender.<br>■ **Synchronous**: The recipient of the AS2 message returns an MDN to the sender through the same HTTP connection used to send the original AS2 message.<br>■ **Asynchronous**: The recipient of the AS2 message returns an MDN to the sender through a different HTTP connection instead of the one used to send the original AS2 message. |
| **Request Signed MDN** | Select this option if you want the recipient to sign an AS2 MDN.<br>Ensure that you also select an option in the **Request MDN** field if you want the recipient to sign and return an AS2 MDN. |
| **Asynchronous MDN Endpoint** | Type your endpoint URL that accepts an inbound AS2 MDN if you selected the **Asynchronous** option for **Request MDN**. |
| **AS2 Version** | Select the AS2 protocol version to use from the list. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server** |

| Field | Description |
|---|---|
| | **URL** field, specify the host name value in the **SNI Server Name** field. |

## AS2 Predefined Operations

The following predefined Applicability Statement 2 (AS2) operations are available:

### receive

Receives an AS2 message from a recipient.

You can perform the following configurations in the AS2 application using the receive service.

- "Configuring the Auto Detect Option" on page 117
- "Creating an Endpoint URL" on page 117

### Input Parameters

*contentStream* **Object** Receives an AS2 message of content type other than `application/xml`.

*node* **Object** Optional. Receives an AS2 message of content type `application/xml` only.

### Output Parameters

*status* **String** Status of an inbound message.

*statusMessage* **String** Processing status of an inbound message.

*request* **Document** Receives the raw stream, extracted payload, and attachments of an inbound message.

- *stream* **Object** Raw output stream received by an application.
- *headers* **Document** AS2 message headers.
  - *AS2-To* **String** AS2 ID of the recipient.
  - *AS2-From* **String** AS2 ID of the sender.
  - *Message-ID* **String** Message ID of the inbound message.
  - *AS2-Version* **String** AS2 protocol version used for the inbound message.

        - *Content-Type* **String** MIME content type of the inbound message.
        - *EDIINT-Features* **String** Optional features supported by the application.
        - *Receipt-Delivery-Option* **String** Optional. Sender's asynchronous MDN endpoint URL.
        - *Disposition-Notification-To* **String** Optional. Acknowledgment request for the inbound message.
        - *Disposition-Notification-Options* **String** Optional. Acknowledgment request to be signed for the inbound message.
    - *payload* **Document** Extracted payload that you receive.
        - *stream* **Object** Extracted payload stream.
        - *contentType* **String** Content type assigned to the payload.
        - *headers* **Document** Headers assigned to the payload.
    - *attachments* **Document Array** Optional. Attachments you receive with an inbound message, if any.
        - *stream* **Object** Output stream of the attachment.
        - *contentType* **String** Content type assigned to the attachment.
        - *headers* **Document** Headers assigned to the attachment.

- *response* **Document** Sent MDN or received asynchronous MDN response.
    - *status* **String** Status of the sent or received MDN.
    - *statusMessage* **String** Status message of the sent or received MDN.
    - *receipt* **Document** Optional. Sent or received MDN.
        - *stream* **Object** Object stream of the sent or received MDN.
        - *headers* **Document** Headers of the sent or received MDN.
            - *AS2-To* **String** AS2 ID of the recipient.
            - *AS2-From* **String** AS2 ID of the sender.

| | |
|---|---|
| *Message-ID* | **String** Message ID of the inbound or outbound MDN. |
| *AS2-Version* | **String** AS2 protocol version used for the inbound or outbound MDN. |
| *Content-Type* | **String** MIME content type of the inbound or outbound message. |

### send

Sends an AS2 message to a recipient's defined endpoint.

## Input Parameters

| | | |
|---|---|---|
| *data* | **Document** Payload you want to send. | |
| | *stream* | **Object** java.io.InputStream object you want map from EDI or XML data. |
| | *contentType* | **String** Content type to assign to an outbound message. The following options are available by default:<br>■ application/edi-x12<br>■ application/edifact<br>■ application/xml<br>You can also type a custom value. |
| | *otherHeaders* | **Document** Optional. *key* and *value* strings of the header for an outbound message. |
| *attachments* | **Document Array** Optional. Attachments for a message, if any. | |
| | *stream* | **Object** java.io.InputStream object you want add to the attachment. |
| | *contentType* | **String** Content type of the attachment. For example, `application/zip` if the attachment is a `.zip` file. |
| | *otherHeaders* | **Document** Optional. *key* and *value* strings of the header you want to add to the attachment. |
| *customHeaders* | **Document** Optional. Custom headers you want to include in an AS2 message. | |

    - *key* **String** Key for the custom header.
    - *value* **String** Value for the customer header.

## Output Parameters

- *status* **String** Status of an outbound message.
- *statusMessage* **String** Processing status of an outbound message.
- *request* **Document** AS2 message sent to a recipient.
    - *stream* **Object** AS2 message stream.
    - *headers* **Document** Optional. AS2 message headers.
        - *AS2-To* **String** AS2 ID of the recipient.
        - *AS2-From* **String** AS2 ID of the sender.
        - *Message-ID* **String** Message ID of the outbound message.
        - *AS2-Version* **String** AS2 protocol version used for the outbound message.
        - *Content-Type* **String** MIME content type of the outbound message.
        - *EDIINT-Features* **String** Optional features supported by the application.
        - *Receipt-Delivery-Option* **String** Optional. Recipient's asynchronous MDN endpoint URL.
        - *Disposition-Notification-To* **String** Optional. Acknowledgment request for the outbound message.
        - *Disposition-Notification-Options* **String** Optional. Acknowledgment request to be signed for the outbound message.
- *response* **Document** Received MDN response.
    - *status* **String** Status of the received MDN.
    - *statusMessage* **String** Status message of the received MDN.
    - *receipt* **Document** Optional. Received MDN.
        - *stream* **Object** Object stream of the received MDN.

| | | |
|---|---|---|
| *headers* | **Document** Optional. Headers of the received MDN. | |
| | *AS2-To* | **String** AS2 ID of the recipient. |
| | *AS2-From* | **String** AS2 ID of the sender. |
| | *Message-ID* | **String** Message ID of the inbound MDN. |
| | *AS2-Version* | **String** AS2 protocol version used for the inbound MDN. |
| | *Content-Type* | **String** MIME content type of the inbound message. |

## Configuring the Auto Detect Option

You can select the **Auto Detect** option for the AS2 application to automatically identify an account based on the **AS2-From** and **AS2-To** headers of an inbound message.

This option enables the AS2 application to compare an account configured with **From** and **To** fields with the **AS2-From** and **AS2-To** headers of an inbound message and vice versa. In addition, specifying this option allows the use of an individual endpoint URL with multiple partners.

**Note:**
**Auto Detect** option is supported only for the **receive** operation.

**Important:**
Configuring multiple accounts with identical values for the **From** and **To** fields might generate unpredictable results. This happens when the application uses the account that matches the first **AS2-From** and **AS2-To** headers of an inbound message. Therefore, if you have multiple accounts configured with identical values for the **From** and **To** fields, then do not select the **Auto Detect** option.

## Creating an Endpoint URL

A sender requires a recipient's endpoint URL to transfer AS2 messages. You must create an endpoint URL and share it with your partner to send AS2 messages to the endpoint URL.

### To create an endpoint URL

1. Create an orchestrated integration. For instructions, see "Orchestrated Integrations" on page 387. Ensure that you specify a signature with `contentStream` and *`node`* as input parameters of type

Object. For instructions, see "Pipeline and Signatures" on page 400. Ensure you specify a name for the integration.

Alternatively, you can define a Document Type as a signature with *contentStream* and *node* as input parameters of type **Object**. For instructions, see "Creating Document Types from Scratch" on page 637.

2. Configure the AS2 application with the **receive** operation to work with a new or existing account, or select the **Auto Detect** option for the application. For information about accounts and configuring an account using the **Auto Detect** option, see "Adding or Editing Accounts" on page 362 and "Configuring the Auto Detect Option" on page 117 respectively.

3. Map the `contentStream` and `node` parameters of the Pipeline Input signature defined in step 1 with the AS2 application's `receiveInput` parameter. For more information, see "Pipeline and Signatures" on page 400.

4. Select the **Enable Integration to be invoked over HTTP** option for the integration. For more information, see "Integration Details" on page 408.

   An endpoint URL for this integration is generated. Share this endpoint URL with your partner to enable the partner to send AS2 messages.

## Alfabet

Integration Cloud connects to Alfabet using the Interface for RESTful Web Services and supports working with the various object types as defined in Alfabet. You can use it to query, retrieve, create, update, and delete objects of any type, and also manage relations between the objects.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the Account configuration. The URL depends on where the required instance of Alfabet is installed. It is possible to either include or omit the endpoint suffix "/Alfabet/api/vXX" in the URL. For example, both these options are equivalent:<br>■ https://myalfabet.com<br>■ https://myalfabet.com/Alfabet/api/v1 |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Authorization Token** | The Alfabet Authorization Token as defined in the web.config file of the Alfabet Web Application on the server side, under the <alfaSection> element.<br>**Note:**<br>See the *Authorization* chapter in the Alfabet Interface for RESTful Web Services reference manual for required configurations in the server side for Alfabet and for details about the different authorization modes. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |

| Field | Description |
|---|---|
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Element Character Set** | The encoding to use for the HTTP message components, such as request line and headers. |
| **Wait For Continue Time** | The number of milliseconds that the client connection should wait for a 100 Continue response from the server when the Expect/Continue handshake is used. |
| **Strict Transfer Encoding** | Whether the connection validates the HTTP Transfer Encoding header.<br>Valid values:<br>true: The connection validates the Transfer Encoding header and returns an error when the header is invalid.<br>false: The connection does not validate the Transfer Encoding header. |

## Amazon DynamoDB

Integration Cloud connects to Amazon DynamoDB using the REST interface and allows you to create a database table that can store and retrieve any amount of data, and serve any level of request traffic.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://dynamodb.<instance>.amazonaws.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a |

| Field | Description |
|---|---|
| | value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Access Key** | This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key. |
| **Secret Key** | This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots. |
| **Region** | An area specific value. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in |

| Field | Description |
|---|---|
| | the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Amazon Kinesis

Amazon Kinesis is a managed service that scales elastically for real-time processing of streaming big data. The most common Amazon Kinesis use case scenario is rapid and continuous data intake and aggregation.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://kinesis.<Region>.amazonaws.com. |
| **Access Key** | This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key. |
| **Secret Key** | This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Region** | An area specific value. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Amazon Simple Storage Service (S3)

Integration Cloud connects to Amazon Simple Storage Service (S3) using the REST interface and provides read, write, and delete access to the Amazon S3 buckets and objects within the Amazon instance.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://s3.amazonaws.com/. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Use Stale Checking** | If enabled, additional processing is performed to test if the socket is still functional each time the socket is used. |
| **Validate After Inactivity** | This field is used in conjunction with the **Use Stale Checking** field to control the period of inactivity after which persistent connections must be revalidated prior to being leased. This field is considered only if the **Use Stale Checking** field is enabled, else this field is ignored. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |

| Field | Description |
|---|---|
| **Use Expect Continue** | Whether to use the Expect/Continue HTTP/1.1 handshake and send the Expect request header. When the client sends the Expect request header, the client waits for the server to confirm that it will accept the request, before the client sends the request body. Enable this option to use the Expect/Continue handshake. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Access Key** | This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key. |
| **Secret Key** | This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots. |
| **Region** | An area specific value. |
| **Signing Algorithm** | Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Amazon Simple Notification Service(SNS)

Integration Cloud connects to Amazon Simple Notification Service (Amazon SNS) using the REST interface and allows you to publish messages and deliver them to subscribers and other applications.

| Field | Description |
|---|---|
| **Server URL** | The endpoint to connect with AWS SNS. Prefix the endpoint with https://, for example, https://sns.(Region).amazonaws.com. This is the native provider endpoint target for the Account configuration. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Access Key** | Access Key obtained from AWS Identity and Access Management (IAM) Console. This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key. |
| **Secret Key** | Secret key obtained from AWS Identity and Access Management (IAM) Console. This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots. |
| **Region** | An area specific value. The region is different for different users. |

| Field | Description |
|---|---|
| **Signing Algorithm** | Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Amazon Simple Queue Service (SQS)

Integration Cloud connects to Amazon Simple Queue Service (SQS) using the REST interface and provides access to the SQS objects within the Amazon instance.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://sqs.us-east-1.amazonaws.com/. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |

| Field | Description |
|---|---|
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Access Key** | This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key. |
| **Secret Key** | This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots. |
| **Region** | An area specific value. |
| **Signing Algorithm** | Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.<br>**Note:**<br>This field is not applicable for Amazon SQS Version 4. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is |

| Field | Description |
|---|---|
| | other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## CloudStreams Connector for Anaplan®

Using the REST interface, CloudStreams Connector for Anaplan® allows you to interact with data in your models and securely upload files, download files, import and export data, and run actions programmatically.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: http://host:port.<br>Replace < host:port > with your actual JIRA instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | The name of the user account on the SaaS provider that the connection will use to connect to the SaaS provider. |
| **Password** | The password for the user name provided in the Username field. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header.<br>If you enter the username and password, then set the authorization type as **basic** . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic |

| Field | Description |
|---|---|
| | authentication using a username and password. Select the Authorization Type as **basic**. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Apache Solr Search

Solr is an open source enterprise search platform built on Apache Lucene. Solr is a standalone enterprise search server with a REST-like API. You can place documents in it (called "indexing") using JSON, XML, CSV, or binary over HTTP. You can query it using HTTP GET and receive JSON, XML, CSV, or binary results. Integration Cloud connects to Apache Solr using the REST API Version 6.1 and allows you to execute search operations over the indexed data.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: https://<hostName>.<br>Replace <hostName> with your actual back end system server URL hosting Apache Solr as the search engine. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | Username received from the back end system hosting Apache Solr as the search engine. |
| **Password** | This is the password received from the back end system hosting Apache Solr as the search engine. |
| **Authorization Type** | Apache Solr REST APIs use Basic Authentication. The Username and Password is passed when you invoke any of the REST API endpoints. This is the type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable |

| Field | Description |
|---|---|
| | hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Avalara AvaTax

Integration Cloud connects to Avalara AvaTax using the Avalara SOAP API and allows you to calculate taxes, modify documents, and validate addresses.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://<instance_name>.avalara.net, where <instance_name> represents the actual instance name. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |
| **Password** | Provide a password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **clientname** | Client application name and version. This should uniquely identify the software client that is calling the AvaTax service. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote |

| Field | Description |
|---|---|
| | server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Atlassian Jira

Integration Cloud connects to JIRA using the Interface for RESTful Web Services. You can use it for bug tracking, issue tracking, and project management functions.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: http://host:port.<br>Replace < host:port > with your actual JIRA instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | User name of the JIRA account. |
| **Password** | Password of the JIRA account. |

| Field | Description |
|---|---|
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic** . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. Select the Authorization Type as **basic**. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

# Cloud Deployment and webMethods Cloud Container

Use this Application to invoke Java and Flow services for any Cloud Deployment solution.

| Field | Description |
|---|---|
| **Run As** | Select the user name you want Integration Cloud to use while running the cloud deployment service. Integration Cloud runs the service as if the user you specify is the authenticated user that invoked the service. If the service is governed by an Access Control List, be sure to specify a user that is allowed to invoke the service.<br>**Note:**<br>You must map the selected user's Access Profile to a webMethods Integration Server user group (**Settings** ⚙ **> Access Profiles > Select an Access Profile > Solution Permissions**), else you will not be able to view, edit, or run webMethods Integration Server services in a solution. |
| **Stage** | Select the stage where the cloud deployment services are available. |

## webMethods Cloud Container

Use this Application to invoke Java and Flow services for any Cloud Container solution.

| Field | Description |
|---|---|
| **Server URL** | The cloud server URL where your on-premises assets are deployed, for example, https://{sub-domain}.{domain-name}.{domain-suffix}. |
| **User Name** | User name for the account on the cloud server. |
| **Password** | The password for the specified user. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

## Cloud Deployment Operations

Integration Cloud Integrations can invoke Cloud Deployment solution webMethods Integration Server Java and Flow services for the same tenant, by using the "Cloud Deployment" on page 136

Application. You can use these services along with other Integrations and services, both in Orchestration and Point-to-Point Integrations.

**Note:**
The Cloud Deployment Application appears under the **Predefined Applications** category.

To import Cloud Deployment solution webMethods Integration Server services, do the following:

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > Predefined Applications > Cloud Deployment**. The **Cloud Deployment** Application details page appears.
2. Select **Accounts > Add New Account** to create a new "Cloud Deployment account" on page 136.
3. Click **Operations**.
4. Click **Import Operations** if you want to import multiple Java or Flow services. If any of the services have a *Document Reference* in its input/output signature, then those services will not be imported. To import those services, select **Add new Operation** and import those services individually.
5. If you click **Import Operations**, the **Import Operations** window will appear. Select the cloud deployment account, and then select the services that you want to import in Integration Cloud.
6. Click **Import**. The services will appear on the **Operations** list page for the Cloud Deployment Application.

   **Note:**
   Importing will copy the metadata of the services in Integration Cloud. After import, you can use the operation in an Integration. The operation execution happens on the Cloud Deployment solution webMethods Integration Server.

## CloudStreams Connector for Microsoft Azure Cosmos DB

Integration Cloud connects to Microsoft Azure Cosmos DB and provides access to Microsoft's fully managed NoSQL database. You can use this Application to create, query, and manage resources in a NoSQL database.

| Field | Description |
|---|---|
| **Server URL** | The login endpoint to initiate communication with the SaaS provider. For example, for the CloudStreams Connector for Microsoft Azure Cosmos DB, the end point URL will be of the format: https://<accountName>.documents.azure.com:443/.<br>Replace <accountName> with the name of your Microsoft Azure Cosmos DB account. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, |

| Field | Description |
|---|---|
| | increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Access Key** | The Access key token that contains the secret of the account required by the SaaS provider. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## webMethods CloudStreams Connector for Microsoft Azure Data Lake Store

Integration Cloud connects to CloudStreams Connector for Microsoft Azure Data Lake Store using the REST API and allows you to manage File System resources through the Hadoop Distributed File System ( HDFS) API. You can create directories, folders, and files in your Azure Data Lake Store instance that can store and retrieve data.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<yourstorename>.azuredatalakestore.net/webhdfs/v1. Replace <yourstorename> with your actual Microsoft Azure Data Lake Store account. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. To get the Access Token, you first need to have an Azure subscription. Then create an Azure Active Directory native application. See End-user authentication with Data Lake Store using Azure Active Directory on how to create it. Successful authentication will provide an access token and a refresh token. The access token gets attached to each request made to Data Lake Store and is valid for one hour by default. The refresh token can be used to obtain a new access token and is valid for two weeks by default, if used regularly. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the |

| Field | Description |
|---|---|
| | request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |

## CloudStreams Connector for Microsoft Azure Storage

Integration Cloud connects to Microsoft Azure Storage and allows you to store, load, and query data. It includes a set of storage services, such as, Blob storage (object storage) for unstructured data, File storage for SMB-based cloud file shares, Table storage for NoSQL data, and Queue storage to reliably store messages.

| Field | Description |
|---|---|
| **Server URL** | The login endpoint to initiate communication with the SaaS provider. For example, for the CloudStreams Connector for Microsoft Azure Storage, the end point URL will be of the format: https://<accountName>.<br>Replace <accountName> with the name of your Microsoft Azure Storage account. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |

| Field | Description |
|---|---|
| **Access Key** | Microsoft Azure Storage REST APIs use Shared Key for authentication. Type the Access key that contains the secret of your Microsoft Azure Storage account. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## CloudStreams Connector for NetSuite™

Integration Cloud connects to NetSuite™ SuiteTalk platform using the SuiteTalk web services. It provides programmatic access to NetSuite™ data related to accounting, order management/inventory, CRM, professional services automation (PSA), and eCommerce applications through operations like addList, get, updateList, upsertList, and deleteList.

The NetSuite SOAP Web Service Application for v2018_2 uses the Token Based authentication mechanism. With each request, Integration Cloud calculates the signature, and the token passport details are sent as part of the request headers.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the Account configuration. You may need to specify the correct URL for your exact instance, for example, https://webservices.netsuite.com/services/NetSuitePort_2018_2. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Email** | The user email account that the connection will use to connect to the SaaS provider. |
| **Password** | The password of the user email account. |
| **Authorization Type** | This is the type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Account Id** | The Account ID or Account number received from NetSuite$^{TM}$. |
| **Consumer Key** | Consumer Key received from NetSuite. |
| **Consumer Secret** | Consumer Secret received from NetSuite. |
| **Token Id** | Token received from NetSuite. |
| **Token Secret** | Token Secret received from NetSuite. |
| **Role** | Specify the role with which you want to execute the web services, for example, Administrator. |
| **ApplicationID** | When you create the NetSuite$^{TM}$ Account, it sends a login request to the back end using email, password, Account, and the Application Id. Application Id is required for requests using end point 2015.2 or later. |

| Field | Description |
|---|---|
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is **org.apache.http.conn.ssl.DefaultHostnameVerifier**, which will enable hostname verification. Select **org.apache.http.conn.ssl.NoopHostnameVerifier** to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Concur

Integration Cloud connects to Concur using the Concur API and allows you to manage expenses and travel requests. It includes the Expense and Travel Request services.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<instance>/api. Replace <instance> with your actual Concur instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |

| Field | Description |
|---|---|
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. Concur REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Coupa

Integration Cloud connects to Coupa using the Coupa API and allows you to create, update, and query individual entries (records) within Coupa. It manages indirect purchases, invoices, and expenses in real time and provides executive dashboards and expense management.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, for Coupa, the end point URL would be of the format: https://<instance>.com.<br>Replace <instance> with your actual Coupa instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, |

| Field | Description |
|---|---|
|  | it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **X-COUPA-API-KEY** | The API Key received from the user account.<br>Coupa REST APIs authentication requests require a unique API key generated in Coupa. All API requests must pass an X-COUPA-API-KEY header with an API key. A key can be created from the API Keys section of the Administration tab by an administrator. The key is a 40-character long case-sensitive alphanumeric code. The API key is associated with an API user who is the equivalent of an administrator in Coupa. Any changes to resources through the API are attributed to the API user. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Cumulocity

Integration Cloud connects to Cumulocity and allows you to manage assets and Internet of Things (IoT) devices.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, for Cumulocity, the end point URL would be of the format: https://<instance>.<br>Replace <instance> with your actual Cumulocity instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |
| **Password** | Provide a password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |

| Field | Description |
|---|---|
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Database

Integration Cloud allows you to connect to a database using the Database Application. The Database Application allows you to perform database operations with cloud databases using a JDBC driver.

Complete the following fields to create a new account:

| Field | Description |
|---|---|
| **Database** | Select the supported database. |
| **Driver Group** | Select the driver group to connect to the database.<br>You can upload a new JDBC driver through the **Add Driver** option from the drop-down list box. You can upload only certified jars. |

| Field | Description |
|---|---|
| **Transaction Type** | Select the transaction type for transaction support for the account. The following are the supported transaction types:<br>■ **NO_TRANSACTION**: The connection automatically commits operations.<br>■ **LOCAL_TRANSACTION**: The connection uses local transactions. With this transaction type, all of the operations on the same account in a single transaction boundary are either committed or rolled back together. |
| **DataSource Class** | Select the DataSource class.<br>This field specifies the name of the JDBC driver's DataSource class. |
| **Server Name** | Type the name of the server that hosts the database.<br>**Note:**<br>If the tenant cannot connect to the cloud database, then check the security settings of the cloud database. |
| **User** | Type the user name that the connection uses to connect to the database. |
| **Password** | Type the password for the user name specified in **User**. |
| **Database Name** | Type the database name to which the connection connects to. |
| **Port Number** | Type the port number that the connection uses to connect to the database. |
| **Truststore Alias** | Select the alias name of the Integration Cloud truststore configuration. The truststore contains trusted certificates that are used to determine the trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Network Protocol** | Indicates the name of the network protocol that the connection uses when connecting to the database.<br>Type `TCP` or `TCPS` to indicate the network protocol. |
| **Other Properties** | Select or type the property name. Also, enter the value for the corresponding property in the input text field.<br>**Note:**<br>Properties listed for you to select from the drop-down list box are driver-dependent.<br>Example: Use this field to choose a property such as `TableFilter`. You can either select or type the `TableFilter` property in the |

| Field | Description |
|---|---|
| | drop-down list box and enter the `‘<current catalog>'.‘Accounting'` in the input text field.<br>Use `{}` to configure a combination of multiple key-value pairs. |
| | Example,<br>`connectionProperties={oracle.jdbc.V8Compatible=true,`<br>`includeSynonymns=true`<br>`}`<br>By default, the `loginTimeout` is set to 60. This value specifies the login timeout in seconds that a connection waits while attempting to connect to a database. |

The `<current catalog>` represents the default catalog associated with an account.

The `<current schema>` represents the default schema associated with an account.

### PostgreSQL Database

When you configure an operation to select a table, you must always select the table under `<current catalog>`.

### MariaDB

If you configure an operation for MariaDB using the driver versions greater than or equal to 2.4.0, select the table under `<current catalog>`.

### Database Application Operations

To use Database Application, you must create operations. Database operations allow you to connect to the database and initiate an operation on the database from Integration Cloud.

You call database operations from **Integration** to perform database operation on tables, views, or synonyms. The database operations are performed by calling JDBC APIs.

Operations are based on templates provided with Database Application. Each template represents an SQL statement for doing an operation on a database. For example, use `Select` operation to retrieve specified information from the database tables.

Creating a new operation from a template is straightforward. You will have to create an account using which you can create a new operation. During this process, select the operation template and configure the operation using the operation wizards.

Database Application provides the following operation templates:

| Operation | Description |
|---|---|
| Batch Insert | Inserts new information into a database table. Use this operation when you insert a large volume of data into a single table. |
| Batch Update | Updates the existing information in a database table. Use this operation when you update a large volume of data in a single table. |
| Custom SQL | Executes custom SQL to perform a database operation. |
| Delete | Deletes rows from a database table. |
| Insert | Inserts new information into a database table. |
| Select | Retrieves specified information from the database table. |
| Stored Procedure | Calls a stored procedure. It obtains the stored procedure's input/output parameters by introspecting when you configure the operation. |
| Update | Updates the existing information in a database table. |

### Advanced Options

The following options provide additional capabilities for Database operations:

| Parameter | Description |
|---|---|
| **Select DISTINCT** | Select the checkbox to suppress the duplicate rows in the query output.<br>**Note:** This option is applicable only for Select operation. |
| **Convert all to STRING** | Select the checkbox to convert the fields with input or output data type to java.lang.String. The fields which are already added to the table gets automatically converted to a String data type. It does not impact the fields which cannot be converted to String data type.<br>**Note:** This option is applicable for Select, Insert, Update, Delete, Batch Insert, and Batch Update operations. |

| Parameter | Description |
|---|---|
| **Use SQL Expression (Advanced)** | Use the following formats to provide SQL expressions:<br>- For table columns use `tableAlias.columnName`.<br>- For valid database functions use `databaseFunction(tableAlias.columnName)`.<br>For example, `upper(tableAlias.columnName)`.<br>**Note:** This option is applicable for Select, Update, Delete, and Batch Update operations. |

### Configuring the Parameter Values

### Custom SQL Operation

You can configure the following parameters for Custom SQL operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement.<br>**Note:** Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Maximum Row** | Type the number of rows to be retrieved from a database table.<br>**Note:** Select **No Limit** to indicate no limit on the number of rows retrieved. |
| **Row Count Field Name** | Type the field name of the Row Count Field. |

| Parameter | Description |
|---|---|
| | This field specifies the name of the output field whose value contains the number of affected rows while executing an SQL query. |
| **Row Count Field Type** | Select the data type of the Row Count Field. |

### Select Operation

You can configure the following parameters for Select operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a Statement to execute.<br>**Note:**<br>Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Maximum Row** | Type the number of rows to be retrieved from a database table.<br>**Note:**<br>Select **No Limit** to indicate no limit on the number of rows retrieved. |
| **Row Count Field Name** | Type the field name of the Row Count Field.<br>This field specifies the name of the output field whose value contains the number of affected rows while executing a Select SQL query. |
| **Row Count Field Type** | Select the data type of the Row Count Field. |

### Update Operation

You can configure the following parameters for Update operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute. |

| Parameter | Description |
|---|---|
| | **Note:**<br>Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Row Count Field Name** | Type the field name of the Row Count Field.<br>This field specifies the name of the output field whose value contains the number of affected rows while executing an Update SQL query. |
| **Row Count Field Type** | Select the data type of the Row Count Field. |

### Insert Operation

You can configure the following parameters for Insert operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute.<br>**Note:**<br>Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Row Count Field Name** | Type the field name of the Row Count Field.<br>This field specifies the name of the output field whose value contains the number of affected rows while executing a SQL statement. |
| **Row Count Field Type** | Select the data type of the Row Count Field. |

### Delete Operation

You can configure the following parameters for Delete operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute.<br>**Note:** |

| Parameter | Description |
|---|---|
| | Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Row Count Field Name** | Type the field name of the Row Count Field.<br>This field specifies the name of the output field whose value contains the number of affected rows while executing a SQL statement. |
| **Row Count Field Type** | Select the data type of the Row Count Field. |

### Stored Procedure Operation

You can configure the following parameters for Stored Procedure operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute.<br>**Note:**<br>Select **No Timeout** if you do not want to specify the Query Time Out. |

### Batch Insert Operation

You can configure the following parameters for Batch Insert operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute.<br>**Note:**<br>Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Batch Result Output Name** | Type the output name of the Batch Result Output Name.<br>The output of the batch operation is a string list. Elements in the list appear in the order in which you add the Insert SQL queries for execution in the batch |

| Parameter | Description |
|---|---|
| | mode. Depending on the JDBC driver you use, the elements in the list contains one of the following values:<br>■ A number greater than or equal to 0. This indicates the insert SQL query is successfully executed and returns the number of rows affected in the database.<br>■ A value of -2. This indicates that the insert SQL query was executed successfully but the number of rows affected is unknown. |

### Batch Update Operation

You can configure the following parameters for Batch Update operation:

| Parameter | Description |
|---|---|
| **Query Time Out** | Type or Select the query time out value in seconds.<br>This field specifies the number of seconds the database connection will wait for a SQL statement to execute.<br>**Note:** Select **No Timeout** if you do not want to specify the Query Time Out. |
| **Batch Result Output Name** | Type the output name of the Batch Result Output Name.<br>The output of the batch operation is a string list. Elements in the list appear in the order in which you add the Update SQL queries for execution in the batch mode. Depending on the JDBC driver you use, the elements in the list contains one of the following values:<br>■ A number greater than or equal to 0. This indicates the Update SQL query is successfully executed and returns the rows affected in the database.<br>■ A value of -2. This indicates that the command was processed successfully but the number of rows affected is unknown. |

## JDBC Data Type to Java Data Type Mappings

Each column in the database table is assigned a SQL type. The JDBC driver maps each SQL data type to a JDBC data type. Database Application then maps each JDBC data type to one or more Java data types that are used as the input or output of the database operation.

The following table shows the JDBC data type to Java data type mappings. You can map each JDBC data type to a set of Java data types by choosing one from the set. The JDBC data type you select during configuration will then map to the input or output of the database operation.

For a list of data types for which Database Application has some constraints, see "JDBC Data Type to Java Data Type Mapping Constraints" on page 163.

**Note:**
Database Application does not support the TIMESTAMP WITH TIME ZONE and TIMESTAMP WITH LOCAL TIME ZONE data types in Oracle 10g.

**Note:**
Database Application does not support user-defined data types, Oracle PL/SQL collections, or Oracle PL/SQL records.

**Note:**
UROWID data type is not supported for Oracle database.

**Note:**
If the DATE JDBC type contains String as Java type, then the date format which Database Application accepts is YYYY-MM-DD.

| JDBC Data Type | Java Data Type |
|---|---|
| ARRAY | java.sql.Array<br>java.lang.Object |
| BIT | java.lang.Boolean<br>java.lang.String<br>java.lang.Object |
| TINYINT | java.lang.Byte<br>java.lang.Integer<br>java.lang.String<br>java.lang.Object<br>SetAsString |
| SMALLINT | java.lang.Short<br>java.lang.Integer<br>java.lang.String |

| JDBC Data Type | Java Data Type |
|---|---|
| | java.lang.Object |
| INTEGER | java.lang.Integer<br>java.lang.String<br>java.lang.Object |
| BIGINT | java.lang.Long<br>java.lang.String<br>java.lang.Object |
| FLOAT | java.lang.Double<br>java.lang.String<br>java.lang.Object<br>java.math.BigDecimal<br>SetAsString |
| REAL | java.lang.Float<br>java.lang.String<br>java.lang.Object<br>java.math.BigDecimal |
| BOOLEAN | java.lang.Boolean<br>java.lang.String<br>java.lang.Object |
| DOUBLE | java.lang.Double<br>java.lang.String<br>java.lang.Object<br>java.math.BigDecimal<br>SetAsString |
| NUMERIC | java.math.BigDecimal<br>java.lang.String<br>java.lang.Object |
| DECIMAL | java.math.BigDecimal |

| JDBC Data Type | Java Data Type |
|---|---|
| | java.lang.String<br>java.lang.Object |
| CHAR | java.lang.String<br>java.lang.Character<br>java.lang.Object |
| VARCHAR | java.lang.String<br>java.lang.Object |
| LONGVARCHAR | java.lang.String<br>java.lang.Object |
| DATE | java.sql.Date<br>java.util.Date<br>java.lang.String<br>java.lang.Object<br>*SetAsString |
| TIME | java.sql.Time<br>java.util.Date<br>java.lang.String<br>java.lang.Object<br>SetAsString |
| TIMESTAMP | java.sql.Timestamp<br>java.util.Date<br>java.lang.String<br>java.lang.Object<br>SetAsString |
| TIMESTAMP WITH TIME ZONE | |
| TIMESTAMP WITH LOCAL TIME ZONE | |
| BINARY | byte array (byte [])<br>java.lang.Object |

| JDBC Data Type | Java Data Type |
|---|---|
| VARBINARY | byte array (byte[])<br>java.lang.Object |
| LONGVARBINARY | byte array (byte[])<br>java.lang.Object |
| LONGNVARCHAR | java.lang.String<br>java.lang.Object |
| NCHAR | java.lang.String<br>java.lang.Object |
| NULL | java.lang.String<br>java.lang.Object |
| NVARCHAR | java.lang.String<br>java.lang.Object |
| CLOB | java.sql.Clob<br>java.lang.String<br>java.io.Reader<br>java.lang.Object |
| BLOB | java.sql.Blob<br>byte array<br>java.io.InputStream<br>java.lang.Object |
| ORACLECURSOR | java.lang.Object |
| ORACLEFIXED_CHAR | java.lang.String |
| STRUCT | java.sql.Struct<br>java.lang.Object |
| OTHER | java.lang.Object<br>java.lang.String<br>java.sql.Struct<br>java.sql.Array |

### Important Considerations When Using BLOB and CLOB Data Types

- When passing large CLOB or BLOB data, use the Java data types java.io.Reader for CLOB and java.io.InputStream for BLOB to prevent Database Application from running out of memory. When using these data types, Database Application streams the data into bytes thus allowing to pass large data. The data types java.io.Reader and java.io.InputStream are supported only for the Oracle database using the Oracle driver.
- When using the CLOB data with java.io.Reader as input data type, it is recommended that you use the InputStreamReader implementation of java.io.Reader with the correct encoding parameter.
- When Integration Cloud executes a SELECT operation that has its output type set to java.sql.Blob for a BLOB data type, Integration Cloud issues a java.io.NotSerializableException error. The work around for this issue is to use with **Integration**.

### Important Considerations When Using the Array and Struct Database Specific Data Types

- In database operation, when using the java.lang.Object as the output field type for a database column of type ARRAY or STRUCT, Database Application returns the data as a java.lang.Object array, provided that the ARRAY or STRUCT data in the database table is composed of primitive data types.
- When using the java.sql.Array or java.sql.Struct as the output field type for a database column of type ARRAY or STRUCT, Database Application returns the java.sql.Array and the java.sql.Struct objects, respectively, as returned by the driver. However, when serializing the data across the JVMs, this returned data may not be serializable and may result into a java.io.NotSerializableException. Therefore, before serializing the data across the JVMs, it is important that you use Integration to process the java.sql.Struct and java.sql.Array objects as required, and then drop them from the pipeline.

  **Note:**
  The java.sql.Struct and java.sql.Array data types are available only for Database Application operations.

### Using the SetAsString Data Type in Database Application

The SetAsString data type is a dummy string data type. When using this data type, Database Application does not try to convert the input data into the equivalent JDBC data type, but passes the data to the underlying database driver as a string data type. Thus, you have the flexibility to specify the format of the equivalent JDBC data type by using a database specific function.

For example, you can specify the format for date, time, or timestamp using the to_date function or a similar database function for Oracle database. Database Application treats the input data as a string data type and does not convert it to the equivalent JDBC data type. The to_date function then uses the string data to provide the required format of the date, time or timestamp.

If your database has native database specific functions that can convert string data type to any other data type, you may use the SetAsString data type.

**Note:**
The SetAsString data type is available only for database operations.

### JDBC Data Type to Java Data Type Mapping Constraints

Database Application has some constraints when mapping JDBC data types to Java data types.

If you select one of the following Java data types, the data type will map exactly to the **Input/Output** of operation:

- java.lang.String
- java.lang.Byte
- java.lang.Boolean
- java.lang.Character
- java.lang.Double
- java.lang.Float
- java.lang.Integer
- java.lang.Long
- java.lang.Short
- java.util.Date
- java.math.BigDecimal
- java.math.BigInteger
- java.lang.Object

Those data types not included in this list will map to java.lang.Object. In these cases, if the JDBC data type you specify is for input, you will need to pass in the object with the selected Java data type. If the JDBC type is for output, you can cast the object to the selected Java data type.

## Driver Management

The **Driver Management** screen is available only for Database Application. This screen lists all the available drivers uploaded for Database Application. It also gives you an option to upload a custom driver jar for a particular database. This uploaded driver can be used when you create an account. You can add or delete a driver jar from this screen.

Pre-bundled driver jars are provided to connect to the databases like Microsoft SQL Server, Oracle, PostgreSQL. You cannot delete these pre-bundled driver group. The below table shows the driver groups and supported databases.

| Driver Group | Supported Databases | Version |
|---|---|---|
| Progress® DataDirect®. | Microsoft SQL Server, Oracle | 5.1.4 |
| Microsoft JDBC Driver for SQL Server | Microsoft SQL Server | 4.0.0 |
| PostgreSQL JDBC Driver | PostgreSQL | 42.2.5 |

**Note:**
If you use Progress® DataDirect® driver to create a Non-SSL connection for Microsoft SQL Server database, then set the **Other Properties** with

```
validateservercertificate=false;encryptionmethod=ssl.
```

**Note:**
For Oracle database, if you are using Progress® DataDirect® driver group to create a connection for Stored Procedure operation, then the Stored Procedure operation cannot fetch the metadata for SYS_CURSOR data type.

### To add a new driver

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > Predefined Applications**.

2. Select **Database** and then click **Driver Management**.

3. From the **Driver Management** page, click **Add Driver**.

4. Select the supported **Database**.

5. Type or select the driver from the **Driver Group**. Browse the jar in the **Select the Driver** field for a particular database.

   The driver jar gets uploaded for the corresponding database.

   **Note:**
   You can upload only the certified jars.

   **Note:**
   The uploaded driver jars can be used across the projects in the tenant.

## Driver Management Fields

You can configure the following fields for Driver Management:

| Parameter | Description |
|---|---|
| **Database** | Select the supported database. |

| Parameter | Description |
|---|---|
| **Driver Group** | Type a new driver group name or select the existing group name from the drop-down<br>The uploaded JDBC driver jars are referred by the given name when you create an account. |
| **Select the Driver** | Browse and select the driver jar for a particular vendor. You must select the driver jars which are certified by Software AG. |

### Certified Databases and JDBC Driver Jars

You can upload only the following list of certified driver jars:

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| Oracle | 11.2.0.4 | ojdbc5.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>documentation/jdbc-112010-090769.html | 5 | Oracle 11.2.0.4 |
| | 11.2.0.4 | ojdbc6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>documentation/jdbc-112010-090769.html | 6,7,8 | Oracle 11.2.0.4 |
| | 11.2.0.4 | xdb6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>documentation/jdbc-112010-090769.html | 6,7 | Oracle 11.2.0.4 |
| | 11.2.0.4 | xmlparserv2-11.2.0.4.jar<br>**Download Path**<br>https://www.oracle.com/content/secure/<br>maven/content/com/oracle/jdbc/xmlparserv2/ | NA | Oracle 11.2.0.4 |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | | 11.2.0.4/xmlparserv2-11.2.0.4.jar | | |
| | 12.1.0.1 | ojdbc6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html | 6 | Oracle 11.2.0.4, 12.1.0.1 |
| | 12.1.0.1 | ojdbc7.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html | 7,8 | Oracle 11.2.0.4, 12.1.0.1 |
| | 12.1.0.1 | xdb6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html | 6,7 | Oracle 11.2.0.4, 12.1.0.1 |
| | 12.1.0.1 | xmlparserv2-12.1.0.1.jar<br>**Download Path**<br>https://www.oracle.com/content/secure/maven/content/com/oracle/jdbc/xmlparserv2/12.1.0.1/xmlparserv2-12.1.0.1.jar | NA | Oracle 11.2.0.4, 12.1.0.1 |
| | 12.1.0.2 | ojdbc6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/database/features/jdbc/default-2280470.html | 6 | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2 |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | 12.1.0.2 | ojdbc7.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>database/features/jdbc/default-2280470.html | 7,8 | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2 |
| | 12.1.0.2 | xdb6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>database/features/jdbc/default-2280470.html | 6,7 | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2 |
| | 12.1.0.2 | xmlparserv2-12.1.0.2.jar<br>**Download Path**<br>https://www.oracle.com/content/secure/<br>maven/content/com/oracle/jdbc/<br>xmlparserv2/12.1.0.2/xmlparserv2-12.1.0.2.jar | NA | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2 |
| | 12.2.0.1 | ojdbc8.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/<br>database/features/jdbc/jdbc-ucp-122-3110062.html | 8 | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2, 12.2.0.1 |
| | 12.2.0.1 | xdb6.jar<br>**Download Path**<br>https://www.oracle.com/technetwork/database/<br>features/jdbc/jdbc-ucp-122-3110062.html | 7,8 | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2, 12.2.0.1 |
| | 12.2.0.1 | xmlparserv2-12.2.0.1.jar<br>**Download Path**<br>https://www.oracle.com/content/secure/<br>maven/content/com/oracle/jdbc/<br>xmlparserv2/12.2.0.1/xmlparserv2-12.2.0.1.jar | NA | Oracle 11.2.0.4, 12.1.0.1, 12.1.0.2, 12.2.0.1 |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| MySQL | 8.0.16 | mysql-connector-java-8.0.16. jar<br>**Download Path**<br>https://downloads.mysql.com/archives/c-j/ | 8 | MySQL 8.0, 5.7, 5.6, 5.5 |
| | 8.0.17 | mysql-connector-java-8.0.17.jar<br>**Download Path**<br>https://dev.mysql.com/downloads/connector/j/<br>**Note:** If you are unable to find the driver jar for the specified version, then download the driver from the archived version's download link as shown below:<br>https://downloads.mysql.com/archives/c-j/ | 8 | MySQL 8.0, 5.7, 5.6, and 5.5 |
| Microsoft SQL Server | 4.1 | sqljdbc41.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=841533 | 7 | Microsoft SQL Server 2008, 2012, 2014, 2016, and Azure SQL Database |
| | 4.2 | sqljdbc41.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=841534 | 7 | Microsoft SQL Server 2008, 2012, 2014, 2016, and Azure SQL Database |
| | 4.2 | sqljdbc42.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=841534 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, and Azure SQL Database |
| | 6.0 | sqljdbc41.jar<br>**Download Path** | 7 | Microsoft SQL Server 2008, 2012, |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | | https://go.microsoft.com/fwlink/?LinkId=245496 | | 2014, 2016 and Azure SQL Database |
| | 6.0 | sqljdbc42.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?LinkId=245496 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, and Azure SQL Database |
| | 6.2.2 | mssql-jdbc-6.2.2.jre7.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=852460 | 7 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017, and Azure SQL Database |
| | 6.2.2 | mssql-jdbc-6.2.2.jre8.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=852460 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017 and Azure SQL Database |
| | 6.4.0 | mssql-jdbc-6.4.0.jre7.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=868290 | 7 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017, and Azure SQL Database |
| | 6.4.0 | mssql-jdbc-6.4.0.jre8.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=868290 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017, and Azure SQL Database |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | 7.0.0 | mssql-jdbc-7.0.0.jre8.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=2063159 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017, and Azure SQL Database |
| | 7.2.2 | mssql-jdbc-7.2.1.jre8.jar<br>**Download Path**<br>https://go.microsoft.com/fwlink/?linkid=2063159 | 8 | Microsoft SQL Server 2008, 2012, 2014, 2016, 2017, and Azure SQL Database |
| PostgreSQL | 42.2.5 | postgresql-42.2.5.jre6.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 6 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.5 | postgresql-42.2.5.jre7.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 7 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.5 | postgresql-42.2.5.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 8 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.6 | postgresql-42.2.6.jre6.jar<br>**Download Path** | 6 | Aurora PostgreSQL (Compatible |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | | https://jdbc.postgresql.org/download.html | | with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.6 | postgresql-42.2.6.jre7.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 7 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.6 | postgresql-42.2.6.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 8 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.7 | postgresql-42.2.7.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 8 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.7 | postgresql-42.2.7.jre6.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 6 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.7 | postgresql-42.2.7.jre7.jar<br>**Download Path** | 7 | Aurora PostgreSQL (Compatible |

| Database | Driver Version | Driver Jar Name | Java Version | Supported DB Versions |
|---|---|---|---|---|
| | | https://jdbc.postgresql.org/download.html | | with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| | 42.2.8 | postgresql-42.2.8.jar<br>**Download Path**<br>https://jdbc.postgresql.org/download.html | 8 | Aurora PostgreSQL (Compatible with PostgreSQL 9.6.8, 9.6.9, 10.4, 10.5, 10.6) |
| Tibero | 6.0 | tibero6-jdbc.jar<br>**Download Path**<br>The driver jar is available in the Tibero 6 installation directory,<br>`($TB_HOME/client/lib/jar).` | NA | Tibero 6 |
| MariaDB | 2.3.0 | mariadb-java-client-2.3.0.jar<br>**Download Path**<br>http://downloads.mariadb.com/Connectors/java/ | 8 | MariaDB 10.3.8 |

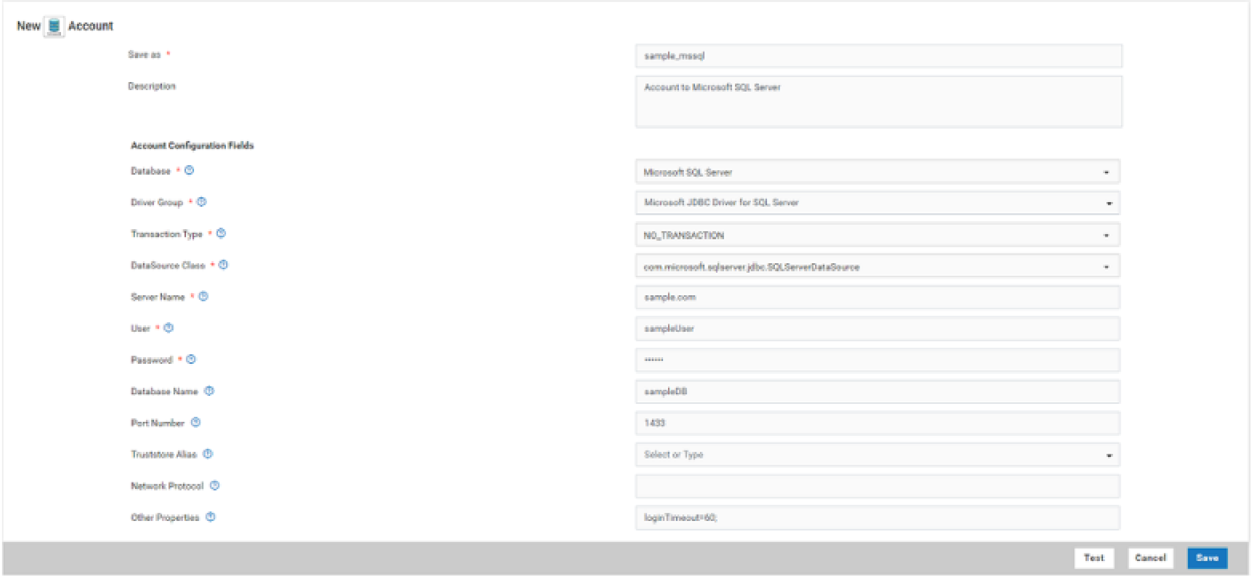
**Note:**
For MySQL driver, ensure that you download the **Platform Independent** variant.

## Samples Accounts

The following are the sample accounts created for secured and non secured connection for various databases:

### Microsoft SQL Server

The following are the fields to be configured for a non-secure Microsoft SQL Server account:



New Account

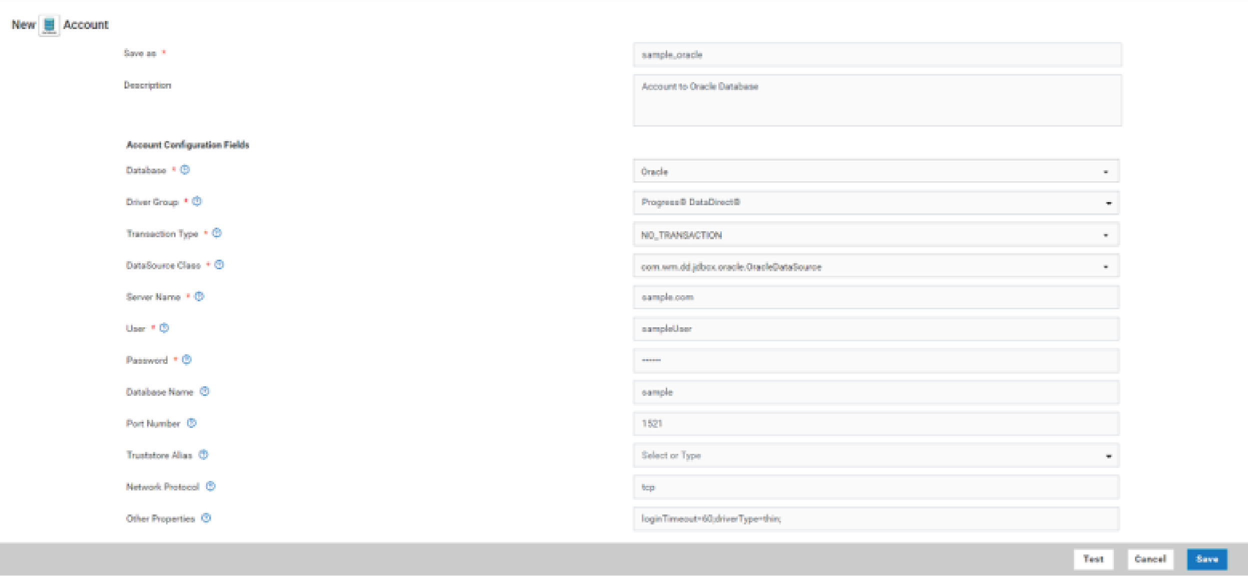
| | |
|---|---|
| Save as * | sample_mssql |
| Description | Account to Microsoft SQL Server |
| **Account Configuration Fields** | |
| Database * | Microsoft SQL Server |
| Driver Group * | Microsoft JDBC Driver for SQL Server |
| Transaction Type * | NO_TRANSACTION |
| DataSource Class * | com.microsoft.sqlserver.jdbc.SQLServerDataSource |
| Server Name * | sample.com |
| User * | sampleUser |
| Password * | ****** |
| Database Name | sampleDB |
| Port Number | 1433 |
| Truststore Alias | Select or Type |
| Network Protocol | |
| Other Properties | loginTimeout=60; |

**Note:**
For a secured connection using Microsoft SQL database, edit the following fields:

- Select **Truststore Alias** from the drop-down. If this option is not available in the drop-down, then upload your certificate using **Add New Truststore**.
- Set the **Other Properties** to `loginTimeout=60;encrypt=true`.

### Oracle

The following are the fields to be configured for a non-secure Oracle account:



New Account

| | |
|---|---|
| Save as * | sample_oracle |
| Description | Account to Oracle Database |
| **Account Configuration Fields** | |
| Database * | Oracle |
| Driver Group * | Progress® DataDirect® |
| Transaction Type * | NO_TRANSACTION |
| DataSource Class * | com.wm.dd.jdbcx.oracle.OracleDataSource |
| Server Name * | sample.com |
| User * | sampleUser |
| Password * | ------ |
| Database Name | sample |
| Port Number | 1521 |
| Truststore Alias | Select or Type |
| Network Protocol | tcp |
| Other Properties | loginTimeout=60;driverType=thin; |

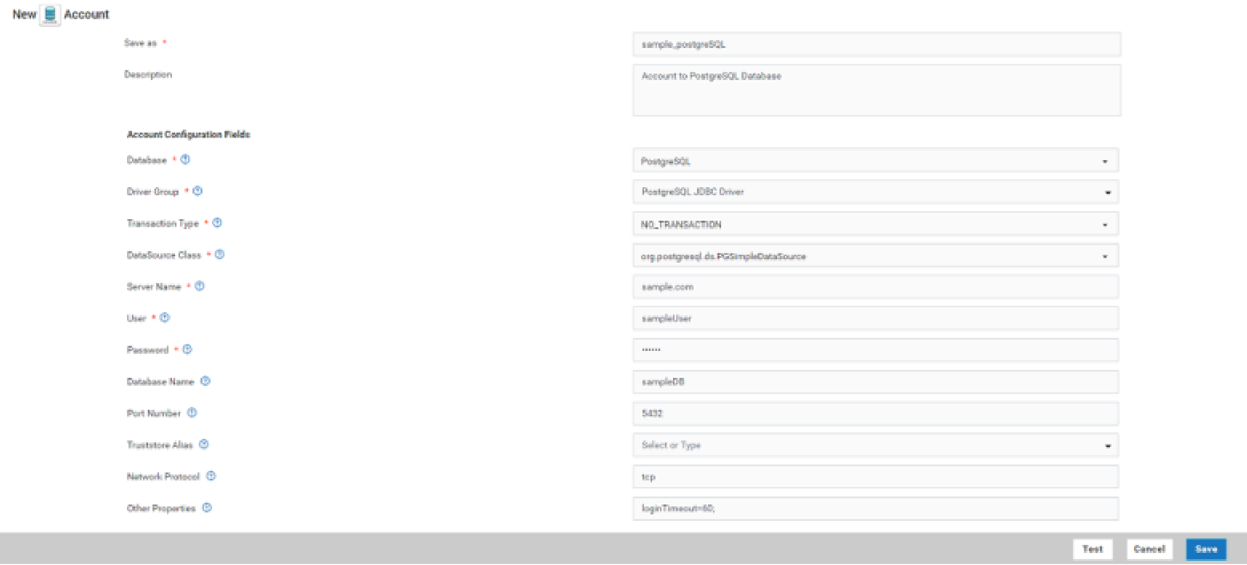
**Note:**
For a secured connection using Oracle database, edit the following fields:

- Select **Truststore Alias** from the drop-down. If this option is not available in the drop-down, then upload your certificate using **Add New Truststore**.
- Type `tcps` for **Network Protocol**

**Note:**
By default, 2484 is the `tcps` port.

### PostgreSQL

The following are the fields to be configured for a non-secure PostgreSQL account:



New Account

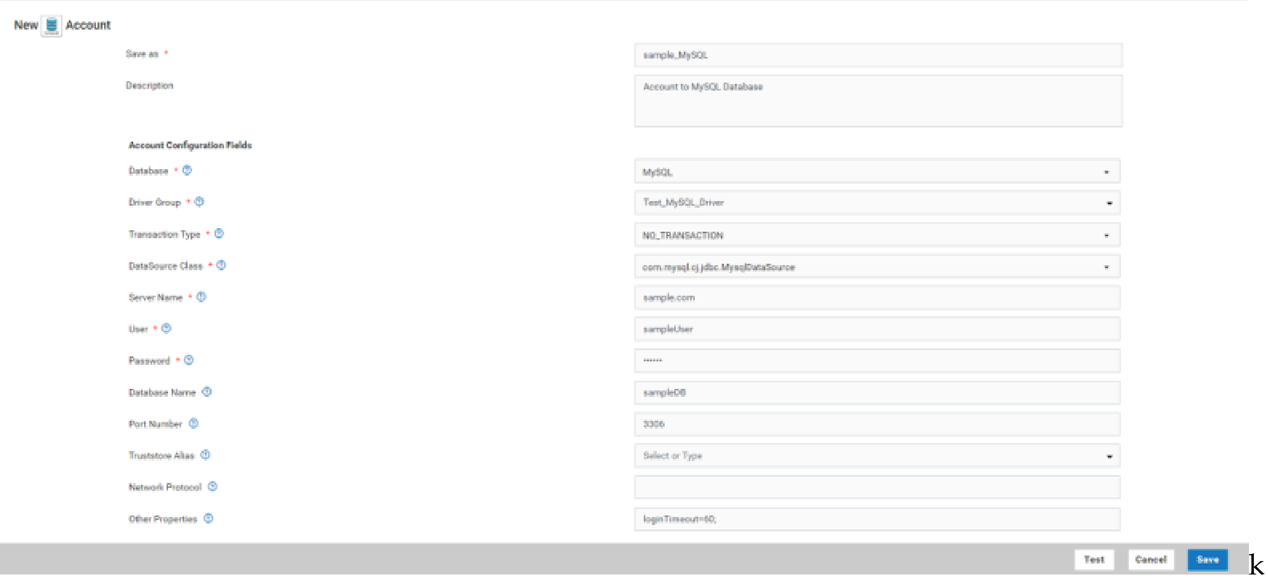
| | |
|---|---|
| Save as * | sample_postgreSQL |
| Description | Account to PostgreSQL Database |
| **Account Configuration Fields** | |
| Database * | PostgreSQL |
| Driver Group * | PostgreSQL JDBC Driver |
| Transaction Type * | NO_TRANSACTION |
| DataSource Class * | org.postgresql.ds.PGSimpleDataSource |
| Server Name * | sample.com |
| User * | sampleUser |
| Password * | ······ |
| Database Name | sampleDB |
| Port Number | 5432 |
| Truststore Alias | Select or Type |
| Network Protocol | tcp |
| Other Properties | loginTimeout=60; |

**Note:**
For secured connection using PostgreSQL, you need to select the **Truststore Alias** from the drop-down. If this option is not available in the drop-down, then upload your certificate using **Add New Truststore**.

**Note:**
The **Network Protocol** field is not mandatory for PostgreSQL account.

### MySQL

The following are the fields to be configured for a non-secure MySQL account:



New Account

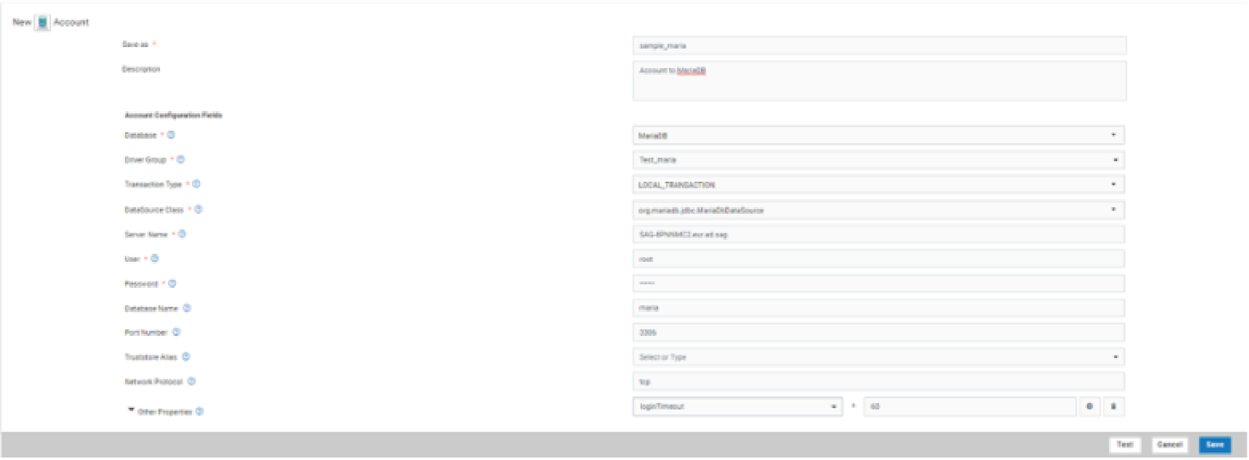
| | |
|---|---|
| Save as * | sample_MySQL |
| Description | Account to MySQL Database |
| **Account Configuration Fields** | |
| Database * | MySQL |
| Driver Group * | Test_MySQL_Driver |
| Transaction Type * | NO_TRANSACTION |
| DataSource Class * | com.mysql.cj.jdbc.MysqlDataSource |
| Server Name * | sample.com |
| User * | sampleUser |
| Password * | •••••• |
| Database Name | sampleDB |
| Port Number | 3306 |
| Truststore Alias | Select or Type |
| Network Protocol | |
| Other Properties | loginTimeout=60; |

k

**Note:**
For secured connection using MySQL database, you need to select the **Truststore Alias** from the drop-down. If this option is not available in the drop-down, then upload your certificate using **Add New Truststore**.

### MariaDB

The following are the fields to be configured for a non-secure MariaDB account:



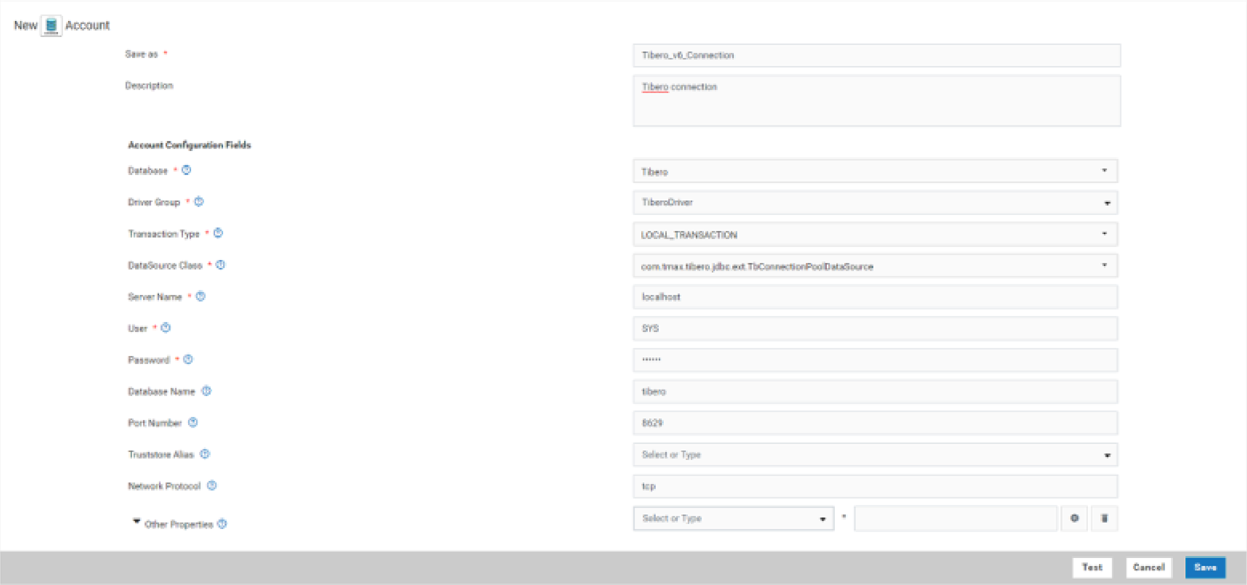
New Account

| | |
|---|---|
| Save as * | sample_maria |
| Description | Account to MariaDB |
| **Account Configuration Fields** | |
| Database * | MariaDB |
| Driver Group * | Test_maria |
| Transaction Type * | LOCAL_TRANSACTION |
| DataSource Class * | org.mariadb.jdbc.MariaDbDataSource |
| Server Name * | SAG-8PNWMC2.eur.ad.sag |
| User * | root |
| Password * | ••••• |
| Database Name | maria |
| Port Number | 3306 |
| Truststore Alias | Select or Type |
| Network Protocol | tcp |
| Other Properties | loginTimeout * 60 |

**Note:**
For secured connection using MariaDB, select the **Truststore Alias** from the drop-down list. If this option is not available in the drop-down list, then upload your certificate using **Add New Truststore**.

### Tibero

The following are the fields to be configured for a non-secure Tibero account:



New Account

| | |
|---|---|
| Save as * | Tibero_v6_Connection |
| Description | Tibero connection |

**Account Configuration Fields**

| | |
|---|---|
| Database * | Tibero |
| Driver Group * | TiberoDriver |
| Transaction Type * | LOCAL_TRANSACTION |
| DataSource Class * | com.tmax.tibero.jdbc.ext.TbConnectionPoolDataSource |
| Server Name * | localhost |
| User * | SYS |
| Password * | •••••• |
| Database Name | tibero |
| Port Number | 8629 |
| Truststore Alias | Select or Type |
| Network Protocol | tcp |
| Other Properties | Select or Type |

## Supported Cloud Databases

The following table lists the supported cloud databases:

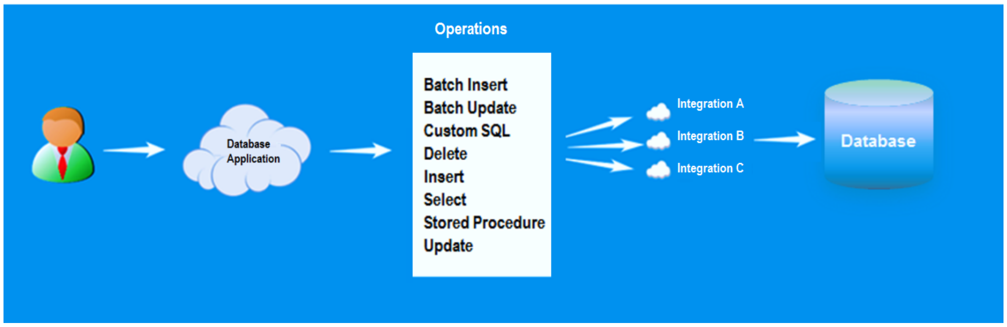
| Cloud Database | Database Version |
|---|---|
| Microsoft Azure | Microsoft SQL Azure (RTM) - 12.0.2000.8 |
| Amazon RDS Oracle | Oracle 12c, Version 12.1.0.2 |
| Amazon RDS MS SQL Server | SQL Server 2016 SP1 CU7 13.00.4466.4, released |
| AWS Aurora PostgreSQL | Aurora PostgreSQL 10.6 |
| AWS Aurora MySQL | Aurora MySQL 5.7.12 |
| Amazon RDS MariaDB | MariaDB 10.3.8 |
| Tibero on Amazon EC2 | Tibero 6 |

## How to use the Insert and Select operations using the Database Application

### What is a Database Application?

The **Database** Application allows you to perform database operations with cloud databases. All the database operations are performed using JDBC Driver provided by the database vendor.

You can create an integration with any cloud databases using the Database Application which supports operations such as Insert, Select, Delete, Update, and so on. All the operations in the **Database** Application share similar user interfaces and design approaches.



## Actors

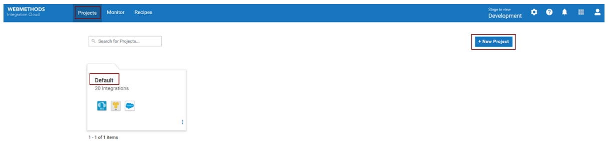
- Administrator
- User who creates integration to the database

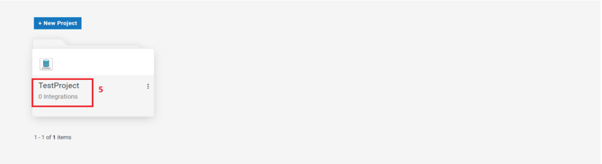
## Preconditions

- Knowledge to create and execute an operation
- Basic knowledge of Database and SQL queries
- An existing database with appropriate permissions
- A valid tenant that contains Integration Cloud subscription with access to Database Application

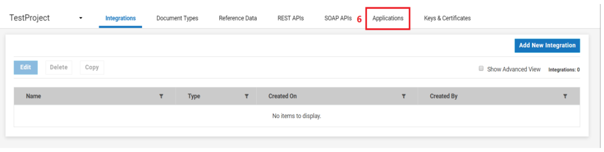
## Basic Flow

1. Login to Integration Cloud with the tenant credentials.
2. You can create a new project or use an existing project to create an account, operation, and so on using the below screen:

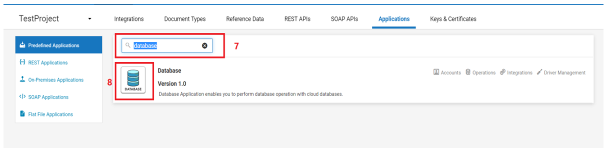




3. Click **Applications** and find the Database application as shown below:

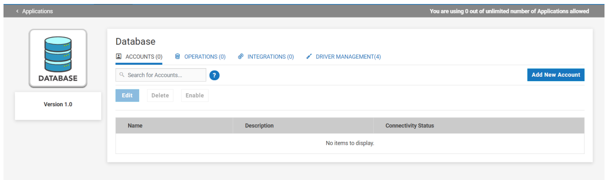


| Name | Type | Created On | Created By |
|---|---|---|---|
| No items to display. | | | |



Database Application enables you to perform database operation with cloud databases.

4. You can now create an account, operation, and integration using the below screen:



You are using 0 out of unlimited number of Applications allowed

No items to display.

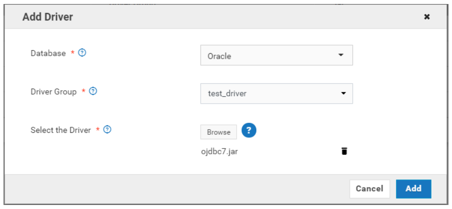
5. Let us assume that you want to insert a row into an EMPLOYEE table using the **INSERT** operation and then select the inserted row using the **SELECT** operation. The sample structure for the *Employee* table can be as shown below:

```
EMP_NO     NUMBER(4)
EMP_NAME           VARCHAR2(10)
JOB                VARCHAR2(9)
MANAGER            NUMBER(4)
HIRE_DATE          DATE
SALARY             NUMBER(7,2)
COMM               NUMBER(7,2)
DEPT_NO            NUMBER(2)
```

6. Do the following to insert a row into the *Employee* table:

   a. Add a JDBC driver.

      Navigate to the particular project where you want to create the operation Then, go to **Applications** > **Database** > **DRIVER MANAGEMENT** > **Add Driver**. Provide values as shown below and click **Add**.

      Database * : Oracle

      Driver Group * : test_driver

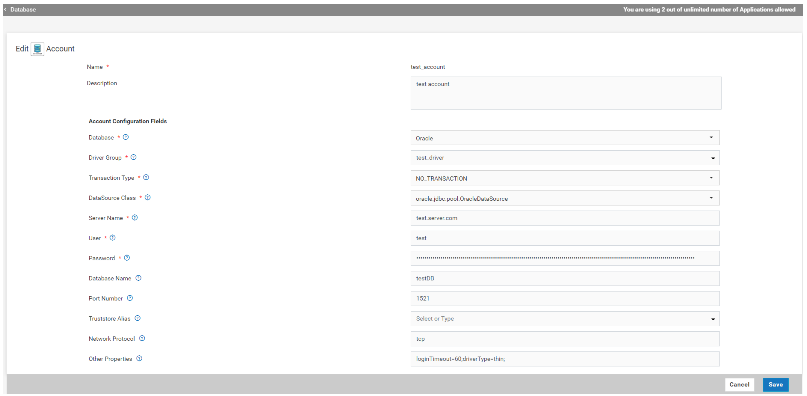
      Select the Driver * : ojdbc7.jar

      > **Note:**
      > The Database Application also provides pre-bundled JDBC drivers. If you want to use a pre-bundled JDBC driver, you need not add a new JDBC driver.

   b. Create a new account.

      Navigate to the particular project where you have already added the JDBC driver. Then go to **Applications** > **Database** > **ACCOUNTS** > **Add New Account**.

      Provide the values for each field as shown below and save the account details:


Database
You are using 2 out of unlimited number of Applications allowed
Cancel
Save

Edit Account

| | |
|---|---|
| Name * | test_account |
| Description | test account |

**Account Configuration Fields**

| | |
|---|---|
| Database * | Oracle |
| Driver Group * | test_driver |
| Transaction Type * | NO_TRANSACTION |
| DataSource Class * | oracle.jdbc.pool.OracleDataSource |
| Server Name * | test.server.com |
| User * | test |
| Password * | •••••••••••••••••••••••••••••••••••••••••••••••••••••••••••• |
| Database Name | testDB |
| Port Number | 1521 |
| Truststore Alias | Select or Type |
| Network Protocol | tcp |
| Other Properties | loginTimeout=60;driverType=thin; |

**Note:**
The **Truststore Alias** field is used only when you create a secure connection.

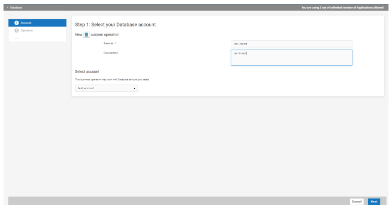
c. Configure the Insert operation.

Navigate to the particular project where you have created the account.

Then go to **Applications** > **Database** > **OPERATIONS** > **Add New Operation**.

Follow the below steps to configure the **Insert** operation:

a. On the **Account** screen, provide the name of the operation, description and choose the account to create the operation as shown below:


Database
You are using 2 out of unlimited number of Applications allowed
1 Account
2 Operation
Cancel
Next

Step 1: Select your Database account

New custom operation

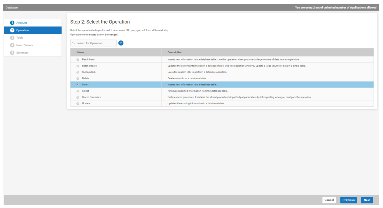
| | |
|---|---|
| Save as * | test_insert |
| Description | test insert |

Select account

This business operation may work with Database account you select.

test_account

b. On the **Operation** screen, select the **Insert** operation from the list of operation templates:

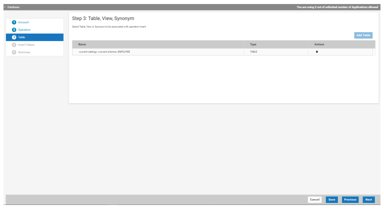


Step 2: Select the Operation

Select the operation to be performed. It determines SQL query you will form at the next step
Operation once selected cannot be changed

| Name | Description |
|---|---|
| ○ Batch Insert | Inserts new information into a database table. Use this operation when you insert a large volume of data into a single table. |
| ○ Batch Update | Updates the existing information in a database table. Use this operation when you update a large volume of data in a single table. |
| ○ Custom SQL | Executes custom SQL to perform a database operation. |
| ○ Delete | Deletes rows from a database table. |
| ● Insert | Inserts new information into a database table. |
| ○ Select | Retrieves specified information from the database table. |
| ○ Stored Procedure | Calls a stored procedure. It obtains the stored procedure's input/output parameters by introspecting when you configure the operation. |
| ○ Update | Updates the existing information in a database table. |

c. For **Tables**, click **Add Table**, and select the *Employee* table as shown below:

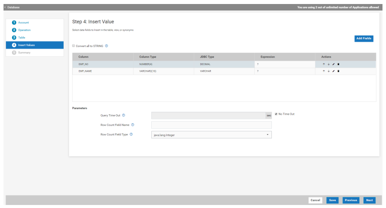


Step 3: Table, View, Synonym

Select Table, View or Synonym to be associated with operation Insert

| Name | Type | Actions |
|---|---|---|
| <current catalog>.<current schema>.EMPLOYEE | TABLE | ▪ |

d. For **Insert Values**, click **Add Fields**, and select the table columns:



Step 4: Insert Value

Select data fields to insert in the table, view, or synonyms

☐ Convert all to STRING

| Column | Column Type | JDBC Type | Expression | Actions |
|---|---|---|---|---|
| EMP_NO | NUMBER(4) | DECIMAL | ? | ↑ ↓ ✎ 🗑 |
| EMP_NAME | VARCHAR2(10) | VARCHAR | ? | ↑ ↓ ✎ 🗑 |

Parameters

Query Time Out: sec ☑ No Time Out

Row Count Field Name:

Row Count Field Type: java.lang.Integer

**Note:**
The default value for the **Expression** field is ?, which specifies to provide the value for that particular column when you execute the **Insert** operation.

e. The **Summary** displays the operation details such as the name of the **Operation**, **Account**, and **SQL** query formed. Click **Finish** or **Save**.

d. Execute the **Insert** operation and provide values for the selected columns.

7. To select a row from the *Employee* table, do the following:

a. Add a JDBC Driver.

The JDBC Driver is already added while creating the Insert operation.

b. Create a new account.

The account is already created while creating the **Insert** operation.

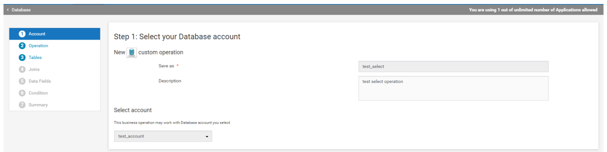
c. Configure the **Select** operation.

To configure, navigate to the respective project where you have created the account.

Later navigate to **Applications** > **Database** > **OPERATIONS** > **Add New Operation**.

Do the following to configure the **Select** operation:

a. On the **Account** screen, provide the name of the operation, description, and choose the account to create the operation as shown below:



Step 1: Select your Database account

New custom operation

Save as *: test_select

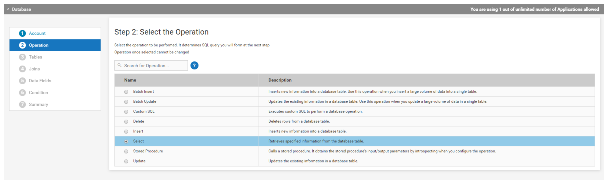
Description: test select operation

Select account

This business operation may work with Database account you select.

test_account

b. On the **Operation** screen, choose the **Select** operation from the list of operation templates as shown below:



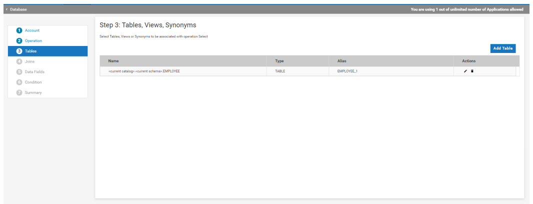
Step 2: Select the Operation

Select the operation to be performed. It determines SQL query you will form at the next step

Operation once selected cannot be changed

| Name | Description |
|---|---|
| ○ Batch Insert | Inserts new information into a database table. Use this operation when you insert a large volume of data into a single table. |
| ○ Batch Update | Updates the existing information in a database table. Use this operation when you update a large volume of data in a single table. |
| ○ Custom SQL | Executes custom SQL to perform a database operation. |
| ○ Delete | Deletes rows from a database table. |
| ○ Insert | Inserts new information into a database table. |
| ● Select | Retrieves specified information from the database table. |
| ○ Stored Procedure | Calls a stored procedure. It obtains the stored procedure's input/output parameters by introspecting when you configure the operation. |
| ○ Update | Updates the existing information in a database table. |

c. For **Tables**, click **Add Table** and select the *Employee* table as shown below:



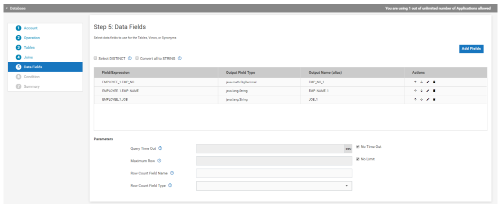
Step 3: Tables, Views, Synonyms

Select Tables, Views or Synonyms to be associated with operation Select

| Name | Type | Alias | Actions |
|---|---|---|---|
| <current catalog>.<current schema>.EMPLOYEE | TABLE | EMPLOYEE_1 | |

d. If you select multiple tables, select **Joins** to configure the joins for those tables. You can skip this option if you have selected a single table.

e. For **Data Fields**, click **Add Fields**, and select the table columns as shown below:



Step 5: Data Fields

Select data fields to use for the Tables, Views, or Synonyms

□ Select DISTINCT □ Convert all to STRING

| Field/Expression | Output Field Type | Output Name (alias) | Actions |
|---|---|---|---|
| EMPLOYEE_1.EMP_NO | java.math.BigDecimal | EMP_NO_1 | |
| EMPLOYEE_1.EMP_NAME | java.lang.String | EMP_NAME_1 | |
| EMPLOYEE_1.JOB | java.lang.String | JOB_1 | |

Parameters

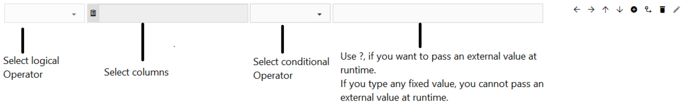
Query Time Out: sec ☑ No Time Out

Maximum Row: ☑ No Limit

Row Count Field Name:

Row Count Field Type:

f. In the **Conditions** screen, configure the WHERE clause of the SQL query as shown below:

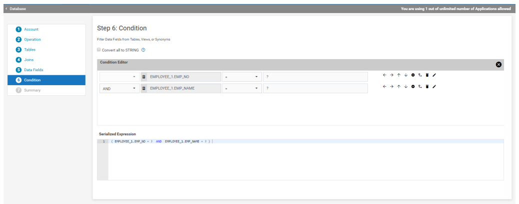


Select logical Operator

Select columns

Select conditional Operator

Use ?, if you want to pass an external value at runtime.
If you type any fixed value, you cannot pass an external value at runtime.



Step 6: Condition

Filter Data Fields from Tables, Views, or Synonyms

□ Convert all to STRING

Condition Editor

| | | | |
|---|---|---|---|
| | EMPLOYEE_1.EMP_NO | = | ? |
| AND | EMPLOYEE_1.EMP_NAME | = | ? |

Serialized Expression

```
( EMPLOYEE_1.EMP_NO = ? AND EMPLOYEE_1.EMP_NAME = ? )
```

**Note:**
The default value for the **Expression** field is ?, which specifies to provide the value for that column when you execute the **Select** operation.

g. The **Summary** displays the operation details such as the name of the operation, account, and SQL query formed. Click **Finish** or **Save**.

d. Execute the **Select** operation.

## DocuSign

Integration Cloud connects to DocuSign using the DocuSign API. It provides electronic signature technology and digital transaction management services for facilitating electronic exchanges of contracts and signed documents.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, for the DocuSign connector version 2, the end point URL is of the format: https://demo.DocuSign.net/restapi. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. DocuSign REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. The Access Token is valid in all future API calls to authenticate the user, until the token is revoked. It is not affected by password changes. |

| Field | Description |
|---|---|
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Electronic Data Interchange

Use the predefined operations in the Electronic Data Interchange (EDI) application to parse, validate, and transform EDI messages received from the webMethods.io B2B application. Use these transformed messages to create orchestrations.

This application requires you to have access to the webMethods.io B2B product instance.

> **Note:**
> Only the documents you add in the webMethods.io B2B product tenant appear on the EDI application page of the corresponding Integration Cloud tenant. And these documents are available across all projects.

### Predefined Operations

The following predefined operations are available for EDI application:

### addGroupEnvelope

Adds a functional group header to an EDI message according to the UN/EDIFACT, ANSI X12, UCS, or VICS standards.

### Input Parameters

| | |
|---|---|
| *ediMessages* | **String** EDI messages to which a group envelope is added. |
| *documentType* | Either **X12/UCS/VICS** or **UN/EDIFACT**. |

*functionalIDcode* **String** Functional ID Code of the EDI document.

*senderQual* **String** EDI ID qualifier for the sender ID. It is used with *sender*.

*sender* **String** Identifies sender in the group envelope. For example, if you specify 01 for *senderQual* (indicating a D-U-N-S number), specify the D-U-N-S number for *sender*.

*receiverQual* **String** EDI ID qualifier for the receiver ID. It is used with *receiver* .

*receiver* **String** Receiver to be identified in the group envelope. For example, if you specify 01 for *receiverQual* (indicating a D-U-N-S number), specify the D-U-N-S number for *receiver* .

*grpCtlNumber* **String** Group control number of the EDI document.

*agencyCode* **String** Responsible agency code as per EDI standard: `T` (default) or `X`.

*verCode* **String** EDI standard version and release code. For example, `4010`, `3040`.

*UN/EDIFACT* **Document** When the document type is **UN/EDIFACT**, assign values to the following parameters:

| Key | Description |
|---|---|
| *syntaxVersion* | **String** Syntax version of the envelope level. |
| *releaseCode* | **String** EDI message standard release code. For example, `96A`, `97b`. |
| *assignedCode* | **String** EDI message standard assigned code. For example, `OD`, `EN`. |
| *password* | **String** The recipient transmission reference password. |

*delimiters* **Document** Delimiters used in the EDI document:

| Key | Description |
|---|---|
| *segment* | String Segment terminator for the EDI document. For example, `\u000a`.<br>**Default** `New line character` |
| *field* | **String** Field separator for each EDI segment. For example, exclamation mark (`!`).<br>**Default** `*` |
| *subfield* | **String** Separator for composite elements. For example, caret (`^`).<br>**Default** **:** |

*release* **String** Release character for composite elements. For example, caret (`^`).

**Default** `?`

## Output Parameters

*ediMessage* **String** Contains the *ediMessage*.

*statusCode* **String** Status code based on success or failure.

*statusMessage* **String** Status message after executing the operation.

## addInterchangeEnvelope

Adds an interchange control header to an EDI message according to the UN/EDIFACT, ANSI X12, UCS, or VICS standards.

## Input Parameters

*ediMessages* **String** EDI messages to which a group envelope is added.

*documentType* Either **X12/UCS/VICS** or **UN/EDIFACT**.

*senderQual* **String** EDI ID qualifier for the sender ID. It is used with *sender*.

*sender* **String** The sender to be identified in the group envelope. For example, if you specify 01 for *senderQual* (indicating a D-U-N-S number), specify the D-U-N-S number for *sender*.

*receiverQual* **String** EDI ID qualifier for the receiver ID. It is used with *receiver*.

*receiver* **String** The receiver to be identified in the group envelope. For example, if you specify 01 for *receiverQual* (indicating a D-U-N-S number), specify the D-U-N-S number for *receiver*.

*ctlNumber* **String** The interchange control number of the EDI document.

*ackRequested* **String** Indicates whether to request an acknowledgment for this interchange:

| Value | Meaning |
|---|---|
| `false` | Do not request an acknowledgment.<br>**Default** `False` |
| `true` | Requests an acknowledgment. |

*delimiters* **Document** Delimiters used in the EDI document.

| Key | Description |
|---|---|
| *segment* | **String** Segment terminator for the EDI document. For example, `\u000a`.<br>**Default** `\` |
| *field* | **String** Field separator for each EDI segment. For example, exclamation mark (`!`).<br>**Default** `*` |
| *subfield* | **String** Separator for composite elements. For example, caret (`^`).<br>**Default** `:` |
| *release* | **String** Release character for composite elements. For example, caret (`^`).<br>**Default** `?` |
| *decimal* | **String** Decimal character for composite elements. For example, caret (`^`). If *UN/EDIFACT > unaSegmentRequired* is `true`, the decimal delimiter is used.<br>**Default** `.`<br>If *UN/EDIFACT > unaSegmentRequired* is `false`, *decimal* is ignored. |

*X12/UCS/VICS* **Document** When the document type is **X12/UCS/VICS**, assign values to the following parameters:

| Key | Description |
|---|---|
| *authQual* | **String** Authorization qualifier for the interchange envelope. |
| *authInfo* | **String** Authorization information for the interchange envelope. |
| *securityQual* | **String** Security qualifier for the interchange envelope. |
| *securityInfo* | **String** Security information for the interchange envelope. |
| *ctlVersion* | **String** Version of the EDI standard used, with a `00` prefix. For example, if version is 4010, specify `004010`. |

*addLeadingZerosToCtlNumber*

**String** (ANSI X12 only) Adds leading zeros to the interchange control number to make it a nine-digit number.

| Value | Meaning |
|---|---|
| `false` | Do not add leading zeros to the interchange control number.<br>**Default** `false` |
| `true` | Add leading zeros to the interchange control number to make it a nine digit number. For example, `12` becomes `000000012`. |

*repSeparator*

**String** A separator for the repeated occurrences of a simple data element or a composite data structure. Field length: 1

> **Note:**
> The *repSeparator* must be different from the *record*, *field*, or *subfield* delimiters.

*messageAlignment*

**String** Aligns the messages.

| Value | Description |
|---|---|
| left | Left justify. |
| right | Right justify. |
| none | No justification.<br>**Default** |

*UN/EDIFACT*

**Document** When the document type is **UN/EDIFACT**, assign values to the following parameters:

| Key | Description |
|---|---|
| *syntaxID* | **String** Syntax identifier. |
| *syntaxVersion* | **String** Syntax version of the envelope level. |
| *password* | **String** Recipient transmission reference password. |
| *passwordQual* | **String** Recipient reference password qualifier. |
| *applReference* | **String** Application reference. |

*priority* **String** Processing priority code.

*agreementID* **String** Interchange agreement identifier.

*unaSegmentRequired* **String** Adds a UNA segment to the resulting final message.

| Value | Description |
|---|---|
| `true` | Create UNA segment. |
| `false` | Do not create UNA segment. |

*directoryVersionNumber* **String** Type the directory version number in the UNB01/S00103 subfield of the EDIFACT envelope interchange header.

*characterEncoding* **String** Type the character encoding in the UNB01/S00104 subfield of the EDIFACT envelope interchange header.

*syntaxReleaseNumber* **String** Type the syntax release number in the UNB01/S00105 subfield of the EDIFACT envelope interchange header.

*senderInternalId* **String** Type the sender's internal ID in the UNB02/S00203 subfield of the EDIFACT envelope interchange header.

*senderInternalSubId* **String** Type the sender's internal sub ID in the UNB02/S00204 subfield of the EDIFACT envelope interchange header.

*receiverInternalId* **String** Type the receiver's internal ID in the UNB03/S00203 subfield of the EDIFACT envelope interchange header.

*receiverInternalSubId* **String** Type the receiver's internal sub ID in the UNB03/S00204 subfield of the EDIFACT envelope interchange header.

## Output Parameters

*ediMessage* **String** Contains the *ediMessage*.

*statusCode* **String** Status code based on success or failure.

*statusMessage* **String** Status message after executing the operation.

## convertDocumentToEDIMessage

Converts a document to an EDI message. When creating an orchestration with **ConvertDocumentToEDIMessage**, you must select the appropriate document type. This is for the structure of an *ediDocument* to appear in the pipeline data when you map with another *ediDocument* during orchestration.

### Input Parameters

*ediDocument* **Document** The *ediDocument* you want to convert to an EDI message.

*documentType* Either **Default** or **TRADACOMS**.

*encoding* **String** The type of encoding used to write a message to the output file.

**Default** `UTF-8`

*delimiters* **Document** Delimiters used in the outbound EDI document.

**Note:**
This parameter is not applicable to **TRADACOMS** messages.

| Key | Description |
|---|---|
| *segment* | **String** Segment terminator character to be appended at the end of each record in the output string. |
| *field* | **String** Field separator to be inserted between each field for each segment. |
| *subfield* | **String** Separator for composite elements. |
| *release* | **String** Release character to use as the escape character for composite elements. |

*messageAlignment* **String** Aligns the messages.

| Value | Description |
|---|---|
| left | Left justify. |
| right | Right justify. |
| none | No justification. |

**Default**

*noEmptyTrailingFields* **String** Indicates whether to remove empty trailing fields from the output.

| Value | Description |
|---|---|

`true` Removes empty trailing fields from the output. For example, AAA*01*02! (where `!` is the segment terminator).

`false` This does not remove empty trailing fields. Instead it uses the field separator to denote an empty field. For example, `AAA*01*02********!`(where * is the field separator and `!` is the segment terminator).

### Output Parameters

*ediMessage* **String** Contains the EDI message.

*errors[]* **String List** Error messages describing the errors encountered during conversion. If there are no errors, then *error[]* is null.

*segmentCount* **String** The number of segments in the *ediMessage*.

*statusCode* **String** Status code based on success or failure.

*statusMessage* **String** Status message after executing the operation.

## convertEDIMessageToDocument

Converts an EDI message that is stream or string into a document. When creating an orchestration with **ConvertEDIMessageToDocument**, you must select the appropriate document type. This is for the structure of an *ediDocument* to appear in the pipeline data when you map with another *ediDocument* during orchestration.

### Input Parameters

*ediMessage* **String** The *ediMessage* you want to convert to a document.

*edistream* **InputStream** The *ediMessage* stream you want to convert to a document.

*documentType* Either **Default** or **TRADACOMS**

*encoding* **String** The encoding of the message passed into an *ediMessage* or *edistream*.

*delimiters* **Document** Delimiters to parse the input message. If no delimiters are specified, then the parameter uses the default delimiter defined for the flat file schema.

> **Note:**
> This parameter is not applicable to **TRADACOMS** messages.

| Key | Description |
|---|---|

*segment* **String** Segment terminator used in the input message.

*field* **String** Field separator used in the input message.

*subfield* **String** Subfield separator used in the input message.

*release* **String** Release character used in the input message.

*validate* **String** Validates the *ediMessage*.

| Value | Description |
|---|---|
| `true` | Return errors describing how the given *ediMessage* violates the constraints described in the flat file schema. |
| `false` | Do not return error messages describing how the *ediMessage* differs from the specified flat file schema. This is the default value. |

*processOnlyTopLevelRecord* **String** Processes the segments one at a time or process all input data at one time. Specify `true` or `false`.

| Value | Meaning |
|---|---|
| **`true`** | Starts processing segment structures with a top-level record as defined by the flat file schema. The parameter returns to the caller when it encounters another top-level record in the input data. |
| `false` | Processes all input data at one time. This is the default value. |

*ignoreSpaces* **String** Ignores white space from the beginning of the segments. Specify `true` or `false`.

| Value | Meaning |
|---|---|
| `true` | Ignore white spaces. This is the default value. |
| `false` | Use the segments as they are. Specify `false` when the data contains positional data records. |

*repeatingFieldSeparator* **String** Inserts field separator between repeating fields of an EDI document.

> **Note:**
> This parameter is not applicable to **TRADACOMS** messages.

## Output Parameters

*ediDocument* **Document** The Document representation of the input *ediMessage*.

*isValid*

**String** Checks the validity of *ediMessage*.

| Value | Description |
|---|---|
| `true` | *validate* input parameter was set to `true` and no errors were found. |
| `false` | *validate* input parameter was set to `true` and errors were found. |

*errors[]*

**Document List** The validation errors, if any, that were found in *edimessage*.Validation errors are returned in *errors* only if *validate* is set to `true` and *returnErrors* is set to `asArray` or `both`. The list includes the path of the errors.

*statusCode*

**String** Status code based on success or failure.

*statusMessage*

**String** Status message after executing the operation.

## processEnvelope

Accepts and processes the envelope of an ANSI X12/UN/EDIFACT/UCS/VICS/EANCOM EDI or **TRADACOMS** message and converts the envelope header segments to a document. When creating an orchestration with **processEnvelope**, you must select the appropriate document type. This is for the structure of an *ediDocument* to appear in the pipeline data when you map with another *ediDocument* during orchestration.

### Input Parameters

*ediMessage*

**String** Processes the input EDI message. Input must not have manual line breaks.

*validate*

**String** Whether to validate the envelopes against a predefined flat file schema.

| Value | Description |
|---|---|
| `true` | Validate the envelopes against a predefined flat file schema. |
| `false` | Does not validate the envelope. |

*complianceCheck*

**String** Performs a compliance check against the interchange.

> **Note:**
> This parameter is not applicable to **TRADACOMS** messages.

| Value | Description |
|---|---|

| | |
|---|---|
| `true` | Perform a compliance check. The parameter stops executing after encountering the first error. This is the default value. |
| `false` | Does not perform the compliance check. |

## Output Parameters

*ediEnvelopeDocument* **Document** Resulting EDI envelope document.

*standard* **String** The standard to which the EDI document adheres. For example, `X12` or `UNEDIFACT`.

*hasError* **String** Whether the validation or compliance check resulted in error.

| Value | Description |
|---|---|
| `false` | If the *validate* is `true`, validation errors are retrieved from *errorArray*.Otherwise, it indicates errors from the compliance check. |
| `true` | No errors. |

*errors[]* **Document List** Validation errors, if any, that were found in *edimessage*.Validation errors are returned in *errors* only if *validate* is set to `true` -AND- *returnErrors* is set to `asArray` or `both`. The list includes the path of the errors.

*statusCode* **String** Status code based on success or failure.

*statusMessage* **String** Status message after executing the operation.

## createTradacomsMessage

Creates a **TRADACOMS** message containing the *STX* header segment, payload, and the *END* segment. When the *validate* parameter is set to `true`, EDI application validates the content of the message.

## Input Parameters

*STX* **Document** The document representation of **TRADACOMS** STX segment.

*includeRSGRSG* **String** Whether the service creates an RSGRSG (reconciliation) message in the output message.

| Value | Description |
|---|---|
| `true` | Creates an RSGRSG message in the output message. This is the default value. |

| | |
|---|---|
| `false` | Does not create an RSGRSG message. |

*encoding* **String** *Optional*. The encoding format to use in the message.

*tradacomsMessages* **String** The **TRADACOMS** payload.

*END* **Document** The document representation of **TRADACOMS** *END* segment.

*validate* **String** Validates the *tradacomsMessages*.

| Value | Description |
|---|---|
| `true` | Returns errors describing how the given *tradacomsMessages* violates the constraints described in the flat file schema. |
| `false` | Does not return error messages describing how the *tradacomsMessages* differs from the specified flat file schema. This is the default value. |

### Output Parameters

*ediMessage* **String** Contains the EDI message.

*errors[]* **String List** Validation errors, if any, that are found in *tradacomsMessages*. Validation errors are returned only if *validate* is set to `true`.

*isValid* **String** Checks the validity of *tradacomsMessages*.

| Value | Description |
|---|---|
| `true` | If the *validate* is set to `true` and no errors are found. |
| `false` | If the *validate* is set to `true` and there are errors. |

*statusCode* **String** Status code based on success or failure.

*statusMessage* **String** Status message after performing the operation.

## File Transfer Protocol (FTP/FTPS)

Integration Cloud connects to an FTP server using the FTP protocol and provides operations to list, download, upload, and delete files. It also supports FTPS (FTP over SSL).

**Note:**
FTP is not a secure file transfer protocol and it has security vulnerabilities. It does not provide any encryption for data transfer. It is recommended to use secured protocols such as HTTPS and FTPS for securing the data transmitted over the network.

**Note:**

See this video on how to create an Account for an FTP Application and test the connection.

| Field | Description |
|---|---|
| **Host** | Host name or IP address or the domain name of the FTP server. |
| **Port** | FTP port defined on the FTP server. |
| **User** | Valid user name on the FTP server. |
| **Password** | Password of the FTP user. |
| **SSL Configuration - Select this option for secured FTP connection.** | |
| **Secure Data** | Select **True** to secure the data channel.<br>Select **False** if you do not want to secure the data channel. |
| **Keystore Alias** | Alias to the keystore that contains the private key used to connect to the host securely. You can also add a new Keystore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Keystores. |
| **Key Alias** | Alias to the key in the keystore that contains the private key used to connect to the host securely. The key must be in the keystore specified in the **Keystore Alias** field. |
| **Truststore Alias** | The alias for the truststore, which contains the trusted root of a certificate or signing authority (CA). You can also add a new Truststore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Truststores. |

## FTP Predefined Operations

The following predefined FTP operations are available:

### getFile

Retrieves a file from a remote FTP server.

## Input Parameters

*remoteFile* **String** Name of the remote file.

*transferType* **String** FTP file transfer mode (ASCII or binary). The default is ASCII.

## Output Parameters

*contentStream* **Object** A java.io.InputStream object.

*statusCode* **String** Standard FTP protocol status code.

*statusMessage* **String** Standard FTP protocol status message.

### listFiles

Returns a list of file names in a specified remote directory. If path is not specified, the operation retrieves the file listing of the current remote directory. The operation also retrieves additional details such as permissions and ownership information.

## Input Parameters

*remotePath* **String** Optional. Absolute or relative path of the remote directory. If *remotepath* is not specified, the listFiles operation retrieves the directory listing of the current remote directory.

You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path.

*listFilter* **String** Optional. Filter that specifies the names of the files to include in the list (for example, *.txt).

## Output Parameters

*fileList []* **String List** List of file names matching *listFilter*.

*statusCode* **String** Standard FTP protocol status code.

*statusMessage* **String** Standard FTP protocol status message.

### deleteFiles

Delete file(s) from a remote FTP server.

### Input Parameters

*remotePath* **String** Optional. Absolute or relative path of the remote directory. If *remotepath* is not specified, the deleteFiles operation deletes the directory listing of the current remote directory.

You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path.

*deleteFilter* **String** Optional. Filter that specifies the names of the files to be deleted (for example, *.txt).

### Output Parameters

*filesDeleted []* **String List** List of deleted files that match the *deleteFilter*.

*filesNotDeleted []* **String List** List of files not deleted.

*statuscode* **String** Standard FTP protocol status code.

*statusmsg* **String** Standard FTP protocol status message.

## putFile

Transfers a file to a remote FTP server.

### Input Parameters

*remoteFile* **String** The name of the remote file.

*transferType* **String** FTP file transfer mode (`ascii` or `binary`). The default is `ascii`.

*writeOption* **String** Optional. Indicates whether to send a STOR or a STOU (Store as Unique File) command to the remote FTP server. Set to:

- `true` to send a STOU (Store as Unique File) command.
- `false` to send a STOR command. This is the default.

*contentStream* **java.io.InputStream, byte[ ], or String** Data to be transferred to the remote file.

### Output Parameters

*statusCode* **String** Standard FTP protocol status code.

*statusMessage* **String** Standard FTP protocol status message.

### Usage Notes

Some FTP servers do not support “putting” a unique file. When using the putFile operation to put a unique file to an FTP server that does not support putting a unique file, you may encounter the following error:

```
com.wm.app.b2b.server.ServiceException: 500 'STOU': command not understood.
```

#### renameFile

Renames a file on a remote FTP server.

### Input Parameters

| | |
|---|---|
| *oldFileName* | **String** Fully qualified name of the file you want to rename (for example, `temp/oldfilename.txt`). |
| *newFileName* | **String** Fully qualified name of the new file (for example, `temp/newfilename.txt`). |

### Output Parameters

| | |
|---|---|
| *StatusCode* | **String** Standard FTP protocol status code. |
| *StatusMessage* | **String** Standard FTP protocol status message. |

## Google Analytics

Integration Cloud connects to Google Analytics using the API for Management services. You can use Management services to retrieve, create, update, and delete analytics configuration data (accounts, metrics, dimensions, and custom data sources).

Integration Cloud also connects to Google Analytics using the API v4 for Core Reporting services. You can use Reporting services to generate customized reports based on dimensions, date range, and metrics.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token.<br>Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect |

| Field | Description |
|---|---|
| | to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Apps Admin

Integration Cloud connects to Google Apps Admin and supports the functionality to create and list users.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/admin. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |

| Field | Description |
|---|---|
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token.<br>Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google BigQuery

Integration Cloud connects to Google BigQuery using the Google BigQuery API and allows you to create, update, and delete data sets and tables. You can also load, copy, extract, and query data from BigQuery's Bigtable.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/admin. |

| Field | Description |
|---|---|
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |

| Field | Description |
|---|---|
| **Refresh URL** | This is the provider specific URL to refresh an Access Token.<br>Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Calendar

Integration Cloud connects to Google Calendar using Google Calendar APIs. It enables you to manage calendar data such as Secondary Calendars, Events, and Quick Event Add.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/calendar/v3. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Contacts

Integration Cloud connects to Google Contacts using Google Contacts APIs. It enables you to manage a user's contact list. The contacts are usually stored in the user's Google Account.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.google.com/m8/feeds. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |

| Field | Description |
|---|---|
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Drive

Integration Cloud connects to Google Drive using the Google Drive API. It provides functionality of file storage and access to list, upload, and delete files.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Cloud Pub/Sub

Integration Cloud connects to Google Cloud Pub/Sub and allows you to create, get, delete, set policy, and get policy on topics and subscription resources.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with Google Cloud Pub/Sub, for example, https://pubsub.googleapis.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |

| Field | Description |
|---|---|
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Issuer** | Applicable when you select the OAuth 2.0 JWT Token Flow as the Authentication Type. This is the Client ID, or Identifier, or name of the server or system issuing the JWT token. |
| **Subject** | Applicable when you select the OAuth 2.0 JWT Token Flow as the Authentication Type. This is the identifier or the name of the user this token represents. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. Google Cloud Pub/Sub REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. The Access Token is valid for 1 hour. It is not affected by password changes. The client application is responsible for storing and protecting this token. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Cloud Storage

Integration Cloud connects to Google Cloud Storage using the Google Cloud Storage API and allows you to create and manage Buckets, Objects, and AccessControls.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with Google Cloud Storage. Example: https://www.googleapis.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |

| Field | Description |
|---|---|
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Prediction

Integration Cloud connects to Google Prediction using the Google Prediction API. It includes the Hosted Model and Trained Model services that are used to predict data by using machine learning.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/prediction/v1.6. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Google Sheets

Integration Cloud connects to Google Sheets and allows you to create, update, and get a spreadsheet, as well as append values to a spreadsheet.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://sheets.googleapis.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Consumer ID** | Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server. |

| Field | Description |
|---|---|
| **Consumer Secret** | Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier. |
| **Access Token** | Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## IBM Watson Tone Analyzer

Integration Cloud connects to IBM Watson Tone Analyzer using the REST interface to detect emotional, social, and language tones in written text. You can use the Application to learn the tone of your customer's communications and to respond to each customer appropriately, or to understand and improve customer conversations.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | IBM Bluemix Watson username. |
| **Password** | IBM Bluemix Watson password. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic** . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. Select the Authorization Type as **basic**. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS |

| Field | Description |
|---|---|
| | provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Magento eCommerce Platform

Integration Cloud connects to Magento using the Magento REST API. You can use it to manage customers, customer addresses, sales orders, inventory, products, and so on, without having to directly work on Magento.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: http://<yourhost>/api/rest. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |

| Field | Description |
|---|---|
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. |
| **Access Token Secret** | A secret used by the Consumer to establish ownership of a given Access Token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Marketo

Integration Cloud connects to Marketo using the Marketo REST API and allows you to create, retrieve, and remove entities and data stored within Marketo.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<instance>.mktorest.com. Replace <instance> with your actual Marketo instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, |

| Field | Description |
|---|---|
| | increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in |

| Field | Description |
|---|---|
| | the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Microsoft Dynamics CRM

Integration Cloud connects to **Microsoft Dynamics CRM** using the Microsoft Dynamics CRM SOAP API. You can manage CRM data and access metadata that defines the specific CRM instance to which you are connecting.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<organization>.api.crm.dynamics.com/<br>XRMServices/2011/Organization.svc, where <organization> must be replaced with your actual Microsoft Dynamics CRM organization. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |

| Field | Description |
|---|---|
| | The Account will use this credential to connect to the SaaS provider. |
| **Password** | Provide a password for the user name to initiate communication with the SaaS provider. |
| **Authorization Type** | This is the type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you |

| Field | Description |
|---|---|
| | must specify values in both the Keystore Alias and the Client Key Alias fields. |

## Microsoft Dynamics CRM 365

Integration Cloud connects to **Microsoft Dynamics CRM 365** using the OData API Version 4.0 and allows you to manage CRM data and access metadata that defines the specific CRM instance to which you are connecting. This Application performs standard CRUD operations on business objects by connecting to the OData service endpoint.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, for Microsoft Dynamics CRM 365, the end point URL is the OData service endpoint. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |

| Field | Description |
|---|---|
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Caching** | Select this option if you want the Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in |

| Field | Description |
|---|---|
| | the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## OData v2.0

Integration Cloud connects to any cloud application that exposes its services using the OData Version 2.0 Specification. It supports only those OData providers, which strictly adhere to the OData Version 2.0 Specification and allows you to perform standard CRUD operations on business objects by connecting to the OData service endpoint.

| Field | Description |
|---|---|
| **Server URL** | This is the OData service endpoint to initiate communication with the OData provider. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the OData provider that the Account will use to connect to the OData provider. |
| **Password** | Provide the password for the user name provided in the **Username** field. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a **Username** and **Password**, and also select **none**, you do not specify a value for the **Authorization Type**, so the user credentials are not inserted into an Authorization header. If you enter the **Username** and **Password**, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. If you enter the username and password, then set the authorization type as **basic**. |

| Field | Description |
|---|---|
| **Caching** | Select this option if you want the OData v2.0 Application to cache the backend metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## OData v4.0

Integration Cloud connects to any cloud application that exposes its services using the OData Version 4.0 Specification. It supports only those OData providers, which strictly adhere to the OData Version 4.0 Specification and allows you to perform standard CRUD operations on business objects by connecting to the OData service endpoint.

| Field | Description |
|---|---|
| **Server URL** | This is the OData service endpoint to initiate communication with the OData provider. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Caching** | Select this option if you want the OData v4.0 Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache will be refreshed |

| Field | Description |
|---|---|
| | every 12 hours. It is recommended to enable the cache to increase the performance. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Salesforce

### Salesforce CRM

Integration Cloud connects to Salesforce using the Partner SOAP API. It supports all business objects (for example, Account) and operations including any customizations done on the Salesforce instance. It also supports Salesforce analytics using wave.

**Note:**
Click “here” on page 364 for answers to some of the most common questions on Account configuration.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://login.salesforce.com/services/Soap/u/44.0. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |
| **Password** | The password for the user name provided in the Username field. When you access Salesforce from outside your company's trusted networks, you must append a security token (provided by Salesforce) to your password. Your security token may have been emailed to you when you had set up your Salesforce account or if you had reset your password. For example, if your password is *abc* and your security token is *xxxx*, then you must enter *abcxxxx*. For more information about logging in to Salesforce, see the Salesforce documentation. |
| **JWT Keystore** | The keystore used to encrypt the JWT payload.<br>Use the same keystore which contains the private key of the certificate (Public keys) uploaded in your Integration in Adobe Experience Platform. |
| **JWT Key Alias** | This alias is the value that is used to sign the outgoing request from Integration Cloud to the authentication server. It is auto-populated based on the keystore selected in the JWT Keystore field. This field lists all the aliases available in the chosen keystore. You must provide a key alias to sign the JWT payload. |
| **Expiration Time(mins)** | Expiration Time (mins) is the time after which the JWT token expires. The generated access token might be valid post expiration time as well. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. The access token is passed when you invoke any of the REST API endpoints. The client application is responsible for storing and protecting this token.<br>Integration Cloud will get an Access Token using the JSON Web Token (JWT) Flow after you save the Account. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, |

| Field | Description |
|---|---|
| | the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Session Timeout (min)** | The number of minutes you want Integration Cloud to wait before terminating an idle session. The value should be equal to the session time-out value specified at the SaaS provider back end. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Connection TimeOut** | The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. It is recommended that you specify a value other than 0 to avoid using a socket with no timeout. |
| **Connection Retry Count** | The number of times the system should attempt to initialize the connection at startup if the initial attempt fails.<br>The system retries to establish a connection when an I/O error occurs while sending the request message to the back end. If an I/O exception occurs when the system is reading a response back from the back end, the system will only retry if **Retry on Response Failure** is enabled. |
| **Issuer** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the Client ID, or Identifier, or name of the server or system issuing the JWT token. |
| **Subject** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the identifier or the name of the user this token represents. |
| **Consumer ID** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. Also referred to as the Client ID, this is a client identifier |

| Field | Description |
|---|---|
| | issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. This is the provider specific URL to refresh an Access Token. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |

| Field | Description |
|---|---|
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |
| **Enable Connection Pooling** | Select this option if you want to enable connection pooling for a connection.<br>Integration Cloud includes a connection management service that dynamically manages connections and connection pools based on configuration settings that you specify for the connection. A connection pool is a collection of connections with the same set of attributes. Connection pools improve performance by enabling Integrations to reuse open connections instead of opening new connections for every service request.<br>When you enable connection pooling, Integration Cloud creates the number of connection instances you specified in the connection's Minimum Pool Size field. Whenever an Integration needs a connection, Integration Cloud provides a connection from the pool. If no connections are available in the pool, and the Maximum Pool Size has not been reached, Integration Cloud creates one or more new connections (according to the number specified |

| Field | Description |
|---|---|
| | in the Pool Increment Size field) and adds them to the connection pool.<br>If the pool is full (as specified in the Maximum Pool Size field), the requesting service will wait for Integration Cloud to obtain a connection till one sec, until a connection becomes available. Periodically, Integration Cloud inspects the pool and removes inactive connections that have exceeded the expiration period of one sec. |
| **Minimum Pool Size** | The minimum number of connection objects that remain in the connection pool at all times, if connection pooling is enabled. When the connector creates the pool, it creates this number of connections. |
| **Maximum Pool Size** | The maximum number of connection objects that can exist in the connection pool if connection pooling is enabled. When the connection pool has reached its maximum number of connections, the connector will reuse any inactive connections in the pool, or, if all connections are active, it will wait for a connection to become available. |
| **Pool Increment Size** | The number of connections by which the pool will be incremented, up to the maximum pool size, if connection pooling is enabled and connections are needed. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Grant Type** | Specify the grant type through which applications can gain Access Tokens and by which you grant limited access to your resources to another entity without exposing credentials. The Authorization Code grant type is used by confidential and public clients to exchange an authorization code for an access token. The Refresh Token grant type is used by clients to exchange a refresh token for an access token when the access token has expired. |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the |

| Field | Description |
|---|---|
| | specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Block Timeout (msec)** | The number of milliseconds that Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error.<br>For example, you have a pool with Maximum Pool Size of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the Block Timeout to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available.<br>If you set the Block Timeout value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the Maximum Pool Size to accommodate such bursts in processing.<br>Default: 1000 msec |
| **Expire Timeout (msec)** | The number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled.<br>The connection pool will remove inactive connections until the number of connections in the pool is equal to the Initial Pool Size. The inactivity timer for a connection is reset, when the connection is used by the Application.<br>This setting should be tuned in conjunction with the Initial Pool Size to avoid excessive opening and closing of connections during normal processing.<br>The general recommendation is to keep the Expire Timeout value equal to the Session Timeout value.<br>Default: 1000 msec |

### Salesforce Bulk Data Loader

Integration Cloud connects to Salesforce using the Salesforce Bulk API and supports Job and Batch resources. You can use it to create, update, delete, query jobs and batches, and operate on large number of records asynchronously by submitting batches which are processed in the background by Salesforce.

> **Note:**
> Click "here" on page 364 for answers to some of the most common questions on Account configuration.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://login.salesforce.com/services/Soap/u/31.0. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Issuer** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the Client ID, or Identifier, or name of the server or system issuing the JWT token. |
| **Subject** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the identifier or the name of the user this token represents. |
| **JWT Keystore** | The keystore used to encrypt the JWT payload.<br>Use the same keystore which contains the private key of the certificate (Public keys) uploaded in your Integration in Adobe Experience Platform. |
| **JWT Key Alias** | This alias is the value that is used to sign the outgoing request from Integration Cloud to the authentication server. It is auto-populated based on the keystore selected in the JWT Keystore field. This field lists all the aliases available in the chosen keystore. You must provide a key alias to sign the JWT payload. |
| **Expiration Time(mins)** | Expiration Time (mins) is the time after which the JWT token expires. The generated access token might be valid post expiration time as well. |

| Field | Description |
|---|---|
| **Connection TimeOut** | The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. It is recommended that you specify a value other than 0 to avoid using a socket with no timeout. |
| **Connection Retry Count** | The number of times the system should attempt to initialize the connection at startup if the initial attempt fails.<br>The system retries to establish a connection when an I/O error occurs while sending the request message to the back end. If an I/O exception occurs when the system is reading a response back from the back end, the system will only retry if **Retry on Response Failure** is enabled. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. The access token is passed when you invoke any of the REST API endpoints. The client application is responsible for storing and protecting this token.<br>Integration Cloud will get an Access Token using the JSON Web Token (JWT) Flow after you save the Account. |
| **Consumer ID** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | Applicable when you select the **OAuth V2.0 (Authorization Code Flow)** as the Authentication Type. This is the provider specific URL to refresh an Access Token. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |

| Field | Description |
|---|---|
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |
| **Password** | The password for the user name provided in the Username field. When you access Salesforce from outside your company's trusted networks, you must append a security token (provided by Salesforce) to your password. Your security token may have been emailed to you when you had set up your Salesforce account or if you had reset your password. For example, if your password is *abc* and your security token is *xxxx*, then you must enter *abcxxxx*. For more information about logging in to Salesforce, see the Salesforce documentation. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Session Timeout (min)** | The number of minutes you want Integration Cloud to wait before terminating an idle session. The value should be equal to the session timeout value specified at the SaaS provider back end. |
| **Grant Type** | Specify the grant type through which applications can gain Access Tokens and by which you grant limited access to your resources to another entity without exposing credentials. The Authorization Code grant type is used by confidential and public clients to exchange an authorization code for an access token. The Refresh Token grant type is used by clients to exchange a refresh token for an access token when the access token has expired. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |

| Field | Description |
|---|---|
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client’s identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable Connection Pooling** | Select this option if you want to enable connection pooling for a connection.<br>Integration Cloud includes a connection management service that dynamically manages connections and connection pools based on configuration settings that you specify for the connection. A connection pool is a collection of connections with the same set of attributes. Connection |

| Field | Description |
|---|---|
| | pools improve performance by enabling Integrations to reuse open connections instead of opening new connections for every service request.<br>When you enable connection pooling, Integration Cloud creates the number of connection instances you specified in the connection's Minimum Pool Size field. Whenever an Integration needs a connection, Integration Cloud provides a connection from the pool. If no connections are available in the pool, and the Maximum Pool Size has not been reached, Integration Cloud creates one or more new connections (according to the number specified in the Pool Increment Size field) and adds them to the connection pool.<br>If the pool is full (as specified in the Maximum Pool Size field), the requesting service will wait for Integration Cloud to obtain a connection till one sec, until a connection becomes available. Periodically, Integration Cloud inspects the pool and removes inactive connections that have exceeded the expiration period of one sec. |
| **Minimum Pool Size** | The minimum number of connection objects that remain in the connection pool at all times, if connection pooling is enabled. When the connector creates the pool, it creates this number of connections. |
| **Maximum Pool Size** | The maximum number of connection objects that can exist in the connection pool if connection pooling is enabled. When the connection pool has reached its maximum number of connections, the connector will reuse any inactive connections in the pool, or, if all connections are active, it will wait for a connection to become available. |
| **Pool Increment Size** | The number of connections by which the pool will be incremented, up to the maximum pool size, if connection pooling is enabled and connections are needed. |
| **Block Timeout (msec)** | The number of milliseconds that Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error.<br>For example, you have a pool with Maximum Pool Size of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the Block Timeout to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to |

| Field | Description |
|---|---|
| | the pool, the pending requests will fail and return an error message stating that no connections are available.<br>If you set the Block Timeout value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the Maximum Pool Size to accommodate such bursts in processing.<br>Default: 1000 msec |
| **Expire Timeout (msec)** | The number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled.<br>The connection pool will remove inactive connections until the number of connections in the pool is equal to the Initial Pool Size. The inactivity timer for a connection is reset, when the connection is used by the Application.<br>This setting should be tuned in conjunction with the Initial Pool Size to avoid excessive opening and closing of connections during normal processing.<br>The general recommendation is to keep the Expire Timeout value equal to the Session Timeout value.<br>Default: 1000 msec |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Salesforce® Bulk v2 Data Loader

Using the Bulk API 2.0, Salesforce supports Job resource, and allows you to create, update, delete, upsert jobs, and operate on large number of records asynchronously by submitting jobs which are processed in the background.

**Note:**
Click “here” on page 364 for answers to some of the most common questions on Account configuration.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, https://<instance>.salesforce.com. Replace <instance> with your actual Salesforce instance. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **JWT Keystore** | The keystore used to encrypt the JWT payload.<br>Use the same keystore which contains the private key of the certificate (Public keys) uploaded in the *Digital Certificate* section on your *Connected Apps* in Salesforce. |
| **JWT Key Alias** | This alias is the value that is used to sign the outgoing request from Integration Cloud to the authentication server. It is auto-populated based on the keystore selected in the JWT Keystore field. This field lists all the aliases available in the chosen keystore. You must provide a key alias to sign the JWT payload. |
| **Expiration Time(mins)** | Expiration Time (mins) is the time after which the JWT token expires. The generated access token might be valid post expiration time as well. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. Salesforce REST APIs use OAuth 2.0. The access token is passed when you invoke any of the REST API endpoints and is valid for one hour. It is not affected by password changes. The client application is responsible for storing and protecting this token. You can manage the Salesforce REST connection by enabling the connection pool and session management. |

| Field | Description |
|---|---|
| | If you have selected the *OAuth V2.0 (JWT Flow)* as the *Authentication Type*, Integration Cloud will get an Access Token using the JWT flow after you save the Account. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token, for example, https://<instance>.salesforce.com/services/oauth2/token. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Session Timeout (min)** | The number of minutes you want Integration Cloud to wait before terminating an idle session. The value should be equal to the session timeout value specified at the SaaS provider back end. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |

| Field | Description |
|---|---|
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Grant Type** | Specify the grant type through which applications can gain Access Tokens and by which you grant limited access to your resources to another entity without exposing credentials. The Authorization Code grant type is used by confidential and public clients to exchange an authorization code for an access token. The Refresh Token grant type is used by clients to exchange a refresh token for an access token when the access token has expired. |
| **Block Timeout (msec)** | The number of milliseconds that Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error. |

| Field | Description |
|---|---|
| | For example, you have a pool with Maximum Pool Size of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the Block Timeout to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available.<br>If you set the Block Timeout value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the Maximum Pool Size to accommodate such bursts in processing.<br>Default: 1000 msec |
| **Expire Timeout (msec)** | The number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled.<br>The connection pool will remove inactive connections until the number of connections in the pool is equal to the Initial Pool Size. The inactivity timer for a connection is reset, when the connection is used by the Application.<br>This setting should be tuned in conjunction with the Initial Pool Size to avoid excessive opening and closing of connections during normal processing.<br>The general recommendation is to keep the Expire Timeout value equal to the Session Timeout value.<br>Default: 1000 msec |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |

## Salesforce® CRM REST

Integration Cloud connects to Salesforce® CRM REST using the REST API and allows you to manage security for inbound requests, log payloads and specify run-time performance conditions for consumers for outbound requests. It also supports multiple authentication mechanisms.

**Note:**
Click "here" on page 364 for answers to some of the most common questions on Account configuration.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, https://<instance>.salesforce.com. Replace <instance> with your actual Salesforce instance. |
| **Issuer** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the Client ID, or Identifier, or name of the server or system issuing the JWT token. |
| **Subject** | Applicable when you select the **OAuth V2.0 (JWT Flow)** as the Authentication Type. This is the identifier or the name of the user this token represents. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token, for example, https://login.salesforce.com/services/oauth2/token. |
| **JWT Keystore** | The keystore used to encrypt the JWT payload.<br>Use the same keystore which contains the private key of the certificate (Public keys) uploaded in the *Digital Certificate* section on your *Connected Apps* in Salesforce. |
| **JWT Key Alias** | This alias is the value that is used to sign the outgoing request from Integration Cloud to the authentication server. It is auto-populated based on the keystore selected in the JWT Keystore field. This field lists all the aliases available in the chosen keystore. You must provide a key alias to sign the JWT payload. |
| **Expiration Time(mins)** | Expiration Time (mins) is the time after which the JWT token expires. The generated access token might be valid post expiration time as well. |

| Field | Description |
|---|---|
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. Salesforce REST APIs use OAuth 2.0. The access token is passed when you invoke any of the REST API endpoints and is valid for one hour. It is not affected by password changes. The client application is responsible for storing and protecting this token. You can manage the Salesforce REST connection by enabling the connection pool and session management.<br>If you have selected the *OAuth V2.0 (JWT Flow)* as the *Authentication Type*, Integration Cloud will get an Access Token using the JWT flow after you save the Account. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Session Timeout (min)** | The number of minutes you want Integration Cloud to wait before terminating an idle session. The value should be equal to the session timeout value specified at the SaaS provider back end. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Enable Connection Pooling** | Select this option if you want to enable connection pooling for a connection.<br>Integration Cloud includes a connection management service that dynamically manages connections and connection pools based on configuration settings that you specify for the connection. A connection pool is a collection |

| Field | Description |
|---|---|
| | of connections with the same set of attributes. Connection pools improve performance by enabling Integrations to reuse open connections instead of opening new connections for every service request.<br>When you enable connection pooling, Integration Cloud creates the number of connection instances you specified in the connection's Minimum Pool Size field. Whenever an Integration needs a connection, Integration Cloud provides a connection from the pool. If no connections are available in the pool, and the Maximum Pool Size has not been reached, Integration Cloud creates one or more new connections (according to the number specified in the Pool Increment Size field) and adds them to the connection pool.<br>If the pool is full (as specified in the Maximum Pool Size field), the requesting service will wait for Integration Cloud to obtain a connection till one sec, until a connection becomes available. Periodically, Integration Cloud inspects the pool and removes inactive connections that have exceeded the expiration period of one sec. |
| **Minimum Pool Size** | The minimum number of connection objects that remain in the connection pool at all times, if connection pooling is enabled. When the connector creates the pool, it creates this number of connections. |
| **Maximum Pool Size** | The maximum number of connection objects that can exist in the connection pool if connection pooling is enabled. When the connection pool has reached its maximum number of connections, the connector will reuse any inactive connections in the pool, or, if all connections are active, it will wait for a connection to become available. |
| **Pool Increment Size** | The number of connections by which the pool will be incremented, up to the maximum pool size, if connection pooling is enabled and connections are needed. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed |

| Field | Description |
|---|---|
| | certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client’s identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Connection TimeOut** | The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. It is recommended that you specify a value other than 0 to avoid using a socket with no timeout. |
| **Connection Retry Count** | The number of times the system should attempt to initialize the connection at startup if the initial attempt fails.<br>The system retries to establish a connection when an I/O error occurs while sending the request message to the back end. If an I/O exception occurs when the system is reading a response back from the back end, the system will only retry if **Retry on Response Failure** is enabled. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Grant Type** | Specify the grant type through which applications can gain Access Tokens and by which you grant limited access to your resources to another entity without exposing credentials. The Authorization Code grant type is used by confidential and public clients to exchange an authorization code for an access token. The Refresh Token grant type is used by clients to exchange a refresh token for an access token when the access token has expired. |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Block Timeout (msec)** | The number of milliseconds that Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error. |

| Field | Description |
|---|---|
| | For example, you have a pool with Maximum Pool Size of 20. If you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. If you set the Block Timeout to 5000, the 10 requests will wait for a connection for 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available.<br>If you set the Block Timeout value too high, you may encounter problems during error conditions. If a request contains errors that delay the response, other requests will not be sent. This setting should be tuned in conjunction with the Maximum Pool Size to accommodate such bursts in processing.<br>Default: 1000 msec |
| **Expire Timeout (msec)** | The number of milliseconds that an inactive connection can remain in the pool before it is closed and removed from the pool, if connection pooling is enabled.<br>The connection pool will remove inactive connections until the number of connections in the pool is equal to the Initial Pool Size. The inactivity timer for a connection is reset, when the connection is used by the Application.<br>This setting should be tuned in conjunction with the Initial Pool Size to avoid excessive opening and closing of connections during normal processing.<br>The general recommendation is to keep the Expire Timeout value equal to the Session Timeout value.<br>Default: 1000 msec |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in |

| Field | Description |
|---|---|
| | the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## SAP Cloud for Customer(C4C) OData v2.0

Integration Cloud connects to SAP Cloud for Customer (C4C) including SAP Cloud for Sales, SAP Cloud for Service, and SAP Cloud for Social Engagement solutions using the REST interface, and allows you to do standard CRUD operations on business objects by connecting to the OData Service endpoint.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. This is the OData service endpoint to initiate communication with the SAP C4C OData provider. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SAP C4C OData provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider. |
| **Password** | Provide the password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a **Username** and **Password**, and also select **none**, you do not specify a value for the **Authorization Type**, so the user credentials are not inserted into an Authorization header. If you enter the **Username** and **Password**, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. |

| Field | Description |
|---|---|
| | This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Metadata Caching** | Select this option if you want the SAP C4C Application to cache the backend metadata. Caching of the metadata significantly increases the performance of a request sent through SAP C4C. If this option is selected, the cache will be refreshed every 12 hours. It is recommended to enable the metadata cache to increase the performance. |
| **Use CSRF Token** | To prevent cross site request forgery, SAP C4C protects its resources by using a CSRF token. Select this option if you want Integration Cloud to use the CSRF token key, received in the response from SAP C4C, to perform any state changing requests on SAP C4C. By default, the CSRF token is enabled by the SAP C4C back end. You must enable this option particularly when the entity state changing operation is invoked. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in |

| Field | Description |
|---|---|
| | the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## SAP S/4HANA Marketing Cloud

Integration Cloud connects to SAP S/4HANA Marketing Cloud using the OData based REST interface, which allows you to do only bulk imports. You can create or update Interaction Contacts, Interactions, Interests, Corporate Accounts, Product Categories, Products, and so on.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. This is the OData service endpoint to initiate communication with the SAP S/4HANA Marketing Cloud provider. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SAP S/4HANA Marketing Cloud provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider. |
| **Password** | Provide the password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a **Username** and **Password**, and also select **none**, you do not specify a value for the **Authorization** |

| Field | Description |
|---|---|
| | **Type**, so the user credentials are not inserted into an Authorization header. If you enter the **Username** and **Password**, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Metadata Caching** | Select this option if you want the SAP S/4HANA Marketing Cloud Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent through SAP S/4HANA Marketing Cloud. If this option is selected, the cache will be refreshed every 12 hours. It is recommended to enable the metadata cache to increase the performance. |
| **Validate Metadata** | Whether to validate the $metadata xml during edm object creation. Select this option to enable the metadata validation. |
| **Use CSRF Token** | To prevent cross site request forgery, SAP S/4HANA Marketing Cloud protects its resources by using a CSRF token. Select this option if you want Integration Cloud to use the CSRF token key, received in the response from SAP S/4HANA Marketing Cloud, to perform any state changing requests on SAP S/4HANA Marketing Cloud. By default, the CSRF token is enabled by the SAP S/4HANA Marketing Cloud back end. You must enable this option particularly when the entity state changing operation is invoked. |
| **Use Chunking** | Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote |

| Field | Description |
|---|---|
| | server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Secure File Transfer Protocol (SFTP)

The SSH File Transfer Protocol (SFTP) is a network protocol that is based on the Secure Shell protocol (SSH). SFTP facilitates secure file access, file transfer, and file management over any reliable data stream. The Secure File Transfer Protocol (SFTP) Application downloads files from or uploads files to an SFTP-enabled server using the secure file transport channel.

Integration Cloud connects to an SFTP server using the SSH File Transfer Protocol (SFTP) and provides operations to retrieve, transfer, rename, and delete files or directories in the SFTP server. You can also change the permission or ownership of files in the SFTP server.

You can configure Integration Cloud to connect to an SFTP server to perform the following tasks using the SFTP protocol:

- Transfer files between Integration Cloud and the SFTP server. You can get a file from the SFTP server or upload a file to the SFTP server.
- Access files in the SFTP server. You can view the directories and files in the SFTP server and also view their permissions and ownership information.
- Manage directories or files in the SFTP server. You can create, rename, or delete files or directories in the SFTP server. You can also change the permissions or ownership of files in the SFTP server.

| For this parameter... | Specify... |
|---|---|
| **Host Name or IP Address** | The host name or IP address of the SFTP server. |
| **Port Number** | The port number of the SFTP server. The port number must be within the range of 1 and 65535 (inclusive). |

| For this parameter... | Specify... |
|---|---|
| **Host Public Key** | The public key of the SFTP server. Select **Auto Retrieve** if you want Integration Cloud to automatically retrieve the public key of the SFTP server. Select **Upload** if you have the public key of the SFTP server. Integration Cloud will use the uploaded public key. |
| **Finger Print** | The host public key fingerprint of the SFTP server.<br>**Note:** This field is visible only after you have established a connection to an SFTP server.<br>Before establishing a connection, the SFTP server sends an encrypted fingerprint of its host public keys to ensure that the SFTP connection will be exchanging data with the correct server. Save the fingerprint information locally. This enables you to check the fingerprint information against the data you have saved every time you establish a new connection. |
| **User Name** | The user name for the SFTP user account. |
| **Authentication Type** | The type of authentication Integration Cloud uses to authenticate itself to the SFTP server. Client authentication can be either by password or by public and private keys. Select **Password** if you want to use password authentication. If you are using password authentication, enter the password for the specified user to connect to the SFTP server. Select **Public Key** if you want to authenticate Integration Cloud by using public and private keys. To use this authentication type, the SFTP server and Integration Cloud must each have access to their own private key and each other's public key. |
| **Private Key** | If you selected **Public Key** as the authentication type, select the private key file of the specified SFTP user. |
| **PassPhrase** | If you selected **Public Key** as the authentication type and if the private key you specified requires a passphrase, enter the passphrase for the private key file of the specified user. |
| **Advanced Options** | |
| **Maximum Retries** | The number of times Integration Cloud attempts to connect to the SFTP server. The maximum allowed value is 6. The minimum allowed value is 1. |
| **Response Timeout** | The amount of time (measured in seconds) Integration Cloud waits for a response from the SFTP server before timing out and terminating the request. A value of 0 indicates that the session will never time out. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. |
| **Session Timeout** | The number of minutes you want Integration Cloud to wait before terminating an idle session. The session timeout value must be within the range of 10 and 60 minutes. |

| For this parameter... | Specify... |
|---|---|
| **Preferred Key Exchange Algorithms** | The algorithms that Integration Cloud presents to the SFTP server for key exchange. You can specify the order in which Integration Cloud presents the algorithms to the SFTP server by moving the available algorithms up or down by clicking **Move Up** or **Move Down**. The SFTP server has its own set of preferred algorithms configured. At the time of key exchange, one of the algorithms supported by both Integration Cloud and the SFTP server will be chosen. |
| **Compression** | Whether or not to compress the data to reduce the amount of data that is transmitted. Integration Cloud supports compression using the compression algorithm zlib. You can use compression only if the SFTP server that you are connecting to supports compression. Select **None** if you do not want to compress the data. Select **zlib** if you want to compress the data that is transmitted between the SFTP server and Integration Cloud. |
| **Compression Level** | The compression level to use if you selected the compression algorithm **zlib** in the **Compression** field. The minimum allowed value is 1 (fast, less compression) and the maximum allowed value is 9 (slow, most compression). |

## SFTP Predefined Operations

The following predefined SFTP operations are available:

### cd

Changes the working directory on the remote SFTP server.

#### Input Parameters

| | |
|---|---|
| *path* | **String** Absolute or relative path of the directory that you want as the working directory on the remote SFTP server. |

#### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### chgrp

Changes the group ownership of one or more remote files.

## Input Parameters

| | |
|---|---|
| *groupId* | **String** Numeric group identifier of the group to which you want to transfer ownership of the remote files. |
| *path* | **String** Absolute or relative path of the remote files. |

## Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### chmod

Changes permissions of one or more remote files.

## Input Parameters

| | |
|---|---|
| *mode* | **String** The permission mode to apply to the remote file (for example, `777`). |
| *path* | **String** Absolute or relative path of the remote files. |

## Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### chown

Changes the owning user of one or more remote files.

## Input Parameters

| | |
|---|---|
| *uid* | **String** Numeric user ID of the new owning user of the file. |
| *path* | **String** Absolute or relative path of the remote files. |

## Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |

| | |
|---|---|
| *returnMsg* | **String** Text message describing the return code. |

### get

Retrieves a file from a remote SFTP server.

#### Input Parameters

| | |
|---|---|
| *remoteFile* | **String** Absolute or relative path of the remote file. |

#### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |
| *contentStream* | **Object** A java.io.InputStream object. |

### ls

Retrieves the remote directory listing of the specified path. If path is not specified, the ls service retrieves the file listing of the current remote directory. The ls service also retrieves additional details such as permissions and ownership information.

#### Input Parameters

| | |
|---|---|
| *path* | **String** Optional. Absolute or relative path of the remote directory. If no *path* is specified, the ls service retrieves the directory listing of the current remote directory.<br>You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path. |

#### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |
| *dirList* | **Document** List of directories matching the pattern specified in the *path* parameter. This document has the following parameters:<br>fileName: **String** Specifies the name of the remote file. . |

fileSize: **String** Specifies the size of the remote file.

permissions: **String** Specifies the access permission of the file (read, write, or execute).

lastAccessTime: **String** Specifies the time when the file was last accessed.

lastModifiedTime: **String** Specifies the time when the file was last modified.

uid: **String** Specifies the user ID of the file owner.

gid: **String** Specifies the group ID associated with the file.

longName: **String** Specifies the long name of the *ls* entry. It contains all the parameters separated by a space.

### mkdir

Creates a new remote directory.

#### Input Parameters

| | |
|---|---|
| *path* | **String** Absolute or relative path of the remote directory where you want to create a new directory. |

#### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### put

Transfers a file to a remote SFTP server.

#### Input Parameters

*contentStream* — **java.io.InputStream** Data to be transferred to the remote file.

*remoteFile* — **String** Absolute or relative path of the remote file to which the *contentStream* would be written based on the *mode*.

*mode* — **String** Optional. Specifies how the local file is to be transferred to the remote SFTP server. Set to:

- `overwrite` to overwrite the contents of the remote file with the contents of the *contentStream*. This is the default.

- `append` to append the entire contents of the *contentStream* to the remote file.
- `resume` to resume writing the contents of the *contentStream* to the remote file from the point the writing was stopped during previous SFTP sessions.

### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

## pwd

Displays the remote working directory in the SFTP server.

### Input Parameters

None.

### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |
| *path* | **String** Absolute or relative path of the working directory on the remote SFTP server. |

## rename

Renames a file or directory on a remote SFTP server.

### Input Parameters

| | |
|---|---|
| *oldPath* | **String** Fully qualified name of the file you want to rename (for example, `temp/oldname.txt`). |
| *newPath* | **String** New fully qualified name for the file (for example, `temp/newname.txt`). |

### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### rm

Deletes one or more remote files on the SFTP server.

## Input Parameters

| | |
|---|---|
| *path* | **String** Absolute or relative path of the file you want to delete. |

## Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

### rmdir

Deletes one or more remote directories on the SFTP server.

## Input Parameters

| | |
|---|---|
| *path* | **String** Absolute or relative path of the directory you want to delete. |

## Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

## Usage Notes

The remote directories that you want to delete must be empty.

### symlink

Creates a symbolic link between the old path and the new path of a file.

## Input Parameters

| | |
|---|---|
| *oldPath* | **String** Old path of the file for which you want to create a symbolic link. |
| *newPath* | **String** New path of the file to which the symbolic link should point. |

### Output Parameters

| | |
|---|---|
| *returnCode* | **String** Standard SFTP protocol return code. |
| *returnMsg* | **String** Text message describing the return code. |

## ServiceNow Enterprise Service Management

Integration Cloud connects to different areas (Incident, Problem, and Change management) of ServiceNow using the Geneva version of the ServiceNow API. You can create incidents, get details of created incidents, and update and delete them. Similar operations are available for problem and change management cloud applications.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://<instance_name>.service-now.com, where <instance_name> represents the actual instance name. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. |
| **Password** | Provide a password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if |

| Field | Description |
|---|---|
| | the Application requires or supports HTTP Basic authentication using a username and password. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Shopify

Integration Cloud connects to Shopify using the Shopify REST API and allows you to organize your products, customize your storefront, accept credit card payments, track, and respond to orders.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<my-store>.myshopify.com. Replace <my-store> with your actual Shopify store. |
| **Shopify Access Token** | This token is used for authentication and is issued by the Authorization Server. The Access Token is passed when you invoke any of the REST API endpoints. Shopify REST APIs authentication requests require a unique API key generated in Shopify. All API requests must pass an X-Shopify-Access-Token header with an API key. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect |

| Field | Description |
|---|---|
| | to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Siemens MindSphere

Integration Cloud connects to Siemens MindSphere using the API version 2.0 and allows you to create aspects and post data into a MindSphere asset.

**Note:**
The fields displayed may vary according to the version of the Application.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, https://mindconnectcom.apps.mindsphere.io/ |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for |

| Field | Description |
|---|---|
| | the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **MSU Limit** | MindSphere Unit (MSU) limit for an asset. MSU is the basis for fees invoiced monthly per asset and as per a used application. |
| **Security Profile** | The type of security profile/algorithm used for token generation. Default is SHARED_SECRET. |
| **Agent Id** | The ID of the agent managing a MindSphere asset. |
| **Schemas** | Schemas used by the MindSphere platform. Default is urn:siemens:mindsphere:v1. |
| **IAT** | Initial access token for agent onboarding. |
| **Tenant** | The tenant ID of the onboarded agent. |
| **Links** | Links returned upon successful agent onboarding. |
| **JWT Signing Algorithm** | The JSON Web Token (JWT) signing algorithm. Default is HS256 (HMAC with SHA 256). |
| **JWT Token Expiration (msec)** | The JWT token expiration time in milliseconds. Default is 600000. |
| **JWT Audience** | The JWT Audience. Default is MindSphere AS. |

| Field | Description |
|---|---|
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Simple Mail Transfer Protocol (SMTP)

Integration Cloud allows you to connect to an SMTP server using the Simple Mail Transfer Protocol (SMTP). This Application provides the "sendEmail" on page 273 predefined operation to send emails to specified recipients. You can attach one or more files to the message.

| Field | Description |
|---|---|
| **Host** | Host name or IP address or the domain name of the SMTP server. |
| **Port** | SMTP port defined on the SMTP server. This is the number of the port on which the SMTP host listens. |
| **User** | Valid user name on the SMTP server. This is the user name used to connect with the mail server. |
| **Password** | Password of the SMTP user. This is the password to connect with the mail server. |
| **From** | The email address of the sender, that is, the person that is going to send the messages. |
| **To** | E-mail address of the receiver. If you specify multiple addresses, separate them with commas without any spaces. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt |

| Field | Description |
|---|---|
| | to connect to the SMTP server. In case the network is slow or the SMTP server processing takes longer than usual, increase this value. |
| **Read Timeout** | The number of milliseconds in which Integration Cloud must read a response message from the SMTP server. |
| **Write Timeout** | Socket write timeout value in milliseconds. |
| **SSL configuration details** | |
| **Transport Layer Security** | Type of security protocol Integration Cloud uses when communicating with the SMTP server port. Set to:<br>■ **none** to use a non-secure mode when communicating with the port on the SMTP server. This is the default.<br>■ **explicit** to use explicit security when communicating with the port on the SMTP server. With explicit security, Integration Cloud establishes an un-encrypted connection to the email server, and then switches to the secure mode.<br>■ **implicit** to use implicit security when communicating with the port on the SMTP server. With implicit security, Integration Cloud always establishes an encrypted connection to the email server. |
| **Truststore Alias** | This is the alias for the truststore that contains the list of certificates that Integration Cloud uses to validate the trust relationship. Integration Cloud uses the default truststore if you do not specify a truststore alias. You can add a new truststore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete truststores. |

## Using the SMTP Application to send emails

## Summary

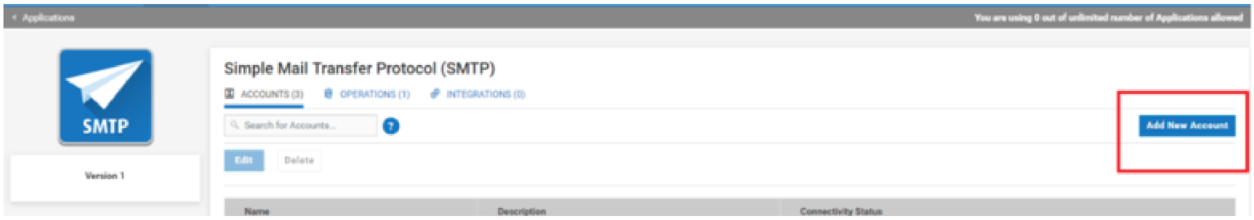
In this tutorial, we will create an integration that queries Salesforce accounts and create opportunities in Marketo, and then sends an email containing the total number of records queried from Salesforce and inserted into Marketo.

## Before you begin

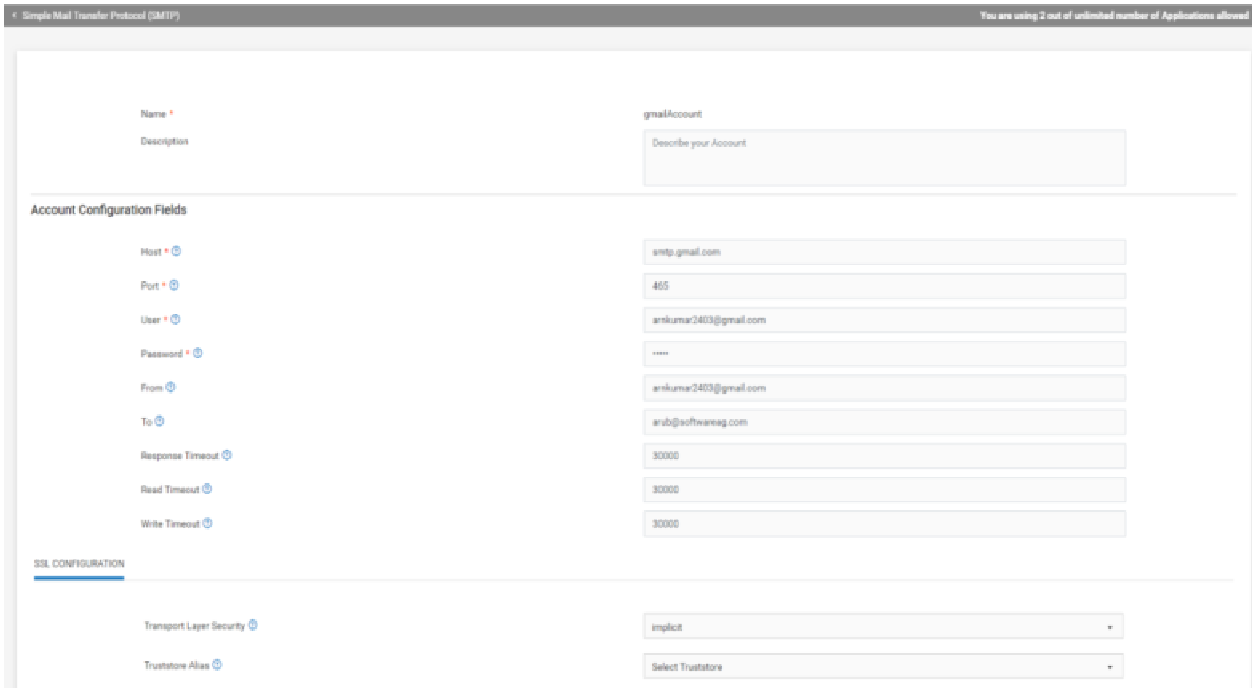
- A valid Gmail account as we will use the Gmail SMTP server to send emails.

## Using the SMTP Application to send emails

1. Log in to Integration Cloud.
2. Click **Projects > *Select a Project*> Applications > Predefined Applications > Simple Mail Transfer Protocol (SMTP)**.
3. Click **Add New Account**.



4. Configure the Gmail SMTP settings as shown below. For information on the Account Configuration fields, see " Simple Mail Transfer Protocol (SMTP)" on page 265. You can specify the **From** and **To** addresses either at the Account level here or while executing the operation. If you provide them at both places, Integration Cloud will use the addresses provided by you when you execute the operation. Further, as shown below, set the **Transport Layer Security** to **Implicit** to connect to SSL port 465. Integration Cloud internally sets the `mail.smtp.ssl.enable` property to `true`.


Simple Mail Transfer Protocol (SMTP)
You are using 2 out of unlimited number of Applications allowed

| | |
|---|---|
| Name * | gmailAccount |
| Description | Describe your Account |

Account Configuration Fields

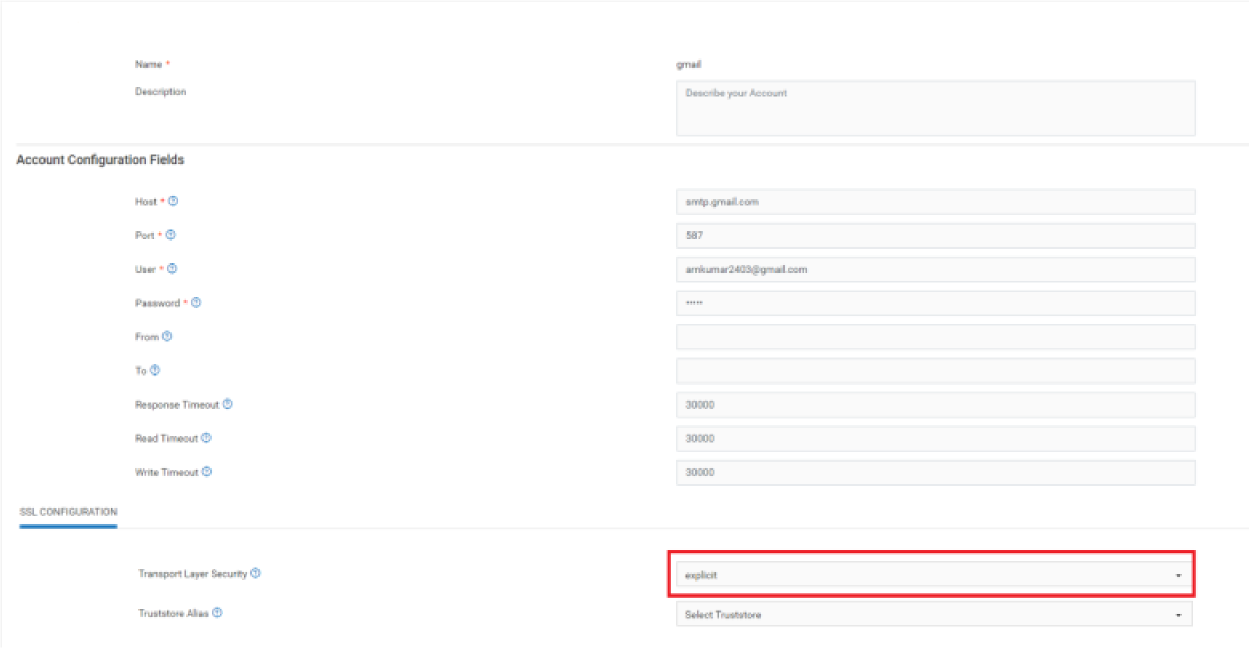
| | |
|---|---|
| Host * | smtp.gmail.com |
| Port * | 465 |
| User * | arnkumar2403@gmail.com |
| Password * | ***** |
| From | arnkumar2403@gmail.com |
| To | arub@softwareag.com |
| Response Timeout | 30000 |
| Read Timeout | 30000 |
| Write Timeout | 30000 |

SSL CONFIGURATION

| | |
|---|---|
| Transport Layer Security | implicit |
| Truststore Alias | Select Truststore |

If you want to connect to Gmail SMTP server TLS Port (587), specify the mandatory account configuration fields. Also, as shown below, specify the **Transport Layer Security** as **Explicit**.

If **Transport Layer Security** is set as **Explicit**, Integration Cloud adds the `mail.smtp.tls.required` property as `true` by default, while executing the operation. Also, Integration Cloud connects to the SMTP server and issues STARTTLS to change the unencrypted connection to an encrypted connection.



| | |
|---|---|
| Name * | gmail |
| Description | Describe your Account |

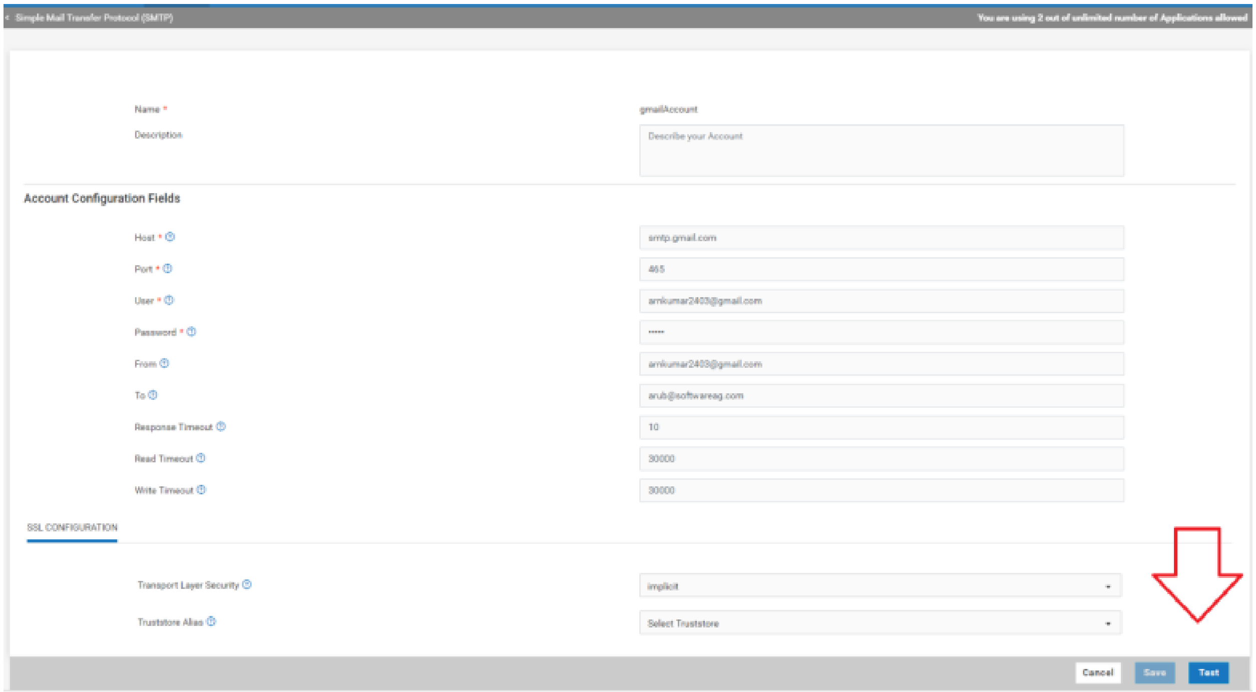
Account Configuration Fields

| | |
|---|---|
| Host * | smtp.gmail.com |
| Port * | 587 |
| User * | arnkumar2403@gmail.com |
| Password * | ***** |
| From | |
| To | |
| Response Timeout | 30000 |
| Read Timeout | 30000 |
| Write Timeout | 30000 |

SSL CONFIGURATION

| | |
|---|---|
| Transport Layer Security | explicit |
| Truststore Alias | Select Truststore |

5. After you have provided the above account configuration values, click **Test** on the **Account Configuration** page as shown below to test the SMTP server configuration.



< Simple Mail Transfer Protocol (SMTP)

You are using 2 out of unlimited number of Applications allowed

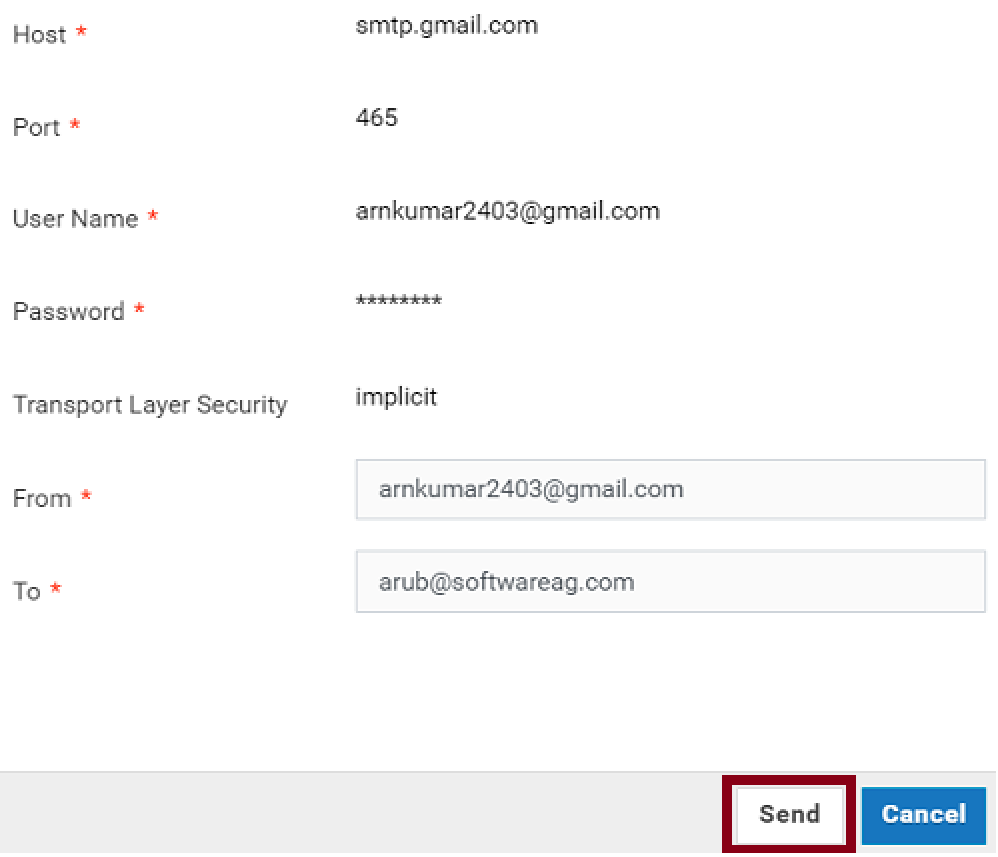
| | |
|---|---|
| Name * | gmailAccount |
| Description | Describe your Account |

Account Configuration Fields

| | |
|---|---|
| Host * | smtp.gmail.com |
| Port * | 465 |
| User * | arnkumar2403@gmail.com |
| Password * | ••••• |
| From | arnkumar2403@gmail.com |
| To | arub@softwareag.com |
| Response Timeout | 10 |
| Read Timeout | 30000 |
| Write Timeout | 30000 |

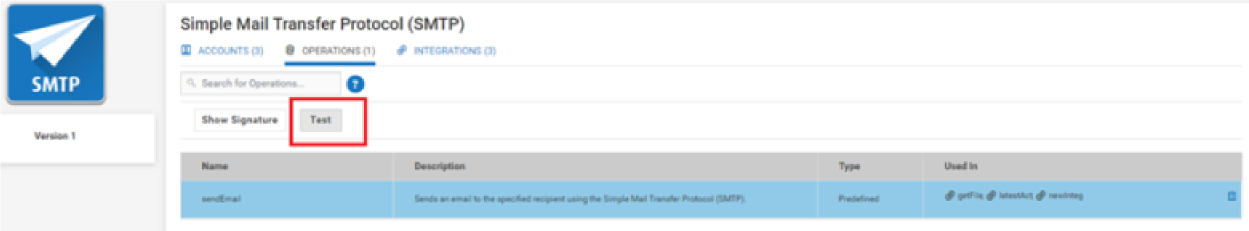
SSL CONFIGURATION

| | |
|---|---|
| Transport Layer Security | implicit |
| Truststore Alias | Select Truststore |



| | |
|---|---|
| Host * | smtp.gmail.com |
| Port * | 465 |
| User Name * | arnkumar2403@gmail.com |
| Password * | ******** |
| Transport Layer Security | implicit |
| From * | arnkumar2403@gmail.com |
| To * | arub@softwareag.com |

6. Click **Send** to send the test email to the recipient. You can also click **Test** on the **Operations** page to test the SMTP server configuration.

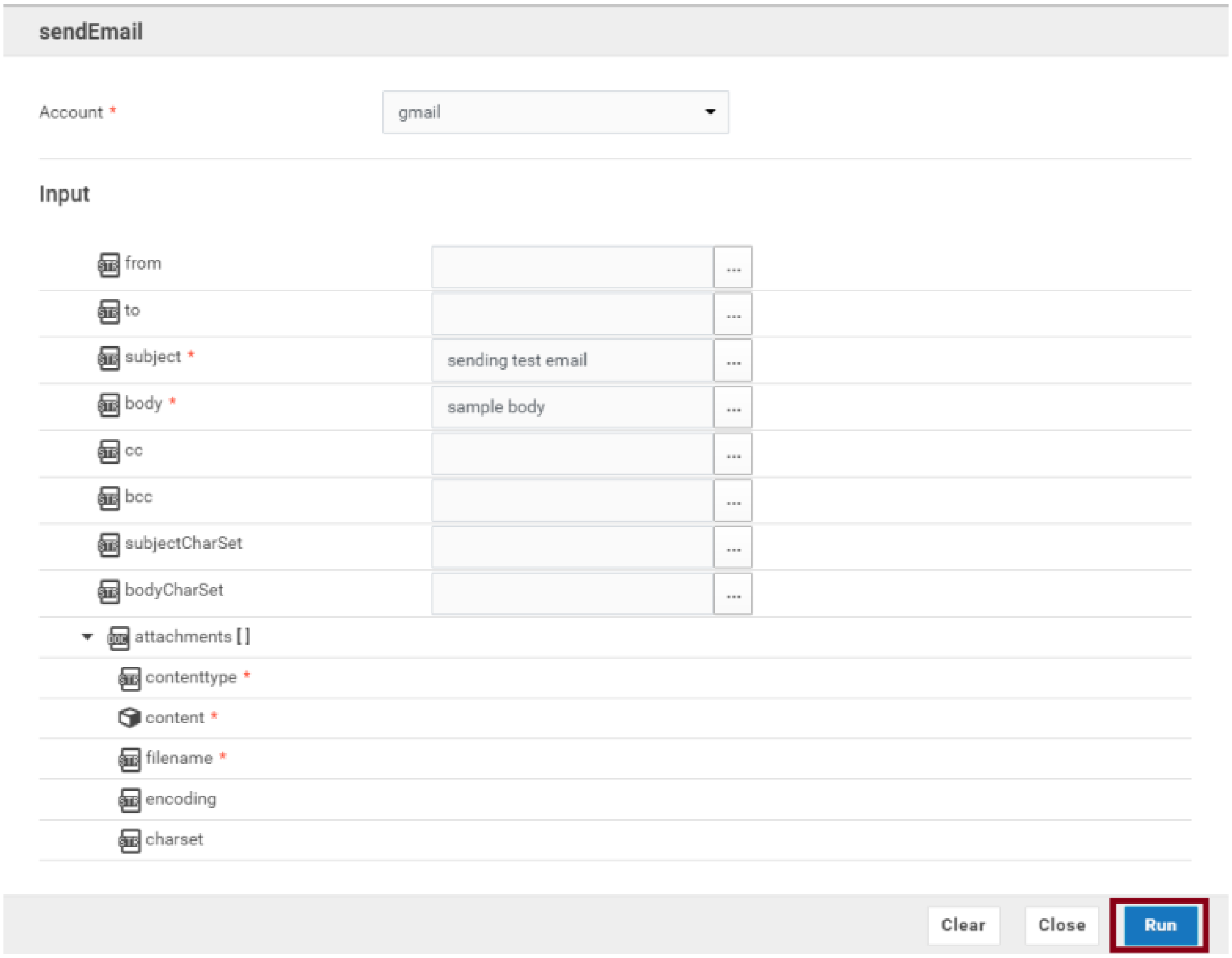


Simple Mail Transfer Protocol (SMTP)

| Name | Description | Type | Used In |
|---|---|---|---|
| sendEmail | Sends an email to the specified recipient using the Simple Mail Transfer Protocol (SMTP). | Predefined | getFile, latestAct, newInteg |

7. After you click **Test** on the **Operations** page, add the input values as shown below, and then click **Run**.

**Note:**
You can add attachments only while creating an Integration.



sendEmail

Account * gmail

Input

from

to

subject * sending test email

body * sample body

cc

bcc

subjectCharSet

bodyCharSet

attachments [ ]

contenttype *

content *

filename *

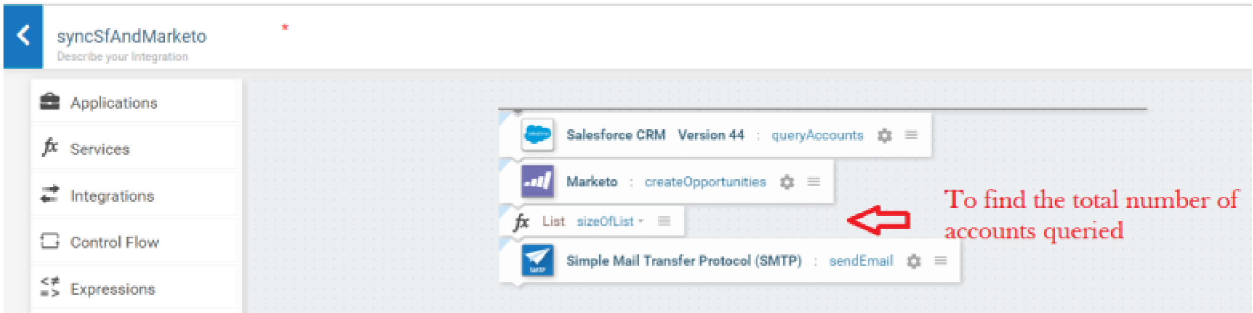
encoding

charset

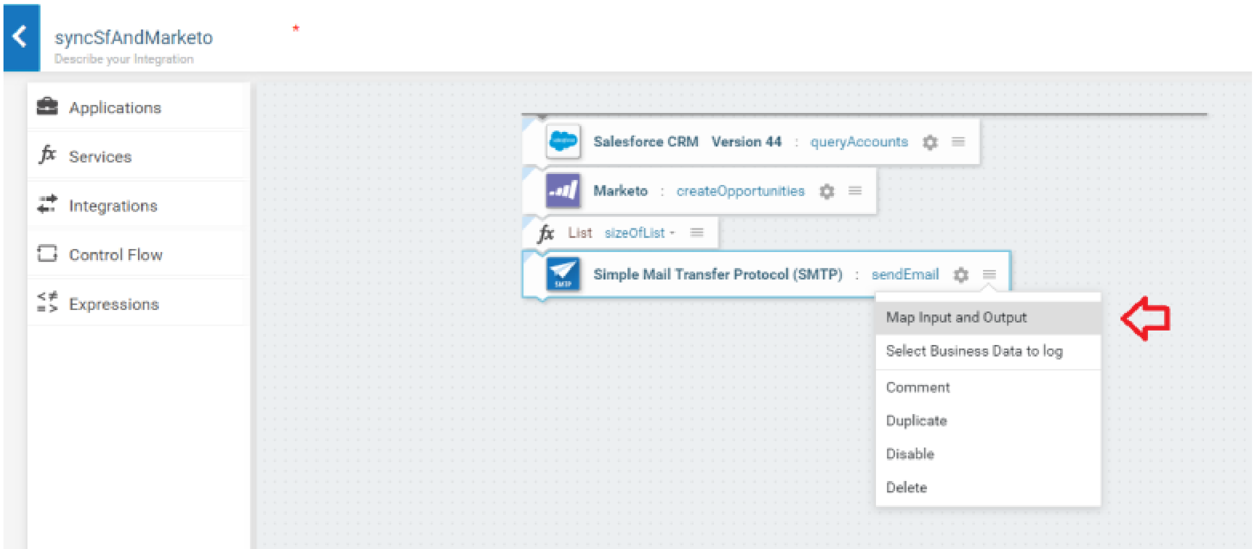
8. Use the SMTP Application to create the Orchestrated Integration as shown below. See "Orchestrated Integrations" on page 387 on how to create an Orchestrated Integration.

Do the following:

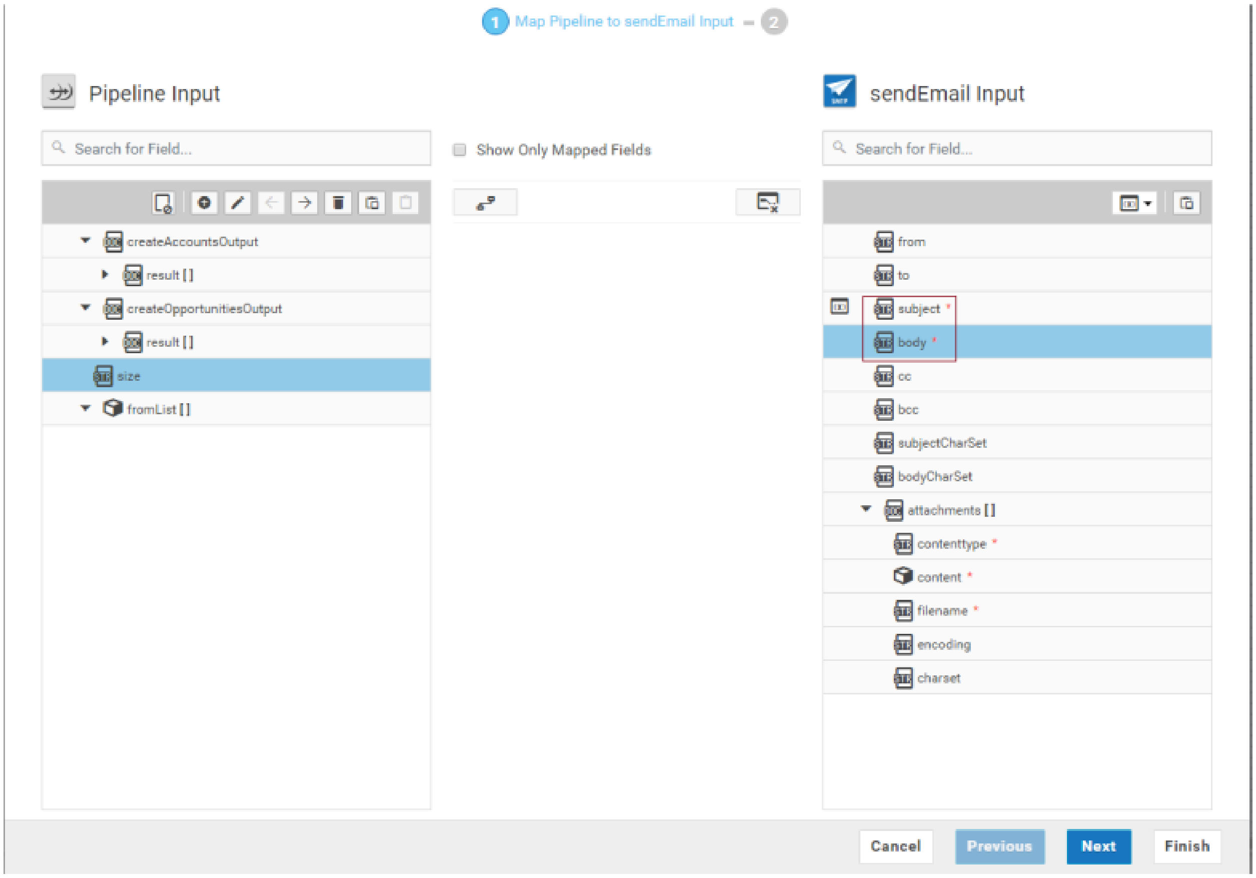
- Add the QueryAccounts operation from Salesforce.
- Add the createOpportunities operation from Marketo.
- Map the queried Salesforce accounts to Marketo createOpportunities.
- Add the sizeOfList service from the List category.
- Map the queried Salesforce accounts to the sizeOfList service to get the total number of accounts queried.
- Add the "sendEmail" on page 273 operation from the SMTP Application.


syncSfAndMarketo
Describe your Integration
Applications
Services
Integrations
Control Flow
Expressions
Salesforce CRM Version 44 : queryAccounts
Marketo : createOpportunities
List sizeOfList
Simple Mail Transfer Protocol (SMTP) : sendEmail
To find the total number of accounts queried

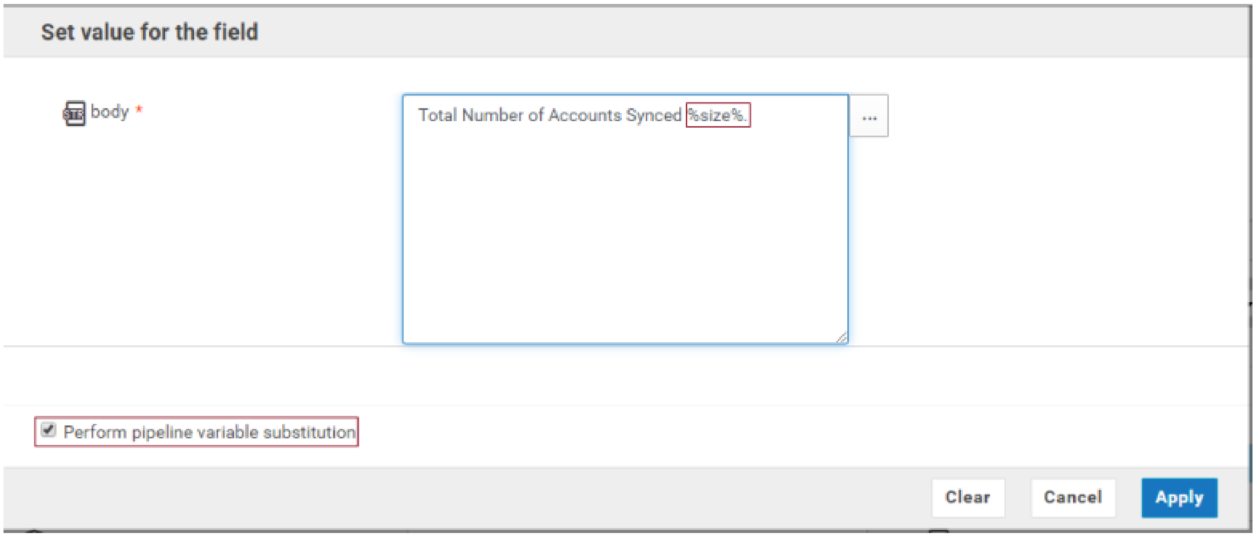
9. Map input and output.


syncSfAndMarketo
Describe your Integration
Applications
Services
Integrations
Control Flow
Expressions
Salesforce CRM Version 44 : queryAccounts
Marketo : createOpportunities
List sizeOfList
Simple Mail Transfer Protocol (SMTP) : sendEmail
Map Input and Output
Select Business Data to log
Comment
Duplicate
Disable
Delete

10. Click `subject` and set a value. Then click `body`.



11. Set a value for the `body`, that is, type the email body. Wrap the dynamic fields which are in the pipeline with % and click **Perform pipeline variable substitution**.



body *: Total Number of Accounts Synced %size%.

☑ Perform pipeline variable substitution

12. Click **Apply**, save the integration, and then run the Integration.

An email is sent to the recipients specified in the **To** field on the SMTP Account Configuration page.

## Exceptions

Timeout errors may occur while executing the “sendEmail” on page 273 operation. Do the following as remedial actions:

- Check if the host and port details are configured correctly.
- Increase the **Response Timeout** value on the **Account Configuration** page.

## SMTP Predefined Operation

### sendEmail

Sends emails to specified recipients. You can attach one or more files to the message.

### Input Parameters

*from* — **String** Optional. E-mail address of the sender.

*to* — **String** Optional. E-mail address of the receiver. If you specify multiple addresses, separate them with commas.

*subject* — **String** Subject of the message.

*body* — **String** The content of the message.

*cc*: — **String** Optional. E-mail addresses of additional receivers. If you specify multiple addresses, separate them with commas.

*bcc*: — **String** Optional. E-mail addresses of additional receivers. If you specify multiple addresses, separate them with commas.

*subjectCharSet*: — **String** Optional. The character set used to encode the subject. Default: UTF-8.

*bodyCharSet*: — **String** Optional. The character set used to encode the email message. Default: UTF-8.

*attachments* — **Document List** Attachments to the email message.

| Key | Description |
|---|---|
| *contenttype* | **String** Content type of the attachment. For example: application/pdf. |
| *content* | **byte[ ], String, or java.io.InputStream** Content of the attachment. |

| | |
|---|---|
| *filename* | **String** Name to assign to the attachment. |
| *encoding* | **String** Encoding of the attachment, for example, `base64` or `7bit`. If *encoding* is not specified, 7bit is used. |
| *charset* | **String** Character set encoding of the attachment. If *charset* is not specified, then UTF-8 encoding is used. |

### Output Parameters

| | |
|---|---|
| *status* | **String** Final status of the operation. |

### Usage Notes

- You must define the *from* and *to* fields either in the **Account Configuration** section or while executing the operation.
- If you are using *filename* to attach a file to the message and the file is not a plain text file, you must set the *contenttype* and *encoding*.

## Slack

Integration Cloud connects to Slack using the Slack REST API. You can use it to collaborate in your team within persistent chat rooms, private groups, and direct messaging, where all the content is searchable.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider.<br>The URL for Slack REST Application depends on the team name: https://YOURTEAM.slack.com<br>Example: https://exampleteam.slack.com |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Access Token** | The token used for authentication and issued by the Authorization Server. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## StrikeIron Contact Verification

Integration Cloud connects to StrikeIron using the StrikeIron Contact Verification APIs, and provides access to email verification and hygiene services, along with the North America address verification service.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://ws.strikeiron.com/StrikeIron. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider. |
| **Password** | Provide a password for the user name provided in the **Username** field to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |

| Field | Description |
|---|---|
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## SuccessFactors HCM

Integration Cloud connects to SuccessFactors using the SuccessFactors web service SFAPI, and performs SuccessFactors operations (Create, Read, Update, Delete, Fetch, Insert, Query, queryMore, and Upsert) over HTTP using synchronous SOAP protocols. This Application has been tested with the following business objects: GOAL$1, GOAL$2, GOAL$3, GoalMilestone$2, GoalMilestone$3, GoalTask$2, GoalTask$3, MatrixManager, and CustomManager.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://api.successfactors.com/sfapi/v1/soap<br>https://<instance_name>.successfactors.com, where <instance_name> represents the actual instance name. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |

| Field | Description |
|---|---|
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider. |
| **Password** | Provide a password for the user name to initiate communication with the SaaS provider. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the Account. The SuccessFactors Application does not use Authorization headers.<br>If you specify **none**, no additional authorization scheme will be executed at run time. If you specify a Company ID, Username, and Password, but do not specify a value for Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Company ID** | The company ID that SuccessFactors provided, when your company registered with them. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |

| Field | Description |
|---|---|
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## SugarCRM

Integration Cloud connects to SugarCRM using the Interface for RESTful Web Services v10 and manages the CRM data. You can use it to retrieve, query, create, update, and delete business objects of any type.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<instance>/rest/v10. Replace <instance> with your actual SugarCRM instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |

| Field | Description |
|---|---|
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. SugarCRM REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is |

| Field | Description |
|---|---|
| | other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Twilio

Using the REST interface, Twilio allows you to programmatically make and receive phone calls and send and receive text messages.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format:<br>https://api.twilio.com/2010-04-01 |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header.<br>If you enter the username and password, then set the authorization type as **basic** . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. Select the Authorization Type as **basic**. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | The name of the user account on the SaaS provider that the connection will use to connect to the SaaS provider. |
| **Password** | The password for the user name provided in the Username field. |

| Field | Description |
|---|---|
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## webMethods.io B2B

The webMethods.io B2B application allows you to:

- Interact, accept requests, and build integrations for the webMethods.io B2B product instance.
- Exchange business documents between trading partners.

### webMethods.io B2B account configuration

You must configure an account in the webMethods.io B2B application for you to utilize its predefined operations in Integration Cloud to build orchestrations.

**Note:**
webMethods.io B2B application on Integration Cloud requires you to have access to the webMethods.io B2B product instance.

| Field | Description |
|---|---|
| **B2B Environment** | Connects to the webMethods.io B2B product instance within your environment. |

**Note:**
You can only create one account in the webMethods.io B2B application. To create a new account, you must delete the existing account. If the existing account has integrations, then the integrations must be updated with the new account details.

## Predefined Operations

The following predefined operations are available for webMethods.io B2B application:

### changeUserStatus

Changes the user status of a specified transaction.

### Input Parameters

| | |
|---|---|
| *transactionID* | **String** Transaction ID of the document in exchange. |
| *userStatus* | **String** The user status for a transaction. For example, New, Open, or Fixed. Character length limit is 255. |

### Output Parameters

*isUpdated*

**String** Indicates if the operation changed the user status of a transaction.

- `true`. The operation changed the user status of the transaction.
- `false`. The operation has not changed the status of the transaction.

### parseContent

Parses request content passed by the processing rule **Call an integration** action as bytes or string data type. For more information on **Call an integration**, see *webMethods B2B Cloud Help*

## Input Parameters

*inputContent* — **String** Content passed in the integration request.

*loadContentAs* — **String** Data type in which *outputContent* field value is passed.

- **bytes**. *outputContent* is generated as bytes.
- **string**. *outputContent* is generated as a string.

*encoding* — **String** Type of character set used for encoding *inputContent* value. This can be any IANA registered character set.

**Default** `UTF-8`

## Output Parameters

*outputContent* — **Object** Content data corresponds to the data type option set for the *loadContentAs* field.

## generateResponse

Generates a response for the processing rule **Call an integration** action.

## Input Parameters

*inputContent* — **Object** Content data sent as part of integration response after encoding.

*readContentAs* — **String** Data type in which *inputContent* field value is passed.

- **bytes**. *inputContent* is read as bytes.
- **string**. *inputContent* is read as a string.

*contentType* — **String** Content-Type passed corresponds to *inputContent* field. For example, application/EDI, application/x12, text/plain and so on.

*errorCode* — **String** Value of error code passed as part of integration response.

*errorMessage* — **String** Value of error message passed as part of integration response.

*encoding* — **String** Type of character set used for encoding *inputContent* value. This can be any IANA registered character set.

**Default** `UTF-8`

## Output Parameters

*response* **Document** Response is a composite object containing the parameters:

| Key | Description |
|---|---|
| *content* | **String** Encoded in `Base64`. Value is specified in *encoding*. |
| *type* | **String** Value specified in *contentType*. |
| *encoding* | **String** Type of encoding used to encode the *inputContent*. Value is specified in *encoding*. |

*error* **Document** Error is a composite object containing the fields:

| Key | Description |
|---|---|
| *code* | **String** Error code received from the input. Value is specified in *errorCode*. |
| *message* | **String** Error message associated with the input. Value is specified in *errorMessage*. |

### submit

Submits the business documents to webMethods.io B2B product instance.

webMethods.io B2B recognizes the type of document and process it. Every document that you submit through this service is created as a new transaction in webMethods.io B2B.

## Input Parameters

*inputContent* **Object** Content to submit to webMethods.io B2B product instance for processing.

*readContentAs* **String** Data type in which *inputContent* field value is passed.

- **bytes**. *inputContent* is read as bytes.
- **string**. *inputContent* is read as string.

*contentType* **String** Specifies the type of content.

- **text/xml**. Content is read as an XML input.
- **application/EDIStream**. Content is read as an EDI InputStream.
- **application/EDI**. Content is read as EDI input.
- **application/x12**. Content is read as X12 input.

- **application/UNEDIFACT**. Content is read as UNEDIFACT input.

> **Note:**
> This field is case sensitive. Additionally, if you provide a type other than the one mentioned above, webMethods.io B2B identifies the document as *unknown*.

*encoding*

**String** Type of character set used for encoding *inputContent* value. This can be any IANA registered character set.

**Default:** `UTF-8`

*params*

**Document** A document that provides parameters to govern how webMethods.io B2B recognizes and processes a document.

For XML documents, you can optionally add the following fields:

- *DoctypeName* - **String** The name of the document type. This field is case sensitive.

  The *DoctypeName* field identifies the webMethods.io B2B document type to use, thus bypassing document recognition and eliminating the overhead of searching for the webMethods.io B2B document type.

- *processingRuleName* - The name of the processing rule. This field is case sensitive.

  The *processingRuleName* field identifies the processing rule to use, thus bypassing the processing rule lookup and eliminating the overhead of searching for a processing rule.

- *$bypassRouting* - Indicates whether webMethods.io B2B uses a processing rule to process the document. Valid values are:

  - `true`. Disables the processing rule routing.

  - `false`. Enables the processing rule routing

- *pipelineMatching* - Allows you to pass the key-value pair defined in an XML document. These variables are used to identify webMethods.io B2B document type for processing. For more information on pipeline matching, see the product help for webMethods.io B2B.

## Output Parameters

*transaction*

**Document** Contains the following information about the transaction created in webMethods.io B2B product instance.

| Key | Description |
|---|---|

| | |
|---|---|
| *transactionID* | **String** Transaction ID of the document created in webMethods.io B2B product instance. |
| *senderID* | **String** Sender ID of the document. |
| *receiverID* | **String** Receiver ID of the document. |
| *documentTypeID* | **String** Internal ID of the document type in the webMethods.io B2B product instance. |
| *userStatus* | **String** User-defined status of the document. |
| *transactionStatus* | **String** Status of the transaction submitted in the webMethods.io B2B product instance. |
| *customAttributes* | **Document** Extracted custom attributes from *inputContent*. Key is the attribute name and value is the attribute value. |

*params* **Document** A document that contains parameters that webMethods.io B2B uses to recognize and process a document.

## getAddresses

Retrieves all addresses of a partner.

### Input Parameters

*partnerID* **String** The internal identifier of the partner for which you want to retrieve the addresses.

### Output Parameters

*addresses* **Document list** List of addresses, each containing the following details:

| Key | Description |
|---|---|
| *addressLine1* | **String** The first line of the address. |
| *addressLine2* | **String** The second line of the address. |
| *addressLine3* | **String** The third line of the address. |
| *city* | **String** The city specified for the address. |
| *country* | **String** The country for the address. |
| *zipCode* | **String** The ZIP code or postal code for the address. |
| *stateProvince* | **String** The state or province for the address. |

| | | |
|---|---|---|
| | *addressType* | **String** Type of address. For example, corporate or contact address of the partner. |
| | *partnerID* | **String** The internal identifier of the partner. |
| *extendedFields* | **Document list** The extended fields for the address. Each extended field is in the following structure: | |

> **Note:**
> If there are no extended fields for the group, then this field appears blank.

| | |
|---|---|
| *name* | **String** The name of the extended field. |
| *value* | **String** The value of the extended field. |

## getContacts

Retrieves all contacts of a partner.

### Input Parameters

*partnerID* — **String** The internal identifier of the partner for which you want to retrieve the contacts.

### Output Parameters

*contacts* — **Document list** List of contact details each containing the following information:

| Key | Description |
|---|---|
| *firstName* | **String** The first name of the contact. |
| *lastName* | **String** The last name of the contact. |
| *role* | **String** The role of the contact in the organization. |
| *type* | **String** The type of contact. For example, technical contact or administrative contact. |
| *email* | **String** The e-mail address of the contact. |
| *faxNumber* | **String** The facsimile number of the contact. |
| *telephoneExtension* | **String** The telephone extension of the contact |
| *telephone* | **String** The telephone number of the contact. |
| *partnerID* | **String** The internal identifier of the partner. |

*address* **Document list** The address of the contact. For the various address fields, see "getAddresses " on page 287.

*extendedFields* **Document list** The extended fields for *contacts* . Each extended field is in the following structure:

> **Note:**
> If there are no extended fields for the group, then this field appears blank.

*name* **String** The name of the extended field.

*value* **String** The value of the extended field.

### getCorporation

Retrieves the corporation information of a partner.

#### Input Parameters

*partnerID* **String** The internal identifier of the partner for which you want to retrieve the corporate information.

#### Output Parameters

*corporate* **Document list** Contains the following corporate information of a partner:

| Key | Description |
|---|---|
| *corporationName* | **String** The name of the corporation. |
| *orgUnitName* | **String** The name of the organizational unit within the corporation. |
| *status* | **String** The status of the partner in your webMethods.io B2B. This value can be *Active* or *Inactive*. |
| *isEnterprise* | **Boolean** Indicates if the partner profile represents the enterprise profile. Valid values are:<br>■ `true`. It is an enterprise profile.<br>■ `false`. It is not an enterprise profile. |
| *partnerID* | **String** The internal identifier of the partner. |

*extendedFields* **Document list** The extended fields for the corporation. Each extended field has the following structure:

> **Note:**

If there are no extended fields for the group, then this field appears blank.

| | |
|---|---|
| *name* | **String** The name of extended field. |
| *value* | **String** The value of the extended field. |

## getPartnerIdentities

Retrieves all external IDs of a partner.

### Input Parameters

| | |
|---|---|
| *partnerID* | **String** The internal identifier of the partner for which you want to retrieve the identities. |

### Output Parameters

*identities* **Document list** Contains the following details of the identities of the partner.

| Key | Description |
|---|---|
| *type* | **String** The type of identity. For example, `Duns`, `Duns+4`. |
| *value* | **String** The value of identity. |
| *code* | **String** The code for the type of the identity. For example, 1, 2 , 3. |
| *partnerID* | **String** The internal identifier of the partner. |

*extendedFields* **Document list** The extended fields for the *identities* . Each extended field has the following structure:

**Note:**
If there are no extended fields for the group, then this field appears blank.

| | |
|---|---|
| *name* | **String** Name of extended field to use in the criteria. |
| *value* | **String** The extended fields to match. |

## getExtendedFields

Retrieves a set of extended fields for a partner.

Extended fields are custom fields that you can define to maintain additional information about you partners. For example, you might want to define extended fields for preferred shipping method, cost centers, or customer codes.

### Input Parameters

*partnerID* **String** The internal identifier of the partner for which you want to retrieve the extended fields.

*groupName* **String** (Optional) The group for which you want to retrieve the extended fields. Specify the name of the field group associated with the extended fields to retrieve. You can specify one of the following standard field groups:

- `Corporation`. Retrieves Corporation extended fields.
- `Contact`. Retrieves Contact extended fields.
- `Delivery`. Retrieves Delivery extended fields.
- `ExternalID`. Retrieves IDs extended fields.
- `Address`. Retrieves Addresses extended fields.
- `Custom`. Retrieves Custom extended fields.

> **Note:**
> Apart from the above standard field groups, you can also specify the groups created under the **Field Groups** section in webMethods.io B2B.

### Output Parameters

*groups* **Document list** The extended field group details of the extended fields. Each extended field group is in the following format:

*name* **String** The group name for the extended field.

*id* **String** The id of the group.

*fields* **Document list** The list of extended fields within the group. Following are the fields:

- `name`. **String** The name of the extended field.
- `value`. **String** The value of the extended field.

> **Note:**
> If the group you selected has no extended fields, then this field appears blank.

### queryPartners

Creates a query for partner profiles and returns the first batch of matching results.

**Note:**
A user can execute 15 queries concurrently.

## Input Parameters

*batchSize*

**String** (Optional) The maximum number of profiles present in each batch. You can provide a value between 20 and 500. The default is 100.

> **Note:**
> If you provide a value outside the specified range, then webMethods.io B2B displays an error message.

*criteria*

**Document** (Optional) The criteria to filter the query results. Provide the following details:

- *isOrQuery*. **String** Whether the join condition for the filter criteria that you specify in between the *fields* variable is *true* (OR) or *false* (AND). Default is *false*.
- `fields`. **Document list**

  The criteria to filter the results for fields such as corporationName, orgUnitName. Each field can have multiple criteria. For example, `orgUnitName IS NOT NULL AND orgUnitName = ABC*`, where `orgUnitName` is the field name, `IS NOT NULL` and '=' are the operators, `AND` is the join condition (isOrQuery), and `ABC*` is the value. With this criteria, webMethods.io B2B will select all the profiles for which the organization unit name is not null and the name starts with *ABC*.

  - *fieldName* - **String** The name of the field for which you specify the filter criteria in the *criteria* variable. Valid values are: *corporationName* , *orgUnitName* , *status*.
  - *isOrQuery* - **String** Whether the join condition for the filter criteria in the *criteria* variable is AND or OR. Valid values are *true* (OR) or *false* (AND). The default value is *false*.
  - *criteria* - **String** The filter criteria for *fieldName*.

    Each document in the document list contains the following variables:

    - *operator* - **String** The operator for the filter criteria. Valid values are: *IS NULL*, *IS NOT NULL*, =, and <> (not equal to).
    - *value* - **String** The value for the filter criteria. You can use the wildcard * in searches to match one or more characters, or the wildcard ? to match one character. For example, to search for data ending with *abc*, specify the value as **abc*. Similarly, to search for data starting with *ab* and ending with *c* with length 4, specify the value as *ab?c*.

*sortOrder*

**Document list** (Optional) The field name by which the results must be sorted. Each document in the document list contains the following variables:

- *fieldName* - **String** The field name for sorting.
  - `corporationName`
  - `orgUnitName`
  - `status`
- *isAscending* - **Document** The order by which the result set must be sorted (ascending or descending). The valid values are `ASC` (ascending) or `DESC` (descending).

*resultFields*

**String list** (Optional) The list of fields in the result set. Valid values are:

- `corporationName`
- `orgUnitName`
- `status`

If the field is left blank, then webMethods.io B2B returns all the above fields including the `isEnterprise` and `partnerID` fields. For details on these values, see "getCorporation " on page 289

By default, the query results contain `isEnterprise` and `partnerID` fields.

## Output Parameters

*queryLocator*

**String** The ID of the executed query for which results are retrieved. It is used in `queryMorePartners` operation to retrieve subsequent sets of records from the query results when the value of *done* is *false*. Similarly, it is also used in `resetQueryPartners` operation to release the queryLocator.

*done*

**Boolean** Indicates whether the query has additional rows to retrieve. This is used in the `queryMorePartners` operation. Valid values are:

- `true`. There are no additional rows to retrieve.
- `false`. There are additional rows to retrieve.

*size*

**String** The total number of rows retrieved in the query result. The value is equal to the total number of *records* that webMethods.io B2B returns.

*records*

**List** The list of partner profiles.

Based on the values set in the *resultField*, you can view the following information:

- `corporationName`. **String** Name of the corporation.
- `orgUnitName`. **String** The name of the organizational unit within the corporation.
- `status`. **String** The status of the partner in your webMethods.io B2B. This value can be `Active` or `Inactive`.
- `isEnterprise`. **Boolean** Indicates if the partner profile represents the Enterprise profile. The values are `true` or `false`, indicating an enterprise or non-enterprise respectively.
- `partnerID`. **String** The internal identifier of the partner.

## queryMorePartners

Loops over the data cached from the `queryPartners` operation and returns the *batchSize* of data.

### Input Parameters

*queryLocator*

**String** The ID of the query. You can use this ID to view the query results. You obtain this ID from the `queryPartners` operation.

### Output Parameters

*done*

**Boolean** Indicates whether the query has additional rows to retrieve. Valid values are:

- `true` . There are no additional rows to retrieve.
- `false`. There are additional rows to retrieve.

*size*

**String** The total number of records retrieved in the query result.

*records*

**List** The list of partner profiles.

Based on the values set in the *resultField*, you can view the following information:

- `corporationName`. **String** Name of the corporation.
- `orgUnitName`. **String** The name of the organizational unit within the corporation.
- `status`. **String** The status of the partner in your webMethods.io B2B. This value can be `Active` or `Inactive`.
- `isEnterprise`. **Object** Indicates if the partner profile represents the Enterprise profile.

- `partnerID`. **String** The internal identifier of the partner.

## resetQueryPartners

Resets the *queryLocator* making it available for the next concurrent execution.

**Note:**
The *queryLocator* reset involves clearing the cached results. If you do not reset a *queryLocator*, it remains active for 15 minutes. After which, webMethods.io B2B automatically resets it.

### Input Parameters

*queryLocator* **String** The ID of the query obtained from the `queryPartners` operation. You obtain this ID from the `queryPartners` operation.

### Output Parameters

*status* **Boolean** Indicates whether reset is complete or not.

## log

Adds an entry into the activity log.

### Input Parameters

*type* **String** (Optional) The type of activity log entry. The supported types are: *ERROR*, *WARNING*, and *MESSAGE*.

*category* **String** (Optional) The category for the activity log entry. webMethods.io B2B supports either the following categories or any custom value:

- `General`. Any unspecified error messages and warnings.
- `Processing`. Processing a document using a processing rule to be used .

*message* **String** A brief message for the activity log entry. The value can be a string with maximum of 240 characters.

**Note:**
webMethods.io B2B displays an error message when you exceed the character limit.

*details*

**String** (Optional) A detailed message about the reason for adding the activity log entry. The value can be a string with maximum of 1024 characters.

> **Note:**
> If the webMethods.io B2B displays an error message when you exceed the character limit.

*transactionID*

**String** The transaction ID of the document created in webMethods.io B2B product instance related to this activity log entry.

*partnerID*

**String** (Optional) The internal ID of the partner related to this activity log entry.

*content*

**Object** (Optional) The payload content for the activity log. The maximum size of the content is 2 MB.

> **Note:**
> If the content size exceeds 2 MB, webMethods.io B2B displays an error message.

*readContentAs*

**String** (Optional) Data type in which *content* field value is passed.

- **bytes**. *content* is read as bytes.
- **string**. *content* is read as a string.

*encoding*

**String** (Optional) The character set in which the value of the content is encoded. Specify an IANA-registered character set. For example, *UTF-8* and *ISO-8859-1*. Default is *UTF-8*.

### Output Parameters

*success*

**Boolean** Indicates whether the activity log is added for the provided transaction ID. Valid values are:

- `true`. Indicates that the activity log entry is added successfully.
- `false`. Indicates that the addition of activity log entry has failed.

## Workday

Integration Cloud connects to Workday using the SOAP API and allows you to interact with Workday web services to manage Workday objects.

| Field | Description |
|---|---|
| **Server URL** | This is the login endpoint to initiate communication with the SaaS provider. |

| Field | Description |
|---|---|
| | Specify the URL for your exact instance, for example: https://<instance>.workday.com/ccx/service/<tenantID>/.<br>For example, wd2-impl-services1 is the instance and softwareag_pt1 is the tenantID. |
| **Username** | User name for the Workday tenant to log in to Workday. Enter the user name appended with the Workday Tenant ID, for example, username@tenantID. |
| **Password** | The password of the Workday account. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you select **none**, no additional authorization scheme will be executed at run time.<br>If you specify a **Username** and **Password** and also select **none**, you do not specify a value for the **Authorization Type**, so the user credentials are not inserted into an Authorization header. If you enter the **Username** and **Password**, then set the authorization type as **basic**.<br>Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. If you enter the username and password, then set the authorization type as **basic**. |
| **Instance** | The Workday instance, for example, wd2-impl-services1. |
| **Tenant ID** | The Workday tenant ID that you want to access, for example, softwareag_pt1. |
| **Version** | SOAP API version of Workday. The version value is obtained from the soapbind:address location in the WSDL file. For example, if the soapbind:address location is https://wd2-impl-services1.workday.com/ccx/service/softwareag_pt1/Staffing/v31.1, then the version is v31.1. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |

| Field | Description |
|---|---|
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keep Alive Interval** | The keep alive interval in milliseconds defines the interval for which a connection will be kept alive, if the back end does not respond with a Keep-Alive header. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Idle Timeout** | The idle timeout interval in milliseconds defines the interval for which a connection will be kept alive if it's not in use. A value > 0 keeps the connection alive for the specified value. The default value of -1 implies that the connection will be kept alive until a request fails due to a connection error. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client’s identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is |

| Field | Description |
|---|---|
| | other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Zendesk

Integration Cloud connects to Zendesk using the Zendesk API v2. It includes ticketing system, self-service options, and customer support features, and allows you to create, update, and solve customer support tickets and also track problems and questions.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<domain>.zendesk.com. Replace <domain> with your actual Zendesk instance. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. Zendesk REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

## Zuora

### Zuora

Integration Cloud connects to Zuora using the Zuora SOAP API. You can use it to connect to Zuora and automate billing, commerce, and financial operations.

### Zuora REST

Integration Cloud connects to Zuora REST using the Zuora REST API. Zuora allows you to manage Zuora objects in the Zuora Business Object Model, process revenue schedules, and perform other financial operations.

| Field | Description |
|---|---|
| **Server URL** | Provide the login endpoint to initiate communication with the SaaS provider. Example: https://api.zuora.com/apps/services/a/76.0. For Zuora REST: https://rest.<your_tenant_name>.zuora.com. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Username** | This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider. When you use a SOAP based Zuora Application, it is not recommended to use more than one Account with the same user name. |
| **Password** | Provide a password for the user name provided in the **Username** field to initiate communication with the SaaS provider. When you access Zuora from outside your company's trusted network, you must add a security token (provided by Zuora) to your password. For more information about logging on Zuora, see the Zuora documentation. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the Account. The Zuora Application does not use Authorization headers. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is |

| Field | Description |
|---|---|
| | org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. |
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |
| **Consumer ID** | Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. |
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. The Access Token is passed when you invoke any of the REST API endpoints. |
| **Refresh Token** | A token used by the client to obtain a new access token without involving the resource owner. |
| **Refresh URL** | This is the provider specific URL to refresh an Access Token. Example: https://rest.apisandbox.zuora.com/oauth/token. |

## REST Applications

REST (Representational State Transfer) is an architectural style that requires web applications to support the HTTP GET, POST, PUT, and DELETE methods and to use a consistent, application-independent interface.

### Endpoint URL

The endpoint of an API is an unique URL, which represents an object or collection of objects. The endpoint is a reference to a URI that accepts web requests. It is the login endpoint URL to initiate communication with the SaaS provider. To get the endpoint, go through the SaaS provider documentation available on the internet. For example, https://api.twitter.com/1.1/ is the Twitter endpoint.

### Authentication Type

Every back end provides its own Authentication mechanism to provide authorized access to its APIs. You need to get the authentication details from the SaaS provider documentation. For example, for Twitter, go to https://apps.twitter.com, create a new application, and then get the credentials. For Twitter, the authentication is OAuth V1.0a, which you can get from https://apps.twitter.com.

### Resource

A resource refers to some object or set of objects that are exposed at an API end point. It is a representation of a thing (a noun) on which the REST APIs (verbs) operate. A resource has a type, one or more parameters, and some standard operations that allow you to manipulate or retrieve it from a remote location if you know its endpoint URL. Each resource derives its path from the namespace of the resource. For example, if the REST resource is named myREST.myRESTResource, the path is "/myREST.myRESTResource".

### Action

Actions are tasks that act on a Resource. You must create at least one Action for a Resource after you have created the Resource. You can add a Method, Request Parameters, Request and Response Headers, and a Request and Response body to an Action.

### HTTP Method

The primary or most-commonly-used HTTP verbs (or methods, as they are properly called) are POST, GET, PUT, and DELETE. These correspond to create, read/retrieve, update, and delete (or CRUD) operations, respectively. You use the following HTTP methods to map the CRUD operations to HTTP requests. In a REST request, the resource that you are working with is specified in the URL – Uniform Resource Locator. The URL is a special case of the URI – Uniform Resource Identifier.

- **GET** - Used to read or retrieve a representation of a resource. For example, GET <endpointurl>/addresses/2 will retrieve an address with an ID of 2.
- **POST** - Creates a resource. For example, POST <endpointurl>/addresses will create a new address.

- **PUT** - Updates an existing resource. For example, PUT <endpointurl>/addresses/3 will modify the address with an ID of 3.
- **DELETE** - Used to delete a resource identified by a URI. For example, DELETE <endpointurl>/addresses/4 will delete an address with an ID of 4.

| Resource | GET | PUT | POST | DELETE |
|---|---|---|---|---|
| http://example.com/<br>api/resource/ | Lists details and perhaps URIs of the resources in this collection. | Replaces the entire collection. | Creates a new item in the collection. | Deletes a collection. |
| http://example.com/<br>api/resource/123/ | Retrieves a specific item in the collection. | Updates the item in the collection and possibly creates an item if it does not exist. | Creates a new item in the collection. | Deletes an item from the collection. |

### Headers and Parameters

REST is not a standard in itself but instead makes use of the HTTP standard. HTTP headers allow the client and the server to pass additional information with the request or the response. For example, the *Accept* and *Content-Type* HTTP headers can be used to describe the content being sent or requested within an HTTP request. The client may set Accept to application/json if it is requesting a response in JSON or application/xml if it is requesting a response in XML, that is, when sending data, setting the Content-Type to application/xml tells the client that the data being sent in the request is XML.

REST calls (requests) and responses are sent over the HTTP protocol, hence REST requests are in the form of URLs that point to the resource(s) on the server. Required parameters are attached to the end of the URL. For example, in the resource URL http://<name>.com/user/789, user is the resource and 789 is the parameter that is passed to the URL of the resource. You can use any REST client to make REST calls.

REST parameters specify the variable parts of your resources, that is, the data that you are working with. QUERY parameters are the most common type of parameters, which is appended to the path of the URI when submitting a request. For example, `https://api.twitter.com/1.1/users/show.json?screen_name=twitterdev` is an example of a QUERY parameter URI where screen_name is the name of the parameter and twitterdev is the value of the parameter.

### HTTP Status Codes

HTTP Status Codes indicate the status of the HTTP response:

- 1XX - Informational
- 2XX - Success
- 3XX - Redirection
- 4XX - Client error

- 5XX - Server error

## Creating and updating REST Applications

These screens allow you to define a REST Application, define Resources and Actions, and then create a REST Application. See "REST Applications" on page 303 for conceptual information on REST Resources, HTTP Methods, HTTP Status Codes, HTTP Headers, and Parameters.

### To create a REST Application

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > REST Applications > Add New Application**.

2. In the **Define Application Details** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Name** | Type a name for the REST Application. |
| **Description** | Type an optional description for the REST Application. The description you enter here will appear in the **Applications** page. |
| **Default Endpoint URL** | Specify the **Endpoint** for the Application. It is the login endpoint URL to initiate communication with the SaaS provider. To get the end point, see the back end documentation available on the internet for the SaaS provider. |
| **Authentication Type** | Every back end provides its own authentication mechanism. Get the authentication details from the back end documentation and select the supported **Authentication Type** from the drop-down list. |
| **Application Icon** | Click **Browse** and select another icon for the REST Application, if necessary. |

3. Click **Next**.

   The **Define Resources and Actions** page appears.

4. In the **Define Resources and Actions** page, click **Add Resource** to create a new REST Resource.

   The **Add Resource** dialog box appears. In the **Add Resource** dialog box, complete the following fields:

| Field | Description |
|---|---|
| **Name** | Type the Resource name. |
| **Path** | Type the path to the Resource. The Resource path is relative to the endpoint specified. Each REST Resource derives its path from the namespace of the REST Resource. For example, if the REST Resource is named myREST.myRESTResource, the path is “/myREST.myRESTResource”.<br>You can define dynamic parameters in the resource path by enclosing each parameter within { } brackets. For example, to get the employee data corresponding to a dynamic parameter called employeeID, specify the resource path as /employee/{employeeID}. To get item information from a particular department in a store, specify the resource path as /store/{departmentID}/{itemID}.<br>**Note:**<br>While adding an **Action**, if your Resource path contains { } brackets, for example, /user/{userID}, you must add a request parameter having the same name, that is, "userID", and set the **Parameter Type** to **URI_CONTEXT**. |

5. Click **Add** to create the Resource. You can **Edit** or **Delete** the Resource from the **Define Resources and Actions** page.

6. In the **Define Resources and Actions** page, select the Resource and click **Add Action**.

   **Note:**
   Every Resource must have an Action associated with it.

   In the **Add Action to Resource** dialog box, complete the following fields:

| Field | Description |
|---|---|
| **Method** | Select an HTTP Method.<br>■ **GET** - Reads or retrieves a representation of a resource. For example, GET <endpointurl>/addresses/2 will retrieve an address with an ID of 2.<br>■ **PUT** - Updates an existing resource. For example, PUT <endpointurl>/addresses/3 will modify the address with an ID of 3. |

| Field | Description |
|---|---|
| | ■ **POST** - Creates a resource. For example, POST <endpointurl>/addresses will create a new address.<br>■ **DELETE** - Deletes a resource identified by an URI. For example, DELETE <endpointurl>/addresses/4 will delete an address with an ID of 4. |
| **Description** | Type an optional description for the Action. |
| **Request Parameter** | You can set parameters that become part of the outgoing request. Parameters specify the variable parts of your resources. Click **Add Parameter** to add a parameter to the request. Complete the following fields:<br>**Name** - Type the parameter name.<br>**Value** - Type a value for the parameter.<br>**Parameter Type** - Select the parameter's type, which determines how the parameter should be used.<br>f you select an AWS authentication type, then you must add a *mandatory request parameter* in all the Actions you create. The parameter name must be *aws.service* and the parameter type must be *CFG_PARAM*. Type the service name in the endpoint URL as the parameter value. For example, if the endpoint URL is https://<instance>.s3.com/, type the parameter value as s3.<br>REST services rely on HTTP methods (GET, POST, PUT, and DELETE) to make requests to a SaaS provider. Thus the parameters are closely tied to these HTTP methods, as they are sent as part of these HTTP method requests. The parameters are part of the HTTP URI. *CFG_PARAM* is an internal configuration parameter.<br>*URI_CONTEXT* parameters are passed as the path component of a REST Resource URI, and the parameter names correspond to the URI path variable names specified in the {} annotation. For example, if the URI is https://api.twitter.com/1.1/users/{id}, the Resource path will be /users/{id}, the parameter type will be uriContext, the parameter name will be id, and the value could be the user id, for example, either 1, or 2, or 3. |

| Field | Description |
|---|---|
|  | *QUERYSTRING_PARAM* parameters are passed as the query component of a REST resource invocation request. For example, if the URI is https://api.twitter.com/<br>1.1/users/show.json?screen_name=twitterdev, the resource path will be /users/show.json, screen_name is the name of the parameter, twitterdev is the value of the parameter, and the parameter type is query.<br>*FORM_ENCODED_PARAM* - Define a parameter of type FORM_ENCODED_PARAM if you want to send the parameter as part of the Request body. FORM_ENCODED_PARAM allows you to send simple key value parameters embedded in the Request body for POST or PUT requests. This uses the default web form encoding, which is application/x-www-form-urlencoded.<br>**Note:**<br>For passing parameters of FORM_ENCODED_PARAM type, you will not be able to define the Request body for a Resource, as the generated parameter key value string will be automatically embedded in the Request body.<br>**Required** - Select this option if you want this parameter to be made mandatory while creating an Integration. |
| **Request Header** | HTTP headers allow the client and the server to pass additional information with the request or the response.<br>**Note:**<br>Do not add an authorization header if you use **credentials** as the mode of authentication.<br>Click **Add Header** to add a request HTTP header. In the **Add Header** dialog box, complete the following fields:<br>**Name** - Type the Header name.<br>**Value** - Type a value for the Header.<br>**Required** - Select this option if you want this Header to be made mandatory while creating an Integration. |

| Field | Description |
|---|---|
| **Request Body** | In the Request Body pane, complete the following fields:<br>**Content Type** If the documentation of the SaaS provider specifies that the content type of the request body is JSON, select **application/json** as the content type. If the documentation of the SaaS provider specifies that the content type of the request body is XML, select **application/xml** as the content type. If the documentation of the SaaS provider specifies that the content type of the request body is binary, select **Binary** as the content type. These options allow you to control the content in an HTTP request body.<br>**Document Type** - Select a Document Type for the request body or click **Create Document Type** to create a new Document Type. See "Creating Document Types from Scratch" on page 637.<br>**Note:**<br>Document Types created for a REST Application do not appear in the **Develop > Document Types** screen but appears only in the **Document Types** panel for the selected REST Application.<br>If the request payload is an array of object, then first create a document type resembling the content of the object, and then select the **Array** option. |
| **Response Header** | In the Response Header pane, click **Add Header** to add a Response HTTP header.<br>**Note:**<br>Do not add an authorization header if you use **credentials** as the mode of authentication.<br>Complete the following fields:<br>**Name** - Type the Header name.<br>**Value** - Type a value for the Header.<br>**Required** - Select this option if you want this Header to be made mandatory while creating an Integration. |
| **Response Body** | In the Response Body pane, complete the following fields: |

| Field | Description |
|---|---|
| | **HTTP Code** - Type a single HTTP status code or a code range to indicate the status of the response. Valid values are 100, 101, 102...599 or a range from 100-599.<br>**Content Type** If the documentation of the SaaS provider specifies that the content type of the response body is JSON, select **application/json** as the content type. If the documentation of the SaaS provider specifies that the content type of the response body is XML, select **application/xml** as the content type. If the documentation of the SaaS provider specifies that the content type of the response body is binary, select **binary** as the content type. These options allow you to control the content in an HTTP response body.<br>**Document Type** - Select a **Document Type** for the Response Body or click **Create Document Type** to create a new Document Type. See "Creating Document Types from Scratch" on page 637.<br>**Note:**<br>Document Types created for a REST Application do not appear in the **Develop > Document Types** screen but appears only in the **Document Types** panel for the selected REST Application. |

7. Click **Save**.

   The Action appears in the **Define Resources and Actions** page. You can **Edit** or **Delete** the Action from the **Define Resources and Actions** page.

   **Note:**
   Do not edit or delete an Action if it is already used in an Operation. If the Action is edited or deleted, the Operations that are dependent on the Action including the Integrations that are dependent on the affected Operations, will not function properly.

8. Click **Next**.

   The **Confirm REST Application** page appears.

9. Click **Finish** to create the REST Application.

   The new REST Application appears in the **REST Applications** page.

10. To edit the REST Application, click the REST Application link and then click **Edit Application**. You can change the **Description** and **Application Icon**. After you click **Finish**, the **Update**

**REST Application** window appears, which provides a summary of the impacted Accounts, Operations, and Integrations. Click **Update** to update the REST Application. To delete the REST Application, click **Delete Application**.

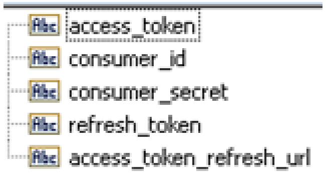
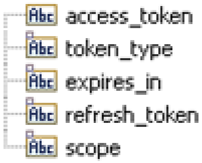
## REST Applications - Account Configuration Details

Integration Cloud allows you to create custom REST Applications. REST (Representational State Transfer) is an architectural style that requires web applications to support the HTTP GET, POST, PUT, and DELETE methods and to use a consistent, application-independent interface.

| Field | Description |
|---|---|
| **Server URL** | This is the login **Endpoint URL** you have specified in the **Define Application Details** page while creating the REST Application. |
| **Authorization Type** | The type of HTTP authorization scheme to use for the connection. If you enter the username and password, then set the authorization type as **basic**. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. If you select **none**, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the **Response Timeout** value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response. |
| **Retry Count on Response Failure** | The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the **Retry on Response Failure** option. |
| **Retry on Response Failure** | Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection. |
| **Consumer ID** | Also referred to as the Client ID, in OAuth 2.0, this is a client identifier issued to the client to identify itself to the authorization server. For Auth 1.0a, it is the Consumer Key issued by the Service Provider and used by the consumer to identify itself to the Service Provider. |

| Field | Description |
|---|---|
| **Access Token** | This token is used for authentication and is issued by the Authorization Server. For OAuth 1.0a, it is a value used by the Consumer to gain access to the Protected Resources on behalf of the User, instead of using the User's Service Provider credentials. |
| **Access Token Secret** | A secret used by the Consumer to establish ownership of a given Access Token. For OAuth 1.0a, it is the secret used by the Consumer to establish ownership of a given Access Token. |
| **Refresh URL** | The provider specific URL to refresh an Access Token. |
| **Session Timeout (min)** | The maximum number of minutes a session can remain active, in other words, how long you want the server to wait before terminating a session. The value should be equal to the session timeout value specified at the SaaS provider back end. |
| **Username** | The username credentials for the current Account configuration. |
| **Password** | The password credentials for the current Account configuration. |
| **Trust store Alias** | Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. This option is available only if **Credentials** is selected as the **Authentication Type** while creating the Application. |
| **Hostname verifier** | Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification. This option is available only if **Credentials** is selected as the **Authentication Type** while creating the Application. |
| **Enable SNI** | Server Name Indication (SNI) is an extension to the TLS protocol by which a client indicates which host name it is attempting to connect to at the start of the handshaking process. Enable this option if the SaaS provider offers SNI-based TLS connectivity, and if you want to connect to an SNI enabled SAAS provider to send the host name specified in the **Server URL** field, as part of the TLS SNI Extension `server_name` parameter. |
| **SNI Server Name** | If you want to explicitly specify a host name to be included as a part of the SNI extension `server_name` parameter, in case the host name is other than the host name specified in the **Server URL** field, specify the host name value in the **SNI Server Name** field. |

| Field | Description |
|---|---|
| **Keystore Alias** | Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. This option is available only if **Credentials** is selected as the **Authentication Type** while creating the Application. |
| **Client Key Alias** | Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. This option is available only if **Credentials** is selected as the **Authentication Type** while creating the Application. |
| **Consumer Secret** | Also referred to as the Client Secret, this is a secret matching to the client identifier. For Auth 1.0a, it is the secret used by the Consumer to establish ownership of the Consumer Key. |
| **Refresh Token** | Issued with the OAuth v2.0 access token only. A token used by the client to obtain a new access token without having to involve the resource owner. |
| **Refresh URL Request** | Options for sending the parameters in the Access Token refresh request. The options are **Body Query String**, **URL Query String**, and **Custom Integration**.<br>**Body Query String** - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the body of the POST request.<br>**URL Query String** - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the URL of the POST request.<br>**Custom ESB Service** - Use this option if the back end requires refresh requests in a custom format. If you select this option, you must specify the name of your Integration in the **Refresh Custom ESB Service** field. You can also create an Integration by clicking the link. |
| **Refresh Custom ESB Service** | To refresh the access tokens for accounts which use the OAuth 2.0 protocol, you can specify a call-back Integration which will execute when the access token expires.<br>This is a user implemented service for refreshing the *OAuth 2.0* Access Token. This option allows you to create an Integration, which will be executed when the Access Token has expired or is not valid. |

| Field | Description |
|---|---|
| | To create this Integration, create a custom REST Application with an Operation that invokes the back end to fetch the refresh token. You can use this custom REST Application along with its Operation to create this Integration.<br>Click the *Integration link* to create a new Integration.<br>The Integration must have a specification whose input parameters are:<br><br>access_token<br>consumer_id<br>consumer_secret<br>refresh_token<br>access_token_refresh_url<br>and the output parameters are:<br><br>access_token<br>token_type<br>expires_in<br>refresh_token<br>scope<br>**Note:**<br>The integration will be pre-populated with the input/output adhering to the above mentioned specification.<br>The newly created Integration will generate an access token that is mapped to the access token field in the output signature.<br>**Note:**<br>While editing the Account, this new Integration appears in the **Refresh Custom ESB Service** field in the Account Configuration page. |

# Tweet the status on Twitter using the REST Application

## Summary

In this tutorial, we will see how to create a REST Application using the Twitter REST API and use that REST Application to post a tweet on Twitter.

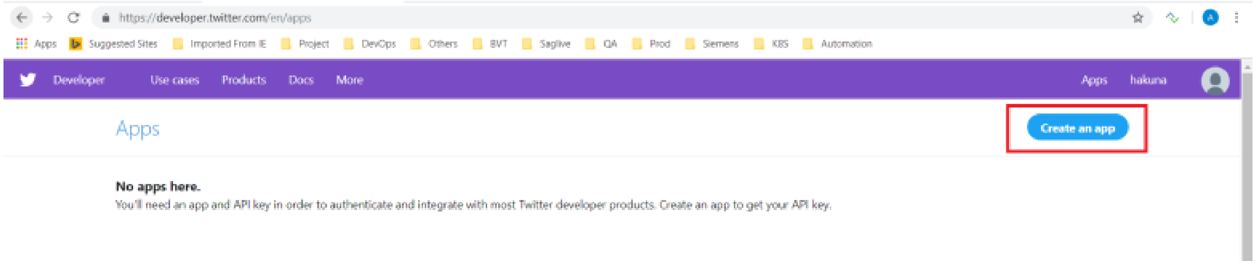
## Before you begin

- Permissions to create Operations and Integrations for a project in Integration Cloud.

- An email account ID, for example, twitapplication@gmail.com.
- A Twitter account. You can create one using your email ID.

## Steps

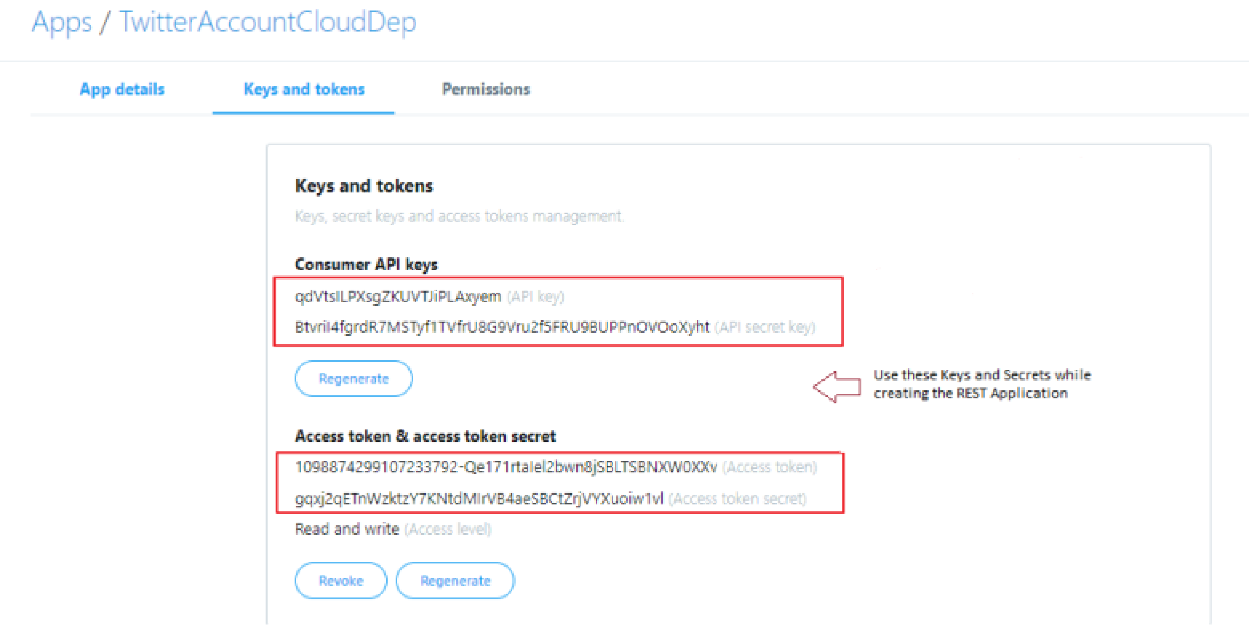
1. Log in to Twitter, go to https://developer.twitter.com/en/apps, and click **Create an app** to create a Twitter app.



Apps

**No apps here.**
You'll need an app and API key in order to authenticate and integrate with most Twitter developer products. Create an app to get your API key.

2. Create an application by filling the form and click **Create**.



Apps / TwitterAccountCloudDep

**Keys and tokens**

Keys, secret keys and access tokens management.

**Consumer API keys**

qdVtsILPXsgZKUVTJiPLAxyem (API key)

Btvril4fgrdR7MSTyf1TVfrU8G9Vru2f5FRU9BUPPnOVOoXyht (API secret key)

Use these Keys and Secrets while creating the REST Application

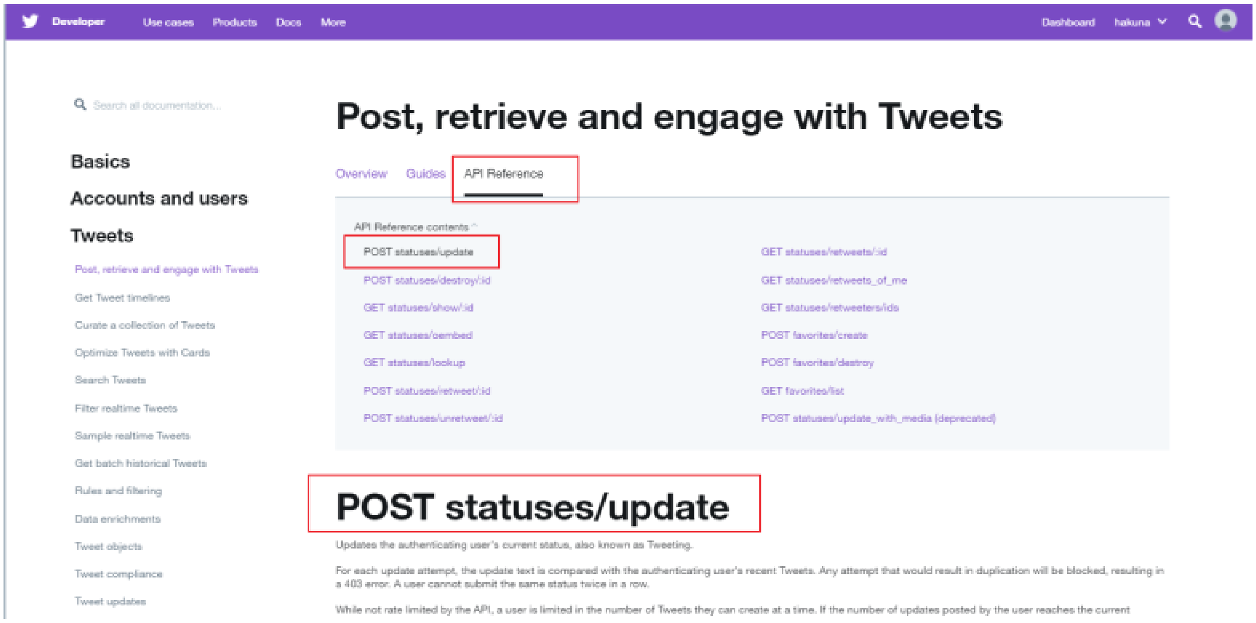
**Access token & access token secret**

1098874299107233792-Qe171rtalel2bwn8jSBLTSBNXW0XXv (Access token)

gqxj2qETnWzktzY7KNtdMIrVB4aeSBCtZrjVYXuoiw1vl (Access token secret)

Read and write (Access level)

**Note:**
See https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/post-statuses-update for information on POST statuses/update.



Post, retrieve and engage with Tweets

API Reference contents

| | |
|---|---|
| POST statuses/update | GET statuses/retweets/:id |
| POST statuses/destroy/:id | GET statuses/retweets_of_me |
| GET statuses/show/:id | GET statuses/retweeters/ids |
| GET statuses/oembed | POST favorites/create |
| GET statuses/lookup | POST favorites/destroy |
| POST statuses/retweet/:id | GET favorites/list |
| POST statuses/unretweet/:id | POST statuses/update_with_media (deprecated) |

POST statuses/update

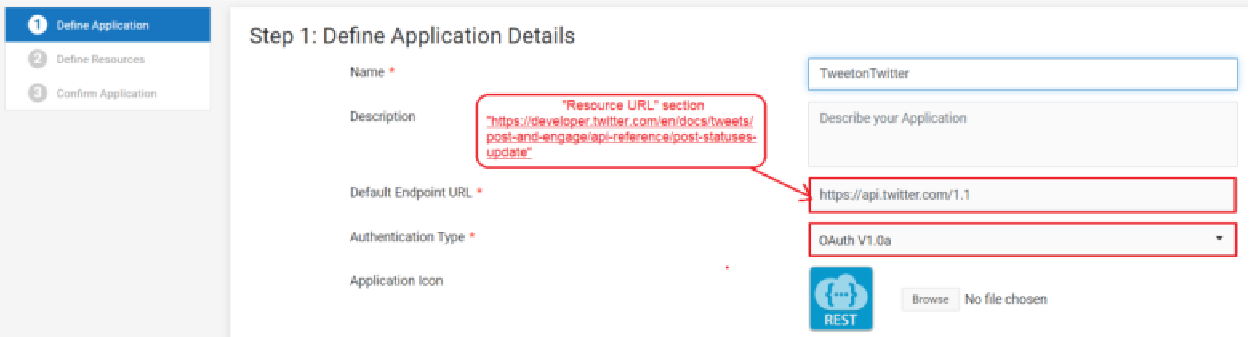
Updates the authenticating user's current status, also known as Tweeting.

For each update attempt, the update text is compared with the authenticating user's recent Tweets. Any attempt that would result in duplication will be blocked, resulting in a 403 error. A user cannot submit the same status twice in a row.

While not rate limited by the API, a user is limited in the number of Tweets they can create at a time. If the number of updates posted by the user reaches the current

3. Log in to Integration Cloud and go to **Projects > <Select a Project> > Applications > REST Applications > Add New Application**. Fill in the details as shown below in the **Define Application Details** screen and then click **Next**.

> **Note:**
> Default Endpoint URL should not contain the path of the endpoint. For example, if the complete endpoint URL is https://api.twitter.com/1.1/statuses/update.json, then https://api.twitter.com/1.1 is the Default Endpoint URL. Rest of the path `/statuses/update.json` will be used as the Resource Path while adding a Resource later.



Step 1: Define Application Details

Name * TweetonTwitter

Description Describe your Application

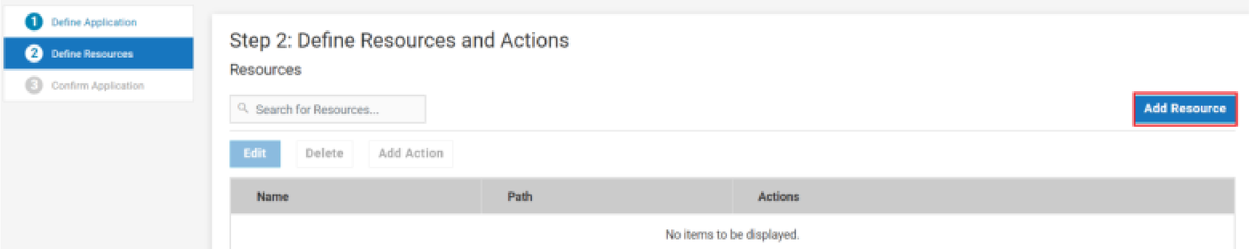
"Resource URL" section "https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/post-statuses-update"

Default Endpoint URL * https://api.twitter.com/1.1

Authentication Type * OAuth V1.0a

Application Icon No file chosen

4. On the **Define Resources and Actions** page, click **Add Resource**.

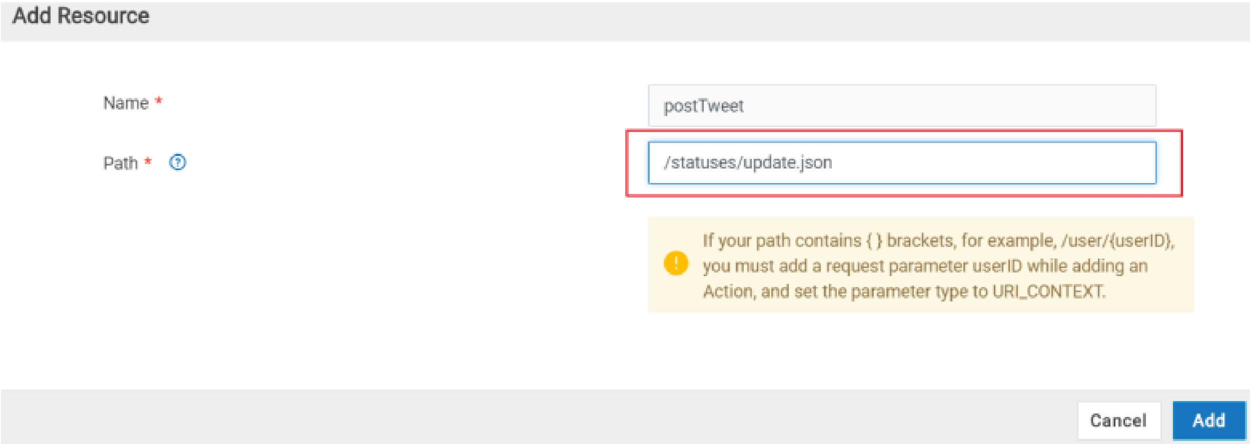


Step 2: Define Resources and Actions

Resources

| Name | Path | Actions |
|---|---|---|
| No items to be displayed. | | |

5. Let us add the *postTweet* Resource. Click **Add**.



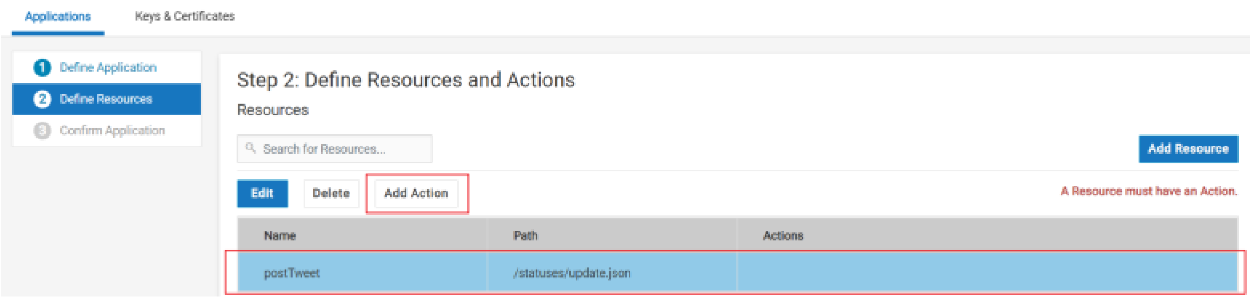
Add Resource

Name * postTweet

Path * /statuses/update.json

If your path contains { } brackets, for example, /user/{userID}, you must add a request parameter userID while adding an Action, and set the parameter type to URI_CONTEXT.

6. Select the postTweet Resource and click **Add Action**.



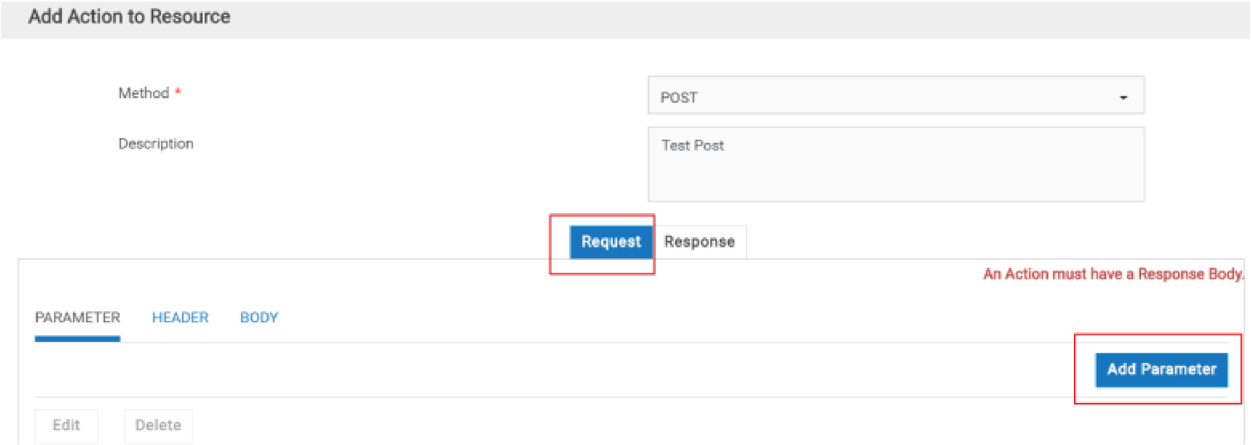
Step 2: Define Resources and Actions

Resources

A Resource must have an Action.

| Name | Path | Actions |
|---|---|---|
| postTweet | /statuses/update.json | |

7. On the **Add Action to Resource** page, select **POST** as the **Method**. In our example, we are considering a POST request which needs a Request PARAMETER and a Response BODY. Click **Add Parameter** to add the Request PARAMETER.



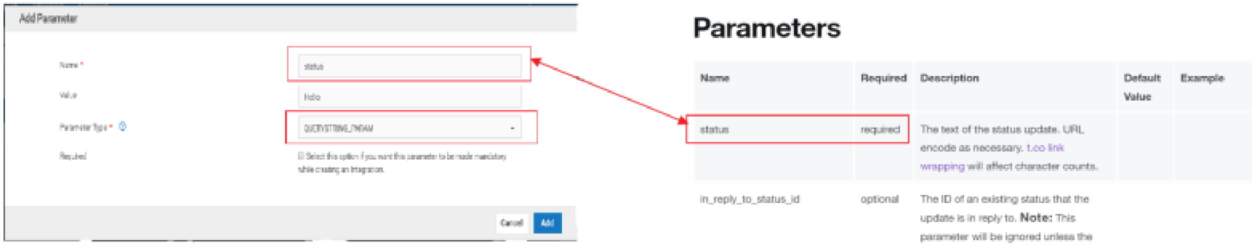
Add Action to Resource

Method * POST

Description Test Post

An Action must have a Response Body.

8. Specify the PARAMETER name exactly as it is in the API documentation (https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/post-statuses-update).



Add Parameter

Name * status

Value Hello

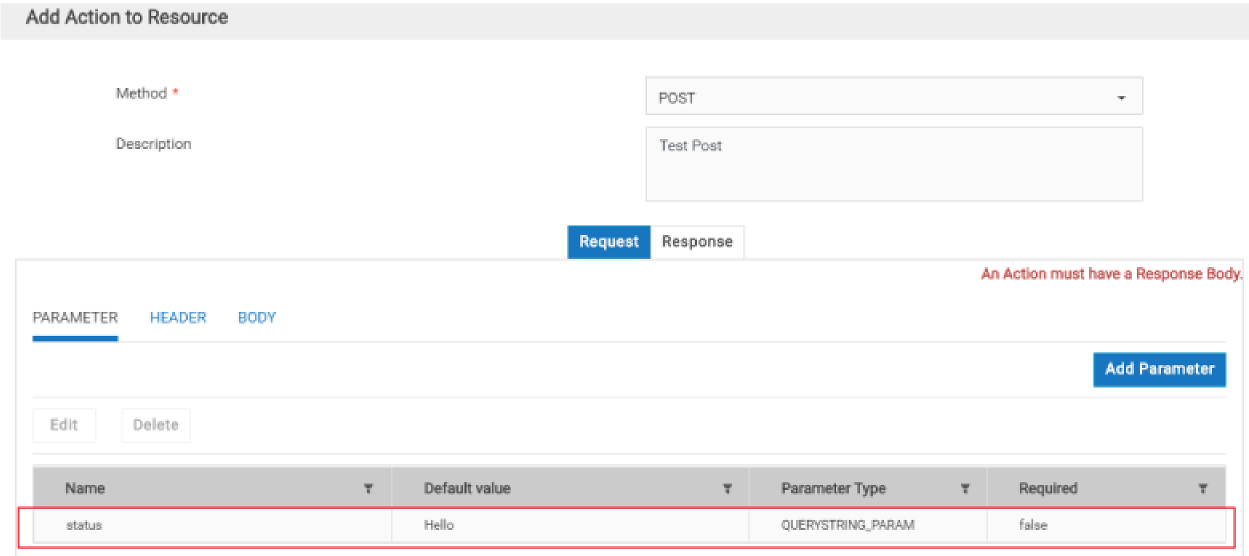
Parameter Type * QUERYSTRING_PARAM

Required ☐ Select this option if you want this parameter to be made mandatory while creating an integration.

Cancel Add

**Parameters**

| Name | Required | Description | Default Value | Example |
|---|---|---|---|---|
| status | required | The text of the status update. URL encode as necessary. t.co link wrapping will affect character counts. | | |
| in_reply_to_status_id | optional | The ID of an existing status that the update is in reply to. **Note:** This parameter will be ignored unless the | | |



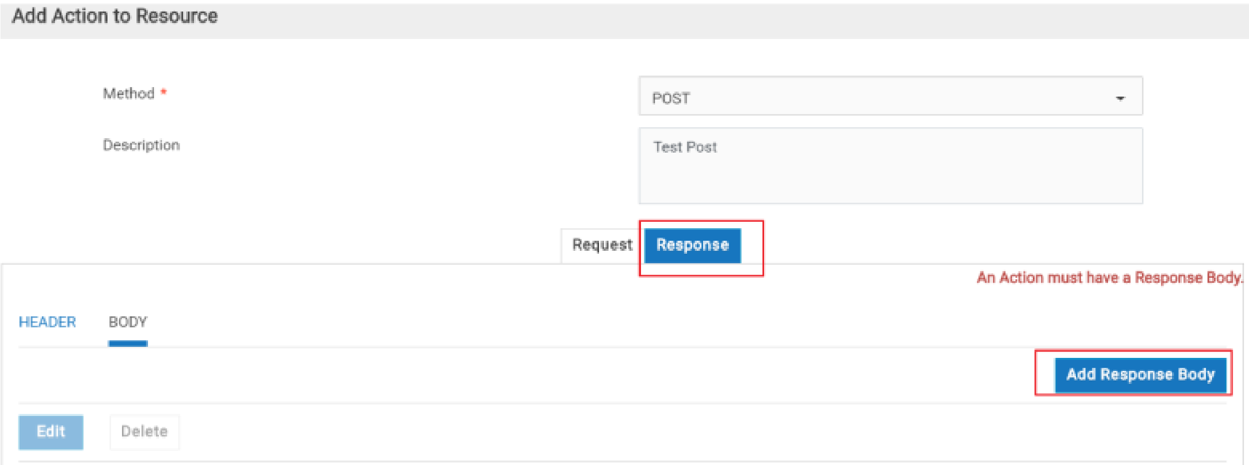
Add Action to Resource

Method * POST

Description Test Post

| Name | Default value | Parameter Type | Required |
|---|---|---|---|
| status | Hello | QUERYSTRING_PARAM | false |

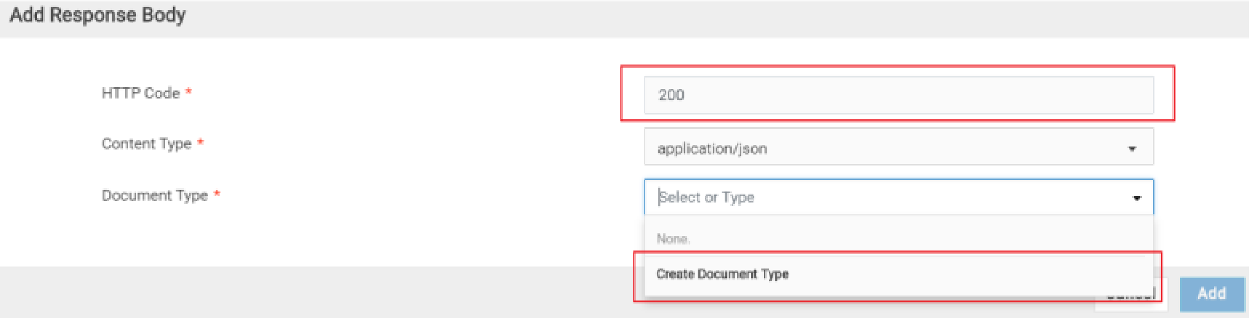
9. Now let us add the Response Body. Click **Response > BODY > Add Response Body**.



Add Action to Resource

Method * POST

Description Test Post



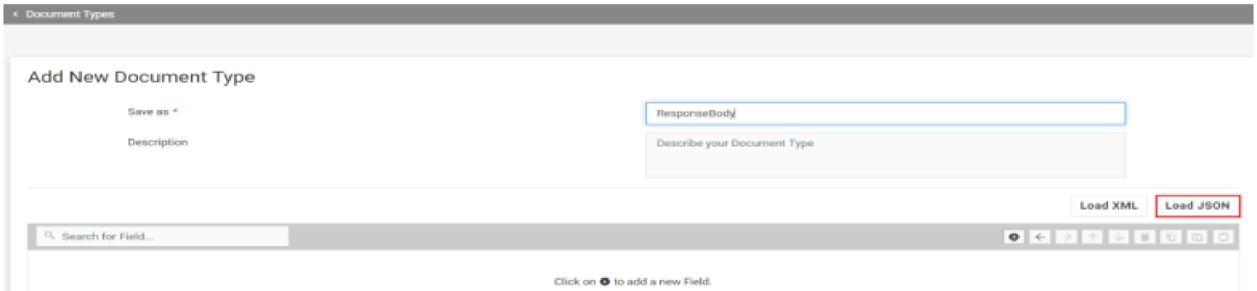
Add Response Body

HTTP Code * 200

Content Type * application/json

Document Type * Select or Type

10. In the **Document Type** drop-down list, click **Create Document Type** and then click **Load JSON** in the **Add New Document Type** window.



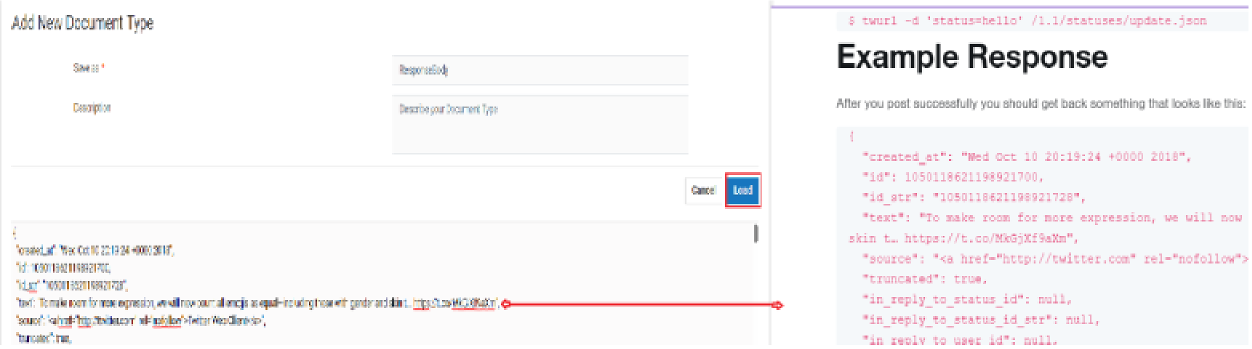
Add New Document Type

Save as * ResponseBody

Description: Describe your Document Type

Click on ● to add a new Field.

11. Copy the *Example Response* from the Twitter API Documentation page (https://developer.twitter.com/en/docs/tweets/post-and-engage/api-reference/post-statuses-update), paste it in the text area of the page as shown below, and click **Load**. Note that there is a known issue in the response section of the Twitter API Documentation. As a workaround, while pasting, omit the record starting with the string *source*. After loading, click **Save** and then **Add** to add the Response BODY.



Add New Document Type

Save as * ResponseBody

Description: Describe your Document Type

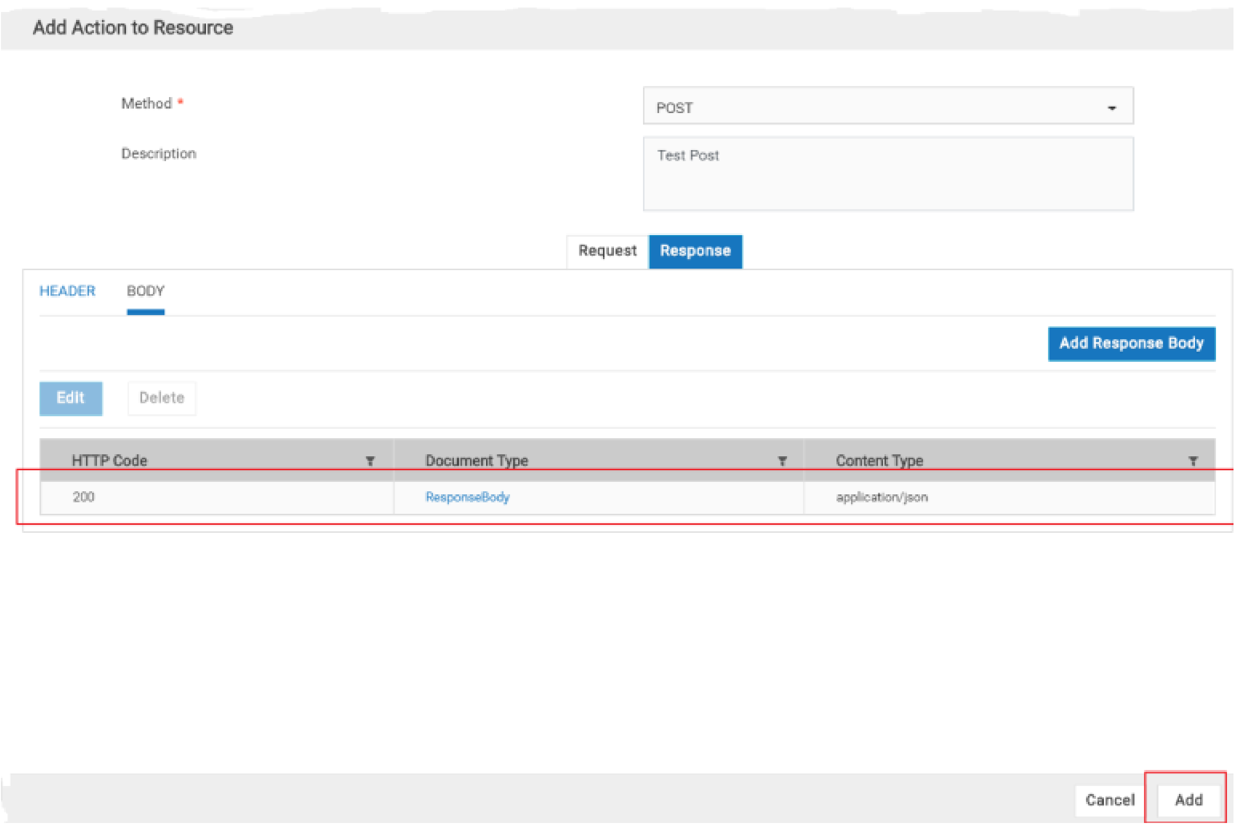
```
$ twurl -d 'status=hello' /1.1/statuses/update.json
```

Example Response

After you post successfully you should get back something that looks like this:

```
{
  "created_at": "Wed Oct 10 20:19:24 +0000 2018",
  "id": 1050118621198921700,
  "id_str": "1050118621198921728",
  "text": "To make room for more expression, we will now
skin t… https://t.co/MkGjXf9aXm",
  "source": "<a href="http://twitter.com" rel="nofollow">
  "truncated": true,
  "in_reply_to_status_id": null,
  "in_reply_to_status_id_str": null,
  "in_reply_to_user_id": null,
```

12. Click **Add** to add the Action.



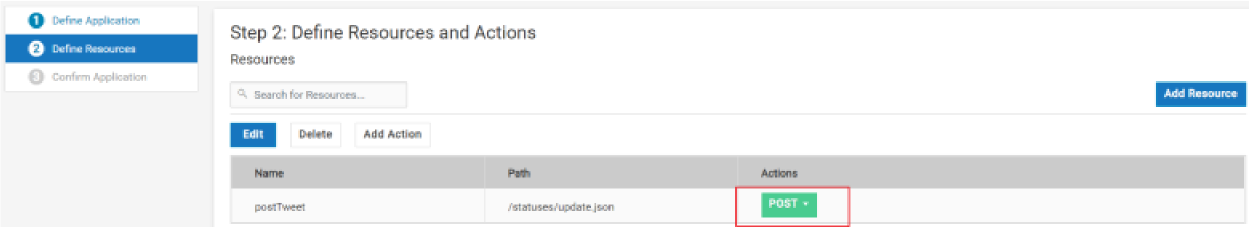
Add Action to Resource

Method * POST

Description Test Post

| HTTP Code | Document Type | Content Type |
|---|---|---|
| 200 | ResponseBody | application/json |

13. The **Actions** column displays the **POST** action.

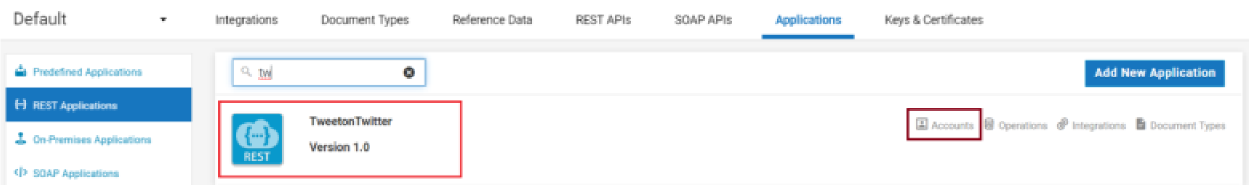


Step 2: Define Resources and Actions

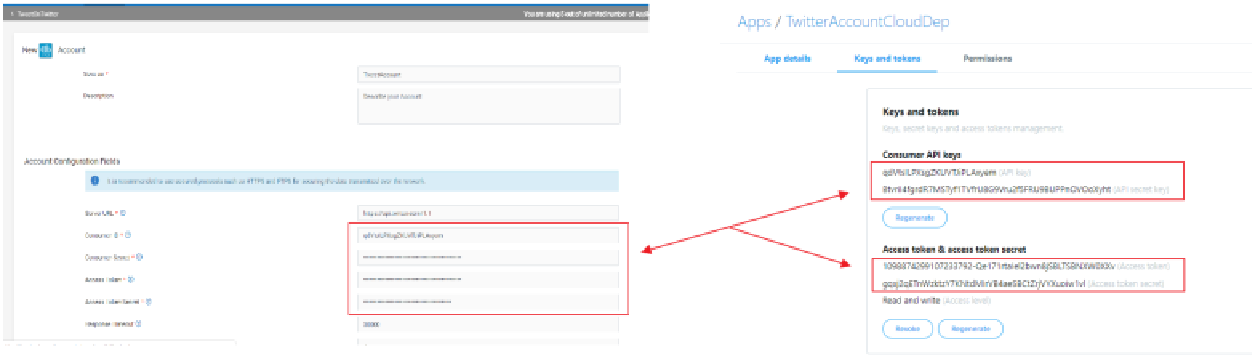
Resources

| Name | Path | Actions |
|---|---|---|
| postTweet | /statuses/update.json | POST ▾ |

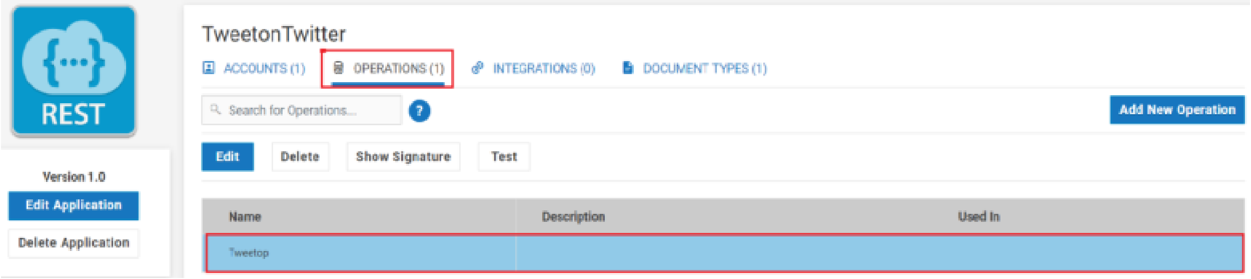
14. Click **Next** and then click **Finish** to create the REST Application **TweetOnTwitter**. Then create a new Account by clicking **Accounts** as shown below.



15. Then click **Add New Account**, fill in the Account configuration details obtained from the Twitter app (Consumer API Keys and Access Tokens) as shown below, and click **Save**.

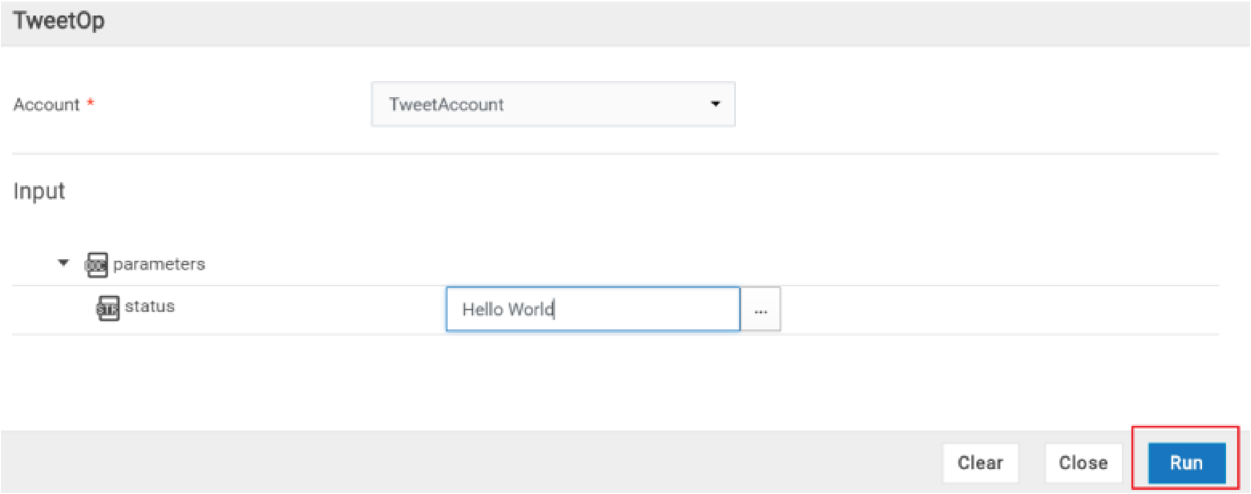


16. Now click **OPERATIONS > Add New Operation**. Provide all the details and create the **Tweetop** operation.



| Name | Description | Used In |
|---|---|---|
| Tweetop | | |

17. Select the operation, click **Test**, and then **Run** the operation.


TweetOp

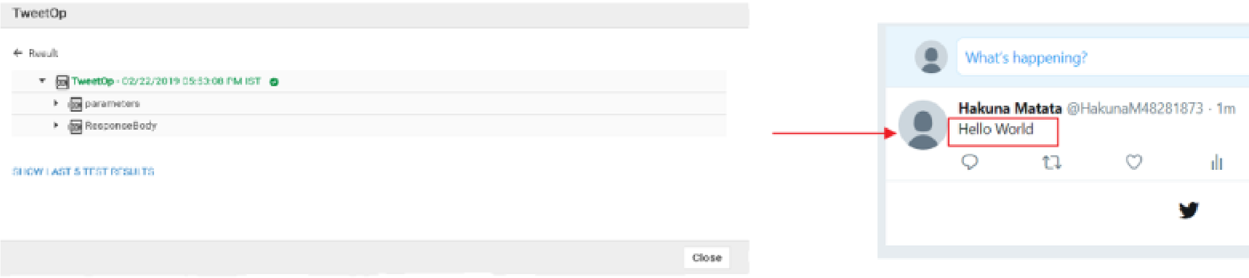
Account * TweetAccount

Input

parameters

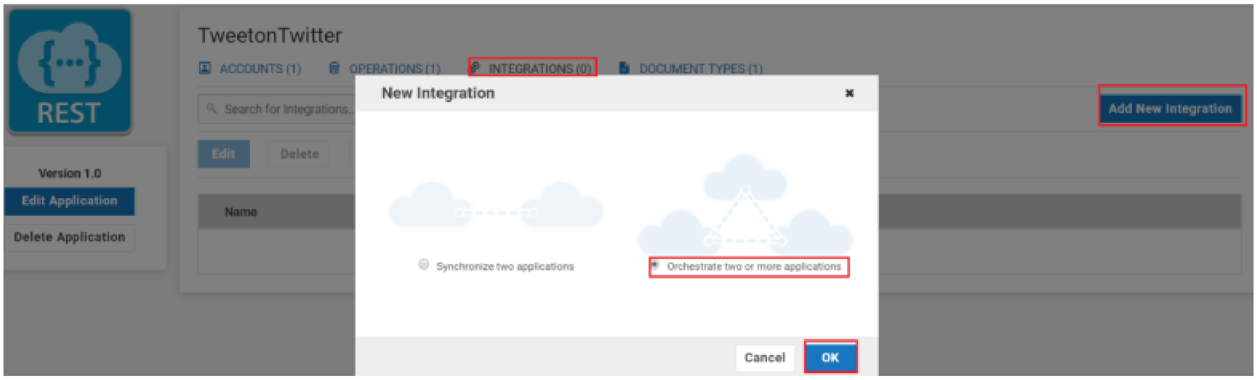
status Hello World

18. You will see that running the operation has triggered a tweet on your Twitter account.

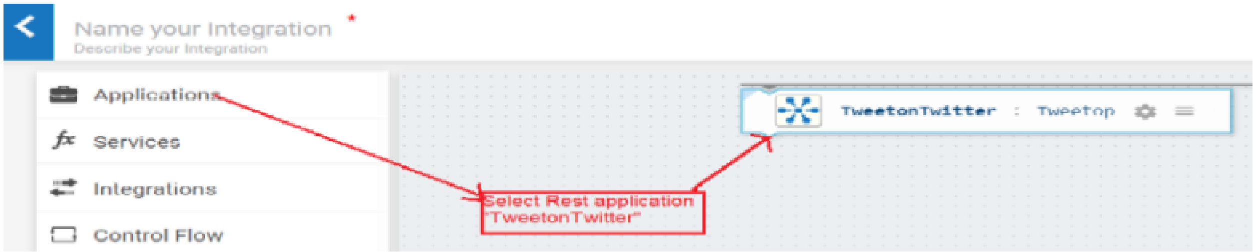


On Twitter Account of the User

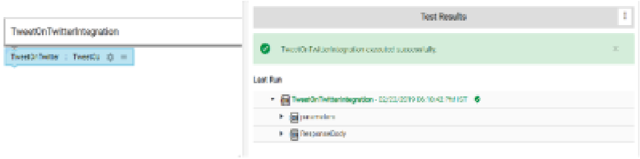
19. You can also trigger the REST Application from an integration. Click **INTEGRATIONS > Add New Integration**, and select the **Orchestrate two or more applications** option.



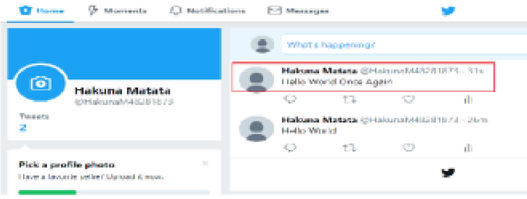
20. Setup the Integration as shown below using the REST Application, and then click **Save**.



21. Run and test the Integration. This will tweet as *Hello World Once Again*.


TweetOnTwitterIntegration
TweetOnTwitte : TweetOp
Test Results
TweetOnTwitterIntegration executed successfully.
Last Run
TweetOnTwitterIntegration - 02/22/2019 06:10:42 PM IST
parameters
ResponseBody

Integration Execution



On Twitter Account of the User

## Next Steps

Whenever you execute the Integration, you will see a new tweet on the user's Twitter account with your message.

## On-Premises Applications

On-Premises applications uploaded from on-premises systems are listed in the **On-Premises Applications** page, but you will not be able to create Accounts or Operations for on-premises applications. Those can be uploaded only from webMethods Integration Server. Further, when you upload services as part of an application from the on-premises webMethods Integration Server to webMethods Integration Cloud, the comments field of the service is uploaded and displayed in the webMethods Integration Cloud application. This field will be displayed if present and cannot be edited. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for more information.

If you select an Account for an on-premises Application and click **Test Connection**, the screen displays the status of the connection. If you have configured the Account details incorrectly in any stage, the state appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the state appears in green color and if an Account is not configured in a particular stage, the state appears in white color. For on-premises Applications, you can use the Account to execute services on the on-premises webMethods Integration Server.

Performance, scalability, and availability of on-premises connectivity for hybrid integration scenarios have been enhanced by having dedicated Software AG Universal Messaging (UM) nodes for each tenant. A tenant can be associated with dedicated UMs based on the license. Contact Software AG Global Support for assistance in setting up the dedicated hybrid infrastructure. If you are using hybrid connectivity and when you update the on-premises settings for the first time after the upgrade, restart the on-premises webMethods Integration Server to resume hybrid connectivity. If you have whitelisted the Cloud UM hostname or IP in the firewall, then you have to also whitelist the new UM hostname and IP along with the old ones. Click "here" on page 323 for information on the IP addresses.

**Note:**
The maximum message size, that is, the hard limit for message sizes that can be transported between on-premises webMethods Integration Server and Integration Cloud is 20 MB (20971520 bytes). This is the hard limit on the size of messages that can be transported over Software AG Universal Messaging between Integration Cloud and on-premises webMethods Integration Server. Overloading the Software AG Universal Messaging cluster with large payloads may adversely impact performance. If you need to transport larger volumes of data, consider an alternative approach that is better suited for file transfers, such as, webMethods ActiveTransfer.

### Dedicated infrastructure support for hybrid integration scenarios

Performance, scalability, and availability of on-premises connectivity for hybrid integration scenarios have now been enhanced by having dedicated Software AG Universal Messaging (UM) nodes for each tenant. A tenant can be associated with dedicated UMs based on the license. Contact Software AG Global Support for assistance in setting up the dedicated hybrid infrastructure.

If you are using hybrid connectivity and when you update the on-premises settings for the first time after the upgrade, restart the on-premises webMethods Integration Server to resume hybrid connectivity. If you have whitelisted the Cloud UM hostname or IP in the firewall, then you have to also whitelist the new UM hostname and IP along with the old ones. See the following table for information on the IP addresses.

| Zones | NAT Gateway IP Addresses | Hybrid Connectivity IP Addresses |
|---|---|---|
| **US, Oregon** | 52.39.87.1 | um.webmethodscloud.com |
| | 52.39.97.85 | 34.214.4.79 |
| | | 52.24.188.207 |
| **DE, Germany** | 35.157.214.174 | um.webmethodscloud.de |
| | 52.57.210.55 | 18.159.91.233 |
| | | 18.158.241.179 |
| **EU, Ireland** | 52.209.82.145 | um.webmethodscloud.eu |
| | 52.16.34.106 | 54.228.75.61 |
| | | 63.33.112.226 |

| Zones | UM Host Name | UM IP Addresses |
|---|---|---|
| **US, Oregon** | um-01-142ebc71-a.webmethodscloud.com | 52.27.56.188 |
| | um-02-142ebc71-a.webmethodscloud.com | 52.27.70.226 |
| | um-03-142ebc71-a.webmethodscloud.com | 52.27.83.194 |
| **DE, Germany** | um-01-e8c55c80-a.webmethodscloud.de | 52.57.68.150 |
| | um-02-e8c55c80-b.webmethodscloud.de | 52.28.72.223 |
| | um-03-e8c55c80-a.webmethodscloud.de | 52.59.84.12 |
| **EU, Ireland** | um-01-76de9912-a.webmethodscloud.eu | 52.212.8.149 |
| | um-02-76de9912-a.webmethodscloud.eu | 52.51.244.127 |
| | um-03-76de9912-a.webmethodscloud.eu | 52.50.237.197 |

## Uploading on-premises services to Integration Cloud

### Summary

In this tutorial, we will see how to upload on-premises services from webMethods Integration Server to Integration Cloud.

### Actors

- On-premises webMethods Integration Server user
- Integration Cloud instance to upload the on-premises application

- Integration Cloud user

## Before you begin

- Software AG Universal Messaging installed and running on-premises along with webMethods Integration Server. Software AG Universal Messaging is required for communication between on-premises webMethods Integration Server and Integration Cloud.

  **Note:**
  The maximum message size, that is, the hard limit for message sizes that can be transported between on-premises webMethods Integration Server and Integration Cloud is 20 MB (20971520 bytes). This is the hard limit on the size of messages that can be transported over Software AG Universal Messaging between Integration Cloud and on-premises webMethods Integration Server. Overloading the Software AG Universal Messaging cluster with large payloads may adversely impact performance. If you need to transport larger volumes of data, consider an alternative approach that is better suited for file transfers, such as, webMethods ActiveTransfer.

- Your Access Profile must have the permission to create Accounts, Operations, Applications, and Integrations.
- Your Access Profile must have the administer stages permission.

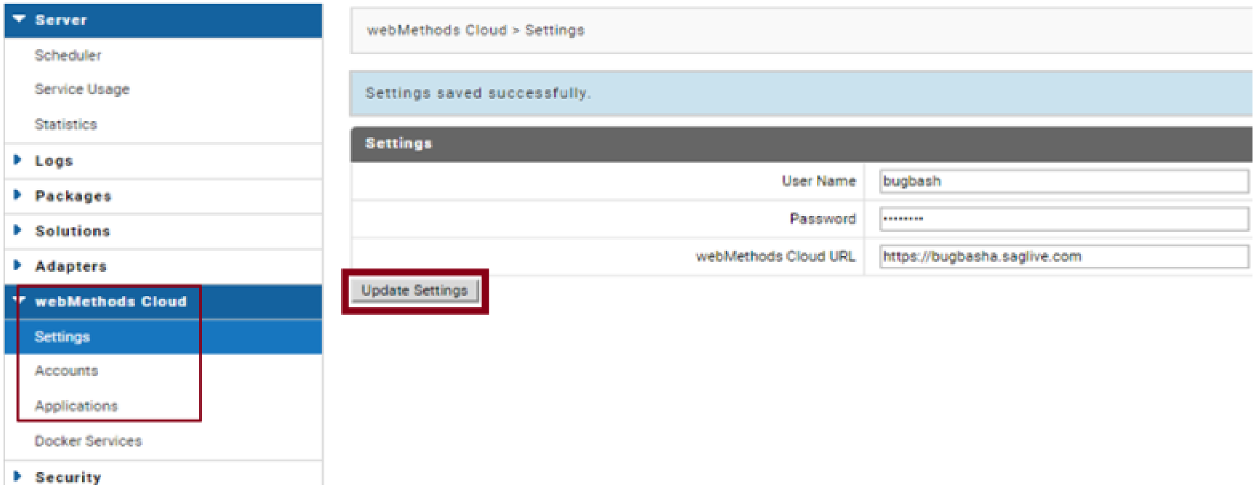
## Basic Flow

**Update Settings and create Account and Application in webMethods Integration Server.**

1. Log in to the webMethods Integration Server instance, for example, http://<hostName>:5555/.

   **Note:**
   Software AG Universal Messaging should be running.

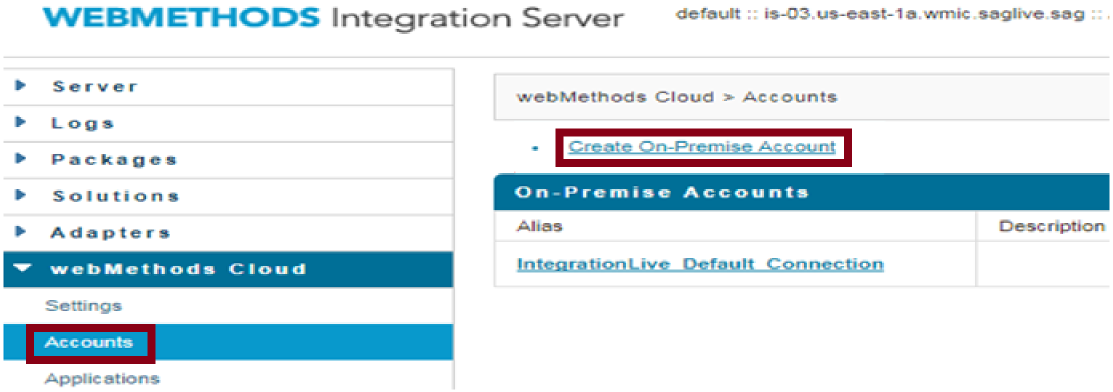
2. Go to **webMethods Cloud > Settings**.
3. Enter your Integration Cloud URL, for example, `https://{instanceName}.webmethodscloud.com`, user name, password, and click **Update Settings**.



Settings saved successfully.

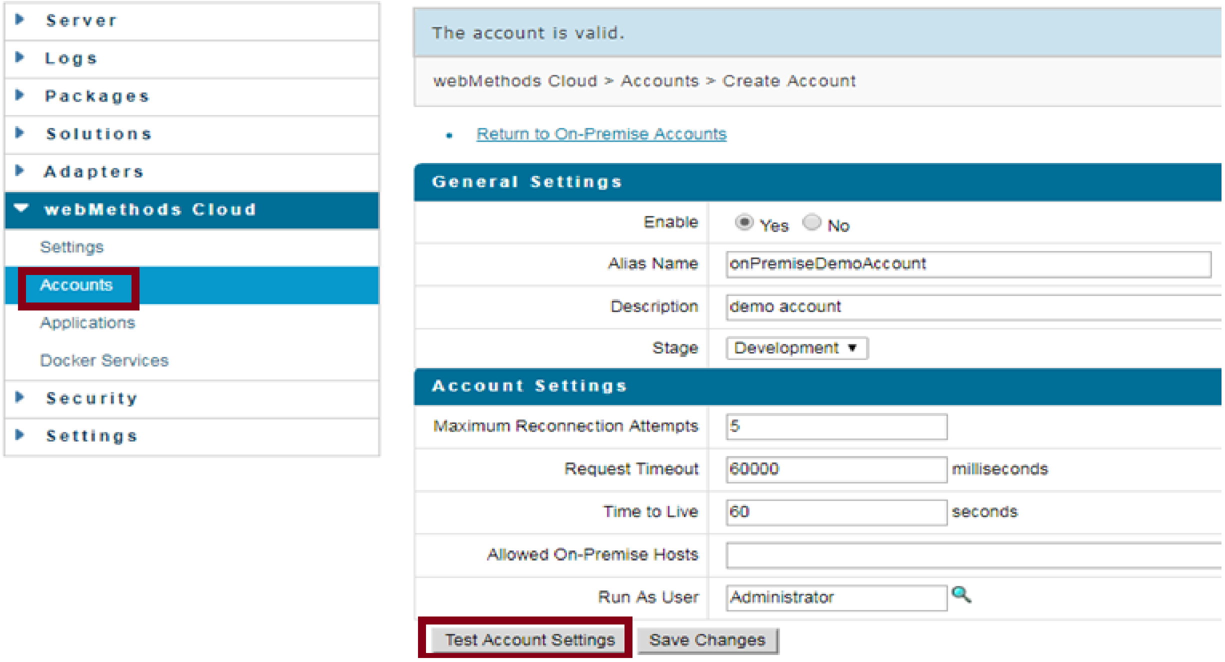
| | |
|---|---|
| User Name | bugbash |
| Password | ········ |
| webMethods Cloud URL | https://bugbasha.saglive.com |

4. Go to **webMethods Cloud > Accounts** and click **Create On-Premise Account**.


WEBMETHODS Integration Server
default :: is-03.us-east-1a.wmic.saglive.sag :: .
Server
Logs
Packages
Solutions
Adapters
webMethods Cloud
Settings
Accounts
Applications
webMethods Cloud > Accounts
Create On-Premise Account
On-Premise Accounts

| Alias | Description |
|---|---|
| IntegrationLive_Default_Connection | |

5. Enter the details as shown below on the **Create Account** page and click **Test Account Settings**.



The account is valid.

webMethods Cloud > Accounts > Create Account

- Return to On-Premise Accounts

**General Settings**

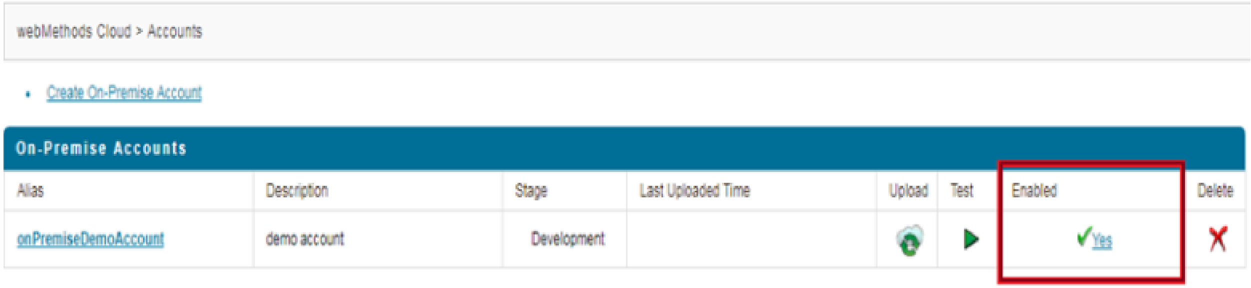
| | |
|---|---|
| Enable | ● Yes ○ No |
| Alias Name | onPremiseDemoAccount |
| Description | demo account |
| Stage | Development |

**Account Settings**

| | |
|---|---|
| Maximum Reconnection Attempts | 5 |
| Request Timeout | 60000 milliseconds |
| Time to Live | 60 seconds |
| Allowed On-Premise Hosts | |
| Run As User | Administrator |

**Note:**
The **Alias Name** is the name of the account that will appear on Integration Cloud and the **Stage** is the stage where you want to expose the webMethods Integration Server services.

6. Click **Save Changes** and check if the account is enabled.



webMethods Cloud > Accounts

- Create On-Premise Account

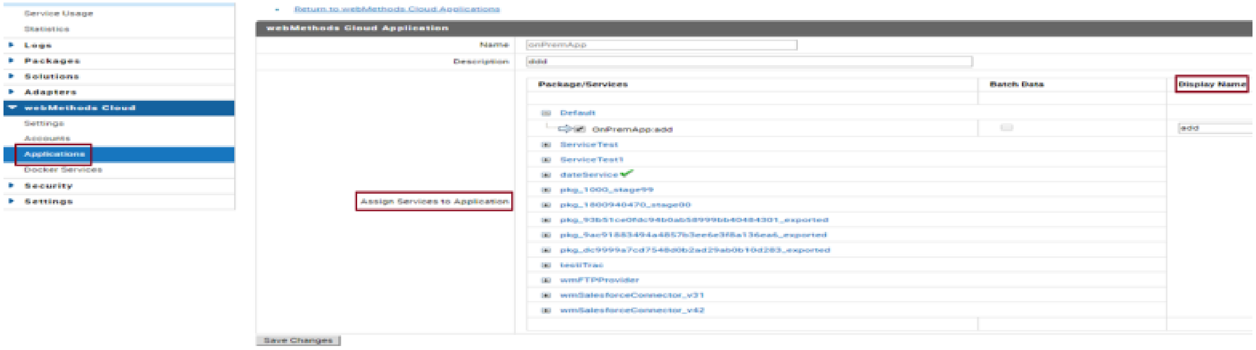
**On-Premise Accounts**

| Alias | Description | Stage | Last Uploaded Time | Upload | Test | Enabled | Delete |
|---|---|---|---|---|---|---|---|
| onPremiseDemoAccount | demo account | Development | | | ▶ | ✓Yes | ✗ |

7. Go to **webMethods Cloud > Applications** and click **Define webMethods Cloud Application**.

8. Select the webMethods Integration Server services that you want to expose on Integration Cloud, provide a **Display Name** for the service, and **Save** your changes.

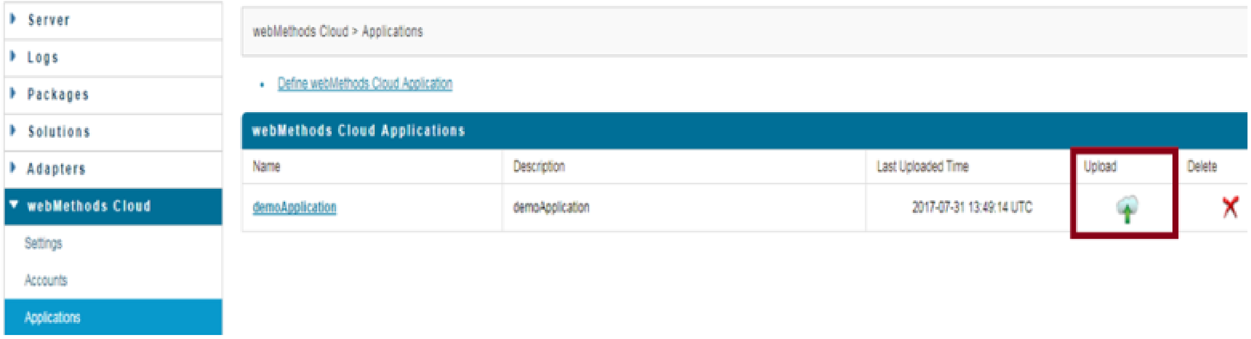
**Note:**
Services that have wrapper type fields, for example, fields of type other than String and IData in its Input or Output signature, cannot be exposed to Integration Cloud and they will not be available in the **Package/Services** column.



| | |
|---|---|
| Name | onPremApp |
| Description | ddd |

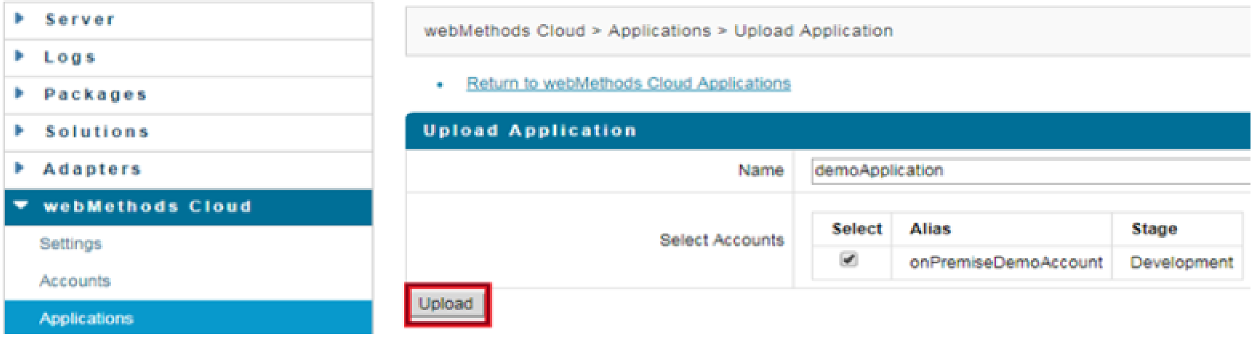
| Package/Services | Batch Data | Display Name |
|---|---|---|
| Default | | |
| OnPremApp:add | ☐ | add |
| ServiceTest | | |
| ServiceTest1 | | |
| dateService ✓ | | |
| pkg_1000_stage99 | | |
| pkg_1600940470_stage00 | | |
| pkg_93b51ce0fdc94b0ab58999bb40484301_exported | | |
| pkg_9ac91883494a4857b3ee6e3f8a136ea6_exported | | |
| pkg_dc9999a7cd7548d0b2ad29ab0b10d283_exported | | |
| testiTrac | | |
| wmFTPProvider | | |
| wmSalesforceConnector_v31 | | |
| wmSalesforceConnector_v42 | | |

9. Go to **webMethods Cloud > Applications** and click the upload icon.


Server
Logs
Packages
Solutions
Adapters
webMethods Cloud
Settings
Accounts
Applications
webMethods Cloud > Applications
Define webMethods Cloud Application
webMethods Cloud Applications

| Name | Description | Last Uploaded Time | Upload | Delete |
|---|---|---|---|---|
| demoApplication | demoApplication | 2017-07-31 13:49:14 UTC | ⬆ | ✗ |

10. Select the account and click **Upload**. The Application will be uploaded to Integration Cloud.

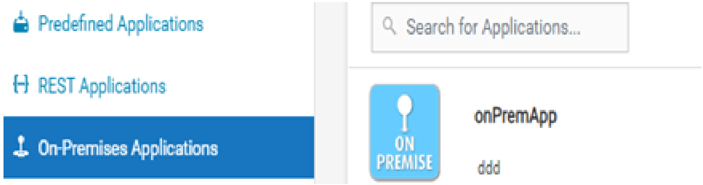


Name: demoApplication

Select Accounts

| Select | Alias | Stage |
|---|---|---|
| ☑ | onPremiseDemoAccount | Development |

11. Log in to your Integration Cloud instance and go to **Projects > <Select a Project> > Applications > On-Premises Applications**. The on-premises Application appears in the list. Use the operations of the on-premises Application to create Integrations.



## SOAP Applications

The SOAP Application enables you to access third party Web Services hosted in the cloud. The SOAP Application uses a WSDL to create consumer operations.

**The following features are supported for SOAP Applications:**

- A SOAP Application implementation follows the WS-I Basic Profile 1.1 specification.
- SOAP Applications can be created by uploading a WSDL file or by using a valid WSDL URL that can be accessed over a network.
- SOAP Applications can be created with WSDLs that are annotated with WS-Security Policy/Policies.
- SOAP Applications with SOAP version 1.1 and 1.2 and Style/Use as Document/Literal and RPC/Literal (RPC/Encoded model is not supported for SOAP version 1.2).
- The following SOAP Binding types are supported:
    - SOAP over HTTP.
    - SOAP over HTTPS.
- Authentication type: HTTP Basic Token.

**SOAP Applications have the following restrictions:**

- The WSDL and associated schema(s) must be accessible through a publicly or locally accessible URL.
- Only WSDLs with WS-Security policies are supported. Any other policies, for example, WS-Addressing, WS-Reliable Messaging, and so on, are not supported. If you create SOAP Applications with WSDLs having non-WS-Security Policies, exceptions may appear while executing Integrations.
- Manual addition of WS-Security Policies in a SOAP Application is not supported. SOAP Applications with WS-Security can be created with only policy-annotated WSDLs, that is, WSDLs that already have WS-Security Policies annotated in them.
- SOAP over JMS is not supported.
- Only Basic Authentication is supported. Other authentication types such as Digest, NTLM, and Kerberos are not supported.
- You will not be able to attach or upload a file while executing an Integration.

### To add a SOAP Application

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > SOAP Applications > Add New Application**.

2. Provide a name and description of your SOAP Application. The description you enter here will appear in the **SOAP Applications** page. Required fields are marked with an asterisk on the screen.

3. Click **Browse** next to the **Application Icon** if you want to select a different icon for your SOAP Application. The icon must be a PNG file and the size cannot exceed 50 KB, else the default image is displayed.

4. Click **Next** and specify the **WSDL Source**.

   - Select **URL** if you want to specify the URL of the WSDL. The URL should begin with http:// or https://. The URL is used to retrieve the WSDL for the SOAP Application.
   - Select **File** and then click **Browse** if you want to select the WSDL from your local file system. The size of the WSDL cannot exceed 5 MB. You can click the ⊞ icon beside the **Browse** button if you want to add separate elements of a service definition after import, such as WSDLs or XSDs, to the primary WSDL.

   **Note:**
   Ensure that you add the primary WSDL as the first WSDL, and then add separate elements of the service definition, for example, dependent WSDLs and XSDs to the primary WSDL.

5. Enter the user name and password in the **Authentication** section if authentication is required to access the WSDL URL.

6. Click **Next** to review the details you have entered.

7. Click **Finish** to create the SOAP Application.

**Editing SOAP Applications**

From the **SOAP Applications** page, click the SOAP Application link, and then click **Edit Application**. You can change the **Description** and the **Application Icon**.

In the Application details page, **Update WSDL** section, select **No, keep existing WSDL** if you do not want to modify the WSDL URL or the WSDL file. Select **Yes, override WSDL** if you want to specify a new WSDL URL or upload a new WSDL file in the **WSDL Source** section.

Confirm the updated Application. After you click **Finish**, the **Update SOAP Application** window appears, which provides a summary of the impacted Accounts, Operations, and Integrations. Click **Update** to update the SOAP Application. Updating the WSDL may result in addition or removal of Operations or fields in the Input/Output signature of an Operation. This may lead to incorrect mappings if you have used that Operation in an Integration. To delete a SOAP Application, click **Delete Application**.

## SOAP Applications - Account Configuration Details

Integration Cloud allows you to create Custom SOAP Applications. Custom SOAP Applications enable you to access third party web services hosted in the cloud or on-premises environment. The Custom SOAP Application uses a WSDL that is accessible through publicly or locally accessible URLs.

| Field | Description |
|---|---|
| **Port Binding** | Select the bind address from the drop-down list, that is, the concrete protocol and data format specification for the web service. |
| **URL** | This is the URL for the web service. You can edit the URL to specify a different web service endpoint. |
| **Response Timeout** | The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the response timeout value. It is recommended to specify a value other than 0.<br>If you specify 0, Integration Cloud will wait indefinitely for a response.<br>If you do not specify a timeout, Integration Cloud will consider 5 minutes as the response timeout.<br>If the connection to the host provider ends before Integration Cloud receives a response, the web service operation ends with an exception and a status code of 408. |
| **User** | User name used to authenticate the consumer at the HTTP or HTTPS transport level on the host server. |
| **Password** | The password used to authenticate the consumer on the host server. |
| **Keystore Alias** | Alias to the keystore that contains the private key used to connect to the Web Service host securely. You can also add a new Keystore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Keystores. |
| **Key Alias** | Alias to the key in the keystore that contains the private key used to connect to the Web Service host securely. The key must be in the keystore specified in the **Keystore Alias** field. |

**Show Advanced Options** - WS-Security properties are used by the SOAP processor to provide security information in the WS-Security header of the SOAP message.

| Field | Description |
|---|---|
| **Security Credentials** | |
| **User Name** | Name to include with the Username Token, if the Web Service's security policy requires one. |
| **Password** | The password to include with the UsernameToken (must be plain text). |
| **Keystore / Truststore** | |
| **Keystore Alias** | The alias for the keystore, which contains private keys and certificates associated with those private keys. You can also add a new Keystore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Keystores. |
| **Key Alias** | The text identifier for the private key associated with the **Keystore Alias**. |
| **Truststore Alias** | The alias for the truststore, which contains the trusted root of a certificate or signing authority (CA). You can also add a new Truststore from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Truststores. |
| **Partner Certificate Alias** | The file that contains the partner's self-signed certificate. You can also add a new Partner Certificate from this field.<br>**Note:**<br>Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Partner Certificates. |
| **Timestamp** | |
| **Timestamp Precision** | Whether the timestamp placed in the Timestamp element of the security header of an outbound message is precise to seconds or milliseconds. If the precision is set to milliseconds, the timestamp format yyyy-MM-dd'T'HH:mm:ss:SSS'Z' is used. If the precision is set to seconds, the timestamp format yyyy-MM-dd'T'HH:mm:ss'Z' is used. |
| **Timestamp TTL** | The time-to-live value for an outbound message in seconds. This value is used to set the expiry time in the Timestamp element of outbound |

| Field | Description |
|---|---|
| | messages. The timestamp precision value is used only when WS-Security is implemented through a WS-Policy. |
| **Timestamp Max Skew** | The maximum number of seconds that the Web Services client and host clocks can differ so that the timestamp expiry validation does not fail. The timestamp precision value is used only when WS-Security is implemented through a WS-Policy. The inbound SOAP message is validated only if the creation timestamp of the message is less than the sum of the timestamp maximum skew value and the current system clock time. |

## SOAP Signature

All SOAP Application operations have an identical input and output signature with the exception of the variables used to represent the input and output messages. For information about how a SOAP Application operation represents the input and output messages in the signature, see *How a SOAP Application Operation represents the Input and Output Messages*.

### How a SOAP Application Operation represents the Input and Output Messages

How a SOAP Application operation represents the contents of the input and output message in the signature depends on the style/use of the binder for the SOAP Application operation.

- For a SOAP Application operation that uses a style/use of Document/Literal:
  - The input signature contains an optional document reference to a document type created to represent the operation input message. At run time, if you do not specify any input for the document reference variable or any of its child variables, Integration Cloud sends an empty SOAP body in the SOAP message.
  - The output signature contains a document reference to a document type created to represent the operation output message. This document reference is conditional and is only returned by the SOAP Application operation if the SOAP Application operation executes successfully. If returned at run time, this document reference contains the response from a successful invocation of a SOAP Application operation. If the SOAP Application operation receives a SOAP fault, it is converted to an exception that can be caught with the try catch block in an Orchestrated Integration.
- For a SOAP Application operation that uses a style/use of RPC/Encoded or RPC/Literal:
  - The input signature contains variables that represent the top-level elements in the operation input message. All of these variables are optional. At run-time, if you do not specify any input for the variable (or variables) that represent the input message, Integration Cloud sends an empty SOAP body in the SOAP message.
  - The output signature contains variables that represent the top-level elements in the operation output message. All of these variables are conditional and are only returned by the SOAP Application operation if the SOAP Application operation executes successfully. If returned,

these variables contain the response from a successful invocation of a SOAP Application operation.

## Input Parameters

*transportHeaders* **Document** Optional. Transport-specific header fields that you want to explicitly set in the request. Specify a key in *transportHeaders* for each header field that you want to set, where the key's name represents the name of the header field and the key's value represents the value of that header field.

The names and values supplied to *transportHeaders* must be of type String. For information about using *transportHeaders* with HTTP/S requests including a description of the default behavior, see *Setting Transport Headers for HTTP/S*.

## Output Parameters

*transportInfo* **Document** Conditional. Headers from response and request messages.

The contents of the transportInfo vary depending on the actual transport (HTTP or HTTPS) used.

*transportInfo* contains the following keys:

| Key | Value |
|---|---|
| *requestHeaders* | **Document** Conditional. Header fields from the request message. The contents of the *requestHeaders* document are not identical to *transportHeaders* used as input. The transport can add, remove, or alter specific headers while processing the request.<br>Whether or not the SOAP Application operation returns the *requestHeaders* parameter depends on the success or failure of the operation. In the case of failure, the point at which the failure occurs determines the presence of the *requestHeaders* parameter. For more information, see *Transport and Exceptions Returned by a SOAP Application Operation.*<br>**For the HTTP or HTTPS transports**, the *requestHeaders* parameter will not contain any HTTP headers that the transport mechanism added or modified when sending the request. |
| *responseHeaders* | **Document** Conditional. Header fields from the response. Each key in *responseHeaders* represents a field (line) of the response header. Key names represent the names of header |

fields. The keys' values are Strings containing the values of the fields.

Whether or not the SOAP Application operation returns the *responseHeaders* parameter depends on the success or failure of the operation. In the case of failure, the point at which the failure occurs determines the presence of the *responseHeaders* parameter. For more information, see *Transport and Exceptions Information Returned by a SOAP Application Operation.*

**For the HTTP or HTTPS transports**, the *responseHeaders* parameter contains any HTTP/HTTPS headers present in the response.

*status*

**String** Conditional. Status code from the request, returned by the underlying transport.

For more information about status codes and status messages returned by a SOAP Application operation, see *Transport and Exceptions Information Returned by a SOAP Application Operation.*

*statusMessage*

**String** Conditional. Description of the status code returned by the transport.

For more information about status codes and status messages returned by a SOAP Application operation, see *Transport and Exceptions Information Returned by a SOAP Application Operation.*

### Setting Transport Headers for HTTP/S

When creating a service that executes a SOAP Application operation, you can pass transport header information directly into the SOAP Application operation by passing name/value pairs in to the *transportHeaders* input parameter. When creating the SOAP request, Integration Cloud adds a transport header for each name/value pair.

Keep the following information in mind when setting *transportHeaders* for an HTTP/S request:

- Specify a key in *transportHeaders* for each header field that you want to set, where the key’s name represents the name of the header field and the key’s value represents the value of that header field.
- The names and values supplied to *transportHeaders* must be of type String. If a transport header has a name or value that is not of type String, the header will not be included in the message.
- For any header name/value pair supplied in *transportHeaders* for an HTTP/S request, Integration Cloud simply passes through the supplied headers and does not perform any validation for the headers beyond verifying that the name and value are of type String.

- If you do not set *transportHeaders* or do not specify the following header fields in *transportHeaders*, Integration Cloud adds and specifies values for the following standard header fields:
  - `Accept`
  - `Authorization`
  - `Connection`
  - `Content-Type`
  - `SOAPAction` (Added when *soapProtocol* is SOAP 1.1 only)
  - `User-Agent`

  > **Note:**
  > Pass in the preceding headers to *transportHeaders* only if you are an experienced SOAP Application developer. Incorrect header values can result in failure of the request.

- For a SOAP Application operation, Integration Cloud sets the value of the `Host` header and overwrites any supplied value.
- If you specify `Authorization` in *transportHeaders*, the values specified for the *auth/transport* document and its children will not be used in the `Authorization` header.
- If you specify `Content-Type` in *transportHeaders* and the SOAP Protocol is SOAP 1.2, Integration Cloud ignores the value of `soapAction` obtained from the WSDL used to create the SOAP Application operation.
- If you specify the `SOAPAction` header in *transportHeaders* and the SOAP Protocol is SOAP 1.1, Integration Cloud ignores the value of `SOAPAction` obtained from the WSDL used to create the SOAP Application operation.
- Integration Cloud sets the value of `Content-Length` automatically and overwrites any value passed in to *transportHeaders*.
- Integration Cloud automatically adds the `Cookie` header to the HTTP header and supplies any cookies established between Integration Cloud and the HTTP server with which it is interacting. If you supply the `Cookie` header to *transportHeaders*, Integration Cloud prepends the values you supply to the already established `Cookie` header value.
- The following headers are considered to be standard and require the specified capitalization: `Accept`, `Authorization`, `Connection`, `Content-Type`, `Cookie`, `Host`, `SOAPAction`, `User-Agent`.

  > **Important:**
  > Using capitalization other than that which is specified results in undefined behavior.

  > **Important:**
  > Supplying duplicate entries for any standard header results in undefined behavior.

### Transport and Exceptions Returned by a SOAP Application Operation

The transport information, such as headers, status codes, and status messages, returned by a SOAP Application operation varies depending on the following:

- The transport used to send and receive the SOAP message
- The success or failure of the SOAP Application operation
- The point at which failure occurs
- The message exchange pattern (MEP) for the operation

**Note:**
Transport information is returned in the *transportInfo* output parameter.

If the SOAP Application operation receives a SOAP fault, it is converted to an exception that can be caught with the try catch block in an Orchestrated Integration.

## Flat File Applications

You can use the Flat File Application to translate documents into and from flat file formats. To set up the translation, you create the *definition and structure of the Flat File Application, which is called a flat file schema*. The schema also contains the instructions for parsing or creating the flat file and defines how to identify individual records within a flat file and what data is contained in each of those records.

**What is a flat file definition and structure?**

The definition and structure of a flat file Application contains the instructions for parsing or creating a flat file. It details the structure of the document, including delimiters, records, and repeated record structures. It also acts as the model against which you can validate an inbound flat file. A flat file structure consists of hierarchical elements that represent each record, field, and subfield in a flat file.

**What are the different approaches to create a flat file Application?**

You can create a flat file Application using any one of the following approaches:

- “Create manually” on page 338: In this approach, you define the definition and structure of a flat file Application and then manually add the elements or properties.
- “Use a sample file” on page 350: In this approach, you use a sample file to define the definition and structure of a flat file Application. Here, you use the automated wizards to create the structure of the flat file.

**What are the high-level steps to create a flat file Application?**

**Step 1** **Define the Application details.** In this stage, you define the Application details and the approach. You can create the flat file Application either manually or by using a sample file.

**Step 2** **Define the record parser and specify a record identifier.** In this stage, you associate a record parser with the flat file Application that will process flat files inbound to

Integration Cloud. You also specify how you want the record to be identified after it is parsed.

**Step 3** **Define the flat file structure.** If you are creating the flat file Application manually, in this stage, you specify the hierarchical structure of the flat file by creating and nesting the record definitions.

**Invoke Operations.**

After creating the flat file Application, create an Orchestrated Integration, select the Flat File Application created, and invoke the following predefined operations:

- "convertFlatFileToDocument" on page 355 - Converts the flat file to a document (inbound)
- "convertDocumentToFlatFile" on page 357 - Converts a document to a flat file (outbound)

## Creating a flat file Application manually

Integration Cloud can process flat files in which:

- The records in the flat file are defined using one of the following methods:

| Property | Description |
|---|---|
| **Delimiter** | Each record in the flat file is separated by a delimiter. |
| **Fixed length** | Each record is a fixed number of bytes (for example, mainframe punch or print records). |
| **Variable length** | Each record is *preceded* by two bytes that indicate the length of the record. Records in the flat file can have different lengths. |

- If the flat file contains record identifiers, the record identifiers must be located in the same location in all records in the file.

Integration Cloud can then identify fields in these records based on:

- **Fixed position** - Each field is defined by 1) the number of bytes from the beginning of the record and 2) the field length. This can be used regardless of whether a field delimiter has been specified.
- **Delimiters** - Each field is separated by a delimiter, and you can specify the *N*th delimited field in a record to represent the record identifier. This can be used only when a field delimiter (and, if necessary, subfield delimiter) has been specified.

**To create a flat file Application manually**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Flat File Applications > Add New Application**.

2. Type a name and description of the flat file Application and select the creation mode as **Create manually**.

3. Click **Next** and in the **Flat File Definition** page, configure the **Record parser** and the **Record identifier**.

4. In the **Record parser** section, to configure a delimited record parser, select **Delimiter** in the **Record parser** area and specify the following fields.

   For a record delimiter, you can specify a character (for example, !) or character representation (for example, `\r\n` for carriage return).

   Use the **Delimiter** record parser for the Flat File Application when each record is separated by a delimiter.

| Delimiter | Record type | Description |
|---|---|---|
| **Record** | **Character** | Character that separates records in a flat file document. |
| | --OR-- | |
| | **Character Position** | Starting from the beginning of the document and counting from zero (0), the character position at which the record delimiter for this document is located.<br>**Note:** For example, if you specify 3 as the character position, you have indicated that the record delimiter appears in the fourth character position from the beginning of the document. |

| Delimiter | Field or composite type | Description |
|---|---|---|
| **Field or composite** | **Character** | Character that separates fields in a flat file document. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the field delimiter for this document is located.<br>**Note:** |

| Delimiter | Field or composite type | Description |
|---|---|---|
| | | For example, if you specify 4 as the character position, you have indicated that the field delimiter appears in the fifth character position from the beginning of the document. |

| Delimiter | Subfield type | Description |
|---|---|---|
| **Subfield** | **Character** | Character that separates subfields in a flat file document. The default is a period “.”. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the subfield delimiter for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the subfield delimiter appears in the sixth character position from the beginning of the document. |

| Delimiter | Quoted release character type | Description |
|---|---|---|
| **Quoted release character** | **Character** | Character used to enable a section of text within a field to be represented as its literal value. Any delimiter characters that appear within this section will not be treated as delimiters.<br>**Note:**<br>For example, your field delimiter is (,) and your release character is “. When you want to use (,) within a field as text, you must prefix it with your quoted release character. When using the convertFlatFileToDocument operation to create the strings `Doe, John` and `Doe, Jane`, the record would appear as “`Doe, John`”,“`Doe, Jane`”. When using the |

| Delimiter | Quoted release character type | Description |
|---|---|---|
| | | convertDocumentToFlatFile operation to create “`Doe, John`”,“`Doe, Jane`”, the value of the record would be `Doe, John and Doe, Jane`. When using the convertDocumentToFlatFile operation, if you have specified both the Release Character and the Quoted Release Character, the Quoted Release Character will be used. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the quoted release character for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the quoted release character appears in the sixth character position from the beginning of the document. |

| Delimiter | Release character type | Description |
|---|---|---|
| **Release character** | **Character** | Character used to enable a delimiter to be used for its intended, original meaning. The character following the release character will not be treated as a delimiter.<br>**Note:**<br>For example, your field delimiter is + and your release character is \. When using + within a field as text, you must prefix it with your release character. When using the convertFlatFileToDocument operation to create the strings a+b+c and d+e+f, the record would appear as a\+b\+c+d\+e\+f. When using the convertDocumentToFlatFile operation to create a\+b\+c+d\+e\+f, the |

| Delimiter | Release character type | Description |
|---|---|---|
| | | value of the record would be a+b+c and d+e+f. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the field delimiter for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the field delimiter appears in the sixth character position from the beginning of the document. |

Next, set the record identifier.

5. Use a **Fixed Length** record parser type when each record is of a fixed length (for example, mainframe punch or print records). This parser splits a file into records of the same pre-specified length. To configure a fixed length record parser, in the **Record Parser Type** area, select **Fixed Length**.

   Specify the following fields:

| Fixed Length | Field or composite type | Description |
|---|---|---|
| **Field or composite** | **Character** | Character that separates fields or composites in a flat file document. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the field delimiter for this document is located.<br>**Note:**<br>For example, if you specify 4 as the character position, you have indicated that the field delimiter appears in the |

| Fixed Length | Field or composite type | Description |
|---|---|---|
| | | fifth character position from the beginning of the document. |

| Fixed Length | Subfield type | Description |
|---|---|---|
| **Subfield** | **Character** | Character that separates subfields in a flat file document. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the subfield delimiter for this document is located.<br>**Note:** For example, if you specify 5 as the character position, you have indicated that the subfield delimiter appears in the sixth character position from the beginning of the document. |

| Fixed length | Quoted release character type | Description |
|---|---|---|
| **Quoted release character** | **Character** | Character used to enable a section of text within a field to be represented as its literal value. Any delimiter characters that appear within this section will not be treated as delimiters.<br>**Note:** For example, your field delimiter is (,) and your release character is “. When you want to use (,) within a field as text, you must prefix it with your quoted release character. When using the convertFlatFileToDocument operation to create the strings `Doe, John` and `Doe, Jane`, the record would appear as “`Doe, John`”,“`Doe, Jane`”. When using the convertDocumentToFlatFile operation to create “`Doe, John`”,“`Doe, Jane`”, the |

| Fixed length | Quoted release character type | Description |
|---|---|---|
| | | value of the record would be `Doe, John and Doe, Jane`.When using the convertDocumentToFlatFile operation, if you have specified both the Release Character and the and the Quoted Release Character, the Quoted Release Character will be used. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the quoted release character for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the quoted release character appears in the sixth character position from the beginning of the document. |

| Fixed length | Release character type | Description |
|---|---|---|
| **Release character** | **Character** | Character used to enable a delimiter to be used for its intended, original meaning. The character following the release character will not be treated as a delimiter.<br>**Note:**<br>For example, your field delimiter is + and your release character is \. When using + within a field as text, you must prefix it with your release character. When using the convertFlatFileToDocument operation to create the strings a+b+c and d+e+f, the record would appear as a\+b\+c+d\+e\+f. When using the convertDocumentToFlatFile operation to create a\+b\+c+d\+e\+f, the value of the record would be a+b+c and d+e+f. |

| Fixed length | Release character type | Description |
|---|---|---|
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the field delimiter for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the field delimiter appears in the sixth character position from the beginning of the document. |

| Fixed length | Description |
|---|---|
| **Record length** | In the **Record Length** field, enter the length, in characters, of each record in the flat file. Record length cannot be empty. |

Next, set the record identifier.

6. The **Variable Length** record parser type expects each record to be preceded by two bytes that indicate the length of the record. Each record may be a different length.

   To configure a variable length record parser, in the **Record Parser** area, select **Variable length**, and specify the following fields:

| Variable Length | Field or composite type | Description |
|---|---|---|
| **Field or composite** | **Character** | Character that separates fields or composites in a flat file document. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the field delimiter for this document is located.<br>**Note:**<br>For example, if you specify 4 as the character position, you have indicated that the field delimiter appears in the |

| Variable Length | Field or composite type | Description |
|---|---|---|
| | | fifth character position from the beginning of the document. |

| Variable length | Subfield type | Description |
|---|---|---|
| **Subfield** | **Character** | Character that separates subfields in a flat file document. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the subfield delimiter for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the subfield delimiter appears in the sixth character position from the beginning of the document. |

| Variable length | Quoted release character type | Description |
|---|---|---|
| **Quoted release character** | **Character** | Character used to enable a section of text within a field to be represented as its literal value. Any delimiter characters that appear within this section will not be treated as delimiters.<br>**Note:**<br>For example, your field delimiter is (,) and your release character is “. When you want to use (,) within a field as text, you must prefix it with your quoted release character. When using the convertFlatFileToDocument operation to create the strings `Doe, John` and `Doe, Jane`, the record would appear as “`Doe, John`”,“`Doe, Jane`”. When using the convertDocumentToFlatFile operation to create “`Doe, John`”,“`Doe, Jane`”, the value of the |

| Variable length | Quoted release character type | Description |
|---|---|---|
| | | record would be `Doe, John and Doe, Jane`.When using the convertDocumentToFlatFile operation, if you have specified both the Release Character and the and the Quoted Release Character, the Quoted Release Character will be used. |
| | --OR-- | |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at which the quoted release character for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the quoted release character appears in the sixth character position from the beginning of the document. |

| Variable length | Release character type | Description |
|---|---|---|
| **Release character** | **Character** | Character used to enable a delimiter to be used for its intended, original meaning. The character following the release character will not be treated as a delimiter.<br>**Note:**<br>For example, your field delimiter is + and your release character is \. When using + within a field as text, you must prefix it with your release character. When using the convertFlatFileToDocument operation to create the strings a+b+c and d+e+f, the record would appear as a\+b\+c+d\+e\+f. When using the convertDocumentToFlatFile operation to create a\+b\+c+d\+e\+f, the value of the record would be a+b+c and d+e+f. |
| | **Character position** | Starting from the beginning of the document and counting from zero (0), the character position at |

| Variable length | Release character type | Description |
|---|---|---|
| | | which the field delimiter for this document is located.<br>**Note:**<br>For example, if you specify 5 as the character position, you have indicated that the field delimiter appears in the sixth character position from the beginning of the document. |

Next, set the record identifier.

7. Specifying a Record Identifier

When parsing a file, Integration Cloud looks at a record and extracts an identifier out of the data and uses that identifier to connect the record definition with a particular record in the flat file. The name of the record definition must match the value obtained by the record identifier.

To set the record identifier for a definition, if the flat file contains a record identifier, select **Yes** in the **Record identifier** area, and set the record identifier to one of the following values.

| Record identifier | Value | Description |
|---|---|---|
| | **Start at position** | Identifies the character position in the record (counting from zero) where the record identifier is located. **Start at position** record identifiers compare the value that occurs in the record, at the specified offset, to all the record names defined in the flat file structure. Note that the **Start at position** identifier cannot distinguish between all types of record names.<br>**Note:**<br>For example, if you name records “Rec1” and “Rec,” some instances of “Rec1” may be identified as “Rec,” because “Rec1” begins with “Rec.” |
| | **Nth field** | Identifies the field in the record (counting from zero) that contains the identifier. **Nth field** identifiers use the value of the specified field as the record identifier. These identifiers count from zero (0).<br>**Note:** |

| Record identifier | Value | Description |
|---|---|---|
| | | For example, if 2 is specified, the third field is used as the record identifier. |

Select **No** if the flat file does not contain a record identifier.

8. Click **Next** to define the **Flat File Structure**. Use the **Flat File Structure** page to add records to the flat file structure and define the hierarchical relationships between them.

   The application receiving the flat file uses the structure to read the flat file. This structural information identifies the parent-child relationships between different records in the flat file. By nesting record elements in the flat file structure (adding record elements to a record), you can represent the hierarchical structure of the data in the flat file.

   To add the first record, click the **Add new element** icon and in the **Element type** field, select **Record Definition**. Specify a name for the record definition and click **Finish**.

9. To add a **Composite Definition** after you have added a record definition, specify the following:

| Extractor type | Description |
|---|---|
| **Nth field** | Field number in the record that contains the composite you want to extract. This pulls the subfield data from the composite. If you leave this property empty, the composite will not be extracted. |

To add a **Field Definition** after you have added a record definition, specify the following:

| Extractor type | Description | |
|---|---|---|
| **Fixed Position** | Counting from zero (0), indicates a fixed number of bytes to be extracted from a record. | |
| | **Position** | Type the first byte to extract from the record. Type the first byte that is not included in the extraction. If you enter a negative number (for example, –1), the extractor returns all bytes from the byte specified in Start to the last byte in the record or composite. |
| **Nth field** | Counting from zero (0), indicates the field that you want to extract from the record. | |
| | **Extractor** | |

| Extractor type | Description |
|---|---|
| | Type a value to indicate the position of the field that you want to extract from the record. This value cannot be null and must be an integer greater than or equal to zero (0).<br>**Note:**<br>For example, if you type 1, the second field will be extracted.<br>This option is available only if you specified a field delimiter when configuring the definition and structure of the Flat File Application. This extractor returns the field as a key–value pair. The key is the name of the field. The value is the String value of the field. |

10. Click **Finish** to create the flat file Application.

    After creating the flat file Application, create an Orchestrated Integration, select the Flat File Application created, and invoke the following predefined operations:

    - "convertFlatFileToDocument" on page 355 - Converts the flat file to a document (inbound)
    - "convertDocumentToFlatFile" on page 357 - Converts a document to a flat file (outbound)

## Creating a flat file Application from a sample file

When you create a flat file Application from a sample file, ensure that the format of the sample file is *.txt.

**To create a flat file Application from a sample file**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Flat File Applications > Add New Application**.

2. Type a name and description of the flat file Application and select the creation mode as **Create from a sample file**.

3. Click **Browse** and select the sample file from your computer. The **Preview** pane displays the sample file.

4. Click **Next** and in the **Record Parser** panel, select one of the following to indicate how the data in the sample file is formatted:

| Record Parser | Description |
|---|---|
| **Delimiter** | Use this parser when each record is separated by a delimiter. |

| Record Parser | Description |
|---|---|
| **Fixed length** | Use a fixed length record parser when each record is of a fixed length (for example, mainframe punch or print records). This parser splits a file into records of the same pre-specified length. |
| **Variable length** | This parser expects each record to be preceded by two bytes that indicate the length of the record. Each record may be of different length. |

- If you select **Delimiter** as the parser type, specify the parser properties in the record parser panel. Integration Cloud updates the **Preview** contents based on your selections. Use the following table to specify the delimiters used in the sample file.

| Property | Description |
|---|---|
| **Record** | Character that separates records in a flat file document.<br>**Note:**<br>If a new line character (`\n`) exists at the end of each record and the **Record** delimiter is not ending with `\n`, then Integration Cloud cannot render the records properly under the `Preview` pane. |
| **Field or composite** | Character that separates fields in a flat file document. |
| **Subfield** | Character that separates subfields in a flat file document. |
| **Quoted release character** | Character used to enable a section of text within a field to be represented as its literal value. Any delimiter characters that appear within this section will not be treated as delimiters.<br>**Note:**<br>For example, your field delimiter is (,) and your quoted release character is ". When you want to use (,) within a field as text, you must prefix it with your quoted release character. When using the convertFlatFileToDocument operation to create the strings `Doe, John` and `Doe, Jane`, the record would appear as "`Doe, John`","`Doe, Jane`". When using the convertDocumentToFlatFile operation to create "`Doe, John`","`Doe, Jane`", the value of the record would be `Doe, John and Doe, Jane`. When using the convertDocumentToFlatFile operation, if you have specified both the Release Character and the Quoted Release Character, the Quoted Release Character will be used. |

| Property | Description |
|---|---|
| **Release character** | Character used to enable a delimiter to be used for its intended, original meaning. The character following the release character will not be treated as a delimiter.<br>**Note:**<br>For example, your field delimiter is + and your release character is \. When using + within a field as text, you must prefix it with your release character. When using the convertFlatFileToDocument operation to create the strings a+b+c and d+e+f, the record would appear as a\+b\+c+d\+e\+f. When using the convertDocumentToFlatFile operation to create a\+b\+c+d\+e\+f, the value of the record would be a+b+c and d+e+f. |

- If you select **Fixed length** as the parser type, specify the parser properties in the record parser panel. Integration Cloud updates the **Preview** contents based on your selections. Specify the position of each field in `Field separators`. The `Field separators` must be comma separated integer values, for example, 5, 10, 15. You can set a value for `Record length` to adjust the record so that it appears correctly under the `Preview` pane. `Record length` supports only positive integer values between one and total number of characters in the file. and cannot be empty.

- If you selected **Variable length** as the parser type, specify the parser properties in the Record Parser panel. Integration Cloud updates the `Preview` contents based on your selections. Use the following table to specify the delimiters used in the sample file.

| Property | Description |
|---|---|
| **Field** | Character that separates fields in a flat file document. |
| **Subfield** | Character that separates subfields in a flat file document. |
| **Quoted release character** | Character used to enable a section of text within a field to be represented as its literal value. Any delimiter characters that appear within this section will not be treated as delimiters.<br>For example, your field delimiter is (,) and your quoted release character is “. When you want to use (,) within a field as text, you must prefix it with your quoted release character. When using the convertToValues service to create the strings `Doe, John` and `Doe, Jane`, the record would appear as “`Doe, John`”,“`Doe, Jane`”. When using the convertToString service to create “`Doe, John`”,“`Doe, Jane`”, the value of the record would be `Doe, John and Doe, Jane`. When using the convertToString service, if you have specified both the Release Character and the Quoted Release Character, the Quoted Release Character will be used. |

| Property | Description |
|---|---|
| **Release character** | Character used to enable a delimiter to be used for its intended, original meaning. The character following the release character will not be treated as a delimiter. For example, your field delimiter is + and your release character is \. When using + within a field as text, you must prefix it with your release character. When using the convertToValues service to create the strings a+b+c and d+e+f, the record would appear as a\+b\+c+d\+e\+f. When using the convertToString service to create a\+b\+c+d\+e\+f, the value of the record would be a+b+c and d+e+f. |

5. Specifying a Record Identifier

   When parsing a file, Integration Cloud looks at a record and extracts an identifier out of the data and uses that identifier to connect the record definition with a particular record in the flat file. The name of the record definition must match the value obtained by the record identifier.

   To set the record identifier for a definition, if the flat file contains a record identifier, select **Yes** in the **Record identifier** area, and then set the record identifier to one of the following values.

| Record identifier | Value | Description |
|---|---|---|
| | **Start at position** | Identifies the character position in the record (counting from zero) where the record identifier is located. **Start at position** record identifiers compare the value that occurs in the record, at the specified offset, to all the record names defined in the flat file structure. Note that the **Start at position** identifier cannot distinguish between all types of record names.<br>**Note:**<br>For example, if you name records “Rec1” and “Rec,” some instances of “Rec1” may be identified as “Rec,” because “Rec1” begins with “Rec.” |
| | **Nth field** | Identifies the field in the record (counting from zero) that contains the identifier. **Nth field** identifiers use the value of the specified field as the record identifier. These identifiers count from zero (0).<br>**Note:** |

| Record identifier | Value | Description |
|---|---|---|
| | | For example, if 2 is specified, the third field is used as the record identifier. |

Select **No** if the flat file does not contain a record identifier.

6. Click **Next** to view the **Flat File Structure**. You can add records to the flat file structure and define the hierarchical relationships between them. The application receiving the flat file uses the structure to read the flat file. This structural information identifies the parent-child relationships between different records in the flat file. By nesting record elements in the flat file structure (adding record elements to a record), you can represent the hierarchical structure of the data in the flat file. To add a record, click the **Add new element** icon and select **Record Definition** from the **Element type** field. Specify a name for the record definition and click **Finish**.

7. To add a **Composite Definition** after you have added a record definition, click the **Add new element** icon and select **Composite Definition** from the **Element type** field. Specify a name for the record definition and click **Finish**.

| Extractor type | Description |
|---|---|
| **Nth field-Extractor** | Field number in the record that contains the composite you want to extract. This pulls the subfield data from the composite. If you leave this property empty, the composite will not be extracted. |

To add a **Field Definition** after you have added a record definition, click the **Add new element** icon and select **Field Definition** in the **Element type** field.

<table>
<tr><th>Extractor Type</th><th>Description</th></tr>
<tr><td><b>Fixed Position</b></td><td>Counting from zero (0), indicates a fixed number of bytes to be extracted from a record.
<table>
<tr><th>Extractor</th><th>Description</th></tr>
<tr><td><b>Position</b></td><td>Type the first byte to extract from the record. Type the first byte that is not included in the extraction. If you enter a negative number (for example, –1), the extractor returns all bytes from the byte specified in Start to the last byte in the record or composite.</td></tr>
</table>
</td></tr>
</table>

| Extractor Type | Description | |
|---|---|---|
| **Nth Field** | Counting from zero (0), indicates the field that you want to extract from the record. | |
| | **Extractor** | Type a value to indicate the position of the field that you want to extract from the record. This value cannot be null and must be an integer greater than or equal to zero (0).<br>**Note:**<br>For example, if you type 1, the second field will be extracted.<br>This option is available only if you specified a field delimiter when configuring the definition and structure of the flat file Application. This extractor returns the field as a key–value pair. The key is the name of the field. The value is the String value of the field. |

8. Click **Finish** to create the flat file Application.

   After creating the flat file Application, create an Orchestrated Integration, select the flat file Application created, and invoke the following predefined operations:

   - "convertFlatFileToDocument" on page 355 - Converts the flat file to a document (inbound)
   - "convertDocumentToFlatFile" on page 357 - Converts a document to a flat file (outbound)

## Flat File Predefined Operations

The following predefined Flat File operations are available:

### convertFlatFileToDocument

Converts a flattened flat file data into a structured data, which will conform to the document type associated with the flat file Application.

### Input Variables

*ffDataString* **String** The flat file input with type of String.

*ffData* **Object** The flat file input with type of InputStream or ByteArray. If both *ffData* and *ffDataString* are parsed, *ffDataString* takes precedence.

*ffIterator* **Object** Optional. An object that encapsulates and keeps track of the input data during processing. It is used only when the *iterate* variable has been set to true.

*encoding* **String** Optional. The encoding of the InputStream passed in to *ffData*. The default encoding is UTF–8.

*delimiters* **Document** Optional. A document object that contains the segment terminator and the field and subfield separators. If the delimiter is `null`, it will be located using the information defined in the definition and structure of the Flat File Application. To specify a delimiter, you can specify:

- One character or character representation (for example, *, `\n` for line terminator, `\t` for tab)
- The space character

| Variable | Description |
|---|---|
| *record* | **String** Character used to separate records. If you want to specify the two–character carriage return line feed (CRLF) characters, specify `\r\n`. |
| *field* | **String** Character used to separate fields. |
| *subfield* | **String** Character used to separate subfields. |
| *release* | **String** Character used to ignore a *record*, *field*, or *subfield* delimiter in a field. If a release character occurs in a field or subfield before the delimiter, it will be prefixed with the *release*. |
| *quotedRelease* | **String** Character to use to ignore a *record*, *field*, or *subfield* delimiter in a field. If a quoted release character occurs in a field or subfield before the delimiter, it will be prefixed with *quotedRelease* before being written to the output *string*. The string is pre- and appended with the quoted release character.<br>For example, if * is a delimiter, the field value is a*b, and the quoted release character is “, the string appears as “a*b”. |

*iterate* **String** Optional. Whether you want to process the input all at one time.

*createIfNull* **String** Optional. Whether to create the document object if all the fields are null.

*skipWhiteSpace* **String** Optional. Whether white space at the beginning of records will be ignored.

*keepResults* **String** Optional. Whether to return the parsed data.

*validate* **String** Optional. Whether to return error messages that describe how *ffData* differs from the definition and structure of the Flat File Application.

*returnErrors* **String** Optional. Whether to return the validation errors.

*maxErrors* **String** Optional. The maximum number of errors that can be returned from one record. When the flat file parser encounters more than the maximum number of

| | |
|---|---|
| | errors within a record, the parser will stop parsing and return the parsed data and errors processed up until that point. |
| *flags* | **String** Optional. Flags that you can set to govern convertFlatFileToDocument options. |

| Variable | Description |
|---|---|
| *addRecordCount* | **String** Whether you want the operation to add an additional field (*@record–count*) to each parsed record in the resulting document object. |
| *detailedErrors* | **String** Whether you want detailed conditional validation error information. |
| *skipToFirstRecord* | **String** Whether you want the operation to wait until it finds the first valid record before reporting invalid records as errors. |
| *trimWhitespace* | **String** Whether you want the operation to delete any blank spaces at the beginning of fields, at the end of fields, or both. |
| *resultAsArray* | **String** Whether you want the operation to return the document object that represents the input flat file data as a document reference that can be mapped to the document types generated. |

### Output Variables

| | |
|---|---|
| *<flatfile_application_name>_dt* | **Document** The name of the output document will be *flatfile_application_name_dt*, where *flatfile_application_name* is the name of the flat file Application. For example, if the Application name is *test*, then the name of the output document will be *test_dt*. The structure of the output document will be similar to what you have defined in your Flat File Application. |
| *isValid* | **String** Whether flat file contains validation errors. |
| *errors* | **String** Optional. An array containing the validation errors, if any, that were found in *ffData* or *ffDataString*. |

### convertDocumentToFlatFile

Converts a structured data conforming to the given document type structure associated with the Application, to the flattened flat file data.

### Input Variables

| | |
|---|---|
| *<flatfile_application_name>_dt* | **Document** Structure conforming to the Document type created for the corresponding application of this operation. |

*spacePad*

**String** Optional. How to position the records in the flat file.

*signalError*

**String** Whether to create errors in the output.

*noEmptyTrailing Fields*

**String** Whether trailing empty fields are to be removed from the output. Used only with records that have delimited fields.

*noEmptyTrailing SubFields*

**String** Whether trailing empty subfields are to be removed from the output. Used only with records that have delimited fields.

If no value is specified for the *noEmptyTrailingSubFields* parameter, Integration Cloud uses the value set for the *noEmptyTrailingFields* parameter.

*delimiters*

**Document** Optional. The separator characters used to construct the output string. To specify a delimiter, you can specify:

- One character or character representation (for example, *, \n for line terminator, \t for tab)

| Value | Description |
|---|---|
| *record* | **String** Character to use to separate records. If you want to specify the two–character carriage return line feed (CRLF) characters, specify `\r\n`. |
| *field* | **String** Character to use to separate fields. |
| *subfield* | **String** Character to use to separate subfields. |
| *release* | **String** Character to use to ignore a *record*, *field*, or *subfield* delimiter in a field. If a release character occurs in a field or subfield before the delimiter, it will be prefixed with *release* before being written to the output *string*. |
| *quotedRelease* | **String** Character to use to ignore a *record*, *field*, or *subfield* delimiter in a field. If a quoted release character occurs in a field or subfield before the delimiter, it will be prefixed with *quotedRelease* before being written to the output *string*. The string is pre- and appended with the quoted release character. |

For example, if * is a delimiter, the field value is a*b, and the quoted release character is “, the string appears as “a*b”.

*FormatInfo* **Document** Any values mapped to the *FormatInfo* variable will be passed unmodified to all format operations invoked by convertDocumentToFlatFile and convertFlatFileToDocument.

*outputFileName* **String** Optional. If you want the output returned in a file instead of in the *string* output variable, provide the name of the file you want created as a result of this operation.

*Encoding* **String** The type of encoding used to write data to the output file. The default encoding is UTF–8.

*sortInput* **String** Optional. Whether you want the operation to sort the input records to match the definition and structure of the Flat File Application.

*returnAsBytes* Returns the document as a string or as a byte array instead of a string.

## Output Variables

*string* **String** Data that represents the flat file document.

*bytes* **Object** If the input variable *returnAsBytes* is `true`, returns the output as a byte array encoded using the specified encoding. The string value is not returned.

*errorArray* **Object** String array containing messages pertaining to errors that occurred during conversion.

## Usage Note

When the convertDocumentToFlatFile operation executes, the field that is defined to start after the end of the fixed length record will not be included in the output data if the following conditions are met:

- The definition and structure of the Flat File Application uses a fixed length record delimiter.
- The definition and structure of the Flat File Application contains a fixed position field that begins beyond the defined length of the fixed length record.
- The input to the convertDocumentToFlatFile operation contains a value for the fixed position field that begins beyond the defined length of the fixed length record.

# 6 Accounts, Operations, Listeners

# Accounts

This screen lists all the available Accounts created for an Application.

If you select an Account for an FTP, SFTP, custom SOAP, or on-premises Application and click **Test Connection**, the screen displays the status of the connection. If you have configured the Account details incorrectly in any stage, the status appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the status appears in green color and if an Account is not configured in a particular stage, the status appears in white color.

For on-premise Applications, the Account can be used to execute services on the on-premises webMethods Integration Server. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for information on how to configure webMethods Integration Server as an on-premise server for use with Integration Cloud.

**Note:**
Only enabled or active Accounts are listed in the drop down list of the Operation wizard, Integration wizard, Look up Transformer, and Manage Stages page.

You can create, edit, or delete an Account for a particular application from this screen.

**Note:**
Users who have the required permissions under **Settings** ⚙ > **Project Permissions** can create, update, or delete the Accounts information.

### To create or edit an Account

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > <Application category>**.

2. Select an Application from the Application category page and then click **Accounts**.

   To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

3. From the Accounts screen, click **Add New Account** to add an Account or click **Edit** to update an existing Account.

## Adding or Editing Accounts

Use the **Accounts** page to add, edit, or delete Accounts. The options available may vary according to the selected Application.

**Note:**
See the "Predefined Applications" on page 109 section for information on the Account configuration fields for each Application.

### To add or edit an Account

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications > <Application category>**.

2. Select an Application from the page, and then click **Accounts**.

   To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

3. From the **Accounts** page, click **Add New Account** to add an Account or click **Edit** to change any field in an existing Account.

4. On the **New Account** or **Edit Account** page, complete the following fields. Required fields are marked with an asterisk on the page.

   > **Note:**
   > Based on the Application you had selected, applicable fields are displayed.

| Field | Description |
|---|---|
| **Save As** | Provide a valid name for the Account. This field is common for all Applications. Names can contain alphanumeric characters, underscores (_), and hyphens (-). The name must not be null and cannot be an empty string. |
| **Description** | Provide a description for the Account. This field is common for all Applications. |
| **Authentication Type** | Every back end provides its own authentication mechanism. Here you can select different authentication schemes for the same Application. Get the authentication details from the back end documentation for the Application and select the supported **Authentication Type** from the drop-down list. The Account Configuration Fields may vary according to the selected authentication type.<br>**Note:** The **Authentication Type** field may not be available for all Applications.<br>**Credentials**: Basic authentication credentials.<br>**OAuth V2.0 (Authorization Code Flow)**: Authorization by OAuth v2.0. |

| Field | Description |
|---|---|
| | **OAuth V2.0 (JWT Flow)**: Authorization by a signed JWT.<br>**Note:** While editing an Account in any stage, you can select a different *Authentication Type* without impacting any integrations. So if an Integration is using an Account with a specified *Authentication Type*, the integration will run with the changed Account configuration. Further, while editing or creating an Integration, after you select the Operation, the Account field will list all the Accounts that are supported for the execution of the selected Operation.<br>The **Account Configuration Fields** section allows you to provide details to connect with the Application. The fields available may vary according to the selected Application. See the "Predefined Applications" on page 109 section for information on the Account configuration fields for each predefined Application.<br>**Note:** Click "here" on page 364 for answers to some of the most common questions on Account configuration. |

5. Click **Save** or **Update** to save your settings.

   A new Account will be created.

## Troubleshooting tips on Account configurations

Find answers to some of the most common questions on Account configurations.

### Connection Reset

- **I see connection reset errors. What is the root cause of these errors?**

The root cause of connection reset errors is stale connections. It is recommended to set the **Keep Alive Interval** to two minutes.

### Time Out Exceptions

- **I see a transaction has failed with the following exception: ERROR_INVOKING_CLOUD_SERVICE: An IO or timeout exception is encountered for invoke "xxx" method. What is the root cause and the resolution?**

The root cause may be linked to stale connections or a low response timeout value. To handle stale connections, set the **Keep Alive Interval** to two minutes. To handle receipt of large payloads or a slow network impacting the responsiveness of the back end, increase the **Response Timeout** value, for example, to five minutes or higher.

## Session Expired or Invalid Session ID

- **I am using the Salesforce Credential based account configuration and I see the following invalid_session_id errors while executing operations. How can I prevent an invalid session time out?**

*INVALID_SESSION_ID: Invalid Session ID found "message": "Session expired or invalid", "errorCode": INVALID_SESSION_ID"*

*Salesforce operation execution fails sometime with Invalid Session ID found error*

*Invalid Session ID found in SessionHeader: Illegal Session. Session not found. This error usually occurs after a session expires or a user logs out. Decoder: DataInDbSessionKeyDecoder*

These errors are observed if the client (Integration Cloud Account) and the server (for example, Salesforce) session time out values are not in sync. Based on the Salesforce back end session timeout value, set a **Session Timeout (min)** value in the Integration Cloud Account configuration screen. This value should be less than the Salesforce back end session timeout value so that the token is refreshed before Salesforce invalidates it. For corner case scenarios, specify a value that is one minute less than the Salesforce back end session timeout value. For example, if on the server side (Salesforce), you have configured the session timeout value to 15 minutes, then on the client side (Integration Cloud Account), ensure that the session time out value is less than the server (Salesforce) session time out value, that is 14 minutes.

## Disabled Connection Issue

- **Sometimes integrations start failing and I see the following cloud connection disabled errors. Why do they appear and what is the resolution?**

*DISABLED_CLOUD_CONNECTION: Connection xxx is disabled*

*INVALID_LOGIN: Invalid username, user not active*

Connection disabled errors may appear due to the following reasons:

- Salesforce back end account password has been changed or has expired.

  In this case, check if the password is still valid.

- Salesforce login limit is breached.

If 3600 login calls are sent to Salesforce in 15 minutes, then depending on the configured **Lockout effective period** in Salesforce, the integration fails, and the *DISABLED_CLOUD_CONNECTION: Connection xxx* message appears. For example, If 3600 login calls are sent to Salesforce in 15 mins, and you have set the **Lockout effective period** to 60 minutes, then from the 16th minute till the 60th minute, all integrations will fail.

Usually credentials based connections get disabled once the Salesforce login limit is breached. You may have reached the login limit quota for the back end.

*How to minimize your login calls to Salesforce*

Salesforce allows 3600 logins per hour. In Integration Cloud, select **Enable Connection Pooling** to enable the connection pooling option. Also adjust the **Minimum Pool Size**, **Maximum Pool Size**, and the **Expire Timeout (msec)** values. Expire Timeout will ensure that a connection in the Integration Cloud Account pool is kept alive for the configured time interval after the Account is created. In the absence of the Expire Timeout configuration, the connection will be invalidated immediately to maintain the Minimum Pool Size as one. The intention is to limit the number of login requests, so the recommendation is to keep the Expire Timeout value equal to the Session Timeout value, because the connection is anyways invalidated after Session Timeout. For example, if the session timeout in Salesforce back end is configured to two hours, specify the Expire Timeout and the Session Timeout values in the Integration Cloud Account Configuration screen to be slightly less than two hours.

| Account Configuration Fields | Value |
|---|---|
| **Session Timeout** | Slightly less than the Salesforce back end session timeout settings. For example, 119 minutes, if the Salesforce back end session timeout value is 120 minutes. |
| **Keep Alive Interval** | Two minutes. |
| **Response Timeout** | Five minutes. |

*Do the following if your Salesforce login limit is breached*

If your Salesforce logins have exceeded 3600 logins/hour, then your Salesforce account gets disabled. This is usually observed in the case of credentials-based authentication connection with the back end. In this scenario, login to your Salesforce back end account, go to **Setup > Manage User > Users**, select the User profile, go to the **Password Policies** section, and check the time configured for the **Lockout effective period** field. The Lockout effective period is the time for which the Salesforce back end account will be locked once the login limit is breached. You can either wait till the time specified in the Lockout effective period field has passed or you can configure a different lockout value as per your need.

- **Why do I see the UNABLE_TO_RETRIEVE_CONNECTION_FROM_POOL error while enabling Accounts or executing Operations?**

Increase the **Maximum Pool Size** or the **Block Timeout** values. Regarding the maximum pool size, when the connection pool has reached its maximum number of connections, the connector will reuse any inactive connections in the pool, or if all connections are active, it will wait for a connection to become available. Increasing the Block Timeout value increases the time Integration Cloud will wait to obtain a connection with the SaaS provider before the connection times out and returns an error. For example, if you have a pool with Maximum Pool Size of 20 and if you receive 30 simultaneous requests for a connection, 10 requests will be waiting for a connection from the pool. Now if you set the Block Timeout to 5000 msec, the 10 requests will wait for a connection for 5000 msec or 5 seconds before they time out and return an error. If the services using the connections require 10 seconds to complete and return connections to the pool, the pending requests will fail and return an error message stating that no connections are available.

- **I see integrations failing with the DISABLED_CLOUD_CONNECTION: Connection xxx error but the Account shows enabled in the User Interface.**

Do the following to resolve this issue:

1. Go to the stage where the Account is found disabled.
2. On the Projects page, select the Default project.
3. Edit any asset in the same stage. For example, click the Reference Data tab, select a reference data, update the reference data description, and then save the reference data.
4. Wait for a minute. This will trigger an update and will resolve the disabled Account issue. The integration will now run successfully.

## Operations

Integration Cloud provides pre-configured applications. The Applications contain SaaS provider-specific information that enables you to connect to a particular SaaS provider. Further, each Application uses an Account to connect to the provider's back end and perform Operations.

**Note:**
Users who have the required permissions under **Settings** ⚙ **> Project Permissions** can create, update, or delete Operations.

Each application comes with a predefined set of Operations. You can also create your own custom Operations and also edit/delete those custom Operations. This page lists all the available Operations for a selected application including predefined Operations.

See "FTP Predefined Operations" on page 195 for information on the FTP operations.

See "SFTP Predefined Operations" on page 254 for information on the SFTP operations.

See "AS2 Predefined Operations" on page 113 for information on the Applicability Statement 2 (AS2) operations.

See "Database Application Operations" on page 151 for information on the Database operations.

See "Cloud Deployment Operations" on page 136 for information on how to import Cloud Deployment services.

### To create or edit a custom Operation

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications**.
2. Select an Application category and then click **Operations**.

To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click I **do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

3. From the Operations page for the selected Application, click **Add New Operation** to create a new Operation or select an Operation and click **Edit** to update an existing Operation. You can click **Delete** to delete an existing Operation, click **Show Signature** to view the input and output signature of the Operation, or click **Test** to test the Operation. The **Add New Operation** option is not available for some Applications. While testing an Operation, you can select another Account created with a different *Authentication Type*, if the Account is supported for the execution of the selected Operation.

   Select an Operation and click **Show Signature** to view the input and output signature of the operation. The input and output fields cannot be edited. This option is available for all predefined and custom operations. Click the input and output fields to view the field properties.

   From the **Input** or **Output** pane, click the ⧉ icon to copy a field. Depending on the context, you can either paste the field or the field path.

   Click the **Test** option and in the test dialog box, specify the **Account** name and the **Input** data. If an operation does not have an input signature, the input fields are not displayed. The **Test** option is available for all predefined and custom operations.

   Click **Run** to test the Operation and view the test results in the test results window. Click the

   ← icon beside **Result** if you want to go back to the test dialog box and enter another set of values. The last 5 test results are also displayed and are applicable only for the same test operation run, that is, if you close the test results window, you will not be able to view the test results later. Further, a test result appears in red color if the test run is unsuccessful and appears in green color for a successful test run.

## Adding or Editing Operations

Use the **Operations** page to add, edit, or delete custom operations.

### To add or edit a custom operation

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Applications**.

2. Select an Application from an Application category and then select **Operations**.

   To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click I **do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

3. From the Operations page for the selected Application, click **Add New Operation** to add a custom operation or click **Edit** to update an existing custom operation. The **Add New Operation** option is not available for some Applications.

4. On the **Select your <...> account** page, complete the following fields. Required fields are marked with an asterisk on the page.

| Field | Description |
|---|---|
| **Save as** | Provide a name for the custom operation. |
| **Description** | Provide a description for the custom operation. |
| **Select account** | Select an Account created for the Application from the drop-down list. Only active or enabled Accounts are listed in the drop-down list.<br>**Note:**<br>If you select an Account with a different Authentication Type, existing Integrations that are using this Account may not work. |
| **Select functional area** | Select the Application service from the drop-down list.<br>**Note:**<br>This option is available only for certain Applications. |
| **Enable dynamic account configuration at runtime** | For REST Application operations, you can now provide account configurations at *runtime* during integration execution. Select the **Enable dynamic account configuration at runtime** check box and then click **Configure Fields** to run the operation by passing authentication details that may be *different* from the authentication details configured in the Account configuration page. Dynamic authentication overwrites the Account configuration details.<br>For example, suppose you have configured authentication details X for Account A. You can use different authentication details Y to run the service, by overwriting the authentication details X of Account A.<br>**Note:**<br>**Enable dynamic account configuration at runtime** is currently supported only for custom REST Applications. Further, the option is enabled only after you have selected an Account in the **Select account** field.<br>While creating an operation, if you select the **Enable dynamic account configuration at runtime** check box and click **Configure Fields**, the **Dynamic Account Configuration Fields** window appears. Select the fields you want to include in the operation input signature. When you select the fields, the operation input signature will be automatically updated based on your selections. Only the fields that are part of the |

| Field | Description |
|---|---|
| | signature gets mapped as the input request. You can override the field values at run time. Default Account configuration values will be used for other fields while creating the operation.<br>**Note:**<br>■ You cannot clear encrypted fields like Access Token and Consumer Secret. Further, you cannot enter a value for encrypted fields. You can enter the values for encrypted fields only while running the operation.<br>■ While running the operation, if you have provided details in both the Account configuration page and in the **Dynamic Account Configuration Fields** dialog box, Integration Cloud reads the data from the Account configuration page and merges the data with the data provided in the **Dynamic Account Configuration Fields** dialog box. Existing Account configuration details will not be modified.<br>■ Details specified in the **Dynamic Account Configuration Fields** dialog box take precedence over the details specified in the Account configuration page.<br>■ If you pass incorrect values in the **Dynamic Account Configuration Fields** dialog box, service execution will fail while running the operation. |

5. Click **Next**.

   The **Select the Operation** page appears. If you do not want to add **Headers** and **Parameters**, skip to Step 9.

6. For a REST-based Application, select an operation and click **Headers** to add input Headers, if required. Integration Cloud displays the default HTTP transport headers for the operation, along with their default values. At run time, while processing the headers, Integration Cloud substitutes the values as necessary. In order to customize the headers, do the following:

   a. Click **Add Header** to add a custom Header. Specify the **Header** name. To specify an optional default value for the header variable, click the **Default Value** box and type or paste the new value. If the variable is null in the input pipeline, this value will be used at run time. If the variable already has an existing default value defined, this value will overwrite the existing value at run time.

   b. If headers appear in the signature, select **Active** to activate the headers in the signature.

   c. To delete a custom header that you have added, select the header and click **Delete**.

   **Note:**

You cannot delete the required headers.

7. For a REST-based Application, select an operation and click **Parameters**. To customize the parameters, do the following:

   a. Review the operation parameter details. Integration Cloud displays the parameter **Name** and **Description**, the **Data Type** used to represent the kind of information the parameter can hold, the parameterization **Type** of the request, and the default value needed to access the operation.

   b. To specify a default value for the parameter, click the **Default Value** box and then type or paste the default value. The default value is used at run time.

   You cannot add or delete request parameters. You can only modify the default value of a parameter. All parameters appear in the signature.

8. For some Applications and Operations, for example, for the *CloudStreams Connector for Salesforce(R) Bulk v2 Data Loader* Application and *Create Job and Upload Job Data* Operation, Integration Cloud supports multipart request body. For example, if you want to create a user as well as upload a photo, the request has to be a multipart request where one part is an image file while the other part is a JSON request body.

   Though application/x-www-form-urlencoded is a more natural way of encoding, it becomes inefficient for encoding binary data or text containing non-ASCII characters. The media type *multipart/form-data* is the preferred media type for request payloads that contain files, non-ASCII, and binary data. Integration Cloud supports this media type using which you can embed binary data such as files into the request body.

   A multipart/form-data request body contains a series of parts separated by a boundary delimiter, constructed using Carriage Return Line Feed (CRLF), "--", and also the value of the boundary parameters. The boundary delimiter must not appear inside any of the encapsulated parts.

   Each part has the Type, Name, Content-Type, and Part ID fields.

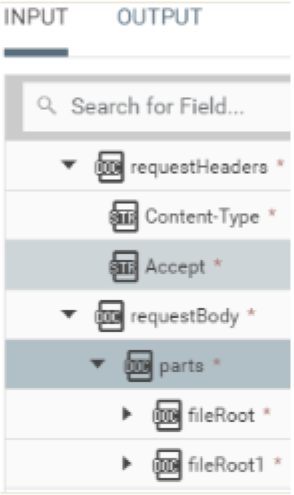
   **Example of a multipart request**

   ```
   --BOUNDARY
   Content-Type: application/json
   Content-Disposition: form-data; name="job"
   {
     "object":"Contact",
     "contentType":"CSV",
     "operation":"insert"
   }
   --BOUNDARY
   Content-Type: text/csv
   Content-Disposition: form-data; name="content"; filename="content"
   (Content of your CSV file)
   --BOUNDARY—
   ```

   **Part Configuration**

The **Part Configuration** window lists all the configured parts to be sent to the service provider. To view the **Part Configuration** window, on the **Select the Operation** page, select **Attachments**.

The configured parts appear in the input signature.



- **Name** - The **Name** field displays the name of the file part as documented in the SaaS provider API documentation.
- **Content Type** - The **Content Type** field displays the content type of the file you are uploading.
- **Type** - **Document** - Some back ends expect application/json or application/xml content in the part body. This represents a part where the content of the part is of type application/json or application/xml. **File** - Represents a binary part of a multipart/form-data payload where the content of the part is binary or the content of the file itself. For file upload kind of use cases, this part is the main or mandatory part required by the service provider.

9. On the **Select the Operation** page, select the operation to be performed and then click **Next**. Depending on whether the operation is Complex, Simple, or Dynamic, the **Business Object** or the **Interactions** page appears. Examples of **Business Objects** are Contact, Account, and so on and examples of **Interactions** are Create, Update, Upsert, Delete, and so on.

**Note:**
For some operations, Integration Cloud displays appropriate Business Object pages, for example, for the createMultiple and updateMultiple operations in the Salesforce v42 Application, or Interactions (sub-operations) pages, for example, for the Batch and ChangeSet operations in the OData 4.0 Application, depending on whether the selected operation requires metadata, such as a business object, fields, and data types of fields. You can add or edit multiple business objects in a single request. You can also add or edit interactions and then associate those interactions with business objects in a single request. Interactions or multiple business objects will be executed in the same sequence as they appear in the table. You can drag and drop the interactions or multiple business objects to change the order in which they will be executed.

Different pages appear based on the scenarios mentioned in the following *Single or Multiple Interactions with Single or Multiple Business Objects with dependencies* section.

**Single or Multiple Interactions with Single or Multiple Business Objects with dependencies**

- **Single operation has a single Business Object** - An operation has only a single business object. The operation has neither multiple interactions nor has records. An example of a single operation and a single object can be a "create" operation that contains only the "contact" business object.
- **Single operation has multiple Business Objects** - A single operation has multiple business objects. An example of a single operation with multiple business objects can be a "create" operation that contains two business objects, "contact" and "account".
- **Single operation has multiple Business Objects with dependencies** - A single operation has multiple business objects and some of the business objects may have dependencies on other business objects.
- **Multiple Interactions have multiple Business Objects** - Multiple Interactions have multiple business objects. For example, the "create" and "update" Interactions can act on the "account" and "contact" business objects respectively.
- **Multiple Interactions have multiple Business Objects with dependencies** - Multiple Interactions have multiple business objects and some of the business objects may have dependencies on other business objects. For example, the "create" and "update" Interactions can act on the "account" and "contact" business objects respectively.

Further, for some operations, for example, for the *Retrieve Contained Or Derived Entity* operation in the OData 4.0 Application, Integration Cloud displays nested, hierarchical, or multi-level business objects if the operation is designed to support nested business objects. You can expand the nested business objects to display the child-level objects.

10. Select the Business Object to be associated with the operation you have selected in the previous step and then click **Next**. For some operations, you must select the Interaction and then the Business Object. Business Objects and Interactions appear only for certain Applications and operations.

11. In the **Data Fields** page, select the data fields for the Business Object you have chosen in the previous step and then click **Next**. Data fields appear only for certain Applications and operations. Mandatory data fields for the Business Object are selected by default and cannot be cleared.

**Note:**
For some operations of certain Applications, for example, Coupa, you can add your own fields. Such fields are called custom fields. Custom fields are marked by *custom* on the page. You can add, edit, or delete only custom fields. Click the ✚ icon to add a custom field. The ✚ icon appears only if you are allowed to add custom fields for the selected operation. After you have added a custom field, you can click on the custom field to edit the field properties.

Simple fields appear with a check box while complex fields appear with a check box followed by an arrow mark. The following states are observed for complex fields:

- **Unchecked** - For unchecked complex fields, only the mandatory child fields are selected but those fields will not be added to the signature, unless you select the parent field.
- **Checked** - If a complex field is selected by default, then only the mandatory child fields are selected. You can select the optional fields, if required. If you select a complex field, then all the child fields are selected.

**Note:**
For some Applications, for example, Microsoft Dynamics CRM, you can choose the way a query can be executed in the **Confirm operation** page. Choose the operation and then click **Finish**.

12. In the **Confirm Operation** page, verify the details. You can click the **Data Fields** link to view the data fields you have added. For some operations, you can click the Business Object link to view the data fields added.

13. Click **Finish** to create the custom operation.

    After you click **Finish** or **Save**, if there are any Business Parameters, you will be asked to configure the Business Parameters.

# Listeners

Some Integration Cloud Applications now support connectivity with streaming APIs and processing of streaming API events. You can use Integration Cloud Accounts to connect to streaming APIs. While creating an Integration Cloud listener, you select a subscription channel from a list of available channels for an endpoint and select the Integration that will be invoked when the events are received. Additionally, you can configure transport or communication related parameters for an Integration Cloud listener as well as enable and disable the listener. A listener receives the streaming API events and processes the received events.

**Note:**
Currently, only some Integration Cloud Applications support streaming API capability, for example, the Salesforce CRM version 44 Application. Salesforce trial accounts have restrictions on the maximum number of concurrent subscribers. See PushTopic Streaming Allocations for information. So if you create multiple listeners using the same trial account details, due to the high availability nature of Integration Cloud, some active listeners may not work properly.

## Creating and updating listeners

### To add a listener

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project > Applications > Predefined Applications**.

   **Note:**

You must have the required project permissions under **Settings** ⚙ **> Project Permissions** to create, update, and delete listeners. Further, to create a Listener and also invoke the associated Integration when events are received, you must have the Listeners **Create** permission and the Integration **Execute** permission.

2. Select a predefined Application and click **Listeners**.

   To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

   The **Listeners** page appears for the selected Application.

   **Note:**
   **Listeners** option appears only for Applications that have streaming API capability.

3. On the **Listeners** page, click **Add New Listener** to add a listener or click **Edit** to update a listener.

4. On the **Configure the Listener** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Field | Description |
|---|---|
| **Save As** | Provide a valid name for the listener. Names can contain alphanumeric characters, underscores (_), and hyphens (-). The name must not be null and cannot be an empty string. |
| **Description** | Provide a description for the listener. |
| **Streaming API Endpoint URL** | Specify the endpoint of the streaming provider. When configuring a Salesforce streaming endpoint, specify the URL in the following format: https://<*Salesforce_Instance*>/cometd/<*Salesforce API version*><br>Example: https://ap5.salesforce.com/cometd/44.0 |
| **Select replay option** | The Integration Cloud listener for Salesforce can subscribe and listen to Salesforce events. This allows the Integration Cloud listener to establish a persistent connection with Salesforce. This connection remains open while transmitting the events until either side closes the connection. Through this established connection, events are streamed to the listener. If the connection is lost due to a network failure or any other reason, you can retrieve the standard-volume events that are within Salesforce's 24-hour retention window. |

**Field** **Description**

You can retrieve Salesforce events only if you are using Salesforce v37.0 or later versions.

Each Salesforce event is assigned an opaque ID which is unique for each event. This ID is contained in the replayId field of the event object. The replayId field value which is populated by Salesforce when the event is delivered to subscribers, refers to the position of the event in the event stream. For consecutive events, replayId values may have a break between them, for example, the event with ID 110 may follow the event with ID 101.

*Example of an event object that Salesforce sends to its subscribed clients*

```
{

"clientId":"a1ps4wpe52qytvcvbsko09tapc",
          "data":{
             "event":{

"createdDate":"2016-03-29T19:05:28.334Z",
                "replayId":55
                },
             "payload":"This is a message."
             },
         "channel":"/u/TestStreaming"
         }
```

To replay events, Salesforce provides a way to configure the replayId while subscribing to a particular channel. You can replay the events by specifying the replay option and can use it on a resubscription, to retrieve events that are within the retention window.

- **New** - Receive new events that are broadcast after the client subscribes and replay only the new events from the time the listener is enabled.
- **All** - Receive all events including past events that are within the retention window including new events. Replay all the events for the last 24 hours.

**Select account**

Select an account to connect to the Application. The streaming functionality will leverage the existing authentication, timeouts, truststore, keystore, and host name verification configurations from the referenced Account selected in the **Select account** drop-down list box.

| Field | Description |
|---|---|
| **Select subscription** | Select a channel from the list of available subscription channels, that is, select the subscriber you want the listener to connect. |

5. Click **Next**.

   The **Parameters and Headers** page appears.

6. In the **Parameters** section, Integration Cloud displays the pre-configured parameters for the selected subscription.

   a. Review the details about the parameters. Integration Cloud displays the parameter name and description, the data type used to represent the kind of information the parameter can hold, and the parameterization type for the subscription request.

   b. To specify the default value for the parameter, select the pre-configured parameter and click **Edit**. In the **Parameter Properties** dialog box, type the default value. For example, for Salesforce, type the Topic Name. The **Active** option includes the parameter as part of the subscription request.

7. In the **Headers** section, Integration Cloud displays the pre-configured HTTP transport headers for the selected subscription.

   a. To specify a value for a pre-configured header variable, click **Edit** and then in the **Header Properties** dialog box, type the new default value.

   b. To add a header, click **Add Header**. Type a name for the header, select the **Active** check box to include the header as part of the subscription request, and provide a default value.

   c. To delete a header that you have added, select the header and click **Delete**.

   **Note:**
   You cannot delete required headers.

8. Click **Next**.

   The **Event** page appears.

9. On the **Event** page, specify the event action configuration for the listener subscriber. You can invoke an Integration based on your configuration.

   a. In the **Integration Name** field, select the Integration that will be invoked when the events are received. In the **Run As User** field, select the Integration Cloud user you want Integration Cloud to use when running the Integration. Integration Cloud runs the Integration as if the user you specify is the authenticated user that invoked the Integration. If the Integration is governed by an Access Control List (ACL), ensure that you specify a user that is allowed to invoke the Integration.

> **Note:**
> You must manually run the Integration before you map that Integration to the listener, else the listener will not execute the mapped Integration.

10. Click **Next**, review the **Summary** details, and then click **Finish** to create the listener.

    The new listener appears on the **Listeners** page.

11. To enable the new listener, move the slider under the **Status** column.

    The listener will be enabled after a short delay. Once the listener is enabled, connection with the streaming API is established, and incoming events will be processed whenever the streaming provider triggers the events. As an example, for Salesforce CRM version 44, say you have created a topic using the following sample query.

```
Sample Query
PushTopic testCvent = new PushTopic();
testCvent.ApiVersion = 42.0;
testCvent.Name = 'TestContact';
testCvent.Description = 'All records for the Contact object';
testCvent.Query = 'SELECT Id, Email, Phone, FirstName, LastName, Title, Salutation

FROM Contact';
insert testCvent;
System.debug('Created new PushTopic: '+ testCvent.Id);
```

After that you have subscribed to that topic by creating a listener in Integration Cloud. Now whenever the Contact object is modified in Salesforce, the Integration Cloud listener is notified. To use the streaming data as an input to an Integration, ensure that you create a JSON document type that has the same fields inside sobject as that of the query parameters of the topic.

```
Sample Document
{
            "data":{
                        "event":{
                                    "createdDate":"",
                                    "replayId":"",
                                    "type":""
                    },
                        "sobject":{
                                    "Email":"",
                                    "Phone":"",
                                    "FirstName":"",
                                    "Title":"",
                                    "LastName":"",
                                    "Id":"",
                                    "Salutation":"",
                    }
            },
            "channel":"/topic/TestContact2"
}
```

**Note:**
If a listener L1 is using an Integration IN1 and if IN1 uses an Operation OP1 which triggers L1 through a Salesforce Topic, then whenever OP1 is run, L1 will be triggered. L1 will again execute IN1 and IN1 will again execute OP1, thereby creating Integration execution loops. Therefore ensure that loops are not created in an Integration, which is mapped to a listener.

# 7 Integrations

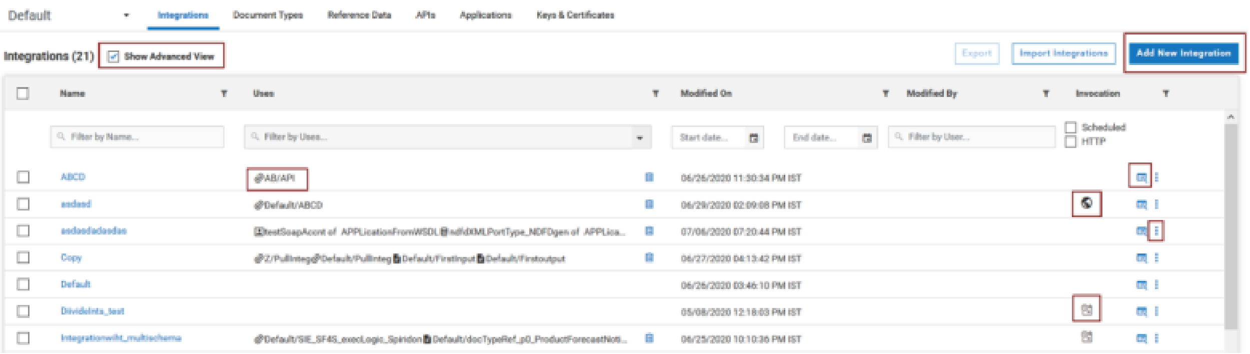
# Overview

An integration is an orchestration of a source and a target Operation with appropriate data mappings and transformations.

> **Note:**
> Users who have the required project permissions under **Settings** ⚙ **> Project Permissions** can create, update, delete, and execute integrations.

### Integrations page

The **Integrations** page lists Point to Point and Orchestrated integrations created for cloud-based SaaS applications with other cloud-based applications and also SaaS applications with on-premises applications.



| □ | Name | Uses | Modified On | Modified By | Invocation |
|---|---|---|---|---|---|
| □ | ABCD | @AB/API | 06/26/2020 11:30:34 PM IST | | |
| □ | asdasd | @Default/ABCD | 06/29/2020 02:09:08 PM IST | | |
| □ | asdasdadasdas | testSoapAccnt of APPLicationFromWSDL ndfdXMLPortType_NDFDgen of APPLica... | 07/06/2020 07:20:44 PM IST | | |
| □ | Copy | @2/PullInteg @Default/PullInteg Default/FirstInput Default/Firstoutput | 06/27/2020 04:13:42 PM IST | | |
| □ | Default | | 06/26/2020 03:46:10 PM IST | | |
| □ | Dividelnts_test | | 05/08/2020 12:18:03 PM IST | | |
| □ | Integrationwiht_multischema | @Default/SIE_SF4S_execLogic_Spindon Default/docTypeRef_p0_ProductForecastNoti... | 06/25/2020 10:10:36 PM IST | | |

The **Name** column in the **Integrations** page displays the name of the integration. You can select an integration and click the integration name link under the **Name** column to modify the integration. The integrations list page by default shows a basic view of all the integrations. Click **Show Advanced View** to view the **Uses** column, which displays the references that are used or utilized to create the integration in the format *project name/referenced asset name*. The **Invocation** column also appears once you click **Show Advanced View**. It shows the invocation channel used (Scheduler, User Interface, HTTP Interface, REST APIs, SOAP APIs, and Listeners) to invoke the integration. Click the 📅 icon to view the scheduled status, scheduled type, and when is the integration execution next scheduled. To view the last five execution results for an integration, select an integration, click the 🔍 icon, and select **Last 5 Execution Results** tab in the integration details page. You can also point to the 🌐 icon to view the request URL and click the ⋮ icon to delete or copy an integration.

## Create, Modify, Delete, and Copy Integrations

### To create an integration:

- On the **Integrations** page, click **Add New Integration**.
- To create a Point to Point integration, select **Synchronize two applications**. To create an Orchestrated integration, select **Orchestrate two or more applications**.

### To modify an integration:

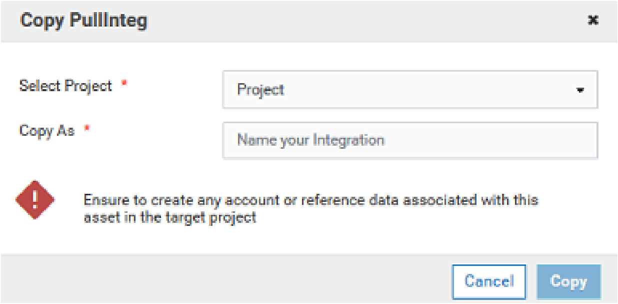
- Select the integration you want to modify.
- Click the integration name link under the **Name** column to modify the integration. The integration opens up for editing in the workspace. You can also edit the integration from the integration details page. To do that, click the ▤ icon, and select **Edit** on the **Overview** page.

**To delete an integration:**

- Select the integration you want to delete.
- Click on the ellipsis ⋮ icon and select **Delete**. The integration is permanently deleted and you cannot recover it. You can also delete the integration from the integration details page. To do that, click the ▤ icon, and select **Delete** on the **Overview** page.

**To copy an integration:**

- Select the integration you want to copy.
- Click on the ellipsis ⋮ icon and select **Copy**. You can make a copy of the integration within the same project or in another project.
- In the **Select Project** field, the currently open project is set as the default option. To copy to another project, select the project from the drop-down list as shown in the following example:



Copy PullInteg

Select Project * Project

Copy As * Name your Integration

Ensure to create any account or reference data associated with this asset in the target project

- Provide a different name in the **Copy As** field
- Click **Copy**. The system creates a copy of this integration with the new name in the integrations page of the target project.

> **Note:**
> You must create the account or reference data associated with this integration in the target project. Further, you can copy a block from an integration and paste that block in another integration across projects.

## Export and Import integrations

To export an integration, select the integration, and then click **Export**. See " Export Integrations" on page 420 for more information.

> **Note:**

Users who have the required permissions under **Settings** > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Assets** can deploy and export assets.

To import integrations, select the integration, and then click **Import Integrations**. See " Import Integrations" on page 421 for more information.

**Note:**
If assets used by an integration are deleted, you will not be able to deploy the integration into subsequent stages or export the integration. See "Deploy Assets" on page 696 for information on how to deploy assets.

## Point-to-Point Integrations

Integration Cloud enables you to integrate your cloud-based Software as a Service (SaaS) applications with other cloud-based SaaS applications. It also integrates your SaaS applications with on-premises applications.

Integration between two cloud providers includes the following steps:

- Invoking a source Operation on an application, which fetches data from it
- Invoking a target Operation on an application, which uploads data into it
- Filtering the data fetched from an application, before it is passed on to the target application
- Mapping the data fetched from an application, to the structure needed by the target application to which you want to upload the data.

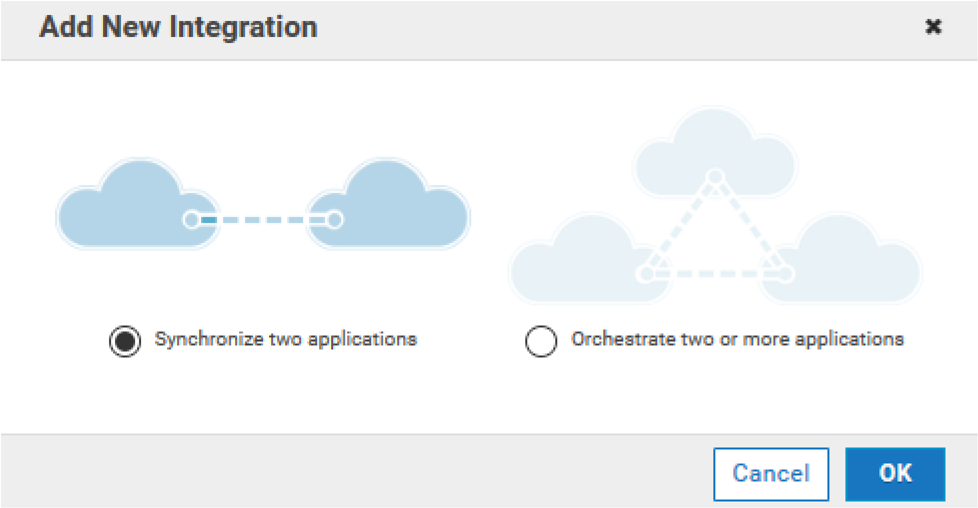
### To add or edit an existing Point-to-Point Integration

1. From the Integration Cloud navigation bar, click **Projects**. The **Projects** screen appears.

2. Select a project in which you want to create the Integration. You can also create a new project. See "Projects" on page 94 for more information.

3. To edit an existing Integration, select an Integration from the **Integrations** screen and click **Edit**.

4. To create a new point-to-point Integration, from the **Integrations** screen, click **Add New Integration**, select **Synchronize two applications**, and click **OK**. See "Orchestrated Integrations" on page 387 for information on how to create an orchestrated Integration.



Add New Integration

● Synchronize two applications

○ Orchestrate two or more applications

5. Provide a name and description of your Integration. Required fields are marked with an asterisk on the screen.

6. Drag and drop your applications to the **Source** and **Target** sections. You can also double-click an Application to move it to the required section. To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.

7. Select an Account, and then select a custom or a predefined Operation in both the Source and Target sections. Only active or enabled Accounts are listed in the drop down list.

   **Note:**
   If you had already done the mapping for a source and target Operation, and you want to change any of the source and target Operations, all the mappings you had performed before will be removed.

8. Click **Next** to filter the data fetched by the application selected in the source section, before it is passed on to the application selected in the target section. Click **Load Data** to preview the data as well as view the data filters. The source Operation fetches the data and displays a sample of the data in the preview pane. Out of all the records fetched, you may want to upload only selected records. To do this, you can have a selection or a filter criteria so that you can view only a few records. A **sample preview** of only a few records can be viewed to analyze the kind of data that exists in the system. After you analyze the records, you can set filters, to upload, for example only Accounts that are based out of California to the target application. After you set the filters, whenever you run the Integration, all records will be fetched from the source application, but only the filtered records will be moved to the target application after mapping and transformation.

9. Click **Next** to map the data fetched by the application selected in the source section, to the structure needed by the application selected in the target section.

Select a field from the source section and drag and drop it on to a relevant field in the target section. Select a mapping and then click the **Unmap** icon to unmap only the selected mapping. Click the **Clear All** icon to unmap all the mapped elements, values set for fields, and transformers. To view only the mapped fields, select the **Show Only Mapped Fields** option.

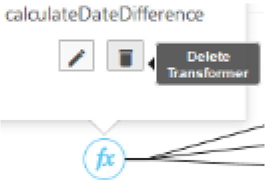
You can select a field in the target Operation table, and then choose to set a new value of the selected field in the target Operation. You can assign a value to a field when the field is not linked or when the field is only implicitly linked to another value in the pipeline. You cannot assign values to fields that are explicitly linked to another value in the pipeline or fields that have been dropped from the pipeline.

You can copy a field from the fields panel by clicking the ⧉ icon. Depending on the context, you can either paste the field or the field path. For example, if you copy a field and paste the field in the **Set Value** window in an Integration (double-click a field to set a value), the field path will be pasted. If you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index [10]. But if it is pasted in the document tree, it will appear as an array, like A[ ]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B.

**Note:**
The paste option is not applicable for Point-to-Point Integrations.

Click the **Add Transformer** icon to add a transformer in the **Transform Data** page. This page allows you to transform the source Operation data, for example, concatenate two strings and map it to a single field. Several *built-in services* specifically designed to translate values between formats are provided. You can transform time and date information from one format to another, perform simple arithmetic calculations (add, subtract, multiply, and divide) on integers and decimals contained in String fields, or transform String values in various ways. Reference Data is also available while transforming the data.

The **Transform Data** screen also allows you to look up and use data from another source Operation to transform the data. After adding a transformer, you can click the fx icon and select **Edit Transformer** or **Delete Transformer** to either modify the transformer or delete it.



10. Click **Next** to review your Integration.

11. Click **Save** and then click **Finish** to create your Integration.

The new Integration appears in the **Integrations** page.

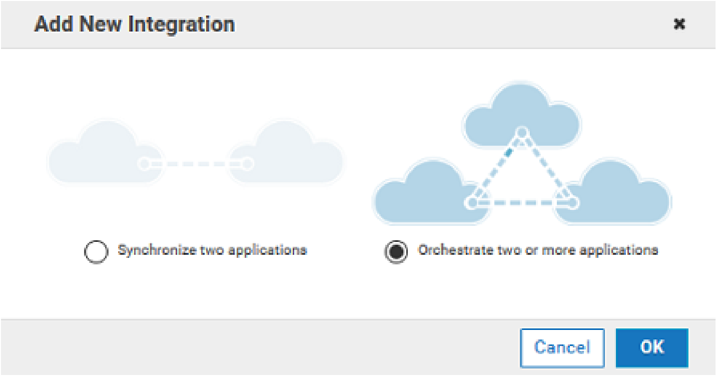
# Orchestrated Integrations

Orchestrated Integration is the process of integrating two or more applications together, to automate a process, or synchronize data in real-time. Orchestrated Integration enables you to integrate applications and provides a way to manage and monitor your integrations.

Integration Cloud supports advanced integration scenarios involving multiple application endpoints, complex routing, and Integrations involving multiple steps. Using a graphical drag and drop tool, you can create complex, orchestrated integrations and run them in the Integration Cloud environment.

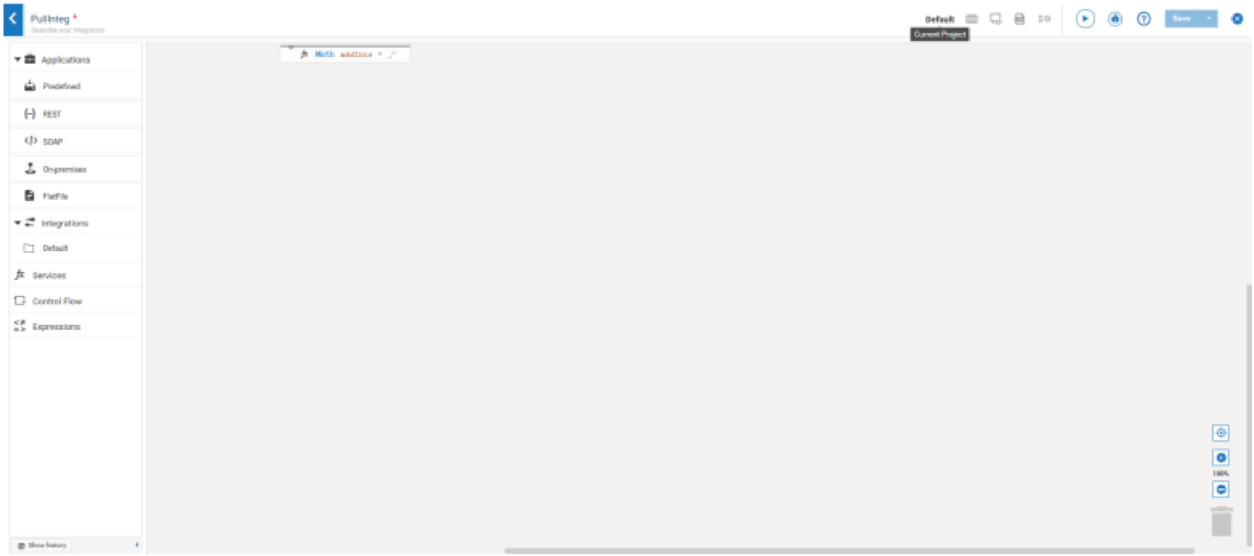
### To create an orchestrated integration

1. From the Integration Cloud navigation bar, click **Projects**. The **Projects** screen appears.
2. Select a project in which you want to create the Integration. You can also create a new project. See "Projects" on page 94 for more information.
3. To create a new Integration, from the **Integrations** screen, click **Add New Integration**.
4. Select **Orchestrate two or more applications**, and then click **OK**.


Add New Integration
×
○ Synchronize two applications
● Orchestrate two or more applications
Cancel
OK

The user interface consists of a *tool bar* and a *workspace*. The tool bar holds all the available categories with blocks. You can browse through the menu of blocks and can set up your own Integration by plugging blocks together in the workspace. The menu of blocks comes with a large number of predefined blocks from Applications, Services, Integrations, conditions to looping structures. You can drag relevant blocks from the tool bar and drop them at the anchor point.

**Note:**
Click the *Show Keyboard Shortcuts* icon available above the integration workspace area to view the available shortcut keys.

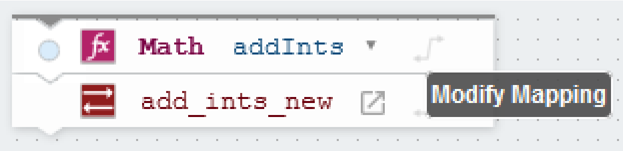


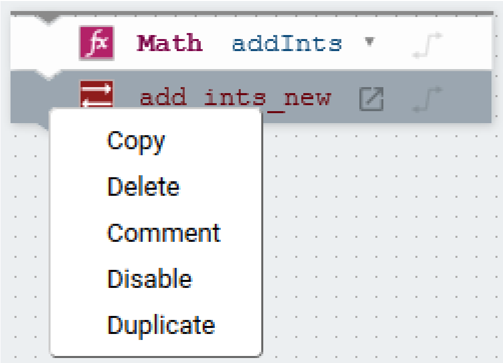
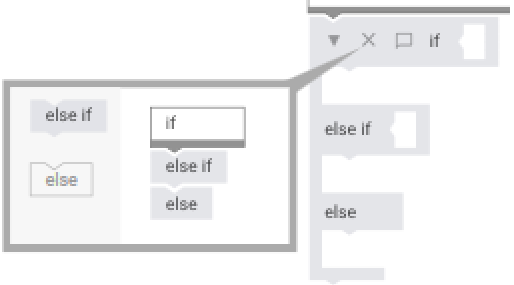
The tool bar has a large number of blocks for common instructions and the blocks are divided into the following categories:

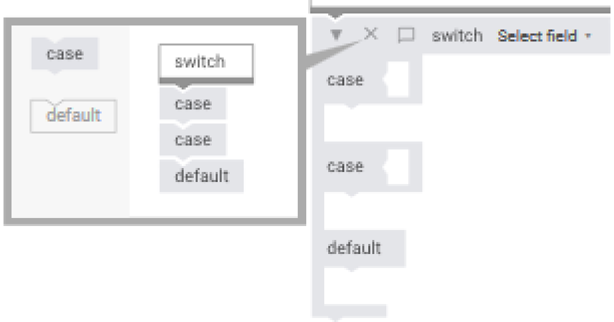
- Applications
  - Predefined Applications
  - REST Applications
  - SOAP Applications
  - On-Premises Applications
  - Flat File Applications
- Integrations
- Services
- Control Flow
- Expressions

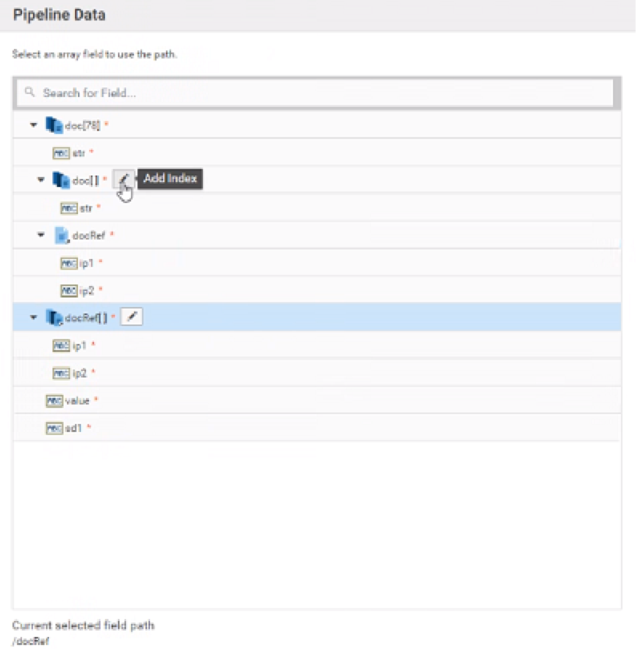
**Note:**
You can copy a block from an integration and paste that block in another integration across projects.

| Block category | Icons | Description |
|---|---|---|
| **Applications** | 💼 | Displays the Application categories available in Integration Cloud. |
| **Services** | *fx* | Use the *Service* blocks (date, math, string, and so on) to specify the service that will be invoked at run time. |

| Block category | Icons | Description |
|---|---|---|
| | | Related services are grouped in blocks. You can sequence services and manage the flow of data among them.<br>**Note:** For information on the different services, see Built-In Services.<br>The **Reference Data** block appears only if a Reference Data service is available at the **Projects > <Select a Project> > Reference Data** page. See Add Reference Data for more information. |
| **Integrations** | ⇄ | Displays the list of Integrations created in Integration Cloud. You can invoke an Integration from another Integration. When copying integrations from one stage to another, all the referred Integrations and their dependents will also be copied.<br>The **Integrations** category also lists all the *shared projects* and the integrations available in the shared projects. Integrations available in the same project are also listed. The display format for a shared asset is <*project name*> <*integration name*> ⇄ A testA, where **A** is the name of the project and **testA** is the name of the integration in the project **A**. See "Sharing Assets across Projects" on page 98 for more information.<br>Click the ↗ icon if you want to view or modify an Integration after it is dropped at the anchor point. The Integration will open up for editing in a new tab.<br>Click the ♪ icon and select **Modify Mapping** if you want to map the input of the operation from the Pipeline and also map the output of the operation into the pipeline.<br> <br>Right-click on a block to add **Comments** for the block. |

| Block category | Icons | Description |
|---|---|---|
| | | <br> Math<br>addInts<br>add ints_new<br>Copy<br>Delete<br>Comment<br>Disable<br>Duplicate<br>Click **Duplicate** to repeat the block, click **Delete** to remove the block from the workspace, click **Copy** to copy the block from the workspace, or click **Disable** to disable a block. If you disable blocks, those blocks will not be considered for execution, test, or debug operations. Click the **Show Inline Comments** option available at the top-right corner of the workspace to view the comments entered for the block. |
| **Control Flow** | ▭ | Conditional expressions, looping structures, and transform pipeline.<br>Conditional expressions perform different computations or actions depending on whether a specified boolean condition evaluates to true or false.The **if** block is used to evaluate a boolean condition and if the condition is true, statements inside the *if* block are executed. The *if* statement can be followed by an optional *else* statement, which executes when the boolean expression is false.<br><br> if<br>else if<br>else<br>else if<br>else<br>if<br>else if<br>else<br>The *if* statements are executed from the top towards the bottom. You can use one *if* or **else if** statement inside another *if* or *else if* statement(s). You cannot have multiple else statements.<br>**Switch** allows a variable to be tested for equality against a list of values. Each value is called a case, and the |

| Block category | Icons | Description |
|---|---|---|
| | | variable being switched on is checked for each case, that is, Switch evaluates a variable and skips to the value that matches the case. For example, if the Switch variable evaluates as "A", then case "A" is executed. A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases are true. You cannot insert multiple default statements.<br> case<br>default<br>switch<br>case<br>case<br>default<br>switch<br>Select field<br>case<br>case<br>default<br>**Note:**<br>You can include case steps that match null or empty switch values. A switch value is considered to be null if the variable does not exist in the pipeline or is explicitly set to null. A switch value is considered to be an empty string if the variable exists in the pipeline but its value is a zero length string.<br>**Note:**<br>Switch executes the first case that matches the value, and *exits the block*.<br>The **try catch** block is used to handle errors and exceptions. If you have a statement in the try block that has thrown an error, the error will be caught in the catch statement.<br>**Note:**<br>If an error is thrown inside the catch section of the try catch block, the error will be ignored and the next statements in the Integration will be executed.<br>**Loops** execute a set of steps multiple times based on the block you have chosen. It repeats a sequence of child steps once for each element in an array that you specify. For example, if your pipeline contains an array of purchase-order line items, you could use a Loop to process each line item in the array. Loop requires you to |

| Block category | Icons | Description |
|---|---|---|
| | | specify an input array that contains the individual elements that will be used as input to one or more steps in the Loop. At run time, the Loop executes one pass of the loop for each member in the specified array. For example, if you want to execute a Loop for each line item stored in a purchase order, you would use the document list in which the order's line items are stored as the Loop's input array.<br>The **while** loop is used to iterate a part of the program several times. If the number of iterations are not fixed, it is recommended to use the while loop.<br>The **do-until** loops are similar except that they repeat their bodies until some condition is true.<br>The **for-each** block traverses items in a collection. Unlike other for loop constructs, for-each loops usually maintain no explicit counter: they essentially say "do this to everything in this set", rather than "do this x times".<br>While selecting a field for the conditions, that is, while using the If, or Loop, or Switch statements, you can click on the **Select Field** expression and choose a field in the **Pipeline Data** dialog box to add its path to the condition. If you want to use the field element inside the array list, select the **Add Index** option to add the index and use the indexed field path.<br><br>Pipeline Data<br>Select an array field to use the path.<br> Search for Field...<br>doc[78] *<br>str *<br>doc[ ] *<br>Add Index<br>str *<br>docRef *<br>ip1 *<br>ip2 *<br>docRef[ ] *<br>ip1 *<br>ip2 *<br>value *<br>ad1 *<br>Current selected field path<br>/docRef |

| Block category | Icons | Description |
|---|---|---|
| | | While selecting a field for the iterations, that is, while using the **for-each** statements, you can click on the **Select Field** expression and choose an array in the **Pipeline Data** dialog box to add its path to the iteration. If you want to use an array element inside the array list, select the **Add Index** option to add the index and use the indexed field path. |
| | | The **Exit Integration signaling success** block allows you to successfully terminate and exit from the currently running Integration. You cannot attach child blocks to the **Exit Integration signaling success** block. |
| | | The **Exit Integration signaling failure "..."** block abnormally terminates the currently running integration with an error message. You can specify the text of the error message that is to be displayed. If you want to use the value of a pipeline variable for this error message, type the variable name between % symbols, for example, %*mymessage*%. The variable you specify must be a String. You cannot attach child blocks to the **Exit Integration signaling failure "..."** block. |
| | | The **Throw error "..."** block can be attached inside any block *except the catch section of the try catch block*, and allows you to *explicitly* throw an exception with a custom error message. If it is used inside the try section of the try catch block, the error will be caught in the catch section. If you want to use the value of a pipeline variable for this custom error message, type the variable name between % symbols, for example, %*mymessage*%. The variable you specify must be a String. You cannot attach child blocks to the **Throw error "..."** block.<br>**Note:**<br>If you add a **Throw error "..."** block inside a **try catch** block, any changes done to the pipeline variables inside the try block will be reset to the previous values existing in the pipeline. |
| | | The **Break out of loop** block should be used only within a loop and allows you to break out of the containing loop, that is, it allows you to break the program execution out of the loop it is placed in. You cannot attach child blocks to the **Break out of loop** block. |

| Block category | Icons | Description |
|---|---|---|
| | | A Loop takes as input an array field that is in the pipeline. It loops over the members of an input array, executing its child steps each time through the loop. For example, if you have a Integration that takes a string as input and a string list in the pipeline, use Loops to invoke the Integration one time for each string in the string list. You identify a single array field to use as input when you set the properties for the Loop. You can also designate a single field for the output. Loop collects an output value each time it runs through the loop and creates an output array that contains the collected output values.<br>Use the **Transform Pipeline** block to make pipeline modifications. See Pipeline and Signatures for more information. |
| **Expressions** | <≠<br>=> | Logical operations, comparisons, and values.<br>The six **comparison operators** are: equal to, not equal to, less than, less than or equal to, greater than, greater than or equal to. Each takes two inputs and returns true or false depending on how the inputs compare with each other.<br>The **and** block will return true only if both of its two inputs are also true. The **or** block will return true if either of its two inputs are true. The **not** block converts its Boolean input into its opposite.<br>You can also type a text value, select a field on which to build an expression (**Select field**), or select a block with no inputs.<br>The **Field exists** block allows you to check if a variable exists or not and can be used with other Control Flow blocks, for example, the *if* block. The *Field exists* block validates the existence of a particular field in the pipeline.<br>**Note:**<br>It is recommended not to leave an input empty. |

5. Provide a valid name and description for the Integration.

6. Click **Applications**. The list of supported Applications appears.

7. Drag and drop an Application to the root block anchor point.

8. To select the Operation and Account for the Application, click ⚙

   While editing an Account in any stage, you can select a different *Authentication Type* without impacting any integrations. So if an Integration is using an Account with a specified **Authentication Type**, the integration will run with the changed Account configuration. Further, while editing or creating an Integration, after you select the Operation, the Account field will list all the Accounts that are supported for the execution of the selected Operation.

   The following table depicts the block interactions:

| Icons | Applicable for... | Action/Description |
|---|---|---|
| 🗨 | Comments for blocks | Inline comments for all blocks. Click the **Show Inline Comments** option to view the comments entered for the blocks. |
| I/O | Applications, Services, Integrations, and the Root block | **Define Input and Output Signature**<br>Click the **Define Input and Output Signature** icon I/O to define the input and output signature of an Integration. You can declare the input and output parameters for an Integration using the **Input** and **Output** tabs. Input and output parameters are the names and types of fields that the Integration requires as input and generates as output. These parameters are also collectively referred to as a signature. For example, an Integration can take two string values, an account number (AcctNum ) and a dollar amount (OrderTotal ) as inputs and produces an authorization code (AuthCode ) as the output. On the Output tab, specify the fields that you want the Integration to return.<br>You can use a **Document Reference** to define the input or output parameters for an Integration. If you have multiple Integrations with identical input parameters but different output parameters, you can use a Document Type to define the input parameters rather than manually specifying individual input fields for each Integration. When you assign a Document Type to the Input or Output |

| Icons | Applicable for... | Action/Description |
|---|---|---|
| | | side, you cannot add, modify, or delete the fields on that part of the tab.<br>You can select a Document Type from the **Document Reference** drop-down list. To create a Document Type, from the Integration Cloud navigation bar, select **Projects > <Select a Project> > Document Types > Add New Document Type**. See “Overview” on page 636 for more information.<br>You can create pipeline variables as document references, create document types comprising of document references, and also define the signature of Integrations comprising of document references.<br>You can also copy a field from the fields panel by clicking the ⧉ icon. Depending on the context, you can either paste the field or the field path by clicking the 📋 icon. For example, if you copy a field and paste the field in the **Set Value** window in an Integration, (double-click a field to set a value), the field path will be pasted.<br>See Creating Document Types from Scratch for more information.<br>**Note:**<br>You cannot modify or paste the child fields of a Document Reference.<br>Select the **Validate input** and **Validate output** options if you want to validate the input and output to the Integration, against the service input or output signature.<br>**Select Business Data to Log**<br>Integration Cloud allows you to log select business data from the Operation and Integration signatures either always, |

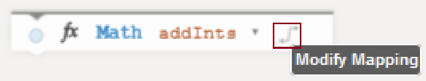
| Icons | Applicable for... | Action/Description |
|---|---|---|
| | | or only when errors occur. Values of logged fields can be viewed in the **Only Business Data** section in the Execution Results screen. You can also create aliases for the logged fields.<br>**Note:** User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.<br>To select input or output fields for logging, click the **Select Business Data to Log** icon, and in the **Select Business Data to Log** dialog box, choose whether you want to log business data only when errors occur (**On Failure**) or choose (**Always**) to always log business data. The default setting is **On Failure**. Then expand the **Input Fields** and **Output Fields** trees to display the fields available in the signature, and select the check boxes next to the fields you want to log. If you want to define an alias for a field, type an alias name beside the field. The alias defaults to the name of the selected field, but it can be modified. Click the icon to clear the selections.<br>When selecting fields for logging, you can create the same alias for more than one field, but this is not recommended. Having the same alias might make monitoring the fields at run time difficult.<br>**Map Input and Output**<br>Map the input of the operation from the Pipeline and also map the output of the operation into the pipeline. |

| Icons | Applicable for... | Action/Description |
|---|---|---|
| | | You can copy a field from the fields panel by clicking the ⧉ icon. Depending on the context, you can either paste the field or the field path by clicking the 📋 icon. If you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index [10]. But if it is pasted in the document tree, it will appear as an array, like A[ ]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B. |
| ⤴ | Control Flow > Modify Mapping | Make pipeline modifications. Edit data mapping, add Transformer, clear all mappings, add, delete, edit, or discard a field, set a value for a field and perform pipeline variable substitutions. |
| ⚙ | Applications | Account and an Operation for the Application is configured. |
| ⚙⚠ | Applications | The block is not configured. Select an Operation and an Account for the Application. |
| ⚠ | Services | The block is not configured. Select a service. |
| ↗ | Orchestrated Integrations | Click to view or modify an Orchestrated Integration after it is moved to the workspace. The Orchestrated Integration will open up for editing in a new tab. |
| * | Orchestrated Integrations | An Orchestrated Integration has been modified or newly created but not saved. |

9. Create the Integration using the available constructs by inserting the blocks, setting properties, declaring the input and output parameters, setting values, performing pipeline variable

substitutions (if you want to replace the value of a pipeline field at run time), and mapping the pipeline data.

10. Click the icon and then select **Modify Mapping** to map the Pipeline Input to the Input Signature.



11. To view only the mapped fields, select the **Show Only Mapped Fields** option. Map the Output Signature to the Pipeline Output in the **Pipeline Data** window, and then click **Finish**.

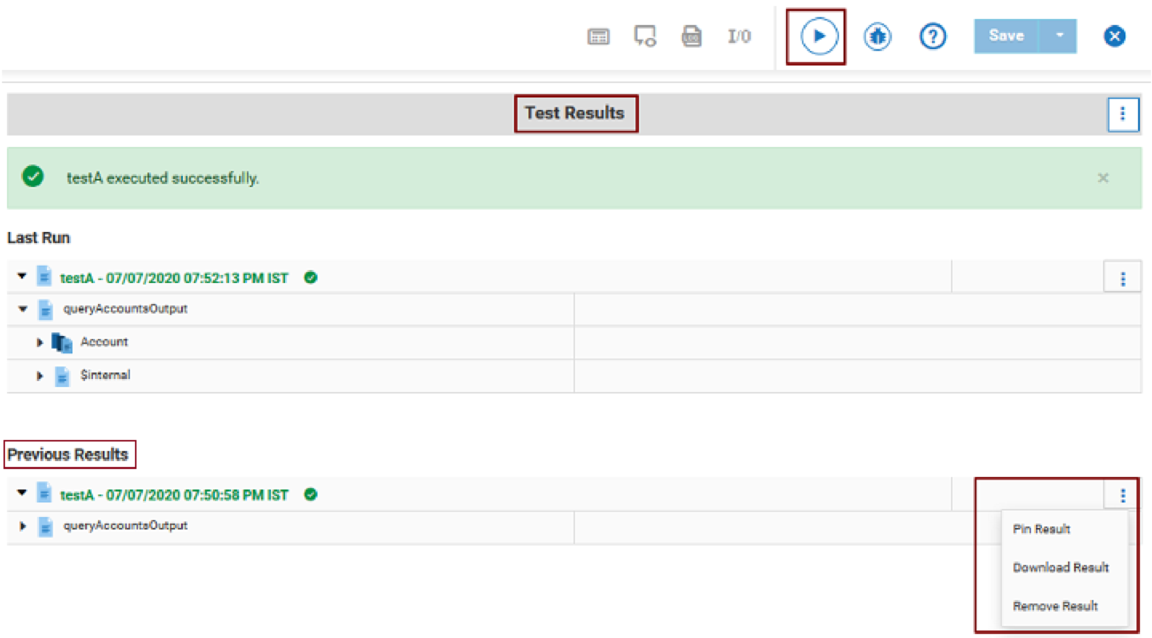
12. Use the **Transform Pipeline** block under the **Control Flow** category to adjust the pipeline at any point in the Integration and make pipeline modifications. Within this step, you can discard or remove an existing pipeline input field, (once you discard a field from the pipeline, it is no longer available subsequently), restore the discarded field, add a field, set a new value or modify the existing value of a selected field, map selected fields, remove the selected map between the fields, or perform value transformations by inserting transformers.

13. Click **Save** to save your Integration or click **Save All** to save all modified Integrations. The new Integration appears in the **Integrations** page. Click on the Integration link in the **Integrations** page to view the Integration details.

    **Note:**
    While running or debugging Integrations and testing Operations, if the input has any duplicate keys, or if the service returns an output with duplicate keys, you can view those keys.

14. After saving the Integration, in the edit Integration page, click the **Run Integration** ▶ icon to run and test the Integration execution in real time and view the execution results on the **Test Results** panel.

    The **Test Results** panel displays up to 25 test entries and the most recent test entry is located at the top of the panel. Click the ⋮ icon on the **Test Results** panel header and click **Remove All** to delete the test results permanently or click **Close** to close the test results panel.



testA executed successfully.

testA - 07/07/2020 07:52:13 PM IST
queryAccountsOutput
Account
$internal

testA - 07/07/2020 07:50:58 PM IST
queryAccountsOutput

Point to a previous test result entry, click the ⋮ icon, and click **Download Result** to save the entry locally in JSON format. Click **Remove Result** to remove the selected entry. Click **Pin Result** if you want to prevent a previous result from getting deleted as more results fill the test results panel. Click **Unpin Result** to move the result to the **Previous Results** panel.

## Pipeline and Signatures

The pipeline is the general term used to refer to the data structure in which input and output values are maintained for an Integration. The pipeline starts with the input to the Integration and collects inputs and outputs from subsequent Applications and services in the Integration. When an operation of an Application or an Integration executes, it has access to all data in the pipeline at that point.

Input and output parameters are the names and types of fields that the Integration requires as input and generates as output. These parameters are also collectively referred to as a *signature*.

For example, an Integration that takes two string values—an account number (*AcctNum*) and a dollar amount (*OrderTotal*)—as input and produces an authorization code (*AuthCode*) as output, has the following input and output parameters:

| **Input Parameters** | | **Output Parameters** | |
|---|---|---|---|
| **Name** | **Data Type** | **Name** | **Data Type** |
| *AcctNum* | String | *AuthCode* | String |
| *OrderTotal* | String | | |

Although you are not required to declare input and output parameters for an Integration, (Integration Cloud will execute an Integration regardless of whether it has a specification or not), there are good reasons to do so:

- Declaring parameters makes the Integration's input and outputs visible in the user interface. Without declared input and output parameters, you cannot:
  - Link data to and/or from the Integration using the Pipeline view.
  - Assign default input values to the Integration on the Pipeline view.
  - Run the Integration and enter initial input values.
- Declaring parameters makes the input and output requirements of your Integration known to other developers who may want to call your Integration from their programs.

For these reasons, it is strongly recommended that you make it a practice to declare a signature for every Integration that you create.

Integration Cloud supports several data types for use in Integrations. Each data type supported by Integration Cloud corresponds to a Java data type and has an associated icon. When working in the editor, you can determine the data type for a field by looking at the icon next to the field name.

The input side describes the initial contents of the pipeline. In other words, it specifies the fields that this Integration expects to find in the pipeline at run time. The output side identifies the fields produced by the Integration and returned to the pipeline.

**Guidelines for specifying input parameters**

When you define the input parameters for an Integration, keep the following points in mind:

- **Specify all inputs that a calling program must supply to this Integration.** For example, if an Integration invokes two other Integrations, one that takes a field called *AcctNum* and another that takes *OrderNum*, you must define both *AcctNum* and *OrderNum* as input parameters for the Integration.

  > **Note:**
  > The purpose of declaring input parameters is to define the inputs that a calling program or client must provide when it invokes this Integration. You do not need to declare inputs that are obtained from within the Integration itself. For example, if the input for one Integration is derived from the output of another Integration, you do not need to declare that field as an input parameter.

- **When possible, use variable names that match the names used by the Integrations.** Variables with the same name are automatically linked to one another in the pipeline. (Remember that variable names are case sensitive.) If you use the same variable names used by Integration's constituent services, you reduce the amount of manual data mapping that needs to be done. When you specify names that do not match the ones used by the constituent Integrations, you must use the Pipeline view to manually link them to one another.

- **Avoid using multiple inputs that have the same name.** Although the user interface permits you to declare multiple input parameters with the same name, the fields may not be processed correctly within the Integrations or by other Integrations that invoke this Integration.
- **Ensure that the variables match the data types of the variables they represent in the Integration.** For example, if an Integration expects a document list called *LineItems*, define that input variable as a document list.
- **Declared input variables appear automatically as inputs in the pipeline.** When you select the Transform Pipeline step in an Integration, the declared inputs appear under **Pipeline Input**.

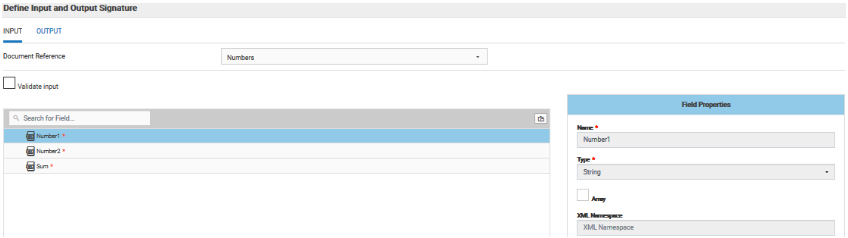
### Guidelines for specifying output parameters

On the output side of the Input/Output tab, you specify the variables that you want the Integration to return to the calling program or client. The guidelines for defining the output parameters are similar to those for defining input parameters:

- **Specify all of the output variables that you want this Integration to return** to the calling program or client.
- **Ensure that the names of output variables match the names used by the Integrations** that produce them. Like input variables, if you do not specify names that match the ones produced by the Integration's constituent services, you must use the Pipeline view to manually link them to one another.
- **Avoid using multiple outputs that have the same name.** Although the user interface permits you to declare multiple output parameters with the same name, the fields may not be processed correctly within the Integration or by other Integrations that invoke this Integration.
- **Ensure that the variables match the data types of the variables they represent in the Integration.** For example, if an Integration produces a String called *AuthorizationCode*, ensure that you define that variable as a String.
- **Declared output variables appear automatically as outputs in the pipeline.**When you select the Transform Pipeline step in an Integration, the declared output variables appear under **Pipeline Output.**

### Declaring Input and Output Parameters

Click the **Define Input and Output Signature** icon I/O to define the input and output parameters. On the **Input** tab, you define the variables that the Integration requires as input. On the **Output** tab, you define the variables the Integration returns to the client or calling program.



Define Input and Output Signature

Document Reference: Numbers

☐ Validate input

Field Properties

Name *: Number1

Type *: String

☐ Array

XML Namespace: XML Namespace

For an Integration, the input side describes the initial contents of the pipeline. In other words, it specifies the variables that this Integration expects to find in the pipeline at run time. The output side identifies the variables produced by the Integration and returned to the pipeline.

> **Note:**
> You can create pipeline variables as document references, create document types comprising of document references, and also define the signature of Integrations comprising of document references.

You can declare a signature in one of the following ways:

- **Reference a document type.** You can use a document type to define the input or output parameters for an Integration. When you assign a document type to the Input or Output side, you cannot add, modify, or delete the variables on that half of the tab.

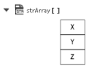
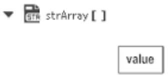
- **Manually insert input and output variables.** Click the ✚ icon to manually insert variables to the Input or Output sides.

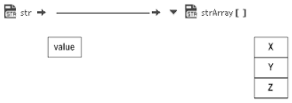
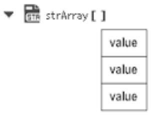
**Using a Document Type to specify Integration input or output parameters**

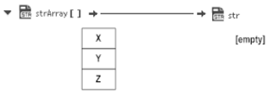
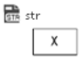
You can use a document type as the set of input or output parameters for an Integration. If you have multiple Integrations with identical input parameters but different output parameters, you can use a document type to define the input parameters rather than manually specifying individual input fields for each Integration. When a document type is assigned to the input or output of an Integration, you cannot add, delete, or modify the fields on that tab.

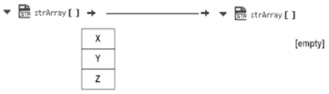
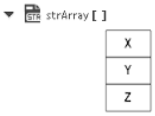
## Default Pipeline Rules for Linking to and from Array Variables

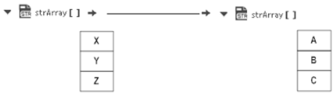
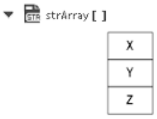
When you create links between scalar and array variables, you can specify which element of the array variable you want to link to or from. Scalar variables are those that hold a single value, such as String, Document, and Object. Array variables are those that hold multiple values, such as String List, Document List, and Object List. For example, you can link a String to the second element of a String List. If you do not specify which element in the array variable that you want to link to or from, default rules in the Pipeline view are used to determine the value of the target variable. The following table identifies the default pipeline rules for linking to and from array variables.

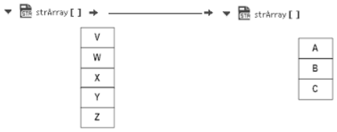
| If you link… | To… | Then… |
|---|---|---|
| A scalar variable | An array variable that is empty (the variable does not have a defined length) | The link defines the length of the array variable; that is, it contains one element and has length of one. The first (and only) element in the array is assigned the value of the scalar variable. |
|  | |  |

| If you link... | To... | Then... |
|---|---|---|
| A scalar variable | An array variable with a defined length | The length of the array is preserved and each element of the array is assigned the value of the scalar variable. |
|  | |  |

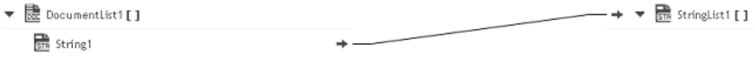
| If you link... | To... | Then... |
|---|---|---|
| An array variable | A scalar variable | The scalar variable is assigned the first element in the array. |
|  | |  |

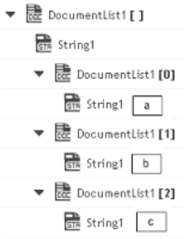
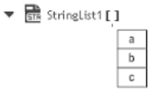
| If you link... | To... | Then... |
|---|---|---|
| An array variable | An array variable that does not have a defined length | The link defines the length of the target array variable; that is, it will be the same length as the source array variable. The elements in the target array variable are assigned the values of the corresponding elements in the source array variable. |
|  | |  |

| If you link... | To... | Then... |
|---|---|---|
| An array variable | An array variable that has a defined length | The length of the source array variable *must* equal the length of the target array variable. If the lengths do not match, the link will not occur. If the lengths are equal, the elements in the target array variable are assigned the values of the corresponding elements in the source array variable. |
|  | |  |

| If you link... | To... | Then... |
|---|---|---|
|   strArray [ ]<br>V<br>W<br>X<br>Y<br>Z<br>strArray [ ]<br>A<br>B<br>C | | No link occurs. |

A source variable that is the child of a Document List is treated like an array because there is one value of the source variable for each Document in the Document List. For example:

| If you link... | To... |
|---|---|
|  | |

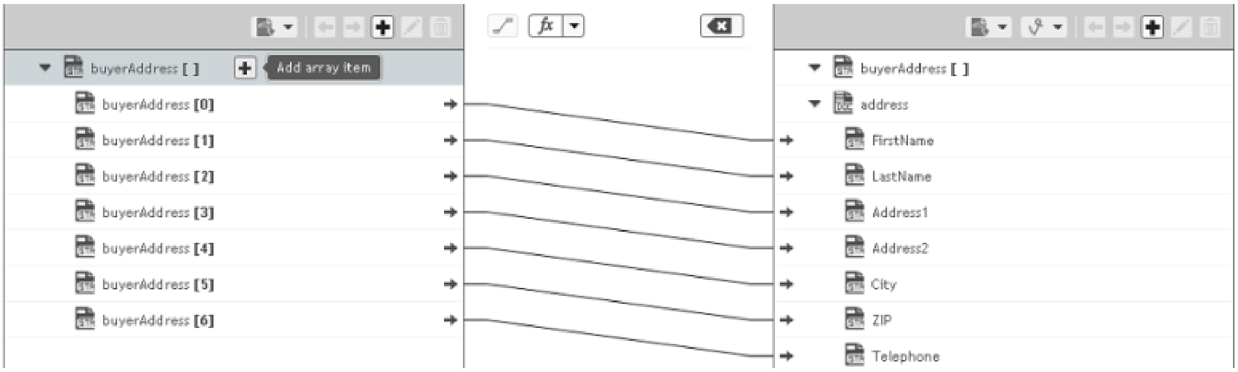
| Where the value of DocumentList1 is... | Then the value of StringList1 is... |
|---|---|
|   DocumentList1 [ ]<br>String1<br>DocumentList1 [0]<br>String1 a<br>DocumentList1 [1]<br>String1 b<br>DocumentList1 [2]<br>String1 c |  |

## Indexed Mapping

You can add an indexed item to a String List, Document List, Document Reference List, or Object List and also map the indexed item. You can delete the selected indexed item provided the indexed item or none of its child fields are mapped.

When you link to an array variable or from an array variable (String List, Document List, Document Reference List, or Object List), you can specify which element in the array you want to link to or from. Click on the **Add Array Item** icon to get an index value for the array item. Then map the indexed item to the target. For example, you can link the second element in a String List to a String or link the third Document in a Document List to a Document variable.

For example, suppose that a buyer's address information is initially stored in a String List. However, the information might be easier to work with if it is stored in a Document. To map the information in the String List to a Document, click on the **Add Array Item** icon to get an index value for the String List. Then map each indexed item to the address fields. In the following pipeline, the elements in buyerAddress String List are mapped to the address Document.



Suppose a String List has length 3 and if you link index 4 of the String List, at run time the String List length is increased from 3 to 5.

When you link a Document or Document List variable to another Document or Document List variable, the structure of the source variable determines the structure of the target variable.

## Smart Mapping

### Overview

Smart mapping provides you with recommendations while mapping the pipeline data. You can use the available services while creating orchestrations. A Machine Learning (ML) algorithm is applied to provide the suggestions. This algorithm learns from the mappings you create and provides suggestions automatically to map similar fields.

The mapping recommendations are not tenant specific. So the mapping inputs from one tenant maybe used to make recommendations for another tenant.

**Note:**
When you perform mapping, the data collected does not reflect immediately as a recommendation for another user. The information is recorded and is processed by our database at specified intervals.

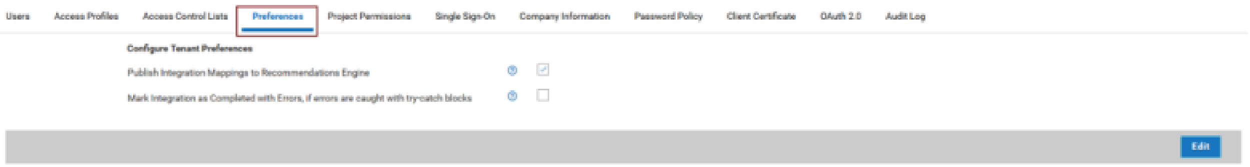
### Preconditions

Smart mapping feature is available only to those tenants who provide their consent to share their mapping information. For all trial tenants, the mapping data is collected by default. The machine learning algorithm benefits from having more data from more number of users. The mapping information collected continues to remain in the database even after a user disables this feature. However, once you disable this feature, the system does not collect any new mapping information.

**Note:**
For paid tenants, only an Admin user who has access profile **ID 4**, has the permission to enable this feature. For trial tenants and Free Forever Edition, this is always enabled and cannot be changed.

To provide your consent to share the mapping information, do the following:

1. Go to **Settings** > **Preferences**.



Configure Tenant Preferences

Publish Integration Mappings to Recommendations Engine ☑

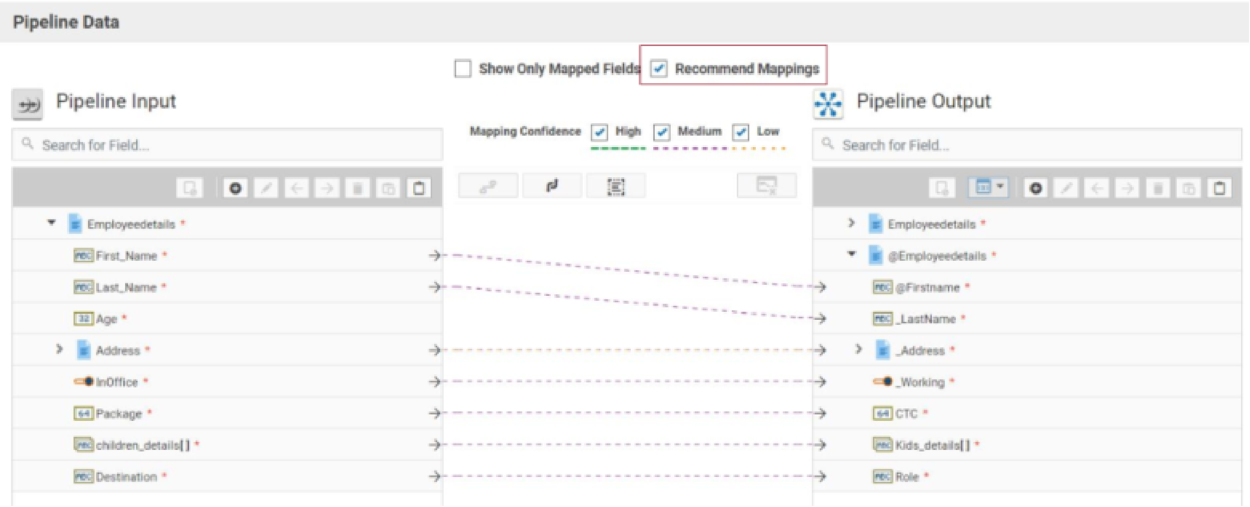
Mark Integration as Completed with Errors, if errors are caught with try-catch blocks □

2. On the **Configure Tenant Preferences** page, click **Edit**.
3. Select the check box for **Publish Integration Mappings to Recommendations Engine**.
4. Click **Save**.

## Basic Flow

You can use the smart mapping feature as follows:

1. Select the integration for which you want to perform smart mapping.
2. Click the Modify Mapping icon.
3. On the **Pipeline Data** screen, select **Recommend Mappings**. You will see a list of all the recommended mappings that are color coded based on their likelihood as shown:



□ Show Only Mapped Fields ☑ Recommend Mappings

Mapping Confidence ☑ High ☑ Medium ☑ Low

The mapping recommendations include the following based on the level of mapping confidence:

- High ---- - These have the highest probability.
- Medium ------ - These have medium probability.
- Low · · · · - These have the least probability.

4. You can clear the relevant **Mapping Confidence** check boxes **High**, **Medium**, or **Low**, depending on the recommendations that you want to view.

5. To hard map any of the recommendations, choose from the following actions:

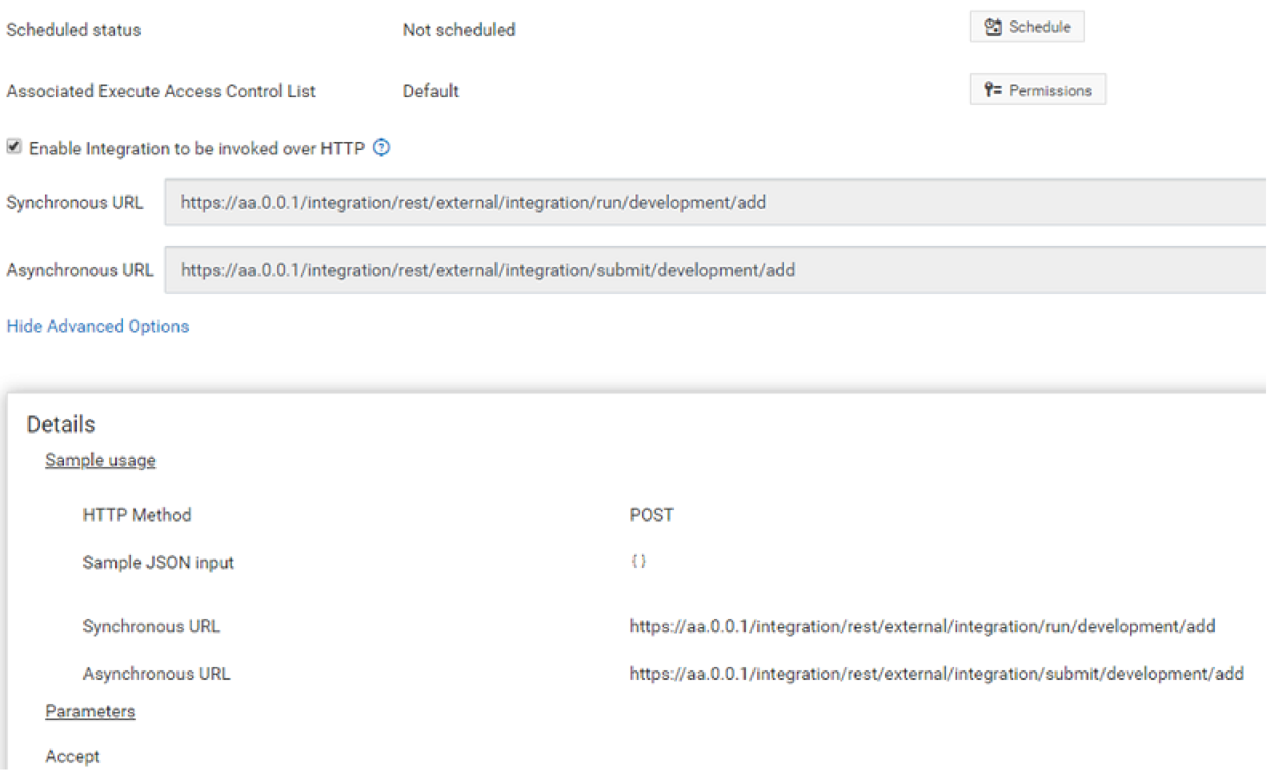
   - When all the mapping recommendations are visible, click ☰ to select all the recommended mappings and then click ✓ to accept the recommendations.
   - Filter the mappings based on the likelihood by selecting the relevant check box and then click ☰, followed by ✓. For example, select **High** and clear the other two check boxes. When you have only the mappings with the highest confidence level, click ☰ and then click ✓.
   - Select the individual mapping recommendations that you want to hard map, and then click ✓.

6. To unmap any of the hard maps, select the hard map and then click 🔗 to unmap. The recommendation disappears. To view this recommendation again, select **Recommend Mappings** again.

## Integration Details

This page allows you to view when the Integration was created or last modified, who created or last modified the Integration, references used in the Integration, and whether the Integration is scheduled. You can also edit, delete, or run the Integration, enable the Integration to be invoked over HTTP and view and add OAuth Scopes containing the exposed Integration URL, enable Integration executions to be restartable, associate another Access Control List (ACL) with the Integration, and also view the last five execution results.

To view the Integration details screen, select an Integration from the Integrations page, and then click the Integration details icon 🗒.



Scheduled status — Not scheduled — Schedule

Associated Execute Access Control List — Default — Permissions

☑ Enable Integration to be invoked over HTTP ?

Synchronous URL: https://aa.0.0.1/integration/rest/external/integration/run/development/add

Asynchronous URL: https://aa.0.0.1/integration/rest/external/integration/submit/development/add

Hide Advanced Options

Details

Sample usage

| | |
|---|---|
| HTTP Method | POST |
| Sample JSON input | { } |
| Synchronous URL | https://aa.0.0.1/integration/rest/external/integration/run/development/add |
| Asynchronous URL | https://aa.0.0.1/integration/rest/external/integration/submit/development/add |

Parameters

Accept

| Option | Description |
|---|---|
| **Overview** | This page allows you to view the components used to create the Integration by clicking the uses ▤ icon, when the Integration was created or last modified, who created or last modified the Integration, and whether the Integration is scheduled. You can edit, delete, or run the Integration, enable the Integration to be invoked over HTTP and view and add OAuth Scopes containing the exposed Integration URL, enable Integration executions to be restartable, and associate another Access Control List (ACL) with the Integration. |
| **Last 5 Execution Results** | Click this tab to view the last five execution results panel. This screen allows you to view the audit trail of the executions that happened in the current stage. See Execution Results for information on the **Last 5 Execution Results** table columns.<br>You can restart or resume integrations from the **Last 5 Execution Results** page. |
| **Preview** | View the pipeline and mapping details for a previous version of an orchestrated integration in a higher stage. You will not be able to make any modifications to the existing pipeline and mapping data. |
| **Uses** | Displays the components used to create the Integration. This field will appear only if the selected Integration has components. |

| Option | Description |
|---|---|
| | **Note:**<br>If assets used by an Integration are deleted, you will not be able to pull the Integration into subsequent stages or export the Integration. |
| **Created on** | Displays the date and time when the Integration was created. |
| **Created by** | Displays the user who created the Integration. |
| **Edit** | Click this option to modify the Integration. |
| **Delete** | Click this option to delete the Integration from the active stage. |
| **Run Now** | Click this option to submit the Integration for execution. You can provide inputs to the Integration based on the defined input signature. |
| **Last modified/Last modified by** | When and by whom was the Integration last modified. |
| **Scheduled status** | If the Integration in the active stage is scheduled, the status of the Integration displays **Scheduled**, else it appears as **Not Scheduled**. The Status appears as **Paused** if the Integration has been paused. |
| **Schedule** | Click this option to define a schedule. Select **Run Once** if you want to schedule the Integration to run just once immediately or run once at a specified date and time.<br>Select **Run Recurrently** if you want to define a recurrence pattern. You can define a recurrence pattern daily, weekly, monthly, and in hours. Select the frequency (Hourly, Daily, Weekly, Monthly) with which the pattern recurs, and then select the options for the frequency. Click the ➕ icon to repeat the execution for daily, weekly, and monthly schedules.<br>Click the 🗑 icon to delete the selected execution time for daily, weekly, and monthly schedules. Select **Prevent concurrent executions** to skip the next scheduled execution if the previous scheduled execution is still running except when the previous scheduled execution is running for more than 3 hours. In this case, the next scheduled execution will start even if the previous scheduled execution is still running. If you do not select this option, the next scheduled execution will start, even if the previous scheduled execution has not yet completed.<br>Click **Delete** if you want to permanently remove the current recurrence schedule.<br>Click **Next** to provide inputs to the Integration based on the defined input signature. |
| **Modify Schedule** | Click this option to change the existing schedule. |
| **Pause** | Click this option to pause the Integration that was scheduled. |

| Option | Description |
|---|---|
| **Resume** | Click this option to start the scheduled Integration that was paused. |
| **Associated Execute Access Control List** | Displays the Access Control List (ACL) associated with the Integration. Integration Cloud associates the default ACL, *Default*, to an Integration when the Integration is created. Click **Permissions** to associate the Integration with another ACL.<br>**Note:**<br>The ACL will be enforced only if an Integration acts as a top-level Integration. For example, if Integration A has Integration B and Integration C as sub-Integrations, then the ACL if associated, will be enforced only on Integration A. |
| **Enable executions to be restartable** | For an orchestrated Integration, select the **Enable executions to be restartable** option if you want to enable Integration executions to be restartable or resumable. If an operation fails, you can resume the Integration execution from the point where it had failed from the **Execution Results** page (**Resume** option). Resuming an execution does not execute the previous successful operations but executes only the failed operations and operations that are not yet executed.<br>You can also restart an execution from the **Execution Results** page (**Restart** option). When an Integration is restarted, the execution occurs from the beginning of an Integration.<br>**Note:**<br>The *Restart/Resume* capability is available only if you have the required license for restarting and resuming Integrations. You must also have the Integration execution (Execute) permission if you want to restart or resume an execution.<br>**Note:**<br>You cannot resume successful Integrations but can only restart them. If an Integration has referenced Integrations, then those referenced Integrations can also be restarted or resumed. Only top level Integrations are displayed in the **Execution Results** page and the referenced Integrations are displayed in the **Execution Details** page of that top level Integration execution. The **Audit Log** will display the "Restart" and "Resume" entries.<br>**Note:**<br>User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.<br>See Execution Results for more information. |

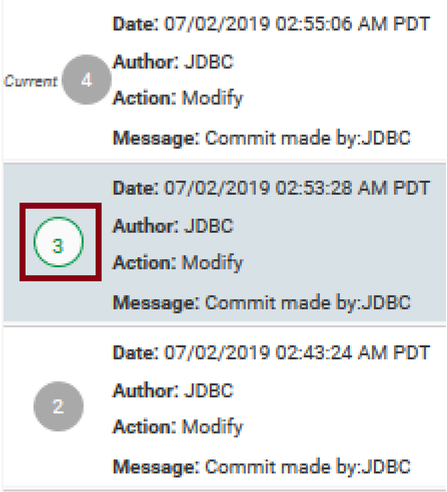
| Option | Description |
|---|---|
| | **Note:**<br>Enabling this option will increase the execution time of this Integration.<br>Once the Integration is updated, you cannot restart or resume its executions that have occurred before the update.<br>Integrations using Operations and other Integrations, which have fields of type “Object” in their signature, may not execute properly when restarted or resumed. |
| **Enable Integration to be invoked over HTTP** | Select the **Enable Integration to be invoked over HTTP** option if you want to trigger the execution of an Integration from an external system. This option provides you with one more way to trigger Integration executions from a software application, for example, a REST client, apart from manual and scheduled integrations from the user interface.<br>Once the Integration is enabled to be invoked over HTTP, the HTTP request URL appears. Click the **Show Advanced Options** link to view the HTTP Method, sample JSON input, and the parameters that are required to invoke this Integration from an external system.<br>You need to provide the HTTP URL, Method, required JSON body, and necessary header parameters in the external program, including the required security credentials (user name and password) for invoking the Integration. After the Integration is executed, the response will contain the pipeline data.<br>**Synchronous Request URL**<br>You can execute integrations synchronously using the run URL:<br>`https://<sub-domain>.<domain>/integration/rest/external/integration/run/<stagename>/<integrationname>run` - Integration will be executed and the response will contain the pipeline data.<br>*sub-domain* is a domain that is part of the primary domain.<br>*run* - Integration will be executed and the response will contain the pipeline data.<br>*stagename* is the name of the active stage.<br>*integrationname* is the name of the Integration.<br>**Note:** |

| Option | Description |
|---|---|
| | You must provide your user name and password to execute the Integration from the external program, else you may encounter the 401 - Unauthorized User Error.<br>**Asynchronous Request URL**<br>You can execute integrations asynchronously using the submit URL:<br>https://<sub-domain>.webmethodscloud.com/integration/<br>rest/external/integration/submit/<stagename>/<integrationname><br>**submit** - Integration has been submitted for execution and the response will contain a status indicating whether the Integration has been submitted for execution. When the request is submitted for execution using the *submit* option, the response will contain a reference to the execution result identifier so that a new HTTP call can be made later to get the execution results.<br>Application Status Codes for *submit*:<br>• 0 - SUCCESS: Successfully submitted the Integration for execution.<br>• -1 - ERROR: Problem while submitted the Integration for execution.<br>To get the execution results, construct the URL of the new HTTP call from the **URI** field available in the Response section.<br>To construct the URL of the new HTTP call, add the response URI obtained from *resultReference* in the Response section to: https://<sub-domain>.<domain>.com<br>Response URI format:<br>https://<sub-domain>.webmethodscloud.com/integration/<br>rest/external/integration/execution/result?resultReference<br>=76ff5733-6a21-4b02-864f-e958f698373<br>*HTTP Status Codes*<br>• 200 - OK<br>• 500 - Internal Server Error<br>• 401 - Unauthorized User Error<br>**Note:**<br>You must provide your user name and password to execute the Integration from the external program, else you may encounter the 401-Unauthorized User Error. Further, if the query response HTTP |

| Option | Description |
|---|---|
| | status code is 404 - Not Found, it means that either the Integration is not yet run or the *resultReference* is not correct. |
| **OAuth Scopes containing the exposed Integration URL** | A “scope” on page 84 defines the services the client can access on behalf of the resource owner and consists of one or more services. If access is granted for a scope, then access is granted for all the services in that scope. When a request is made, Integration Cloud verifies that the scope is defined for a client. The client is allowed to access only the service URLs that are specified for the scope.<br>The **OAuth Scopes containing the exposed Integration URL** option appears when you select the **Enable Integration to be invoked over HTTP** check box. Click **OAuth Scopes** to view the OAuth scopes that contain the exposed URL of the selected Integration.<br>In the **OAuth Scopes** dialog box, click **Add URL to Another Scope**. In the **Add Exposed URL to OAuth Scope** dialog box, select **Add To Existing Scope** to add the exposed Integration URL to an existing scope or select **Add New Scope** to create a new scope and add the exposed Integration URL to that new scope. |

## Versioning of Integrations

Integration Cloud allows you to view the version change history of an Integration. Click the **Show history** option available on the tool bar panel to view the version change history.

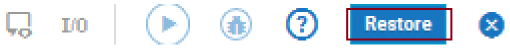


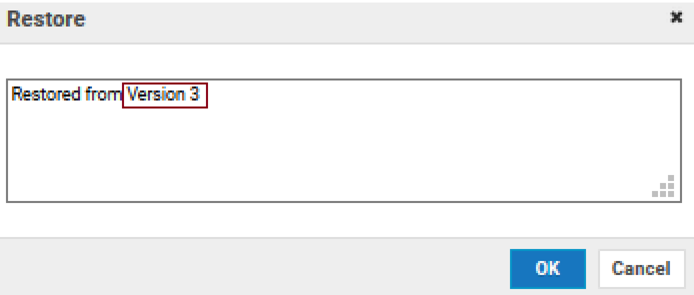
Current 4
Date: 07/02/2019 02:55:06 AM PDT
Author: JDBC
Action: Modify
Message: Commit made by:JDBC

3
Date: 07/02/2019 02:53:28 AM PDT
Author: JDBC
Action: Modify
Message: Commit made by:JDBC

2
Date: 07/02/2019 02:43:24 AM PDT
Author: JDBC
Action: Modify
Message: Commit made by:JDBC

While editing the Integration, you can restore an earlier version of the Integration. To restore an earlier version of an Integration, select the earlier version of the Integration, and then click the **Restore** option to restore that previous version of the Integration. If you have reverted to an earlier version and there is a scheduled execution for the Integration, the reverted version of the Integration will be run as per the defined schedule.





Restored from Version 3

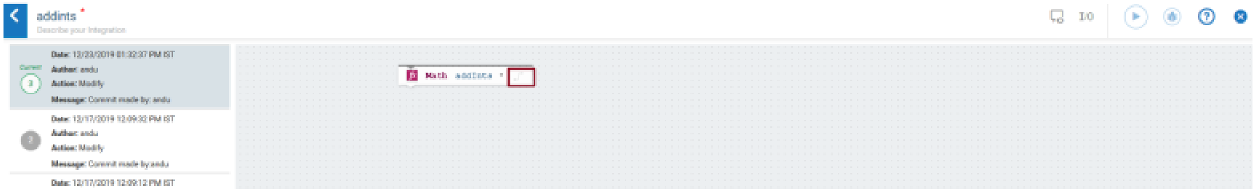
**Note:**
If an Integration references any other Integration, then the input/output mapping of the referenced Integration will be restored to that particular version. But if the input/output mapping of the referenced Integration has been modified in a later version, the modifications will break the mappings and the Integration execution may not be successful.

**Note:**
If you delete an Integration and then create another Integration with the same name, the version history of the deleted Integration will be available.

## Preview Integrations

Integration Cloud allows you to view the pipeline and mapping details for a previous version of an orchestrated integration.

To view the pipeline and mapping details in the **Development** stage, select the Integration from the **Integrations** page and click the Integration name link. Then click the **Show History** option available on the tool bar panel to view the version change history for the integration. On the version change history panel, select a previous version of the integration. When you select an Integration version, you will notice that options to configure accounts and operations are not available. You will not be able to make any modifications to the existing pipeline and mapping data.

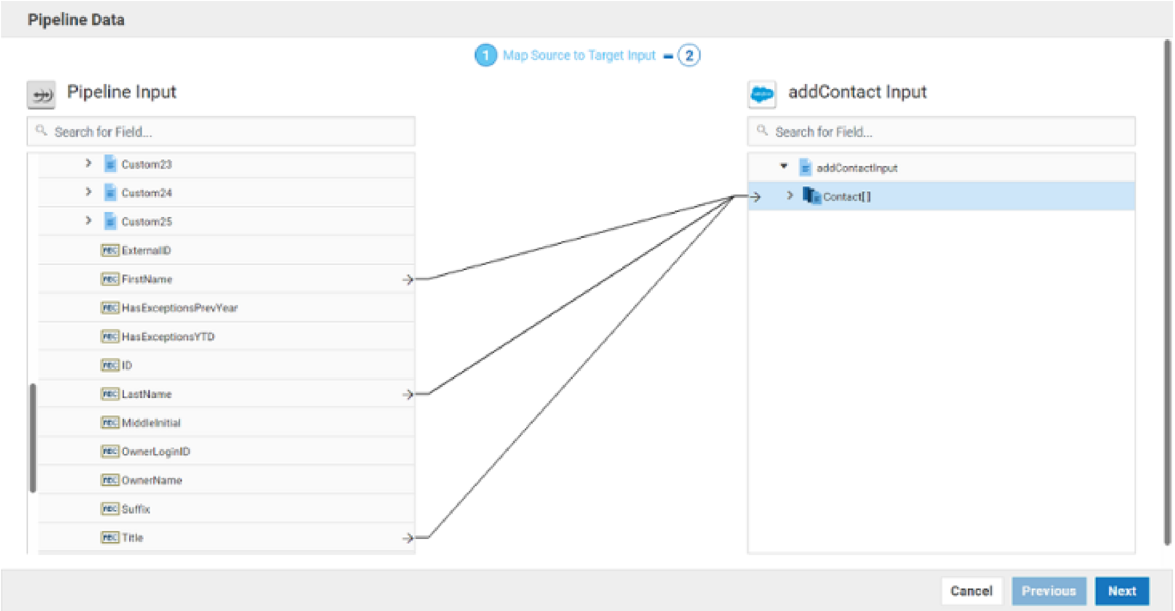


**Date:** 12/23/2019 01:32:37 PM IST
**Author:** andu
**Action:** Modify
**Message:** Commit made by: andu

**Date:** 12/17/2019 12:09:32 PM IST
**Author:** andu
**Action:** Modify
**Message:** Commit made by:andu

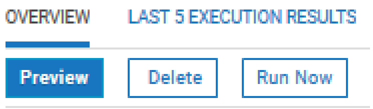
**Date:** 12/17/2019 12:09:12 PM IST

Click  to view the Pipeline Data window, which shows all the mappings that are part of this Integration.



**Note:**
In the pipeline preview, none of the action buttons are available. Also, you cannot modify the existing pipeline or the mapped data.

To view the pipeline and mapping details for a previous version of an orchestrated integration in *any stage other than the Development stage*, change the stage, select the integration from the **Integrations** page, click the icon, and then click **Preview** on the Integration **Overview** page.



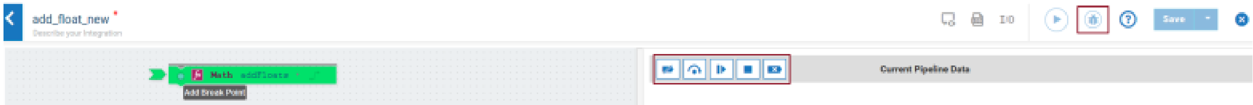
## Debug Integrations

You can debug an orchestrated Integration and can inspect the data flow during the debugging session. You can debug an orchestrated Integration only in the Development stage and after the Integration has been saved.

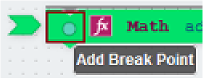
You can do the following in debug mode:

- Start an Integration in debug mode, specify the input values, and inspect the results.
- Examine and edit the pipeline data before and after executing the individual blocks.
- Monitor the execution path, execute the blocks one at a time, or specify breakpoints where you want to halt the execution.

To start the **Debug** mode, from the **Integrations** page. click the Integration name link to modify the Integration, and then click the **Debug Integration** icon .



The following table describes the options available in the **Debug** panel:

| Icons | Applicable for... | Action/Description |
|---|---|---|
|  | **Inserting Breakpoints** | A breakpoint is a point in an Integration where you want processing to halt when you debug that Integration. Breakpoints can help you isolate a section of code or examine data values at a particular point in the execution path. For example, you might want to set a pair of breakpoints before and after a particular block so that you can examine the pipeline before and after that block executes.<br>Breakpoints are recognized only when you execute an Integration in debug session. To insert a breakpoint, in debug mode, click the top-left corner of the block. To remove a breakpoint, click on the inserted breakpoint.<br>When you execute an Integration that contains a breakpoint, the Integration is executed up to, but not including the designated breakpoint. At this point, processing stops and the debug session suspends. To resume processing, select **Resume**. After you resume the debug session, the Integration flow stops at the next breakpoint. |
| ⊘ | **Ignore All Breakpoints** | Ignores all breakpoints inserted in the Integration blocks. You cannot insert breakpoints for variables in the **Expressions** category. |

| Icons | Applicable for... | Action/Description |
|---|---|---|
| ↷ | **Stepover** | Executes the current block. Integration Cloud suspends the debug session immediately before executing the next block in the Integration. |
| ▶ | **Resume** | The debug session resumes but suspends at the next breakpoint. |
| ■ | **Stop** | Terminates the debug session.<br>A debug session may also stop by itself for the following reasons:<br>- The Integration that you are debugging executes to completion (error or success).<br>- You select **Stepover** for the last step in the Integration.<br>- You **Exit** the Integration. |
| ✖ | **Clear All Breakpoints** | Removes all breakpoints inserted in the Integration. |

### Modifying the current pipeline data while debugging

During debugging, you can modify the contents of the pipeline. The changed values will be applied when you perform a **Stepover** or **Resume**.

While modifying the pipeline, keep the following points in mind:

- You can modify the pipeline data only during an active debug session.
- When you modify values in the pipeline, the changes apply only to the current debugging session. The Integration is not permanently changed.
- You can only modify existing variables. You cannot add new variables to the pipeline.

## Lock and Unlock Integrations

An Integration is an orchestration of a source and a target Operation with appropriate data mappings and transformations.

**Note:**
Users who have the required project permissions under **Settings** ⚙ can create, update, delete, and execute Integrations.

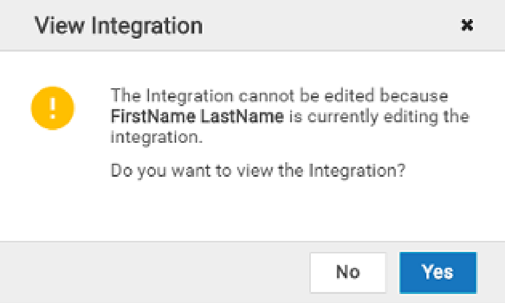
Integration Cloud allows you to manage an Integration during the development life cycle by auto locking. When you edit an Integration, it is automatically locked for you. This restricts multiple users editing that Integration at the same time.

After you edit an Integration and save the changes, you can exit the edit mode to unlock the Integration and make it available for other users. If you have locked the Integration and if another user opens the Integration in the preview mode, then that user gets notified when the lock gets released.

**Note:**
Only the user who locked the Integration or an Administrator can unlock the Integration. To unlock an Integration, from the Integrations page, click the Integration link. The Integration **Overview** page appears. From the Integration **Overview** page, click **Unlock**.

## View the status of a locked Integration

If an integration is locked, the lock status messages tells you who owns the lock.

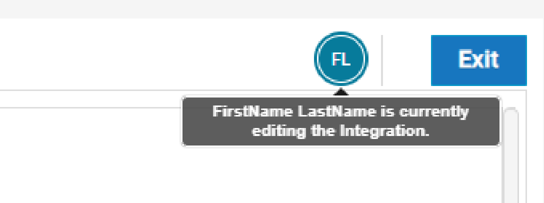


View Integration

The Integration cannot be edited because FirstName LastName is currently editing the integration.

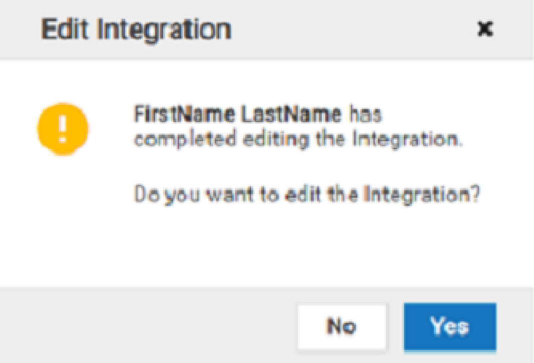
Do you want to view the Integration?

You can view the lock status of an integration in the workspace.



FirstName LastName is currently editing the Integration.

If the integration is already locked and if the current owner of the lock exits the integration, the integration screen notifies that the selected integration is ready for editing.



Edit Integration

FirstName LastName has completed editing the Integration.

Do you want to edit the Integration?

## Export Integrations

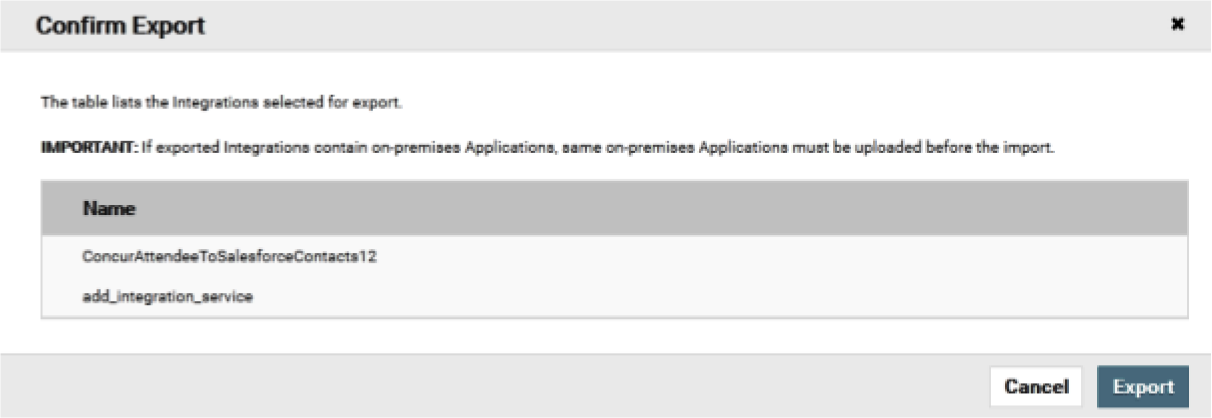
Integration Cloud allows you to export Integrations from the **Integrations** page. The export capability is available only if you have the required license for exporting Integrations. You can export Integrations from one tenant and import those Integrations to another tenant. Ensure that you have the **Export** Integration permission to export Integrations.

### To export Integrations

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Integrations**. The **Integrations** page appears.

2. Select the Integrations from the **Integrations** page and click **Export**. If the Integrations you are exporting use Reference Data, Document Types, SOAP, or REST Applications, then those Applications including the Reference Data and Document Types will also be exported along with the Integrations.

   The **Confirm Export** dialog box appears.



Confirm Export

The table lists the Integrations selected for export.

**IMPORTANT:** If exported Integrations contain on-premises Applications, same on-premises Applications must be uploaded before the import.

| Name |
|---|
| ConcurAttendeeToSalesforceContacts12 |
| add_integration_service |

3. Click **Export** to export the Integrations. The Integrations will be downloaded as a zip file to your default download folder. The zip file size must not be greater than 50 MB. Do not modify the contents of the exported zip file. If you modify the contents of the zip file, the Integrations cannot be imported back to Integration Cloud.

### Exporting Integrations having on-premises Applications

After exporting an Integration that has an on-premises Application, if you want to import the Integration, then before importing the Integration, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the Integration.

## Import Integrations

Integration Cloud allows you to import Integrations from the **Integrations** page. You can import Integrations from a zip file that was earlier exported from Integration Cloud. You can export Integrations from one tenant and import those Integrations to another tenant. You can import Integrations provided you have the **Create** Integration permission.

**Note:**
If you want to import an Integration that has an on-premises Application, before importing the Integration, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the Integration.

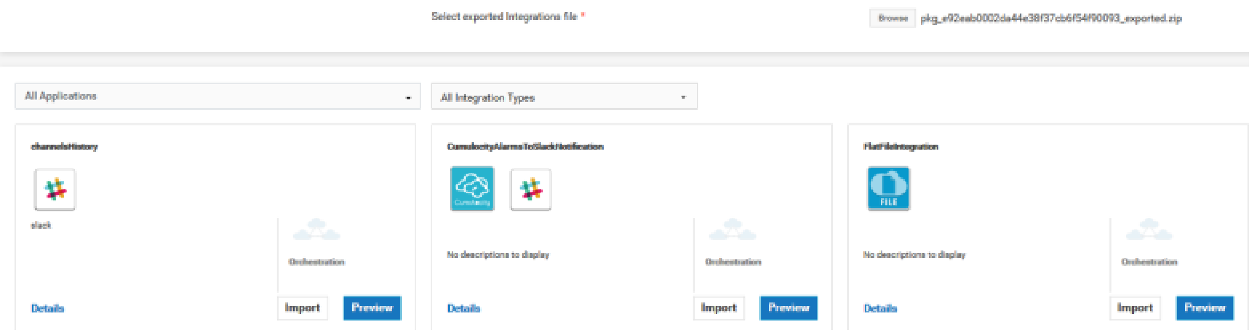
### To import Integrations

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Integrations**. The **Integrations** page appears.

2. Click **Import Integrations**.

   The **Import Integrations** page appears.

3. Click **Browse** and select the zip file that contains the exported Integrations. The zip file size must not be greater than 50 MB. The Integrations available in the zip file appears in the pane.



Select exported Integrations file * pkg_e92eab0002da44e38f37cb6f54f90093_exported.zip

4. Click **Preview** to view an Integration or click **Import** to import an Integration.

5. In the **Configure Project and Applications** screen, select the **Account** for each Application or create a new Account, and then click **Next**.

   **Note:**

If the Integrations you are importing use SOAP or REST Applications and if those Applications do not exist in your system, you will not be able to create Accounts at this step. Continue importing the Integrations. The Applications will also be imported along with the Integrations. After importing, create the Accounts and then configure them in the imported Integrations.

**Note:**
If an Integration you are importing uses an on-premises Application and if the Application does not exist in your system, you will not be able to select the Account at this step. The Account will appear only after you have uploaded the on-premises Application. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for information on how to upload on-premises Applications.

The **Overview and Save Integration** screen appears.

6. In the **Overview and Save Integration** screen, provide a name and description for your Integration. By default, the Integration name and description appears.

7. Click **Finish**. If you have existing references (Reference Data, Document Types, and custom Operations) with the same name in the development stage, the **Copy References** screen appears. Click **Cancel** to go back to the **Overview and Save Integration** screen. By default, all references are selected in the **Copy References** screen.

8. Deselect the references that you do not want to replace and then click **Continue** to replace or overwrite all the selected references from the Integration in the development stage.

   The Integration details screen appears for the newly created Integration.

# 8 Upgrade assets

# How to upgrade assets

You can upgrade assets, for example, Accounts, Operations, and the associated Integrations which uses those assets, from a lower version to a higher version. When an upgrade is available for a version, the upgrade notification text *Upgrade available for this version*, appears beside the relevant Application on the **Applications** screen, else the message *This is the latest version* appears.

**Note:**
Users who have the Application **Upgrade** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Application** can perform the upgrade task.

If an upgrade is available for a version, and if you click the **Add New Account** button on the Application specific Accounts screen, a dialog box appears where you can either select **Upgrade** to start the asset upgrade process or select **Skip** to add a new Account.

If you click the **Upgrade** button, the upgrade confirmation screen appears which displays the number of assets (Accounts, Operations, and the associated Integrations) that will be upgraded. The screen also displays details of all the conflicting assets. Conflicting assets are those assets that exist in the higher version with the same name.

On the upgrade confirmation screen, select **Skip** if you do not want to upgrade the conflicting assets to the higher version. You can also select to **Overwrite** if you want the conflicting assets in the higher version to be replaced with the lower version assets. Here, the higher version assets will be deleted and will be replaced with the lower version assets.

**Note:**
The upgrade process upgrades Accounts from a lower version to a higher version only in the Development stage. If you want to reflect the upgraded Accounts in other stages in the higher version, you must *manually* configure the Accounts in the different stages from the Account configuration screen, and then **Pull** the Integration in the respective stages.

Integration Cloud performs the following tasks if you click **Upgrade** on the upgrade confirmation page:

- Migrates all the Accounts from the lower version to the higher version only in the development stage.
- Migrates all the custom Operations and predefined Operations to the latest version.
- Updates those Integrations which uses the upgraded Accounts and Operations.
- Updates Integrations only in the development stage.
- Displays the upgraded Accounts, Operations, and Integrations in the Integration Cloud Audit Log.
- Displays the upgrade results along with a list of all the modified Accounts, Operations, and Integrations.

- Displays a message in case of an upgrade failure and performs rollback of the Accounts, Operations, and Integrations in case of an error in the upgrade process.

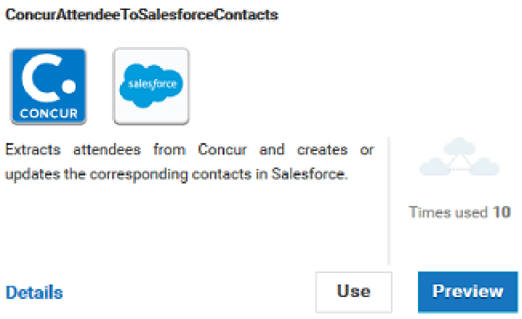
# 9 Recipes

## How to view and use Recipes

**Recipes** are pre-built Orchestrated or Point-to-Point Integration templates that can be used to create an Integration. Recipes are based on the most common integration needs and can significantly reduce the effort required to build an Integration. A recipe includes associated assets, for example Applications, Operations, Reference Data, and so on, that are used to create an Integration. A detailed description of the recipe along with its assets are available for preview, which helps you to select the right recipe. All Integrations created from recipes are initially copied to the development stage. The Recipes page is paginated to identify the sequential order of the pages. You can also select the number of recipes to be viewed per page.

### To view and use recipes

1. From the Integration Cloud navigation bar, click **Recipes**. The **Recipes** page appears. By default, recipes for all Applications and for all Integration types (Orchestrated and Point-to-Point) appears. You can search Recipes by Application names, name of the recipe, and for a specific Integration type. The **Recipes** page also displays the number of times you have used a recipe to create Integrations and the Applications referenced in the recipe. If the main integration created out of a recipe does not have Applications but has sub-integrations, and if the sub-integrations have Applications, then the Applications are pulled from the sub-integrations. The logos of the Applications in the sub-integrations will appear on the Recipes page. The Recipes page is paginated to identify the sequential order of the pages. You can also select the number of recipes to be viewed per page.

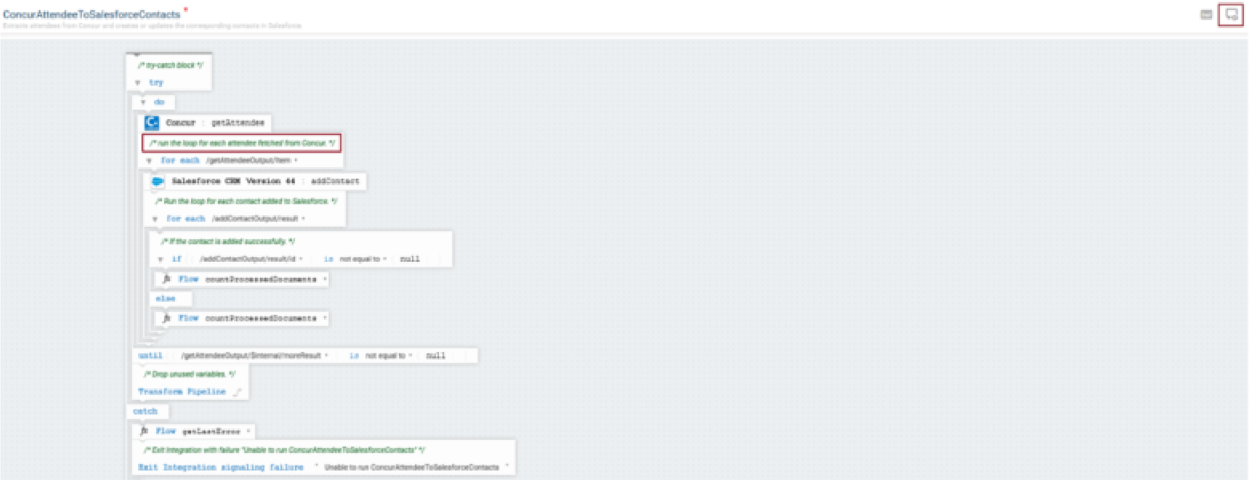


ConcurAttendeeToSalesforceContacts

Extracts attendees from Concur and creates or updates the corresponding contacts in Salesforce.

Times used 10

2. Click **Preview** to see a view-only mode of the Integration details of the recipe. Click the **Show Inline Comments** icon 💬 to view the inline comments.



ConcurAttendeeToSalesforceContacts *

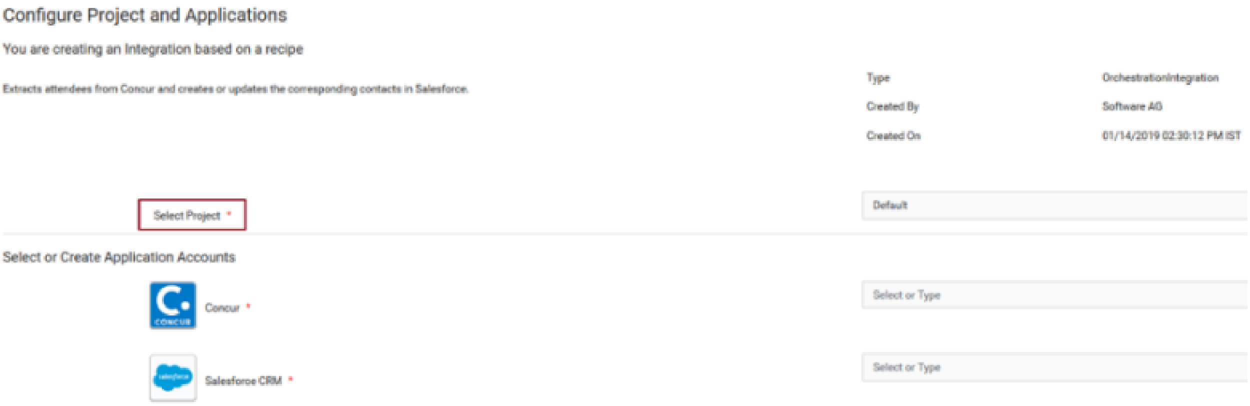
Extracts attendees from Concur and creates or updates the corresponding contacts in Salesforce

```
/* try-catch block */
try
  do
    Concur : getAttendee
    /* run the loop for each attendee fetched from Concur. */
    for each /getAttendeeOutput/item
      Salesforce CRM Version 44 : addContact
      /* Run the loop for each contact added to Salesforce. */
      for each /addContactOutput/result
        /* If the contact is added successfully. */
        if /addContactOutput/result/id is not equal to null
          Flow countProcessedDocuments
        else
          Flow countProcessedDocuments
  until /getAttendeeOutput/$internal/moreResult is not equal to null
  /* Drop unused variables. */
  Transform Pipeline
catch
  Flow getLastError
  /* Exit integration with failure 'Unable to run ConcurAttendeeToSalesforceContacts' */
  Exit Integration signaling failure ' Unable to run ConcurAttendeeToSalesforceContacts '
```

3. Click **Details** to view a detailed description of the recipe and the references in the **Recipe Details** page.

4. From the **Recipes** or **Recipe Details** page, click **Use** if you want to apply the recipe to create a new Integration. The **Configure Project and Applications** page appears.

5. In the **Configure Project and Applications** page, select the project where you want to use this recipe. If an integration with the same name already exists in your selected project, all references and changes made in the existing integration will be overwritten and cannot be recovered.



Configure Project and Applications

You are creating an Integration based on a recipe

Extracts attendees from Concur and creates or updates the corresponding contacts in Salesforce.

| | |
|---|---|
| Type | OrchestrationIntegration |
| Created By | Software AG |
| Created On | 01/14/2019 02:30:12 PM IST |

Select Project * : Default

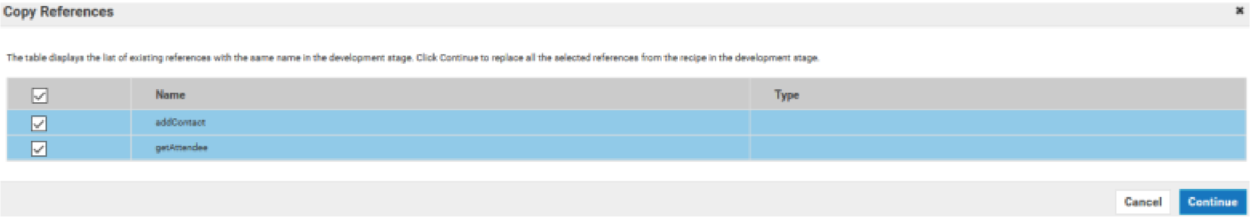
Select or Create Application Accounts

Concur * : Select or Type

Salesforce CRM * : Select or Type

6. Select the **Account** for each Application or create a new Account, and then click **Apply**.

7. If you have existing references (Reference Data, Document Types, and custom Operations) with the same name in the development stage, the **Copy References** window appears. By default, all references are selected in the **Copy References** page.



Copy References ✖

The table displays the list of existing references with the same name in the development stage. Click Continue to replace all the selected references from the recipe in the development stage.

| ☑ | Name | Type |
|---|---|---|
| ☑ | addContact | |
| ☑ | getAttendee | |

8. Deselect the references that you do not want to replace and then click **Continue** to replace or overwrite all the selected references from the recipe in the development stage.

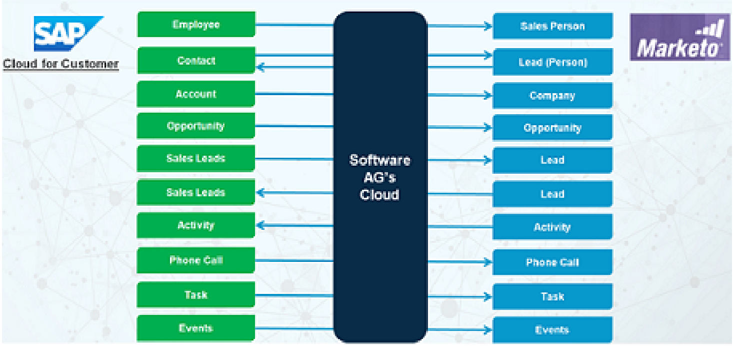
   The Integration is available in the selected project.

# Marketo Lead to SAP C4C Contact Data Sync

This section describes the recipe for the data sync between Marketo **Lead** business object and SAP Cloud for Customer **Contact** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract updated Marketo Contact records and update them in SAP Cloud for Customer.

## Custom Fields

Contact can be linked to some owner or Employee and Account, hence it is always recommended to sync the Employee and Account objects before syncing the Contacts to Marketo. This recipe syncs the Contact object in Marketo with the following custom fields.

| On SAP C4C | Comments |
|---|---|
| MKTOContactID | Custom field in SAP C4C for Contact records to hold the Marketo Contact records id. |
| ZLeadScore | Custom field having the lead score for a Contact. |
| ZAcquisitionProgram | Custom field having the acquisition program id for a Contact. |
| ZAcquisitionDate | Custom field having the acquisition date for a Contact. |
| ZSource | Custom field having the source of a Contact. |

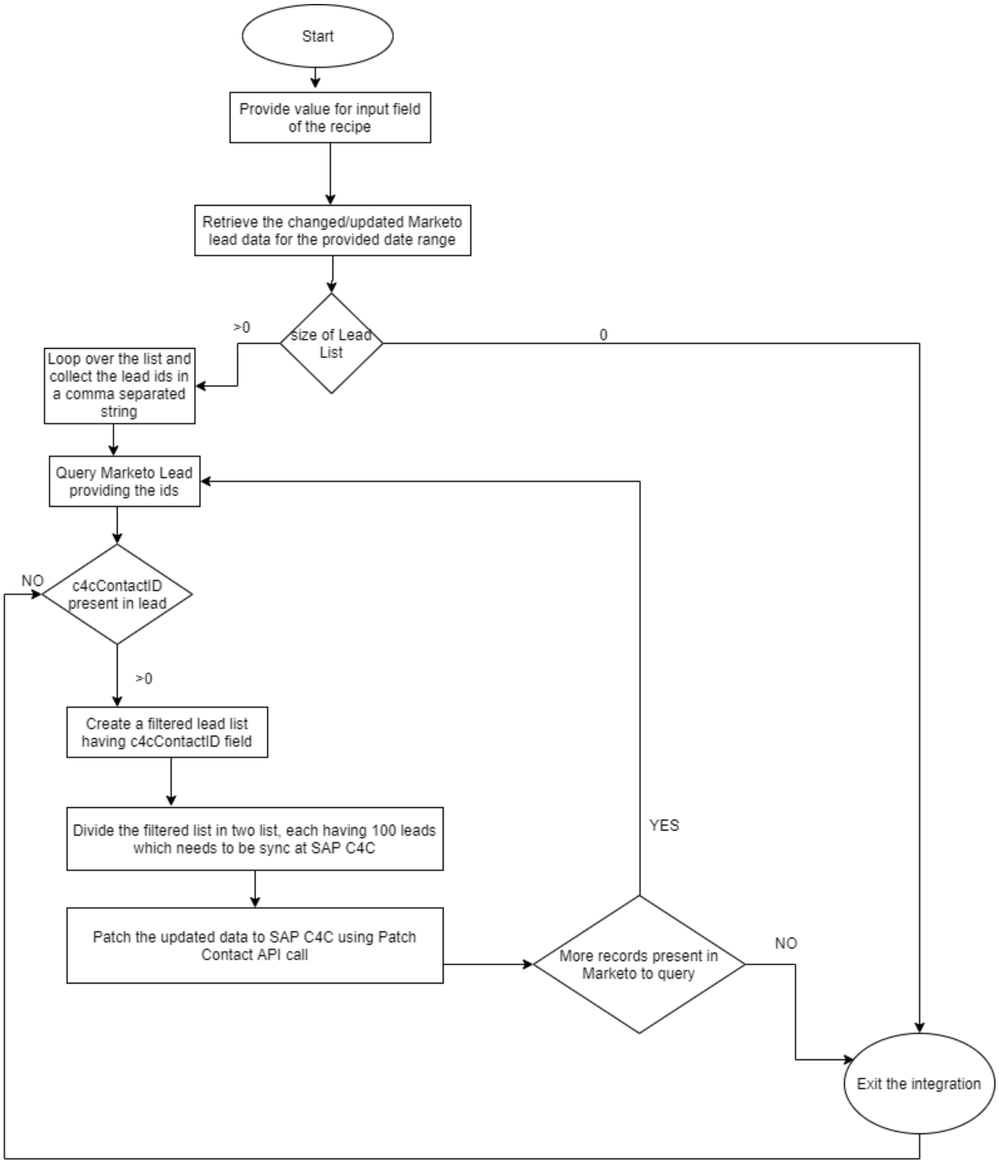
| On Marketo | Comments |
|---|---|
| contactHouseNumber | Custom field in Marketo to hold the house number for SAP C4C Contact. |
| c4CContactID | Custom field in Marketo for Contact records to hold the SAP C4C Contact records id. |
| c4CContactObjectID | Custom field in Marketo for Contact records to hold the SAP C4C Contact object id. |
| isDeleted | Custom field in Marketo to indicate that record has been obsoleted in SAP C4C. |

## Product Compatibility

The recipe is tested for the SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and also older versions of the product.

## Marketo to SAP C4C Contact Sync Flow

This is designed to run for ongoing sync. For the Contact records synced into Marketo from SAPC4C, if an update activity is performed, then the updated data is synced back to SAP C4C.


Start
Provide value for input field of the recipe
Retrieve the changed/updated Marketo lead data for the provided date range
size of Lead List
>0
0
Loop over the list and collect the lead ids in a comma separated string
Query Marketo Lead providing the ids
c4cContactID present in lead
NO
>0
Create a filtered lead list having c4cContactID field
Divide the filtered list in two list, each having 100 leads which needs to be sync at SAP C4C
Patch the updated data to SAP C4C using Patch Contact API call
More records present in Marketo to query
YES
NO
Exit the integration

## Operations Used

This recipe has a sub flow to handle the obsolete contact record in SAP C4C, that is, **updateDeletedSAPC4CContactsToMkto**.

| Operation Name | Application | Description |
|---|---|---|
| getMarketoLeadChangesActivity | Marketo | Queries Marketo Contact record ids which have been synced from SAP C4C and updated later in Marketo. |

| Operation Name | Application | Description |
|---|---|---|
| queryMarketoLeadRecords | Marketo | Queries Marketo Contact records data using ids. |
| patchSAPC4CMultipleContact | SAP C4C | Updates the Marketo Contact data to SAP C4C. |
| getContactBusinessAddressStateCode | SAP C4C | Gets list of State codes for Contact object defined in SAP C4C. |
| getContactBusiness<br>AddressCountryCode | SAP C4C | Gets list of Country codes for Contact object defined in SAP C4C. |
| getContactSalutation | SAP C4C | Gets list of Salutation codes for Contact<br>object defined in SAP C4C. |
| getContactDepartmentCodes | SAP C4C | Gets list of Department codes for Contact<br>object defined in SAP C4C. |

### Usage Notes

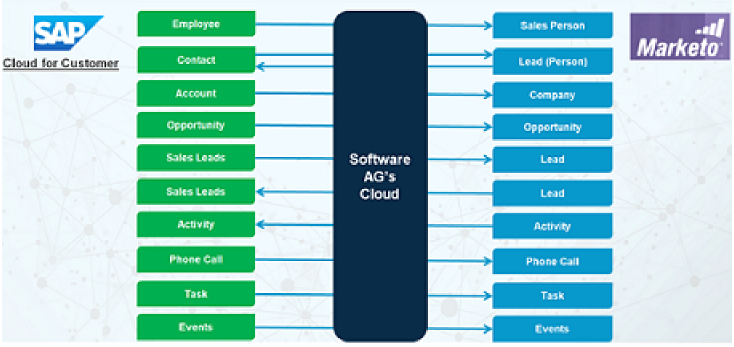
- In order to add new standard or custom fields to the integration, copy the required field and add it to the **filteredLeads** list while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the integration, remove the required field mapping and change or remove the field from the **filteredLeads** list while making the upsert API call to Marketo.

## Marketo to SAP C4C Activities Data Sync

This section describes the recipe for the data sync between Marketo **Activity** business object and SAP Cloud for Customer **Activity** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract Marketo Activity records and create those in SAP Cloud as Custom object records.

## Custom Business Object

Login to SAP C4C CRM and create one custom Business Object with the fields mentioned in the field mapping table.

## Custom OData Service

Once custom object is created, create a custom OData service in SAP C4C. Refer the above created custom object and then activate the service.

## Account Creation

Create a new Account in Integration Cloud providing the Server URL of the above custom OData service and credentials. Then use this Account details in the integration.

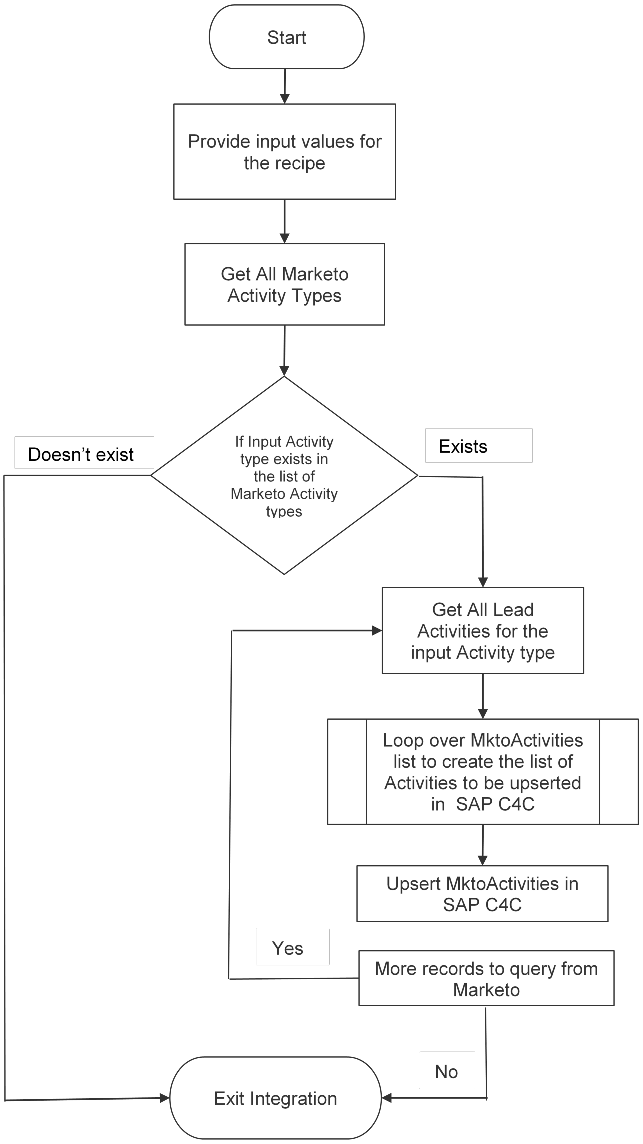
## Product Compatibility

The recipe is tested for the SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and also older versions of the product.

## Data Sync Flow

This integration "MarketoToSAPC4CActivitiesDataSync" is designed to run for ongoing sync. As part of the sync, for the newly created activities in Marketo corresponding records will be created as the custom objects in the SAP Cloud for Customer for the provided date range.

Below flow diagram gives representation of creating activities in SAP C4C from the lead activities in Marketo:


Start
Provide input values for the recipe
Get All Marketo Activity Types
If Input Activity type exists in the list of Marketo Activity types
Doesn't exist
Exists
Get All Lead Activities for the input Activity type
Loop over MktoActivities list to create the list of Activities to be upserted in SAP C4C
Upsert MktoActivities in SAP C4C
Yes
More records to query from Marketo
No
Exit Integration

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | ActivityType | 'Interesting Moment' and 'Filled out form' activity types are supported currently. |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern & timezone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

### Operations Used

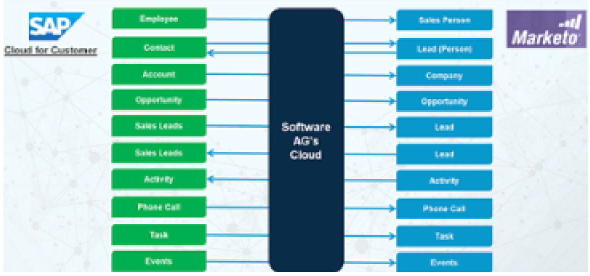
| SL.No | Operation Name | Operation Type | Application | Description |
|---|---|---|---|---|
| 1 | getPagingToken | Pre-defined | Marketo | Retrieves paging token to page through results, or retrieve data updated relative to a given data. |
| 2 | getActivityTypes | Pre-defined | Marketo | Returns a list of available activity types in the Marketo instance. |
| 3 | getLeadActivities | Pre-defined | Marketo | Returns a list of activities for the provided activity type. |
| 4 | createMultipleMktoActivity | Custom | SAP C4C | Creates multiple Marketo activity records in SAP C4C in the custom object. |

## Marketo to SAP C4C Leads Data Sync

This section describes the recipe for the data sync between Marketo **Lead** business object and SAP Cloud for Customer **Lead** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



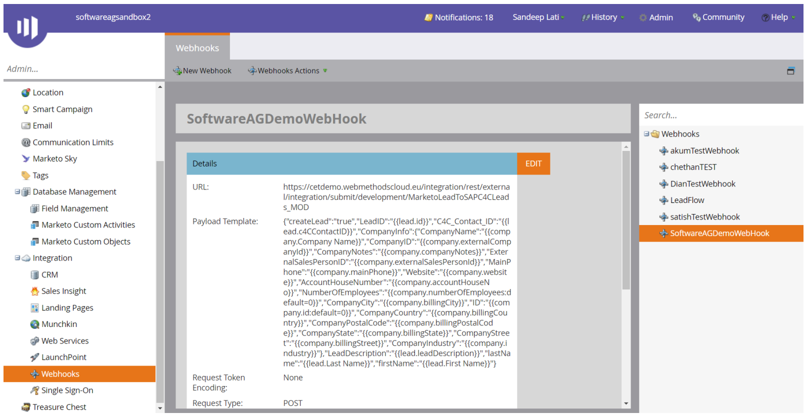
## Business Use Case

The business purpose of this integration is to extract a qualified Lead (based on the Lead Score) from Marketo and sync the Lead into SAP Cloud for Customer.

## Configuring Smart campaign and Webhooks in Marketo

1. Webhooks

   a. Once logged in to Marketo, go to **Admin->Webhooks** to see the below page:



SoftwareAGDemoWebHook

URL: https://cetdemo.webmethodscloud.eu/integration/rest/external/integration/submit/development/MarketoLeadToSAPC4CLeads_MOD

Payload Template: {"createLead":"true","LeadID":"{{lead.id}}","C4C_Contact_ID":"{{lead.c4CContactID}}","CompanyInfo":{"CompanyName":"{{company.Company Name}}","CompanyID":"{{company.externalCompanyId}}","CompanyNotes":"{{company.companyNotes}}","ExternalSalesPersonID":"{{company.externalSalesPersonId}}","MainPhone":"{{company.mainPhone}}","Website":"{{company.website}}","AccountHouseNumber":"{{company.accountHouseNo}}","NumberOfEmployees":"{{company.numberOfEmployees:default=0}}","CompanyCity":"{{company.billingCity}}","ID":"{{company.id:default=0}}","CompanyCountry":"{{company.billingCountry}}","CompanyPostalCode":"{{company.billingPostalCode}}","CompanyState":"{{company.billingState}}","CompanyStreet":"{{company.billingStreet}}","CompanyIndustry":"{{company.industry}}"},"LeadDescription":"{{lead.leadDescription}}","lastName":"{{lead.Last Name}}","firstName":"{{lead.First Name}}"}

Request Token Encoding: None

Request Type: POST

   b. Click **New Webhook** and fill the required fields.

   For URL, enter https://<<ICInstance>>/integration/rest/external/integration/submit/<<StageOfIntegration>>/<<IntegrationName>> and select **Request Type** as "POST".

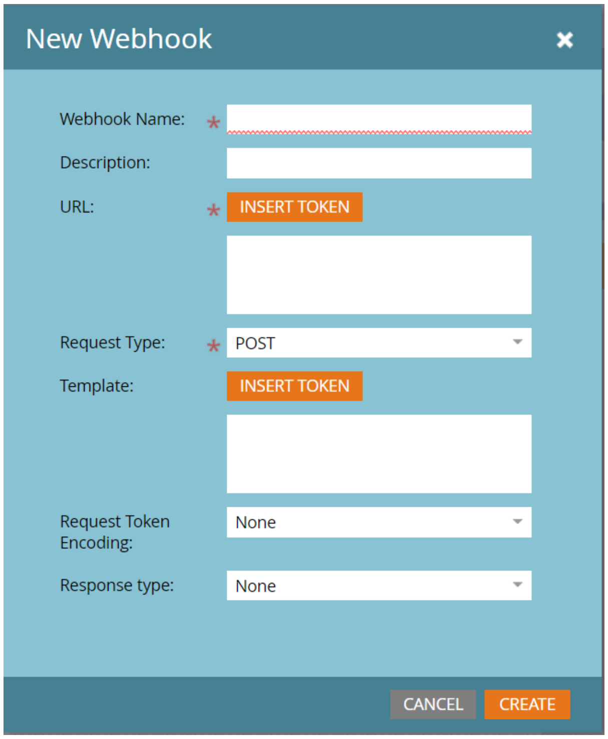
   c. Under **Template**, paste the following content:

```
{"createLead":"true","LeadID":"{{lead.id}}","C4C_Contact_ID":"
{{lead.c4CContactID}}","CompanyInfo":{"CompanyName":"{{company.Company Name}}",
"CompanyID":"{{company.externalCompanyId}}","CompanyNotes":"
{{company.companyNotes}}","ExternalSalesPersonID":"{{company.externalSalesPersonId}}"
,"MainPhone":"{{company.mainPhone}}","Website":"{{company.website}}",
"AccountHouseNumber":"{{company.accountHouseNo}}","NumberOfEmployees":"
{{company.numberOfEmployees:default=0}}","CompanyCity":"{{company.billingCity}}",
"ID":"{{company.id:default=0}}","CompanyCountry":"{{company.billingCountry}}",
"CompanyPostalCode":"{{company.billingPostalCode}}","CompanyState":"
{{company.billingState}}","CompanyStreet":"{{company.billingStreet}}",
"CompanyIndustry":"{{company.industry}}"},"LeadDescription":"{{lead.leadDescription}
}","lastName":"{{lead.Last Name}}","firstName":"{{lead.First Name}}"}
```

   d. Select **Request Token Encoding** as None, and select **Response type** as JSON.

   e. Click **Create**.

   The following image is a screen shot of the **Create Webhook** page:


New Webhook

Webhook Name: *

Description:

URL: *

Request Type: * POST

Template:

Request Token Encoding: None

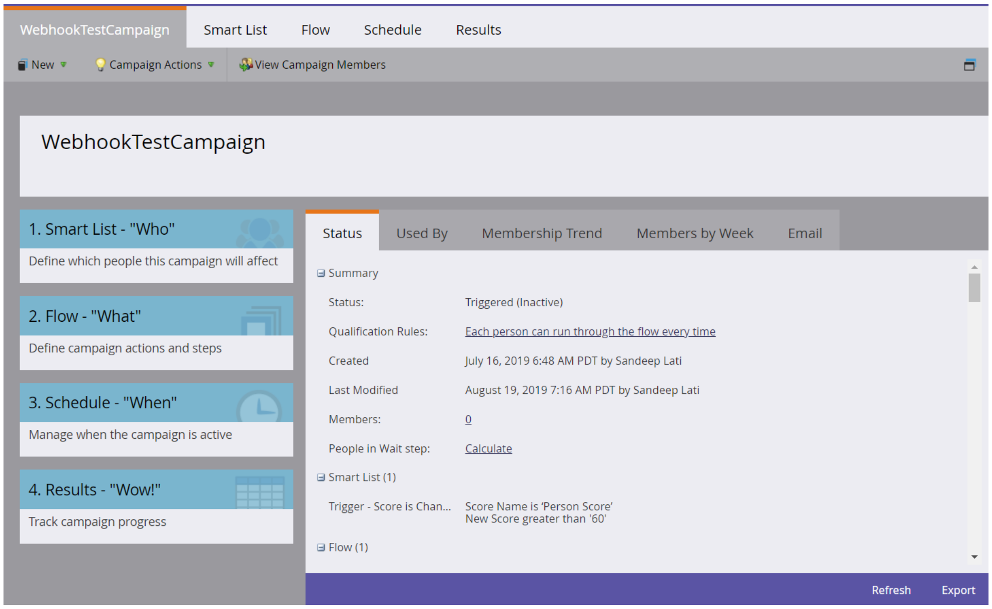
Response type: None

2. Smart Campaign

   a. Navigate to **Marketing Activities** and click **New Smart Campaign**.

   b. Select a folder and enter the **Name**, **Description**, and click **Create**.

      The following image displays the **WebhookTestCampaign** window:


WebhookTestCampaign

Summary

Status: Triggered (Inactive)

Qualification Rules: Each person can run through the flow every time

Created July 16, 2019 6:48 AM PDT by Sandeep Lati

Last Modified August 19, 2019 7:16 AM PDT by Sandeep Lati

Members: 0

People in Wait step: Calculate

Smart List (1)

Trigger - Score is Chan... Score Name is 'Person Score'<br>New Score greater than '60'

Flow (1)

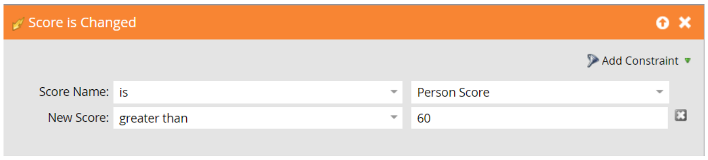
   c. Click **Smart List** tab and on the right pane select **Score is Changed** trigger, and drop it under the **Smart List** tab.

For the default **Score Name** constraint, select condition as **is** and enter the value as **Person Score**.

d. Click **Add Constraint menu**, to add a **New Score** constraint.

Under **New Score**, select the condition as **greater than** and the value as **60**.

A sample screenshot is shown below:



Score is Changed

Add Constraint

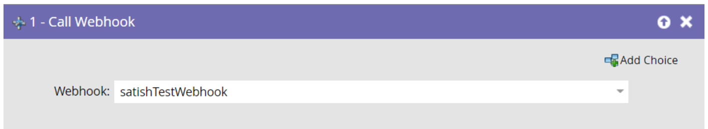
Score Name: is | Person Score

New Score: greater than | 60

e. Click on the **Flow** tab.

On the right side pane, select **Call Webhook** under **Integration** and drop it under the Flow tab.

f. Select the web hook created in **2.1.1**. A sample screen shot is shown below:

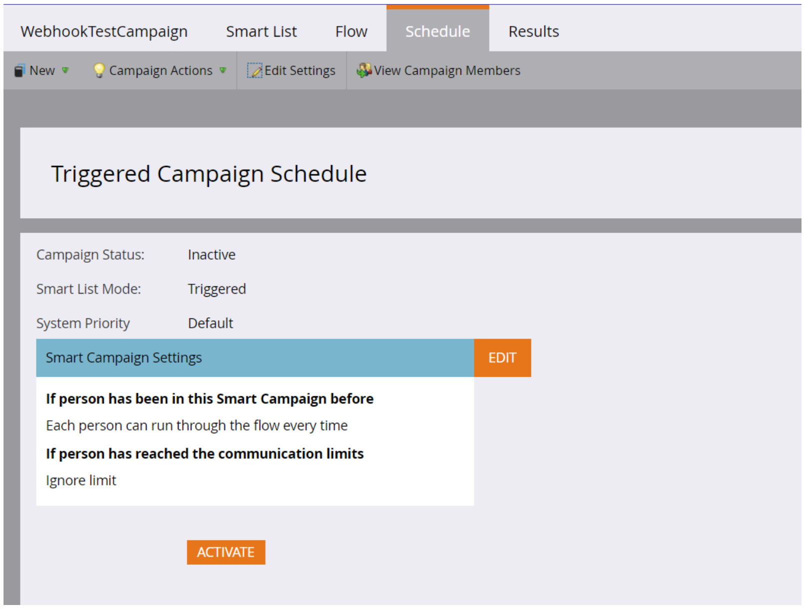


1 - Call Webhook

Add Choice

Webhook: satishTestWebhook

g. Click on the **Schedule** tab and click **Activate**:



Triggered Campaign Schedule

Campaign Status: Inactive

Smart List Mode: Triggered

System Priority Default

Smart Campaign Settings EDIT

**If person has been in this Smart Campaign before**

Each person can run through the flow every time

**If person has reached the communication limits**

Ignore limit

ACTIVATE

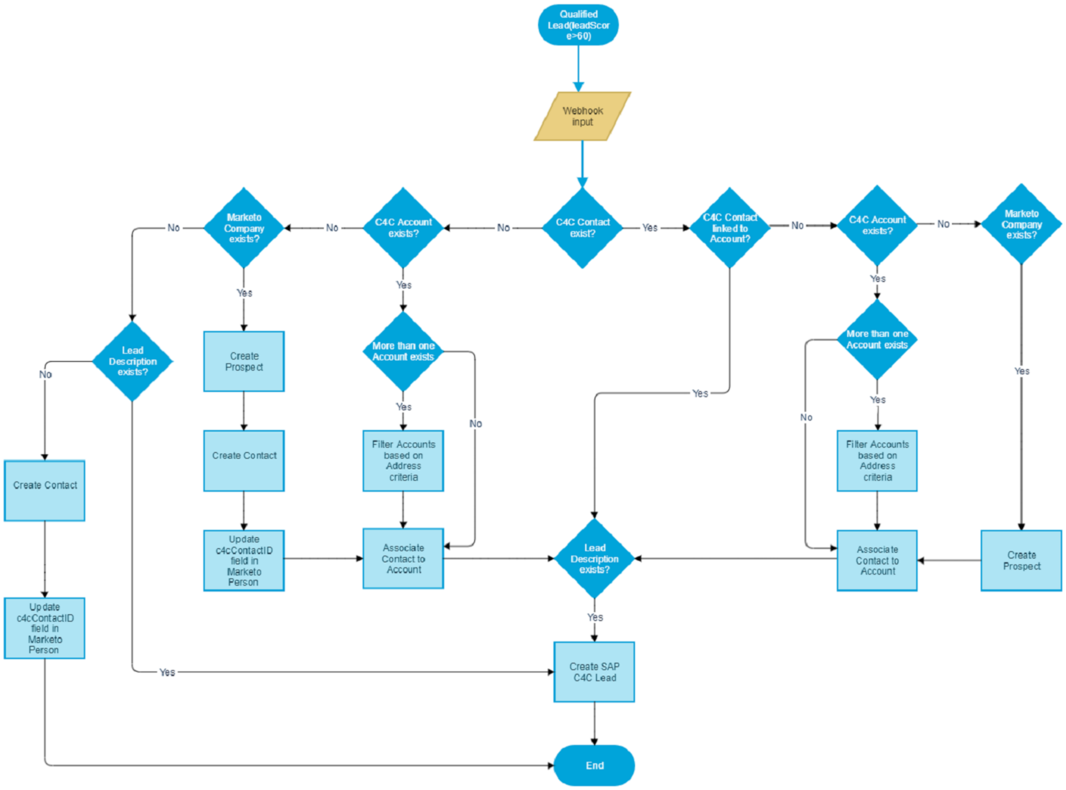
## Product Compatibility

The recipe is tested for the SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and also older versions of the product.

## Data Sync Flow

This integration "MarketoToSAPC4CLeadsDataSync" is designed to run for real-time sync using Webhooks and Smart Campaigns in Marketo. Based on the Lead input from Marketo, appropriate Accounts, Contacts, and Leads should be created in SAP C4C.

The following flow diagram represents the same:


Qualified Lead(leadScore>60)
Webhook input
C4C Contact exist?
No
Yes
C4C Account exists?
No
Yes
Marketo Company exists?
No
Yes
Lead Description exists?
No
Yes
Create Contact
Update c4cContactID field in Marketo Person
Create Prospect
Create Contact
Update c4cContactID field in Marketo Person
More than one Account exists
Yes
No
Filter Accounts based on Address criteria
Associate Contact to Account
C4C Contact linked to Account?
Yes
No
C4C Account exists?
Yes
No
Marketo Company exists?
Yes
More than one Account exists
Yes
No
Filter Accounts based on Address criteria
Associate Contact to Account
Create Prospect
Lead Description exists?
Yes
Create SAP C4C Lead
End

## Operations Used

1. SAP C4C

| SL.NO | Operation Name | Description | Query | Notes |
|---|---|---|---|---|
| 1 | queryAccounts_<br>MarketoLeadTo<br>SAPC4CLea | Query SAP<br>C4C Accounts | AccountID eq '%C4C_Account_ID%'<br>Name eq '%CompanyName%'<br>. | |
| 2 | queryContacts_<br>MarketoLeadTo | Query Contacts<br>from SAP C4C | ContactID eq '%C4C_Contact_ID%' | |

| SL.NO | Operation Name | Description | Query | Notes |
|---|---|---|---|---|
| | SAPC4CLea | | FirstName eq '%firstName%' and LastName eq '%lastName%'<br>LastName eq '%lastName%' | |
| 3 | createLeanLead_MarketoLeadToSAPC4CLe | Create a Lead in SAP C4C | NA | |
| 4 | patchContact_MarketoLeadToSAPC4CLead | Patch Contact to a Particular Account in SAP C4C | NA | |
| 5 | getCorporateAccountCountryCode | Get list of Country codes for Account object defined in SAP C4C | NA | Caching enabled |
| 6 | getCorporateAccountIndustrialSecCode | Get list of Industry Codes for Account object defined in SAP C4C | NA | Caching enabled |
| 7 | getCorporateAccountStateCode | Get list of State codes for Account object defined in SAP C4C | NA | Caching enabled |
| 8 | getContactBusinessAddressCountryCode | Get list of Country codes for Contact object defined in SAP C4C | NA | Caching enabled |
| 9 | getContactBusinessAddressStateCode | Get list of State codes for Contact object defined in SAP C4C | NA | Caching enabled |
| 10 | getContactStatusCode | Get list of Status codes for Contact object defined in SAP C4C | NA | Caching enabled |
| 11 | getContactSalutation | Get list of Salutation codes | NA | Caching |

| SL.NO | Operation Name | Description | Query | Notes |
|---|---|---|---|---|
| | | for Contact object defined in SAP C4C | | enabled |
| 12 | getContactDepartmentCodes | Get list of Department codes for Contact object defined in SAP C4C | NA | Caching enabled |
| 13 | createContact_MarketoLeadToSAPC4CLea | Create Contact in SAP C4C | NA | |
| 14 | createCorporateAccountHasContactPers | Link a Contact to a Department | NA | |
| 15 | createContactTextCollection | Link Person notes to a Contact | NA | |
| 16 | createAcc_MarketoLeadToSAPC4CLeads | Create a Prospect in SAP C4C | NA | |
| 17 | createCorporateAccountTextCollection | Link Company notes to an Account | NA | |
| 18 | createCorporateAccountTeam | Link a SalesPersonID to Owner in an Account | NA | |

### Marketo

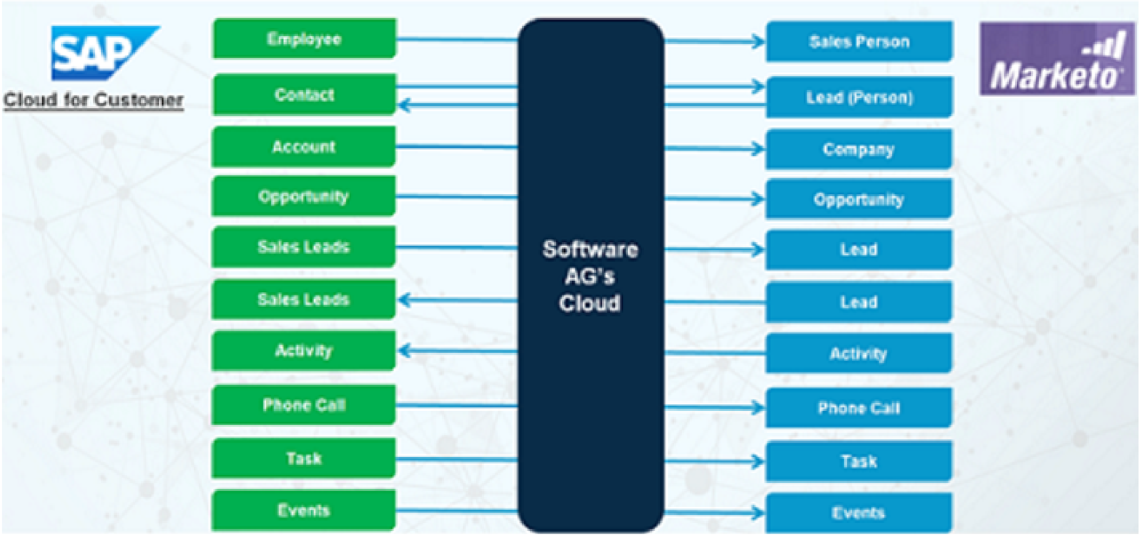
| SL.No | Parameter | Description | Query |
|---|---|---|---|
| 1 | getLeadById_MarketoLeadToSAPC4CLeads | Get Lead information for a matching ID. | NA |
| 2 | upsertLead_MarketoLeadToSAPC4CLeads | Update a Marketo Lead/Person with created Contact in SAP C4C | NA |

## SAP C4C to Marketo Appointment Activity Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Apointment** business object and Marketo custom object for Appointment records

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Appointment records and upserts them into Marketo Custom object for Appointment records.

## Custom Fields

When the Appointment record gets synced in Marketo, it needs to be created or updated with the below mentioned custom fields.

| On Marketo | Comments |
|---|---|
| c4CAppointmentID | Custom field needs to be created in the Marketo custom object for Appointment record. |

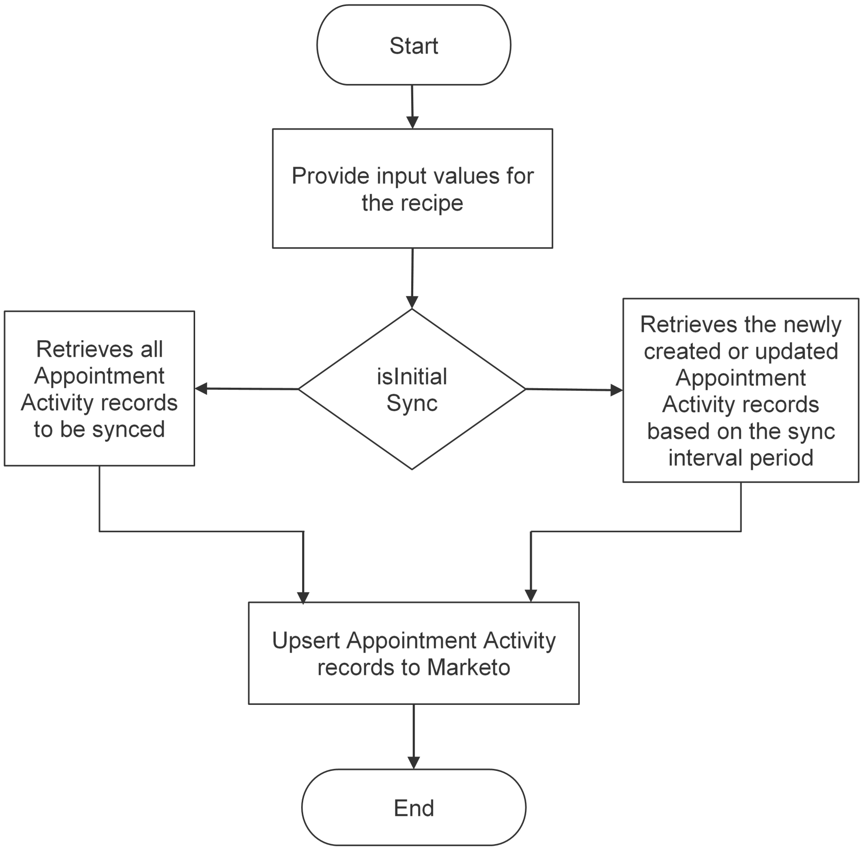
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Data Sync Flow

This integration "SAPC4CToMarketoActvtyAppoinDataSync" is designed to run for initial as well as ongoing sync. As part of the initial sync, all the Customer Appointment records present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated or newly created Customer Appointment records will be synced to Marketo for the provided date range.

Below flow diagram gives representation of creating or updating an appointment activity in Marketo from SAP C4C:


Start
Provide input values for the recipe
isInitial Sync
Retrieves all Appointment Activity records to be synced
Retrieves the newly created or updated Appointment Activity records based on the sync interval period
Upsert Appointment Activity records to Marketo
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync(true/false). |

| Sl.No | Parameter | Description |
|---|---|---|
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern & timezone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

### Operations Used

This recipe has a sub flow to handle the obsolete contact record in SAP C4C, that is, **updateDeletedSAPC4CContactsToMkto**.

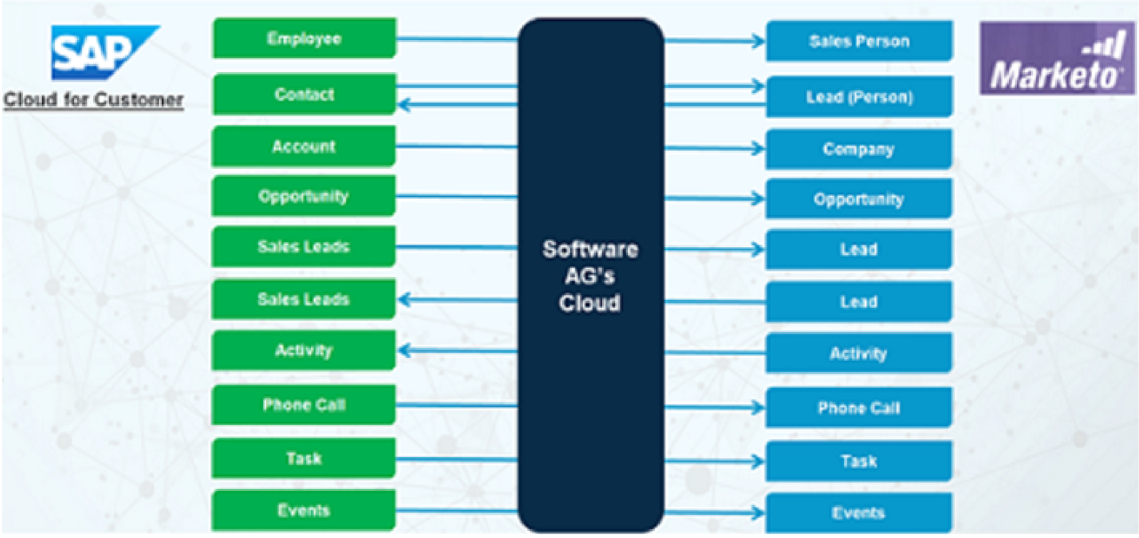
| Sl.No | Operations Used | Application | Description | Query |
|---|---|---|---|---|
| 1 | queryAppointments | C4C | Query SAP C4C Appointments. | LastChangeDateTime ge datetimeoffset'%fromDateString%' |
| 2 | upsertAppointme_ SAPC4CAct_to_MktoAct | Marketo | Upsert Appointments in Marketo. | NA |

## SAP C4C to Marketo CBO Lead Data Syn

This section describes the recipe for the data sync between SAP Cloud for Customer **Lead** business object and Marketo **Lead** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.

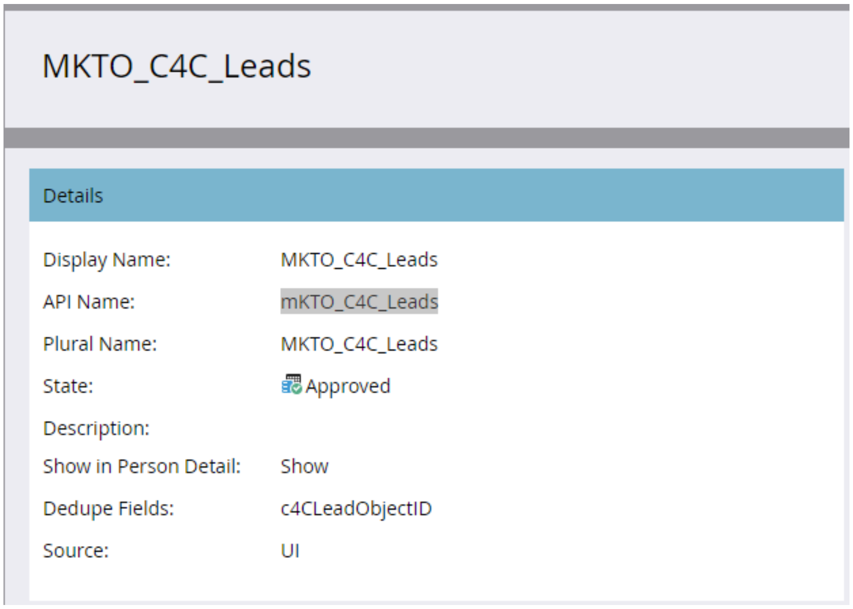


## Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Lead records and upsert those to Marketo as Custom object records.

## Custom Fields

Login to Marketo CRM and create one custom business object "mKTO_C4C_Leads" with the fields mentioned in the field mapping table. The custom field "c4CLeadObjectID" should be marked as Dedupe Field, else the delete scenario will not work.



MKTO_C4C_Leads

Details

| | |
|---|---|
| Display Name: | MKTO_C4C_Leads |
| API Name: | mKTO_C4C_Leads |
| Plural Name: | MKTO_C4C_Leads |
| State: | Approved |
| Description: | |
| Show in Person Detail: | Show |
| Dedupe Fields: | c4CLeadObjectID |
| Source: | UI |

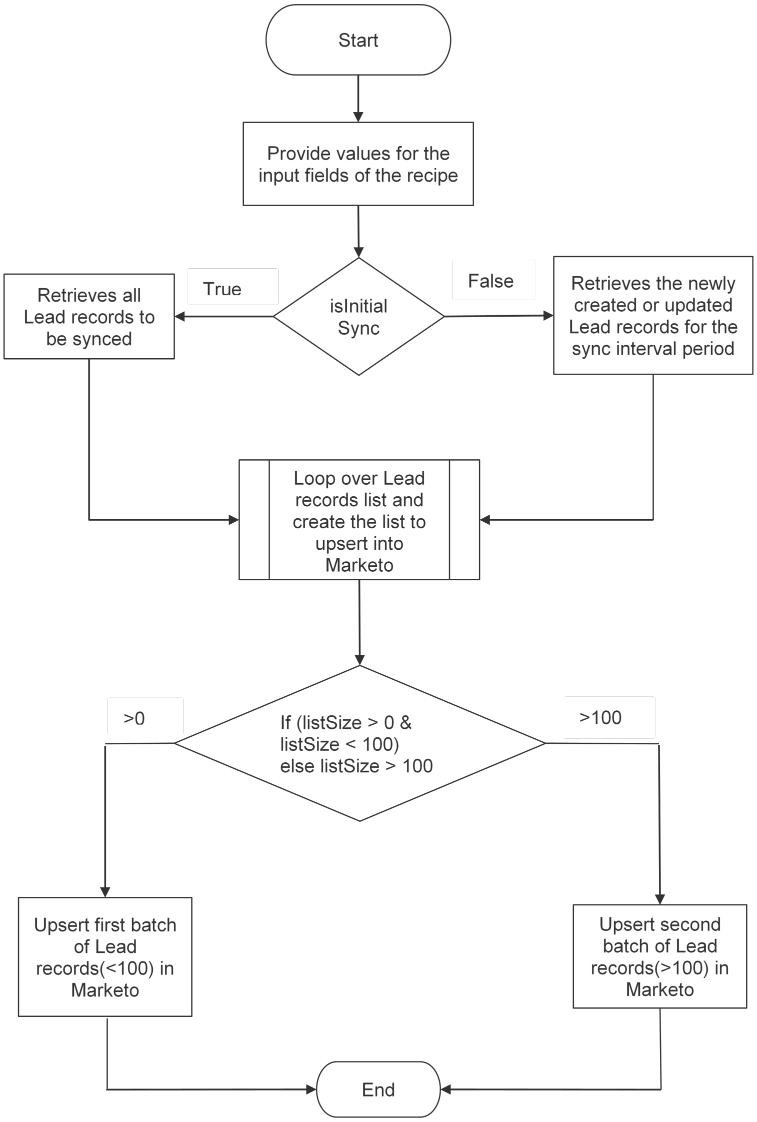
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Lead Sync Flow

This integration "SAPC4CToMarketoCBOLeadDataSync" can be run for initial as well as ongoing sync. As part of initial sync, all the Lead records having AccountID and ContactID present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated or newly created Lead records will be synced to Marketo for the provided date range.

Below flow diagram gives representation of creating a CBO Lead in Marketo from a Lead in SAP C4C:


Start
Provide values for the input fields of the recipe
isInitial Sync
True
False
Retrieves all Lead records to be synced
Retrieves the newly created or updated Lead records for the sync interval period
Loop over Lead records list and create the list to upsert into Marketo
If (listSize > 0 & listSize < 100) else listSize > 100
>0
>100
Upsert first batch of Lead records(<100) in Marketo
Upsert second batch of Lead records(>100) in Marketo
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync(true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

## Operations Used

| Sl.No | Operations Used | Application | Description |
|---|---|---|---|
| 1 | querySAPC4CLeads | SAP C4C | Queries SAP C4C Lead. |
| 2 | queryC4CContacts | SAP C4C | Used to get the Contact custom field **MKTOContactID** data to sync. |
| 3 | upsertMKToCBOLeads | Marketo | Upsert Leads in Marketo as custom object records. |

1. querySAPC4CLeads

   *Input Filter parameter*

| Sl.No | Condition | Filter to query records from SAP |
|---|---|---|
| 1 | IsInitialSync = True | No filter required. |
| 2 | IsInitialSync = False | LastChangeDateTime ge datetimeoffset'%fromDateString%' and LastChangeDateTime le datetimeoffset'%toDateString% |

2. queryC4CContacts

   Used to get the Contact custom field **MKTOContactID** data.

3. upsertMKToCBOLeads

   *Field Mapping*

| SAP Lead | | MKTO - mKTO_C4C_Leads | |
|---|---|---|---|
| ID | Standard | c4CLeadID | CustomField |
| ObjectID | Standard | c4CLeadObjectID | CustomField |
| Name | Standard | name | CustomField |
| AccountPartyID | Standard | accountPartyID | CustomField |
| ContactID | Standard | contactID | CustomField |
| QualificationLevelCode | Standard | qualificationLevelCode | CustomField |
| UserStatusCode | Standard | status | CustomField |
| ResultReasonCode | Standard | resultReasonCode | CustomField |

| **SAP Lead** | | **MKTO - mKTO_C4C_Leads** | |
|---|---|---|---|
| OriginTypeCode | Standard | orginTypeCode | CustomField |
| PriorityCode | Standard | priorityCode | CustomField |
| StartDate | Standard | startDate | CustomField |
| EndDate | Standard | endDate | CustomField |
| CampaignID | Standard | campaignID | CustomField |
| GroupCode | Standard | groupCode | CustomField |
| OwnerPartyID | Standard | ownerPartyID | CustomField |
| Note | Standard | note | CustomField |
| Contact - MKTOContactID | CustomField | mKTOLeadID | CustomField |
| LeadBusinessTransactionDocument - ID | Standard | referenceDocumentID | CustomField |

### Usage Notes

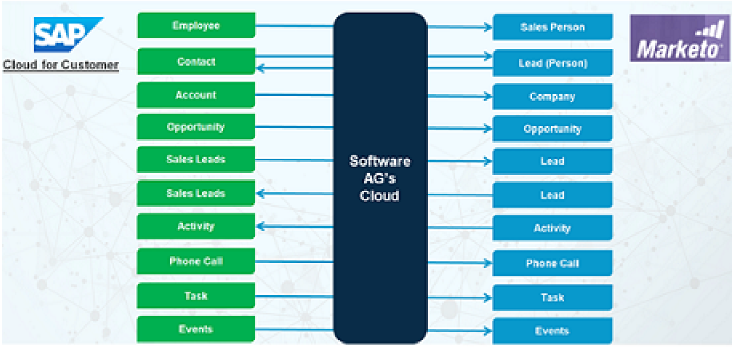
- In order to add new standard or custom fields to the integration, copy the required field and add it to the C4CLeadOutput document while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the integration, remove the required field mapping, and change or remove the field from C4CLeadOutput document while making the upsert API call to Marketo.
- Custom field “c4CLeadObjectID” should be created as Dedupe Field.
- If the Lead is associated to just an account, contact will not be synced to Marketo.

## SAP C4C to Marketo Event Notification for Delete Sync

This section describes the master recipe consisting of SalesPerson, Company, Lead and Opportunity recipes for the data sync between SAP Cloud for Customer business objects and Marketo business objects respectively.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.

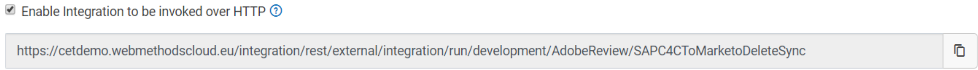


## Business Use Case

The business purpose of this integration is to delete the records from Marketo for the corresponding records deleted from SAP Cloud for Customer.

## Configure Event Notification in SAPC4C

Enable the integration to be invoked over HTTP and copy the REST service URL as follows:



☑ Enable Integration to be invoked over HTTP

https://cetdemo.webmethodscloud.eu/integration/rest/external/integration/run/development/AdobeReview/SAPC4CToMarketoDeleteSync

Login to SAPC4C and navigate to **Administrator** -> **General Settings** -> **Event Notification**.

Configure Delete event notification using the REST service URL copied for the integration, add basic authentication details and then activate it.

## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Delete Sync Flow

This integration "SAPC4CToMarketoDeleteSync" is designed to run in the background triggered by event notification from SAP C4C and no manual execution required. The delete sync is performed for the deletion of records of the following business objects Opportunity, CBO Lead, Phone Call, Task, Appointment activities.

This recipe is a wrapper and consisting of the below five sub-integrations:

- deleteCBOLeadEventNotification
- deletePhoneCallEventNotification
- deleteTaskEventNotification
- deleteAppointmentEventNotification

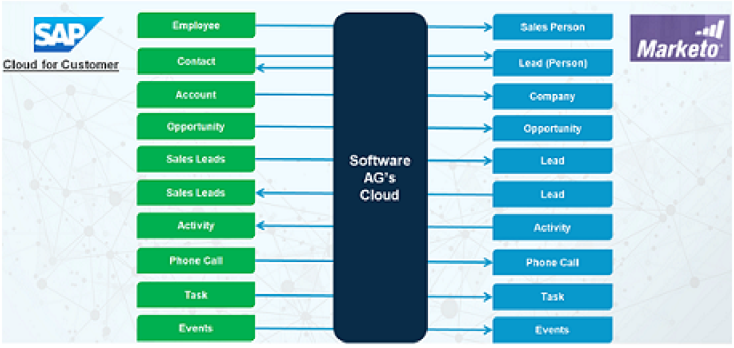
- deleteOpportunityEventNotification

## SAP C4C to Marketo Master Data Sync

This section describes the master recipe consisting of SalesPerson, Company, Lead, and Opportunity recipes for the data sync between SAP Cloud for Customer business objects and Marketo business objects respectively.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



### Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Employee, Account, Contact, and Opportunity records and upserts those to Marketo. For every SAP contact, a corresponding Lead / person record is created in Marketo which then will have the respective 1) SAP Employee synched as salesperson 2) Account as Company and 3) Opportunity as Opportunity of the Marketo Lead / person.

### Business Object Relationship

One or all of the 4 types of records (Employee, Account, Contact, and Opportunity) needs to be created or updated in the source SAP C4C systems.

In the real world scenarios an explicit relationship is maintained between these records or business objects, that is, an Account will have an Employee associated with it as Owner. Further, the Account will have one more Contact associated with it. In other words, every contact is part of some account (company). Lastly, one or more opportunities for the given contact is created with a distinct role/authority within the respective contact.

### Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

### Master Sync Flow

This Master integration "SAPC4CToMarketoMasterDataSync" is a wrapper consisting of four main sub-integrations which were built to sync records of particular business objects only.

The four integrations are as follows:

- SAPC4CToMarketoSalesPersonDataSync
- SAPC4CToMarktoCompanyDataSync
- SAPC4CContactToMarketoLeadDataSync
- SAPC4CToMarketoOpportunityDataSync

### Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync(true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

### Usage Notes

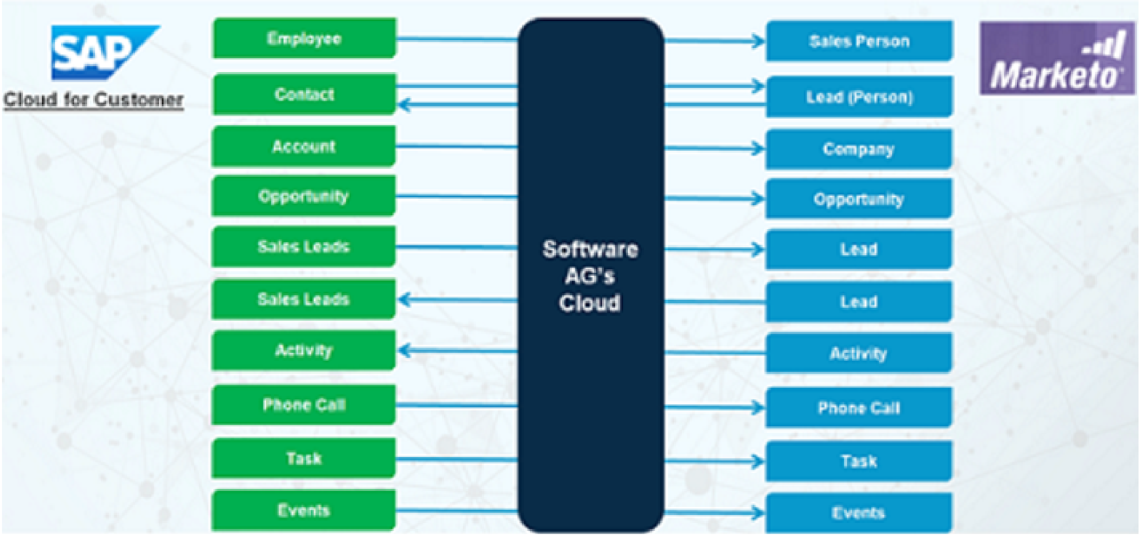
See the respective usage notes of the four main sub-integrations as a part of the Master Sync integration.

## SAP C4C to Marketo Phone Activity Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Phone Call** business object and Marketo **Phone Call** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Phone Call records and upsert them into Marketo Custom object for Phone Call records.

## Custom Fields

When the Phone call record gets synced in Marketo, it need to be created or updated with the below mentioned custom fields.

| On Marketo | Comments |
|---|---|
| c4CPhoneCallID | Custom field needs to be created in the Marketo custom object for Phone call record. |

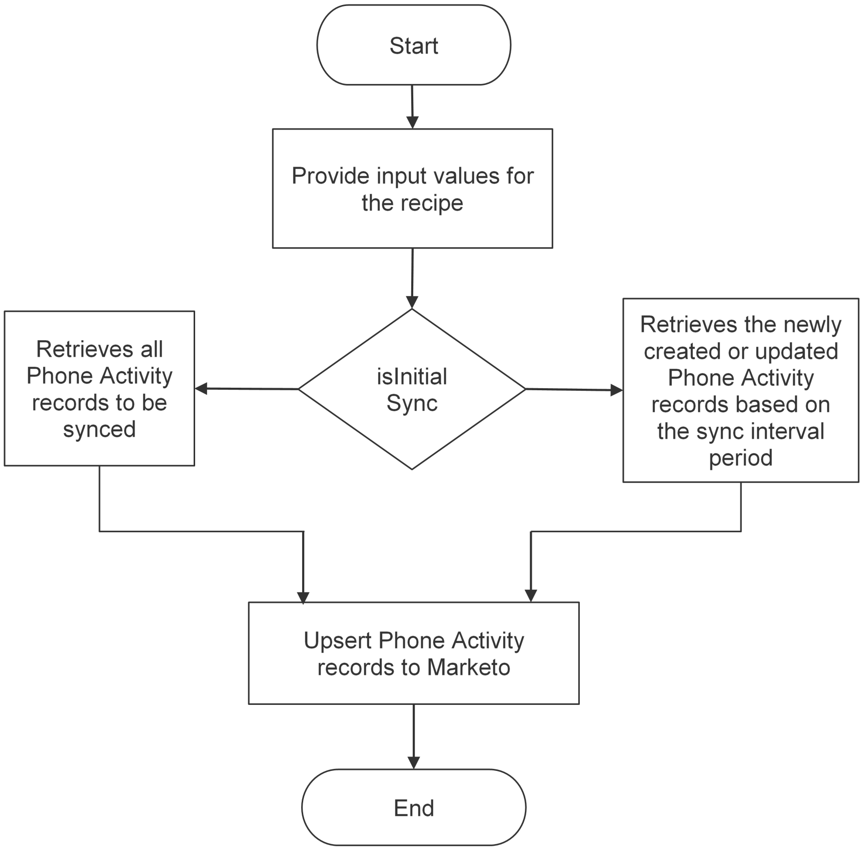
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Data Sync Flow

This integration "SAPC4CToMarketoActvtyPhoneDataSync" is designed to run for initial as well as ongoing sync. As part of the initial sync, all the Customer Phone Calls records present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated or newly created Customer Phone Calls records will be synced to Marketo for the provided date range.

Below flow diagram gives representation of creating or updating a phone activity in Marketo from SAP C4C:


Start
Provide input values for the recipe
isInitial Sync
Retrieves all Phone Activity records to be synced
Retrieves the newly created or updated Phone Activity records based on the sync interval period
Upsert Phone Activity records to Marketo
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync (true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern & time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

### Operations Used

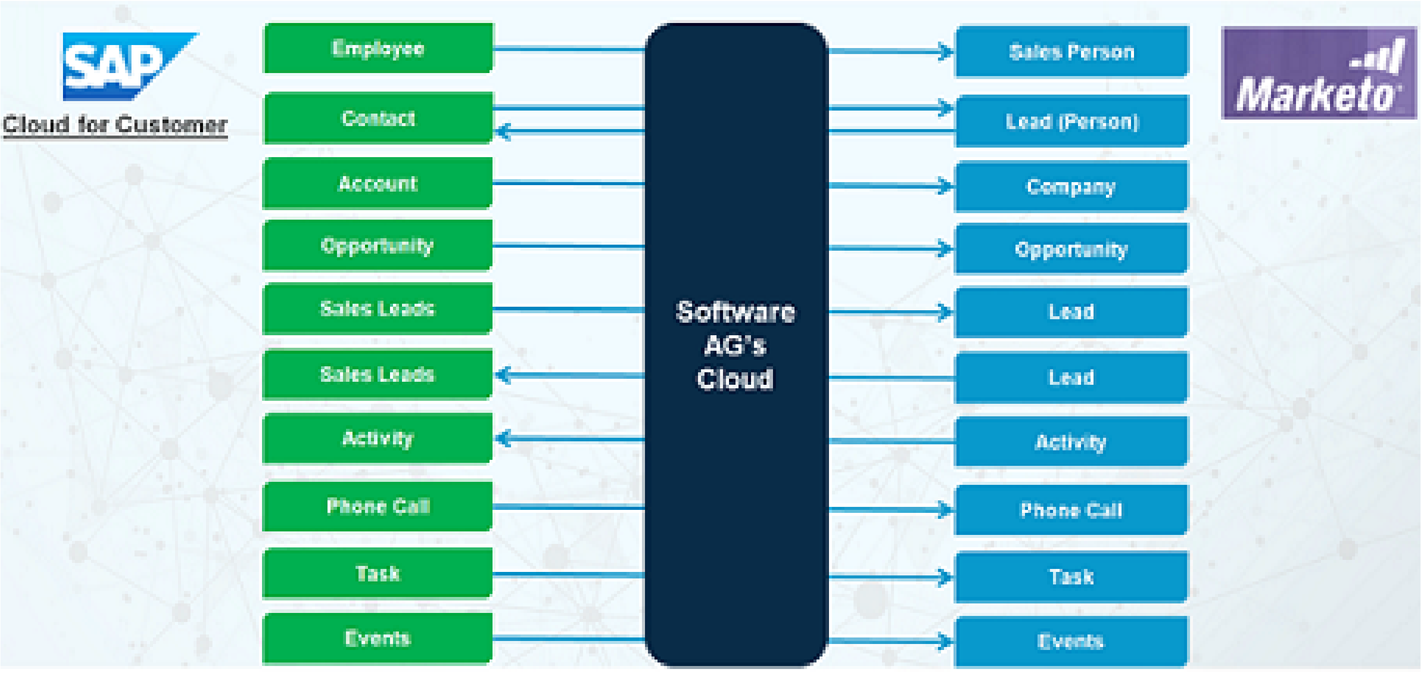
| Sl.No | Operations Used | Application | Description | Query |
|---|---|---|---|---|
| 1 | queryPhoneCalls | C4C | Query SAP C4C Phone calls. | LastChangeDateTime ge datetimeoffset'%fromDateString%' |
| 2 | upsertPhnCal_SAPC4CAct_to_MktoAct_Ph | Marketo | Upsert phone calls in Marketo. | NA |

## SAP C4C to Marketo Task Activity Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Task** business object and Marketo **Task** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



### Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Tasks records and upserts them into Marketo Custom object for Task records.

### Custom Fields

When the Task record gets synced in Marketo, it needs to be created or updated with the below mentioned custom fields.

| On SAP C4C | Comments |
|---|---|
| c4CTaskID | Custom field needs to be created in the Marketo custom object for Task record. |

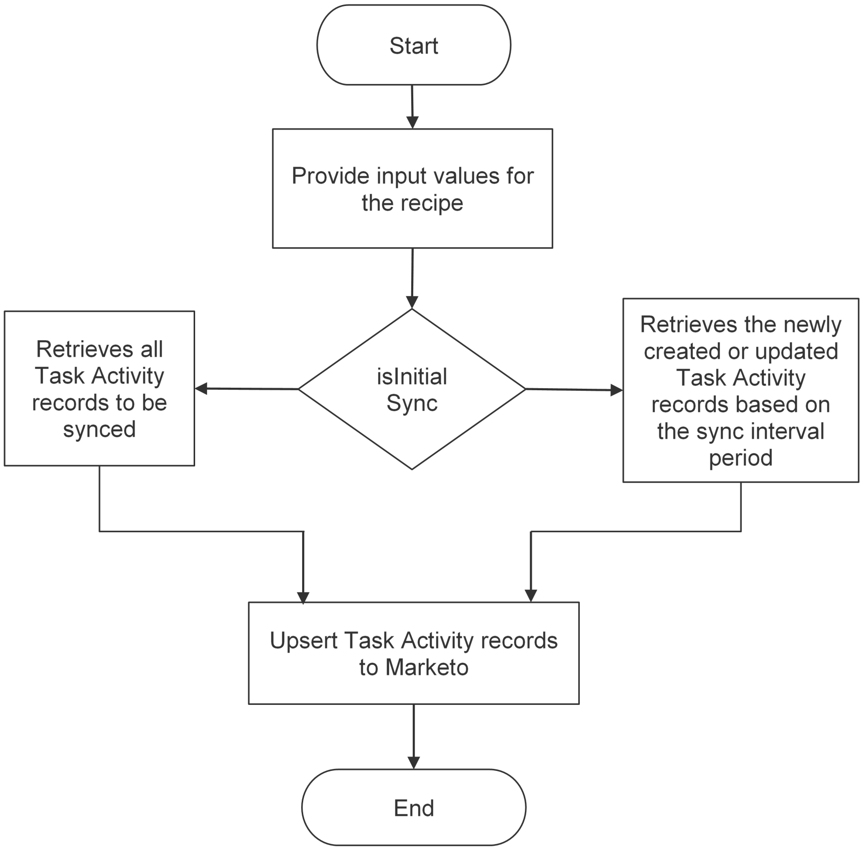
### Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

### Data Sync Flow

This integration "SAPC4CToMarketoActvtyTasksDataSync" is designed to run for initial as well as ongoing sync. As part of the initial sync, all the Customer Tasks records present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated or newly created Customer Tasks records will be synced to Marketo for the provided date range.

Below flow diagram gives representation of creating or updating a task activity in Marketo from SAP C4C:


Start
Provide input values for the recipe
isInitial Sync
Retrieves all Task Activity records to be synced
Retrieves the newly created or updated Task Activity records based on the sync interval period
Upsert Task Activity records to Marketo
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync(true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

## Operations Used

| Sl.No | Operation Name | Application | Description | Query |
|---|---|---|---|---|
| 1 | queryTasks | C4C | Query SAP C4C Tasks | LastChangeDateTime ge datetimeoffset'%fromDateString%' |

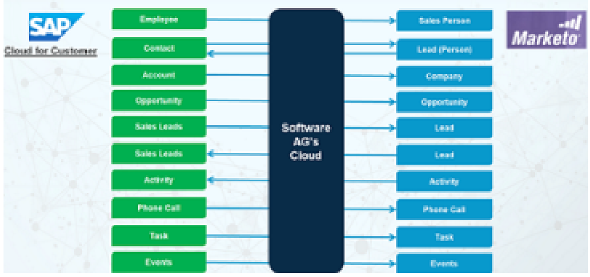
| Sl.No | Operation Name | Application | Description | Query |
|---|---|---|---|---|
| 2 | upsertTasks_SAPC4CAct_ to_MktoAct_Tas | Marketo | Upsert tasks in Marketo | NA |

## SAP C4C Account to Marketo Company Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Account** business object and Marketo **Company** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



### Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Account records and upsert those to Marketo as Company records.

### Custom Fields

Account can be linked to some owner or Employee, hence it is recommended to sync Employee objects before syncing the Accounts to Marketo.

The Account object need to be created or upserted in SAP C4C with the below mentioned custom fields.

Similarly when the Account object gets synced in Marketo, it need to be created or updated with the below mentioned custom fields.

| On SAP C4C | Comments |
|---|---|
| MKTOCompanyID | Custom field in SAP C4C for Account records in SAP C4C to hold the Marketo Company records id. |

| On SAP C4C | Comments |
|---|---|
| NumberofEmployees | Custom field having the number of employees for an Account. |

| On Marketo | Comments |
|---|---|
| AccountHouseNumber | Custom field in Marketo to hold the house number for the SAP C4C Account. |

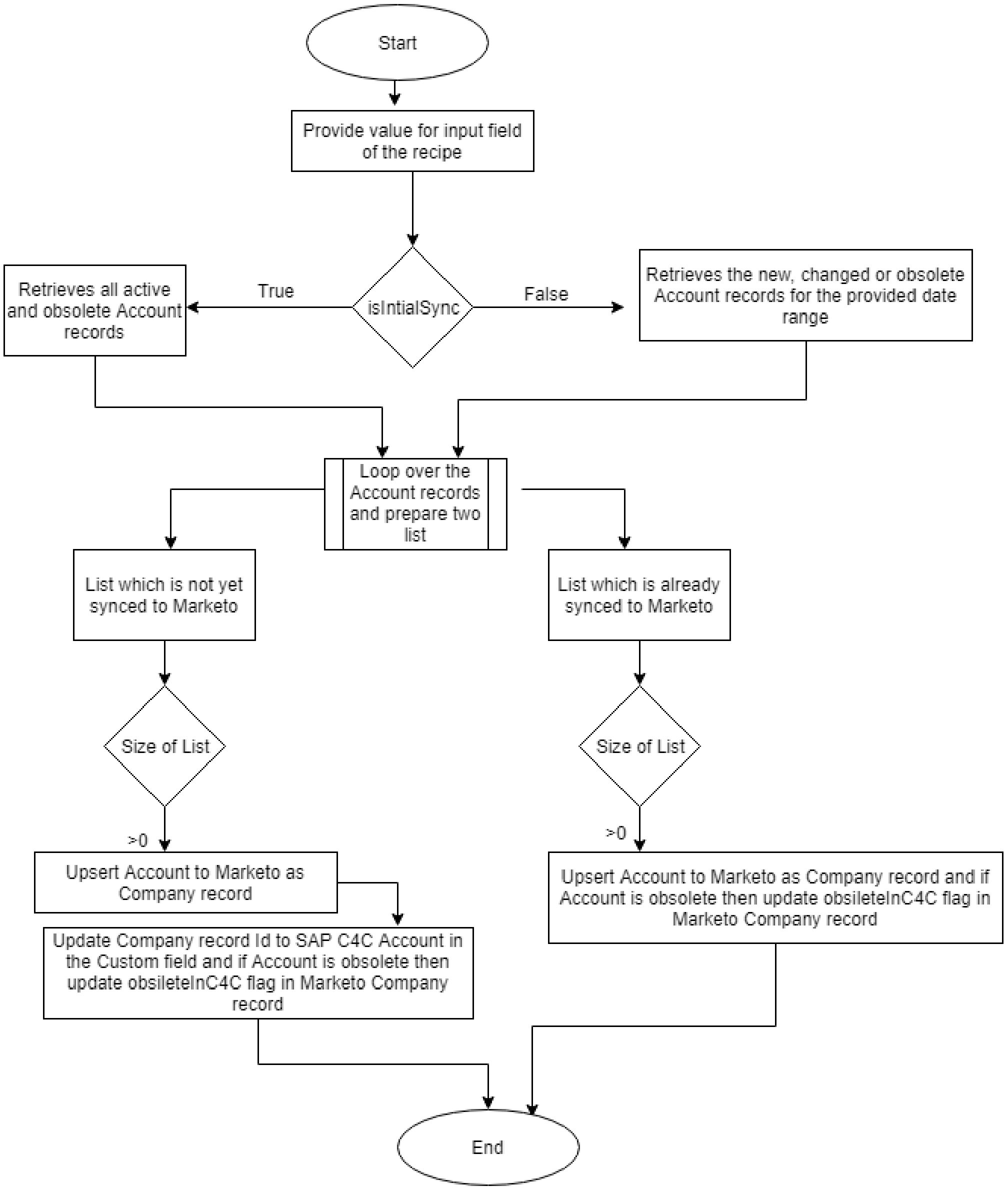
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Account Sync Flow

This integration "SAPC4CToMarktoCompanyDataSync" is designed to run for initial as well as ongoing sync. As part of this initial sync, all the accounts linked to Employees present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the newly created or updated account records will be synced to Marketo for the provided date range. Once the record is successfully created in Marketo, it syncs back the Marketo Company ID in the corresponding SAP C4C Account record (Custom Field: MKTOCompanyID).

The following diagram represents creating a Company in Marketo from an Account in SAP C4C:


Start
Provide value for input field of the recipe
isIntialSync
True
False
Retrieves all active and obsolete Account records
Retrieves the new, changed or obsolete Account records for the provided date range
Loop over the Account records and prepare two list
List which is not yet synced to Marketo
List which is already synced to Marketo
Size of List
Size of List
>0
>0
Upsert Account to Marketo as Company record
Update Company record Id to SAP C4C Account in the Custom field and if Account is obsolete then update obsileteInC4C flag in Marketo Company record
Upsert Account to Marketo as Company record and if Account is obsolete then update obsileteInC4C flag in Marketo Company record
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync (true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

## Operations Used

This recipe has a sub-flow to handle the obsolete contact record in SAP C4C, that is, **updateDeletedSAPC4CContactsToMkto**.

| Sl.No | Operations Used | Application | Description |
|---|---|---|---|
| 1 | querySAPC4CAccount | SAP C4C | Queries SAP C4C Accounts. |
| 2 | upsertMarketoLeads | Marketo | Upserts Account in Marketo as Company records and if Account is obsolete in SAP C4C then it updates the obsoleteinC4C flag to the Company record in Marketo. |
| 3 | patchMultipleAccount | SAP C4C | Once records are successfully created in Marketo, it updates the Marketo Company ID in the corresponding SAP C4C Account record in the **MKTOCompanyID** custom field. |

1. querySAPC4CAccount

   *Input Filter parameter*

| Sl.No | Condition | Filter to query records from SAP |
|---|---|---|
| 1 | IsInitialSync = True | No filter required. |
| 2 | IsInitialSync = False | ChangedOn ge datetimeoffset'%fromDateString%'<br>ChangedOn le datetimeoffset'%toDateString%' |

2. upsertMarketoCompanies

   *Field Mapping*

| SAP Account | | MKTO-COMPANY | |
|---|---|---|---|
| AccountID | Standard | externalCompanyId | Standard |
| City | Standard | billingCity | Standard |
| CountryCodeText | Standard | billingCountry | Standard |
| StreetPostalCode | Standard | billingPostalCode | Standard |
| StateCodeText | Standard | billingState | Standard |
| Street | Standard | billingStreet | Standard |
| Name | Standard | company | Standard |

| SAP Account | | MKTO-COMPANY | |
|---|---|---|---|
| ContactText | Standard | companyNotes | Standard |
| EmployeeID | Standard | externalSalesPersonId | Standard |
| IndustrialSectorCodeText | Standard | industry | Standard |
| Phone | Standard | mainPhone | Standard |
| WebSite | Standard | website | Standard |
| HouseNumber | Standard | accountHouseNo | CustomField |
| NumberofEmployees | Standard | state | Standard |

3. patchMultipleAccount

   Once Account records are successfully created in Marketo as Company records, update Marketo Company ID to the corresponding SAP C4C Account record in the **MKTOCompanyID** custom field.

| MKTO - PERSON | | SAP - ACCOUNT | |
|---|---|---|---|
| ID | Standard | MKTOCompanyID | CustomField |
| Name | Standard | Name | Standard |
| RoleCode | Standard | RoleCode | Standard |
| ObjectID | Standard | ObjectID | Standard |

### Usage Notes

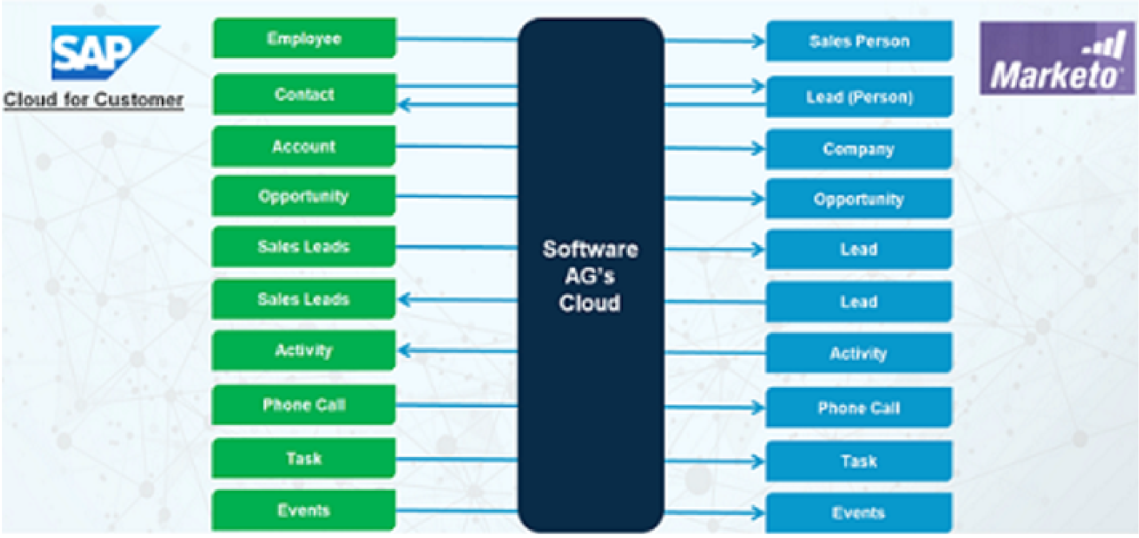
- In order to add new standard or custom fields to the integration, copy the required field and add it to accList_NullMktoID, accList_WithMktoID, delAccList_WithMktoID, and the delAccList_NullMktoID list while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the integration, remove the required field mapping and change or remove the field from accList_NullMktoID, accList_WithMktoID, delAccList_WithMktoID, and delAccList_NullMktoID list while making the upsert API call to Marketo.

## SAP C4C Contact to Marketo Lead Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Contact** business object and Marketo **Lead** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Contact records and upsert them to Marketo as Person records.

## Custom Fields

Contact can be linked to some owner or Employee and Account, hence it is always recommended to sync Employee and Account objects before syncing the Contacts to Marketo.

This recipe syncs Contact object in Marketo with the below mentioned custom fields.

| On SAP C4C | Comments |
|---|---|
| MKTOContactID | Custom field in SAP C4C for Contact records to hold the Marketo Contact records id. |
| ZLeadScore | Custom field having the lead score for a Contact. |
| ZAcquisitionProgram | Custom field having the acquisition program id for a Contact. |
| ZAcquisitionDate | Custom field having the acquisition date for a Contact. |
| ZSource | Custom field having the source of a Contact. |

| On Marketo | Comments |
|---|---|
| contactHouseNumber | Custom field in Marketo to hold the house number for SAP C4C Contact. |
| o c4CContactID | Custom field in Marketo for Contact records to hold the SAP C4C Contact records id. |
| c4CContactObjectID | Custom field in Marketo for Contact records to hold the SAP C4C Contact object id. |
| isDeleted | Custom field in Marketo to indicate that record has been obsoleted in SAP C4C. |

## Required recipes to run before Opportunity

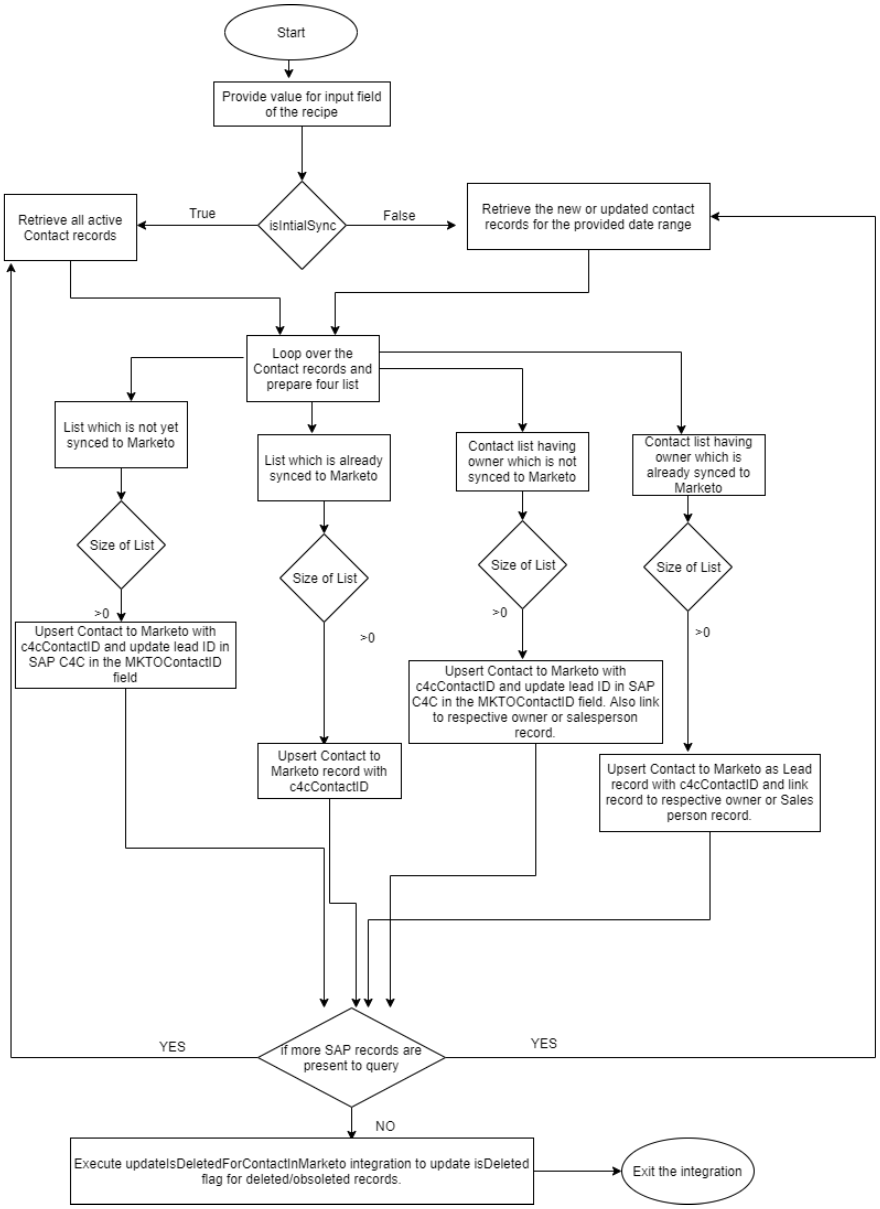
| SAP C4C | Marketo | Required Fields | Comments |
|---|---|---|---|
| Employee | SalesPerson | externalSalesPersonID | If employee is synched in Marketo, then only the Opportunity owner will be updated at Marketo from SAP C4C. |
| Account | Company | externalCompanyID (SAP ProspectPartyID) | If account is synced in Marketo, then only the Opportunity company will be updated at Marketo from SAP C4C. |
| Contact | Lead | c4CContactID (SAP PrimaryContactPartyID) | If contact is synced in Marketo, then only the Opportunity will be upserted at Marketo from SAP C4C. |

## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Contact Sync Flow

This integration "SAPC4CContactToMarketoLeadDataSync" is designed to run for initial as well as ongoing sync. As part of the initial sync, all the active and obsolete Contact records present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated and newly created and obsolete Contact records will be synced to Marketo for the provided date range. Once the record is successfully created in Marketo, it syncs back the Marketo Contact ID in the corresponding SAP C4C Contact record. (Custom Field: MKTOContactID).


Start
Provide value for input field of the recipe
isIntialSync
True
False
Retrieve all active Contact records
Retrieve the new or updated contact records for the provided date range
Loop over the Contact records and prepare four list
List which is not yet synced to Marketo
List which is already synced to Marketo
Contact list having owner which is not synced to Marketo
Contact list having owner which is already synced to Marketo
Size of List
Size of List
Size of List
Size of List
>0
>0
>0
>0
Upsert Contact to Marketo with c4cContactID and update lead ID in SAP C4C in the MKTOContactID field
Upsert Contact to Marketo record with c4cContactID
Upsert Contact to Marketo with c4cContactID and update lead ID in SAP C4C in the MKTOContactID field. Also link to respective owner or salesperson record.
Upsert Contact to Marketo as Lead record with c4cContactID and link record to respective owner or Sales person record.
YES
if more SAP records are present to query
YES
NO
Execute updateIsDeletedForContactInMarketo integration to update isDeleted flag for deleted/obsoleted records.
Exit the integration

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync(true/false). |
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

## Operations Used

This recipe has a sub flow to handle the obsolete contact record in SAP C4C, that is, **updateDeletedSAPC4CContactsToMkto**.

| Sl.No | Operations Used | Application | Description |
|---|---|---|---|
| 1 | querySAPC4CContact | SAP C4C | Queries SAP C4C Contacts. |
| 2 | upsertMarketoLeads | Marketo | Upserts Contact in Marketo and if Contact is obsolete in SAP C4C, it updates isDeleted flag to Contact record in Marketo. |
| 3 | patchSAPC4CMultipleContact | SAP C4C | Once record is successfully created in Marketo, it updates the Marketo Contact ID in the corresponding SAP C4C Contact record in the **MKTOContactID** custom field. |
| 4 | readMultipleEmployees | SAP C4C | Retrieves the Employee/Owner id for the provided UUID. |

1. querySAPC4CContact

   *Input Filter parameter*

| Sl.No | Condition | Filter to query records from SAP |
|---|---|---|
| 1 | IsInitialSync = True | **SAPC4CContact_To_MarketoLead** : StatusCode ne '4'<br>**updateDeletedSAPC4CContactsToMkto** : StatusCode eq '4' |
| 2 | IsInitialSync = False | **SAPC4CContact_To_MarketoLead** : StatusCode ne '4' and ChangedOn ge datetimeoffset'%fromDateString%'<br>**updateDeletedSAPC4CContactsToMkto** : StatusCode eq '4' and ChangedOn ge datetimeoffset'%fromDateString%' |

2. upsertMarketoLeads

   *Field Mapping*

| SAP Contact | | MKTO-PERSON | |
|---|---|---|---|
| StatusCode | Standard | obsoleteinC4C | CustomField |

| SAP Contact | | MKTO-PERSON | |
|---|---|---|---|
| TitleCodeText | Standard | salutation | Standard |
| FirstName | Standard | firstName | Standard |
| MiddleName | Standard | middleName | Standard |
| LastName | Standard | lastName | Standard |
| Email | Standard | email | Standard |
| Phone | Standard | phone | Standard |
| Mobile | Standard | mobilePhone | Standard |
| Fax | Standard | fax | Standard |
| JobTitle | Standard | title | Standard |
| BirthDate | Standard | dateOfBirth | Standard |
| Street | Standard | address | Standard |
| City | Standard | city | Standard |
| StateCodeText | Standard | state | Standard |
| CountryCodeText | Standard | country | Standard |
| StreetPostalCode | Standard | postalCode | Standard |
| ConsentEmail | Standard | unsubscribed | Standard |
| ConsentTelephone | Standard | donotcall | Standard |
| ContactTextCollection - Text | Standard | mktoPersonNotes | CustomField |
| DepartmentCodeText (Relation - CorporateAccountHas ContactPerson) | Standard | department | Standard |
| EmployeeID (Employee) | Standard | externalSalesPersonId | Standard |
| HouseNumber | Standard | contactHouseNumber | CustomField |
| ZLeadScore | CustomField | leadScore | Standard |
| ZAcquisitionProgram | CustomField | acquisitionProgramId | Standard |
| ZAcquisitionDate | CustomField | mktoAcquisitionDate | Standard |
| ZSource | CustomField | leadSource | Standard |

| SAP Contact | | MKTO-PERSON | |
|---|---|---|---|
| ContactID | Standard | c4CContactID | CustomField |

3. patchMultipleContact

   Once Contact records are successfully created in Marketo, update the Marketo ID to the corresponding SAP C4C Contact record in the MKTOContactID custom field.

| MKTO - PERSON | | SAP - CONTACT | |
|---|---|---|---|
| LeadID | Standard | MKTOCompanyID | CustomField |

4. readMultipleEmployees

   If any contact has any owner/Employee, then we have to associate the same owner to the Contact record in Marketo as well. Now as we query Contact records from SAP C4C, we get the Owner UUID instead of the ID. Hence this operation gets us the Owner ID for the corresponding UUID and then maps it to the Marketo Upsert Lead call to associate it to the proper SalesPerson.

### Usage Notes

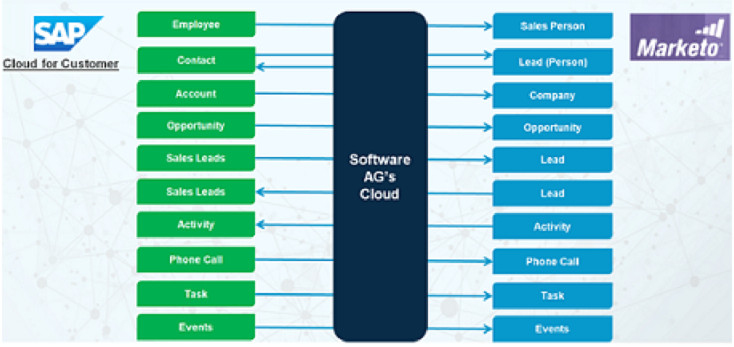
- In order to add new standard or custom fields to the **SAPC4CContact_To_MarketoLead** integration, copy the required field and add it to contactList_NullMKTOID, contactList_WithMKTOID, contactList_WithOwner, and notSyncedcontactList_withOwner list while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the **SAPC4CContact_To_MarketoLead** integration, remove the required field mapping and change or remove the field from contactList_NullMKTOID, contactList_WithMKTOID, contactList_WithOwner, and notSyncedcontactList_withOwner lists while making the upsert API call to Marketo.
- In order to add new standard or custom fields to the **updateDeletedSAPC4CContactsToMkto** integration, copy the required field and add it to contactList_NullMKTOID, contactList_WithMKTOID, contactList_WithOwner, and notSyncedcontactList_withOwner lists while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the **updateDeletedSAPC4CContactsToMkto** integration, remove the required field mapping and change or remove the field from contactList_NullMKTOID, contactList_WithMKTOID, contactList_WithOwner, and notSyncedcontactList_withOwner lists while making the upsert API call to Marketo.

## SAP C4C to Marketo Opportunity Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Opportunity** business object and Marketo **Opportunity** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



## Business Use Case

The business purpose of this integration is to extract SAP Cloud for Customer Opportunity records and upsert them into Marketo as Opportunity records with related Company and Salesperson and then create Marketo Opportunity Contact Role records with a dependency on Marketo Lead, based on the associated Contact of SAP Cloud for Customer.

## Custom Fields

The Opportunity object needs to be created or upserted in SAP C4C with the below mentioned custom fields. Similarly, when the Opportunity object gets synced in Marketo, it needs to be created or updated with the below mentioned custom fields. The required custom fields are as follows:

| On SAP C4C | Comments |
|---|---|
| zTotalNegotiatedValueConvertedcontent | Custom field needs to be created in SAP C4C to hold the Amount for Marketo Opportunity record. |
| zWeightedValueConvertedcontent | Custom field needs to be created in SAP C4C to hold the ExpectedRevenue for Marketo Opportunity record. |
| ZFiscalQuarter | Custom field needs to be created in SAP C4C to hold the FiscalQuarter for Marketo Opportunity record. |
| ZFiscalYear | Custom field needs to be created in SAP C4C to hold the FiscalYear for Marketo Opportunity record. |

| On SAP C4C | Comments |
|---|---|
| Fiscal | Custom field needs to be created in SAP C4C to hold the Fiscal for Marketo Opportunity record. |

| On Marketo | Comments |
|---|---|
| c4COpptID | Custom field in Marketo for Opportunity record to hold the SAP C4C Opportunity record ID. |
| opportunityObjectID | Custom field in Marketo for Opportunity record to hold the SAP C4C Opportunity record ObjectID |

## Required recipes to run before Opportunity

| SAP C4C | Marketo | Required Fields | Comments |
|---|---|---|---|
| Employee | SalesPerson | externalSalesPersonID | If employee is synched in Marketo, then only the Opportunity owner will be updated at Marketo from SAP C4C. |
| Account | Company | externalCompanyID (SAP ProspectPartyID) | If account is synced in Marketo, then only the Opportunity company will be updated at Marketo from SAP C4C. |
| Contact | Lead | c4CContactID (SAP PrimaryContactPartyID) | If contact is synced in Marketo, then only the Opportunity will be upserted at Marketo from SAP C4C. |

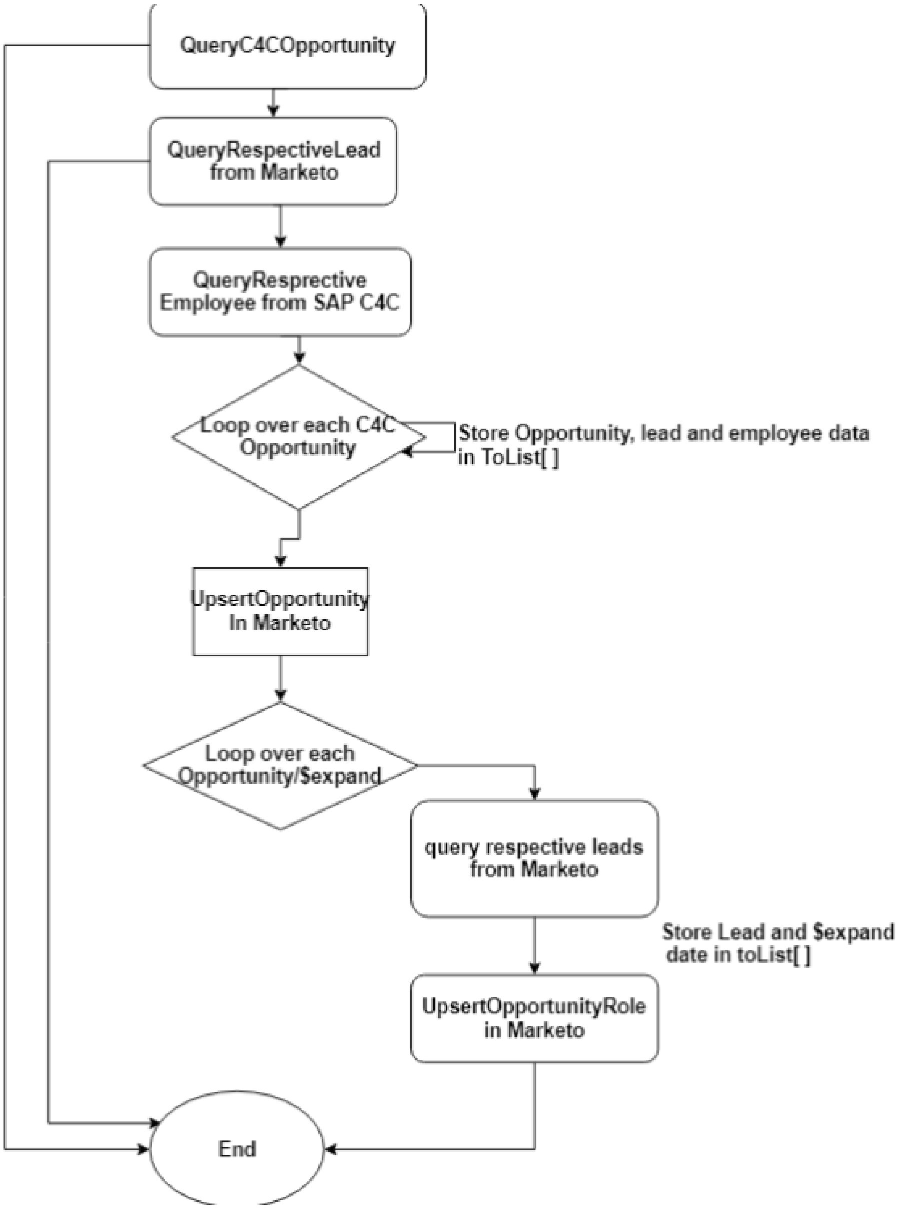
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## Data Sync Flow

This integration "SAPC4CToMarketoOpportunityDataSync" is designed to run for initial as well as ongoing sync. As part of the initial sync, all the Opportunity records present in SAP C4C will be synced to Marketo. On the other hand, as part of the ongoing sync, the updated or newly created Opportunity records will be synced to Marketo for the provided date range.

The following diagram represents creating a Opportunity in Marketo from an Opportunity in SAP C4C:


QueryC4COpportunity
QueryRespectiveLead from Marketo
QueryResprective Employee from SAP C4C
Loop over each C4C Opportunity
Store Opportunity, lead and employee data in ToList[ ]
UpsertOpportunity In Marketo
Loop over each Opportunity/$expand
query respective leads from Marketo
Store Lead and $expand date in toList[ ]
UpsertOpportunityRole in Marketo
End

## Input Parameters

| Sl.No | Parameter | Description |
|---|---|---|
| 1 | isInitialSync | Set to true in case of initial sync and to false in case of ongoing sync (true/false). |

| Sl.No | Parameter | Description |
|---|---|---|
| 2 | syncInterval | The sync interval can be entered either in hours/minutes/seconds or all of them and corresponding pattern and time zone if required. |
| 3 | logRecordStatusFlag | The log status of each record processed. It can either be true or false. |

## Operations Used

| Sl.No | Operations Used | Application | Description |
|---|---|---|---|
| 1 | QueryOpportunity | SAP C4C | Query SAP C4C Opportunity. |
| 2 | queryLead | Marketo | Query Lead from Marketo. |
| 3 | queryEmployee | SAP C4C | Query Employee from SAP C4C. |
| 4 | upsertOpportunities | Marketo | Upsert opportunities in Marketo. |
| 5 | upsertMarketoOpportunityRoles | Marketo | Upsert Opportunity Roles in Marketo. |

1. QueryOpportunity

   *Input Filter parameter*

   - IsInitialSync = True: ProspectPartyID ne null and MainEmployeeResponsiblePartyID ne null
   - IsInitialSync = False: ProspectPartyID ne null and MainEmployeeResponsiblePartyID ne null and ETag ge datetimeoffset'%fromDateString%'

   **Note:**
   Pagination will be done based on this operation. 100 records per API Call.

2. queryLead

   - QueryLead based on c4CCustomerID(CustomField) Identifier. This operation will provide related Marketo Lead(id), which will be mapped to leadId in upsertOpportunities operation.
   - QueryLead based on c4CCustomerID(CustomField) Identifier. This operation will provide related Marketo Lead(id), which will be mapped to leadId in upsertMarketoOpportunitiesRoles operation.

3. queryEmployee

   queryEmployee based on filter constructed with BusinessPartnerIDs to obtain Employee ID in SAP C4C. This operation will provide the related Employee ID, which will be mapped to externalSalesPersonId in upsertOpportunities operation.

   **Note:**

The required response data of queryOpportunity, queryLead, and queryEmployee will be stored in toList[].

a. upsertOpportunities

Field Mapping

| **SAP - OPPORTUNITY Source (ToList)** | **Field Type** | **MKTO - OPPORTUNITY** | **Field Type** |
|---|---|---|---|
| CreationDate | Standard | externalCreatedDate | Standard |
| ExpectedProcessingEndDate | Standard | closeDate | Standard |
| ID | Standard | externalOpportunityId | Standard |
| ID | Standard | c4COpptID | Custom |
| LifeCycleStatusCodeText | Standard | Stage | Standard |
| Name | Standard | description | Standard |
| ProspectPartyID | Standard | externalCompanyId | Standard |
| ProbabilityPercent | Standard | probability | Standard |
| Fiscal_KUT | Custom | fiscal | Standard |
| ZFiscalQuarter_KUT | Custom | fiscalQuarter | Standard |
| zTotalNegotiatedValue Convertedcontent_KUT | Custom | amount | Standard |
| zWeightedValue Convertedcontent_KUT | Custom | expectedRevenue | Standard |
| isClosed | Transformed from standard field StausCodeText | isClosed | Standard |
| isWon | Transformed from standard field StausCodeText | isWon | Standard |
| ZFiscalYear_KUT | Custom | fiscalYear | Standard |
| externalSalesPersonId | Mapped from /queryLeadOutput/leads/ | externalSalesPersonId | Standard |

| SAP - OPPORTUNITY Source (ToList) | Field Type | MKTO - OPPORTUNITY | Field Type |
|---|---|---|---|
| | externalSalesPersonId | | |
| SalesForecastCategoryCodeText | Standard | forecastCategoryName | Standard |
| ObjectID | Standard | opportunityObjectID | Custom |

**Note:**
Pagination will be done based on this operation. 100 records per API Call.

4. QueryOpportunity

   UpsertMarketoOpportunity is a very important step in syncing opportunities as opportunities will appear on Marketo UI only if contact roles are synced. Without this operation, the relationship between lead and opportunity is not defined.

   - One opportunity can be related to many contacts with different or same roles.
   - One opportunity will have only one primary contact and can be related to same contact with multiple roles.
   - One opportunity can be related to contacts with or without roles. In cases where roles are not defined in the SAP C4C side, those records will be skipped while creating the opportunity role.
   - In an opportunity, for every contact role, there will be a new entry of opportunityrole (Opportunity Info tab. See activities logs for further clarification at Marketo side.)

   **Note:**
   The required response data of queryOpportunity/$expand and queryLead will be stored in toList[].

   *FileMapping*

| Source (ToList) | Field Type | MKTO - OPPORTUNITY Role | Field Type |
|---|---|---|---|
| OpportunityPartyContact MainIndicator | Standard | isPrimary | Standard |
| OpportunityId | Standard | externalOpportunityId | Standard |
| LeadID | Standard | leadId | Standard |
| RoleCodeTest | Standard | role | Standard |

### Usage Notes

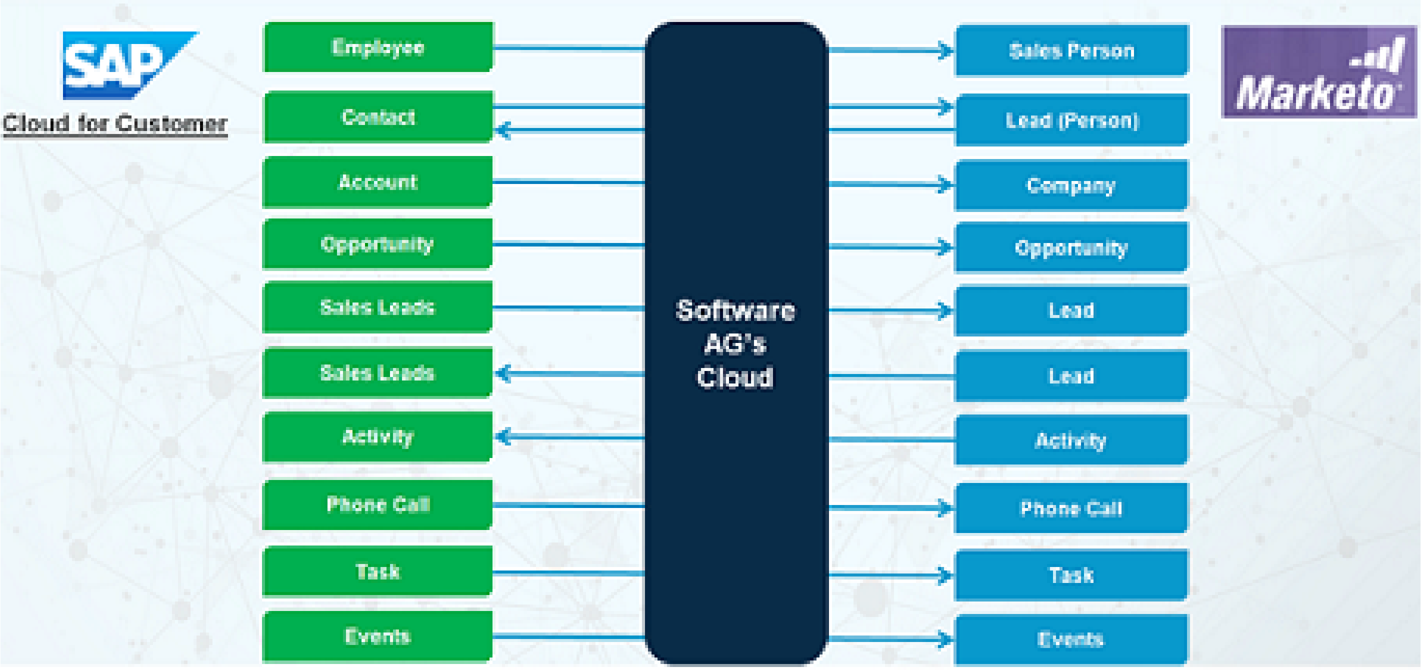
- In order to add new standard or custom fields to the integration, copy the required field to the respective toList array, and proceed with mapping from the respective toList array to the desired field.
- In order to change or remove standard or custom fields from the integration, remove the required field mapping and change or remove the field from the respective toList array.

## SAP C4C Employee to Marketo Sales Person Data Sync

This section describes the recipe for the data sync between SAP Cloud for Customer **Employee** business object and Marketo **Sales Person** business object.

SAP Cloud for Customer (C4C) is a cloud solution to manage customer sales, customer service, and marketing activities efficiently and is one of the key SAP solutions to manage customer relationships.

Marketo provides software platforms and technologies designed for marketing departments such as account-based marketing, which includes email, mobile, social, digital ads, web management, and analytics.



### Business Use Case

The business purpose of this integration is to extract the SAP Cloud for Customer Employee records and upsert them in Marketo as the records for Salesperson.

## Custom Fields

When the Employee object gets synced in Marketo, it needs to be created or updated with the custom fields mentioned below.

| On SAP C4C | Comments |
|---|---|
| MKTOEmployeeID | The custom field needs to be created for Employee records in SAP C4C to hold the Marketo Sales Person records id. |

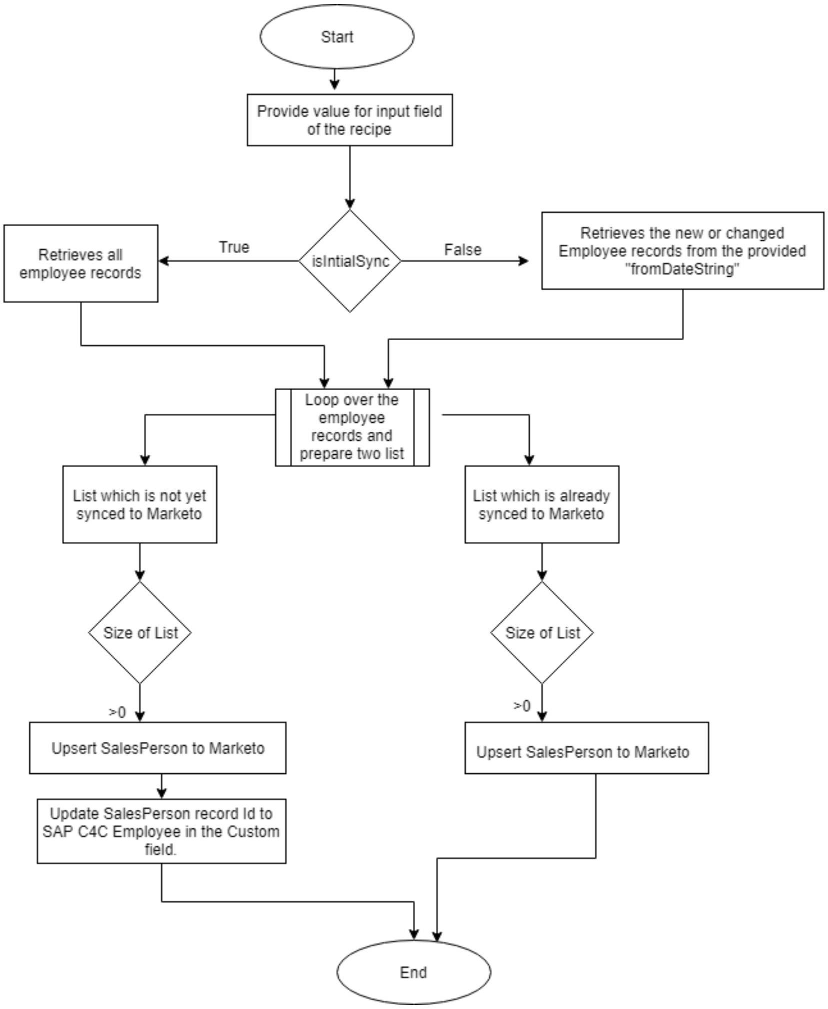
## Product Compatibility

The recipe is tested for SAP Cloud for Customer product version **1908.04.0006**. Hence the recipe is compatible to run for the tested and older versions of the product.

## SAP C4C Employee to Marketo Sales Person Sync Flow

This integration "SAPC4CToMarketoSalesPersonDataSync" is designed to run for initial as well as ongoing sync. As part of initial sync, all the employee records present in SAP C4C will be synced to Marketo. On the other hand, as part of ongoing sync, the newly created or updated Employee records will be synced to Marketo for the provided date range. Once the record is successfully created in Marketo, it syncs back the Marketo Salesperson ID in the corresponding SAP C4C Employee record (Custom Field: MKTOEmployeeID).

The following diagram represents creating a Salesperson in Marketo from an Employee in SAP C4C:


Start
Provide value for input field of the recipe
isIntialSync
True
False
Retrieves all employee records
Retrieves the new or changed Employee records from the provided "fromDateString"
Loop over the employee records and prepare two list
List which is not yet synced to Marketo
List which is already synced to Marketo
Size of List
Size of List
>0
>0
Upsert SalesPerson to Marketo
Upsert SalesPerson to Marketo
Update SalesPerson record Id to SAP C4C Employee in the Custom field.
End

## Operations Used

This recipe has a sub flow to handle the obsolete contact record in SAP C4C, that is, **updateDeletedSAPC4CContactsToMkto**.

| Operation Name | Application | Description |
|---|---|---|
| querySAPC4CEmployees | SAP C4C | Queries SAP C4C Employee. |
| upsertMarketoSalesPerson | Marketo | Upserts Employee in Marketo as SalesPerson records. |
| patchSAPC4CMultipleContact | SAP C4C | Once records are successfully created in Marketo, it updates Marketo SalesPerson ID in the corresponding SAP C4C employee record in the **MKTOEmployeeID** custom field. |

1. querySAPC4CEmployees

*Input Filter parameter*

| Condition | Filter to query records from SAP |
|---|---|
| IsInitialSync = True | No filter required. |
| IsInitialSync = False | ChangedOn ge datetimeoffset'%fromDateString%' |

> **Note:**
> Pagination is taken care in the recipe where it fetches 100 records per API Call.

2. upsertMarketoSalesPerson

   *Field Mapping*

| SAP - EMPLOYEE | | MKTO - SALES PERSON | |
|---|---|---|---|
| EmployeeID | Standard | externalSalesPersonID | Standard |
| Email | Standard | email | Standard |
| FaxNumber | Standard | fax | Standard |
| FirstName | Standard | firstName | Standard |
| LastName | Standard | lastName | Standard |
| MobilePhoneNumber | Standard | mobilePhone | Standard |
| OfficePhoneNumber | Standard | phone | Standard |
| JobName | Standard | title | Standard |

3. patchMultipleEmployee

   Once Employee records are successfully created in Marketo as SalesPerson records, then update Marketo SalesPerson ID to the corresponding SAP C4C employee record in the **MKTOEmployeeID** custom field.

| MKTO - SALES PERSON | | SAP - EMPLOYEE | |
|---|---|---|---|
| ID | Standard | MKTOEmployeeID | CustomField |
| CountryCode | Standard | CountryCode | Standard |
| FirstName | Standard | FirstName | Standard |
| LastName | Standard | LastName | Standard |
| ObjectID | Standard | ObjectID | Standard |

## Usage Notes

- In order to add new standard or custom fields to the integration, copy the required field and add it to empList_NullMKTOID list and empList_WithMKTOID list while making the upsert API call to Marketo.
- In order to change or remove standard or custom fields from the integration, remove the required field mapping and change or remove the field from empList_NullMKTOID list and empList_WithMKTOID list while making the upsert API call to Marketo.

# 10 Built-In Services

# Overview

Services are method-like units of logic that clients can invoke. Integration Cloud has an extensive library of built-in services for performing common integration tasks such as transforming data values, performing simple mathematical operations, and so on. Related services are grouped in blocks. Input and output parameters are the names and types of fields that the service requires as input and generates as output and these parameters are collectively referred to as a signature.

# Built-In Services

Services will be invoked at run time and related services are grouped together. While creating an Integration, you can sequence services and manage the flow of data among them.

## Date Services

Use **Date** services to generate and format date values.

*Pattern String Symbols* - Many of the Date services require you to specify pattern strings describing the data's current format and/or the format to which you want it converted. For services that require a pattern string, use the symbols in the following table to describe the format of your data. For example, to describe a date in the January 15, 1999 format, you would use the pattern string `MMMMM dd, yyyy`. To describe the format 01/15/99, you would use the pattern string `MM/dd/yy`.

| Symbol | Meaning | Presentation | Example |
|---|---|---|---|
| G | era designator | Text | `AD` |
| y | year | Number | `1996 or 96` |
| M | month in year | Text or Number | `July or Jul or 07` |
| d | day in month | Number | `10` |
| h | hour in am/pm (1-12) | Number | `12` |
| H | hour in day (0-23) | Number | `0` |
| m | minute in hour | Number | `30` |
| s | second in minute | Number | `55` |
| S | millisecond | Number | `978` |
| E | day in week | Text | `Tuesday or Tue` |
| D | day in year | Number | `189` |
| F | day of week in month | Number | `2 (2nd Wed in July)` |
| w | week in year | Number | `27` |

| Symbol | Meaning | Presentation | Example |
|---|---|---|---|
| W | week in month | Number | `2` |
| a | am/pm marker | Text | `PM` |
| k | hour in day (1-24) | Number | `24` |
| K | hour in am/pm (0-11) | Number | `0` |
| z | time zone | Text | `Pacific Standard Time or PST or GMT-08:00` |
| Z | RFC 822 time zone (JVM 1.4 or later) | Number | `-0800 (offset from GMT/UT)` |
| ' | escape for text | Delimiter | |
| '' | single quote | Literal | `'` |

*Time Zones* - When working with date services, you can specify time zones. The Earth is divided into 24 standard time zones, one for every 15 degrees of longitude. Using the time zone including Greenwich, England (known as Greenwich Mean Time, or GMT) as the starting point, the time is increased by an hour for each time zone east of Greenwich and decreases by an hour for each time zone west of Greenwich. The time difference between a time zone and the time zone including Greenwich, England (GMT) is referred to as the *raw offset*.

The following table identifies the different time zones for the Earth and the raw offset for each zone from Greenwich, England. The effects of daylight savings time are ignored in this table.

**Note:**
Greenwich Mean Time (GMT) is also known as Universal Time (UT).

| ID | Raw Offset | Name |
|---|---|---|
| MIT | -11 | Midway Islands Time |
| HST | -10 | Hawaii Standard Time |
| AST | -9 | Alaska Standard Time |
| PST | -8 | Pacific Standard Time |
| PNT | -7 | Phoenix Standard Time |
| MST | -7 | Mountain Standard Time |
| CST | -6 | Central Standard Time |
| EST | -5 | Eastern Standard Time |
| IET | -5 | Indiana Eastern Standard Time |

| ID | Raw Offset | Name |
|---|---|---|
| PRT | -4 | Puerto Rico and U.S. Virgin Islands Time |
| CNT | -3.5 | Canada Newfoundland Time |
| AGT | -3 | Argentina Standard Time |
| BET | -3 | Brazil Eastern Time |
| GMT | 0 | Greenwich Mean Time |
| ECT | +1 | European Central Time |
| CAT | +2 | Central Africa Time |
| EET | +2 | Eastern European Time |
| ART | +2 | (Arabic) Egypt Standard Time |
| EAT | +3 | Eastern African Time |
| MET | +3.5 | Middle East Time |
| NET | +4 | Near East Time |
| PLT | +5 | Pakistan Lahore Time |
| IST | +5.5 | India Standard Time |
| BST | +6 | Bangladesh Standard Time |
| VST | +7 | Vietnam Standard Time |
| CTT | +8 | China Taiwan Time |
| JST | +9 | Japan Standard Time |
| ACT | +9.5 | Australian Central Time |
| AET | +10 | Australian Eastern Time |
| SST | +11 | Solomon Standard Time |
| NST | +12 | New Zealand Standard Time |

*Examples* - You can specify *timezone* input parameters in the following formats:

- As a full name. For example:

```
Asia/Tokyo          America/Los_Angeles
```

You can use the java.util.TimeZone.getAvailableIDs() method to obtain a list of the valid full name time zone IDs that your JVM version supports.

- As a custom time zone ID, in the format GMT[+ | -]hh[ [:]mm]. For example:

| | |
|---|---|
| `GMT+2:00` | All time zones 2 hours east of Greenwich (that is, Central Africa Time, Eastern European Time, and Egypt Standard Time) |
| `GMT-3:00` | All time zones 3 hours west of Greenwich (that is, Argentina Standard Time and Brazil Eastern Time) |
| `GMT+9:30` | All time zones 9.5 hours east of Greenwich (that is, Australian Central Time) |

- As a three-letter abbreviation from the table above. For example:

| | |
|---|---|
| `PST` | Pacific Standard Time |

**Note:**
Because some three-letter abbreviations can represent multiple time zones, for example, "CST" could represent both U.S. "Central Standard Time" and "China Standard Time", all abbreviations are deprecated. Use the full name or custom time zone ID formats instead.

*Notes on Invalid Dates* - The dates you use with a date service must adhere to the java.text.SimpleDateFormat class.

If you use an invalid date with a date service, the date service automatically translates the date to a legal date. For example, if you specify `1999/02/30` as input, the date service interprets the date as 1999/03/02 (two days after 2/28/1999).

If you use `00` for the month or day, the date service interprets `00` as the last month or day in the Gregorian calendar. For example, if you specify `00` for the month, the date service interprets it as 12.

If the pattern *yy* is used for the year, the date service uses a 50-year moving window to interpret the value of *yy*. The date service establishes the window by subtracting 49 years from the current year and adding 50 years to the current year. For example, if you are running Integration Cloud in the year 2000, the moving window would be from 1951 to 2050. The date service interprets 2-digit years as falling into this window (for example, 12 would be 2012, 95 would be 1995).

The following **Date** services are available:

| Service | Description |
|---|---|
| calculateDateDifference | Calculates the difference between two dates and returns the result as seconds, minutes, hours, and days. |
| compareDates | Compares two dates and returns the result as integer. |
| currentNanoTime | Returns the current time returned by the most precise system timer, in nanoseconds. |
| dateBuild | Builds a date String using the specified pattern and the specified date services. |

| Service | Description |
|---|---|
| dateTimeBuild | Builds a date/time string using the specified pattern and the specified date services. |
| dateTimeFormat | Converts date/time (represented as a String) string from one format to another. |
| elapsedNanoTime | Calculates the time elapsed between the current time and the given time, in nanoseconds. |
| formatDate | Formats a Date object as a string. |
| getCurrentDate | Returns the current date as a Date object. |
| getCurrentDateString | Returns the current date as a String in a specified format. |
| incrementDate | Increments a date by a specified period. |

## calculateDateDifference

Calculates the difference between two dates and returns the result as seconds, minutes, hours, and days.

### Input Parameters

*startDate* **String** Starting date and time.

*endDate* **String** Ending date and time.

*startDatePattern* **String** Format in which the *startDate* parameter is to be specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the “Pattern String Symbols” section.

*endDatePattern* **String** Format in which the *endDate* parameter is to be specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the “Pattern String Symbols” section.

### Output Parameters

*dateDifferenceSeconds* **String** The difference between the startingDateTime and endingDateTime, truncated to the nearest whole number of seconds.

*dateDifferenceMinutes* **String** The difference between the startingDateTime and endingDateTime, truncated to the nearest whole number of minutes.

*dateDifferenceHours* **String** The difference between the startingDateTime and endingDateTime, truncated to the nearest whole number of hours.

*dateDifferenceDays* **String** The difference between the startingDateTime and endingDateTime, truncated to the nearest whole number of days.

### Usage Notes

Each output value represents the same date difference, but in a different scale. Do not add these values together. Make sure your subsequent Integration steps use the correct output, depending on the scale required.

## compareDates

Compares two dates and returns the result as an integer.

### Input Parameters

*startDate* **String** Starting date to compare against *endDate*.

*endDate* **String** Ending date to compare against *startDate*.

*startDatePattern* **String** Format in which the *startDate* parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.

*endDatePattern* **String** Format in which the *endDate* parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.

### Output Parameters

*result* **String** Checks whether *startDate* is before, the same, or after the *endDate*.

| A value of... | Indicates that... |
|---|---|
| `+1` | The *startDate* is after the *endDate*. |
| `0` | The *startDate* is the same as the *endDate*. |
| `-1` | The *startDate* is before the *endDate*. |

### Usage Notes

If the formats specified in the *startDatePattern* and *endDatePattern* parameters are different, Integration Cloud takes the units that are not specified in the *startDate* and *endDate* values as `0`.

That is, if the *startDatePattern* is `yyyyMMdd HH:mm` and the *startDate* is `20151030 11:11` and if the *endDatePattern* is `yyyyMMdd HH:mm:ss.SSS`and the *endDate* is `20151030 11:11:55:111`, then the compareDates service considers start date to be before the end date and will return the result as `-1`.

To calculate the difference between two dates, use the calculateDateDifference service.

## currentNanoTime

Returns the current time returned by the most precise system timer, in nanoseconds.

### Input Parameters

None.

### Output Parameters

| | |
|---|---|
| *nanoTime* | **java.lang.Long** Current time returned by the most precise system timer, in nanoseconds. |

## dateBuild

Builds a date String using the specified pattern and the specified date services.

### Input Parameters

| | |
|---|---|
| *pattern* | **String** Pattern representing the format in which you want the date returned. For pattern-string notation, see the "Pattern String Symbols" section. If you do not specify *pattern*, dateBuild returns null. If *pattern* contains a time zone and *timezone* is not specified, the default time zone is used. |
| *year* | **String** Optional. The year expressed in *yyyy* or *yy* format (for example, `01` or `2001`). If you do not specify *year* or you specify an invalid value, dateBuild uses the current year. |
| *month* | **String** Optional. The month expressed as a number (for example, `1` for January, `2` for February). If you do not specify *month* or you specify an invalid value, dateBuild uses the current month. |
| *dayofmonth* | **String** Optional. The day of the month expressed as a number (for example, `1` for the first day of the month, `2` for the second day of the month). If you do not specify *dayofmonth* or you specify an invalid value, dateBuild uses the current day. |
| *timezone* | **String** Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the "Time Zones" section, for example, `EST` for Eastern Standard Time. |

If you do not specify *timezone*, the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.

*locale* **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

### Output Parameters

*value* **String** The date specified by *year*, *month*, and *dayofmonth*, in the format of *pattern*.

## dateTimeBuild

Builds a date/time string using the specified pattern and the specified date services.

### Input Parameters

*pattern* **String** Pattern representing the format in which you want the time returned. For pattern-string notation, see the “Pattern String Symbols” section. If you do not specify *pattern*, dateTimeBuild returns null. If *pattern* contains a time zone and the *timezone* parameter is not set, the default time zone is used.

*year* **String** Optional. The year expressed in *yyyy* or *yy* format (for example, `01` or `2001`). If you do not specify *year* or you specify an invalid value, dateTimeBuild uses the current year.

*month* **String** Optional. The month expressed as a number (for example, `1` for January, `2` for February). If you do not specify *month* or you specify an invalid value, dateTimeBuild uses the current month.

*dayofmonth* **String** Optional. The day of the month expressed as a number (for example, `1` for the first day of the month, `2` for the second day of the month). If you do not specify *dayofmonth* or you specify an invalid value, dateTimeBuild uses the current day.

*hour* **String** Optional. The hour expressed as a number based on a 24-hour clock. For example, specify `0` for midnight, `2` for 2:00 A.M., and `14` for 2:00 P.M. If you do not specify *hour* or you specify an invalid value, dateTimeBuild uses 0 as the *hour* value.

*minute* **String** Optional. Minutes expressed as a number. If you do not specify *minute* or you specify an invalid value, dateTimeBuild uses 0 as the *minute* value.

*second* **String** Optional. Seconds expressed as a number. If you do not specify *second* or you specify an invalid value, dateTimeBuild uses 0 as the *second* value.

*millis* **String** Optional. Milliseconds expressed as a number. If you do not specify *millis* or you specify an invalid value, dateTimeBuild uses 0 as the *millis* value.

*timezone* **String** Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the “Time Zones” section, for example, `EST` for Eastern Standard Time.

If you do not specify *timezone*, the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.

*locale* **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

### Output Parameters

*value* **String** Date and time in format of *pattern*.

## dateTimeFormat

Converts date/time (represented as a String) string from one format to another.

### Input Parameters

*inString* **String** Date/time that you want to convert.

> **Important:**
> If *inString* contains a character in the last position, that character is interpreted as 0. This can result in an inaccurate date. For information about invalid dates, see the “Notes on Invalid Dates” section.

*currentPattern* **String** Pattern string that describes the format of *inString*. For pattern-string notation, see the “Pattern String Symbols” section.

*newPattern* **String** Pattern string that describes the format in which you want *inString* returned. For pattern-string syntax, see the “Pattern String Symbols” section.

*locale* **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

*lenient* **String** Optional. A flag indicating whether an exception will appear if the *inString* value does not adhere to the format specified in *currentPattern* parameter. Set to:

- `true` to perform a lenient check. This is the default.

In a lenient check, if the format of the date specified in the *inString* parameter does not match the format specified in the *currentPattern* parameter, the date in the format specified in the *currentPattern* parameter will be interpreted and returned. If the interpretation is incorrect, the service will return an invalid date.

- `false` to perform a strict check.

  In a strict check, an exception will appear if the format of the date specified in the *inString* parameter does not match the format specified in the *currentPattern* parameter.

### Output Parameters

*value* **String** The date/time given by *inString*, in the format of *newPattern*.

### Usage Notes

As described in the “Notes on Invalid Dates" section, if the pattern *yy* is used for the year, dateTimeFormat uses a 50-year moving window to interpret the value of the year.

If *currentPattern* does not contain a time zone, the value is assumed to be in the default time zone.

If *newPattern* contains a time zone, the default time zone is used.

## elapsedNanoTime

Calculates the time elapsed between the current time and the given time, in nanoseconds.

### Input Parameters

*nanoTime* **java.lang.Long** Time in nanoseconds. If *nanoTime* is less than zero, then the service treats it as zero.

### Output Parameters

*elapsedNanoTime* **java.lang.Long** The difference between the current time in nanoseconds and *nanoTime*. If *nanoTime* is greater than the current nano time, the service returns zero.

*elapsedNanoTimeStr* **String** The difference between the current time in nanoseconds and *nanoTime*. The difference is expressed as a String, in this format:

[years] [days] [hours] [minutes] [seconds] [millisec] [microsec] <nanosec>

If *nanoTime* is greater than the current nano time, the service returns zero.

## formatDate

Formats a Date object as a string.

### Input Parameters

*date* **java.util.Date** Optional. Date/time that you want to convert.

*pattern* **String** Pattern string that describes the format in which you want the date returned. For pattern-string notation, see the *Pattern String Symbols* section.

*timezone* **String** Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the *Time Zones* section, for example, `EST` for Eastern Standard Time.

If you do not specify *timezone*, the user's time zone is used, else GMT is used.

*locale* **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

### Output Parameters

*value* **String** The date/time given by *date* in the format specified by *pattern*.

## getCurrentDate

Returns the current date as a Date object.

### Input Parameters

None.

### Output Parameters

*date* **java.util.Date** Current date.

## getCurrentDateString

Returns the current date as a String in a specified format.

### Input Parameters

*pattern* **String** Pattern representing the format in which you want the date returned. For pattern-string notation, see the “Pattern String Symbols” section.

*timezone* **String** Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the “Time Zones” section, for example, `EST` for Eastern Standard Time.

If you do not specify *timezone*, the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.

*locale* **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

### Output Parameters

*value* **String** Current date in the format specified by *pattern*.

## incrementDate

Increments a date by a specified amount of time.

### Input Parameters

*startDate* **String** Starting date and time.

*startDatePattern* **String** Format in which the *startDate* parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.

*endDatePattern* **String** Optional. Pattern representing the format in which you want the *endDate* to be returned. For pattern-string notation, see the "Pattern String Symbols" section.

If no *endDatePattern* is specified, the *endDate* will be returned in the format specified in the *startDatePattern* parameter.

*addYears* **String** Optional. Number of years to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addMonths* **String** Optional. Number of months to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addDays*

**String** Optional. Number of days to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addHours*

**String** Optional. Number of hours to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addMinutes*

**String** Optional. Number of minutes to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addSeconds*

**String** Optional. Number of seconds to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*addMilliSeconds*

**String** Optional. Number of milliseconds to add to *startDate*. The value must be an integer between -2147483648 and 2147483647.

*timezone*

**String** Optional. Time zone in which you want the *endDate* to be expressed. Specify a time zone code, for example, EST for Eastern Standard Time.

If you do not specify *timezone*, the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.

*locale*

**String** Optional. Locale in which the *endDate* is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

## Output Parameters

*endDate*

**String** The end date and time, calculated by incrementing the *startDate* with the specified years, months, days, hours, minutes, seconds, and/or milliseconds. The *endDate* will be in the *endDatePattern* format, if specified. If no *endDatePattern* is specified or if blank spaces are specified as the value, the *endDate* will be returned in the format specified in the *startDatePattern* parameter.

## Usage Notes

The *addYears*, *addMonths*, *addDays*, *addHours*, *addMinutes*, *addSeconds*, and *addMilliSeconds* input parameters can take positive or negative values. For example, If *startDate* is 10/10/2001, *startDatePattern* is MM/dd/yyyy, *addYears* is 1, and *addMonths* is -1, *endDate* will be 09/10/2002.

If you specify only the *startDate*, *startDatePattern*, and *endDatePattern* input parameters and do not specify any of the optional input parameters to increment the period, the incrementDate service just converts the format of *startDate* from *startDatePattern* to *endDatePattern* and returns it as *endDate*.

**Note:**

The format of the date specified in the *startDate* parameter must match the format specified in the *startDatePattern* and the format of the date specified in the *endDate* parameter must match the *endDatePattern* format.

## Document Services

Use **Document** services to perform operations on documents.

The following **Document** services are available:

| Service | Description |
|---|---|
| findDocuments | Searches a set of documents for entries matching a set of Criteria. |
| insertDocument | Inserts a new document in a set of documents at a specified position. |
| deleteDocuments | Deletes the specified documents from a set of documents. |
| documentListToDocument | Constructs a document from a document list by generating key/value pairs from the values of two elements that you specify in the document list. |
| documentToDocumentList | Expands the contents of a document into a list of documents.<br>Each key/value pair in the source document is transformed to a single document containing two keys (whose names you specify). These two keys will contain the key name and value of the original pair. |
| groupDocuments | Groups a set of documents based on specified criteria. |
| documentToBytes | Converts a document to an array of bytes. |
| bytesToDocument | Converts an array of bytes to a document. |
| searchDocuments | Searches a set of documents for entries matching a set of Criteria. |
| sortDocuments | Sorts a set of input documents based on the specified sortCriteria. |

## findDocuments

Searches a set of documents for entries matching a set of criteria.

### Input Parameters

*documents* **Document List** Set of documents from which the documents meeting the retrieve criteria are to be returned.

*matchCriteria* **Document** Criteria on which the documents in the `documents` parameter are to be matched. Parameters for `matchCriteria` are:

**path**: Name of the element in `documentList` whose value provides the value for the search text. The value for `key` can be a path expression. For example, "Family/Chidren[0]/ BirthDate" retrieves the birthday of the first child from the input `Family` document list.

**compareValueAs**: Optional. Allowed values are string, numeric, and datetime. The default value is string.

**datePattern**: Optional. Pattern will be considered only if `compareValueAs` is of type datetime. Default value is MM/dd/yyyy hh:mm:ss a.

**joins**: List of join criteria. Each join criteria consists of:

`operator`: Allowed values are equals, doesNotEqual, greaterThan, greaterThanEqual, lessThan, lessThanEqual, equalsIgnoreCase, contains, doesNotContain, beginsWith, doesNotBeginWith, endsWith, doesNotEndWith.

`value`: Optional. Allowed values are string, numeric, and datetime. The default value is string.

`joinType`: Specifies the way two joins can be linked. Values are “and” or “or”. Default value is “and”.

### Output Parameters

*result documents* **Document List** List of documents that match the retrieve criteria.

## insertDocument

Inserts a new document in a set of documents at a specified position.

### Input Parameters

*documents* **Document List** Set of documents in which a new document is to be inserted.

*insertDocument* **Document** The new document to be inserted to the set of documents specified in the *documents* parameter.

*index* **String** Optional. The position in the seat which the document is to be inserted.

The *index* parameter is zero-based. if the value for the *index* parameter is not specified, the document will be inserted at the end of the document list specified in the *documents* parameter.

### Output Parameters

*documents* **Document List** Document list after inserting the new document.

## deleteDocuments

Deletes the specified documents from a set of documents.

### Input Parameters

*documents* **Document List** Set of documents that contain the documents you want to delete.

*indices* **String List** Index values of documents to be deleted from the *documents* parameter document list.

### Output Parameters

*documents* **Document List** List of documents whose indices do *not* match the values in *indices* parameter.

*deletedDocuments* **Document List** List of deleted documents.

### Usage Notes

The deleteDocuments service returns an error if the *indices* parameter value is less than zero or more than the number of documents in the *documents* input parameter.

## documentListToDocument

Constructs a document from a document list by generating key/value pairs from the values of two elements that you specify in the document list.

## Input Parameters

*documentList* — **Document List** Set of documents that you want to transform into a single document.

> **Note:**
> If the *documentList* parameter contains a single document instead of a Document List, the documentListToDocument service does nothing.

*name* — **String** Name of the element in the *documentList* parameter whose value provides the name of each key in the resulting document.

> **Important:**
> The data type of the element that you specify in the *name* parameter must be String.

*value* — **String** Name of the element in the *documentList* parameter whose values will be assigned to the keys specified in *name*. This element can be of any data type.

## Output Parameters

*document* — **Document** Document containing the key/value pairs generated from the *documentList* parameter.

## Usage Notes

The following example illustrates how the documentListToDocument service would convert a document list that contains three documents to a single document containing three key/value pairs. When you use the documentListToDocument service, you specify which two elements from the source list are to be transformed into the keys and values in the output document. In the following example, the values from the *pName* elements in the source list are transformed into key names, and the values from the *pValue* elements are transformed into the values for these keys.

A documentList containing these three documents:

| Key | Value |
|---|---|
| *pName* | `cx_timeout` |
| *pValue* | `1000` |

| Key | Value |
|---|---|
| *pName* | `cx_max` |
| *pValue* | `2500` |

| Key | Value |
|---|---|
| *pName* | `cx_min` |
| *pValue* | `10` |

Would be converted to a document containing these three key:

| Key | Value |
|---|---|
| *cx_timeout* | `1000` |
| *cx_max* | `2500` |
| *cx_min* | `10` |

## documentToDocumentList

Expands the contents of a document into a list of documents.

Each key/value pair in the source document is transformed to a single document containing two keys (whose names you specify). These two keys will contain the key name and value of the original pair.

### Input Parameters

*document* **Document** Document to transform.

*name* **String** Name to assign to the key that will receive the key name from the original key/value pair. In the example above, this parameter was set to `pName`.

*value* **String** Name to assign to the key that will receive the value from the original key/value pair. In the example above, this parameter was set to `pValue`.

### Output Parameters

*documentList* **Document List** List containing a document for each key/value pair in the *document* parameter. Each document in the list will contain two keys, whose names were specified by the *name* and *value* parameters. The values of these two keys will be the name and value (respectively) of the original pair.

### Usage Notes

The following example shows how a document containing three keys would be converted to a document list containing three documents. In this example, the names *pName* and *pValue* are specified as names for the two new keys in the document list.

A document containing these three keys:

| Key | Value |
|---|---|
| *cx_timeout* | `1000` |
| *cx_max* | `2500` |
| *cx_min* | `10` |

Would be converted to a document list containing these three documents:

| Key | Value |
|---|---|
| *pName* | `cx_timeout` |
| *pValue* | `1000` |

| Key | Value |
|---|---|
| *pName* | `cx_max` |
| *pValue* | `2500` |

| Key | Value |
|---|---|
| *pName* | `cx_min` |
| *pValue* | `10` |

## groupDocuments

Groups a set of documents based on specified criteria.

### Input Parameters

*documents* **Document List** Set of documents to be grouped based on the specified criteria.

*groupCriteria* **Document List** The criteria on which the input documents are to be grouped. Valid values for the *groupCriteria* parameter are:

- *key*. Key in the pipeline. The value for *key* can be a path expression. For example, "`Family/Chidren[0]/BirthDate`" retrieves the birthday of the first child from the input Family document list.
- *compareStringsAs*. Optional. Valid values for *compareStringsAs* are `string`, `numeric`, and `datetime`. The default value is `string`.

- *pattern*. Optional. *pattern* will be considered only if the *compareStringsAs* parameter is of type `datetime`.

  > **Note:**
  > If *key* is not found in all the input documents, the documents that do not match the *groupCriteria* are grouped together as a single group.

### Output Parameters

*documentGroups* **Document List** List of documents where each element represents a set of documents grouped based on the criteria specified.

### Usage Notes

The following example illustrates how to specify the values for the *groupCriteria* parameter:

| key | compareStringsAs | pattern |
|---|---|---|
| name | string | |
| age | numeric | |
| birthdate | datetime | yyyy-MM-dd |

The input documents will be grouped based on name, age, and birth date.

## documentToBytes

Converts a document to an array of bytes.

### Input Parameters

*document* **Document** Document to convert to bytes.

- If *document* is null, the service does not return an output or an error message.
- If *document* is not a document, the service throws an exception.
- If *document* contains no elements, the service produces a zero-length byte array.

### Output Parameters

*documentBytes* **Object** A serialized representation of the document as an array of bytes (byte[]).

### Usage Notes

Use the documentToBytes service with the bytesToDocument service, which converts the byte array created by this service back into the original document.

The documentToBytes service is useful when you want to write a document to a file, an input stream, or a cache.

In order for the document-to-bytes-to-document conversion to work, the entire content of the document must be serializable. Every object in the document must be of a data type known to Integration Cloud, or it must support the java.io.Serializable interface. If Integration Cloud encounters an unknown object in the document that does not support the java.io.Serializable interface, that object's value will be lost. Integration Cloud will replace it with a string containing the object's class name.

## bytesToDocument

Converts an array of bytes to a document. This service can only be used with byte arrays created by executing the documentToBytes service.

### Input Parameters

*documentBytes* **Object** An array of bytes (byte[]) to convert to a document.

- If *documentBytes* is null, the service does not return a document or an error message.
- If *documentBytes* is not a byte array, the service throws an exception.
- If *documentBytes* is zero-length, the service produces an empty document.

### Output Parameters

*document* **Document** A document.

### Usage Notes

Use this service with the documentToBytes service, which converts a document into a byte array. You can pass the resulting byte array to the bytesToDocument service to convert it back into the original document.

In order for the document-to-bytes-to-document conversion to work, the entire content of the document must be serializable. Every object in the document must be of a data type known to Integration Cloud, or it must support the java.io.Serializable interface.

**Note:**

If Integration Cloud encounters an unknown object in the document that does not support the java.io.Serializable interface, that object's value will be lost. It will be replaced with a string containing the object's class name.

## searchDocuments

Searches a set of documents for entries matching a set of Criteria.

### Input Parameters

*documents* **Document List** Set of documents from which the documents meeting the search criteria are to be returned.

*searchCriteria* **Document** Criteria on which the documents in the *documents* parameter are to be searched.

Valid values for *searchCriteria* parameters are:

- *key*. Name of the element in documentList whose value provides the value for the search text. The value for *key* can be a path expression. For example, "`Family/Chidren[0]/BirthDate`" retrieves the birthday of the first child from the input Family document list.
- *value*. Optional. Any search text. If no value is specified, the service searches for null in the document list.
- *compareStringsAs*. Optional. Allowed values are `string`, `numeric`, and `datetime`. The default value is `string`.
- *pattern*. Optional. *pattern* will be considered only if the *compareStringsAs* value is of type datetime. For information about using patterns, see the *Time Zones* section.

*sorted* **String** Optional. The value of the *sorted* parameter is `true` if the document list is already sorted based on the search criteria and same search key; otherwise `false`.

If the value for the *sorted* parameter is set to `true`, the required documents are searched faster.

### Output Parameters

*resultdocuments* **Document List** List of documents which are matching the search criteria.

*documentListIndices* **String List** Positions of search documents in the document list.

*documents* **Document List** List of documents that were input.

### Usage Note

For example, if you want to search a set of documents for documents where BirthDate is 10th January 2008, the values for the *searchCriteria* parameter would be:

| key | value | compareStringsAs | pattern |
|---|---|---|---|
| Birthdate | 2008-01-10 | datetime | yyyy-MM-dd |

## sortDocuments

Sorts a set of input documents based on the specified sortCriteria.

### Input Parameters

*documents* **Document List** Set of documents that are to be sorted.

*sortCriteria* **Document List** Criteria based on which the documents in the *documents* parameter are to be sorted.

Valid values for *sortCriteria* parameters are:

- *key*. Name of the element in documentList whose value provides the value based on which the documents are to be sorted. The value for *key* can be a path expression. For example, "`Family/Chidren[0]/BirthDate`" retrieves the birthday of the first child from the input Family document list.
- *order*. Optional. Allowed values are `ascending` and `descending`. The default value is `ascending`.
- *compareStringsAs*. Optional. Allowed values are `string`, `numeric`, and `datetime`. Default value is `string`.
- *pattern*. Optional. The value for *pattern* will be considered only if the *compareStringsAs* value is of type datetime.

**Note:**
If *key* is not found in all the input documents, the sorted list of documents appears at the end or start of the list based on the *order* specified. If the order is ascending, then all the documents that do not match the sort criteria appears at the top of the list, followed by the sorted list. If the order is descending, the sorted list will appear at the top, followed by the documents that do not match the sort criteria.

### Output Parameters

*documents* **Document List** The documents sorted based on the sort criteria specified in the *sortCriteria* parameter.

### Usage Notes

For example, if you want to sort a set of documents based on name, age, and then on birth date, the values for *sortCriteria* parameter would be:

| key | order | compareStringsAs | pattern |
|---|---|---|---|
| Name | ascending | string | |
| Age | descending | numeric | |
| Birthdate | ascending | datetime | yyyy-MM-dd |

## List Services

Use **List** services to retrieve, replace, or add elements in an Object List, Document List, or String List, including converting String Lists to Document Lists.

The following **List** services are available:

| Service | Description |
|---|---|
| addItemToVector | Adds an item or a list of items to a java.util.Vector object. |
| appendToDocumentList | Adds documents to a document list. |
| appendToStringList | Adds Strings to a String list. |
| sizeOfList | Returns the number of elements in a list. |
| stringListToDocumentList | Converts a String list to a document list. |
| vectorToArray | Converts a java.util.Vector object to an array. |

## addItemToVector

Adds an item or a list of items to a java.util.Vector object.

### Input Parameters

| | |
|---|---|
| *vector* | **java.util.Vector** Optional. The vector object to which you want to add an item or list of items. If no value is specified, the service creates a new java.util.Vector object to which the item(s) will be added. |
| *item* | **Object** Optional. Item to be added to the vector object.<br>**Note:** You can use either *item* or *itemList* to specify the input object. If both *item* and *itemList* input parameters are specified, the item as well as the list of items will be added to the vector object. |
| *itemList* | **Object[ ]** Optional. List of items to be added to the vector object. |
| *addNulls* | **String** Optional. Specifies whether a null item can be added to the vector object. Set to:<br>- `false` to prevent null values from being added to the vector object. This is the default.<br>- `true` to allow null values to be added to the vector object. |

### Output Parameters

| | |
|---|---|
| *vector* | **java.util.Vector** Updated vector object with the list of items added or an empty vector in case no items are added. |

### Usage Notes

Either of the optional input parameters, *item* or *itemList*, is required.

## appendToDocumentList

Adds documents to a document list.

### Input Parameters

| | |
|---|---|
| *toList* | **Document List** Optional. List to which you want to append documents. If you do not specify *toList*, the service creates a new list. |
| *fromList* | **Document List** Optional. Documents you want to append to the end of *toList*. |

*fromItem* **Document** Optional. Document you want to append to the end of *toList*. If you specify both *fromList* and *fromItem*, the service adds the document specified in *fromItem* after the documents in *fromList*.

### Output Parameters

*toList* **Document List** The *toList* document list with the documents in *fromList* and *fromItem* appended to it.

### Usage Notes

The documents contained in *fromList* and *fromItem* are not actually appended as entries to *toList*. Instead, references to the documents in *fromList* and *fromItem* are appended as entries to *toList*. Consequently, any changes made to the documents in *fromList* and *fromItem* also affect the resulting *toList*.

## appendToStringList

Adds Strings to a String list.

### Input Parameters

*toList* **String List** Optional. List to which you want to append Strings. If the value of *toList* is null, a null pointer exception error is thrown. If you do not specify *toList*, the service creates a new list.

*fromList* **String List** Optional. List of Strings to add to *toList*. Strings are added after the entries of *toList*.

*fromItem* **String** Optional. String you want to append to the end of *toList*. If you specify both *fromList* and *fromItem*, the service adds the String specified in *fromItem* after the Strings specified in *fromList*.

### Output Parameters

*toList* **String List** The *toList* String list with the Strings from *fromList* and *fromItem* appended to it.

### Usage Notes

The Strings contained in *fromList* and *fromItem* are not actually appended as entries to *toList*. Instead, references to the Strings in *fromList* and *fromItem* are appended as entries to *toList*. Consequently, any changes made to the Strings in *fromList* and *fromItem* also affect the resulting *toList*.

## sizeOfList

Returns the number of elements in a list.

### Input Parameters

*fromList* **Document List, String List, or Object List** Optional. List whose size you want to discover. If *fromList* is not specified, the service returns a *size* of 0.

### Output Parameters

*size* **String** Number of entries in *fromList*.

*fromList* **Document List, String List, or Object List** Original list.

### Usage Notes

For example, if *fromList* consists of:

*fromList*[0] = "a"

*fromList*[1] = "b"

*fromList*[2] = "c"

The result would be:

*size*="3"

## stringListToDocumentList

Converts a String list to a document list.

### Input Parameters

*fromList* **String List** Optional. List of Strings (a String[ ]) that you want to convert to a list of documents. If *fromList* is not specified, the service returns a zero length array for *toList*.

*key* **String** Optional. Key name to use in the generated document list.

### Output Parameters

*toList* **Document List** Resulting document list.

### Usage Notes

Creates a document list containing one document for each element in the *fromList*. Each document will contain a single String element named *key*.

For example, if *fromList* consists of:

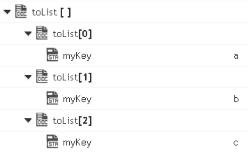
*fromList*[`0`] = "`a`"

*fromList*[`1`] = "`b`"

*fromList*[`2`] = "`c`"

*key* = "`myKey`"

The result would be:



```
toList [ ]
  toList[0]
    myKey          a
  toList[1]
    myKey          b
  toList[2]
    myKey          c
```

## vectorToArray

Converts a java.util.Vector object to an array.

### Input Parameters

*vector* **java.util.Vector** The object to be converted to an array.

*stronglyType* **String** Optional. If this option is specified, the service expects all items in the vector to have the same Java type as the first non-null item in the vector. If the service detects an item of a different type, an error appears.

Set to:

- `false` to convert the vector to an object array. This is the default.
- `true` to convert the vector to a strongly typed array holding the same type of objects.

### Output Parameters

*array* **Object[ ]** Converted object array.

## Math Services

Use **Math** services to perform mathematical operations on string-based numeric values. Services that operate on integer values use Java's long data type (64-bit, two's complement). Services that operate on float values use Java's double data type (64-bit IEEE 754). If extremely precise calculations are critical to your application, you should write your own Java services to perform math functions.

The following **Math** services are available:

| Service | Description |
|---|---|
| addObjects | Adds one java.lang.Number object to another and returns the sum. |
| divideObjects | Divides one java.lang.Number object by another (*num1*/*num2*) and returns the quotient. |
| min | Returns the smallest number from a list of numbers. |
| multiplyObjects | Multiplies one java.lang.Number object by another and returns the product. |
| subtractObjects | Subtracts one java.lang.Number object from another and returns the difference. |
| toNumber | Converts a string to numeric data type. |
| absoluteValue | Returns the absolute value of the input number. |
| addFloatList | Adds a list of floating point numbers (represented in a string list) and returns the sum. |
| addFloats | Adds one floating point number (represented as a String) to another and returns the sum. |
| addIntList | Adds a list of integers (represented in a String list) and returns the sum. |
| addInts | Adds one integer (represented as a String) to another and returns the sum. |
| divideFloats | Divides one floating point number (represented as a String) by another (*num1*/*num2*) and returns the quotient. |
| divideInts | Divides one integer (represented as a String) by another (*num1*/*num2*) and returns the quotient. |
| max | Returns the largest number from a list of numbers. |
| multiplyFloatList | Multiplies a list of floating point numbers (represented in a String list) and returns the product. |

| Service | Description |
|---|---|
| multiplyFloats | Multiples one floating point number (represented as String) by another and returns the product. |
| multiplyIntList | Multiplies a list of integers (represented in a String list) and returns the product. |
| multiplyInts | Multiplies one integer (represented as a String) by another and returns the product. |
| randomDouble | Returns the next pseudorandom, uniformly distributed double between 0.0 and 1.0. |
| roundNumber | Returns a rounded number. |
| subtractFloats | Subtracts one floating point number (represented as a String) from another and returns the difference. |
| subtractInts | Subtracts one integer (represented as a String) from another and returns the difference. |

## addObjects

Adds one java.lang.Number object to another and returns the sum.

### Input Parameters

| | |
|---|---|
| *num1* | **java.lang.Number** Number to add. See the Usage Notes for supported sub-classes. |
| *num2* | **java.lang.Number** Number to add. See the Usage Notes for supported sub-classes. |

### Output Parameters

| | |
|---|---|
| *value* | **java.lang.Number** Sum of the numeric values of *num1* and *num2*. |

### Usage Notes

This service accepts the following sub-classes of java.lang.Number: java.lang.Byte, java.lang.Double, java.lang.Float, java.lang.Integer, java.lang.Long, java.lang.Short.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type Double, the other is converted to Double.
- Otherwise, if either operand is of type Float, the other is converted to Float.
- Otherwise, if either operand is of type Long, the other is converted to Long.

- Otherwise, both operands are converted to type Integer.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

## divideObjects

Divides one java.lang.Number object by another (*num1*/*num2*) and returns the quotient.

### Input Parameters

| | |
|---|---|
| *num1* | **java.lang.Number** Number that is the dividend. See the Usage Notes for supported sub-classes. |
| *num2* | **java.lang.Number** Number that is the divisor. See the Usage Notes for supported sub-classes. |

### Output Parameters

| | |
|---|---|
| *value* | **java.lang.Number** Quotient of *num1* / *num2*. |

### Usage Notes

This service accepts the following sub-classes of java.lang.Number: java.lang.Byte, java.lang.Double, java.lang.Float, java.lang.Integer, java.lang.Long, java.lang.Short.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type Double, the other is converted to Double.
- Otherwise, if either operand is of type Float, the other is converted to Float.
- Otherwise, if either operand is of type Long, the other is converted to Long.
- Otherwise, both operands are converted to type Integer.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

## min

Returns the smallest number from a list of numbers.

### Input Parameters

| | |
|---|---|
| *numList* | **String List** List of numbers from which the smallest number is to be returned. |

### Output Parameters

*minValue* **String** Smallest number from the list of numbers.

## multiplyObjects

Multiplies one java.lang.Number object by another and returns the product.

### Input Parameters

*num1* **java.lang.Number** Number to multiply. See the Usage Notes for supported sub-classes.

*num2* **java.lang.Number** Number to multiply. See the Usage Notes for supported sub-classes.

### Output Parameters

*value* **java.lang.Number** Product of *num1* and *num2*.

### Usage Notes

This service accepts the following sub-classes of java.lang.Number: java.lang.Byte, java.lang.Double, java.lang.Float, java.lang.Integer, java.lang.Long, java.lang.Short.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type Double, the other is converted to Double.
- Otherwise, if either operand is of type Float, the other is converted to Float.
- Otherwise, if either operand is of type Long, the other is converted to Long.
- Otherwise, both operands are converted to type Integer.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

## subtractObjects

Subtracts one java.lang.Number object from another and returns the difference.

### Input Parameters

*num1* **java.lang.Number** Number. See the Usage Notes for supported sub-classes.

*num2* **java.lang.Number** Number to subtract from *num1*. See the Usage Notes for supported sub-classes.

### Output Parameters

*value* **java.lang.Number** Difference of *num1* - *num2*.

### Usage Notes

This service accepts the following sub-classes of java.lang.Number: java.lang.Byte, java.lang.Double, java.lang.Float, java.lang.Integer, java.lang.Long, java.lang.Short.

This service applies the following rules for binary numeric promotion to the operands. The following rules are applied in order:

- If either operand is of type Double, the other is converted to Double.
- Otherwise, if either operand is of type Float, the other is converted to Float.
- Otherwise, if either operand is of type Long, the other is converted to Long.
- Otherwise, both operands are converted to type Integer.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

## toNumber

Converts a string to numeric data type.

### Input Parameters

*num* **String** Number (represented as a string) to be converted to numeric format.

*convertAs* **String** Optional. Specifies the Java numeric data type to which the *num* parameter is to be converted.

Valid values for the *convertAs* parameter are `java.lang.Double`, `java.lang.Float`, `java.lang.Integer`,`java.math.BigDecimal`,`java.math.BigInteger`, `java.lang.Long`. The default value is `java.lang.Double`.

### Output Parameters

*num* **java.lang.Number** Converted numeric object.

## absoluteValue

Returns the absolute value of the input number.

### Input Parameters

*num* **String** Number whose absolute value is to be returned.

### Output Parameters

*positiveNumber* **String** Absolute value of the input number.

## addFloatList

Adds a list of floating point numbers (represented in a string list) and returns the sum.

### Input Parameters

*numList* **String List** Numbers (floating point numbers represented in a string list) to add.

### Output Parameters

*value* **String** Sum of the numbers in *numList*. If a sum cannot be produced, *value* contains one of the following:

| Value | Description |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, adding a number to infinity). |
| `NaN` | The computation produces a value that cannot be represented as a number (for example, any operation that uses NaN as input, such as 10.0 + NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results.

For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## addFloats

Adds one floating point number (represented as a String) to another and returns the sum.

### Input Parameters

*num1* **String** Number to add.

*num2* **String** Number to add.

*precision* **String** Optional. Number of decimal places to which the sum will be rounded. The default value is null.

### Output Parameters

*value* **String** Sum of the numbers in *num1* and *num2*. If a sum cannot be produced, *value* contains one of the following:

| **Value** | **Description** |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, adding a number to infinity). |
| `NaN` | The computation produces a value that cannot be represented as a number (for example, any operation that uses NaN as input, such as 10.0 + NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## addIntList

Adds a list of integers (represented in a String list) and returns the sum.

### Input Parameters

| | |
|---|---|
| *numList* | **String List** Numbers (integers represented as Strings) to add. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Sum of the numbers in *numList*. |

### Usage Notes

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## addInts

Adds one integer (represented as a String) to another and returns the sum.

### Input Parameters

| | |
|---|---|
| *num1* | **String** Number (integer represented as a String) to add. |
| *num2* | **String** Number (integer represented as a String) to add. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Sum of *num1* and *num2*. |

### Usage Notes

Ensure that the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Ensure that the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## divideFloats

Divides one floating point number (represented as a String) by another (*num1*/*num2*) and returns the quotient.

### Input Parameters

*num1* **String** Number (floating point number represented as a String) that is the dividend.

*num2* **String** Number (floating point number represented as a String) that is the divisor.

*precision* **String** Optional. Number of decimal places to which the quotient will be rounded. The default value is null.

### Output Parameters

*value* **String** The quotient of *num1* / *num2*. If a quotient cannot be produced, *value* contains one of the following:

| Value | Description |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, dividing a number by infinity). |
| `NaN` | The computation produces a value that cannot be represented as a number (for example, the result of an illegal operation such as dividing zero by zero or any operation that uses NaN as input, such as 10.0 + NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## divideInts

Divides one integer (represented as a String) by another (*num1*/*num2*) and returns the quotient.

### Input Parameters

*num1* **String** Number (integer represented as a String) that is the dividend.

*num2* **String** Number (integer represented as a String) that is the divisor.

### Output Parameters

*value* **String** The quotient of *num1* / *num2*.

### Usage Notes

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## max

Returns the largest number from a list of numbers.

### Input Parameters

*numList* **String List** List of numbers from which the largest number is to be returned.

### Output Parameters

*maxValue* **String** Largest number from the list of numbers.

## multiplyFloatList

Multiplies a list of floating point numbers (represented in a String list) and returns the product.

### Input Parameters

*numList* **String List** Numbers (floating point numbers represented as Strings) to multiply.

### Output Parameters

*value* **String** Product of the numbers in *numlist*. If a product cannot be produced, *value* contains one of the following:

| Value | Description |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |

| | |
|---|---|
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, multiplying a number by infinity). |
| `NaN` | The computation produces a value that cannot be represented as a number (for example, the result of an illegal operation such as multiplying zero by zero or any operation that uses NaN as input, such as 10.0 + NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## multiplyFloats

Multiples one floating point number (represented as String) by another and returns the product.

### Input Parameters

*num1* **String** Number (floating point number represented as a String) to multiply.

*num2* **String** Number (floating point number represented as a String) to multiply.

*precision* **String** Optional. Number of decimal places to which the product will be rounded. The default value is null.

### Output Parameters

*value* **String** Product of the numeric values of *num1* and *num2*. If a product cannot be produced, *value* contains one of the following:

| Value | Description |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, multiplying a number by infinity). |

| | |
|---|---|
| `NaN` | The computation produces a value that cannot be represented as a number (for example, the result of an illegal operation such as multiplying zero by zero or any operation that uses NaN as input, such as 10.0 + NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## multiplyIntList

Multiplies a list of integers (represented in a String list) and returns the product.

### Input Parameters

| | |
|---|---|
| *numList* | **String List** Numbers (floating point numbers represented as Strings) to multiply. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Product of the numbers in *numList*. |

### Usage Notes

Make sure the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## multiplyInts

Multiplies one integer (represented as a String) by another and returns the product.

### Input Parameters

| | |
|---|---|
| *num1* | **String** Number (integer represented as a String) to multiply. |
| *num2* | **String** Number (integer represented as a String) to multiply. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Product of *num1* and *num2*. |

### Usage Notes

Make sure the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## randomDouble

Returns the next pseudorandom, uniformly distributed double between 0.0 and 1.0.

Random number generators are often referred to as pseudorandom number generators because the numbers produced tend to repeat themselves over time.

### Input Parameters

None.

### Output Parameters

| | |
|---|---|
| *number* | **String** Generated random number. |

## roundNumber

Returns a rounded number.

### Input Parameters

| | |
|---|---|
| *num* | **String** Number to be rounded. |
| *numberOfDigits* | **String** Specifies the number of digits to which you want to round the number. |
| *roundingMode* | **String** Optional. Specifies the rounding method.<br>Valid values for the *roundingMode* parameter are `RoundHalfUp`, `RoundUp`, `RoundDown`, `RoundCeiling`, `RoundFloor`, `RoundHalfDown`, and `RoundHalfEven`. The default value is `RoundHalfUp`. |

### Output Parameters

*roundedNumber* **String** The rounded number.

## subtractFloats

Subtracts one floating point number (represented as a String) from another and returns the difference.

### Input Parameters

*num1* **String** Number (floating point number represented as a String).

*num2* **String** Number (floating point number represented as a String) to subtract from *num1*.

*precision* **String** Optional. Number of decimal places to which the difference will be rounded. The default value is null.

### Output Parameters

*value* **String** Difference of *num1* - *num2*. If a difference cannot be produced, *value* contains one of the following:

| Value | Description |
|---|---|
| `Infinity` | The computation produces a positive value that overflows the representable range of a float type. |
| `-Infinity` | The computation produces a negative value that overflows the representable range of a float type. |
| `0.0` | The computation produces a value that underflows the representable range of a float type (for example, subtracting a number from infinity). |
| `NaN` | The computation produces a value that cannot be represented as a number (for example, the result of an illegal operation such as multiplying zero by zero or any operation that uses NaN as input, such as 10.0 - NaN = NaN). |

### Usage Notes

Make sure the strings that are passed to the service in *num1*and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## subtractInts

Subtracts one integer (represented as a String) from another and returns the difference.

### Input Parameters

| | |
|---|---|
| *num1* | **String** Number (integer represented as a String). |
| *num2* | **String** Number (integer represented as a String) to subtract from *num1*. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Difference of *num1* - *num2*. |

### Usage Notes

Make sure the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Make sure the strings that are passed to the service in *num1* and*num2* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling addFloats in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

## MIME Services

Use MIME services to create MIME messages and extract information from MIME messages.

The following MIME services are available:

| Service | Function |
|---|---|
| addBodyPart | Adds a body part (header fields and content) to a specified MIME object. |
| addMimeHeader | Adds one or more header fields to a specified MIME object. |
| createMimeData | Creates a MIME object. |
| getBodyPartContent | Retrieves the content (payload) from the specified MIME object. |
| getBodyPartHeader | Returns the list of header fields for the specified body part. |

| Service | Function |
|---|---|
| getContentType | Returns the value of the Content-Type message header from the specified MIME object. |
| getEnvelopeStream | Generates an InputStream representation of a MIME message from a specified MIME object. |
| getMimeHeader | Returns the list of message headers from a specified MIME object. |
| getNumParts | Returns the number of body parts in the specified MIME object. |
| getPrimaryContentType | Returns the top-level portion of a MIME object's Content-Type value. |
| getSubContentType | Returns the sub-type portion of a MIME object's Content-Type value. |
| mergeHeaderAndBody | Concatenates the contents of the header and body mapped to the input. |

## addBodyPart

Adds a body part (header fields and content) to a specified MIME object.

### Input Parameters

*mimeData*

**Document** MIME object to which you want to add a body part. (This IData object is produced by createMimeData).

*content*

**java.io.InputStream or Object** Content that you want to add to the MIME object. *content* can be an InputStream or another MIME object. Use an InputStream to add an ordinary payload. Use a MIME object to add a payload that is itself a MIME message.

*isEnvStream*

**String** Flag that specifies whether *content* is to be treated as a MIME entity.

> **Important:**
> This parameter is only used if *content* is an InputStream.

Set this parameter to one of the following values:

- yes to treat *content* as a MIME entity. addBodyPart will strip out the header fields from the top of *content* and add them to *mimeData* as part headers. The remaining data will be treated as the payload.

> **Note:**addBodyPart assumes that all data up to the first blank line represents the entity's header fields.

- `no` to treat *content* as an ordinary payload.

*mimeHeader*

**Document** Specifies the part headers that you want to add with this body part. Key names represent the names of the header fields. The values of the keys represent the values of the header fields.

For example, if you wanted to add the following header fields:

```
X-Doctype: RFQ
X-Severity: 10
```

You would set *mimeHeader* as follows:

| Key | Value |
|---|---|
| *X-Doctype* | `RFQ` |
| *X-Severity* | `10` |

Be aware that the following MIME headers are automatically inserted by getEnvelopeStream when it generates the MIME message:

```
Message-ID
MIME-Version
```

Additionally, you use the *content*, *encoding*, and *description* parameters to set the following fields:

```
Content-Type
Content-Transfer-Encoding
Content-Description
```

If you set these header fields in *mimeHeader* and you create a single-part message, the values in *contenttype*, *encoding*, and *description,* if specified, will override those in *mimeHeader*. See usage notes.

*contenttype*

**String** Optional. The value of the Content-Type header for this body part. For single-part messages, this value overrides the Content-Type value in *mimeHeader*, if one is present. Defaults to text/plain.

See usage notes.

*encoding*

**String** Optional. Specifies how the body part is to be encoded for transport and sets the value of the

Content-Transfer-Encoding header. For single-part messages, this value overrides the Content-Transfer-Encoding value in *mimeHeader*, if one is present. Defaults to 7bit.

See usage notes.

> **Note:**
> This parameter determines how the payload is to be encoded for transport. When you add a payload to *mimeData*, it should be in its original format. The getEnvelopeStream service will perform the encoding (as specified by *encoding*) when it generates the final MIME message.

Set to:

- `7bit` to specify that *content* is 7-bit, line-oriented text that needs no encoding. This is the default.
- `8bit` to specify that *content* is 8-bit, line-oriented text that needs no encoding.

  > **Note:**
  > This encoding value is not recommended for messages that will be transported via SMTP over the Internet, because the data can be altered by intervening mail servers that can't accommodate 8-bit text. To safely transport 8-bit text, use quoted-printable encoding instead.

- `binary` to specify that *content* contains binary information that needs no encoding.

  > **Note:**
  > This encoding value is not recommended for messages that will be transported via SMTP over the Internet, because the data can be altered by intervening mail servers that can't accommodate binary data. To safely transport binary data, use base64 encoding instead.

- `quoted-printable` to specify that *content* contains 7 or 8-bit, line-oriented text that you want to encode using the quoted-printable encoding scheme.
- `base64` to specify that *content* contains an arbitrary sequence of octets that you want to encode using the base64 encoding scheme.
- `uuencode` to specify that *content* contains an arbitrary sequence of octets that you want to encode using the uuencode encoding scheme.

*description*

**String** Optional. Specifies the value of the `Content-Description` header for this body part.

*multipart*

**String** Optional. Flag that determines how addBodyPart behaves if *mimeData* already contains one or more body parts.

By default, addBodyPart simply appends a new body part to *mimeData* if it already contains a payload. (This allows you to construct multi-part messages.) However, you can override this behavior if you want to either replace the existing payload with the new body part or throw an exception under these circumstances (see *replace* parameter, below).

Set to:

- `yes` to append a new body part to *mimeData*. This is the default.
- `no` to replace the existing payload with the new body part. (Depending on the value of *replace*, this setting may cause addBodyPart to throw an exception.)

*replace*

**String** Optional. Flag that specifies whether addBodyPart replaces the existing payload or throws an exception when it receives a *mimeData* that already contains a payload. This parameter is only used when *multipart* is set to `no`.

Set to:

- `yes` to replace the existing payload with the new body part. This is the default.
- `no` to throw an exception.

## Output Parameters

*mimeData*

**Document** MIME object to which the body part was added.

## Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

The way in which the *contenttype* and *encoding* parameters are applied depends on whether the finished message is single-part or multipart.

For single-part messages:

- *contenttype* specifies the Content-Type for the entire MIME message. It overrides any value assigned to the Content-Type header in *mimeHeader*. If Content-Type is not specified in *contenttype* or *mimeHeader*, the value of the Content-Type header defaults to text/plain.

- *encoding* specifies the Content-Transfer-Encoding for the entire MIME message. It overrides any value assigned to the Content-Transfer-Encoding header in *mimeHeader*. If Content-Transfer-Encoding is not specified in *encoding* or *mimeHeader*, the value of the Content-Transfer-Encoding header defaults to 7bit.

For multipart messages:

- *contenttype* specifies the Content-Type for an individual body part. The Content-Type for the entire MIME message is automatically set to multipart/mixed, or to multipart/*subType* if a subtype was specified when the MIME object was created. See createMimeData.
- *encoding* specifies the Content-Transfer-Encoding for an individual body part. The Content-Transfer-Encoding header in *mimeHeader*, if present, specifies the encoding for the entire MIME message. If Content-Transfer-Encoding is not specified in *mimeHeader*, or if the specified value is not valid for a multipart message, the value of the Content-Transfer-Encoding header defaults to 7bit. (7bit, 8bit, and binary are the only encoding values valid for multipart messages.)

## addMimeHeader

Adds one or more header fields to a specified MIME object.

### Input Parameters

*mimeData*

**Document** MIME object to which you want the header fields added. (This IData object is produced by createMimeData.)

*mimeHeader*

**Document** Header fields that you want to add to the MIME object. Key names represent the names of the header fields. The values of the keys represent the values of the header fields. For example, to add the following header fields:

```
X-Doctype: RFQ
X-Severity: 10
```

You would set *mimeHeader* as follows:

| Key | Description |
|---|---|
| *X-Doctype* | `RFQ` |
| *X-Severity* | `10` |

Be aware that the following MIME headers are automatically inserted by getEnvelopeStream when it generates the MIME message:

```
Message-ID
MIME-Version
```

If you set these values in *mimeHeader*, getEnvelopeStream will overwrite them at run time.

### Output Parameters

| | |
|---|---|
| *mimeData* | **Document** MIME object to which the header fields were added. |

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

If you add MIME headers before you add multiple body parts, the header fields will be added to each of the body parts. If you do not want this behavior, either drop *mimeHeader* from the pipeline immediately after you execute addMimeHeader, or invoke addMimeHeader after you've added all body parts to the MIME object.

Be aware that the *contenttype* and *encoding* parameters used by the addBodyPart service will override any Content-Type or Content-Transfer-Encoding settings in *mimeData*. Moreover, in certain cases, the getEnvelopeStream will override these settings when it generates a multipart message. For information about how the Content-Type or Content-Transfer-Encoding headers are derived at run time, see the Usage Notes under addBodyPart.

## createMimeData

Creates a MIME object.

If no *input* parameter is passed to this service, the service creates an empty MIME object. Otherwise, the service creates a MIME object containing the elements (header fields and content) from the MIME message in *input*.

- If you are building a MIME message, you use this service to create an empty MIME object. You populate the empty MIME object with header fields and content, and then pass it to getEnvelopeStream, which produces the finished MIME message.
- If you are extracting data from a MIME message, you use this service to parse the original MIME message into a MIME object so that you can extract its header fields and content using other services.

### Input Parameters

| | |
|---|---|
| *input* | **java.io.InputStream** Optional. MIME entity you want to parse. If *input* is not provided, createMimeData creates an empty MIME object. |
| *mimeHeader* | **Document** Optional. Specifies header fields that you want to add to the MIME object. Key names represent the names of the header fields. The values of the keys represent the values of the header fields.<br>**Note:** |

This parameter is ignored when *input* is passed to this service.

For example, if you wanted to add the following header fields:

```
X-Doctype: RFQ
X-Severity: 10
```

You would set *mimeHeader* as follows:

| Key | Value |
|---|---|
| *X-Doctype* | `RFQ` |
| *X-Severity* | `10` |

Be aware that the following MIME headers are automatically inserted by getEnvelopeStream when it generates the MIME message:

```
Message-ID
MIME-Version
```

If you set these values in *mimeHeader*, getEnvelopeStream will overwrite them at run time.

*subType*

**String** Optional. String that specifies the subtype portion of the Content Type header, when the message is a multipart message and \you want something other than the default value of `mixed`. For example, if you want the Content Type header to be `multipart/related` in the resulting message, set *subType* to `related`.

*subType* is ignored if the resulting message is not a multipart message.

*decodeHeaders*

**String** Optional. Specifies how the MIME header is to be decoded.

Set to:

- `" "`(empty String) to decode headers based on the value of the global watt property watt.server.mime.decodeHeaders. This is the default.
- `NONE` to specify that the MIME header or body part headers do not need decoding.
- `ONLY_MIME_HEADER` to decode the MIME header only.
- `ONLY_BODY_PART_HEADERS` to decode the body part headers only.
- `BOTH` to decode the MIME header and the body part headers.

## Output Parameters

*mimeData*

**Document** MIME object. If *input* was passed to createMimeData, *mimeData* will contain the parsed MIME message. If *input* was not passed to createMimeData, *mimeData* will be empty.

*encrypted*

**String** Conditional. Indicates whether input was an encrypted message. This parameter is not present when the service creates a new, empty MIME object. A value of:

- `true` indicates that the message is encrypted (the original message stream is in *stream*).
- `false` indicates that the message is not encrypted.

*signed*

**String** Conditional. Flag whose value indicates whether *input* was a signed message. This parameter is not present when the service creates a new, empty MIME object. A value of:

- `true` indicates that the message is signed (the original message stream is in *stream*).
- `false` indicates that the message is not signed.

*certsOnly*

**String** Conditional. Flag whose value indicates whether *input* contained only digital certificates. This parameter is not present when the service creates a new, empty MIME object. A value of:

- `true` indicates that the message contains only certificates.
- `false` indicates that the message contains a regular payload.

*stream*

**java.io.InputStream** Conditional. InputStream containing the original MIME message from *input*. This parameter is present only when *input* is an S/MIME message.

## Usage Notes

All of the other MIME services operate on the *mimeData* IData object produced by this service. They do not operate directly on MIME message streams.

**Important:**
You can examine the contents of *mimeData* during testing and debugging. However, because the internal structure of *mimeData* is subject to change without notice, **do not** explicitly set or map data to/from these elements in your service. To manipulate or access the contents of *mimeData*, use **only** the MIME services that is provided.

## getBodyPartContent

Retrieves the content (payload) from the specified MIME object.

You use this service for both single-part and multi-part messages.

To retrieve content from a multi-part message, you set the *index* (to select the part by index number) or *contentID* (to select the part by *contentID* value) parameter to specify the body part whose content you want to retrieve. To get the content from a single-part message, you omit the *index* and *contentID* parameters or set *index* to `0`.

### Input Parameters

*mimeData*

**Document** MIME object whose content you want to retrieve. (This IData object is produced by createMimeData.)

*index*

**String** Optional. Index number of the body part whose content you want to retrieve (if you want to retrieve the content from a specific body part). The first body part is index number zero.

> **Note:**
> If *contentID* is specified, *index* is ignored.

*contentID*

**String** Optional. Value of the `Content-ID` header field of the body part whose content you want to retrieve (if you want to retrieve the payload from a specific body part).

### Output Parameters

*content*

**IData** The payload of the specified body part.

*encrypted*

**String** Flag whose value indicates whether *content* is an encrypted MIME message. A value of:

- `true` indicates that *content* is an encrypted message.
- `false` indicates that *content* is not an encrypted message.

*signed*

**String** Flag indicating whether *content* is a signed MIME message. A value of:

- `true` indicates that *content* is a signed MIME message.
- `false` indicates that *content* is not a signed MIME message.

*certsOnly*

**String** Flag whose value indicates whether *content* is a certs-only MIME message. A value of:

- `true` indicates that *content* is a certs-only message.

- `false` indicates that *content* is not a certs-only message.

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

If you omit *index* or *contentID* when retrieving content from a multi-part message, getBodyPartContent returns the payload from the first body part. If you use *index* or *contentID* to select a body part that does not exist in *mimeData*, *content* will be null.

## getBodyPartHeader

Returns the list of header fields for the specified body part.

### Input Parameters

*mimeData*

**Document** MIME object whose message headers you want to retrieve. (This IData object is produced by createMimeData).

*index*

**String** Optional. Index number of the body part whose header fields you want to retrieve. The first body part is index zero.

**Note:**
If *contentID* is specified, *index* is ignored.

*contentID*

**String** Optional. Value of the `Content-ID` header field of the body part whose header fields you want to retrieve.

*decodeHeaders*

**String** Conditional. Flag whose value indicates whether to decode encoded headers in the MIME object. Set to:

- `true` to indicate that the headers should be decoded.
- `false` to indicate that the headers should not be decoded. This is the default.

### Output Parameters

*mimeHeader*

**Document** IData object containing the message headers. Key names represent the names of the header fields. The value of a key represents the value of that header field.

For example, if the original message contained the following message header fields:

```
Content-Type: text/xml
X-Doctype: RFQ
X-Severity: 0
```

get Body Part Header would return the following IData object:

| Key | Value |
|---|---|
| *Content-Type* | `text/xml` |
| *X-Doctype* | `RFQ` |
| *X-Severity* | `0` |

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

If you omit *index* or *contentID*, getBodyPartHeader returns the message headers from the first body part. If you use *index* or *contentID* to select a body part that does not exist in *mimeData*, *content* will be null.

## getContentType

WmPublic. Returns the value of the Content-Type message header from the specified MIME object.

### Input Parameters

| | |
|---|---|
| *mimeData* | **Document** MIME object whose Content-Type you want to discover. (This IData object is produced by createMimeData.. |

### Output Parameters

*contentType*

**String** Value of the MIME object's Content-Type header field. Note that this service returns only the media type and subtype portion of this header field's value. It does not return any parameters the value may include. For example, if the message's Content-Type header were:

```
Content-Type: text/plain;charset=UTF8
```

*contentType* would contain:

```
text/plain
```

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

## getEnvelopeStream

Generates an InputStream representation of a MIME message from a specified MIME object.

## Input Parameters

| | |
|---|---|
| *mimeData* | **Document** MIME object from which you want to generate the MIME message. (This IData object is produced by createMimeData.) |
| *index* | **String** Optional. Index number of the body part for which you want to generate the MIME message (if you want to generate the message from a specific body part). The first body part is index number zero. |
| *contentID* | **String** Optional. Value of the `Content-ID` header field of the body part from which you want to generate the MIME message (if you want to generate the message from a specific body part).<br>**Note:**<br>If *index* is specified, *contentID* is ignored. |
| *suppressHeaders* | **String List** Optional. Names of header fields that are to be omitted from message. You can use this option to exclude header fields that getEnvelopeStream generates by default, such as `Content-Type` and `content-encoding`. |
| *createMultipart* | **String** Optional. Specifies whether a multipart message is to be created, even if *mimeData* contains only one body part. Set to:<br>- `yes` to create a multipart message (Content-Type message header is set to "multipart/mixed").<br>- `no` to create a message based on the number of body parts in *mimeData*. This is the default.<br>  - If the message contains only one body part, Content-Type is set according to the *contenttype* setting specified when that body part was added to *mimeData*.<br>  - If the message contains multiple body parts, Content-Type is automatically set to "multipart/mixed." |

## Output Parameters

| | |
|---|---|
| *envStream* | **java.io.InputStream** The MIME message as an InputStream. |

## Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

If you omit *index* or *contentID*, getEnvelopeStream generates the MIME message from the entire contents of the *mimeData*. If you use *index* or *contentID* to select a body part that does not exist in *mimeData*, *content* will be null.

getEnvelopeStream automatically inserts the `MIME-Version and Message-ID` message headers into the MIME message it puts into *envStream*.

## getMimeHeader

WmPublic. Returns the list of message headers from a specified MIME object.

### Input Parameters

*mimeData* **Document** MIME object whose message headers you want to retrieve. (This IData object is produced by createMimeData).

### Output Parameters

*mimeHeader* **Document** Conditional. An IData object containing the message headers. Key names represent the names of the header fields. The value of a key represents the value of the header fields.

For example, if the original message contained the following message header fields:

```
Message-ID: <002e01c0f150$6f33010a@sgx.com>
From: "Purch01@GSX.com" <Purch01@GSX.com>To:
<EXPEst@exprint.com>
MIME-Version: 1.0
Content-Type: text/xml
X-Doctype: RFQ
X-Severity: 0
```

getMimeHeader would return the following:

| Key | Value |
|---|---|
| `Message-ID` | `<002e01c0f150$6f33010a@sgx.com>` |
| `From` | `"Purch01@GSX.com"`<br>`<Purch01@GSX.com>` |
| `To` | `<EXPEst@exprint.com>` |
| `MIME-Version` | `1.0` |
| `Content-Type` | `text/xml` |
| `X-Doctype` | `RFQ` |
| `X-Severity` | `0` |

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

## getNumParts

Returns the number of body parts in the specified MIME object.

### Input Parameters

*mimeData* **Document** MIME object whose parts you want to count. (This IData object is produced by createMimeData).

### Output Parameters

*numParts* **String** The number of body parts in the MIME object.

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

## getPrimaryContentType

Returns the top-level portion of a MIME object's Content-Type value.

### Input Parameters

*mimeData* **Document** MIME object whose Content-Type you want to discover. (This IData object is produced by createMimeData).

### Output Parameters

*primContentType* **String** Message's top-level Content-Type. For example, if the message's Content-Type header were:

```
Content-Type: multipart/mixed
```

*primContentType* would contain:

```
multipart
```

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

## getSubContentType

Returns the sub-type portion of a MIME object's Content-Type value.

### Input Parameters

*mimeData*

**Document** MIME object whose sub-type you want to discover. (This IData object is produced by createMimeData).

### Output Parameters

*subContentType*

**String** Message's sub-type. For example, if the message's Content-Type header were:

```
Content-Type: multipart/mixed
```

*subContentType* would contain:

```
mixed
```

### Usage Notes

This service operates on the MIME object (*mimeData*) produced by createMimeData.

## mergeHeaderAndBody

Concatenates the contents of the header and body mapped to the input.

You can use this service to reassemble the message into its original form so that it can be used as input to the createMimeData service (or any other service that requires the entire http response as an InputStream).

### Input Parameters

*headerLines*

**Document** IData object containing the message headers. (The message headers are returned in the *lines* document inside the *header* output parameter).

*body*

**Document** IData object containing the body of the message. This document must contain the body of the message in one of the following keys:

| Key | Description |
|---|---|
| *bytes* | **byte[ ]** Optional. Body of the message. |
| *stream* | **java.io.InputStream** Optional. The body of the message. |

### Output Parameters

| | |
|---|---|
| *stream* | **java.io.InputStream** InputStream containing the reassembled tap message. |

### Usage Notes

Use this service to merge the Headers and Body to get the original MIME message.

## Storage Services

Use **Storage** services to insert, retrieve, update, and remove entries from a data store.

When using the storage services, keep in mind that the short-term store is not intended to be used as a general-purpose storage engine. Rather, it is primarily provided to support shared storage of application resources and transient data in Integration Cloud. It is recommended not to use the short-term store to process high volumes, large data records, or to permanently archive records.

**Note:**
User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.

**Note:**
These services are a tool for maintaining state information in the short-term store. It is up to the developer of the Integration to make sure that the Integration keeps track of its state and correctly handles restarts.

**Locking Considerations**

The following sections describe in general how the storage services handle locking requests.

*Entry Locking*

To maintain data integrity, the short-term store uses locking to ensure that multiple threads do not modify the same entry at the same time. For insertions and removals, the short-term store sets and releases the lock. For updates, the client must set and release the lock. Using locking improperly, that is, creating a lock but not releasing it, can cause deadlocks in the short-term store.

The following guidelines can help you avoid short-term store deadlocks:

- Release locks in the thread through which they were set. In other words, you cannot set a lock in one thread and release it in another. The safest way to do this is to release each lock in the Integration that acquired it.
- Unlock entries before the Integration completes. Entries remain locked until released using a put or an explicit unlock. To accomplish this, always pair a call to get or lock with a call to put or unlock so that every lock is followed by an unlock. In addition, use a try-catch pattern in your Integration so that an exception does not prevent the Integration from continuing and releasing the lock.

*Data Store Locking*

When a storage service locks an entry, the service also implicitly locks the data store in which the entry resides. This behavior prevents another thread from deleting the entire data store and the entries it contains while your thread is working with the entry. When the locked entry is unlocked, the implicit lock on the data store is also released.

Be careful when explicitly unlocking data stores. Consider the following example:

1. User_A locks an item. This creates two locks: an explicit lock on the entry, and an implicit lock on the data store.
2. User_A later unlocks the data store explicitly while still holding the lock on the entry.
3. User_B locks, then deletes the data store, including the entry locked by User_A in the first step.

When User_A explicitly unlocked the data store in step 2, User_B was able to delete the entry the User_A was working with.

*Automatic Promotion to Exclusive Lock*

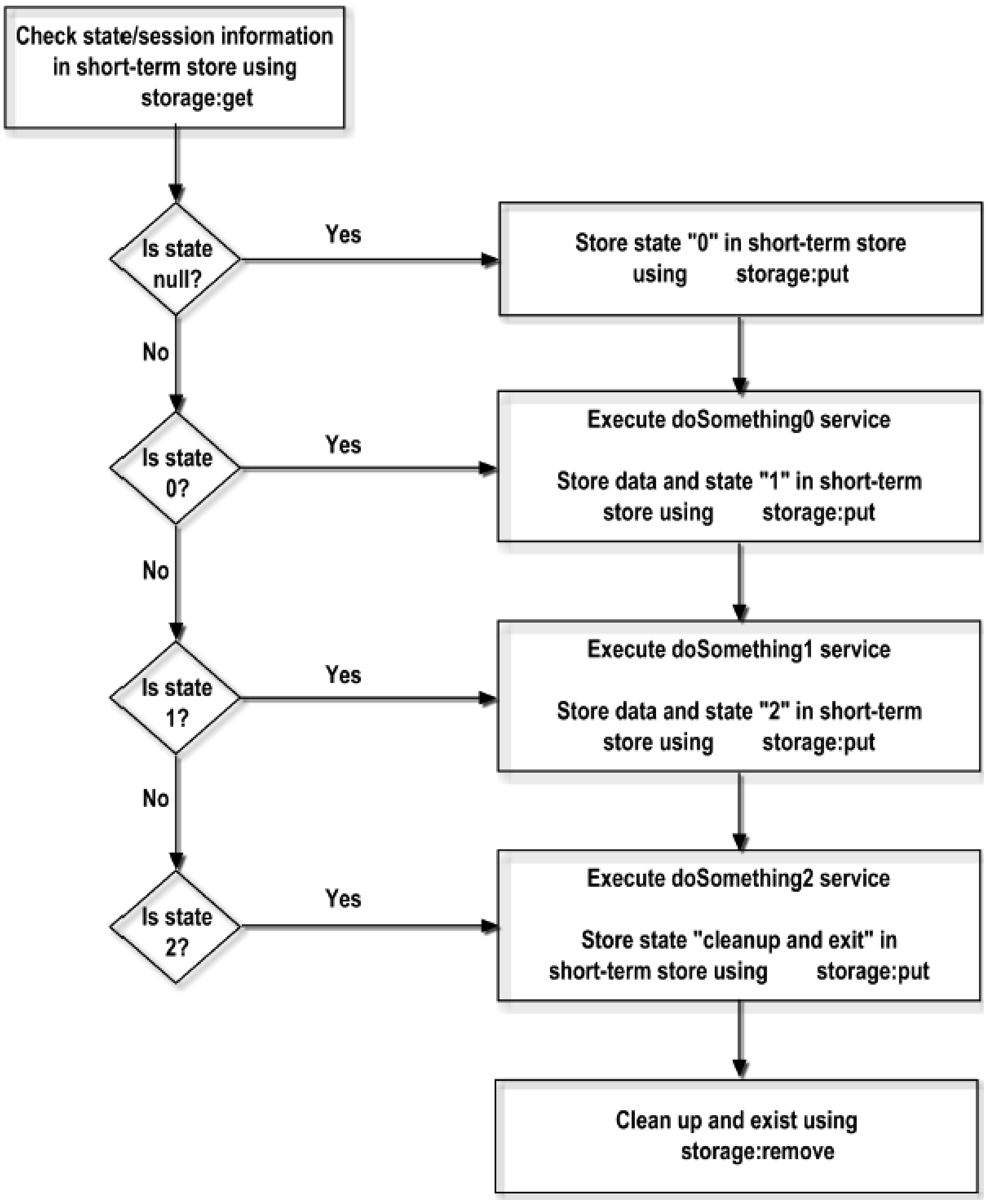
If a storage service tries to acquire an exclusive lock on an object, but finds a shared lock from the same thread already in place on the object, the service will try to promote the lock to an exclusive lock.

If a storage service that requires an exclusive lock encounters a shared or exclusive lock held by another thread, it will wait until the object becomes available. If the object remains locked for the period specified by the *waitlength* parameter passed by the service, the service will fail.

*Sample Integration for Checkpoint Restart*

The following diagram shows how to create checkpoint restarts into your Integrations. It explains the logic of an Integration and shows where the various storage services are used to achieve checkpoint restarts.

**Logic to achieve checkpoint restart**


Check state/session information in short-term store using storage:get
Is state null?
Yes
Store state "0" in short-term store using storage:put
No
Is state 0?
Yes
Execute doSomething0 service
Store data and state "1" in short-term store using storage:put
No
Is state 1?
Yes
Execute doSomething1 service
Store data and state "2" in short-term store using storage:put
No
Is state 2?
Yes
Execute doSomething2 service
Store state "cleanup and exit" in short-term store using storage:put
Clean up and exist using storage:remove

The following **Storage** services are available:

| Element | Package and Description |
|---|---|
| add | Inserts a new entry into a data store. |
| deleteStore | Deletes a data store and all its contents. Any data in the data store is deleted. If the data store does not exist, the service takes no action. |
| get | Retrieves a value from a data store and locks the entry and the data store on behalf of the thread that invoked the service. |
| keys | Obtains a list of all the keys in a data store. |
| lock | Locks an entry and/or data store on behalf of the thread invoking this service. |
| put | Inserts or updates an entry in a data store. If the key does not exist in the data store, the entry is inserted. |
| remove | Removes an entry from a data store. |
| unlock | Unlocks an entry or a data store. |

## add

Inserts a new entry into a data store.

If the key already exists in the data store, the service does nothing.

### Input Parameters

*storeName* **String** Name of the data store in which to insert the entry.

*key* **String** Key under which the entry is to be inserted.

*value* **Document** Value to be inserted.

### Output Parameters

*result* **String** Flag indicating whether the entry was successfully added. A value of:

- `true` indicates that the new entry was inserted successfully.
- `false` indicates that the entry was not inserted (usually because an entry for *key* already exists).

*error* **String** Error message generated while inserting the new entry into the data store.

## deleteStore

Deletes a data store and all its contents. Any data in the data store is deleted. If the data store does not exist, the service takes no action.

### Input Parameters

| | |
|---|---|
| *storeName* | **String** Name of the data store to delete. |
| *waitLength* | **String** Optional. Length of time, in milliseconds, that you want to wait for this data store to become available for deletion if it is already locked by another thread. |

### Output Parameters

| | |
|---|---|
| *count* | **String** Number of data store entries that were deleted. If the store does not exist, this value is 0. |

### Usage Notes

This service obtains an exclusive lock on the data store, but no locks on the individual entries in the data store. If this service finds a shared lock from the same thread on the data store, the service will automatically promote the lock to an exclusive lock. The exclusive lock prevents other threads from acquiring locks on the data store or entries within the data store during the delete operation.

## get

Retrieves a value from a data store and locks the entry and the data store on behalf of the thread that invoked the service.

**Important:**
This service does not automatically release the lock on the data store or entry after performing the get operation, so you need to ensure that the lock is released by calling the put or unlock services. If you do not release the lock, Integration Cloud will release the lock at the end of the Integration execution.

### Input Parameters

| | |
|---|---|
| *storeName* | **String** Name of the data store from which you want to retrieve the entry. |
| *key* | **String** Key of the entry whose value you want to retrieve. |
| *waitLength* | **String** Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread. |

*lockMode*

**String** Optional. Type of lock you want to place on the entry. Set to:

- `Exclusive` to prevent other threads from reading or updating the entry while you are using it. The service also obtains a shared lock on the data store. An exclusive lock on an entry allows you to modify the entry.
- `Read` is obsolete. If this value is specified, the service obtains a shared lock.
- `Share` to prevent other threads from obtaining an exclusive lock on the entry. The service also obtains a shared lock on the data store. A shared lock on an entry allows you to read, but not modify, the entry. This is the default.

### Output Parameters

*value*

**Document** Retrieved entry. If the requested entry does not exist, the value of this parameter is null.

### Usage Notes

If you request an exclusive lock and the service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock on the entry to an exclusive lock.

When this service locks an entry, it also acquires a shared lock on the associated data store to prevent another thread from deleting the data store, and the entries it contains, while your thread has the entry locked.

When storing and retrieving the flow state in the short-term store for checkpoint restart purposes, ensure that the value of *key* is unique to the transaction.

## keys

Obtains a list of all the keys in a data store.

### Input Parameters

*storeName*

**String** Name of the data store from which you want to obtain a list of keys.

### Output Parameters

*keys*

**String List** Keys for the data store specified in *storeName*.

# lock

Locks an entry and/or data store on behalf of the thread invoking this service.

**Important:**
When you lock an entry or data store using this service, you must release the lock by using a put or an explicit unlock. If you do not release the lock, Integration Cloud will release the lock at the end of the Integration execution.

**Important:**
Be careful when releasing locks with the unlock service. If you release a lock on a data store, another thread can obtain a lock on the data store and delete it, and the entries it contains, even if your thread still has locks on one or more of the entries.

## Input Parameters

*storeName*

**String** Name of the data store containing the entry.

*key*

**String** Optional. Key of the entry that you want to lock.

If *key* is not supplied and you request:

- A shared lock, the service obtains a shared lock on the data store, allowing other threads to read and modify entries, but not to delete them.
- An exclusive lock, the service obtains an exclusive lock on the data store, preventing other threads from locking the data store and the entries, thereby preventing those threads from reading, modifying, or deleting the entries or the data store.

If both *storeName* and *key* are specified and you request:

- A shared lock, the service obtains a shared lock on the data store and the entry.
- An exclusive lock, the service obtains a shared lock on the data store and an exclusive lock on the entry.

*waitLength*

**String** Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread.

*lockMode*

**String** Optional. Type of lock you want to place on the entry or data store. Set to:

- `Exclusive` to prevent other threads from obtaining a lock on the data store or entry.

  An exclusive lock on an entry allows you to modify the entry, and prevents other threads from reading or modifying the entry.

An exclusive lock on a data store also locks the entries in the data store. In addition, an exclusive lock on a data store allows you to delete the data store.

- `Read` is obsolete. If this value is specified, the service obtains a shared lock.
- `Share` to prevent other threads from obtaining an exclusive lock on an entry or a data store. A shared lock on an entry allows you to read, but not modify, the entry. A shared lock on a data store prevents another thread from deleting the data store. This is the default.

### Output Parameters

None.

### Usage Notes

If you have not specified a *key*, and your Integration does not invoke put or unlock, or your Integration throws an exception before invoking put or unlock, the entire data store remains locked.

If the key does not exist in the data store at the time your Integration executes, the lock service inserts the key with an empty value and takes the lock on the entry.

If you request an exclusive lock on an entry, the service obtains an exclusive lock on the entry and a shared lock on the data store. If this service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock on the entry to an exclusive lock.

If you request a shared lock on an entry, the service obtains a shared lock on the entry and a shared lock on the data store.

If you request a shared lock on an entry or a data store and this service finds an exclusive lock from the same thread, the existing exclusive lock will be reused. The exclusive lock will not be demoted to a shared lock.

If you request an exclusive lock on a data store, and this service finds a shared lock from the same thread on the data store, the service will automatically promote the shared lock on the data store to an exclusive lock.

## put

Inserts or updates an entry in a data store. If the key does not exist in the data store, the entry is inserted.

If the requested entry is not currently locked by the thread that invoked this service, the put service will automatically attempt to lock the entry for the duration of the put operation.

The service obtains an exclusive lock on the entry and a shared lock on the data store. If the service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock to an exclusive lock.

This service releases the lock when the put operation has completed.

### Input Parameters

*storeName* **String** Name of the data store into which you want to insert or update the entry.

*value* **Document** Value to be inserted or updated.

*waitLength* **String** Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread. If the wait length expires before a lock is obtained, the service fails and throws an exception.

This parameter is used only when your service did not explicitly lock the entry beforehand.

*key* **String** Key where you want to insert or update the entry.

### Output Parameters

*error* **String** Error message generated while inserting the new entry into the data store.

### Usage Notes

When storing and retrieving the flow state in the short-term store for checkpoint restart purposes, ensure that the value of *key* is unique to the transaction.

## remove

Removes an entry from a data store. This service obtains an exclusive lock on the entry and a shared lock on the data store.

### Input Parameters

*storeName* **String** Name of the data store from which to remove an entry.

*key* **String** Key of the entry that you want to remove.

*waitLength* **String** Optional. Length of time, in milliseconds, that you want to wait for this entry to become available for deletion if it is already locked by another thread.

### Output Parameters

*result*

**String** Flag indicating whether the entry was successfully removed. A value of:

- `true` indicates that the entry was removed successfully.
- `false` indicates that the entry was not removed (usually because an entry for key does not exist).

## unlock

Unlocks an entry or a data store.

When an Integration retrieves an entry using the get service, the entry is locked to prevent modification by other users before the Integration completes. The entry remains locked until the lock owner invokes a put service. To unlock a service without using the put service, use the unlock service.

In addition, if an Integration uses the lock service to lock an entry or data store, you must use the unlock or put service to release the lock.

**Important:**
Be careful when releasing locks with this service. If you release a lock on a data store, another thread can obtain a lock on the data store and delete it, and the entries it contains, even if the original thread still has locks on one or more of the entries.

### Input Parameters

*storeName*

**String** Name of the data store in which to unlock an entry.

*key*

**String** Optional. Key of the entry that you want to unlock. If *key* is not supplied, the lock will be removed from the data store specified in *storeName*, but any locks on entries in the data store will remain.

### Output Parameters

None.

## String Services

Use **String** services to perform string manipulation and substitution operations.

The following **String** services are available:

| Service | Description |
|---|---|
| HTMLDecode | Replaces HTML character entities with native characters. |
| HTMLEncode | Replaces HTML-sensitive characters with equivalent HTML character entities. |
| base64Decode | Decodes a Base-64 encoded string into a sequence of bytes. |
| base64Encode | Converts a sequence of bytes into a Base64-encoded String. |
| bytesToString | Converts a sequence of bytes to a String. |
| concat | Concatenates two strings. |
| indexOf | Returns the index of the first occurrence of a sequence of characters in a string. |
| length | Returns the length of a string. |
| lookupDictionary | Looks up a given key in a hash table and returns the string to which that key is mapped. |
| makeString | Builds a single string by concatenating the elements of a String List. |
| messageFormat | Formats an array of strings into a given message pattern. |
| numericFormat | Formats a number into a given numeric pattern. |
| objectToString | Converts an object to string representation using the Java toString() method of the object. |
| padLeft | Pads a string to a specified length by adding pad characters to the beginning of the string. |
| padRight | Pads a string to a specified length by adding pad characters to the end of the string. |
| replace | Replaces all occurrences of a specified substring with a substitute string. |
| stringToBytes | Converts a string to a byte array. |
| substring | Returns a substring of a given string. |
| tokenize | Tokenizes a string using specified delimiter characters and generates a String List from the resulting tokens. |
| toLower | Converts all characters in a given string to lowercase. |
| toUpper | Converts all characters in a given string to uppercase. |
| trim | Trims leading and trailing white space from a given string. |

| Service | Description |
|---|---|
| URLDecode | Decodes a URL-encoded string. |
| URLEncode | URL-encodes a string. |
| fuzzyMatch | A given string is not exactly matched against a set of strings. If the match is above similarityThreshold, it returns the matchedValue. If more than one string has not exactly matched, then the first matched string is returned. |
| isNumber | Determines whether the contents of a string can be converted to a float value. |
| isAlphanumeric | Determines whether a string consists entirely of alphanumeric characters (in the ranges A–Z, a–z, or 0–9). |
| isNullOrBlank | Checks a string for a null or a blank value. |
| isDate | Determines whether a string follows a specified date pattern. |
| substitutePipelineVariables | Replaces a pipeline variable with its corresponding value. |
| compareStrings | Performs a case-sensitive comparison of two strings, and indicates whether the strings are identical. |

## HTMLDecode

Replaces HTML character entities with native characters.

Specifically, the service:

| Replaces this HTML character entity... | With... |
|---|---|
| `&gt;` | `>` |
| `&lt;` | `<` |
| `&amp;` | `&` |
| `&quot;` | `"` |

### Input Parameters

| | |
|---|---|
| *inString* | **String** An HTML-encoded String. |

### Output Parameters

*value* **String** Result from decoding the contents of *inString*. Any HTML character entities that existed in *inString* will appear as native characters in *value*.

## HTMLEncode

Replaces HTML-sensitive characters with equivalent HTML character entities.

Specifically, this service:

| Replaces this native language character... | With... |
|---|---|
| `>` | `&gt;` |
| `<` | `&lt;` |
| `&` | `&amp;` |
| `"` | `&quot;` |
| `‘` | `&#39` |

These translations are useful when displaying text in an HTML context.

### Input Parameters

*inString* **String** The character you want to encode in HTML.

### Output Parameters

*value* **String** Result from encoding the contents of *inString*. Any HTML-sensitive characters that existed in *inString*, for example, > or &, will appear as the equivalent HTML character entities in *value*.

## base64Decode

Decodes a Base-64 encoded string into a sequence of bytes.

### Input Parameters

*string* **String** A Base64-encoded String to decode into bytes.

### Output Parameters

| | |
|---|---|
| *value* | **byte[ ]** The sequence of bytes decoded from the Base64-encoded String. |
| *encoding* | **String** Optional. Specifies the encoding method. Default value is `ASCII`. |

## base64Encode

Converts a sequence of bytes into a Base64-encoded String.

### Input Parameters

| | |
|---|---|
| *bytes* | **byte[ ]** Sequence of bytes to encode into a Base64-encoded String. |
| *useNewLine* | **String** Optional. Flag indicating whether to retain or remove the line breaks. Set to:<br>■ `true` to retain the line breaks. This is the default.<br>■ `false` to remove the line breaks. |
| *encoding* | **String** Optional. Specifies the encoding method. Default value is `ASCII`. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Base64-encoded String encoded from the sequence of bytes. |

### Usage Notes

By default, the base64Encode service inserts line breaks after 76 characters of data, which is not the canonical lexical form expected by implementations such as MTOM. You can use the *useNewLine* parameter to remove the line breaks.

## bytesToString

Converts a sequence of bytes to a String.

### Input Parameters

| | |
|---|---|
| *bytes* | **byte[ ]** Sequence of bytes to convert to a String. |
| *encoding* | **String** Optional. Name of a registered, IANA character set (for example, `ISO-8859-1`). If you specify an unsupported encoding, the system throws an exception. |

To use the default encoding, set *encoding* to `autoDetect`.

*ignoreBOMChars* **String** Optional. Flag indicating whether or not the byte order mark (BOM) characters in the input sequence of bytes are removed before converting the byte array to string. Set to:

- `true` to remove the byte order mark (BOM) characters before converting the input sequence of bytes to string, if the byte array contains BOM characters.
- `false` to include the byte order mark (BOM) characters while converting the input sequence of bytes to string. The default is `false`.

### Output Parameters

*string* **String** String representation of the contents of *bytes*.

## concat

Concatenates two strings.

### Input Parameters

*inString1* **String** String to which you want to concatenate another string.

*inString2* **String** String to concatenate to *inString1*.

### Output Parameters

*value* **String** Result of concatenating *inString1* with *inString2* (*inString1* + *inString2*).

## indexOf

Returns the index of the first occurrence of a sequence of characters in a string.

### Input Parameters

*inString* **String** String in which you want to locate a sequence of characters.

*subString* **String** Sequence of characters to locate.

*fromIndex* — **String** Optional. Index of *inString* from which to start the search. If no value is specified, this parameter contains `0` to indicate the beginning of the string.

### Output Parameters

*value* — **String** Index of the first occurrence of *subString* in *inString*. If no occurrence is found, this parameter contains `-1`.

## length

Returns the length of a string.

### Input Parameters

*inString* — **String** String whose length you want to discover.

### Output Parameters

*value* — **String** The number of characters in *inString*.

## lookupDictionary

Looks up a given key in a hash table and returns the string to which that key is mapped.

### Input Parameters

*hashtable* — **java.util.Hashtable** Hash table that uses String objects for keys and values.

*key* — **String** Key in *hashtable* whose value you want to retrieve.

> **Note:**
> The key is case sensitive.

### Output Parameters

*value* — **String** Value of the string to which *key* is mapped. If the requested key in *hashtable* is null or if *key* is not mapped to any value in *hashtable*, the service returns null.

## makeString

Builds a single string by concatenating the elements of a String List.

### Input Parameters

*elementList* **String List** Strings to concatenate.

*separator* **String** String to insert between each non-null element in *elementList*.

### Output Parameters

*value* **String** Result from concatenating the strings in *elementList*. Strings are separated by the characters specified in *separator*.

## messageFormat

Formats an array of strings into a given message pattern.

### Input Parameters

*pattern* **String** Message that includes "placeholders" where elements from *argumentList* are to be inserted. The message can contain any sequence of characters. Use the `{n}` placeholder to insert elements from *argumentList*, where *n* is the index of the element that you want to insert. For example, the following pattern string inserts elements 0 and 1 into the message:

```
Test results: {0} items passed, {1} items failed.
```

**Note:**
Do not use any characters except digits for *n*.

*argumentList* **String List** Optional. List of strings to use to populate *pattern*. If *argumentList* is not supplied, the service will not replace placeholders in *pattern* with actual values.

### Output Parameters

*value* **String** Result from substituting *argumentList* into *pattern*. If *pattern* is empty or null, this parameter is null.

## numericFormat

Formats a number into a given numeric pattern.

### Input Parameters

*num* **String** The number to format.

*pattern* **String** A pattern string that describes the way in which *num* is to be formatted:

| This symbol... | Indicates... |
|---|---|
| `0` | A digit. |
| `#` | A digit. Leading zeroes will not be shown. |
| `.` | A placeholder for a decimal separator. |
| `,` | A placeholder for a grouping separator. |
| `;` | A separation in format. |
| `-` | The default negative prefix. |
| `%` | That *num* will be multiplied by 100 and shown as a percentage. |
| `X` | Any character used as a prefix or suffix (for example, `A`, `$`). |
| `'` | That special characters are to be used as literals in a prefix or suffix. Enclose the special characters within '' (for example, '#'). |

The following are examples of pattern strings:

| Pattern | Description |
|---|---|
| `#,###` | Use commas to separate into groups of three digits. The pound sign denotes a digit and the comma is a placeholder for the grouping separator. |
| `#,####` | Use commas to separate into groups of four digits. |
| `$#.00` | Show digits before the decimal point as needed and exactly two digits after the decimal point. Prefix with the $ character. |

| | |
|---|---|
| `'#'#.0` | Show digits before the decimal point as needed and exactly one digit after the decimal point. Prefix with the # character. The first character in a pattern is the dollar sign ($). The pound sign denotes a digit and the period is a placeholder for decimal separator. |

### Output Parameters

*value* **String** *num* formatted according to *pattern*. If *pattern* is an empty (not null) string, the default pattern of comma separators is used and the number of digits after the decimal point remains unchanged.

## objectToString

Converts an object to string representation using the Java toString() method of the object.

### Input Parameters

*object* **Object** The object to be converted to string representation.

### Output Parameters

*string* **String** String representation of the input object converted using the Java toString() method of the object.

## padLeft

Pads a string to a specified length by adding pad characters to the beginning of the string.

### Input Parameters

*inString* **String** String that you want to pad.

*padString* **String** Characters to use to pad *inString*.

*length* **String** Total length of the resulting string, including pad characters.

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString* preceded by as many pad characters as needed so that the total length of the string equals *length*. |

### Usage Notes

If *padString* is longer than one character and does not fit exactly into the resulting string, the beginning of *padString* is aligned with the beginning of the resulting string. For example, suppose *inString* equals `shipped` and *padString* equals `x9y`.

| If *length* equals... | Then *value* will contain... |
|---|---|
| `7` | `shipped` |
| `10` | `x9yshipped` |
| `12` | `x9x9yshipped` |

If *inString* is longer than *length* characters, only the last *length* characters from *inString* are returned. For example, if *inString* equals `acct1234` and *length* equals `4`, value will contain `1234`.

## padRight

Pads a string to a specified length by adding pad characters to the end of the string.

### Input Parameters

| | |
|---|---|
| *inString* | **String** String that you want to pad. |
| *padString* | **String** Characters to use to pad *inString*. |
| *length* | **String** Total length of the resulting string, including pad characters. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString* followed by as many pad characters as needed so that the total length of the string equals *length*. |

### Usage Notes

If *padString* is longer than one character and does not fit exactly into the resulting string, the end of *padString* is aligned with the end of the resulting string. For example, suppose *inString* equals `shipped` and *padString* equals `x9y`.

| If *length* equals... | Then *value* will contain... |
|---|---|
| `7` | `shipped` |
| `10` | `shippedx9y` |
| `12` | `shippedx9y9y` |

If *inString* is longer than *length* characters, only the first *length* characters from *inString* are returned. For example, if *inString* equals `1234acct` and *length* equals `4`, value will contain `1234`.

## replace

Replaces all occurrences of a specified substring with a substitute string.

### Input Parameters

| | |
|---|---|
| *inString* | **String** String containing the substring to replace. |
| *searchString* | **String** Substring to replace within *inString*. |
| *replaceString* | **String** Character sequence that will replace *searchString*. If this parameter is null or empty, the service removes all occurrences of *searchString* from *inString*. |
| *useRegex* | **String** Optional. Flag indicating whether *searchString* is a regular expression. When regular expressions are used to specify a search string, *replaceString* may also contain interpolation fields (for example, "$1") that match parenthetical subexpressions in *searchString*.<br>Set to:<br>■ `true` to indicate that *searchString* is a regular expression.<br>■ `false` to indicate that *searchString* is not a regular expression. This is the default. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString* with replacements made. |

## stringToBytes

Converts a string to a byte array.

### Input Parameters

*string* **String** String to convert to a byte[ ].

*encoding* **String** Optional. Name of a registered, IANA character set that specifies the encoding to use when converting the String to an array of bytes (for example: `ISO-8859-1`).

To use the default encoding, set this value to `autoDetect`. If you specify an unsupported encoding, an exception will be thrown.

### Output Parameters

*bytes* **byte[ ]** Contents of *string* represented as a byte[ ].

## substring

Returns a substring of a given string.

### Input Parameters

*inString* **String** String from which to extract a substring.

*beginIndex* **String** Beginning index of the substring to extract (inclusive).

*endIndex* **String** Ending index of the substring to extract (exclusive). If this parameter is null or empty, the substring will extend to the end of *inString*.

### Output Parameters

*value* **String** Substring from *beginIndex* and extending to the character at *endIndex* - 1.

## tokenize

Tokenizes a string using specified delimiter characters and generates a String List from the resulting tokens.

This service does not return delimiters as tokens.

### Input Parameters

*inString* **String** String you want to tokenize, that is, break into delimited chunks.

| | |
|---|---|
| *delim* | **String** Delimiter characters. If null or empty, the service uses the default delimiters `\t\n\r`, where t, n, and r represent the white space characters tab, new line, and carriage return. |

### Output Parameters

| | |
|---|---|
| *valueList* | **String List** Strings containing the tokens extracted from *inString*. |

## toLower

Converts all characters in a given string to lowercase.

### Input Parameters

| | |
|---|---|
| *inString* | **String** String to convert. |
| *language* | **String** Optional. Lowercase, two-letter ISO-639 code. If this parameter is null, the system default is used. |
| *country* | **String** Optional. Uppercase, two-letter ISO-3166 code. If this parameter is null, the system default is used. |
| *variant* | **String** Optional. Vendor and browser-specific code. If null, this parameter is ignored. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString*, with all uppercase characters converted to lowercase. |

## toUpper

Converts all characters in a given string to uppercase.

### Input Parameters

| | |
|---|---|
| *inString* | **String** String to convert. |
| *language* | **String** Optional. Lowercase, two-letter ISO-639 code. If this parameter is null, the system default is used. |
| *country* | **String** Optional. Uppercase, two-letter ISO-3166 code. If this parameter is null, the system default is used. |

| | |
|---|---|
| *variant* | **String** Optional. Vendor and browser-specific code. If null, this parameter is ignored. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString*, with all lowercase characters converted to uppercase. |

## trim

Trims leading and trailing white space from a given string.

### Input Parameters

| | |
|---|---|
| *inString* | **String** String to trim. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString* with white space trimmed from both ends. |

## URLDecode

Decodes a URL-encoded string.

### Input Parameters

| | |
|---|---|
| *inString* | **String** URL-encoded string to decode. |

### Output Parameters

| | |
|---|---|
| *value* | **String** Result from decoding *inString*. If *inString* contains plus (+) signs, they will appear in *value* as spaces. If *inString* contains %*hex* encoded characters, they will appear in *value* as the appropriate native character. |

## URLEncode

URL-encodes a string.

Encodes characters the same way that data posted from a WWW form is encoded, that is, the `application/x-www-form-urlencoded` MIME type.

### Input Parameters

*inString* **String** String to URL-encode.

### Output Parameters

*value* **String** Result from URL-encoding *inString*. If *inString* contains non-alphanumeric characters (except `[-_.*@]`), they will appear in *value* as their URL-encoded equivalents (% followed by a two-digit hex code). If *inString* contains spaces, they will appear in *value* as plus (+) signs.

## fuzzyMatch

A given string is not exactly matched against a set of strings. If the match is above *similarityThreshold*, it returns the *matchedValue*. If more than one string has not exactly matched, then the first matched string is returned.

### Input Parameters

*inString* **String (Required)** Text to be matched. Text should not be empty or null.

*matchData* **String [ ] (Required)** Array of strings, which are used for matching. If the string array value is either empty or null, it is not used for matching.

*similarityThreshold* **String (Optional)** If the inexact match score is above the given threshold, then service output contains the `matchedValue` parameter. Default value is 0.65. Valid values should be between 0.0 and 1.0. Value 0.0 represents no match and value 1.0 represents an exact match.

*algorithm* **String (Optional)** The algorithm used for an inexact match. Default value is Levenshtein. Supported algorithms are Levenshtein and JaroWinkler.

### Output Parameters

*matchedValue* **String (Optional)** If the inexact match is above *similarityThreshold*, then the returned value contains the matched string.

*similarity* **String (Optional)** If the inexact match is above *similarityThreshold*, then it contains a similarity score. It provides the measure of how close the match is. The returned value can be between 0.0 and 1.0. Value 0.0 represents no match and value 1.0 represents an exact match.

### Usage Notes

Search the web for more information about Levenshtein and JaroWinkler algorithms.

## isNumber

Determines whether the contents of a string can be converted to a float value.

### Input Parameters

*inString* **String** Optional. String to be checked for conversion to float.

### Output Parameters

*isNumber* **String** Indicates whether or not *inString* can be converted to a float value.

- `true` indicates that *inString* can be converted to a float value.
- `false` indicates that *inString* cannot be converted to a float value.

The service returns `false` if *inString* is not specified.

## isAlphanumeric

Determines whether a string consists entirely of alphanumeric characters (in the ranges A–Z, a–z, or 0–9).

### Input Parameters

*inString* **String** Optional. String to be checked for alphanumeric characters.

### Output Parameters

*isAlphanumeric* **String** Indicates whether or not all the characters in *inString* are alphanumeric.

- `true` indicates that all the characters in *inString* are alphanumeric.
- `false` indicates that *not all* the characters in *inString* are alphanumeric.

The service returns `false` if *inString* is not specified.

## isNullOrBlank

Checks a string for a null or a blank value.

### Input Parameters

*inString* **String** Optional. String to be checked for a null or a blank value.

### Output Parameters

*isNullorBlank* **String** Indicates whether or not *inString* has a null or a blank value.

- `true` indicates that *inString* has either a null or a blank value.
- `false` indicates that *inString* contains a value that is not null.

**Note:**
If *inString* is not specified, the service considers the string to be blank and returns `true`.

## isDate

Determines whether a string follows a specified date pattern.

### Input Parameters

*inString* **String** Optional. String to be checked for adherence to the specified date *pattern*.

*pattern* **String** Date format for specifying the *inString* parameter (for example, yyyyMMdd HH:mm:ss.SSS).

For more information about the pattern strings that can be specified for the date, see the “Pattern String Symbols” section.

### Output Parameters

*isDate* **String** Indicates whether or not *inString* follows the specified date pattern.

- `true` indicates that *inString* follows the specified date pattern.
- `false` indicates that *inString* does not follow the specified date pattern.

  The service returns `false` if *inString* is not specified.

### Usage Notes

The service returns an error if both *inString* and *pattern* are not specified.

You can specify any random string (for example, `111212`) as both *inString* and *pattern*. The service returns `true` if the same user-defined string is specified as both *inString* and *pattern*. This is because the java.text.SimpleDateFormat class parses the user-defined input string and pattern to a valid date when the particular input values are identical.

## substitutePipelineVariables

Replaces a pipeline variable with its corresponding value.

### Input Parameters

| | |
|---|---|
| *inString* | **String** Optional. String containing the pipeline variable to replace. Specify the name of the pipeline variable between the `%` symbols (for example, `%phone%`). |

### Output Parameters

| | |
|---|---|
| *value* | **String** Contents of *inString* with the pipeline variable replaced. |

### Usage Notes

The service returns an error if *inString* is not specified.

If *inString* does not contain any variable between the `%` symbols, or contains a value other than the pipeline variable between the `%` symbols, the service does not perform any variable substitution from the pipeline.

If you want to include the `%` symbol in the output, you can specify it as `\%` in *inString*. To specify the value of the pipeline variable as a percentage in the output, append `\%` after the variable name in *inString*. For example, suppose a pipeline variable *revenueIncreasePercent* has a value of `100`.

| If *inString* equals... | Then *value* will contain... |
|---|---|
| `%revenueIncreasePercent%\%` | `100%` |

The service cannot be used for substitution of global variables.

## compareStrings

Performs a case-sensitive comparison of two strings and indicates whether the strings are identical.

### Input Parameters

*inString1* **String** Optional. String to compare against *inString2*. This input variable can be null.

*inString2* **String** Optional. String to compare against *inString1*. This input variable can be null.

### Output Parameters

*isEqual* **String** Indicates whether or not *inString1* and *inString2* are identical.

- `true` indicates that *inString1* and *inString2* are identical.
- `false` indicates that *inString1* and *inString2* are not identical.

**Note:**
If both *inString1* and *inString2* are null, the service considers the strings to be identical and returns `true`.

## Flow Services

Use **Flow** services to perform utility-type tasks.

The following **Flow** services are available:

| Service | Description |
|---|---|
| clearPipeline | Removes all fields from the pipeline. You may optionally specify fields that should not be cleared by this service. |
| getLastError | Obtains detailed information about the last error that was trapped within an Integration. |
| getSessionInfo | Obtains detailed information about the current logged-in user session. Also provides the current Integration name and the execution result reference identifier. |
| getHTTPRequest | Gets information about the HTTP request, received by Integration Cloud. |
| setHTTPResponse | Sets the HTTP response information to be returned by Integration Cloud. |
| countProcessedDocuments | Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen. |

| Service | Description |
|---|---|
| logCustomMessage | Logs a message, which can be viewed in the Execution Results screen. |
| sleep | Causes the currently executing Integration to pause for the specified number of seconds. |

## clearPipeline

Removes all fields from the pipeline. You may optionally specify fields that should not be cleared by this service.

### Input Parameters

| | |
|---|---|
| *preserve* | **String List** Optional. Field names that should not be cleared from the pipeline. |

### Output Parameters

None

## getLastError

Obtains detailed information about the last error that was trapped within an Integration.

### Input Parameters

None

### Output Parameters

| | |
|---|---|
| *lastError* | **Document**. Information about the last error, which contains details of the time, error, user, block, and call stack information. |

| Key | Description |
|---|---|
| **time** | **String**. Date and time the event occurred, in the format *yyyy/MM/dd HH:mm:ss.SSS* |
| **error** | **String**. Optional. Error message of the exception. |
| **localizedError** | **String**. Optional. Error message in the language that corresponds to the server locale. |

| | |
|---|---|
| **user** | **String**. User who executed the Integration. |
| **block** | **Document**. Contains the following fields: |

| Key | Description |
|---|---|
| name | **String**. Integration, Operation, or Service name. |
| type | **String**. Application, Integration, or Service. |
| details | **String**. Optional. Account and Application name if the Block Type is "Application". |

| | |
|---|---|
| **callStack** | **Document List**. The call stack information describing where the error occurred including details of the block. Each document represents a block on the call stack. The first document in the list represents the block that threw the error and the last document in the list represents the top level block. It contains the following fields: |

| Key | Description |
|---|---|
| name | **String**. Integration, Operation or Service name. |
| type | **String**. Application, Integration, or Service. |
| details | **String**. Optional. Account and Application name if the Block Type is "Application". |

### Usage Notes

You can use this service in the *catch* section of the *try-catch* block. Each execution of an Integration or a service (whether the Integration or the service succeeds or fails) updates the value returned by getLastError. Consequently, getLastError itself resets the value of lastError. Therefore, if the results of getLastError will be used as input to subsequent Integrations, map the value of lastError to a variable in the pipeline.

If a map has multiple transformers, then a subsequent call to getLastError will return the error associated with the last failed transformer in the map, even if it is followed by successful transformers.

## getSessionInfo

Obtains detailed information about the current logged-in user session. Also provides the current Integration execution result reference identifier.

### Input Parameters

None

### Output Parameters

*$session* **Document** Returns information about the current logged-in user session. Also provides the current Integration name and the execution result reference identifier.

<table>
<tr><th>Key</th><th>Description</th></tr>
<tr><td><i>tenantId</i></td><td><b>String</b> Tenant Identifier.</td></tr>
<tr><td><i>stageId</i></td><td><b>String</b> The stage ID where the current integration resides.</td></tr>
<tr><td><i>user</i></td><td><b>Document</b> Returns user details.
<table>
<tr><th>Key</th><th>Description</th></tr>
<tr><td><i>name</i></td><td><b>String</b> Name of the user who is executing the service.</td></tr>
</table></td></tr>
<tr><td><i>integrationName</i></td><td><b>String</b> The name of the Integration.<br><b>Note:</b><br>If Integration A has a referenced Integration B, and if the getSessionInfo service is called in Integration B, then <code>integrationName</code> will be A but if Integration B is executed independently, then <code>integrationName</code> will be B.</td></tr>
<tr><td><i>executionResultReference</i></td><td><b>String</b> Returns the current Integration execution result reference identifier. For example, you can pass the identifier to an on-premises operation and trace the Integration execution.</td></tr>
</table>

## getHTTPRequest

Gets information about the HTTP request, received by Integration Cloud.

### Parameters

*headers* **Document** Contains the header fields from the HTTP request.

*requestURL* **String** URL used by the client to invoke the service.

*method* **String** HTTP method used by the client to request the top-level service. Possible values are GET, PUT, POST, PATCH, and DELETE.

## setHTTPResponse

Sets the HTTP response information to be returned by Integration Cloud.

### Parameters

*headers* **Document** Optional. Contains the header fields to be returned in the HTTP response.

*responseCode* **String** Optional. HTTP status code to be returned to the client.

The response codes and phrases are defined in https://tools.ietf.org/html/rfc7231#section-6. If you provide a value for *responseCode* that is not listed in RFC 7321, Section 6, you must also provide a value for *reasonPhrase*.

*responsePhrase* **String** Optional. HTTP reason phrase to be returned to the client. If no reason is provided, the default reason phrase associated with *responseCode* will be used. You must provide a *reasonPhrase* for any *responseCode* that is not listed in RFC 7321, Section 6.

*responseString* **String** Optional. Response to be returned to the client, specified as a string.

*responseBytes* **byte[ ]** Optional. Response to be returned to the client, specified as a byte array.

*responseStream* **java.io.InputStream** Optional. Response to be returned to the client, specified as an InputStream.

## countProcessedDocuments

Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen.

### Input Parameters

*status* **String** Optional valid values are "success" or "fail". Set status to "success" to count the number of successfully processed documents, else set it to "fail". Default value is "success".

*incrementBy* **String** Optional. Increment the number of documents processed by an Integration. Every time the service is used, successful or failed documents are incremented by the given value. Default value is 1.

### Output Parameters

None

### Usage Notes

To increment the number of documents processed by a list, use the **sizeOfList** service in the **List** service block.

## logCustomMessage

Logs a message, which can be viewed in the Execution Results screen.

### Input Parameters

*message* **String** Custom message to be logged, which can be viewed in the Execution Results screen.

### Output Parameters

None

## sleep

Causes the currently executing Integration to pause for the specified number of seconds.

### Input Parameters

*seconds* **String** The number of seconds to pause the currently executing Integration. The value must be an integer between 1 second and 60 seconds.

### Output Parameters

None

# Hashtable Services

Use **Hashtable** services to create, update, and obtain information about the hashtable.

The following **Hashtable** services are available:

| Service | Description |
|---|---|
| containsKey | Checks for the existence of a hashtable element. |
| createHashtable | Creates a hashtable object. |
| get | Gets the value for a specified key in the hashtable. |
| listKeys | Lists all the keys stored in the hashtable. |
| put | Adds a key/value pair in the hashtable. |
| remove | Removes a key/value pair from the hashtable. |
| size | Gets the number of elements in the hashtable. |

## containsKey

Checks for the existence of a hashtable element.

### Input Parameters

*hashtable* **java.util.Hashtable** Hashtable in which to check for the existence of a hashtable element.

*key* **String** Hashtable element to be checked for.

### Output Parameters

*containsKey* **String** Indicates whether the specified hashtable element exists. A value of:

- `true` indicates that the element exists.
- `false` indicates that the element does not exist.

## createHashtable

Creates a hashtable object.

### Input Parameters

None.

### Output Parameters

| | |
|---|---|
| *hashtable* | **java.util.Hashtable** The new hashtable object. |

## get

Gets the value for a specified key in the hashtable.

### Input Parameters

| | |
|---|---|
| *hashtable* | **java.util.Hashtable** Hashtable from which to retrieve the specified value. |
| *key* | **String** Key of the hashtable element whose value is to be retrieved. |

### Output Parameters

| | |
|---|---|
| *value* | **Object** Value of the input hashtable element. |

## listKeys

Lists all the keys stored in the hashtable.

### Input Parameters

| | |
|---|---|
| *hashtable* | **java.util.Hashtable** Hashtable from which the keys are to be listed. |

### Output Parameters

| | |
|---|---|
| *keys* | **String[]** List of keys stored in the input hashtable. |

## put

Adds a key/value pair in the hashtable.

### Input Parameters

| | |
|---|---|
| *hashtable* | **java.util.Hashtable** Hashtable to which the key/value pair is to be added. |
| *key* | **String** Key of the element to be added to the hashtable. |
| *value* | **Object** Value of the element to be inserted into the hashtable. |

### Output Parameters

*hashtable* **java.util.Hashtable** Hashtable object after the insertion of the key/value pair.

## remove

Removes a key/value pair from the hashtable.

### Input Parameters

*hashtable* **java.util.Hashtable** Hashtable from which to remove the key/value pair.

*key* **String** Key of the hashtable element to be removed.

*value* **Object** Value of the hashtable element to be removed.

### Output Parameters

*hashtable* **java.util.Hashtable** Hashtable object after the key/value pair is removed.

*value* **Object** Value of the hashtable element that was removed. Returns `null` if the input *key* is not found in the hashtable.

## size

Gets the number of elements in the hashtable.

### Input Parameters

*hashtable* **java.util.Hashtable** Hashtable from which the number of elements stored in it is to be retrieved.

### Output Parameters

*size* **String** Number of elements in the hashtable.

# Flat File Services

Use **Flat File** services to convert data bytes, data stream, and data string to a document and vice versa.

The following **Flat File** services are available:

| Service | Description |
|---|---|
| delimitedDataBytesToDocument | Converts delimited data bytes (byte array) to a document. |
| delimitedDataStreamToDocument | Converts delimited data stream to a document. |
| delimitedDataStringToDocument | Converts delimited data string to a document. |
| documentToDelimitedDataBytes | Converts a document to delimited data bytes (byte array object). |
| documentToDelimitedDataStream | Converts a document to a delimited data stream. |
| documentToDelimitedDataString | Converts a document to a delimited data string. |

## delimitedDataBytesToDocument

Converts delimited data bytes (byte array) to a document.

This service will convert the following delimited data from byte array:

"Date","Pupil","Grade"

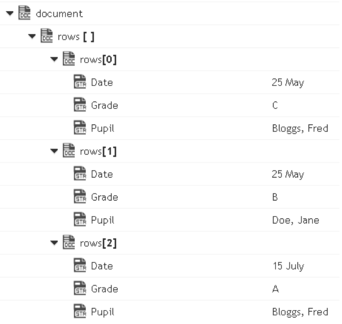
"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

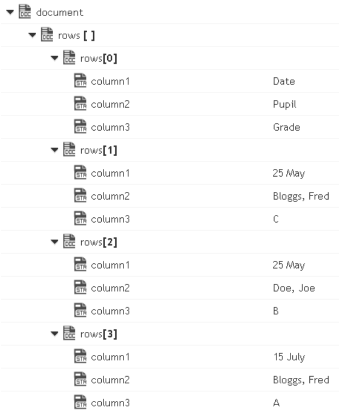
**Note:**
Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)



```
▾ document
  ▾ rows [ ]
    ▾ rows[0]
        Date                  25 May
        Grade                 C
        Pupil                 Bloggs, Fred
    ▾ rows[1]
        Date                  25 May
        Grade                 B
        Pupil                 Doe, Jane
    ▾ rows[2]
        Date                  15 July
        Grade                 A
        Pupil                 Bloggs, Fred
```

or to a document that looks like: (useHeaderRowForFieldNames=false)



```
document
  rows [ ]
    rows[0]
      column1      Date
      column2      Pupil
      column3      Grade
    rows[1]
      column1      25 May
      column2      Bloggs, Fred
      column3      C
    rows[2]
      column1      25 May
      column2      Doe, Joe
      column3      B
    rows[3]
      column1      15 July
      column2      Bloggs, Fred
      column3      A
```

### Input Parameters

*delimited DataBytes* **java.lang.Byte[ ]**. Delimited data in bytes (Byte array) to convert to a document.

*fieldQualifier* **String** Optional. The delimiter to use for separating entries in *delimitedDataBytes*. Default is comma (,).

*textQualifier* **String** Optional. The character to use for quoted elements. Default is double quote (").

*useHeader RowFor FieldNames* **String** Optional. Consider first line as header row and use the delimited data of this line as property names in the output document. Set to:

- *true*. The delimited data of first line will be used as the property name in the output document. This is the default.
- *false*. column1, column2...columnN will be used as the property name in the output document.

*encoding* **String** Optional. The encoding to use while parsing the delimited data.

### Output Parameters

*document* **Document**. Document resulting from the conversion of *delimitedDataBytes*. This document contains document array rows[] corresponding to the delimited data.

## delimitedDataStreamToDocument

Converts delimited data stream to a document. The permissible size of the content stream is based on your tenancy. The permissible size of the content stream is based on your tenancy.

This service converts the following delimited data in a stream:

"Date","Pupil","Grade"

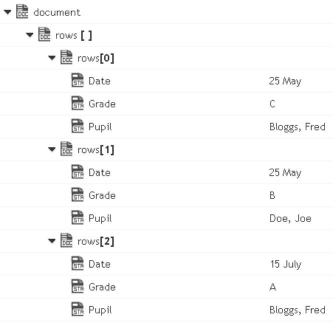
"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

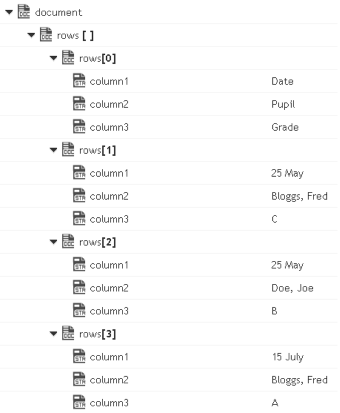
**Note:**
Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)



```
document
  rows [ ]
    rows[0]
      Date        25 May
      Grade       C
      Pupil       Bloggs, Fred
    rows[1]
      Date        25 May
      Grade       B
      Pupil       Doe, Joe
    rows[2]
      Date        15 July
      Grade       A
      Pupil       Bloggs, Fred
```

or to a document that looks like: (useHeaderRowForFieldNames=false)



```
document
  rows [ ]
    rows[0]
      column1     Date
      column2     Pupil
      column3     Grade
    rows[1]
      column1     25 May
      column2     Bloggs, Fred
      column3     C
    rows[2]
      column1     25 May
      column2     Doe, Joe
      column3     B
    rows[3]
      column1     15 July
      column2     Bloggs, Fred
      column3     A
```

## Input Parameters

*delimitedDataStream* **java.io.InputStream**. Delimited data in an input stream to convert to a document.

*fieldQualifier* **String** Optional. The delimiter to use for separating entries in *delimitedDataStream*. Default is comma (,).

*textQualifier* **String** Optional. The character to use for quoted elements. Default is double quote (").

*useHeaderRowForFieldNames* **String** Optional. Consider first line as header row and use the delimited data of this line as property names in the output document. Set to:

- *true*. The delimited data of first line will be used as the property name in the output document. This is the default.
- *false*. column1, column2...columnN will be used as the property name in the output document.

*encoding* **String** Optional. The encoding to use while parsing the delimited data.

### Output Parameters

*document* **Document**. Document resulting from the conversion of *delimitedDataStream*. This document contains document array rows[] corresponding to the delimited data.

## delimitedDataStringToDocument

Converts delimited data string to a document.

This service will convert the following delimited data string:

"Date","Pupil","Grade"

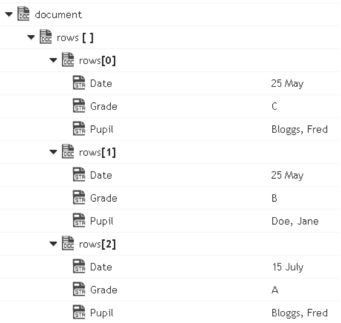
"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

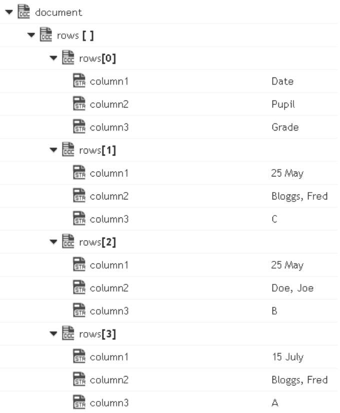
**Note:**
Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)



```
document
  rows [ ]
    rows[0]
      Date        25 May
      Grade       C
      Pupil       Bloggs, Fred
    rows[1]
      Date        25 May
      Grade       B
      Pupil       Doe, Jane
    rows[2]
      Date        15 July
      Grade       A
      Pupil       Bloggs, Fred
```

or to a document that looks like: (useHeaderRowForFieldNames=false)



```
document
  rows [ ]
    rows[0]
      column1          Date
      column2          Pupil
      column3          Grade
    rows[1]
      column1          25 May
      column2          Bloggs, Fred
      column3          C
    rows[2]
      column1          25 May
      column2          Doe, Joe
      column3          B
    rows[3]
      column1          15 July
      column2          Bloggs, Fred
      column3          A
```

## Input Parameters

*delimited DataString* — **String**. Delimited string to convert to a document.

*fieldQualifier* — **String** Optional. The delimiter to use for separating entries in *delimitedDataString*. Default is comma (,).

*textQualifier* — **String** Optional. The character to use for quoted elements. Default is double quote (").

*useHeader RowFor FieldNames* — **String** Optional. Consider first line as header row and use the delimited data of this line as property names in the output document. Set to:

- *true*. The delimited data of first line will be used as the property name in the output document. This is the default.
- *false*. column1, column2...columnN will be used as the property name in the output document.

*encoding* — **String** Optional. The encoding to use while parsing the delimited data.

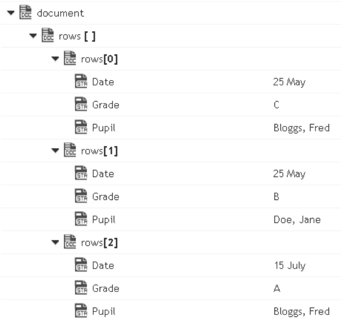
## Output Parameters

*document* — **Document**. Document resulting from the conversion of *delimitedDataString*. This document contains document array rows[] corresponding to the delimited data.

# documentToDelimitedDataBytes

Converts a document to delimited data bytes (byte array object).

This service will convert the following document:



| | |
|---|---|
| document | |
| rows [ ] | |
| rows[0] | |
| Date | 25 May |
| Grade | C |
| Pupil | Bloggs, Fred |
| rows[1] | |
| Date | 25 May |
| Grade | B |
| Pupil | Doe, Jane |
| rows[2] | |
| Date | 15 July |
| Grade | A |
| Pupil | Bloggs, Fred |

To bytes (byte array object) containing the following delimited data:
(useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

**Note:**
Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

To the byte (byte array object) containing the following delimited data:
(useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

## Input Parameters

*document* **Document**. Document to be converted to delimited data bytes (byte array object). This document contains a document array rows[] corresponding to the delimited data.

*fieldQualifier* **String** Optional. The delimiter to use for separating entries in *delimitedDataBytes*. Default is comma (,).

*textQualifier* **String** Optional. The character to use for quoted elements. Default is double quote (").

*useFieldNamesForHeaderRow* **String** Optional. The first line in the output delimited data *delimitedDataBytes* will be constructed using the property names in the input document array document\rows[]. Set to:

- *true*. Property names in the input document array document\rows[] will be used as the first row in the output *delimitedDataBytes*.
- *false*. column1, column2...columnN will be used as the first row in the output *delimitedDataBytes*.

*encoding* **String** Optional. The encoding to use while parsing the delimited data.

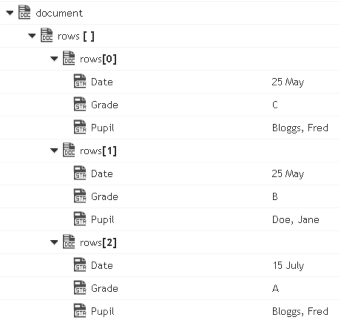
### Output Parameters

*delimitedDataBytes* **Object**. Delimited data byte array object resulting from the conversion of a document.

## documentToDelimitedDataStream

Converts a document to a delimited data stream.

This service will convert the following document:



```
document
  rows [ ]
    rows[0]
      Date      25 May
      Grade     C
      Pupil     Bloggs, Fred
    rows[1]
      Date      25 May
      Grade     B
      Pupil     Doe, Jane
    rows[2]
      Date      15 July
      Grade     A
      Pupil     Bloggs, Fred
```

To the stream containing the following delimited data: (useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

> **Note:**
> Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

or to the stream containing the following delimited data: (useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

> **Note:**
> Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

### Input Parameters

| | |
|---|---|
| *document* | **Document**. Document to be converted to delimited data stream. This document contains a document array rows[] corresponding to the delimited data. |
| *fieldQualifier* | **String** Optional. The delimiter to use for separating entries in *delimitedDataStream*. Default is comma (,). |
| *textQualifier* | **String** Optional. The character to use for quoted elements. Default is double quote ("). |
| *useFieldNamesForHeaderRow* | **String** Optional. The first line in the output delimited data *delimitedDataStream* will be constructed using the property names in the input document array document\rows[]. Set to:<br>■ *true*. Property names in the input document array document\rows[] will be used as the first row in the output *delimitedDataStream*.<br>■ *false*. column1, column2...columnN will be used as the first row in the output *delimitedDataStream*. |
| *encoding* | **String** Optional. The encoding to use while parsing the delimited data. |

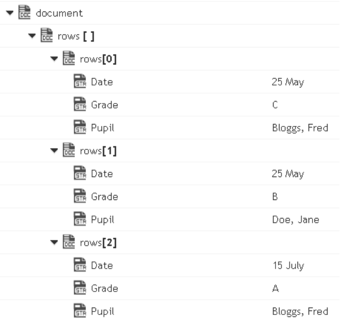
### Output Parameters

| | |
|---|---|
| *delimitedDataStream* | **java.io.InputStream**. Delimited data stream resulting from the conversion of a document. |

## documentToDelimitedDataString

Converts a document to a delimited data string.

This service will convert the following document:



```
document
  rows [ ]
    rows[0]
      Date      25 May
      Grade     C
      Pupil     Bloggs, Fred
    rows[1]
      Date      25 May
      Grade     B
      Pupil     Doe, Jane
    rows[2]
      Date      15 July
      Grade     A
      Pupil     Bloggs, Fred
```

To the string containing the following delimited data: (useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

To the string containing the following delimited data: (useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

## Input Parameters

*document* **Document**. Document to be converted to delimited data string. This document contains document array rows[] corresponding to the delimited data.

*fieldQualifier* **String** Optional. The delimiter to use for separating entries in *delimitedDataString*. Default is comma (,).

*textQualifier* **String** Optional. The character to use for quoted elements. Default is double quote (").

*useField NamesFor HeaderRow* **String** Optional. The first line in the output delimited data *delimitedDataString* will be constructed using the property names in the input document array document\rows[]. Set to:

- *true*. Property names in the input document array document\rows[] will be used as the first row in the output *delimitedDataString*.

- *false*. column1, column2...columnN will be used as the first row in the output *delimitedDataString*.

*encoding* **String** Optional. The encoding to use while parsing the delimited data.

### Output Parameters

*delimited DataString* **String**. Delimited data byte string resulting from the conversion of a document.

## JSON Services

Use **JSON** services to convert JSON content into a document and to convert a document into JSON content.

The following **JSON** services are available:

| Service | Description |
|---|---|
| documentToJSONBytes | Converts a document to JSON bytes (byte array). |
| documentToJSONStream | Converts a document to a JSON stream. |
| documentToJSONString | Converts a document to a JSON string. |
| jsonBytesToDocument | Converts JSON content in bytes (byte array) to a document. |
| jsonStreamToDocument | Converts content from the JSON content stream to a document. |
| jsonStringToDocument | Converts content from the JSON string to a document. |

## documentToJSONBytes

Converts a document to JSON bytes (byte array).

### Input Parameters

*document* **Document**. The document to be converted to JSON bytes (byte array).

### Output Parameters

*jsonBytes* **Object**. JSON bytes (byte array) resulting from the conversion of a document.

## documentToJSONStream

Converts a document to a JSON stream.

### Input Parameters

*document* **Document**. The document to be converted to a JSON stream.

### Output Parameters

*jsonStream* **java.io.InputStream**. JSON stream resulting from the conversion of a document.

## documentToJSONString

Converts a document to a JSON string.

### Input Parameters

*document* **Document**. The document to be converted to a JSON string.

*prettyPrint* **String**. Formats the *jsonString* output parameter for human readability by adding carriage returns and indentation to the JSON content. Set to:

- *true* to format the *jsonString* output field for human readability
- *false* to leave the *jsonString* output field in its unformed state

The service will not add any additional carriage returns or indentation to the JSON content.

### Output Parameters

*jsonString* **Object**. JSON string resulting from the conversion of a document.

## jsonBytesToDocument

Converts JSON content in bytes (byte array) to a document.

### Input Parameters

*jsonBytes* **java.lang.Byte[]**. JSON content in bytes (byte array) to convert to a document.

*decodeRealAsDouble* **String**. Optional. Converts real numbers from *jsonBytes* to either a Float or Double Java wrapper type. Set to:

- *true* to convert real numbers to Double Java wrapper types
- *false* to convert real numbers to Float Java wrapper types

Default value is *true*.

*decodeIntegerAsLong* **String**. Optional. Converts integers from *jsonBytes* to either a Long or Integer Java wrapper type. Set to:

- *true* to convert integers to Long Java wrapper types
- *false* to convert integers to Integer Java wrapper types

Default value is *true*.

### Output Parameters

*document* **Document**. Document resulting from the conversion of *jsonBytes*.

## jsonStreamToDocument

Converts content from the JSON content stream to a document. The permissible size of the content stream is based on your tenancy.

### Input Parameters

*jsonStream* **java.io.InputStream**. JSON content in an input stream to convert to a document.

*decodeRealAsDouble* **String**. Optional. Converts real numbers from *jsonStream* to either a Float or Double Java wrapper type. Set to:

- *true* to convert real numbers to Double Java wrapper types
- *false* to convert real numbers to Float Java wrapper types

Default value is *true*.

*decodeIntegerAsLong* **String**. Optional. Converts integers from *jsonStream* to either a Long or Integer Java wrapper type. Set to:

- *true* to convert integers to Long Java wrapper types
- *false* to convert integers to Integer Java wrapper types

Default value is *true*.

### Output Parameters

*document* **Document**. Document resulting from the conversion of *jsonStream*.

## jsonStringToDocument

Converts content from the JSON content string to a document.

### Input Parameters

*jsonString* **String**. JSON content string to convert to a document.

*decodeRealAsDouble* **String**. Optional. Converts real numbers from *jsonString* to either a Float or Double Java wrapper type. Set to:

- *true* to convert real numbers to Double Java wrapper types
- *false* to convert real numbers to Float Java wrapper types

Default value is *true*.

*decodeIntegerAsLong* **String**. Optional. Converts integers from *jsonString* to either a Long or Integer Java wrapper type. Set to:

- *true* to convert integers to Long Java wrapper types
- *false* to convert integers to Integer Java wrapper types

Default value is *true*.

### Output Parameters

*document* **Document**. Document resulting from the conversion of *jsonString*.

## Transaction Services

Use **Transaction** services only in conjunction with Database Application operations. These services are applicable when the Database Application account is of type Transactional.

The following **Transaction** services are available:

| Service | Description |
|---|---|
| Transaction:commit | Commits an explicit transaction. |
| Transaction:rollback | Rolls back an explicit transaction. |

| Service | Description |
|---|---|
| Transaction:setTimeout | Manually sets a transaction timeout interval for implicit and explicit transactions. |
| Transaction:start | Starts an explicit transaction. |

## Transaction:commit

Commits an explicit transaction.

### Input Parameters

*commitTransactionInput* **Document** Information for each commit request.

| Key | Description |
|---|---|
| *transactionName* | **String** The name of an explicit transaction that you want to commit. The *transactionName* must have been previously used in a call to Transaction:start.<br>This value must be mapped from the most recent Transaction:start that has not previously been committed or rolled back. |

### Output Parameters

None.

### Usage Notes

This service must be used in conjunction with the Transaction:start service. If the *transactionName* parameter was not provided in a prior call to Transaction:start, a run-time error will be returned.

## Transaction:rollback

Rolls back an explicit transaction.

### Input Parameters

*rollbackTransactionInput* **Document List** Information for each rollback request.

| Key | Description |
|---|---|

*transactionName* **String** The name of an explicit transaction that you want to roll back. The *transactionName* must have been previously used in a call to Transaction:start.

This value must be mapped from the most recent Transaction:start that has not previously been committed or rolled back.

### Output Parameters

None.

### Usage Notes

This service must be used in conjunction with the Transaction:start service. If the given *transactionName* parameter was not provided in a prior call to Transaction:start, a run-time error will be returned.

## Transaction:setTimeout

Manually sets a transaction timeout interval for implicit and explicit transactions.

### Input Parameters

*timeoutSeconds* **Integer** The number of seconds that the implicit or explicit transaction stays open before the transaction manager marks it for rollback.

### Output Parameters

None.

### Usage Notes

You must call this service before you call the Transaction:start service.

If the execution of a transaction takes longer than the transaction timeout interval, all transacted operations are rolled back.

## Transaction:start

Starts an explicit transaction.

### Input Parameters

*startTransactionInput* **Document** Information for each start transaction request.

| Key | Description |
|---|---|
| *transactionName* | **String** Optional. Specifies the name of the transaction to be started. If you leave this parameter blank, the Database Application will generate a name for you. In most implementations it is not necessary to provide your own transaction name. |

### Output Parameters

*startTransactionOutput* **Document** Information for each start transaction request.

| Key | Description |
|---|---|
| *transactionName* | **String** The name of the transaction the service just started. |

### Usage Notes

This service is intended for use with the Transaction:commit or Transaction:rollback service. The *transactionName* value returned by a call to this service can be provided to Transaction:commit (to commit the transaction) or Transaction:rollback (to roll back the transaction).

## XML Services

Use **XML** services to convert a document to XML content and XML content to a document.

The following **XML** services are available:

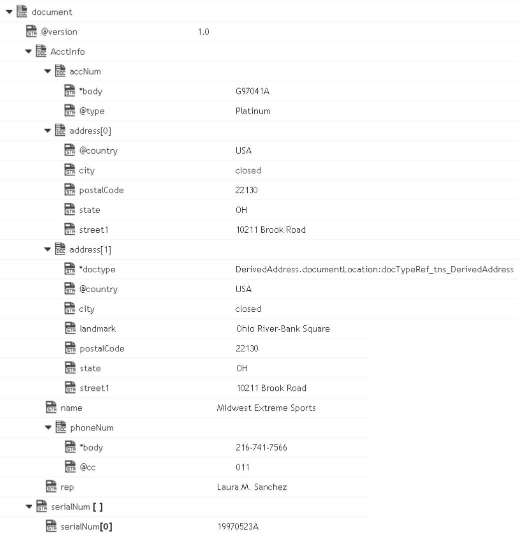
| Service | Description |
|---|---|
| documentToXMLBytes | Converts a document to XML content bytes, as a byte array object. |
| documentToXMLStream | Converts a document to XML stream, as a java.io.InputStream object. |
| documentToXMLString | Converts a document to XML content string. |
| xmlBytesToDocument | Converts XML content bytes (byte array) to a document. |
| xmlNodeToDocument | Converts an XML node to a document. |

| Service | Description |
|---|---|
| xmlStreamToDocument | Converts an XML content stream to a document. |
| xmlStringToDocument | Converts an XML string to a document. |
| xmlStringToXMLNode | Converts a String, byte[], or InputStream containing an XML document to an XML node. |
| getXMLNodeType | Returns information about an XML node. |
| queryXMLNode | Queries an XML node. |

## documentToXMLBytes

Converts a document to xml content bytes, as a byte array object. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements, and the key values are turned into the contents of those elements.

This service will convert the following document:



| | |
|---|---|
| document | |
| @version | 1.0 |
| AcctInfo | |
| accNum | |
| *body | G97041A |
| @type | Platinum |
| address[0] | |
| @country | USA |
| city | closed |
| postalCode | 22130 |
| state | OH |
| street1 | 10211 Brook Road |
| address[1] | |
| *doctype | DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress |
| @country | USA |
| city | closed |
| landmark | Ohio River-Bank Square |
| postalCode | 22130 |
| state | OH |
| street1 | 10211 Brook Road |
| name | Midwest Extreme Sports |
| phoneNum | |
| *body | 216-741-7566 |
| @cc | 011 |
| rep | Laura M. Sanchez |
| serialNum [ ] | |
| serialNum[0] | 19970523A |

To XML document bytes (byte array object), whose content looks like:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA><street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
```

```
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state><postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

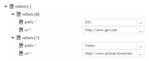
## Input Parameters

*document*

**Document**. Document that is to be converted to XML. Note that if you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

*nsDecls [ ]*

**Document**. Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



nsDecls [ ]
nsDecls [0]
prefix * GSX
uri * http://www.gsx.com
nsDecls [1]
prefix * TxMon
uri * http://www.acrtrak/txmonitor

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

xmlns:gsx="http://www.gsx.com"

xmlns:TxMon="http:www.acrtrak/txMonitor"

Alternatively, you can declare a namespace by including an @xmlns key in document. If you were not using the @ character to designate attributes, use the correct attribute prefix in your code.

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

*addHeader*

**String**. Optional.

Flag specifying whether the header element <?xml version="1.0"?> is to be included in the resulting XML String.

Set to:

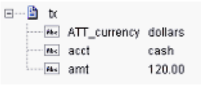
*true* to include the header. This is the default.

*false* to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

*attrPrefix*

**String** Optional. Prefix that designates keys containing attributes. The default prefix is "@".

For example, if you set *attrPrefix* to `ATT_` and *document* contained the following element:



```
tx
    ATT_currency  dollars
    acct          cash
    amt           120.00
```

documentToXMLBytes would convert the `ATT_currency` key to the attribute, `currency=dollars`, in the `<tx>` element as shown below:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
</tx>
```

*encode*

**String** Optional. Flag indicating whether to HTML-encode the data. Set this parameter to true if your XML data contains special characters, including the following: `< > & " '`

Set to:

- `true` to HTML-encode the data.

  For example, the string expression `5 < 6` would be converted to `<expr>5 &lt; 6</expr>`, which is valid.

  If you do not want a leading `&` (ampersand) character encoded when it appears as part of a character or entity reference, set *preserveRefs* to true.

- `false` to not HTML-encode the data. This is the default.

  For example, the string expression `5 < 6`would be converted to `<expr>5 < 6</expr>`, which is invalid.

*preserveRefs*

**String** Optional. Flag indicating whether the leading `&` (ampersand) of a well-formed entity or character reference is left as `&` or further encoded as `&amp;` when the data is to be HTML-encoded (*encode* is set to `true`).

Set to:

- `true` to preserve the leading `&` (ampersand) in an entity or character reference when the service HTML-encodes the data.
- `false` to encode the leading & (ampersand) as &amp; when the & appears in an entity or character reference. This is the default.

The service ignores the value of *preseveRefs* when *encode* is set to false.

*enforceLegalXML*

**String** Optional. Flag indicating whether the service throws an exception when *document* contains multiple root elements or illegal XML tag names. Set to:

- `true` to throw an exception if *document* would produce an XML String containing multiple root elements and/or illegal XML tag names.
- `false` to allow the resulting XML String to contain multiple root elements and/or illegal XML tag names. You would use this setting, for example, to create an XML fragment composed of multiple elements that were not all enclosed within a root element. This is the default.

## Output Parameters

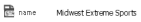
*xmlBytes*

**Object**. XML content bytes (byte array) produced from document.

**Usage Notes**

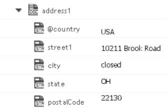
If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in the document as shown below:



name Midwest Extreme Sports

If you want to generate an element that contains children, represent with a document in the document as shown below:

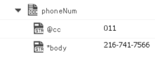


| | |
|---|---|
| address1 | |
| @country | USA |
| street1 | 10211 Brook Road |
| city | closed |
| state | OH |
| postalCode | 22130 |

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named *body that contains the element's value.

For example, if you want to produce the following element:

<phoneNum cc=011>216-741-7566</phoneNum>, you would include the following in document:



| | |
|---|---|
| phoneNum | |
| @cc | 011 |
| *body | 216-741-7566 |

To include namespaces, ensure that you do the following:

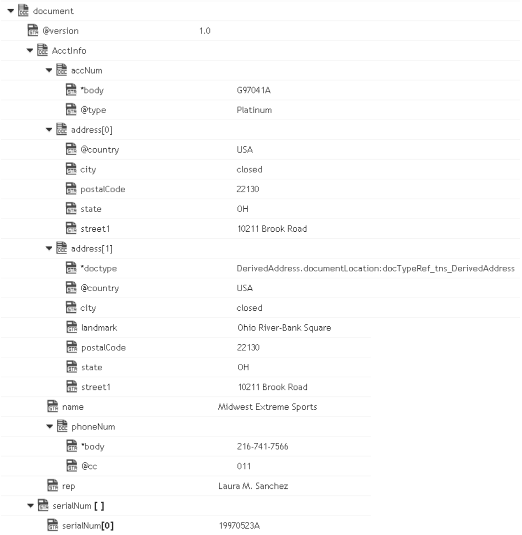
Include the appropriate namespace prefix in the key names in document. For example, to produce an element called acctNum that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named GSX:acctNum in document.

Define the URIs for the prefixes that appear in document. You can do this through nsDecls or by including an @xmlns key in the element where you want the xmlns attribute to be inserted.

## documentToXMLStream

Converts a document to xml stream, as a java.io.InputStream object. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements and the key values are turned into contents of those elements.

This service will convert the following document:



```
document
  @version                 1.0
  AcctInfo
    accNum
      *body                G97041A
      @type                Platinum
    address[0]
      @country             USA
      city                 closed
      postalCode           22130
      state                OH
      street1              10211 Brook Road
    address[1]
      *doctype             DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
      @country             USA
      city                 closed
      landmark             Ohio River-Bank Square
      postalCode           22130
      state                OH
      street1              10211 Brook Road
    name                 Midwest Extreme Sports
    phoneNum
      *body                216-741-7566
      @cc                  011
    rep                  Laura M. Sanchez
  serialNum [ ]
    serialNum[0]         19970523A
```

To an XML document stream, whose content looks like:

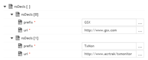
```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

## Input Parameters

*document* **Document**. Document that is to be converted to XML. Note that if you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

*nsDecls [ ]*

**Document**. Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



nsDecls [ ]

nsDecls [0]

prefix * GSX

uri * http://www.gsx.com

nsDecls [1]

prefix * TxMon

uri * http://www.acrtrak/txmonitor

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

xmlns:gsx="http://www.gsx.com"

xmlns:TxMon="http:www.acrtrak/txMonitor"

Alternatively, you can declare a namespace by including an @xmlns key in document. If you were not using the @ character to designate attributes, use the correct attribute prefix in your code.

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

*addHeader*

**String**. Optional.

Flag specifying whether the header element <?xml version="1.0"?> is to be included in the resulting XML String.

Set to:

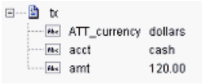
*true* to include the header. This is the default.

*false* to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

*attrPrefix*

**String** Optional. Prefix that designates keys containing attributes. The default prefix is "@".

For example, if you set *attrPrefix* to `ATT_` and *document* contained the following element:



| | |
|---|---|
| tx | |
| ATT_currency | dollars |
| acct | cash |
| amt | 120.00 |

documentToXMLStream would convert the `ATT_currency` key to the attribute, `currency=dollars`, in the `<tx>` element as shown below:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
</tx>
```

*encode*

**String** Optional. Flag indicating whether to HTML-encode the data. Set this parameter to true if your XML data contains special characters, including the following: `< > & " '`

Set to:

- `true` to HTML-encode the data.

  For example, the string expression `5 < 6` would be converted to `<expr>5 &lt; 6</expr>`, which is valid.

  If you do not want a leading `&` (ampersand) character encoded when it appears as part of a character or entity reference, set *preserveRefs* to true.

- `false` to not HTML-encode the data. This is the default.

  For example, the string expression `5 < 6`would be converted to `<expr>5 < 6</expr>`, which is invalid.

*preserveRefs*

**String** Optional. Flag indicating whether the leading `&` (ampersand) of a well-formed entity or character reference is left as `&` or further encoded as `&amp;` when the data is to be HTML-encoded (*encode* is set to `true`).

Set to:

- `true` to preserve the leading `&` (ampersand) in an entity or character reference when the service HTML-encodes the data.
- `false` to encode the leading & (ampersand) as &amp; when the & appears in an entity or character reference. This is the default.

The service ignores the value of *preseveRefs* when *encode* is set to false.

*enforceLegalXML*

**String** Optional. Flag indicating whether the service throws an exception when *document* contains multiple root elements or illegal XML tag names. Set to:

- `true` to throw an exception if *document* would produce an XML String containing multiple root elements and/or illegal XML tag names.
- `false` to allow the resulting XML String to contain multiple root elements and/or illegal XML tag names. You would use this setting, for example, to create an XML fragment composed of multiple elements that were not all enclosed within a root element. This is the default.

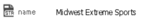
## Output Parameters

*xmlStream* **java.io.InputStream**. XML content stream produced from document.

**Usage Notes**

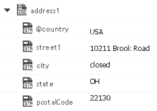
If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in document as shown below:



| | |
|---|---|
| name | Midwest Extreme Sports |

If you want to generate an element that contains children, represent with a document in the document as shown below:



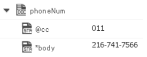
| | |
|---|---|
| address1 | |
| @country | USA |
| street1 | 10211 Brook Road |
| city | closed |
| state | OH |
| postalCode | 22130 |

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named *body that contains the element's value.

For example, if you want to produce the following element:

<phoneNum cc=011>216-741-7566</phoneNum>

You would include the following in document:



| | |
|---|---|
| phoneNum | |
| @cc | 011 |
| *body | 216-741-7566 |

To include namespaces, ensure that you do the following:

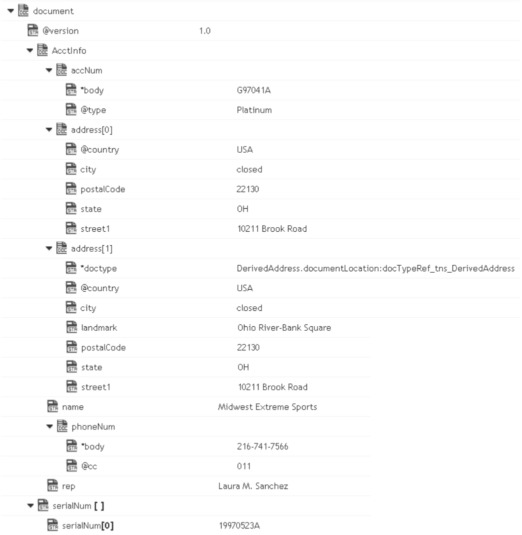
Include the appropriate namespace prefix in the key names in document. For example, to produce an element called acctNum that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named GSX:acctNum in document.

Define the URIs for the prefixes that appear in document. You can do this through nsDecls or by including an @xmlns key in the element where you want the xmlns attribute to be inserted.

## documentToXMLString

Converts a document to xml content string. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements, and the key values are turned into the contents of those elements.

This service will convert the following document:



| | |
|---|---|
| document | |
| @version | 1.0 |
| AcctInfo | |
| accNum | |
| *body | G97041A |
| @type | Platinum |
| address[0] | |
| @country | USA |
| city | closed |
| postalCode | 22130 |
| state | OH |
| street1 | 10211 Brook Road |
| address[1] | |
| *doctype | DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress |
| @country | USA |
| city | closed |
| landmark | Ohio River-Bank Square |
| postalCode | 22130 |
| state | OH |
| street1 | 10211 Brook Road |
| name | Midwest Extreme Sports |
| phoneNum | |
| *body | 216-741-7566 |
| @cc | 011 |
| rep | Laura M. Sanchez |
| serialNum [ ] | |
| serialNum[0] | 19970523A |

To an XML document string, whose content looks like:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
```

```
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

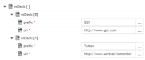
## Input Parameters

*document*

**Document**. Document that is to be converted to XML. If you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

*nsDecls [ ]*

**Document**. Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



| | |
|---|---|
| nsDecls [ ] | |
| nsDecls [0] | |
| prefix * | GSX |
| uri * | http://www.gsx.com |
| nsDecls [1] | |
| prefix * | TxMon |
| uri * | http://www.acrtrak/txmonitor |

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

xmlns:gsx="http://www.gsx.com"

xmlns:TxMon="http:www.acrtrak/txMonitor"

Alternatively, you can declare a namespace by including an @xmlns key in document. If you were not using the @ character to designate attributes, use the correct attribute prefix in your code.

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

*addHeader*

**String**. Optional.

Flag specifying whether the header element <?xml version="1.0"?> is to be included in the resulting XML String.

Set to:

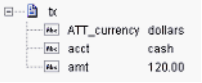
*true* to include the header. This is the default.

*false* to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

*attrPrefix*

**String** Optional. Prefix that designates keys containing attributes. The default prefix is "@".

For example, if you set *attrPrefix* to `ATT_` and *document* contained the following element:



tx

| | |
|---|---|
| ATT_currency | dollars |
| acct | cash |
| amt | 120.00 |

documentToXMLString would convert the `ATT_currency` key to the attribute, `currency=dollars`, in the `<tx>` element as shown below:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
</tx>
```

*encode*

**String** Optional. Flag indicating whether to HTML-encode the data. Set this parameter to true if your XML data contains special characters, including the following: `< > & " '`

Set to:

- `true` to HTML-encode the data.

  For example, the string expression `5 < 6` would be converted to `<expr>5 &lt; 6</expr>`, which is valid.

  If you do not want a leading `&` (ampersand) character encoded when it appears as part of a character or entity reference, set *preserveRefs* to true.

- `false` to not HTML-encode the data. This is the default.

  For example, the string expression `5 < 6`would be converted to `<expr>5 < 6</expr>`, which is invalid.

*preserveRefs*

**String** Optional. Flag indicating whether the leading `&` (ampersand) of a well-formed entity or character reference is left as `&` or further encoded as `&amp;` when the data is to be HTML-encoded (*encode* is set to `true`).

Set to:

- `true` to preserve the leading `&` (ampersand) in an entity or character reference when the service HTML-encodes the data.
- `false` to encode the leading & (ampersand) as &amp; when the & appears in an entity or character reference. This is the default.

The service ignores the value of *preseveRefs* when *encode* is set to false.

*enforceLegalXML*

**String** Optional. Flag indicating whether the service throws an exception when *document* contains multiple root elements or illegal XML tag names. Set to:

- `true` to throw an exception if *document* would produce an XML String containing multiple root elements and/or illegal XML tag names.
- `false` to allow the resulting XML String to contain multiple root elements and/or illegal XML tag names. You would use this setting, for example, to create an XML fragment composed of multiple elements that were not all enclosed within a root element. This is the default.

## Output Parameters

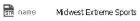
*xmlString*

**Object**. XML document string produced from document.

**Usage Notes**

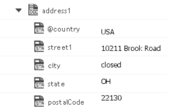
If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in document as shown below:



| | |
|---|---|
| name | Midwest Extreme Sports |

If you want to generate an element that contains children, represent with a document in the document as shown below:



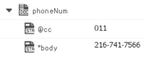
| | |
|---|---|
| address1 | |
| @country | USA |
| street1 | 10211 Brook Road |
| city | closed |
| state | OH |
| postalCode | 22130 |

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named *body that contains the element's value.

For example, if you want to produce the following element:

<phoneNum cc=011>216-741-7566</phoneNum>

You would include the following in document:



| | |
|---|---|
| phoneNum | |
| @cc | 011 |
| *body | 216-741-7566 |

To include namespaces, ensure that you do the following:

Include the appropriate namespace prefix in the key names in document. For example, to produce an element called acctNum that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named GSX:acctNum in document.

Define the URIs for the prefixes that appear in document. You can do this through nsDecls or by including an @xmlns key in the element where you want the xmlns attribute to be inserted.

## xmlBytesToDocument

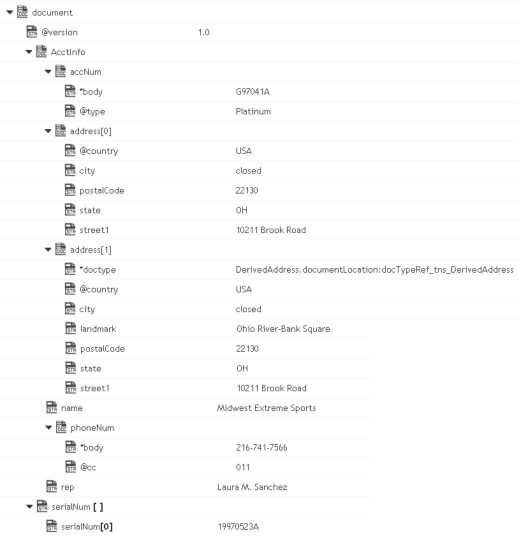
Converts XML content bytes (byte array) to a document. This service transforms each element and attribute in XML content bytes to an element in a Document.

This service will convert XML bytes containing the following XML content:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
```

```
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

To a Document that looks like:

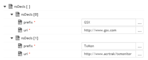


```
▼ document
    @version              1.0
  ▼ AcctInfo
    ▼ accNum
        *body                 G97041A
        @type                 Platinum
    ▼ address[0]
        @country              USA
        city                  closed
        postalCode            22130
        state                 OH
        street1               10211 Brook Road
    ▼ address[1]
        *doctype              DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
        @country              USA
        city                  closed
        landmark              Ohio River-Bank Square
        postalCode            22130
        state                 OH
        street1               10211 Brook Road
      name                  Midwest Extreme Sports
    ▼ phoneNum
        *body                 216-741-7566
        @cc                   011
      rep                   Laura M. Sanchez
  ▼ serialNum [ ]
      serialNum[0]          19970523A
```

## Input Parameters

*xmlBytes* **Object**. XML content bytes that is to be converted to a document.

*nsDecls [ ]* **Document**. Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in nsDecls . This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in nsDecls also define the prefixes used by the arrays, documents, documentTypeName, and collect parameters. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



```
▼ nsDecls [ ]
  ▼ nsDecls [0]
      prefix *        GSX                            ...
      uri *           http://www.gsx.com             ...
  ▼ nsDecls [1]
      prefix *        TxMon                          ...
      uri *           http://www.acrtrak/txmonitor   ...
```

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

`preserveUndeclaredNS` **String** Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- `False` to ignore namespace declarations in the XML document that are not specified in the *nsDecls* parameter. This is the default.

`preserveNSPositions` **String** Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the *nsDecls* parameter in the resulting document.

Set to:

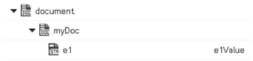
- `True` to preserve the position of namespaces declared in *nsDecls* in the resulting document. For each namespace specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute to the document as a String variable named "@xmlns:*NSprefix*" where "*NSprefix*" is the prefix name specified in *nsDecls*. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the xmlns attribute declaration within the XML document.
- `False` to not maintain the position of the namespace declarations specified in *nsDecls* in the resulting document. This is the default.

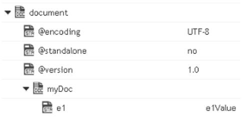
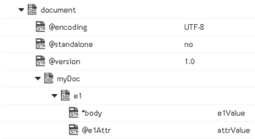
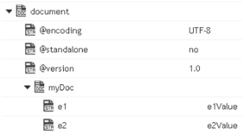
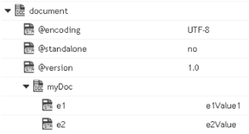
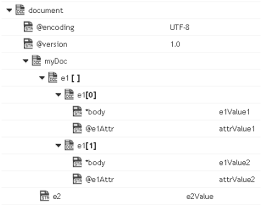
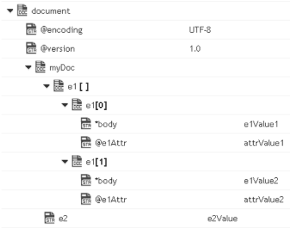
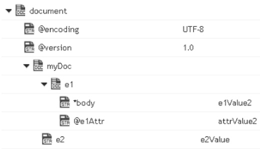
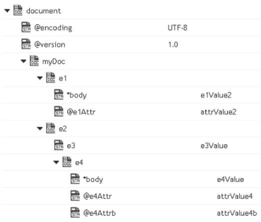
## Output Parameters

*document* **Document**. Document representation of nodes and attributes in node.

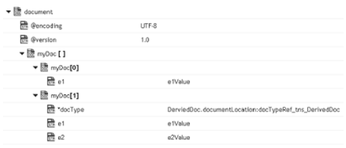
## Usage Notes

Following are examples of XML documents and the documents that *xmlBytesToDocument* will produce:

| XML Document | Document |
|---|---|
| `<myDoc><e1>e1Value</e1>`<br>`</myDoc>` | <br>document<br>myDoc<br>e1 e1Value |

| | |
|---|---|
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><myDoc><e1>e1Value<br></e1></myDoc> | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value |
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><myDoc><e1<br>e1Attr="attrValue"><br>e1Value</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value<br>@e1Attr attrValue |
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><myDoc><e1>e1Value<br></e1><e2>e2Value</e2></myDoc> | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value<br>e2 e2Value |
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><myDoc><e1>e1Value1<br></e1><e2>e2Value</e2><e1>e1Value2<br></e1></myDoc> | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value1<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><myDoc><e1 e1Attr="attrValue1">e1Value1<br></e1><e2>e2Value</e2><e1 e1Attr=<br>"attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1 **[ ]**<br>e1**[0]**<br>*body e1Value1<br>@e1Attr attrValue1<br>e1**[1]**<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><myDoc><e1 e1Attr="attrValue1"><br>e1Value1<br></e1><e2>e2Value</e2><e1<br>e1Attr="attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1 **[ ]**<br>e1**[0]**<br>*body e1Value1<br>@e1Attr attrValue1<br>e1**[1]**<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><myDoc><e1 e1Attr="attrValue1">e1Value1<br></e1><e2>e2Value</e2><e1 e1Attr=<br>"attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><myDoc><e1 e1Attr="attrValue1">e1Value1<br></e1><e2><e3>e3Value</e3><br><e4 e4Attr="attrValue4"e4Attrb=<br>"attrValue4b"><br>e4Value</e4></e2></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value2<br>@e1Attr attrValue2<br>e2<br>e3 e3Value<br>e4<br>*body e4Value<br>@e4Attr attrValue4<br>@e4Attrb attrValue4b |

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><tns:AcctInfo>
xmlns:tns="http://localhost/
DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance">
<myDoc>
<e1>e1Value</e1></myDoc>
<myDoc xsi:type="tns:DerivedDoc">
<e1>e1Value</e1><e2>
e2Value</e2></myDoc>
</tns:AcctInfo>
```



| | |
|---|---|
| ▾ document | |
| @encoding | UTF-8 |
| @version | 1.0 |
| ▾ myDoc [ ] | |
| ▾ myDoc[0] | |
| e1 | e1Value |
| ▾ myDoc[1] | |
| *docType | DervledDoc.documentLocation:docTypeRef_tns_DerivedDoc |
| e1 | e1Value |
| e2 | e2Value |

## xmlNodeToDocument

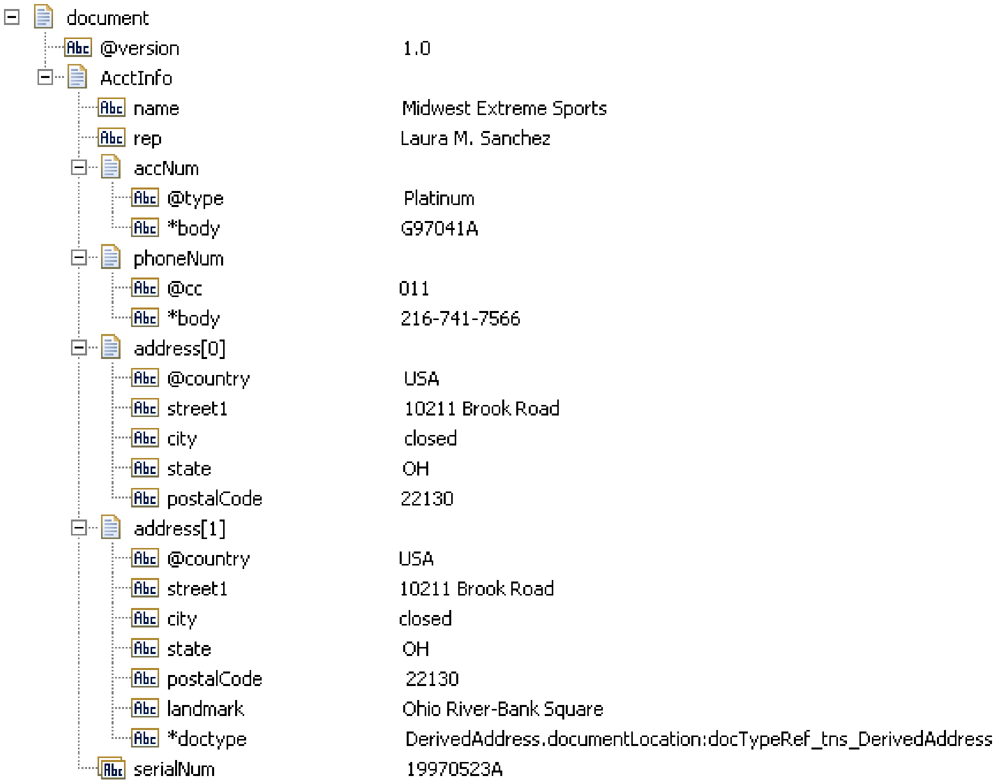
Converts an XML node to a document.

This service transforms each element and attribute in the XML node to an element in a Document. For example:

### This service would convert this XML document...

```
<?xml version="1.0" ?>
  <tns:AcctInfo>
      xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
      <name>Midwest Extreme Sports</name>
      <rep>Laura M. Sanchez</rep>
      <acctNum type=platinum>G97041A</acctNum>
      <phoneNum cc=011>216-741-7566</phoneNum>
      <address country=USA>
          <street1>10211 Brook Road</street1>
          <city>Cleveland</city>
          <state>OH</state>
          <postalCode>22130</postalCode>
      </address>
      <address country=USA xsi:type="tns:DerivedAddress">
      <street1>10211 Brook Road</street1>
          <city>Cleveland</city>
          <state>OH</state>
          <postalCode>22130</postalCode>
          <landMark>Besides Ohio River-Bank Square</landMark>
          <telNo>001222555</telNo>
     </address>
      <serialNum>19970523A</serialNum>
      <serialNum>20001106G</serialNum>
      <serialNum>20010404K</serialNum>
  </tns:AcctInfo>
```

## To a document that looks like this...



```
document
  @version                 1.0
  AcctInfo
    name                   Midwest Extreme Sports
    rep                    Laura M. Sanchez
    accNum
      @type                Platinum
      *body                G97041A
    phoneNum
      @cc                  011
      *body                216-741-7566
    address[0]
      @country             USA
      street1              10211 Brook Road
      city                 closed
      state                OH
      postalCode           22130
    address[1]
      @country             USA
      street1              10211 Brook Road
      city                 closed
      state                OH
      postalCode           22130
      landmark             Ohio River-Bank Square
      *doctype             DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
    serialNum              19970523A
```

Note that:

- The XML version attribute is converted to an element named `@version`.
- The resulting document is given the same name as the XML document's root element (`AcctInfo` in the example above) and is a child of the *document* variable that this service returns.
- Simple elements (such as `<name>` and `<rep>` in the example above) are converted to String elements.
- Complex elements (that is, elements with children, such as `<address>` in the example above) and simple elements that have attributes (such as `<acctNum>` and `<phoneNum>`) are converted to documents. Note that keys derived from attributes are prefixed with a "@" character to distinguish them from keys derived from elements. Also note that when a simple element has an attribute, its value is placed in an element named `*body`.
- Repeated elements (such as `<serialNum>`) can be collected into arrays using the *makeArrays* and/or *arrays* parameters. See *makeArrays* and *arrays* below for additional information about producing arrays.
- While creating a document, the xmlNodeToDocument service assigns a value of emptyString to the fields that are empty in the document.

## Input Parameters

*node*

XML node that is to be converted to a document.

This parameter supports the following types of input:

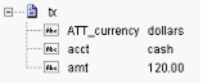
- **com.wm.lang.xml.Node**
- **org.w3c.dom.Node**

*attrPrefix*

**String** Optional. Prefix that is to be used to designate keys containing attribute values. The default is "@". For example, if you set *attrPrefix* to `ATT_` and *node* contained the following element:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
</tx>
```

xmlNodeToDocument would convert the `currency` attribute as follows:



```
tx
  ATT_currency  dollars
  acct          cash
  amt           120.00
```

*arrays[]*

**String List** Optional. Names of elements that are to be generated as arrays, regardless of whether they appear multiple times in *node*. For example, if *arrays* contained the following values for the XML document shown in the example in the description for this service:

```
rep
address
```

xmlNodeToDocument would generate element `rep` as a String List and element `address` as a Document List.

> **Important:**
> If you include namespace prefixes in the element names that you specify in *arrays*, you must define the namespaces associated with those prefixes in *nsDecls*.

*makeArrays*

**String** Optional. Flag indicating whether you want xmlNodeToDocument to automatically create an array for every element that appears in *node* more than once. Set to:

- `true` to automatically create arrays for every element that appears more than once in *node*. This is the default.
- `false` to create arrays for only those elements specified in *arrays*.

*collect*

**Document** Optional. Elements that are to be placed into a new, named array (that is, a "collection"). Within *collect*, use key names to specify the names of the elements that are to be included in the collection. Then set the value of each key to specify the name of the collection in which you want that element placed. For example, if you wanted to place the `<name>` and `<rep>` elements in an array called `originator`, you would set *collect* as follows:

| Key | Value |
|---|---|
| *name* | `originator` |
| *rep* | `originator` |

If the set of elements in a collection are all simple elements, a String List is produced. However, if the set is made up of complex elements, or a combination of simple and complex elements, a Document List is produced. When this is the case, each member of the array will include a child element called `*name` that contains the name of the element from which that member was derived.

You may optionally include namespace prefixes in the element names that you specify in *collect*; however, if you do, you must define the namespaces associated with those prefixes in *nsDecls*.

**Important:**
You cannot include an element in more than one collection.

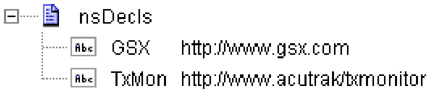
*nsDecls*

**Document** Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting Document. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting Document (for example, `GSX:acctNum`), you would associate the prefix GSX with that namespace in *nsDecls*. (This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix.)

Namespace prefixes in *nsDecls* also define the prefixes used by the *arrays*, *documents*, and *collect* parameters.

Each entry in *nsDecls* represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI.

For example, to define the URIs associated with two prefixes called `GSX` and `TxMon`, you would set *nsDecls* as follows:



nsDecls
GSX http://www.gsx.com
TxMon http://www.acutrak/txmonitor

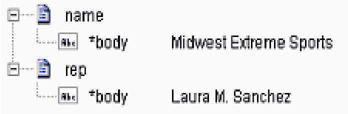
*documents[]*

**String List** Optional. Names of any simple elements that are to be generated as documents instead of Strings. The document produced for each element specified in *documents[]* will have the same name as the source element from which it is derived. It will contain a String element named `*body` that holds the element's value.

For example, if *documents[]* contained the Strings `name` and `rep` and the source document contained the following:

```
.
.
.
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
.
.
.
```

xmlNodeToDocument would produce the following:



name
*body Midwest Extreme Sports
rep
*body Laura M. Sanchez

**Note:**
If you include namespace prefixes in the element names that you specify, you must define the namespaces associated with those prefixes in *nsDecls*.

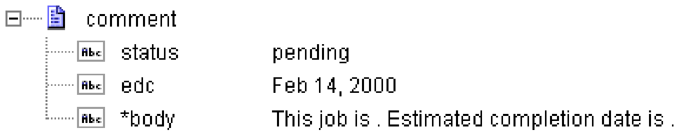
*mixedModel*

**String** Optional. Flag specifying how mixed-content elements (elements containing both text values and child elements) are to be converted. The following is an example of a mixed-content element:

```
<comment>
This job is <status>pending</status>. Estimated
completion date is <edc>Feb 14, 2000</edc>.
</comment>
```

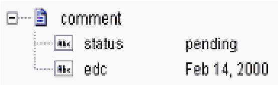
Set to:

- `true` to place top-level text in an element named `*body`. This setting would produce the following Document for the `<comment>` element shown above:



comment<br>
status — pending<br>
edc — Feb 14, 2000<br>
*body — This job is . Estimated completion date is .

- `false` to omit top-level text and include only the child elements from mixed-content elements. This setting would produce the following Document for the `<comment>` element shown above:



comment<br>
status — pending<br>
edc — Feb 14, 2000

*preserveUndeclaredNS*

**String** Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- `False` to ignore namespace declarations in the XML document that are not specified in the *nsDecls* parameter. This is the default.

*preserveNSPositions*

**String** Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the *nsDecls* parameter in the resulting document.

Set to:

- `True` to preserve the position of namespaces declared in *nsDecls* in the resulting document. For each namespace specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute to the document (IData) as a String variable named "@xmlns:*NSprefix*" where "*NSprefix*" is the prefix name specified in *nsDecls*. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the xmlns attribute declaration within the XML document.

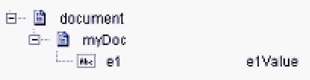
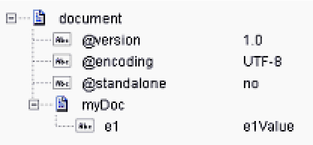
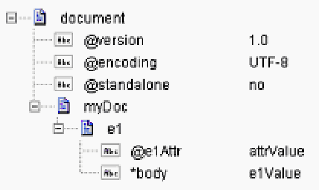
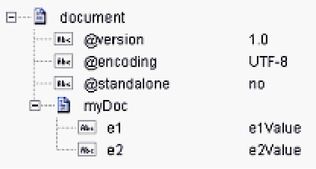
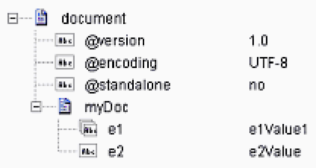
- `False` to not maintain the position of the namespace declarations specified in *nsDecls* in the resulting document. This is the default.

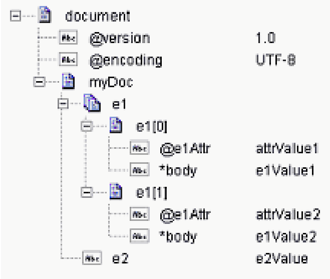
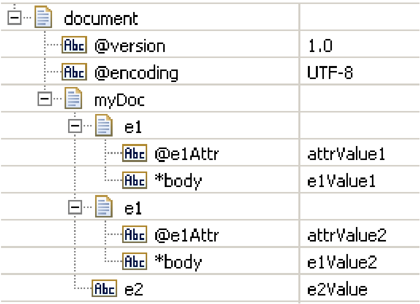
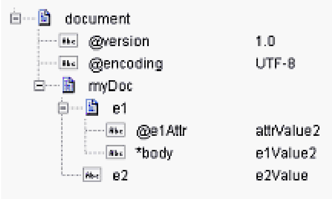
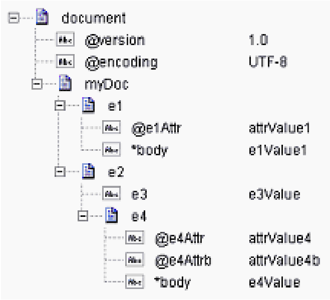
## Output Parameters

*document* **Document** Document representation of the nodes and attributes in node.

## Examples

Following are examples of XML documents and the documents that xmlNodeToDocument would produce.

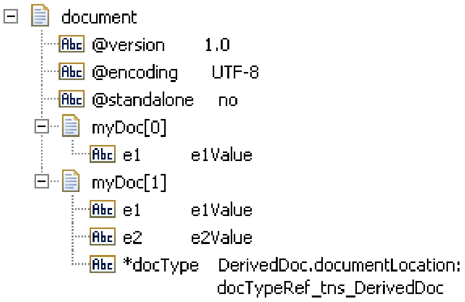
| XML Document | Output from xmlNodeToDocument |
|---|---|
| `<myDoc>`<br>`<e1>e1Value</e1>`<br>`</myDoc>` | <br>document<br>myDoc<br>e1 e1Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?>`<br>`<myDoc>`<br>`<e1>e1Value</e1>`<br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>@standalone no<br>myDoc<br>e1 e1Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?>`<br>`<myDoc>`<br>`<e1 e1Attr="attrValue">e1Value</e1>`<br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>@standalone no<br>myDoc<br>e1<br>@e1Attr attrValue<br>*body e1Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?>`<br>`<myDoc>`<br>`<e1>e1Value</e1>`<br>`<e2>e2Value</e2>`<br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>@standalone no<br>myDoc<br>e1 e1Value<br>e2 e2Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?>`<br>`<myDoc>`<br>`<e1>e1Value1</e1>`<br>`<e2>e2Value</e2>`<br>`<e1>e1Value2</e1>`<br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>@standalone no<br>myDoc<br>e1 e1Value1<br>e2 e2Value |

| XML Document | Output from xmlNodeToDocument |
|---|---|
| `<?xml version="1.0"`<br>`encoding="UTF-8"?>`<br>`<myDoc>`<br>`<e1 e1Attr="attrValue1">e1Value1</e1>`<br><br>`<e2>e2Value</e2>`<br>`<e1 e1Attr="attrValue2">e1Value2</e1>`<br><br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>myDoc<br>e1<br>e1[0]<br>@e1Attr attrValue1<br>*body e1Value1<br>e1[1]<br>@e1Attr attrValue2<br>*body e1Value2<br>e2 e2Value |
| | **Note:**<br>This example assumes that *makeArrays* is set to true. Note that *e1* was created as a document list, which holds both `<e1>` elements from the XML document. |
| `<?xml version="1.0"`<br>`encoding="UTF-8"?>`<br>`<myDoc>`<br>`<e1 e1Attr="attrValue1">e1Value1</e1>`<br><br>`<e2>e2Value</e2>`<br>`<e1 e1Attr="attrValue2">e1Value2</e1>`<br><br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>myDoc<br>e1<br>@e1Attr attrValue1<br>*body e1Value1<br>e1<br>@e1Attr attrValue2<br>*body e1Value2<br>e2 e2Value |
| `<?xml version="1.0"`<br>`encoding="UTF-8"?>`<br>`<myDoc>`<br>`<e1 e1Attr="attrValue1">e1Value1</e1>`<br><br>`<e2>e2Value</e2>`<br>`<e1 e1Attr="attrValue2">e1Value2</e1>`<br><br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>myDoc<br>e1<br>@e1Attr attrValue2<br>*body e1Value2<br>e2 e2Value |
| `<?xml version="1.0"`<br>`encoding="UTF-8"?>`<br>`<myDoc>`<br>`<e1 e1Attr="attrValue1">e1Value1</e1>`<br><br>`<e2>`<br>`<e3>e3Value</e3>`<br>`<e4 e4Attr="attrValue4"`<br>`e4Attrb="attrValue4b">e4Value`<br>`</e4>`<br>`</e2>`<br>`</myDoc>` | <br>document<br>@version 1.0<br>@encoding UTF-8<br>myDoc<br>e1<br>@e1Attr attrValue1<br>*body e1Value1<br>e2<br>e3 e3Value<br>e4<br>@e4Attr attrValue4<br>@e4Attrb attrValue4b<br>*body e4Value |

**XML Document**

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
  <tns:AcctInfo>
xmlns:tns="http://localhost/Derived
Address/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" >
    <myDoc>    <e1>e1Value</e1>
    </myDoc>
    <myDoc xsi:type="tns:DerivedDoc">

    <e1>e1Value</e1>
    <e2>e1Value</e2>
    </myDoc>
  </tns:AcctInfo>
```

**Output from xmlNodeToDocument**



```
document
  @version      1.0
  @encoding     UTF-8
  @standalone   no
  myDoc[0]
    e1          e1Value
  myDoc[1]
    e1          e1Value
    e2          e2Value
    *docType    DerivedDoc.documentLocation:
                docTypeRef_tns_DerivedDoc
```

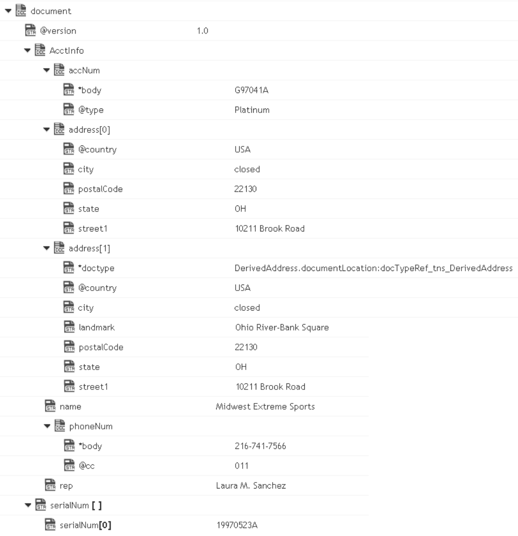
## xmlStreamToDocument

Converts an XML content stream to a document. This service transforms each element and attribute in the XML content stream to an element in a Document.

This service will convert the XML stream containing the following XML content:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

To a Document that looks like:

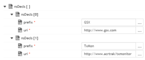


```
▾ document
    @version                     1.0
  ▾ AcctInfo
    ▾ accNum
        *body                        G97041A
        @type                        Platinum
    ▾ address[0]
        @country                     USA
        city                         closed
        postalCode                   22130
        state                        OH
        street1                      10211 Brook Road
    ▾ address[1]
        *doctype                     DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
        @country                     USA
        city                         closed
        landmark                     Ohio River-Bank Square
        postalCode                   22130
        state                        OH
        street1                      10211 Brook Road
      name                       Midwest Extreme Sports
    ▾ phoneNum
        *body                        216-741-7566
        @cc                          011
      rep                        Laura M. Sanchez
  ▾ serialNum [ ]
      serialNum[0]               19970523A
```

## Input Parameters

*xmlStream* **java.io.InputStream**. XML content stream that is to be converted to a document.

*nsDecls [ ]* **Document**. Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in nsDecls . This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in nsDecls also define the prefixes used by the arrays, documents, documentTypeName, and collect parameters. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



```
▾ nsDecls [ ]
  ▾ nsDecls [0]
      prefix *        GSX                           ...
      uri *           http://www.gsx.com            ...
  ▾ nsDecls [1]
      prefix *        TxMon                         ...
      uri *           http://www.acrtrak/txmonitor  ...
```

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

`preserveUndeclaredNS` **String** Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- `False` to ignore namespace declarations in the XML document that are not specified in the *nsDecls* parameter. This is the default.

`preserveNS Positions` **String** Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the *nsDecls* parameter in the resulting document.

Set to:

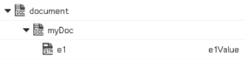
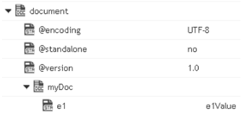
- `True` to preserve the position of namespaces declared in *nsDecls* in the resulting document. For each namespace specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute to the document as a String variable named "@xmlns:*NSprefix*" where "*NSprefix*" is the prefix name specified in *nsDecls*. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the xmlns attribute declaration within the XML document.
- `False` to not maintain the position of the namespace declarations specified in *nsDecls* in the resulting document. This is the default.

## Output Parameters

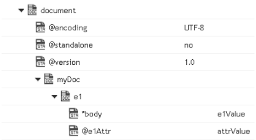
*document* **Document**. Document representation of nodes and attributes in node.

## Usage Notes

Following are examples of XML documents and the documents that *xmlStreamToDocument* will produce:

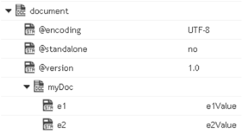
| XML Document | Document |
|---|---|
| `<myDoc><e1>e1Value</e1></myDoc>` | <br>document<br>myDoc<br>e1 e1Value |
| `<?xml version="1.0" encoding="UTF-8" standalone="no"?><myDoc><e1>e1Value</e1></myDoc>` | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value |

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc>
<e1 e1Attr="attrValue">
e1Value</e1></myDoc>
```



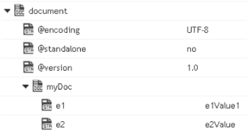
```
document
    @encoding       UTF-8
    @standalone     no
    @version        1.0
    myDoc
        e1
            *body           e1Value
            @e1Attr         attrValue
```

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc>
<e1>e1Value</e1><e2>e2Value</e2>
</myDoc>
```



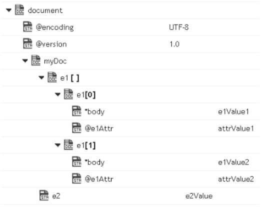
```
document
    @encoding       UTF-8
    @standalone     no
    @version        1.0
    myDoc
        e1                  e1Value
        e2                  e2Value
```

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>
e1Value1</e1><e2>
e2Value</e2><e1>
e1Value2</e1></myDoc>
```



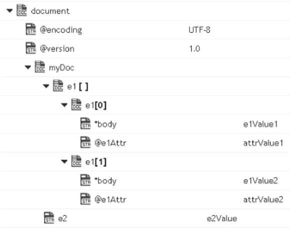
```
document
    @encoding       UTF-8
    @standalone     no
    @version        1.0
    myDoc
        e1                  e1Value1
        e2                  e2Value
```

```
<?xml version="1.0"encoding="UTF-8"?>

<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">e1Value2
</e1></myDoc>
```



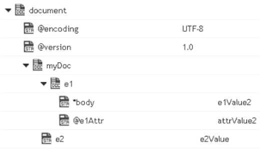
```
document
    @encoding       UTF-8
    @version        1.0
    myDoc
        e1 [ ]
            e1[0]
                *body           e1Value1
                @e1Attr         attrValue1
            e1[1]
                *body           e1Value2
                @e1Attr         attrValue2
        e2              e2Value
```

```
<?xml version="1.0"encoding="UTF-8"?>

<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>
e2Value</e2><e1 e1Attr="attrValue2">
e1Value2</e1></myDoc>
```



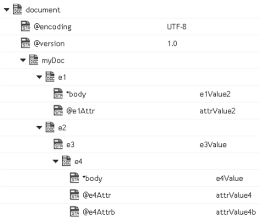
```
document
    @encoding       UTF-8
    @version        1.0
    myDoc
        e1 [ ]
            e1[0]
                *body           e1Value1
                @e1Attr         attrValue1
            e1[1]
                *body           e1Value2
                @e1Attr         attrValue2
        e2              e2Value
```

```
<?xml version="1.0"encoding="UTF-8"?>

<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">
e1Value2</e1>
</myDoc>
```



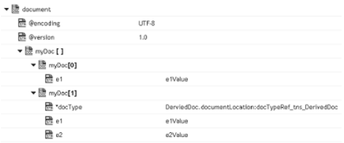
```
document
    @encoding       UTF-8
    @version        1.0
    myDoc
        e1
            *body           e1Value2
            @e1Attr         attrValue2
        e2              e2Value
```

```
<?xml version="1.0"encoding="UTF-8"?>

<myDoc><e1 e1Attr="attrValue1">
e1Value1
</e1><e2><e3>e3Value</e3>
<e4 e4Attr=
"attrValue4"e4Attrb="attrValue4b">
e4Value
</e4></e2></myDoc>
```



```
document
    @encoding       UTF-8
    @version        1.0
    myDoc
        e1
            *body           e1Value2
            @e1Attr         attrValue2
        e2
            e3              e3Value
            e4
                *body           e4Value
                @e4Attr         attrValue4
                @e4Attrb        attrValue4b
```

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><tns:AcctInfo>
xmlns:tns="http://localhost/
DerivedAddress/
schema.xsd"xmlns:xsi="http://www
```



```
document
    @encoding       UTF-8
    @version        1.0
    myDoc [ ]
        myDoc[0]
            e1              e1Value
        myDoc[1]
            *docType        DervиedDoc.documentLocation:docTypeRef_tns_DerivedDoc
            e1              e1Value
            e2              e2Value
```

```
.w3.org/
2001/XMLSchema-instance"><myDoc>
<e1>e1Value</e1></myDoc>
<myDoc xsi:type="tns:DerivedDoc">
<e1>e1Value</e1><e2>e2Value</e2>
</myDoc>
</tns:AcctInfo>
```

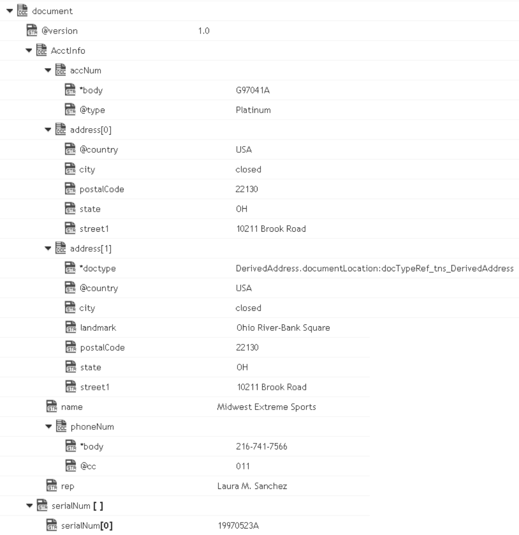
## xmlStringToDocument

Converts an XML string to a document. This service transforms each element and attribute in the XML string to an element in a Document.

This service will convert the following XML string:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

To a Document that looks like:



```
▼ document
    @version                 1.0
  ▼ AcctInfo
    ▼ accNum
        *body                      G97041A
        @type                      Platinum
    ▼ address[0]
        @country                   USA
        city                       closed
        postalCode                 22130
        state                      OH
        street1                    10211 Brook Road
    ▼ address[1]
        *doctype                   DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
        @country                   USA
        city                       closed
        landmark                   Ohio River-Bank Square
        postalCode                 22130
        state                      OH
        street1                    10211 Brook Road
      name                       Midwest Extreme Sports
    ▼ phoneNum
        *body                      216-741-7566
        @cc                        011
      rep                        Laura M. Sanchez
  ▼ serialNum [ ]
      serialNum[0]               19970523A
```

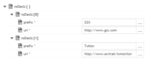
## Input Parameters

*xmlString*

**String**. XML string that is to be converted to a document.

*nsDecls [ ]*

**Document**. Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in nsDecls . This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in nsDecls also define the prefixes used by the arrays, documents, documentTypeName, and collect parameters. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:



```
▼ nsDecls [ ]
  ▼ nsDecls [0]
      prefix *        GSX                            ...
      uri *           http://www.gsx.com             ...
  ▼ nsDecls [1]
      prefix *        TxMon                          ...
      uri *           http://www.acrtrak/txmonitor   ...
```

Parameters for *nsDecls [ ]* are:

**prefix:** Key name.

**uri:** Key value.

`preserveUndeclaredNS`

**String** Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.

- `False` to ignore namespace declarations in the XML document that are not specified in the *nsDecls* parameter. This is the default.

`preserveNSPositions`

**String** Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the *nsDecls* parameter in the resulting document.

Set to:

- `True` to preserve the position of namespaces declared in *nsDecls* in the resulting document. For each namespace specified in the *nsDecls* parameter, Integration Cloud adds the xmlns attribute to the document as a String variable named "@xmlns:*NSprefix*" where "*NSprefix*" is the prefix name specified in *nsDecls*. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the xmlns attribute declaration within the XML document.

- `False` to not maintain the position of the namespace declarations specified in *nsDecls* in the resulting document. This is the default.

*arrays [ ]*

**String List** Optional. Names of elements that are to be generated as arrays, regardless of whether they appear multiple times. For example, if *arrays* contained the following values for the XML document shown in the example in the description for this service:

```
rep
address
```

xmlStringToDocument would generate element `rep` as a String List and element `address` as a Document List.

> **Important:**
> If you include namespace prefixes in the element names that you specify in *arrays*, you must define the namespaces associated with those prefixes in *nsDecls*.

*makeArrays* **String** Optional. Flag indicating whether you want xmlStringToDocument to automatically create an array for every element that appears more than once. Set to:

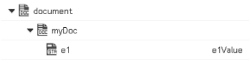
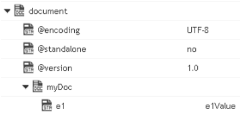
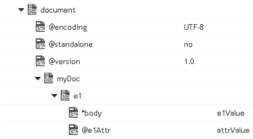
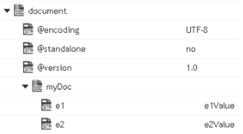
- `true` to automatically create arrays for every element that appears more than once. This is the default.
- `false` to create arrays for only those elements specified in *arrays*.

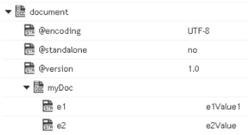
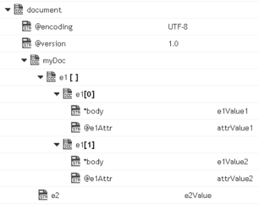
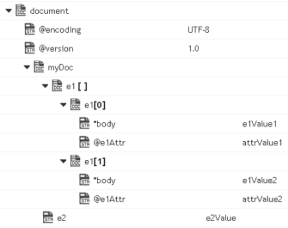
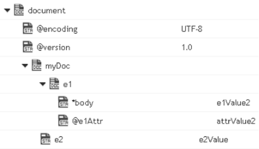
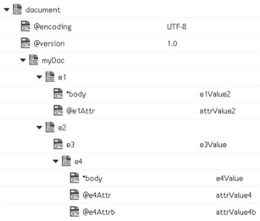
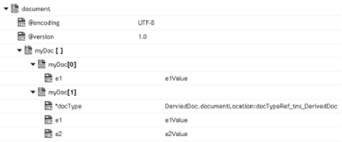
## Output Parameters

*document* **Document**. Document representation of nodes and attributes in node.

## Usage Notes

Following are examples of XML documents and the documents that *xmlStringToDocument* will produce:

| XML Document | Document |
|---|---|
| `<myDoc><e1>e1Value</e1>`<br>`</myDoc>` | <br>document<br>myDoc<br>e1 e1Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?><myDoc><e1>`<br>`e1Value</e1></myDoc>` | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?><myDoc>`<br>`<e1 e1Attr="attrValue">`<br>`e1Value</e1></myDoc>` | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value<br>@e1Attr attrValue |
| `<?xml version="1.0" encoding="UTF-8"`<br>`standalone="no"?><myDoc><e1>`<br>`e1Value</e1><e2>e2Value</e2>`<br>`</myDoc>` | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value<br>e2 e2Value |

| | |
|---|---|
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><myDoc><e1>e1Value1<br></e1><e2>e2Value</e2><e1><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@standalone no<br>@version 1.0<br>myDoc<br>e1 e1Value1<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><br><myDoc><e1 e1Attr="attrValue1"><br>e1Value1</e1><e2>e2Value</e2><br><e1 e1Attr="attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1 [ ]<br>e1[0]<br>*body e1Value1<br>@e1Attr attrValue1<br>e1[1]<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><br><myDoc><e1 e1Attr="attrValue1"><br>e1Value1</e1><e2>e2Value</e2><br><e1 e1Attr="attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1 [ ]<br>e1[0]<br>*body e1Value1<br>@e1Attr attrValue1<br>e1[1]<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><br><myDoc><e1 e1Attr="attrValue1"><br>e1Value1</e1><e2>e2Value</e2><br><e1 e1Attr="attrValue2"><br>e1Value2</e1></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value2<br>@e1Attr attrValue2<br>e2 e2Value |
| <?xml version="1.0"encoding="UTF-8"?><br><br><myDoc><e1 e1Attr="attrValue1"><br>e1Value1</e1><e2><e3>e3Value<br></e3><e4 e4Attr="attrValue4"<br>e4Attrb=<br>"attrValue4b">e4Value</e4><br></e2></myDoc> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc<br>e1<br>*body e1Value2<br>@e1Attr attrValue2<br>e2<br>e3 e3Value<br>e4<br>*body e4Value<br>@e4Attr attrValue4<br>@e4Attrb attrValue4b |
| <?xml version="1.0" encoding="UTF-8"<br>standalone="no"?><tns:AcctInfo><br>xmlns:tns="http://localhost/<br>DerivedAddress/<br>schema.xsd"xmlns:xsi="http:<br>//www.w3.org/2001/<br>XMLSchema-instance"><myDoc><br><e1>e1Value</e1></myDoc><br><myDoc xsi:type="tns:DerivedDoc"><br><e1>e1Value</e1><e2>e2Value<br></e2></myDoc><br></tns:AcctInfo> | <br>document<br>@encoding UTF-8<br>@version 1.0<br>myDoc [ ]<br>myDoc[0]<br>e1 e1Value<br>myDoc[1]<br>*docType DerviedDoc.documentLocation:docTypeRef_tns_DerivedDoc<br>e1 e1Value<br>e2 e2Value |

## xmlStringToXMLNode

Converts a String, byte[], or InputStream containing an XML document to an XML node.

An XML node is a representation of an XML document that can be consumed by Integration Cloud.

## Input Parameters

*xmldata*

**String** Optional. String containing the XML document to convert to an XML node.

> **Note:**
> If you specify *xmldata*, do not specify *$filedata* or *$filestream*.

*$filedata*

**byte[ ]** Optional. byte[ ] containing the XML document to convert to an XML node.

> **Note:**
> If you specify *$filedata*, do not specify *xmldata* or *$filestream*.

*$filestream*

**java.io.InputStream** Optional. InputStream containing the XML document to convert to an XML node.

> **Note:**
> If you specify *$filestream*, do not specify *xmldata* or *$filedata*.

*encoding*

**String** Optional. Character encoding in which text is represented. Specify UTF-8 for XML files and ISO-8859-1 for HTML files. To have the parser attempt to detect the type of encoding, specify `autoDetect` (which is the default, if *encoding* is not specified).

*isXML*

**String** Optional. Flag specifying whether the input document is XML or HTML. (xmlStringToXMLNode must know this so that it can parse the document correctly.) Set to:

- `autoDetect` to parse the document based on its type. When you use this option, xmlStringToXMLNode detects the document's type based on its document type declaration as indicated by a `<!DOCTYPE...\>` or `<?XML...\>` tag. If it cannot determine the document type, it parses it as HTML. This is the default.
- `true` to parse the document as XML.
- `false` to parse the document as HTML.

## Output Parameters

*node*

**com.wm.lang.xml.Node** XML node representation of the XML document in *xmlData*. This object can be used as input to webMethods services that consume XML nodes.

### Usage Notes

The input parameters *xmldata*, *$filedata*, and *$filestream* are mutually exclusive. Specify only one of the preceding parameters. Integration Cloud checks the parameters in the following order: *$filedata*, *$filestream*, and *xmldata*, and uses the value of the first parameter with a value.

## getXMLNodeType

Returns information about an XML node.

### Input Parameters

| | |
|---|---|
| *rootNode* | **com.wm.lang.xml.Document** XML node about which you want information. |

### Output Parameters

| | |
|---|---|
| *systemID* | **String** Conditional. System identifier, as provided by the DTD associated with *rootNode*. If *rootNode* does not have a system identifier, this value is null. |
| *publicID* | **String** Conditional. Public identifier, as provided by the DTD associated with *rootNode*. If *rootNode* does not have a public identifier, this value is null. |
| *rootNamespace* | **String** URI of the XML namespace to which *rootNode's* root element belongs. |
| *rootNSPrefix* | **String** Conditional. Namespace prefix of root element in *rootNode*, if any. |
| *rootLocalName* | **String** Conditional. Local name (excluding the namespace prefix) of the root element in *rootNode*, if any. |

## queryXMLNode

Queries an XML node.

The *fields* parameter specifies how data is extracted from the node to produce an output variable. This output variable is called a "binding," because the *fields* parameter binds a certain part of the document to a particular output variable. At run time, this service must include at least one *fields* entry. The service must include at least one entry in *fields*. The result of each query you specify in *fields* is returned in a variable whose name and type you specify.

## Input Parameters

*node*

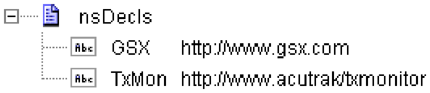
The XML node that you want to query. This parameter supports the following types of input:

- **com.wm.lang.xml.Node** XML node that you want to query. An XML node can be produced by xmlStringToXMLNode or an XML content handler.

*nsDecls*

**Document** Optional. Namespaces associated with any namespace prefixes used element to specify elements in *fields/query*. Each entry in *nsDecls* represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI.

For example, to define the URIs associated with two prefixes called `GSX` and `TxMon`, you would set *nsDecls* as follows:



nsDecls
GSX http://www.gsx.com
TxMon http://www.acutrak/txmonitor

*fields*

**Document List** Optional. Parameters describing how data is to be extracted from *node*. Each document in the list contains parameters for a single query, as follows:

<table>
<tr><th>Key</th><th>Description</th></tr>
<tr><td><i>name</i></td><td><b>String</b> Name to assign to resulting value.</td></tr>
<tr><td><i>resultType</i></td><td><b>String</b> Object type that the query is to yield. The following shows the allowed values for <i>resultType</i>.
<table>
<tr><th>Underlying Value</th><th>Corresponding Data Type</th></tr>
<tr><td><code>Object</code></td><td>Object</td></tr>
<tr><td><code>Object[]</code></td><td>Object List</td></tr>
<tr><td><code>Record</code></td><td>Document</td></tr>
<tr><td><code>Record[]</code></td><td>Document List</td></tr>
<tr><td><code>String</code></td><td>String</td></tr>
<tr><td><code>String[]</code></td><td>String List</td></tr>
<tr><td><code>String[][]</code></td><td>String Table</td></tr>
</table></td></tr>
<tr><td><i>query</i></td><td><b>String</b> Query identifying the data to be extracted from <i>node</i>.</td></tr>
</table>

*queryType*

**String** Query language in which *query* is expressed. Valid values are `WQL` and `XQL`.

*onnull*

**String** Code indicating what you want queryXMLNode to do when the result is null. Set to one of the following:

- `continue` to indicate that all result values are acceptable for this query (including null).
- `fail` to indicate that the service should fail if the result of this query is null and continue in all other cases.
- `succeed` to indicate that the service should continue if the result of this query is null and fail in all other cases.

*fields[]*

**Document List** Parameters that support recursive execution of bindings. Each *fields* list defines bindings for one level of the output with the top level being the pipeline and the first level down being contents of a document or document list in the pipeline.

### Output Parameters

**Document** Results from the queries specified in *fields*. This service returns one element for each query specified in *fields*. The specific names and types of the returned elements are determined by the *fields/name* and *field/resultType* parameters of the individual queries.

### Usage Notes

If queryXMLNode fails, it throws an exception. Common reasons for queryXMLNode to fail include:

- A variable that has no query string assigned to it.
- A syntax error in a query string.
- A query fails the "Allows Null" test.
- The node variable does not exist or it is null.

## IO Services

Use **IO** services to convert data between byte[ ], characters, and InputStream representations. These services are used for reading and writing bytes, characters, and streamed data to the file system. These services behave like the corresponding methods in the java.io.InputStream class.

These services can be invoked only by other services. Streams cannot be passed between clients and the server, so these services will not execute if they are invoked from a client.

The following **IO** services are available:

| Service | Description |
|---|---|
| bytesToStream | Converts a byte[ ] to java.io.ByteArrayInputStream. |
| streamToBytes | Creates a byte[ ] from data that is read from an InputStream. |
| streamToString | Creates a string from data that is read from an InputStream. |
| stringToStream | Converts a string to a binary stream. |

## bytesToStream

Converts a byte[ ] to java.io.ByteArrayInputStream.

### Input Parameters

| | |
|---|---|
| *bytes* | **byte[ ]** The byte array to convert. |
| *length* | **String** Optional. The maximum number of bytes to read and convert. If *length* is not specified, the default value for this parameter is the length of the input byte array. |
| *offset* | **String** Optional. The offset into the input byte array from which to start converting. If no value specified, the default value is zero. |

### Output Parameters

| | |
|---|---|
| *stream* | **java.io.ByteArrayInputStream** An open InputStream created from the contents of the input *bytes* parameter. |

### Usage Notes

This service constructs *stream* from the byte array using the constructor ByteArrayInputStream(byte[ ]). This constructor does not make a copy of the byte array, so any changes to *bytes* will be reflected in the data read from the stream.

## streamToBytes

Creates a byte[ ] from data that is read from an InputStream.

### Input Parameters

*stream* **java.io.InputStream** The InputStream that you want to convert.

### Output Parameters

*bytes* **byte[ ]**The bytes read from *stream*.

### Usage Notes

This service reads all of the bytes from *stream* until the end of file is reached, and then it closes the InputStream.

## streamToString

Creates a string from data that is read from an InputStream.

### Input Parameters

*inputStream* **java.io.InputStream** The InputStream to convert to a string.

*encoding* **String** Optional. Name of a registered, IANA character set (for example, ISO-8859-1). If you specify an unsupported encoding, the system throws an exception. If no value is specified the encoding will be UTF-8.

### Output Parameters

*string* **String** Data read from *inputStream* and converted to a string.

## stringToStream

Converts a string to a binary stream.

### Input Parameters

*string* **String** The string object to be converted.

*encoding* **String** Optional. Name of a registered, IANA character set, for example, ISO-8859-1. If you specify an unsupported encoding, the system throws an exception. If no value is specified, the encoding will be UTF-8.

### Output Parameters

| | |
|---|---|
| *inputStream* | **java.io.ByteArrayInputStream** An open InputStream created from the contents of *string*. |

## Utils Services

Contains utility services.

The following **Utils** services are available:

| Service | Description |
|---|---|
| generateUUID | Generates a random Universally Unique Identifier (UUID). |

## generateUUID

Generates a random Universally Unique Identifier (UUID).

### Input Parameters

None.

### Output Parameters

| | |
|---|---|
| *UUID* | **String** A randomly generated Universally Unique Identifier (UUID). |

# 11 Document Types

## Overview

A **Document Type** contains a set of fields used to define the structure and type of data in a document. You use a document type to specify the input or output parameters for an Integration.

> **Note:**
> Users who have the required project permissions under **Settings** ⚙ **> Project Permissions** can create, update, and delete a document type.

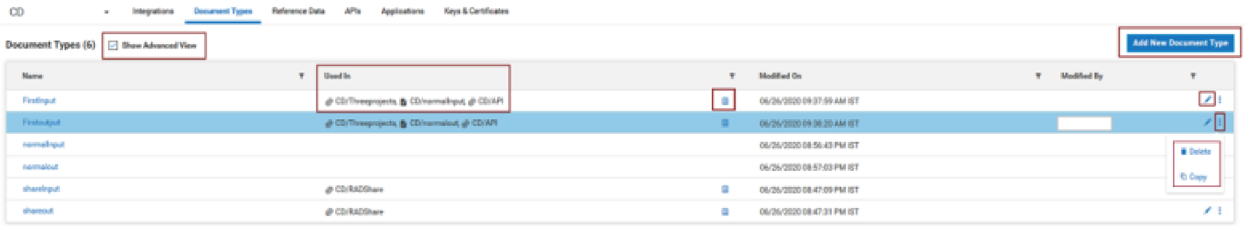
Document types provide the following benefits:

- Using a document type as the input or output signature for an Integration reduces the effort required to build an Integration.
- Using a document type to build document or document list fields reduces the effort and time needed to declare input or output parameters or build other document fields.
- Document types improve accuracy because there is less possibility to introduce a typing error while typing field names.
- Document types make future changes easier to implement because you make a change in one place (the Document Type) rather than everywhere the document type is used.

You use document types to define the input or output parameters for an integration. Input and output parameters are the names and types of fields that the integration requires as input and generates as output. These parameters are also collectively referred to as a signature. For example, an integration takes two string values, an account number (AcctNum ) and a dollar amount (OrderTotal ) as inputs and produces an authorization code (AuthCode ) as the output. If you have multiple integrations with identical input parameters but different output parameters, you use a document type to define the input parameters rather than manually specifying individual input fields for each integration.

### Document Types page

The **Document Types** page lists all the document types. The **Name** column in the **Document Types** page displays the name of the document type. You can select a document type and click the document type name link under the **Name** column to modify the document type. The **Document Types** page by default shows a basic view of all the document types. Click **Show Advanced View** to view the **Used In** column, which displays in which project and where the document type is used in the format *project name/referenced asset name*. You can click the **Used In** icon 🗏 to get more information. Click the ⋮ icon to delete or copy a document type or the ✎ icon to edit a document type.



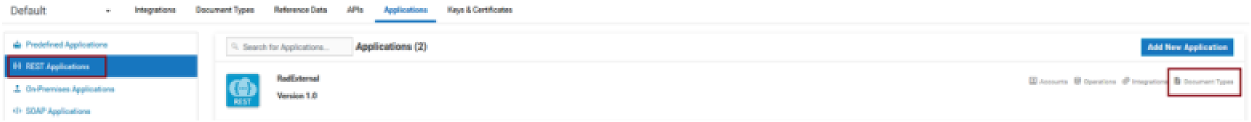
| Name | Used In | Modified On | Modified By |
|---|---|---|---|
| FirstInput | CD/Threeprojects, CD/normalInput, CD/API | 06/26/2020 09:37:59 AM IST | |
| FirstOutput | CD/Threeprojects, CD/normalout, CD/API | 06/26/2020 09:38:20 AM IST | |
| normalInput | | 06/26/2020 08:56:43 PM IST | |
| normalout | | 06/26/2020 08:57:03 PM IST | |
| shareInput | CD/RADShare | 06/26/2020 08:47:09 PM IST | |
| shareout | CD/RADShare | 06/26/2020 08:47:31 PM IST | |

### Creating a Document Type

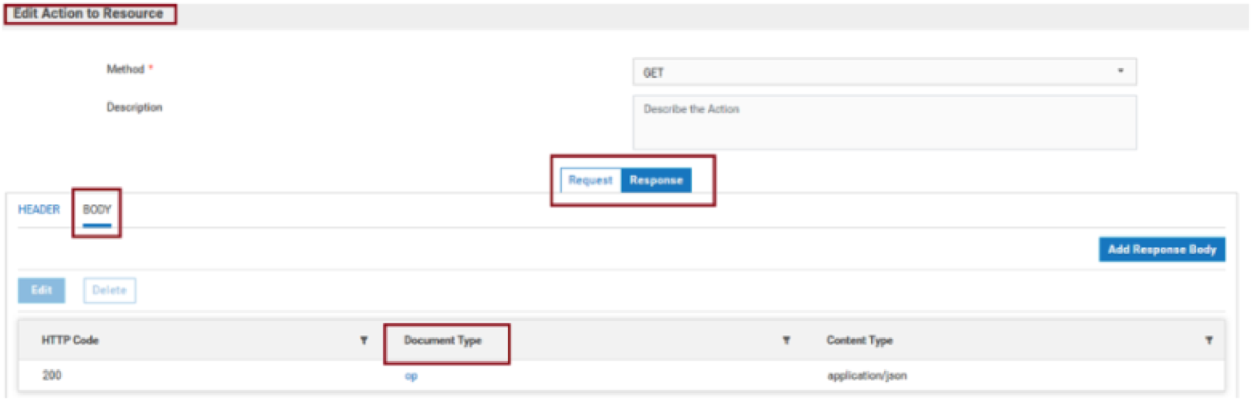
You can create a document type from the **Document Types** page by clicking the **Add New Document Type** option in the following ways:

- **Build from scratch**: Create an empty document type and define the structure of the document type yourself by inserting fields to define its contents and structure.
- **Build from XML schema**: Create a document type from a source file, such as an XML Schema. The structure and content of the document type will match that of the source file.

Integration Cloud also allows you to create document types for already created REST Applications from the **Projects > <Select a Project> > Applications > REST Applications > Document Types** page or from the Request Body and Response Body panels while creating a REST Application.



Document types created for a REST Application do not appear in the **Projects > <Select a Project> > Document Types** page but appears in the **Document Types** panel for the selected REST Application. Document types for REST Applications are used in the Request Body and Response Body of an **Action**.



Method * GET

Description Describe the Action

| HTTP Code | Document Type | Content Type |
|---|---|---|
| 200 | op | application/json |

## Creating Document Types from Scratch

### To add or edit a Document Type from scratch

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Document Types**. The **Document Types** page appears.

   **Note:**
   You can create document types for already created REST Applications from the **Projects > <Select a Project> > Applications > REST Applications > Document Types** link or from the Request Body and Response Body panels while creating a REST Application.

From the **Document Types** page, you can add, edit, delete, or copy a document type.

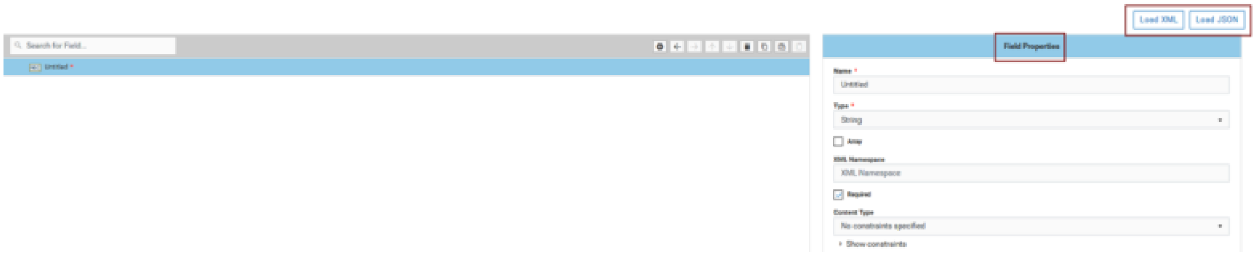
2. To edit an existing document type, select a document type from the **Document Types** page and click the **Edit** icon ✎. Select a field to view the **Field Properties** panel.

3. To create a new document type from scratch, from the **Document Types** page, click **Add New Document Type > Build from scratch**.

4. Provide a name and description of your document type. Required fields are marked with an asterisk on the page.

5. Click **Load XML** to generate a document type from the XML structure or click **Load JSON** to generate a document type from the JSON structure. Use the **Required** option ● Required to mark all the fields as optional. By default, all the fields are marked as mandatory. If a field is mandatory, then you must pass a value for that field when running an Integration. Mandatory fields are marked with an asterisk on the screen.

6. Click the + icon to add a new field. You can update the field properties by using the **Field Properties** window.



Name *
Untitled

Type *
String

□ Array

XML Namespace
XML Namespace

☑ Required

Content Type
No constraints specified

Provide the **Name** and **Type** of the field in order to define the structure and content of the document type. A field can be a String, Document, Document Reference, Object, Boolean, Double, Float, Integer, Long, or Short. In the **Field Properties** panel, if you select the **Type** as **Document Reference**, then in the **Document Reference** dialog box, all the shared projects and their document references appear along with the document references available in the current project. See "Sharing Assets across Projects" on page 98 for more information.

Fields are used to declare the expected content and structure of Integration signatures, document contents, and pipeline contents. In addition to specifying the name and type of a field, and whether the type is an **Array**, you can set properties that specify an **XML Namespace** and indicate whether the field is required at runtime by selecting the **Required** option. Select the **Content Type** you can apply to String, String list, or String table variables. See "About Variable Constraints" on page 642 for information.

You can copy a field from the fields panel by clicking the ⧉ icon. Depending on the context, you can either paste the field or the field path by clicking the 📋 icon. For example, if you copy a field and paste the field in the **Set Value** window in an Integration, the field path will be pasted. If you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index

[10]. But if it is pasted in the document tree, it will appear as an array, like A[ ]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B.

**Note:**
You cannot modify or paste the child fields of a Document Reference.

**Note:**
When defining a document type, avoid adding identically named fields to the document. In particular, do not add identically named fields that are of the same data type.

You can assign an **XML namespace** and prefix to a field by specifying a URI for the XML namespace property and by using the *prefix:fieldName* format for the field name. For example, suppose a field is named *eg:account* and the XML namespace property is set to http://www.example.com. The prefix is *eg*, the localname is *account*, and the namespace name is http://www.example.com.

Keep the following points in mind when assigning XML namespaces and prefixes to a field:

- The field name must be in the format: *prefix:fieldName*
- You must specify a URI in the XML namespace property.
- Do not use the same prefix for different namespaces in the same document type, input signature, or output signature.

7. Click **Apply** after you have entered the details and constraints for each field, and then click **Save** to save the **Document Type**.

**Note:**
When you edit a document type, any change is automatically propagated to all integrations that use or reference the document type.

The new document type appears in the **Document Types** page.

## Creating Document Types from an XML Schema Definition

Keep the following point in mind when creating a document type from an XML Schema Definition:

- You can specify whether Integration Cloud enforces strict, lax, or no content model compliance when generating the document type. Content models provide a formal description of the structure and allowed content for a complex type. The type of compliance that you specify can affect whether Integration Cloud generates a document type from a particular XML Schema definition successfully. Currently, Integration Cloud does not support repeating model groups, nested model groups, or the *any* attribute. If you select strict compliance, Integration Cloud does not generate a document type from any XML schema definition that contains those items.

**To create a document type from an XML Schema definition**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Document Types**. The **Document Types** page appears.

   From the **Document Types** page, you can add, edit, delete, or copy a document type.

2. To edit an existing document type, select a document type from the **Document Types** page and click the **Edit** icon ✏.

3. To create a new document type from an XML Schema Definition, from the **Document Types** page, click **Add New Document Type > Build from XML schema**.

4. Provide a name of your document type. Required fields are marked with an asterisk on the page.

5. On the **Specify the location of the XML schema file** panel, under **XML schema source**, do one of the following to specify the source file for the document type:

   - To use an XML Schema definition that resides on the Internet as the source, select **URL**. Then, type the URL of the resource. (The URL you specify must begin with `http:` or `https:`.)

   - To use an XML Schema definition that resides on your local file system as the source, select **File**. Then click **Browse** and select the file. The maximum file upload size is 5 MB.

6. Click **Next** and on the **Select schema-related processing options** panel, under **Content model compliance**, select one of the following to indicate how strictly Integration Cloud represents content models from the XML Schema definition in the resulting document type.

| Select... | To... |
|---|---|
| **Strict** | Generate the document type only if Integration Cloud can represent the content models defined in the XML Schema definition correctly. Document type generation fails if Integration Cloud cannot accurately represent the content models in the source XML Schema definition.<br>Currently, Integration Cloud does not support repeating model groups, nested model groups, or any attribute. If you select strict compliance, Integration Cloud does not generate a document type from any XML schema definition that contains those items. |
| **Lax** | When possible, generate a document type that correctly represents the content models for the complex types defined in the XML schema definition. If Integration Cloud cannot correctly represent the content model in the XML Schema definition in the resulting document type, Integration Cloud generates the document type using a compliance mode of None.<br>When you select lax compliance, Integration Cloud will generate the document type even if the content models in the XML schema definition cannot be represented correctly. |

| Select... | To... |
|---|---|
| **None** | Generate a document type that does not necessarily represent or maintain the content models in the source XML Schema definition. |

7. If you select strict or lax compliance, next to **Preserve text position**, do one of the following to specify whether document types generated will contain multiple **body* fields to preserve the location of text in instance documents.

   - Select the **Preserve text position** check box to indicate that the document type generated preserves the locations for text in instance documents. The resulting document type contains a **body* field after each field and includes a leading **body* field. In instance documents for this document type, Integration Cloud places text that appears after a field in the **body*.
   - Clear the **Preserve text position** check box to indicate that the document type generated does not preserve the locations for text in instance documents. The resulting document type contains a single **body* field at the top of the document type. In instance documents for this document type, text data around fields is all placed in the same **body* field.

8. If you want Integration Cloud to use the Xerces parser to validate the XML Schema definition, select the **Validate schema using Xerces** check box.

   **Note:**Integration Cloud automatically uses an internal schema parser to validate the XML Schema definition. However, the Xerces parser provides stricter validation than the internal schema parser. As a result, some schemas that the internal schema parser considers to be valid might be considered invalid by the Xerces parser.

9. Click **Next** and on the **Select root nodes** panel, under **Select root nodes**, select the elements that you want to use as the root elements for the document type. The resulting document type will contain all of the selected root elements as top-level fields in the generated document type. To select multiple elements, press the CTRL key while selecting elements.

10. Click **Finish**.

    Integration Cloud creates the document type.

    **Notes**

    - If you have selected strict compliance and Integration Cloud cannot represent the content model in the complex type accurately, Integration Cloud does not generate any document type.
    - If you have selected lax compliance and indicated that Integration Cloud should preserve text locations for content types that allow mixed content (you selected the **Preserve text position** check box), Integration Cloud adds *body fields in the document type only if the complex type allows mixed content and Integration Cloud can correctly represent the content model declared in the complex type definition. If Integration Cloud cannot represent the content model in a document type, Integration Cloud adds a single *body field to the document type.

- If the XML schema definition contains an element reference to an element declaration whose type is a named complex type definition (as opposed to an anonymous complex type definition), Integration Cloud creates a document type for the named complex type definition only if it is referred multiple times in the schema.
- Integration Cloud uses the prefixes declared in the XML Schema or the ones you specified as part of the field names. Field names have the format *prefix*:*elementName* or *prefix*:*@attributeName*.
- If the XML Schema does not use prefixes, Integration Cloud creates prefixes for each unique namespace and uses those prefixes in the field names. Integration Cloud uses “ns” as the prefix for the first namespace, “ns1” for the second namespace, “ns2”.
- If the XML Schema definition contains a user-specified namespace prefix and a default namespace declaration, both pointing to the same namespace URI, Integration Cloud uses the user-specified namespace prefix and not the default namespace.
- If the namespace prefix in the XML Schema as well as the default namespace point to the same namespace URI, Integration Cloud gives preference to the user-specified namespace prefix over the default namespace.

## About Variable Constraints

You apply content constraints to variables in the document types that you want to use as blueprints in data validation. Content constraints describe the data a variable can contain. At validation time, if the variable value does not conform to the content constraints applied to the variable, the validation engine considers the value to be invalid.

When applying content constraints to variables, do the following:

- **Select a content type** - A content type specifies the type of data for the variable value, such as string, integer, boolean, or date. A content type corresponds to a simple type definition in a schema.
- **Set constraining facets** - Constraining facets restrict the content type, which in turn, restrict the value of the variable to which the content type is applied. Each content type has a set of constraining facets. For example, you set a length restriction for a string content type, or a maximum value restriction for an integer content type.

For example, for a String variable named *itemQuantity*, specify a content type that requires the variable value to be an integer. You could then set constraining facets that limit the content of *itemQuantity* to a value between 1 and 100.

The content types and constraining facets described in this appendix correspond to the built-in data types and constraining facets in XML Schema. The World Wide Web Consortium (W3C) defines the built-in data types and constraining facets in the specification *XML Schema Part 2: Datatypes* (http://www.w3c.org/TR/xmlschema-2).

## Applying Constraints to a Variable

You apply content constraints to variables in the document types that you want to use as blueprints in data validation.

### To apply constraints to a variable

1. Select a document type from the **Document Types** page and click **Edit**. Select a field to view the **Field Properties** panel.

   You apply constraints to variables in document types declared on the Input/Output tab. If the selected variable is a String or String list and you want to specify content constraints for the variable, then do the following:

   - If you want to use a content type that corresponds to a built-in simple type in XML schema, in the **Content type** list, select the type for the variable contents. To apply the selected type to the variable, click **Save**.

2. Repeat this procedure for each variable to which you want to apply the constraints in the document type and click **Save**.

## Content Types

The following table identifies the content types you can apply to String or String list variables. Each of these content types corresponds to a built-in simple type defined in the specification *XML Schema Part 2: Datatypes*.

| Content Types | Description |
|---|---|
| **anyURI** | A Uniform Resource Identifier Reference. The value of anyURI may be absolute or relative.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern<br>**Note:**<br>The anyURI type indicates that the variable value plays the role of a URI and is defined like a URI. webMethods Integration Server does not validate URI references because it is impractical for applications to check the validity of a URI reference. |
| **base64Binary** | Base64-encoded binary data.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern |
| **boolean** | True or false. |

**Content Types** **Description**

**Constraining Facets**

pattern

**Example**

```
true, 1, false, 0
```

**byte**

A whole number whose value is greater than or equal to –128 but less than or equal to 127.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
-128, -26, 0, 15, 125
```

**date**

A calendar date from the Gregorian calendar. Values need to match the following pattern:

CCYY-MM-DD

Where CC represents the century, YY the year, MM the month, DD the day. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.

**Constraining Facets**

enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern

**Example**

`1997-08-09` (August 9, 1997)

**dateTime**

A specific instant of time (a date and time of day). Values need to match the following pattern:

CCYY-MM-DDThh:mm:ss.sss

Where CC represents the century, YY the year, MM the month, DD the day, T the date/time separator, hh the hour, mm the minutes, and ss the seconds. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.

**Constraining Facets**

| Content Types | Description |
|---|---|
| | enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`2000-06-29T17:30:00-05:00` represents 5:30 pm Eastern Standard time on June 29, 2000. (Eastern Standard Time is 5 hours behind Coordinated Universal Time.) |
| **decimal** | A number with an optional decimal point.<br>**Constraining Facets**<br>enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits<br>**Example**<br>`8.01, 290, -47.24` |
| **double** | Double-precision 64-bit floating point type.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`6.02E23, 3.14, -26, 1.25e-2` |
| **duration** | A length of time. Values need to match the following pattern:<br>P*n*Y*n*M*n*DT*n*H*n*M*nS*<br>Where *n*Y represents the number of years, *n*M the number of months, *n*D the number of days, T separates the date and time, *n*H the number of hours, *n*M the number of minutes and *nS* the number of seconds. Precede the duration with a minus (-) sign to indicate a negative duration.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`P2Y10M20DT5H50M` represents a duration of 2 years, 10 months, 20 days, 5 hours, and 50 minutes |

| Content Types | Description |
|---|---|
| **ENTITIES** | Sequence of whitespace-separated ENTITY values declared in the DTD. Represents the ENTITIES attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength |
| **ENTITY** | Name associated with an unparsed entity of the DTD. Represents the ENTITY attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern, whiteSpace |
| **float** | A number with a fractional part.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`8.01, 25, 6.02E23, -5.5` |
| **gDay** | A specific day that recurs every month. Values must match the following pattern:<br>---DD<br>Where DD represents the day. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`---24` indicates the 24th of each month |
| **gMonth** | A Gregorian month that occurs every year. Values must match the following pattern:<br>--MM<br>Where MM represents the month. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time. |

| Content Types | Description |
|---|---|
|  | **Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`--11` represents November |
| **gMonthDay** | A specific day and month that recurs every year in the Gregorian calendar. Values must match the following pattern:<br>--MM-DD<br>Where MM represents the month and DD represents the day. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`--09-24` represents September 24th |
| **gYear** | A specific year in the Gregorian calendar. Values must match the following pattern:<br>CCYY<br>Where CC represents the century, and YY the year. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`2001` indicates 2001 |
| **gYearMonth** | A specific month and year in the Gregorian calendar. Values must match the following pattern:<br>CCYY-MM |

| Content Types | Description |
|---|---|
| | Where CC represents the century, YY the year, and MM the month. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.<br>**Constraining Facets**<br>enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern<br>**Example**<br>`2001-04` indicates April 2001 |
| **hexBinary** | Hex-encoded binary data.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern |
| **ID** | A name that uniquely identifies an individual element in an instance document. The value for ID needs to be a valid XML name. The ID datatype represents the ID attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern, whiteSpace |
| **IDREF** | A reference to an element with a unique ID. The value of IDREF is the same as the ID value. The IDREF datatype represents the IDREF attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern, whiteSpace |
| **IDREFS** | Sequence of white space separated IDREFs used in an XML document. The IDREFS datatype represents the IDREFS attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength |
| **int** | A whole number with a value greater than or equal to -2147483647 but less than or equal to 2147483647.<br>**Constraining Facets**<br>enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits |

**Content Types** **Description**

**Example**

```
-21474836, -55500, 0, 33123, 4271974
```

**integer**

A positive or negative whole number.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
-2500, -5, 0, 15, 365
```

**language**

Language identifiers used to indicate the language in which the content is written. Natural language identifiers are defined in IETF RFC 1766.

**Constraining Facets**

enumeration, length, maxLength, minLength, pattern, whiteSpace

**long**

A whole number with a value greater than or equal to -9223372036854775808 but less than or equal to 9223372036854775807.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
-55600, -23, 0, 256, 3211569432
```

**Name**

XML names that match the Name production of XML 1.0 (Second Edition).

**Constraining Facets**

enumeration, length, maxLength, minLength, pattern, whiteSpace

**NCName**

Non-colonized XML names. Set of all strings that match the NCName production of Namespaces in XML.

**Constraining Facets**

enumeration, length, maxLength, minLength, pattern, whiteSpace

**negativeInteger**

An integer with a value less than or equal to –1.

**Constraining Facets**

| Content Types | Description |
|---|---|
| | enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits<br>**Example**<br>`-255556, -354, -3, -1` |
| **NMTOKEN** | Any mixture of name characters. Represents the NMTOKEN attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern, whiteSpace |
| **NMTOKENS** | Sequences of NMTOKENS. Represents the NMTOKENS attribute type from the XML 1.0 Recommendation.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength |
| **nonNegativeInteger** | An integer with a value greater than or equal to 0.<br>**Constraining Facets**<br>enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits<br>**Example**<br>`0, 15, 32123` |
| **nonPositiveInteger** | An integer with a value less than or equal to 0.<br>**Constraining Facets**<br>enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits, whiteSpace<br>**Example**<br>`-256453, -357, -1, 0` |
| **normalizedString** | Represents white space normalized strings. Set of strings (sequence of UCS characters) that do not contain the carriage return (#xD), line feed (#xA), or tab (#x9) characters.<br>**Constraining Facets**<br>enumeration, length, maxLength, minLength, pattern, whiteSpace<br>**Example**<br>`MAB-0907` |

**Content Types** **Description**

**positiveInteger**

An integer with a value greater than or equal to 1.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
1, 1500, 23000
```

**short**

A whole number with a value greater than or equal to -32768 but less than or equal to 32767.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
-32000, -543, 0, 456, 3265
```

**string**

Character strings in XML. A sequence of UCS characters (ISO 10646 and Unicode). By default, all white space is preserved for variables with a string content constraint.

**Constraining Facets**

enumeration, length, maxLength, minLength, pattern, whiteSpace

**Example**

```
MAB-0907
```

**time**

An instant of time that occurs every day. Values must match the following pattern:

hh:mm:ss.sss

Where hh indicates the hour, mm the minutes, and ss the seconds. The pattern can include a Z at the end to indicate Coordinated Universal Time or to indicate the difference between the time zone and coordinated universal time.

**Constraining Facets**

enumeration, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern

**Example**

**Content Types** **Description**

`18:10:00-05:00` (6:10 pm, Eastern Standard Time) Eastern Standard Time is 5 hours behind Coordinated Universal Time.

**token**

Represents tokenized strings. Set of strings that do not contain the carriage return (#xD), line feed (#xA), or tab (#x9) characters, leading or trailing spaces (#x20), or sequences of two or more spaces.

**Constraining Facets**

enumeration, length, maxLength, minLength, pattern, whiteSpace

**unsignedByte**

A whole number greater than or equal to 0, but less than or equal to 255.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
0, 112, 200
```

**unsignedInt**

A whole number greater than or equal to 0, but less than or equal to 4294967295.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
0, 22335, 123223333
```

**unsignedLong**

A whole number greater than or equal to 0, but less than or equal to 18446744073709551615.

**Constraining Facets**

enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits

**Example**

```
0, 2001, 3363124
```

**unsignedShort**

A whole number greater then or equal to 0, but less than or equal to 65535.

**Constraining Facets**

| Content Types | Description |
|---|---|
| | enumeration, fractionDigits, maxExclusive, maxInclusive, minExclusive, minInclusive, pattern, totalDigits<br>**Example**<br>`0, 1000, 65000` |

## Customizing a String Content Type

Instead of applying an existing content type or simple type to a String or String list, you can customize an existing type and apply the new, modified type to a variable. You customize a content type or simple type by changing the constraining facets applied to the type.

When you customize a type, you actually create a new content type. Designer saves the changes as a new content type. The constraining facets you can specify depend on the content type. Note that content types and constraining facets correspond to datatypes and constraining facets defined in XML Schema. For more information about constraining facets for a datatype, see the specification *XML Schema Part 2: Datatypes* (http://www.w3.org/TR/xmlschema-2/).

### To customize a content type

1. Select the variable to which you want to apply a customized content type.

2. In the **Constraints** category on the Properties view, click the **Content type** browse button and then do one of the following to select the content type you want to customize:

   - In the **Content type** list, select the content type you want to customize.
   - If you want to customize a simple type from an IS schema, click **Browse**. In the Browse dialog box, select the IS schema containing the simple type. Then, select the simple type you want to customize and apply to the variable. Click **OK**.

3. Click **Customize**. Designer makes the constraining facet fields below the **Content type** list available for data entry (that is, changes the background of the constraining facet fields from grey to white). Designer changes the name of the content type to `contentType`_customized.

4. In the fields below the **Content type** list, specify the constraining facet values you want to apply to the content type.

5. Click **OK**. Designer saves the changes as a new content type named `contentType`_customized.

   **Note:**
   The constraining facets displayed below the **Content type** list depend on the primitive type from which the simple type is derived. *Primitive types* are the basic data types from which all other data types are derived. For example, if the primitive type is string, Designer displays the constraining facets **enumeration**, **length**, **minLength**, **maxLength**, and **pattern**. For

more information about primitive types, refer to *XML Schema Part 2: Datatypes* at http://www.w3.org/TR/xmlschema-2/.

# 12 Reference Data

# Overview

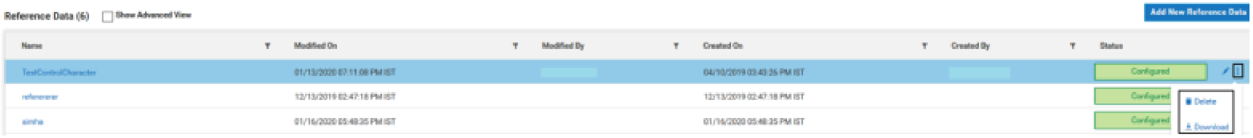
Reference data is data that defines the set of permissible values to be used by other data fields. It is a collection of *key-value pairs*, which can be used to determine the value of a data field based on the value of another data field. For example, the value of a status field in an Application can be “Canceled” and that needs to be interpreted as “CN” in another Application. If you have the required project permissions under **Settings** ⚙ **> Project Permissions**, you can create, update, or delete a Reference Data.

Integration Cloud allows you to upload Reference Data from a text file containing tabular data separated by a character, for example, a comma, semicolon, and so on. The uploaded file should not have an empty column heading or space in the first row, and the first row cannot be empty.

**Note:**
See this video on how to upload Reference Data, access the uploaded Reference Data in an Orchestrated Integration, and view the input and output parameters.

The Reference Data block appears under **Services** in the Orchestrated Integration workspace, only after you have created a Reference Data. See Reference Data Signature for information on the Input and Output parameters. The Reference Data is also available in Point-to-Point Integrations while transforming data. You can access the uploaded Reference Data in Orchestrated Integrations as a list of documents by using the *Reference Data block* and providing an appropriate name. You can filter the documents returned into the pipeline by the Reference Data block.

You can create a Reference Data only in the **Development** stage but can view, edit, delete, and download the Reference Data *in all stages*. The **Status** column in the Reference Data table displays **Configured** if the Reference Data is available in the current stage (Stage in view) and displays **Not Configured** if the Reference Data is not available in the current stage but available in any other stage. You can **Delete** or **Download** a Reference Data if the Reference Data is available in the current stage. You can **Edit** a Reference Data in all stages. The **Download** option allows you to download the previously uploaded Reference Data, edit it, and then upload the modified file. In this way you can upload different sets of data in different stages.



Reference Data (6) □ Show Advanced View

| Name | Modified On | Modified By | Created On | Created By | Status |
|---|---|---|---|---|---|
| TestControlCharacter | 01/13/2020 07:11:08 PM IST | | 04/10/2019 03:43:26 PM IST | | Configured |
| refererer | 12/13/2019 02:47:18 PM IST | | 12/13/2019 02:47:18 PM IST | | Configured |
| simha | 01/16/2020 05:48:35 PM IST | | 01/16/2020 05:48:35 PM IST | | Configured |

You can pull an Integration only if the Integration is consistent in the source stage, that is, all references of the Integration are present in the source stage. While pulling an Integration from the source stage to the target stage, if the same Reference Data is not available in the target stage, then you must configure the Reference Data in the target stage. If the same Reference Data is available in the target stage with a signature mismatch, for example, the number of columns or the column names are not the same, then you can either reconfigure the Reference Data or skip the reconfiguration. If the Reference Data is already available in the target stage and the signature is same as in the source stage, then it will not be copied from the source stage to the target stage while pulling the Integration.

**Note:**

If a Reference Data is in the Development stage, the Reference Data can be deleted even it is referenced or used in an Integration. Integrations using the deleted Reference Data will be in an inconsistent state.

If a Reference Data is in a stage other than the Development stage, the Reference Data can be deleted only if it is configured in that stage and not used in any Integration.

## Add Reference Data

**To add or edit a Reference Data**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Reference Data**. The **Reference Data** page appears.

2. To edit an existing Reference Data, select a configured Reference Data from the **Reference Data** page and click the **Edit** icon ✎.

3. To create a new Reference Data, from the **Reference Data** page, click **Add New Reference Data**. You can create a Reference Data only in the **Development** stage.

   The **Upload Reference Data** page appears.

4. Provide a name and description of your Reference Data. Required fields are marked with an asterisk on the page.

5. For **Reference Data File**, click **Browse** and select the file. Only a text file having tabular data is supported. The maximum file size you can upload is 1 MB. Further, the file should not have an empty column heading or space in the first row and the first row cannot be empty. This is because the first row of data is read as column headings.

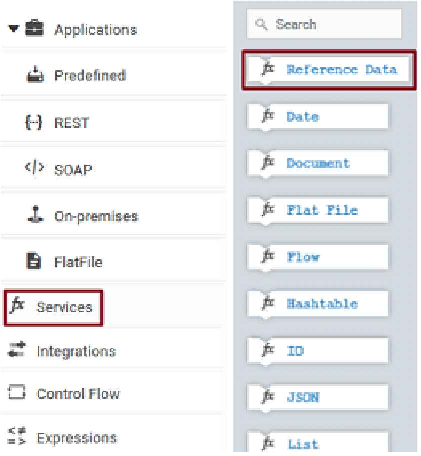
6. Click **Next** to define and preview the Reference Data. Select the field separator and the text qualifier.

7. Determine the encoding of the Reference Data file and from the **File Encoding** drop down list, select the same encoding. Click **Load Preview** to preview the data. If you select an incorrect encoding, garbage characters will appear in the preview pane.

8. Click **Next** to review the Reference Data and then click **Finish** to create the Reference Data.

The new Reference Data appears in the **Reference Data** page with the status as **Configured**.

**Note:**
The **Reference Data** block appears under **Services** only after you have created a Reference Data and the Reference Data service will be available while creating an Orchestrated Integration.



The Reference Data is also available in Point-to-Point Integrations while transforming data.

## Reference Data Signature

Reference Data signature is derived from the column names of the uploaded text file. You can filter the Reference data by providing an appropriate **matchCriteria**. The output of Reference Data is a list of documents that match the specified **matchCriteria**.

**Note:**
The root element in the output of Reference Data created from version 2.1.0 has the same name as the Reference Data.

### Input Parameters

*matchCriteria* **Document** Criteria on which documents from the Reference Data will be matched.

Parameters for `matchCriteria` are:

**path**: Column names of the Reference Data.

**compareValueAs**: Optional. Allowed values are string, numeric, and datetime. The default value is string.

**datePattern**: Optional. Pattern will be considered only if `compareValueAs` is of type datetime. Default value is MM/dd/yyyy hh:mm:ss a.

**joins**: List of join criteria.

Each join criteria consists of:

`operator`: Allowed values are equals, doesNotEqual, greaterThan, greaterThanEqual, lessThan, lessThanEqual, equalsIgnoreCase, contains, doesNotContain, beginsWith, doesNotBeginWith, endsWith, doesNotEndWith.

`value`: Optional. Allowed values are string, numeric, and datetime. The default value is string.

`joinType`: Specifies the way two joins can be linked. Values are “and” or “or”. Default value is “and”.

## Output Parameters

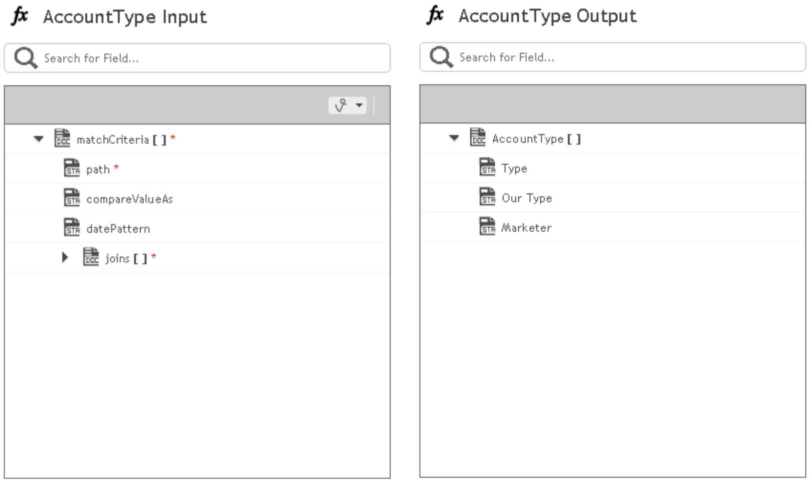
*<Reference Data Name>* **Document List** List of documents that match the retrieve criteria.

In the following example, the flat file contains “Type”, Our Type”, and Marketer” as headers and has one or more data rows.

Type,Our Type,Marketer

Existing - Growth,Growth,HUNT & SONS INC

The following graphic illustrates the generated Reference Data signature:



# 13 Monitor

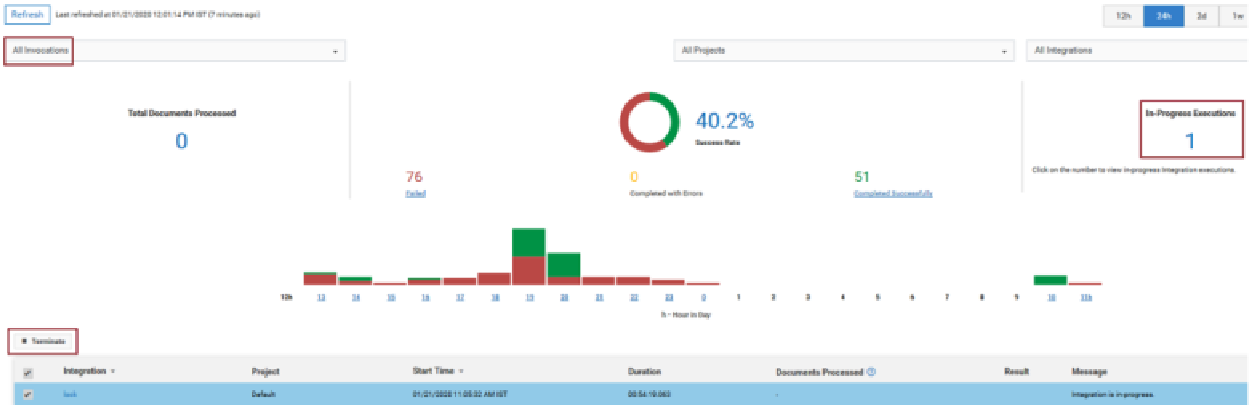
# Overview

The **Dashboard** allows you to view and monitor Integration executions and performance details. The **Execution Results** page allows you to view the audit trail of all the executions that happened in a stage for an Integration or for all Integrations for a project or for all projects. You can also create **Alert Rules** and send email messages to selected users for one or more Integrations based on Integration execution results. The **Reports** page provides you information about the number of times an Integration is run for the specified time frame. The **Clear Storage Locks** page provides you information on how to clear locks on integrations.

# Dashboard

The **Dashboard** provides a centralized and intuitive way to view and monitor Integration executions and performance details. To view the **Dashboard**, click **Monitor > Dashboard**.

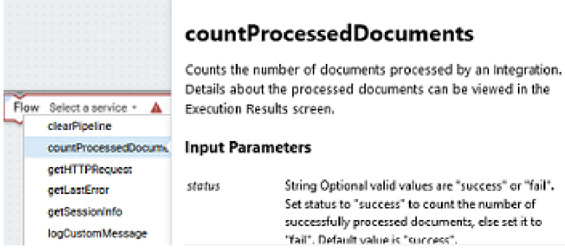
You can identify and diagnose problems for those Integrations that are available in the active stage. To select another active stage, click **Stages > Change Stage To View**. You can view the **Dashboard** if the access profile assigned to you is also specified for that stage in the **Manage** Stages page. Further, you must be able to access the stages to view the dashboard.



| ☑ | Integration | Project | Start Time | Duration | Documents Processed | Result | Message |
|---|---|---|---|---|---|---|---|
| ☑ | task | Default | 01/21/2020 11:05:32 AM IST | 00:54:19.063 | - | | Integration is in-progress. |

The **Dashboard** displays the following details:

- When was the Dashboard last refreshed.
- Options to select the Invocation channel, Project, and Integration to view the execution details.
- Total number of documents processed by an Integration or for all Integrations in the active stage. Documents processed appear only if the Integration invokes the **countProcessedDocuments** service to count the number of documents processed by the Integration. See the **countProcessedDocuments** service available in the **Flow** block under the **Services** category for more information.



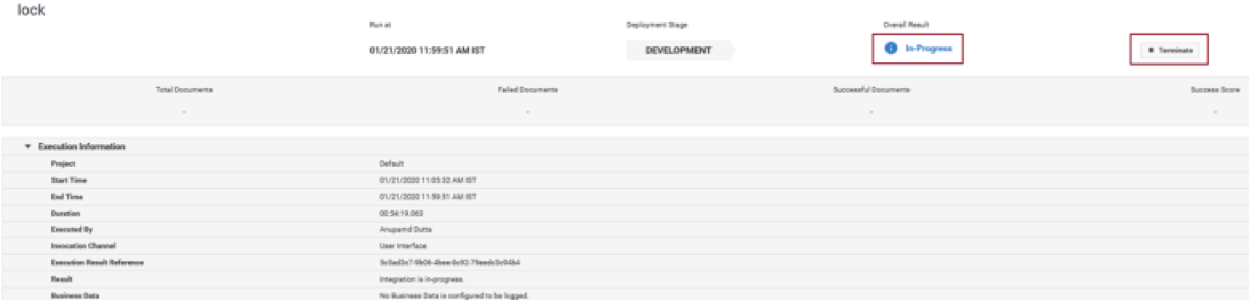
**countProcessedDocuments**

Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen.

**Input Parameters**

| | |
|---|---|
| *status* | String Optional valid values are "success" or "fail". Set status to "success" to count the number of successfully processed documents, else set it to "fail". Default value is "success". |

- Number of completed and failed Integration executions that happened during the selected time period in the active stage.
- Number of Integration executions that have completed with errors during the selected time period in the active stage.
- Completed Integration executions, failed Integration executions, and Integration executions that completed with errors displayed in a pie chart, along with the **Success Rate**, that is, the percentage of completed Integration executions compared to the total Integration executions, during the selected time period in the active stage.
- Number of in-progress Integration executions in the active stage. You can click on the **In-Progress Executions** number to view the in-progress Integration execution details in a table. You can terminate in-progress Integration executions from the in-progress Integration execution details table as well as from the Execution Results details page. The *Terminate* audit log entry is created. You can terminate an in-progress Integration execution if you have the *Execute* Integration permission. To terminate an in-progress execution from the Execution Results details page, click the **Terminate** option available in the Execution Results details page.



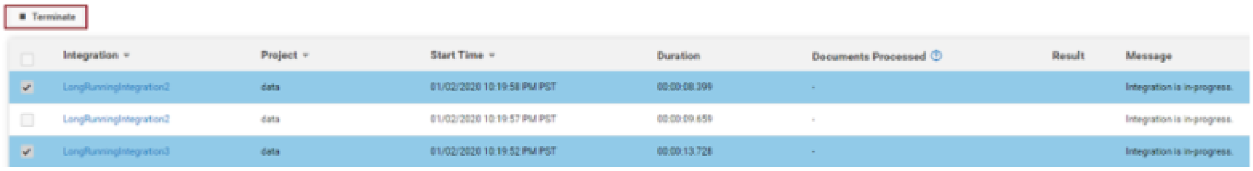
| Total Documents | Failed Documents | Successful Documents | Success Score |
|---|---|---|---|
| - | - | - | - |

▾ Execution Information

| | |
|---|---|
| Project | Default |
| Start Time | 01/21/2020 11:05:32 AM IST |
| End Time | 01/21/2020 11:59:51 AM IST |
| Duration | 00:54:19.063 |
| Executed By | Anupamd Dutta |
| Invocation Channel | User Interface |
| Execution Result Reference | 5c5ad3c7-9b06-4bee-8c92-79eedc0c94b4 |
| Result | Integration is in-progress. |
| Business Data | No Business Data is configured to be logged. |

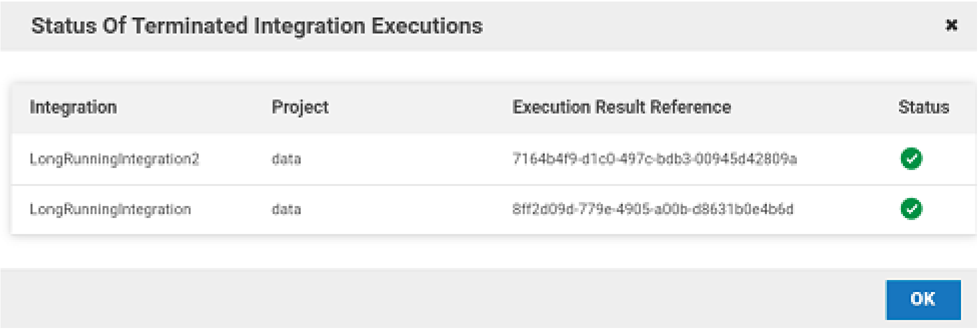
**Note:**
The **Terminate** option is available only if the current logged in user has the Execute Integration permission and the Integration status is in-progress.

To terminate in-progress Integration executions from the in-progress Integration execution details table, select the in-progress integrations you want to terminate and then click the **Terminate** option available above the in-progress Integration execution details table.



| □ | Integration ▾ | Project ▾ | Start Time ▾ | Duration | Documents Processed ⓘ | Result | Message |
|---|---|---|---|---|---|---|---|
| ☑ | LongRunningIntegration2 | data | 01/02/2020 10:19:58 PM PST | 00:00:08.399 | - | | Integration is in-progress. |
| □ | LongRunningIntegration2 | data | 01/02/2020 10:19:57 PM PST | 00:00:09.659 | - | | Integration is in-progress. |
| ☑ | LongRunningIntegration3 | data | 01/02/2020 10:19:52 PM PST | 00:00:13.728 | - | | Integration is in-progress. |

The status of the terminated in-progress integration executions appears.



Status Of Terminated Integration Executions

| Integration | Project | Execution Result Reference | Status |
|---|---|---|---|
| LongRunningIntegration2 | data | 7164b4f9-d1c0-497c-bdb3-00945d42809a | ✓ |
| LongRunningIntegration | data | 8ff2d09d-779e-4905-a00b-d8631b0e4b6d | ✓ |

**Note:**
The **Terminate** option is available only if the current logged in user has the Execute Integration permission and the Integration status is in-progress.

- Restart or resume an Integration execution.
- Successful Integration executions, failed Integration executions, and Integration executions that completed with errors displayed in a bar chart along with clickable links, for the selected time period in the active stage. You can click the Integration execution links available above the bar charts to display the relevant Integration execution details in the table. You can also point to each bar in the chart to view the date and time when the Integration executed and the result of the Integration execution.
- Name of the Integration, stage name, when the Integration started, the Integration run duration, documents processed details, result of the Integration execution (Completed Successfully, Failed, Completed with errors), and the Integration execution message displayed in a tabular format. The **Documents Processed** column displays the total number of documents processed by an integration, the number of documents processed successfully, and the success percentage. Values in this column appear only if the Integration invokes the **countProcessedDocuments** service to count the number of documents processed by the Integration. For more information, see the **countProcessedDocuments** service available in the **Flow** block under the **Services** category.
- If you click a row on the table, you can view the execution information as well as the operations details for the Integration . See Execution Results for more information.

## Execution Results

The **Execution Results** page allows you to view the audit trail of all the Integration executions that happened in the current stage, during a specified time period, for a selected project or for all projects. You can also restart or resume an Integration execution, specify the number of days to retain the entries, and download the entries.

You can also restart or resume an Integration execution from the **Monitor > Dashboard** page.

The following table provides information on when you can restart or resume an Integration execution:

| Execution Result Status | Restartable | Resumable |
|---|---|---|
| Completed Successfully | Yes | No |
| Completed with Errors | Yes | No |
| Failed | Yes | Yes |
| In-Progress | No | No |

**Note:**
To view execution results, ensure that the Access Profile of the user is assigned to the current stage.

### To view the execution results

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Integrations**.

   The **Integrations** page appears.

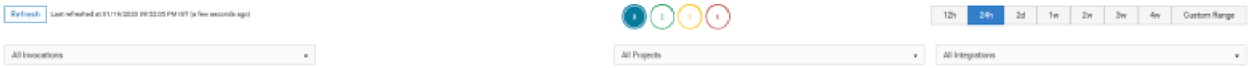
2. From the **Integrations** page, select the Integration for which you want to view the execution results.

3. Click the *Integration link* to view the Integration **Overview** page. You can click **Edit** to modify the Integration, click **Delete** to delete the Integration from this page, or click **Run Now** to execute the Integration. You can also see the last five execution results in the **Last 5 Execution Results** tab.

   You can also access the **Execution Results** by clicking **Monitor > Execution Results**.

4. On the **Execution Results** page, select the **Invocation** channel, a project, an Integration, and the time period for which you want to view the execution results. Select **All Invocations**, **All Projects**, and **All Integrations** if you want to view the execution details of Integrations based on all the invocation channels, for all integrations, and in all projects in the active stage, for the specified time period. The **Custom Range** option allows you to set a time period to view the results. The default time period is for the last 24 hours (24h).

   Execution results are displayed in a tabular form. You can filter the results in the table by clicking on the status filter circles available on the top-right corner above the table. The numbers inside the status circles indicate the sum of the execution counts for that status.



- **All** - All operations of an Integration, which have **Completed Successfully**, **Failed**, and **Completed with errors** are displayed.
- **Completed Successfully** - All operations of an Integration that completed successfully while executing are displayed.

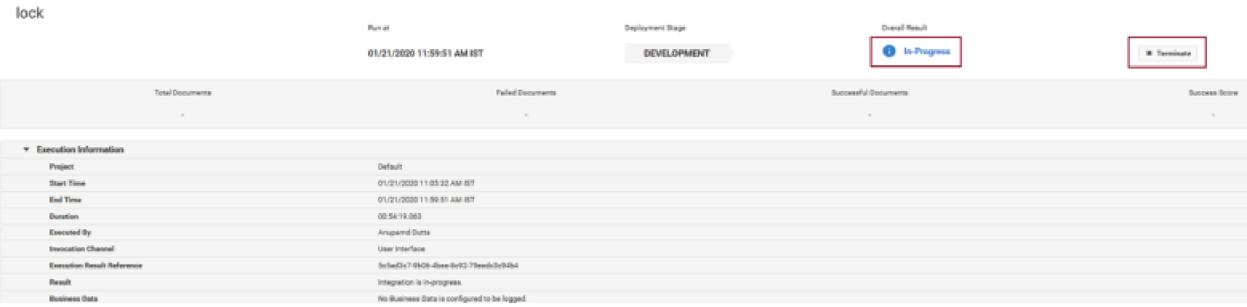
- **Failed** - Exceptions occurred while executing an operation in an Integration.
- **Completed with errors** - Exceptions occurred while executing an operation in an Integration and caught by the try-catch block in an orchestrated Integration.

5. In the **Execution Results** page, click **Download Results** to download the execution results, or click **Modify Retention Period** and specify the number of days to retain the execution result entries. You can retain entries up to 30 days. Entries whose age exceeds the specified retention period are deleted. Default value of the Retention Period is 30 days.

   > **Note:**
   > User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.

6. On the **Execution Results** page, click an Integration in the table to view more information about the selected Integration execution. The **Execution Details** page appears.

   In the **Execution Details** page, the **Documents** row displays the total number of documents processed by the integration, the number of documents processed successfully, the number of documents that did not process successfully, and the success percentage. Values in this row appear only if the Integration invokes the **countProcessedDocuments** service to count the number of documents processed by the Integration. See the **countProcessedDocuments** service in the **Flow** block under the **Services** category for more information. You can also terminate in-progress Integration executions from the Execution Results details page. The **Terminate** option is available only if you have the *Execute* Integration permission and the Integration status is in-progress. The **Execution Information** section displays when the Integration execution was started, when it ended, the duration of the execution, who executed the Integration, through which channel the Integration was executed, for example, Scheduler, User Interface, and REST Interface, and so on, the execution result reference, that is, the Integration execution result reference identifier, and the business data details.



lock

| Run at | Deployment Stage | Overall Result |
|---|---|---|
| 01/21/2020 11:59:51 AM IST | DEVELOPMENT | In-Progress |

| Total Documents | Failed Documents | Successful Documents | Success Score |
|---|---|---|---|
| - | - | - | - |

▾ Execution Information

| | |
|---|---|
| Project | Default |
| Start Time | 01/21/2020 11:05:32 AM IST |
| End Time | 01/21/2020 11:59:51 AM IST |
| Duration | 00:54:19.063 |
| Executed By | Anupamd Dutta |
| Invocation Channel | User Interface |
| Execution Result Reference | 5c5ad3c7-9b06-4bee-8c92-79eedc3c94b4 |
| Result | Integration is in-progress. |
| Business Data | No Business Data is configured to be logged. |

7. The **Execution Results** page also provides information about operations for the selected Integration. Click **Show Everything** to view all information about the operation execution including business data and custom messages. Click **Only Business Data** to view only the logged business data information. Click **Only Custom Messages** to view only custom messages. You can filter the results in the table by clicking on the status filter circles on the top-right corner of the Operations table. Click on the **All** (blue) circle to view operation information, business data, and custom messages. Click on the **Successful** (green) circle to view only successful operation information and business data. Click on the **Failed** (red) circle to view

only failed operation information and business data. Custom messages appear only if you have set up log messages. See the **logCustomMessage** service in the **Flow** block under the **Services** category for more information on how to set up custom messages in an Integration.

8. In the **Execution Results** page, select a row and click **Restart** to edit the input data and restart the Integration execution from the beginning, even though the previous execution has been successful. When an Integration is restarted, the Audit Log entry displays "Restart".

   Click **Resume** to edit the input data for failed operations and execute the failed and not yet executed operations. When an Integration is resumed, the Audit Log entry displays "Resume".

   > **Note:**
   > You can also restart or resume an Integration execution from the **Monitor > Dashboard** page.

   The "Restart/Resume" capability is available only if you have the required license for restarting and resuming Integrations. You must also have the Integration execution (Execute) permission if you want to restart or resume an execution.

   You must select the **Enable executions to be restartable** option in the **Integration Details** page in order to enable Integration executions to be restartable or resumable. See Integration Details for more information.

## Alert Rules

Integration Cloud allows you to create alert rules and send email messages to selected users for one or more Integrations, including the Integrations used in REST APIs and SOAP APIs, based on the execution results (Failed, Completed with Errors, or Completed Successfully) for the current stage. All executions that have occurred within the specified time period and have alert rules configured are sent as email messages to specified users.

> **Note:**
> Email messages are sent only if there are executions that match the alert rules.

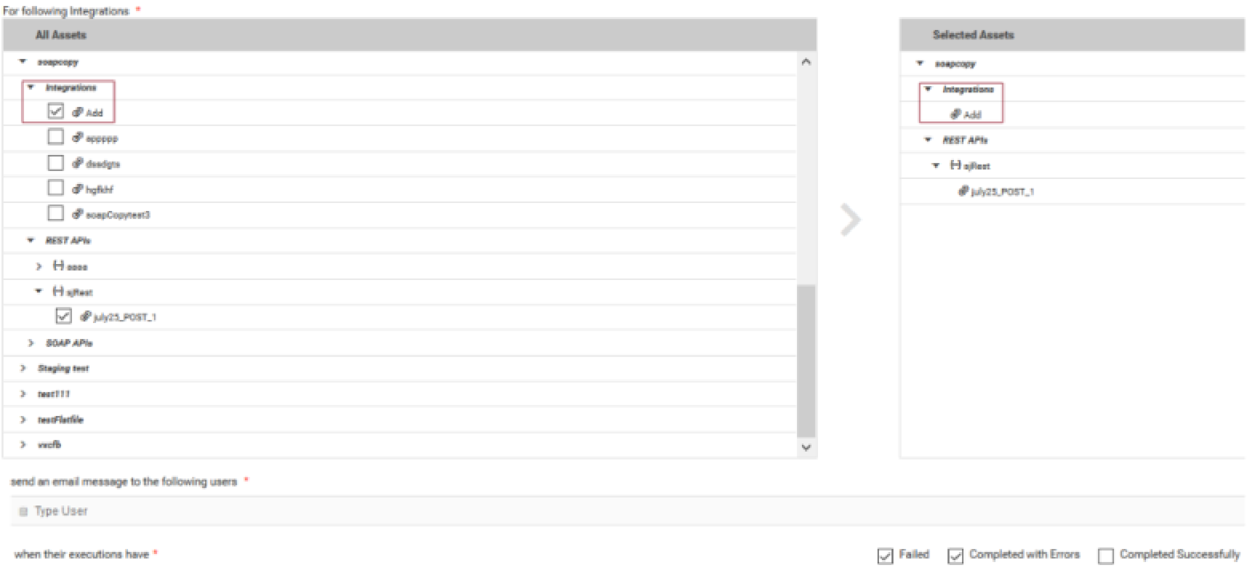
### To create a new alert rule

1. From the Integration Cloud navigation bar, click **Monitor > Alert Rules**. The **Alert Rules** page appears. You can edit, delete, activate, or deactivate an existing alert rule from this page.

2. From the **Alert Rules** page, click **Add New Alert Rule** to create a new alert rule.

   The **Add New Alert Rule** page appears.

3. On the **Add New Alert Rule** page, provide a name and description for the alert rule and complete the following fields. Required fields are marked with an asterisk on the page.



For following Integrations *

send an email message to the following users *

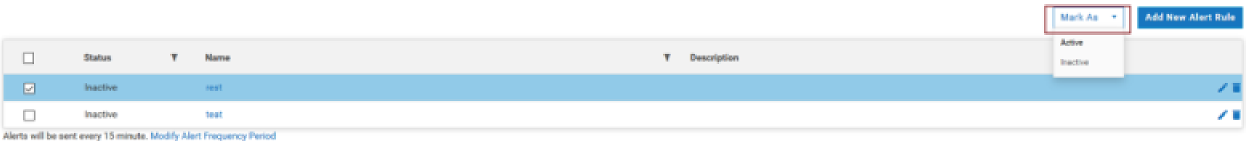
Type User

when their executions have * ☑ Failed ☑ Completed with Errors □ Completed Successfully

| Field | Description |
|---|---|
| **For following Integrations** | On the **All Assets** panel, select the Integrations, including the Integrations used in REST APIs and SOAP APIs that you want this alert rule to apply. The selected Integrations, including the Integrations used in REST APIs and SOAP APIs will appear in the **Selected Assets** panel. |
| **when their executions have** | Select the Integration execution results.<br>■ **Failed** - Exceptions occurred while executing an operation in an Integration.<br>■ **Completed with Errors** - Exceptions occurred while executing an operation in an Integration and caught by the try-catch block in an orchestrated Integration.<br>■ **Completed Successfully** - All operations of an Integration that completed successfully while executing. |
| **send an email message to the following users** | Click the ⊕ icon and select the active users to whom you want to send email alerts. Alerts will be sent to the email address specified in the **Basic** tab of the user's profile (**Settings** ⚙ **> Users > User Profile > Basic > Email**). |

4. Click **Save and Activate** to save and enable the new alert rule.

The new alert rule appears in the **Alert Rules** page. You can disable an alert rule by selecting the alert rule and clicking **Mark As > Inactive**. You can also enable the alert rule by selecting the alert rule and clicking **Mark As > Active**.



| ☐ | Status | Name | Description |
|---|---|---|---|
| ☑ | Inactive | test | |
| ☐ | Inactive | test | |

Alerts will be sent every 15 minute. Modify Alert Frequency Period

5. On the **Alert Rules** page, click **Modify Alert Frequency Period** to specify the alert frequency period to send email messages. If you do not want to receive email alerts for Integration executions, open the alert rule for editing, and remove your username from the user's list.

## Reports

The Reports page provides you information about the number of times an Integration is run for the specified time frame. The report shows the count for the currently active stage and the reports data is collected daily, at the end of day.

By default, the data shown on the Reports page is for all the Integrations run across all projects. When the reports data is collected initially, it is compiled for the number of days as specified in the retention period on the Execution Results page. Default value of the retention period is 30 days.

A zero count may mean that either the integration is not executed for that month, or the integration execution had occurred earlier than the value specified in the retention period. This is applicable only during the initial period.

**Note:**
The data you see on the Reports page depends mainly on two factors:

- When you view the data
- When data collection ends

The Reports page displays data collected till the day before yesterday. Daily around 9:10 PM, data collection starts. The time when data collection ends depends on the volume of data and for how long Integration Cloud takes to collect the data.

**Example**

Let us assume that the current date and time is *18th October*, 9 PM. If you view the Reports page now at 9 PM, data collection is yet to start for today (it will start at 9:10 PM), and so the Reports page will show the integration execution counts that have occurred till *16th October* 11:59 PM. If the data collection starts at 9:10 PM today and ends by 11 PM, then after 11 PM, the Reports page will show the integration execution counts till *17th October* 11:59 PM.

You can know when the data collection has completed from the entries in the Audit Log:

- *Type*: IntegrationResults History
- *Operation*: Update

### To view the reports

1. From the Integration Cloud navigation bar, click **Monitor > Reports**. The **Reports** page appears.

2. Select the time frame for which you want to view the report. You can set from the following options:

   - **1m** - Show data for 1 month from the current system date.
   - **3m** - Show data for 3 months from the current system date.
   - **6m** - Show data for 6 months from the current system date.
   - **12m** - Show data for 12 months from the current system date.
   - **Custom Range** - In the **Set Custom Range** dialog box, set the **Start date** and **End date** for the time range.

3. To download the report, click **Download Reports**. Currently, you can download reports in the following formats:

   **Note:**
   The downloaded report also shows the stage for which you have run the report.

   - PDF
   - CSV
   - PDF (detailed) - In addition to the details provided in PDF format, here you can view how many times in each month an Integration has run.
   - CSV (detailed) - In addition to the details provided in CSV format, here you can view how many times in each month an Integration has run.

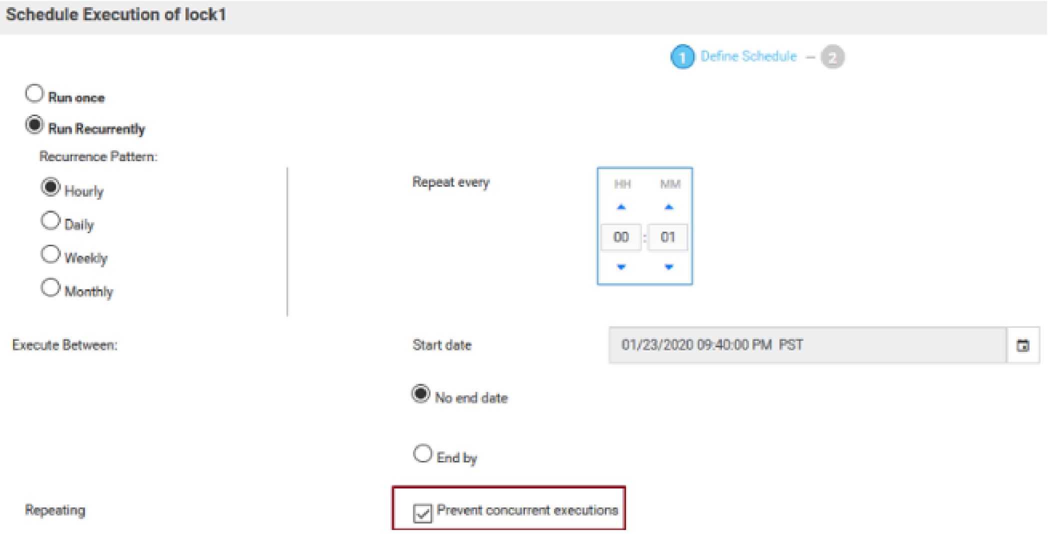
## Clear Storage Locks

If the environment goes down while an integration is running, the lock taken on the integration will not be automatically removed immediately. So any scheduled executions for the same integration will be skipped.

The following scenarios describe the actions you need to take to clear the locks.

**Scenario 1**

While scheduling an integration, if you have selected the **Prevent concurrent executions** option, then before each execution, a lock will be taken on the integration. If the Integration Server goes down when the scheduled integration is in progress, then the lock will not be automatically removed. This will cause any further scheduled executions for the same integration to be skipped.



Schedule Execution of lock1

○ Run once

● Run Recurrently

Recurrence Pattern:

● Hourly

○ Daily

○ Weekly

○ Monthly

Repeat every HH MM 00 : 01

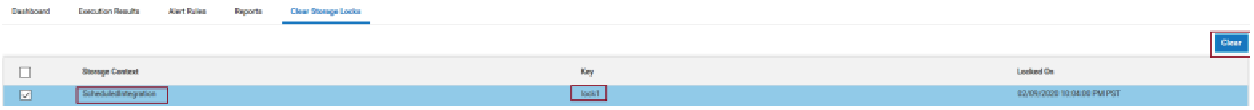
Execute Between: Start date 01/23/2020 09:40:00 PM PST

● No end date

○ End by

Repeating ☑ Prevent concurrent executions

In this case, go to **Monitor > Clear Storage Locks** and clear the lock on the integration. The *Integration Name* will appear in the **Key** column and *ScheduledIntegration* will appear under **Storage Context** in the **Clear Storage Locks** page.

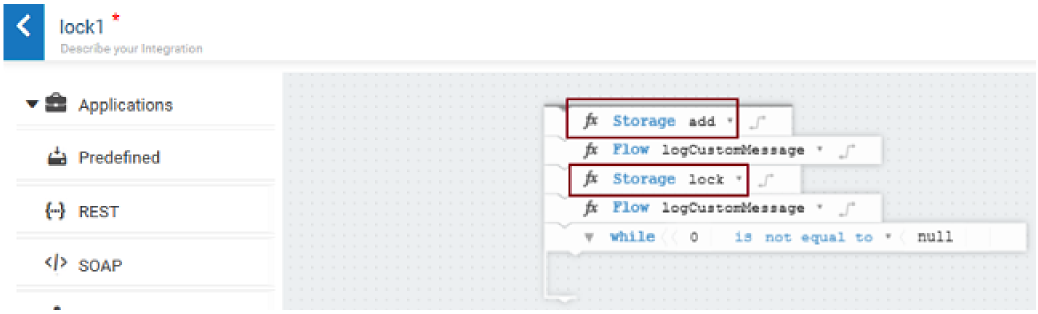


| □ | Storage Context | Key | Locked On |
|---|---|---|---|
| ☑ | ScheduledIntegration | lock1 | 02/09/2020 10:04:00 PM PST |

The next scheduled integration will be invoked successfully. If you do not clear the lock manually, Integration Cloud will automatically remove the integration locks periodically.

### Scenario 2

If you have added the Storage add and lock services in an integration, you have taken a lock on the entry. This lock will get automatically unlocked when the integration execution completes.



```
fx Storage add
fx Flow logCustomMessage
fx Storage lock
fx Flow logCustomMessage
▾ while 0 is not equal to null
```

### Scenario 3

If you have added the Storage add and lock services in an integration, you have taken a lock on the entry. After taking the lock and before the execution has completed, if Integration Server goes down, and then if you want to re-run the Integration, the integration execution will stop at the Storage lock step as the earlier lock is not removed. Then if you want to clear the lock, go to the **Clear Storage Locks** page and clear the lock on the storage entry. The current integration execution will immediately start running after the lock is cleared. If you do not clear the lock manually, Integration Cloud will automatically remove the integration locks periodically.

### Scenario 4

You have an integration that may run for a few hours, say for example, more than three hours, and the integration has taken a lock either by using the Storage add and lock services or by using the **Prevent concurrent executions** option in the scheduler. Now even if the integration is running, Integration Cloud will automatically remove the integration lock after the periodic schedule for removing locks. To prevent this scenario, contact Support.

# 14 REST APIs

## Overview

Integration Cloud allows you to write integration logic to integrate different types of applications. This logic can be exposed to the external world using REST APIs.

These REST APIs can be created by using an existing set of Integrations (from scratch) or by using a file containing the Open API specification (formerly known as the Swagger specification) as a template.

A REST API consists of many Resource Operations and each Resource Operation has a Path, one or more HTTP Methods, and an associated Integration.

A REST Resource Operation can be tried out from the **Swagger** screen of a REST API. When the Resource Operation is invoked using the HTTP Method, the associated Integration gets executed.

**Note:**
Users who have the required project permissions under **Settings** ⚙ **> Project Permissions** can create, update, delete, and execute REST APIs.

**Note:**
If you have created a REST API by using a file containing the Open API specification (formerly known as the Swagger specification) as a template (**Build with Swagger** approach), and have now uploaded a new file, Integrations and Document Types that are created are now based on the new Swagger file.

## Creating REST APIs from scratch

### To create a REST API from scratch

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**.

   The **REST APIs** page appears.

2. From the **REST APIs** page, click **Add New REST API**, select **Build from scratch**, and then click **OK**.

3. Complete the following fields and click **Save**.

   The new REST API appears in the **REST APIs** page.

| In this field... | Specify... |
|---|---|
| **Save As** | Type a name for the REST resource using any combination of letters, numbers, and/or underscore characters. |
| **Swagger Title** | Provide a title for the REST API. |

| In this field... | Specify... |
|---|---|
| **Description** | Type an optional description for the REST API. |
| **Version** | A version number. The value is typically 1.0. You need to change the version number if you want to modify and republish the REST API. |
| **Consumes** | Select the MIME types that the API consumes. |
| **Produces** | Select the MIME types that the API produces. The MIME types you select here will be part of the Swagger definition and will appear in the **Responses** section of the Swagger editor. While executing the REST API, you can choose the format in which you want to view the response.<br>You must specify an *Accept* header in the REST client, else the default response will conform to the Content Type *text/html*. |

4. Click **Save**.

   The REST API is created and appears in the **REST APIs** page. When you create a REST API, Integration Cloud uses the general information that you supplied to populate the **Overview** page in the REST API. You can click **Edit** to change the information in the **Overview** page. The name cannot be modified.

5. In the **Resources** page, click **Add New Resource** to add a Resource to the REST API. Type a **Path**, select a **Mapped Integration** that you want to invoke with this path, and select the **HTTP Methods** that can be used to invoke this Integration with this path. You can also provide a description for each method in the **Operation Summary** field. You can add more Resource Operations, if needed. You can modify a REST resource even if the resource has an operation with a deleted integration mapped to the operation.

6. Click **Save and Continue** to add parameters and responses and then click **Save and Finish**. See “Modifying Resource Operations” on page 676 for information on how to modify a REST Resource Operation.

7. To run an Integration with a certain path and method, go to the **REST APIs** page, select a REST API, and click the REST API link. The REST API details page appears.

8. Go to the **Swagger** page from the REST API details page. Click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs. Select the path and method pair and click **Try it out**. If required, pass the parameters and click **Execute**.

   Responses will be displayed on the pane.

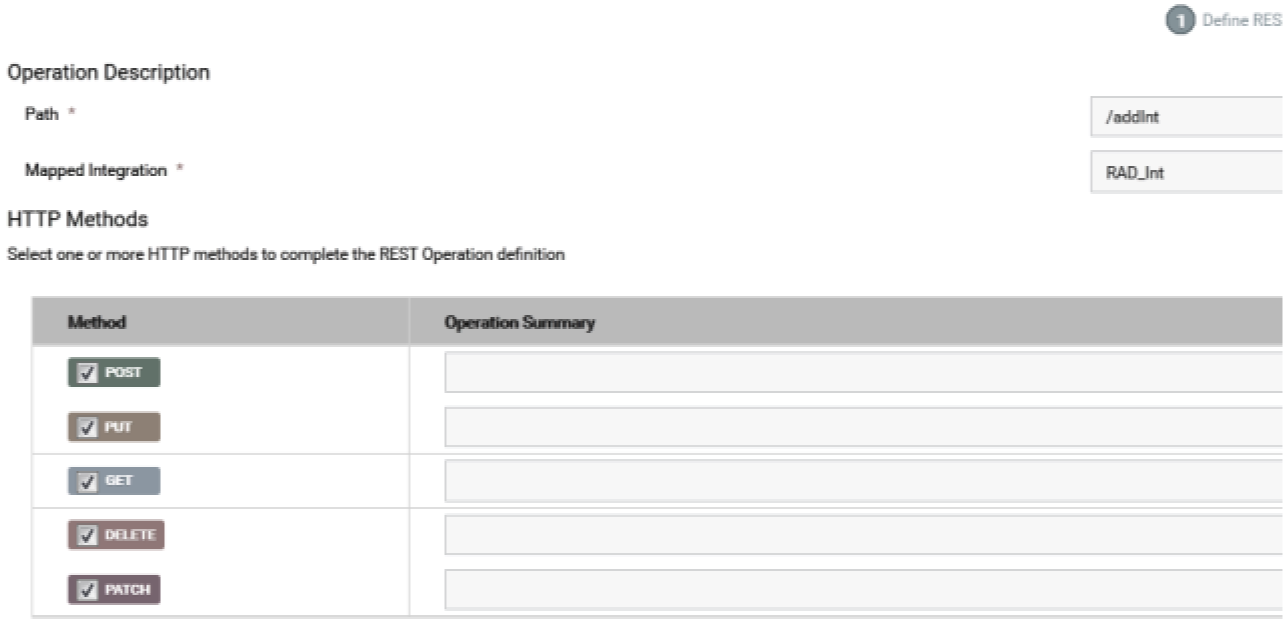
# Modifying Resource Operations

### To modify a REST Resource Operation

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**.

   The **REST APIs** page appears.

2. Click on the link of a REST API that has been *created from scratch*, go to the **Resources** page, select a resource operation, and then click the **Edit** icon.



Operation Description

Path * /addInt

Mapped Integration * RAD_Int

HTTP Methods

Select one or more HTTP methods to complete the REST Operation definition

| Method | Operation Summary |
|---|---|
| ☑ POST | |
| ☑ PUT | |
| ☑ GET | |
| ☑ DELETE | |
| ☑ PATCH | |

**Modifying the Path**

On the **Resources** page, select a resource operation and then click the **Edit** icon. Change the path. If you modify the path, the Integration will run on the new path. To run the Integration, for a given method, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the new path and method pair.

**Modifying the HTTP Methods**

On the **Resources** page, select a resource operation and then click the **Edit** icon. Select new HTTP Methods with respect to which you want to expose the given Integration. To run the integration now, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the path and one of the new methods that was selected.

**Changing the mapped Integration**

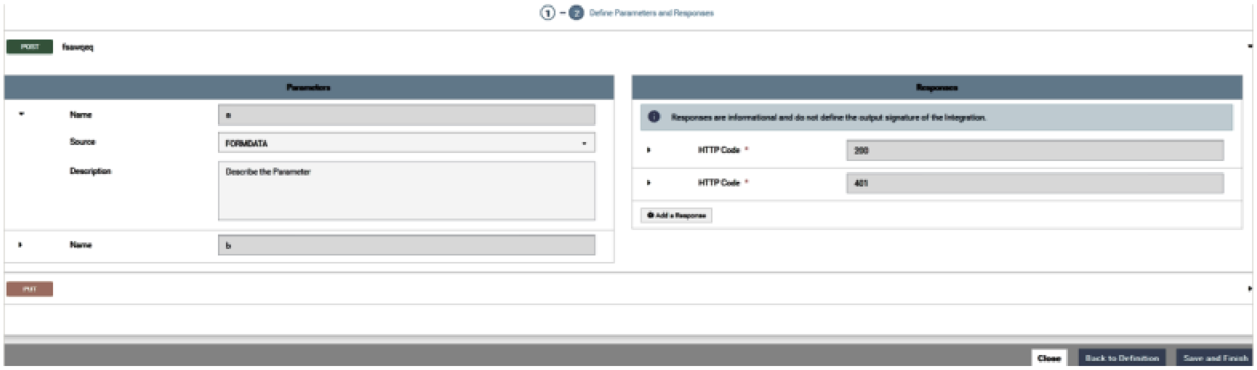
On the **Resources** page, select a resource operation and then click the **Edit** icon. Choose a new **Mapped Integration** from the drop-down list. Note that you can modify a REST resource even if the resource has an operation with a deleted integration mapped to the operation. To run the new Integration, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the Integration method and path pair.

### Modifying an existing Integration

Select an Integration in the **Integrations** page that is exposed in a REST API and click the **Edit** icon. Change the input signature and output signature of the Integration. Now go to the **REST APIs** page, select a REST API, and click on the REST API link. Go to the **Resources** page. Identify any method and path pair that corresponds to the Integration whose signature was modified. Then go to the **Swagger** page and navigate to the method and path pair that was identified. You can see that the parameters have changed and they now correspond to the new Integration input signature. Note that the responses have also changed and they correspond to the output signature of the Integration. Click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and then click **Try it out**, supply the new parameter values, and run the Integration.

### Changing Parameter Types

On the **Resources** page, select a resource operation and then click the **Edit** icon. Click **Save and Continue**. Parameters corresponding to the Integration appears.



Name: a
Source: FORMDATA
Description: Describe the Parameter
Name: b

Responses are informational and do not define the output signature of the Integration.

HTTP Code *: 200
HTTP Code *: 401

- If the parameter type is **body**, you cannot change its type.
- If the parameter type is **formdata**, you can change it to a header or a query parameter.
- If the parameter type is **header**, you can change it to a query or formdata parameter.
- If the parameter type is **query**, you can change it to a form or a header parameter.

### Adding Responses

On the **Resources** page, select a resource operation and then click the **Edit** icon. Click **Continue**. You will see the responses corresponding to the output signature of the Integration in the panel. You can add more response codes, if needed.

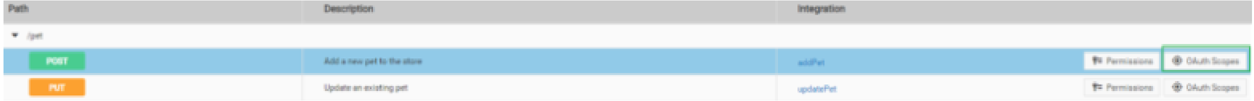
**Note:**

Click **Back to Definition** to go back to the REST Resource Operation definition page.

3. A “scope” on page 84 defines the services the client can access on behalf of the resource owner and consists of one or more services. If access is granted for a scope, then access is granted for all the services in that scope. When a request is made, Integration Cloud verifies that the scope is defined for a client. The client is allowed to access only the service URLs that are specified for the scope.

   For a Resource Path with Method, click **OAuth Scopes** to view the OAuth Scopes that contain the REST Resource path with Method.

   In the **OAuth Scopes** dialog box, click **Add URL to Another Scope**. In the **Add REST Resource Path to OAuth Scope** dialog box, select **Add To Existing Scope** to add the REST Resource path with Method to an existing scope or select **Add New Scope** to create a new scope and add the REST Resource path with Method to that new scope.



| Path | Description | Integration | |
|---|---|---|---|
| ▾ /pet | | | |
| POST | Add a new pet to the store | addPet | Permissions ⊕ OAuth Scopes |
| PUT | Update an existing pet | updatePet | Permissions ⊕ OAuth Scopes |

## Creating REST APIs with Swagger

### To create a REST API using a Swagger file

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**.

   The **REST APIs** page appears.

2. From the **REST APIs** page, click **Add New REST API**, select **Build with Swagger**, and then click **OK**.

   In the **New REST API** page, complete the following fields. Required fields are marked with an asterisk on the screen.

| Select... | To... |
|---|---|
| **Save As** | Type a name for the REST API using any combination of letters, numbers, and/or the underscore character. |
| **Swagger Source** | Select **URL** and enter the URL for the Swagger document. The URL should begin with https://.<br>OR<br>Select **File** and click **Browse** to select a Swagger document on your local file system. Swagger documents must be |

| Select... | To... |
|---|---|
| | in either JSON format with a .json file extension or YAML format with a .yaml or .yml file extension. |

Click **Save**.

Note the following points after you click **Save**:

- The REST API is created based on the Swagger definition that is uploaded or the Swagger definition that is referred to by the given URL.
- Integrations are generated based on the Operation IDs in the Swagger file. One Integration is created for each Operation ID in the Swagger file. A Swagger file cannot have two Operation IDs that are similar.
- Resource Operations are created for a unique path in the Swagger file.

3. To view the Swagger definition, from the **REST APIs** page, click the REST API link. The **Overview** page provides general information on the REST API. The **Resources** page allows you to view the resource operations. The **Swagger** page allows you to view the Swagger definition.

   You can use Access Control Lists (ACLs) to control the execution permission of a REST API. Integration Cloud associates the default ACL, *Default*, to a REST API when the REST API is created using a Swagger file. Click **Permissions** to associate the REST API with another ACL.

   **Note:**
   The ACL will be enforced only if an Integration acts as a top-level Integration. For example, if Integration A has Integration B and Integration C as sub-Integrations, then the ACL if associated, will be enforced only on Integration A.

4. To edit the Integration, go to the **Resources** page, select a Resource Operation, expand its path, and click on the integration link. The Integration will be open for editing in the Edit Integrations page. Add necessary blocks and perform mappings.

   **Note:**
   You will not be able to modify the Input/Output signature of the Integration because the signature is derived from the Swagger definition.

5. To run the Integration with the given path and method, go to the **Swagger** page for the REST API, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and then click **Try it out**.

## Copying REST APIs

Integration Cloud allows you to create a copy of a REST API from the **REST APIs** page. You can copy a REST API if you have the required permissions.

### To copy a REST API

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**. The **REST APIs** page appears.

2. Select a REST API, click on the ellipses icon ⋮ and select **Copy**.

   The **Copy** dialog box appears.

3. By default, in the **Select Project** field, the current project is selected. To copy the REST API to another project, select a different project from the **Select Project** drop-down list.

   > **Note:**
   > Ensure that you create any account or reference data associated with this REST API in the target project.

4. Type a new name in the **Copy As** field.

5. Click **Copy**. The system creates a copy of the REST API with the new name and it appears in the **REST APIs** page of the target project.

   If you create a REST API by using a file containing the Open API specification (formerly known as the Swagger specification) as a template, a copy of the REST API (including its dependencies namely its Integrations, Document Types, and Resources) is created. All resources and document types are renamed as per the new name. When Integrations encapsulated by the copied REST API are modified, the changes are reflected only in the copied REST API and not in the REST API from which it was copied. This is because, if you have created a REST API using the **Build with Swagger** approach, each REST API has its own set of Integrations.

   If you create a REST API by using an existing set of Integrations (**Build from scratch**), a copy of the REST API is created. You can type a new name at the time of copying and a REST API with the new name is created. The same REST APIs that are referred to in the original REST API are also referred to in the copied REST API. If the Integration to which it is referred is modified in the copied REST API, then the changes will reflect in the original REST API because both the REST APIs refer or point to the same Integration. However, if another path with a different Integration is added to the copied REST API, this does not show in the original REST API.

## Exporting REST APIs

Integration Cloud allows you to export REST APIs from the **REST APIs** page. The export capability is available only if you have the required license for exporting REST APIs. You can export REST APIs from one tenant and import those REST APIs to another tenant. Ensure that you have the **Export** Assets permission to export REST APIs.

When a REST API is exported, all its dependencies (Integrations, Document Types, and Resources) are exported. Integrations referred to by the REST API, their dependencies, and all dependant Document Types of the referred Integrations are also exported.

**Note:**
If assets used by a REST API are deleted, you will not be able to export the REST API.

**To export REST APIs**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**. The **REST APIs** page appears.
2. Select the REST APIs from the **REST APIs** page and click **Export**. If the REST APIs you are exporting use Reference Data, Document Types, SOAP, or REST Applications, then those Applications including the Reference Data and Document Types will also be exported along with the REST APIs.

   The **Confirm Export** dialog box appears.
3. Click **Export** to export the REST APIs. The REST APIs will be downloaded as a zip file to your default download folder. The zip file size must not be greater than 50 MB. Do not modify the contents of the exported zip file. If you modify the contents of the zip file, the REST APIs cannot be imported back to Integration Cloud.

## Importing REST APIs

Integration Cloud allows you to import REST APIs from the **REST APIs** page. You can import REST APIs from a zip file that was earlier exported from Integration Cloud. You can export REST APIs from one tenant and import those REST APIs to another tenant. You can import REST APIs provided you have the **Create** Integration permission.

**Note:**
If you want to import a REST API that has an on-premises Application, before importing the REST API, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the REST API.

**To import REST APIs**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > REST APIs**. The **REST APIs** page appears.
2. Click **Import REST APIs**.

   The **Import REST APIs** page appears.
3. Click **Browse** and select the zip file that contains the exported REST APIs. The zip file size must not be greater than 50 MB. The REST APIs available in the zip file will appear in the pane.

4. Select the REST APIs that you want to import and then click **Import**.

   The REST APIs appear in the **REST APIs** page.

   > **Note:**
   > While importing a REST API, the REST API including its dependencies are imported. If there is a REST API with the same name, you will be asked to provide a new name. If you have Integrations that have the same name as those referred to by the REST API, you will be asked whether you want to override those integrations. If you choose to override, then the Integrations in your current tenant will be removed. If you want to retain the Integrations, you can cancel importing the REST API. If you want to retain your current Integrations and also import the REST API, create a copy of your Integrations by providing another name and then override your current Integrations at the time of import of the REST API.

# 15 SOAP APIs

## Overview

Web services are building blocks for creating open, distributed systems. A web service is a collection of functions that are packaged as a single unit and published to a network for use by other software programs. For example, you could create a web service that checks a customer's credit or tracks delivery of a package. If you want to provide higher-level functionality, such as a complete order management system, you could create a web service that maps to many different Integrations, each performing a separate order management function.

A SOAP API defines a web service and it encapsulates all the information of a web service. The SOAP API contains the message formats, data types, transport protocols, and transport serialization formats that should be used between the consumer (requester) and the provider of the web service. In essence, the SOAP API represents an agreement governing the mechanics of interacting with that service.

Integration Cloud allows you to write integration logic to integrate different types of applications. This logic can be exposed to the external world using SOAP APIs. A SOAP API is a service *provided* to external users.

**Note:**
Users who have the required project permissions under **Settings** ⚙ **> Project Permissions** can create, update, delete, and execute SOAP APIs.

An *operation* is the WSDL element that exposes some functions of a web service and defines how data is passed back and forth. A SOAP API exposes one or more Integrations as *operations*, so each operation in a SOAP API corresponds to an Integration. The input for the Integration corresponds to the request body for the operation. The output of the Integration is the response body for the operation.

Using a SOAP client, you can invoke the SOAP operation *externally* by using either Basic Authentication or 2-way SSL. When the SOAP operation is invoked, the associated Integration gets executed.

You can create SOAP APIs by using an existing set of Integrations (from scratch) or by using a WSDL file:

- **Build from scratch**: You can create a SOAP API from an existing Integration. In this case, you specify the protocol and the use and style for the operations when creating the SOAP API. The Integration becomes an operation in the SOAP API. Integration Cloud uses the existing service signature as the input and output messages for the operation. You can add operations to a SOAP API or delete operations from a SOAP API that is created from scratch.

- **Build with WSDL**: You can create a SOAP API from an existing WSDL document. In this case, Integration Cloud uses the operation definitions from the WSDL to generate an Integration for each operation in the WSDL. You cannot add operations to a SOAP API created from a WSDL.

## Creating SOAP APIs from scratch

When you create a SOAP API from scratch, you select one or more Integrations to use as operations. The operation signature becomes the input and output messages for the operations in the WSDL document. However, Integration Cloud allows constructs within operation signatures that cannot be represented in certain web service use/style combinations. When adding an Integration to or creating a SOAP API from scratch, Integration Cloud verifies that the operation signature can be represented in the use/style specified for the SOAP API. If an operation signature does not meet the use/style signature requirements, Integration Cloud will not add the Integration as an operation. Or, in the case of creating a SOAP API from scratch, Integration Cloud will not create the SOAP API.

Following is a list of operation signature restrictions and requirements for each use/style. Note that this list may not be exhaustive.

| Signature Restrictions for Document - Literal |
|---|
| *body fields are not allowed at the top level |
| @attribute fields (fields starting with the “@” symbol) are not allowed at the top level |
| String table fields are not allowed |

| Signature Restrictions for RPC - Literal |
|---|
| *body fields are not allowed at the top level |
| @attribute fields (fields starting with the “@” symbol) are not allowed at the top level |
| String table fields are not allowed |
| List fields (String List, Document List, Document Reference List, and Object List) are not allowed at the top level |
| Duplicate field names (identically named fields) are not allowed at the top level |
| Top-level fields cannot be namespace qualified |
| Top-level field names cannot be in the format *prefix:localName* |

| Signature Restrictions for RPC - Encoded |
|---|
| * body fields are not allowed |
| @attribute fields are not allowed (fields starting with the “@” symbol) |
| Top-level fields cannot be namespace qualified |
| Top-level field names cannot be in the format *prefix:localName* |

### To create a SOAP API from scratch

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > SOAP APIs**.

   The **SOAP APIs** page appears.

2. From the **SOAP APIs** page, click **Add New SOAP API**, select **Build from scratch**, and then click **OK**.

3. Complete the following fields and click **Save**.

   The new SOAP API appears in the **SOAP APIs** page.

| Field | Description |
|---|---|
| **Name** | Type a name for the SOAP API using any combination of letters, numbers, and/or underscore character. |
| **SOAP Version** | Whether SOAP messages for this SOAP API should use **SOAP 1.1** or **SOAP 1.2** message format. |
| **Select Integration** | Select the Integration to use as an operation. The operation signature becomes the input and output messages for the operation in the WSDL document. |
| **Use and style for operations** | A WSDL document describes a web service and a WSDL binding describes how the service is bound to a SOAP messaging protocol. A WSDL SOAP binding can be either a document style binding or a Remote Procedure Call (RPC) style binding. A SOAP binding can also have a literal or an encoded use.<br>Select the use/style for operations in the SOAP API:<br>■ Document - Literal<br>■ RPC - Literal<br>■ RPC - Encoded |
| **Enforce WS-I Basic Profile 1.1 compliance** | Select this option if you want Integration Cloud to validate all the SOAP API objects and properties against the WS-I requirements before creating the SOAP API. |
| **Validate schema using Xerces** | Integration Cloud automatically uses an internal schema parser to validate the schemas associated with the XML Schema definition. Select this option if you want Integration Cloud to also use the Xerces parser to validate the schemas associated with the XML Schema definition. However, the Xerces parser provides stricter validation. |

| Field | Description |
|---|---|
| | As a result, some schemas that the internal schema parser considers to be valid might be considered invalid by the Xerces parser. |
| **Attachment Enabled** | The **Attachment Enabled** option is displayed when you edit the SOAP API. If attachments are enabled for the SOAP API, instances of XML-type base64Binary are transported using MIME attachments, which improves the performance of large binary payload transport. Integration Cloud supports SOAP attachments only for SOAP APIs that specify style/use of RPC-Literal or Document-Literal. |

4. Click **Save**.

   The Integration becomes an operation in the SOAP API. Integration Cloud uses the existing operation signature as the input and output messages for the operation. You can add operations to a SOAP API created from scratch.

5. To add operations, on the **Operations** page, click **Add New Operation**.

   The **Add New Operation** dialog box appears.

6. In the **Add New Operation** dialog box, type a name for the WSDL operation, select an Integration, and then click **Add**.

   The new operation appears on the **Operations** page. Using a SOAP client, you can invoke the SOAP operation *externally* by using either Basic Authentication or 2-way SSL. When the SOAP operation is invoked, the associated Integration gets executed.

## Creating SOAP APIs with WSDL

You can create a SOAP API from a WSDL document accessed through a URL or by selecting a WSDL file. You can specify whether Integration Cloud enforces strict, lax, or no content model compliance when generating document types from the XML Schema definition contained or referenced in the WSDL document. Content models provide a formal description of the structure and allowed content for a complex type. The type of compliance that you specify can affect whether Integration Cloud generates a document type from a particular XML Schema definition successfully.

Do not create a SOAP API from a WSDL that specifies RPC - Encoded, contains attributes in its operation signature, and/or has complex type definitions with mixed content. Integration Cloud might successfully create a SOAP API from such WSDLs but the SOAP API may exhibit unexpected runtime behavior.

### To create a SOAP API with WSDL

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > SOAP APIs**.

   The **SOAP APIs** page appears.

2. From the **SOAP APIs** page, click **Add New SOAP API**, select **Build with WSDL**, and then click **OK**.

   The **New SOAP API** page appears.

3. Provide a name of your SOAP API and complete the following details. Required fields are marked with an asterisk on the screen.

| Select... | To... |
|---|---|
| **SOAP API Source** | Specify the **SOAP API Source**.<br>**URL** - Select **URL** if you want to specify the URL of the WSDL. The **WSDL URL** should begin with http:// or https://. The URL is used to retrieve the WSDL for the SOAP API. Enter the **User Name** and **Password** if authentication is required to access the WSDL URL.<br>**File** - Select **File** and then click **Browse** if you want to select the WSDL from your local file system. The size of the **WSDL file** cannot exceed 5 MB. Click the ⊞ icon beside the **Browse** button if you want to add separate elements of a service definition after import, such as WSDLs or XSDs, to the primary WSDL. Ensure that you add the primary WSDL as the first WSDL, and then add separate elements of the service definition, for example, dependent WSDLs and XSDs to the primary WSDL. If you upload a new file, Integrations and Document Types that are created are now based on the uploaded file. |
| **Content model compliance** | In **Content model compliance**, select one of the following to indicate how strictly Integration Cloud enforces content model compliance when creating document types from the XML Schema definition in the WSDL document.<br>■ **Strict** - Generate the document type only if Integration Cloud can represent the content models defined in the XML Schema definition correctly. Document type generation fails if Integration Cloud cannot accurately represent the content models in the source XML Schema definition. Currently, Integration Cloud does not support repeating or nested model groups. If you select strict compliance, Integration Cloud does not generate a document type from any XML schema definition that |

| Select... | To... |
|---|---|
| | contains those items. If Integration Cloud cannot generate a document type that complies with the content model in the XML schema definition in the WSDL document, Integration Cloud will not generate the SOAP API.<br>■ **Lax** - When possible, generate a document type that correctly represents the content models for the complex types defined in the XML schema definition from the WSDL document. If Integration Cloud cannot correctly represent the content model in the XML Schema definition in the resulting document type, Integration Cloud generates the document type using a compliance mode of **None**. When you select **Lax** compliance, Integration Cloud will generate the document type even if the content models in the XML schema definition cannot be represented correctly.<br>■ **None** - Generate a document type that does not necessarily represent or maintain the content models in the source XML Schema definition. |
| **Enforce WS-I Basic Profile 1.1 compliance** | Select this option if you want Integration Cloud to validate the SOAP API objects and properties against the WS-I requirements before creating the SOAP API. The **Enforce WS-I Basic Profile 1.1 compliance** option specifies whether the SOAP API enforces compliance with the *WS-I Basic Profile 1.1*, a set of guidelines for using web services specifications to maximize interoperability (including guidance for such core web services specifications such as SOAP, WSDL, and UDDI).<br>As an example, using the RPC-Encoded style and use is not supported by the WS-I profile. If a SOAP API makes use of the RPC/Encoded style, and **Enforce WS-I Basic Profile 1.1 compliance** is enabled, Integration Cloud will indicate that the SOAP API is not compliant.<br>Enforcing WS-I compliance also affects the contents and signature for operations in the SOAP API. |
| **Validate schema using Xerces** | Integration Cloud automatically uses an internal schema parser to validate the schemas associated with the XML Schema definition. Select this option if you want Integration Cloud to also use the Xerces parser to validate the schemas associated with the XML Schema definition. However, the Xerces parser provides stricter validation. As a result, some schemas that the internal schema parser considers to be valid might be considered invalid by the Xerces parser. |

| Select... | To... |
|---|---|
| **Attachment Enabled** | The **Attachment Enabled** option is displayed when you edit the SOAP API. If attachments are enabled for the SOAP API, instances of XML-type base64Binary are transported using MIME attachments, which improves the performance of large binary payload transport. Integration Cloud supports SOAP attachments only for SOAP APIs that specify style/use of RPC-Literal or Document-Literal. |

4. Click **Save** to create the SOAP API.

   The SOAP API details page appears where you can edit the SOAP API.

5. Click the **Operations** tab to go to the **Operations** page. Integration Cloud uses the operation definitions from the WSDL to generate an Integration for each operation in the WSDL. You cannot add operations to a SOAP API created from a WSDL.

6. Click the **WSDL** tab to go to the WSDL page.

   On the WSDL page, you can view the WSDL document associated with the SOAP API.

   - The displayed WSDL document contains all the information to invoke the operations described in the WSDL.
   - For a SOAP API created from a WSDL that contains relative URIs that are anonymously addressable, Integration Cloud replaces any relative URIs with an absolute URI using the base URI of the WSDL file.

## Copying SOAP APIs

Integration Cloud allows you to create a copy of a SOAP API from the **SOAP APIs** page. You can copy a SOAP API if you have the required permissions.

### To copy a SOAP API

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > SOAP APIs**. The **SOAP APIs** page appears.

2. Select a SOAP API, click the ellipses icon ⋮, and select **Copy**.

   The **Copy** dialog box appears.

3. By default, in the **Select Project** field, the current project is selected. To copy the SOAP API to another project, select a different project from the **Select Project** drop-down list.

   **Note:**

> Ensure that you create any account or reference data associated with this SOAP API in the target project.

4. Type a new name in the **Copy As** field.

5. Click **Copy**. The system creates a copy of the SOAP API with the new name and it appears in the **SOAP APIs** page of the target project.

   If you create a SOAP API by using a WSDL file, a copy of the SOAP API (including its dependencies namely its Integrations, Document Types, and Operations) is created. When Integrations encapsulated by the copied SOAP API are modified, the changes are reflected only in the copied SOAP API and not in the SOAP API from which it was copied. This is because, if you have created a SOAP API using the **Build with WSDL** approach, each SOAP API has its own set of Integrations.

   If you create a SOAP API by using an existing set of Integrations (**Build from scratch**), a copy of the SOAP API is created. You can type a new name at the time of copying and a SOAP API with the new name is created. The same integrations that are referred to in the original SOAP API are also referred to in the copied SOAP API (provided it is copied in the same project). If the referred Integration is modified in the copied SOAP API, then the changes will reflect in the original SOAP API because both the SOAP APIs refer or point to the same Integration (This does not happen when the **Build from scratch** SOAP API is copied to a different project as the referenced integrations are also copied to the target project). However, if another operation with a different Integration is added to the copied SOAP API, this does not show in the original SOAP API.

## Exporting SOAP APIs

Integration Cloud allows you to export SOAP APIs from the **SOAP APIs** page. The export capability is available only if you have the required license for exporting SOAP APIs. You can export SOAP APIs from one tenant and import those SOAP APIs to another tenant. Ensure that you have the **Export** Assets permission to export SOAP APIs.

When a SOAP API is exported, all its dependencies (Integrations, Document Types, and Operations) are exported. Integrations referred to by the SOAP API, their dependencies, and all dependant Document Types of the referred Integrations are also exported.

> **Note:**
> If assets used by a SOAP API are deleted, you will not be able to export the SOAP API.

### To export SOAP APIs

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > SOAP APIs**. The **SOAP APIs** page appears.

2. Select the SOAP APIs from the **SOAP APIs** page and click **Export**. If the SOAP APIs you are exporting use Reference Data, Document Types, REST, or SOAP Applications, then those

Applications including the Reference Data and Document Types will also be exported along with the SOAP APIs.

The **Confirm Export** dialog box appears.

3. Click **Export** to export the SOAP APIs. The SOAP APIs will be downloaded as a zip file to your default download folder. The zip file size must not be greater than 50 MB. Do not modify the contents of the exported zip file. If you modify the contents of the zip file, the SOAP APIs cannot be imported back to Integration Cloud.

## Importing SOAP APIs

Integration Cloud allows you to import SOAP APIs from the **SOAP APIs** page. You can import SOAP APIs from a zip file that was earlier exported from Integration Cloud. You can export SOAP APIs from one tenant and import those SOAP APIs to another tenant. You can import SOAP APIs provided you have the **Create** Integration permission.

**Note:**
If you want to import a SOAP API that has an on-premises Application, before importing the SOAP API, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the SOAP API.

### To import SOAP APIs

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > APIs > SOAP APIs**. The **SOAP APIs** page appears.

2. Click **Import SOAP APIs**.

   The **Import SOAP APIs** page appears.

3. Click **Browse** and select the zip file that contains the exported SOAP APIs. The zip file size must not be greater than 50 MB. The SOAP APIs available in the zip file will appear in the pane.

4. Select the SOAP APIs that you want to import and then click **Import**.

   The SOAP APIs appear in the **SOAP APIs** page.

   **Note:**
   While importing a SOAP API, the SOAP API including its dependencies are imported. If there is a SOAP API with the same name, you will be asked to provide a new name. If you have Integrations that have the same name as those referred to by the SOAP API, you will be asked whether you want to override those integrations. If you choose to override, then the Integrations in your current tenant will be removed. If you want to retain the Integrations, you can cancel importing the SOAP API. If you want to retain your current Integrations and also import the SOAP API, create a copy of your Integrations by providing another name and then override your current Integrations at the time of import of the SOAP API.

# 16 Manage stages, Deploy assets

## Manage Stages

Stages provide safe environments for development and testing that are separated from the production environment. They allow assets to migrate from one environment to another environment. When you set up a stage, an environment is created for testing and executing the Integrations.

Integration Cloud provides ways to manage the Integration development life cycle. The typical life cycle of an Integration development involves creating Integrations, testing them, and making them production worthy. Each of these activities can be termed as different stages of an Integration development life cycle. To aide these activities, Integration Cloud provides you with **Stages**.

A predefined set of stages is allowed, each representing an activity in the Integration life cycle development. They are:

- Development
- Test
- Pre-Live
- Live

By default, every user gets a **Development Stage**. *In the Development Stage, you can create, update, delete, or view Integrations. You will not be able to create new assets in other stages.* In other stages, Integrations can be pulled from a preceding stage or deleted. Further, Integrations can be pulled into a stage only from a preceding stage.

**Note:**
You can access a stage only if your Access Profile is assigned to the stage. See "Apply Access Profiles to a Stage" on page 695 for more information.

### Adding and deleting stages

You can add stages only in the following order:

- Live
- Test
- Pre-Live

You can delete stages only in the following order:

- Pre-Live
- Test
- Live

**Note:**
Users who have the Stages **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Stages** can add or delete stages.

# Apply Access Profiles to a Stage

The typical life cycle of an Integration development involves creating Integrations, testing them, and making them production worthy. Each of these activities can be termed as different stages of an Integration development.

Every stage can be assigned a number of Access Profiles and users who are assigned the required Access Profiles can perform activities on that stage. For example, if in an Access Profile, **Execute Integrations** permission is granted, the user assigned with that Access Profile can execute Integrations on the stages to which the Access Profile is assigned. If the Access Profile needs to perform scheduling activity on the Live stage, the Access Profile needs to have access to that stage as well. The Development stage can be accessed by everyone.

Click **Add New Stage** to add the next stage. Multiple boundary arrows indicate that more stages can be added.

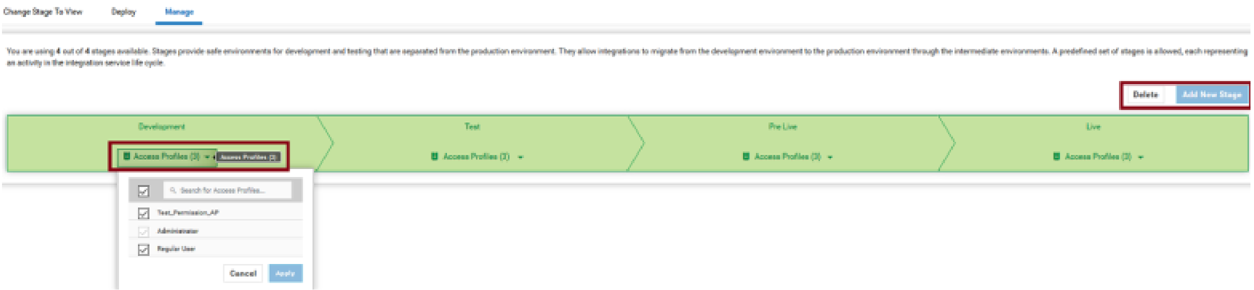
Click **Delete** to delete a stage. You cannot delete the **Development** stage.

> **Note:**
> When a stage is deleted, everything it contains is erased and cannot be recovered.

### To apply Access Profiles to a stage

1. From the Integration Cloud navigation bar, click **Stage in view > Manage**. All stages added including the Development stage are displayed. Initially, before any other stages are added, the Development stage is displayed.

2. Click **Access Profiles** and select the Access Profiles you want to apply to the stage.



You are using 4 out of 4 stages available. Stages provide safe environments for development and testing that are separated from the production environment. They allow integrations to migrate from the development environment to the production environment through the intermediate environments. A predefined set of stages is allowed, each representing an activity in the integration service life cycle.

> **Note:**
> By default, the **Administrator** and **Regular User** Access Profiles are associated with the Development Stage. If you have created a new Access Profile, ensure that the Access Profile you have created is associated with the Development Stage.

3. Click **Apply**.

   The Access Profiles are applied to the selected active stage.

> **Note:**

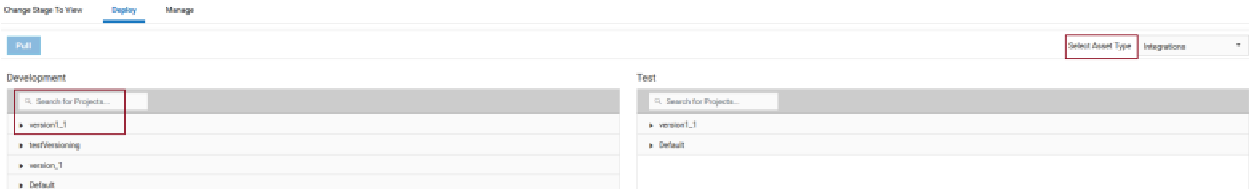
When an Integration is pulled, all its dependents will also be pulled and copied to that stage.

## Deploy Assets

This page displays all the assets under the relevant projects that are available in the current stage and the previously selected stage. Click the **Change Stage To View** link to deploy assets to another stage.

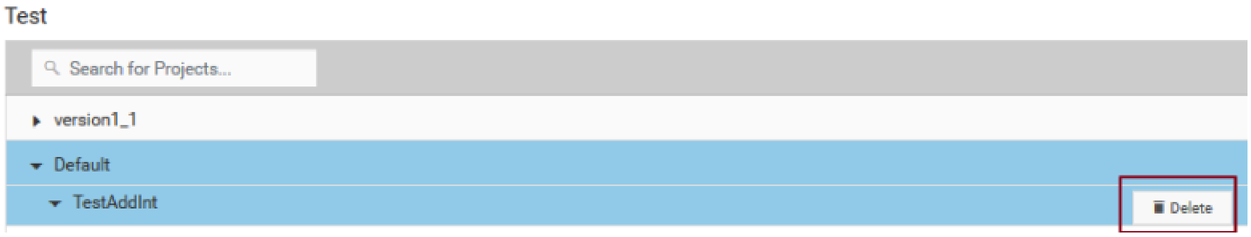
**Note:**
Users who have the Assets **Deploy** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Assets** can deploy assets.

After you change the current stage to any other stage other than the Development stage, the **Deploy** page gets populated with the relevant assets selected in the **Select Asset Type** drop-down list.



You can select an asset that was in the earlier stage (left panel) and click **Pull** to pull the asset to the current stage (right panel). The assets are categorized under projects. If an asset is pulled and if the associated project is not present in the current stage, the project along with the asset will be available in the current stage.

You can pull an asset only if the asset is consistent in the source stage (left panel), that is, all references of the asset are present in the source stage. The pulled asset can be deleted from the current stage (right panel).



**Note:**
You can access a stage only if your Access Profile is assigned to that stage.

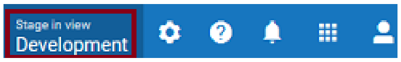
## Change Stage To View

This page allows you to change the current stage. Only active and accessible stages appear in the drop-down list for selection in the **Stage to view** field.

**Note:**
You can access a stage only if your Access Profile is assigned to that stage.

After you change the current stage and click **Submit**, assets and services will be displayed in the user interface pages only for the selected current stage.

The **Stage In View** label on the navigation bar displays the current stage.



# 17 Keys and Certificates

## Overview

Keystores and truststores are files that function as repositories for storage of keys and certificates necessary for SSL authentication, encryption/decryption, and digital signing/verification services. Keystores and truststores provide added layers of security and ease of administration, compared to maintaining the keys and certificates in separate files.

Integration Cloud stores its private keys and SSL certificates in keystore files and the trusted roots for the certificates in truststore files. Keystores and truststores are secure files with industry-standard file formats.

If you want to run services that submit HTTPS requests to other resources on the Internet, your server will be acting as a client and will receive certificates from these resources. In order for these transactions to work, your server must have copies of their public keys and signing CA certificates.

To identify a particular keystore or truststore file, or private key within a keystore, aliases are used. The use of aliases simplifies keystore and truststore management, because you do not need to enter path information when specifying a keystore, truststore, or the private key.

**Note:**
You can add, edit, or view keystore and truststore aliases and partner's self-signed certificates from **Projects > <Select a Project> > Keys & Certificates** and can use them to secure your Application Accounts. Some Applications, including custom REST Applications allow two-way SSL authentication by providing keystore and truststore aliases in the Account Configuration section. Users who have the **Administer** permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Keystores, Truststores, and Partner Certificates.

To add a Keystore, from the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Keystores > Add Keystore**.

To add a Truststore, from the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Truststores > Add Truststore**.

To add a Partner Certificate, from the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Partner Certificates > Add Certificate**.

## Add Keystore

Integration Cloud allows you to upload a Keystore file to store SSL certificates and keys. A Keystore file contains one or more pairs of a private key and signed certificate for its corresponding public key. From this screen, you can create aliases for the Keystore, so that they can be referenced while creating an Account for an Application.

### To add a Keystore

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Keystores > Add Keystore**.

2. Provide a name and description for the **Keystore file**.

3. In the **Type** field, select the Keystore file format. The default file format is **JKS**. You can also use **PKCS12**, a commonly used, standardized, certificate file format that provides a high degree of portability.

4. In the **Provider** field, select the provider from the list of available providers. The corresponding provider will be available in the provider list for a selected Keystore type.

5. Click **Browse** to select the Keystore file.

6. In the **Passphrase** field, enter the passphrase for the Keystore file. The passphrase must have been defined at the time the Keystore was created.

7. Click **Next** to protect the Key Aliases with passphrases. A key alias is a label for specific key within a Keystore. Enter a passphrase for each Key Alias found in the Keystore file, and then click "Finish" to upload the Keystore file.

The uploaded Keystore file can be used while creating an Account for an Application.

## Add Truststore

Integration Cloud allows you to upload a Truststore file, which contains the trusted root of the certificate or signing authority (CA). From this screen, you can create aliases for the Truststore, so that they can be referenced while creating an Account for an Application.

### To add a Truststore

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Truststores > Add Truststore**.

2. Provide a name and description for the Truststore file.

3. In the **Type** field, select the Truststore file format. The default file format is **JKS**. You can also use **PKCS12**, a commonly used, standardized, certificate file format that provides a high degree of portability.

4. In the **Provider** field, select the provider from the list of available providers. The corresponding provider will be available in the provider list for a selected Truststore type.

5. Click **Browse** to select the Truststore file.

6. In the **Passphrase** field, enter the passphrase for the Truststore file. The passphrase must have been defined at the time the Truststore was created and is used to protect the contents of the Truststore.

7. Click **Save** to upload the Truststore file.

The uploaded Truststore file can be used while creating an Account for an Application.

## Add Partner Certificate

Integration Cloud allows you to upload the Partner's certificate which contains its public key. The Partner's certificate with the public key is required to encrypt outbound request messages and verify the signature of inbound messages.

From this screen, you can create aliases for Partner Certificates, so that they can be referenced while creating an Account for an Application.

**To add a Partner Certificate**

1. From the Integration Cloud navigation bar, click **Projects > <Select a Project> > Keys & Certificates > Partner Certificates > Add Certificate**.

2. Provide a name and description for the Partner Certificate file.

3. Click **Browse** to select the Partner Certificate file.

4. Click **Save** to upload the Partner Certificate file.

The uploaded Partner Certificate can be used while creating an Account for an Application.

# 18 Containers

## Overview

Integration Cloud allows you to package existing webMethods Integration Server services as images or repositories and upload them on Integration Cloud using the Docker CLI. Docker is an open-source technology that allows you to deploy applications to software containers. A Docker container is an instance of a Docker image, where the Docker image is the application, including the file system and runtime parameters. To facilitate running webMethods Integration Server in a Docker container, webMethods Integration Server provides a script to use to build a Docker image and then push the resulting Docker image to a Docker registry hosted in Integration Cloud. You can store Docker images in a registry on Integration Cloud and manage those Docker images from the Integration Cloud user interface. Images or a repository are versioned or labeled using tags, that is, a tag is a label applied to an image or a repository. Tags help you to distinguish various images or repositories.

**Note:**
Integration Cloud documentation assumes that you are familiar with Docker technology. An in-depth discussion of Docker and container technology is beyond the scope of this document. For information on using webMethods Integration Server with Docker, see the *webMethods Integration Server Administrator's Guide*.

Images are read-only templates from which containers are instantiated, that is, a container is a runtime instance of an image. A container also consists of an execution environment and a standard set of instructions. After uploading an image on Integration Cloud, you can create and launch services from the image/tag to the desired active stage, specify the number of containers and the container port for each service, and see details of the running instances. A service can contain one or more containers and is defined as a named group of containers created out of a single image tag.

**Note:**
Universal Messaging (UM) can also be run as a container in Integration Cloud. For creating the UM Docker images supported by Integration Cloud, you must run the Integration Cloud UM script to modify the base image before it is uploaded into Integration Cloud.

You can access containers if you have the **Settings** ⚙ **> Access Profiles > Administrative Permissions > Container > Access** permission. You can administer containers if you have the **Settings** ⚙ **> Access Profiles > Administrative Permissions > Container > Administer** permission. On the **Settings** ⚙ **> Access Profile > Container** tab, enter the names of the webMethods Integration Server Access Control List (ACL) groups separated by a comma, for example, Administrators, Developers, and so on. Users who are assigned to this Access Profile will also be now part of the webMethods Integration Server container user group (s) and can perform tasks allowed for those user groups. Note that Integration Cloud Administrator profiles are not automatically assigned to the webMethods Integration Server Administrators ACL group. If you do not map an Access Profile to an webMethods Integration Server group, you will not be able to invoke webMethods Integration Server services.

**Note:**
Enabling the CSRF security feature will prevent CSRF attacks. Enable **CSRF Guard** and configure the CSRF guard settings in webMethods Integration Server Administrator before you create

the Docker image and upload it to Integration Cloud. See the *webMethods Integration Server Administrator's Guide* on the Software AG Documentation website at http://documentation.softwareag.com for information on how to enable CSRF Guard.

You can use the Docker Command Line Interface (CLI) to perform the following tasks:

Log in to the system: `#docker login -u <username> -p <password> https://<subdomain>.<domain.com>/`, for example, `docker login -u x@x.com -p test123 https://john.wmic1.com/`.

Tag an image or repository: `#docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2`.

Push or upload an image or repository: `#docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker push john.wmic1.com/john/development/is_912:withkeystore2`.

Pull or download an image or repository: `docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker pull john.wmic1.com/john/development/is_912:withkeystore2`.

## Managing Repositories

Images or repositories are read-only templates from which containers are instantiated. The **Repositories** screen allows you to view, delete, and add repositories for the active stage. You can select a repository and click **Delete** to delete a repository or click **Add New Repository** to add a repository. In the **Add New Repository** screen, select the **Deployment Stage** and enter the **Repository Name**. If a stage is not enabled to access containers, contact Support to enable the stage.

### To view the details of a tag from the Repositories screen

1. Select a repository and then click the repository link under the **Name** column.

   The **Image Tags** screen is displayed. A list of all image tags is displayed along with their names.

2. From the **Image Tags** screen, you can select an image tag and delete it or add a new service for the tag. See "Viewing Tag Details" on page 706 for more information.

3. Click the **Commands** tab to view and copy the commands on how to log in, tag images, push images, or pull images.

   Log in to the system: `#docker login -u <username> -p <password> https://<subdomain>.<domain.com>/`, for example, `docker login -u x@x.com -p test123 https://john.wmic1.com/`.

Tag an image or repository: `#docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2`.

Push or upload an image or repository: `#docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker push john.wmic1.com/john/development/is_912:withkeystore2`.

Pull or download an image or repository: `docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker pull john.wmic1.com/john/development/is_912:withkeystore2`.

## Viewing Tag Details

This screen displays details of all the tags for a repository. You can delete an image tag or add a new service. You can also click the **Commands** tab to view and copy commands on how to log in, tag images, push images, or pull images.

### To view the image tag details

1. From the Integration Cloud navigation bar, click **Containers > Repositories**.

2. Select a repository and then click on the repository link. The **Image Tags** screen appears. A list of all Docker image tags is displayed along with their names.

   From the **Image Tags** screen, select a tag and then click **Delete** to delete the image tag if it is not used by any service or click **Add New Service** to create a new service for the image tag. In the **New Service** window, specify the **Service Name**, the **Volume Name**, and the number of **Docker Containers** to instantiate.

3. From the **Image Tags** screen, select an image tag and click on the image tag link to view the Image Tag details screen. The Image Tag details screen displays the deployment stage of the image tag including the image tag label details. The image tag label details are as follows:

   - Image Type - Mandatory field. Indicates the type of image, for example, webMethods Integration Server or Universal Messaging.

   - Description, Build Number, and Version - Optional fields you had defined while creating the image.

   - Pushed At - System generated value. Date and time when the Docker image was pushed to the repository.

   - Size - System generated value. Indicates the size of the Docker image.

   - Exposed Port - Mandatory field. The port you had defined while creating the image. Ensure that the exposed port is the same as defined on your application, that is, on webMethods Integration Server or Universal Messaging.

The Image Tag details screen also displays when the screen was last refreshed, used and available containers for the displayed active stage, and information on all services created for the tag. You can edit, delete, start, stop, or add a new service from the Image Tag details screen. Click on a service link to view the service details screen. See Managing Services for more information.

4. The **Commands** screen allows you to view and copy the commands on how to log in, tag images, push images, or pull images:

   Log in to the system: `#docker login -u <username> -p <password> https://<subdomain>.<domain.com>/`, for example, `docker login -u x@x.com -p test123 https://john.wmic1.com/`.

   Tag an image or repository: `#docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2`.

   Push or upload an image or repository: `#docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker push john.wmic1.com/john/development/is_912:withkeystore2`.

   Pull or download an image or repository: `docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker pull john.wmic1.com/john/development/is_912:withkeystore2`.

## Managing Services

The **Services** screen displays details of all services in the active stage, the repository name and tag name, status of the service, and number of Docker containers instantiated for a tag. You can start a service, stop a service, edit a service, delete a service, add a new service, refresh the screen, and view details on when the screen was last refreshed.

> **Note:**
> You can also add a new service for a specified image tag from the **Containers > Repositories > Click a repository link > Image Tags** screen.

From the **Services** screen, select a service and then click the link to view the **Service details** screen. In the Service details screen, you can add or remove docker containers for the service for the selected repository and image tag.

The Service Details screen provides information on the service status (Stopped, Pending, Running), the repository, Deployment stage, and the image tag. You can start a service if it is in a stopped state. The **Docker Containers** pane in the Service Details screen displays information on the docker containers, their status, and the **Admin URL**, which is the webMethods Integration Server Administrator URL. You can modify Docker containers to instantiate by clicking **Manage Containers**, or click the **Admin URL** link to log into webMethods Integration Server Administrator. In the **Exposed Services** pane, you can view the **Exposed Docker Services**. The exposed Docker service URL can be used to execute the webMethods Integration Server Administrator service running in the Docker container.

> **Note:**
> Enabling the CSRF security feature will prevent CSRF attacks. Enable **CSRF Guard** and configure the CSRF guard settings in webMethods Integration Server Administrator before you create the Docker image and upload it to Integration Cloud. See the *webMethods Integration Server Administrator's Guide* on the Software AG Documentation website at http://documentation.softwareag.com for information on how to enable CSRF Guard.

> **Note:**
> You can now invoke services exposed by webMethods Integration Servers running in the Docker Containers from Integrations. For more information, see the *On-Premises Applications* section and also the *Configuring On-Premise Integration Servers for webMethods Cloud* document.

To add a new service, from the **Containers > Services** screen, click **Add New Service**. The **New Service** screen appears. In the **New Service** screen, enter the following fields:

- **Repository** - Select a repository.
- **Image Tag** - Select an image tag.
- **Service Name** - Enter a valid service name. The service name can contain only alphanumeric characters and should not be more than 16 characters.
- **Volume Name** - Enter the storage volume name. The Volume name is displayed only for Universal Messaging (UM) docker images and represents the data volume for the docker container to persist the container data.
- **Docker Containers** - Enter the number of containers to instantiate for the service.

Click **Save** to create a new service.

To stop a service, from the **Services** screen, select a service that is in **Running** state and then click **Stop**. Click ↻ and refresh the screen to view the latest service status.

To delete a service, from the **Services** screen, select a service that is not in **Running** state and then click **Delete**.

To start a service that is in **Stopped** state, select a service and then click **Start**. The launch service screen appears. Specify the number of **Docker Containers** to instantiate for the selected service and then click **Start**. Click ↻ and refresh the screen to view the latest service status.

# 19 End-to-End Monitoring

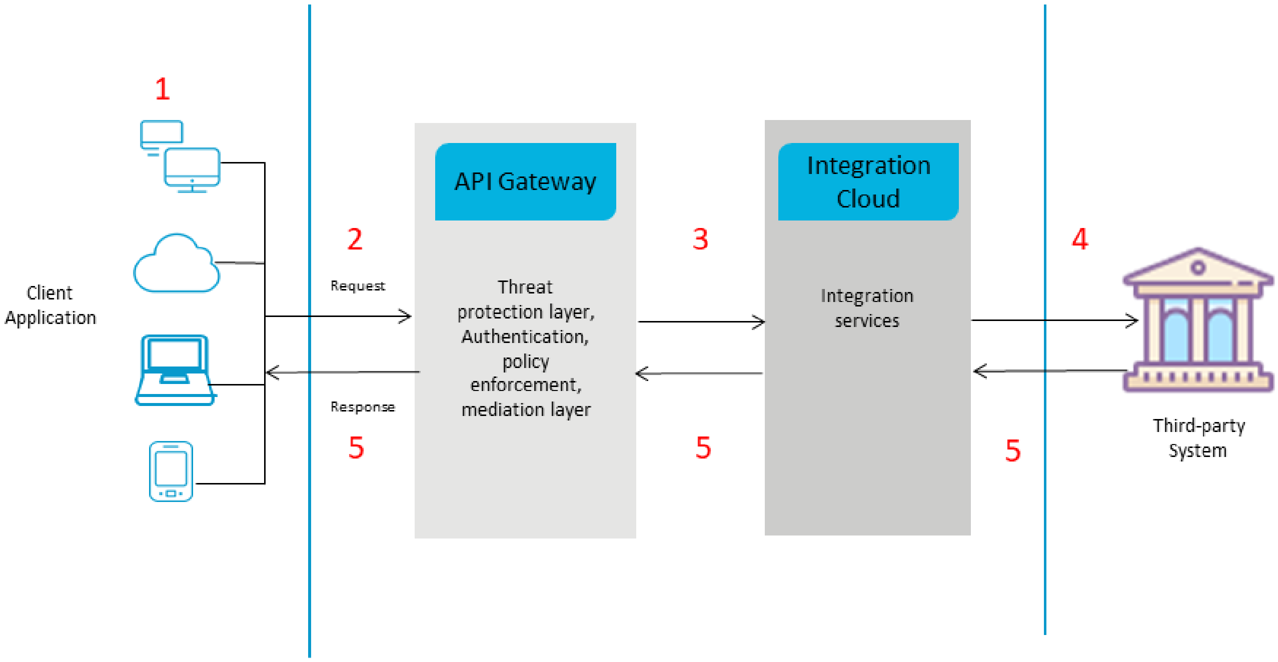
## Overview

End to End Monitoring is a cloud offering by Software AG . It is your interface to view the flow of a business transaction in a unified manner from start to finish across webMethods cloud systems. Not just that, it allows you to monitor a business transaction and identify any error that occurs during a business transaction along with the details of the application and the time at which the error occurred.

To know more, check out "User Journey " on page 728.

### Architecture

The section explains how End to End Monitoring helps a user to get the details of a business transaction.

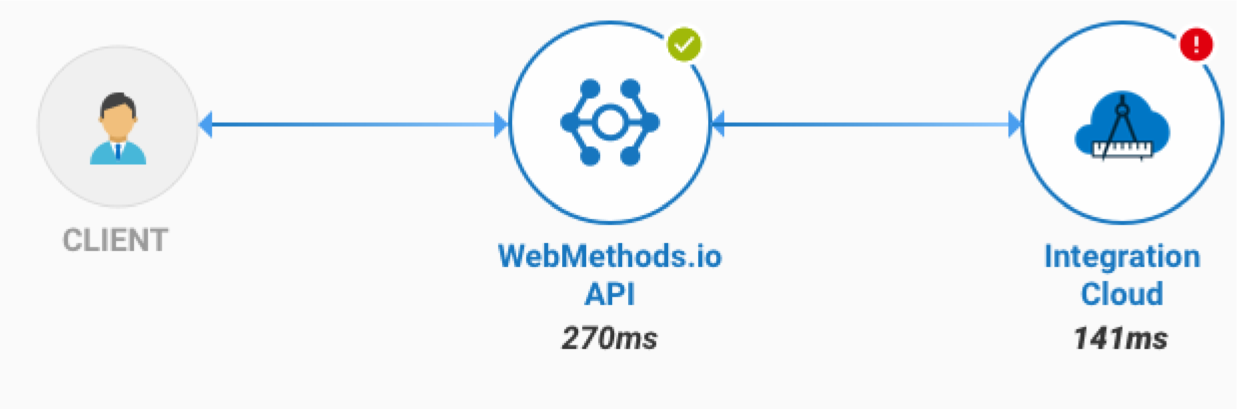


Let us consider an example, where a customer is requesting for an address change with his bank using a banking application. Here, the banking application is represented as a Third-party system. Following actions take place when the customer wants to change the address:

1. Customer raises an address change request using the client application. Client application is accessible using any device.
2. The request makes an API call to API gateway where the user is validated. Any policy enforcement required happens here.
3. After successful validation, an Integration service call is made to webMethods Integration Cloud. For example, in this case let us say that the Integration service call made to update the customer record is **updateaddress**.
4. The Integration service call authenticates the user with the bank and updates the record in the database of the bank.
5. On successful update of the record, a confirmation message is sent to the client application as a response.

### The Role of End to End Monitoring

End to End Monitoring tracks the business transaction from start to finish and provides details of the transaction. A graphical representation of a business transaction flow is as follows. For more information, see “Business Transaction Details page” on page 725.



## General Data Protection Regulation (GDPR) Considerations

From a GDPR perspective, the following considerations are to be noted:

- As part of the rule creation and group creation, the application collects and stores the user ID and email address of the user in the Elastic Search database. The user ID and email address may be personal data or personally identifiable information.

  **Note:**
  The user ID is the ID using which the user logs into webMethods Integration Cloud.

- The user does not have an option to delete this personal data or personally identifiable information from the Elastic Search database. However, if a tenant is deleted, then the user related data is also deleted.
- End to End Monitoring does not store any log information in the database.

# Working with End to End Monitoring

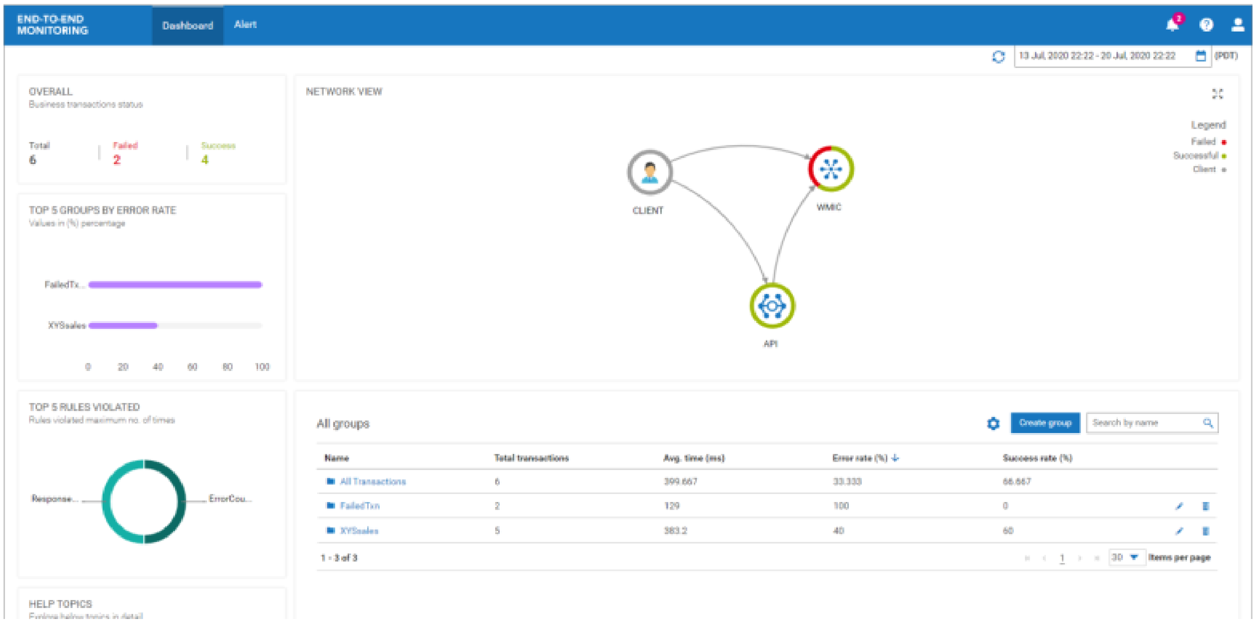
This section provides information about the various features of End to End Monitoring.

## Dashboard for End to End Monitoring

The dashboard for End to End Monitoring gives you a collective view of all the business transactions carried out within your cloud platform. The default **Dashboard** view includes the transaction widgets and the default **All Transactions** group. The **All Transactions** group contains all the transactions within the application and cannot be deleted.

**Note:**
When you perform sorting, the **All Transactions** group remains on the top of the list.

An example of the dashboard that shows the **All Transactions** group along with the custom groups created by an end user is as follows:



| Name | Total transactions | Avg. time (ms) | Error rate (%) ↓ | Success rate (%) |
|---|---|---|---|---|
| All Transactions | 6 | 399.667 | 33.333 | 66.667 |
| FailedTxn | 2 | 129 | 100 | 0 |
| XYSsales | 5 | 383.2 | 40 | 60 |

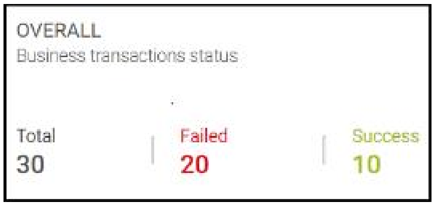
- 🔔 - An alert icon to notify you of any new transaction notifications.
- ❓ - User help that includes the help content for this application.
- 👤 - Use the link provided here to log off from the application.
- Time range - Below 🔔 is the drop-down list for time range selection. By default, it is set to half an hour prior from the current system time. For example, if current system time is 3:30 PM, then the time range shows as 3 PM to 3 30 PM. Click 📅 next to the time range to select a specific duration.

  **Note:**

The time zone is set automatically based on the system time.

- Transaction widgets - Following widgets are available on the dashboard:
  - **Overall** - This is a summary of all the transactions processed based on the following parameters:
    - Failed
    - Success

  An example of the **Overall** widget is as follows:

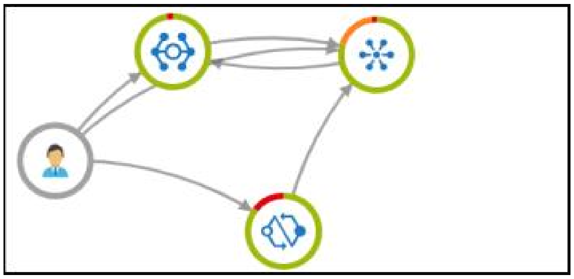


OVERALL
Business transactions status

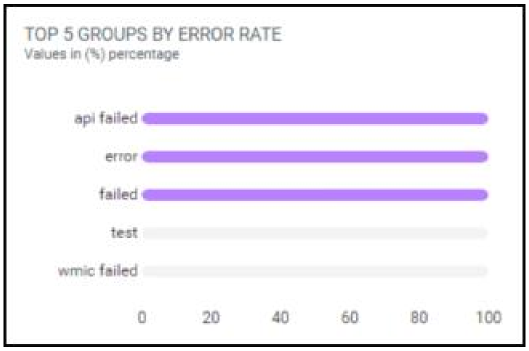
| Total | Failed | Success |
|---|---|---|
| 30 | 20 | 10 |

  - **Network View** - This is a graphical representation of all the transactions across all the supported nodes. It shows the transactions flowing through each runtime as well as the status of each runtime for the selected time range.

  An example of Network View is as follows:

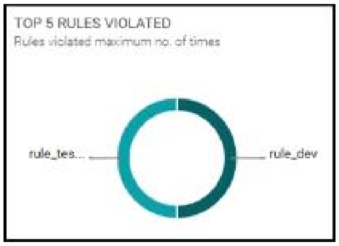


  - **Top 5 Groups by Error Rate** - This is a graphical representation which shows the top 5 groups based on the error rate. The values are displayed in percentage. The **All Transactions** group is excluded from this widget. An example of this widget is as follows:



TOP 5 GROUPS BY ERROR RATE
Values in (%) percentage

- **Top 5 Rules Violated** - This is a graphical representation which shows the top 5 rules that are violated the maximum number of times. It is a Pi chart that shows a count for the number of times these rules are violated. An example of this widget is as follows:



TOP 5 RULES VIOLATED
Rules violated maximum no. of times

- **Help Topics** - This provides you quick access to the help content for some of the commonly used topics.
- **All groups** - This lists all the transaction groups you create using the application and also the default **All Transactions** group. At the group level, you can see the following parameters for the selected time range:

  > **Note:**
  > You can customize the parameters to view by clicking ⚙ next to **Create Group**.

  - **Name** - Name of the group.
  - **Total transactions** - The total number of transactions for each group.
  - **Avg. time (ms)** - Average time of execution for all the transactions within a group.
  - **Error rate (%)** - The rate of error in percentage for all the transactions within a group.
  - **Success rate (%)** - The rate of success in percentage for all the transactions within a group.
- **Create group** - Select this option to create a custom group of transactions. For more information, see "Creating a Group of Transactions" on page 714.
- **Search by name** - Search for a transaction group by typing its name in the search box.

## Creating a Group of Transactions

Creating a group of transactions allows you to categorize the transactions based on your requirements.

### To create a group of transactions

1. On the Dashboard page, click **Create Group**.
2. In the Create group dialog box, provide the following details:

| Parameter | Description |
|---|---|
| Group Name | Type a name for this group. |

| Parameter | Description |
|---|---|
| | **Note:**<br>The group name cannot be edited after you save this form. |
| Status | Group the transactions on the basis of their status. It can be a combination of:<br>- Success<br>- Failed |
| Starts with | Group the transactions on the basis of the application from where the business flow starts. You can choose one or more products when creating the group. For example, you can have a group where you have one transaction starting from webMethods API Gateway and another transaction starting from Integration Cloud. You can select from the following:<br>- webMethods API Gateway<br>- Integration Cloud<br>- webMethods.io B2B |
| Duration (ms) | Group the transactions on the basis of their execution time in milliseconds. You can select from the following:<br>- ≤ - Select this operator to group the transactions that have an execution time which is lesser than or equal to the value specified here.<br>- ≥ - Select this operator to group the transactions that have an execution time which is greater than or equal to the value specified here.<br>- ≤ ≥ - Select this operator to group the transactions that have an execution time which is between the values specified here. For example, if you specify 20 and 30 (ms), then both the values 20 and 30 are also taken into consideration. |
| Product(s) involved | Group the transactions on the basis of the product(s) involved in a business transaction. Use the drop-down list to select one or more products. The group includes all the transactions for the selected product. If you select more than one product, it will list the transactions for the standalone products along with the transactions that involve both the products. For example, if you select webMethods API Gateway and Integration Cloud, the group will include the following:<br>- all transactions that involve only Integration Cloud |

| Parameter | Description |
|---|---|
| | - all transactions that involve only webMethods API Gateway<br>- all transactions that involve both Integration Cloud and webMethods API Gateway. |
| Transaction name(s) | Group the transactions on the basis of their names. Click **+** to add more than one entry.<br>**Note:**<br>Click **X** to remove the added names. |
| Error Message(s) | Group the transactions on the basis of an error message. Click **+** to add more than one entry. |

3. Click **Save**.

## Viewing the Transactions List

1. On the Dashboard page, select the group for which you want to view the transactions. On the Transactions list page, you will see all the transactions for the selected time frame and the set filter(s).

2. Click ⚙ on the top-right corner of the screen to open the **Show Columns** dialog box.

3. Select the parameters for which you want to view the transaction details and click **Save**. The transactions list shows the following parameters:

4. You can also filter the transactions from this page. For more information, see "Filtering the Business Transactions" on page 717

| Parameter | Description |
|---|---|
| Status | Status of the transaction. Currently, we have the following states:<br>- ✓ - Transaction successful.<br>- ! - Transaction failed. |
| Name | Name of the transaction. This name is based on the starting point of the transaction. For example, if the transaction starts from webMethods API Gateway, then this name would be the name of the API. |
| Duration (ms) | Total time taken by the transaction to complete. |

| Parameter | Description |
|---|---|
| Started from | The application from which the transaction starts. For example, webMethods Integration Cloud. |
| Started on | Start time for the transaction. The time zone is set based on the system time. |
| Error message | Message with which the transaction failed. The message displayed here pertains to the first component in which the error occurs. |
| Trace ID | This is a unique identifier for the transaction. |

## Editing a Transactions Group

1. On the Dashboard page, select ✎ in the row for the transaction that you want to edit.

   > **Note:**
   > Editing a transaction group impacts all the rules associated with it.

2. Make the necessary changes and click **Update**.

## Deleting a Group

1. On the Dashboard page, select 🗑 in the row for the transaction that you want to delete.

   > **Note:**
   > Deleting a transaction group deletes all the rules and the rule violations associated with it.

2. Click **Delete**.

   On successful deletion, you will see a confirmation message on the screen.

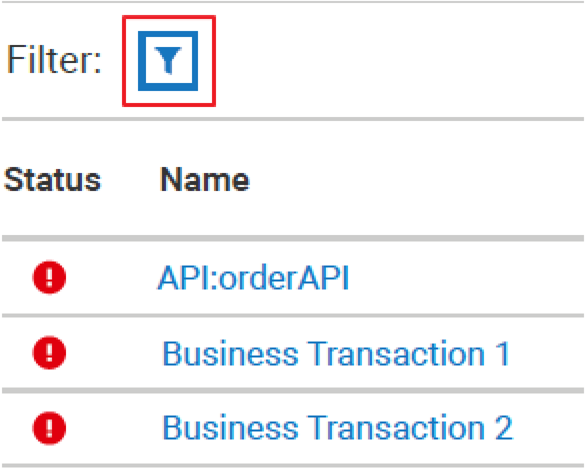
## Filtering the Business Transactions

You can filter the business transactions within any group to view a custom set of transactions.

> **Note:**
> Filters that were set during group creation are already applied when you open the group.

**To filter the transactions**

1. On the Transactions list page, click ▼ as shown in the following example:



Filter:

| Status | Name |
|---|---|
| ❗ | API:orderAPI |
| ❗ | Business Transaction 1 |
| ❗ | Business Transaction 2 |

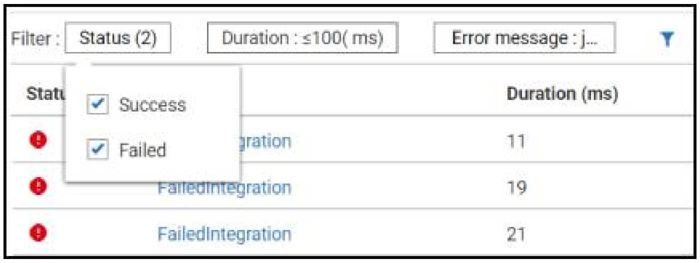
2. In the Filter dialog box, you can choose from the following filters:

> **Note:**
> Filters set during group creation are greyed out and not available for selection.

| Value | Description |
|---|---|
| Status | Select the transaction status. You can select from the following and click **Save**:<br>■ **Success** - Lists all the completed business transactions.<br>■ **Failed** - Lists all the failed transactions. |
| Name | Type the name of the business transaction and click **Save**. To add more than one transaction name, click **+** |
| Duration | Select from the following operators using the drop-down list and provide a valid time in milliseconds:<br>■ ≤ - Select this operator to filter the transactions that have an execution time which is lesser than or equal to the value specified here.<br>■ ≥ - Select this operator to filter the transactions that have an execution time which is greater than or equal to the value specified here.<br>■ ≤ ≥ Select this operator to group the transactions that have an execution time which is between the values specified here. For example, if you specify 20 and 30 (ms), then both the values 20 and 30 are also taken into consideration. |

| Value | Description |
|---|---|
| Starts with | Filter the transactions on the basis of the application from where the business flow starts. Select the checkbox next to the product. You can select more than one product. |
| Error message | Filter the transactions based on the error message with which it failed. Type the error message and click **Save**. For example, server not found. You can filter based on more than one error message by clicking **+** next to the error message box. |
| Product(s) involved | Filter the transactions based on the products that are involved with this transaction. Select the checkbox next to the product and click **Save**. You can select more than one product. |
| Trace ID | Filter the transactions based on this unique identifier associated with a transaction. Enter the trace ID and click **Save**. You can filter based on more than one Trace ID by clicking **+** next to the trace ID box. |

3. The applied filters show up on the top of the page as shown in the following example:



Filter : Status (2) Duration : ≤100( ms) Error message : j...

☑ Success
☑ Failed

| Statu | | Duration (ms) |
|---|---|---|
| ● | ration | 11 |
| ● | FailedIntegration | 19 |
| ● | FailedIntegration | 21 |

You can close a particular filter by selecting the filter and clicking **X** adjacent to the filter name.

4. After you apply the filters and retrieve the list of transactions, you can further sort the list by clicking the column headings. Sort in ascending or descending order by toggling the column header.

## Alert

The Alert page lists all the Rule violations and the Rule list. From the Alert page, you can create **Rules**. These rules are a set of conditions. When these conditions are met by a group of transactions, the rule violation occurs. You can configure a rule such that it triggers an alert in the form of an on-screen notification or email or both. The rules that you create are applicable to all the transactions of a group with which it is associated. For information on creating a rule, see "Creating a Rule" on page 720

### Notification Alerts

Notifications for End to End Monitoring are currently shown on the top-right corner of your application screen. For example, when there are two new rule violations, you will see 🔔.

## Email Alerts

When you create a rule, you have an option to be notified through an email. Whenever a group violates the conditions defined in a rule, you will receive system generated emails with the details of the violation.

## Rule Violations

This includes a list of all the rules that are violated by the transactions in a group. For more information, see “Working with Rule Violations” on page 723.

## Rule List

This includes the list of all the rules that you create in the application. It also lists all the pre-defined rules available with the application. For more information on pre-defined rules, see “Default Rules” on page 722.

## Viewing Alerts

### To view the alerts

1. Click 🔔 at the top of the screen to open the Rule violations pane. Here, you will see a list of all the new rule violations along with the older ones.

   If there are new rule violations, you will see a number with 🔔. For example, two new rule violations are indicated by 🔔 at the top of the screen.

2. In the Rule violations pane, click **View all** to go to the Rule violations section of the Alert page.

3. In the **Rule violations** tab, you will see all the rules violations listed in a table format.

## Creating a Rule

The application triggers alerts when a group of transactions violates the conditions defined in a rule associated with it.

### To create a rule

1. On the Alert page, click **Rule list**.

2. Select **Create rule**.

   **Note:**
   The application provides you a set of default rules. For more information, see “Default Rules” on page 722

3. In the Create rule dialog, provide the following details:

   - In the General Information section, provide the details for the following and click **Next**.

| Value | Description |
|---|---|
| Name | Provide a name for the rule.<br>**Note:**<br>You cannot change the rule name once you save the rule. |
| Description | Provide a meaningful description for the rule. |
| Group | Select the group from the drop-down list.<br>**Note:**<br>Currently, you can associate a rule only to a single group. However, a group can be associated with multiple rules. |

   - In the Rule Expression section, provide the details for the following and click **Next**.

| Value | Description |
|---|---|
| KPI | This is the key performance index. Select from the following options:<br>- Error count<br>- Error rate<br>- Average response time |
| Time range | The time range on the basis of which this rule should trigger an alert when violated. Select from the following options:<br>- In last 30 mins<br>- In last 1 hour<br>- In last 12 hours<br>- In last 24 hours |
| Operator | Select the operator on the basis of which this rule should trigger an alert when violated. Select from the following options:<br>- Less than (<) - lesser than the value specified in the **Value** field.<br>- Greater than (>) - greater than the value specified in the **Value** field. |

| Value | Description |
|---|---|
| | • Equals to (=) - equal to the value specified in the **Value** field.<br>**Note:**<br>**Value** could be a number, percentage, or time and changes on the basis of the KPI you select. |
| Value | This field varies on the basis of your selection in the **KPI** field. The value field changes on the basis of the KPI you select:<br>• For **Error count**, it is set to **Value(number)**.<br>• For **Error rate**, it is set to **Value(%)**.<br>• For **Average response time**, it is set to **Value(ms)**. |

- In the **Action** section, select your notification preference and click **Next**. You can select either or both of the following options:

| Value | Description |
|---|---|
| Show app notification | Select this option to get on-screen app notifications. |
| Send email | Select this option to subscribe for email alerts whenever a group violates a rule. Provide the recipient email address. You can enter more than one email address separated by a comma. |

- In the **Summary** section, verify all the details provided by you and click **Save**.

### Default Rules

The application provides you with a set of pre-defined rules. By default, these rules are disabled. These pre-defined rules are as follows:

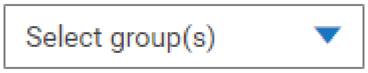
**Note:**
Other than the **Name**, you can modify all the other values as per your requirements.

| Name | Description | Group | KPI | Time Range | Operator | Value | Show app notification | Send email |
|---|---|---|---|---|---|---|---|---|
| Average response time is out of compliance | The average response time for all transactions in last 1 hour is greater than 1000 ms. | All Transactions | Average response time | In last 1 hour | > | 100 (ms) | True | False |
| Error count is out of compliance | The error count for all transactions in last 1 hour is more than 100. | All Transactions | Error count | In last 1 hour | > | 10 | True | False |
| Error rate is out of compliance | The error rate for all transactions in last 1 hour exceeds 10%. | All Transactions | Error rate | In last 1 hour | > | 10% | True | False |

### Working with Rule Violations

The Rule violations page provides details of all the violations. Use this page to search for specific violations and to view all the violations within the various groups for a specified duration. The following table provides you details of all the actions possible in the Rule violations page:

| Action | Description |
|---|---|
| Show all | Use this drop-down list to filter all the violations on the basis of duration or status. The values include:<br>■ **All open**<br>■ **All closed**<br>■ **Last 1 hour**<br>■ **Last 12 hours**<br>■ **Today**<br>■ **Yesterday**<br>■ **Last 7 days** |
| ⟳ | This is the refresh button. Click this to refresh the list of rule violations. It shows the time when the list was last refreshed. |
| ⚙ | This is the column view settings button. Click this to customize the columns you want to view for the rule violations. |

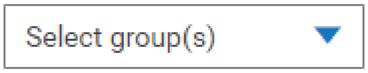
| Action | Description |
|---|---|
|  Select group(s) ▼ | Select this drop-down list to filter the rule violations on the basis of group name. You also get a search box when you click this drop-down list. Use this search box to type a group name that you want to search.<br>**Note:**<br>You can select more than one group. |
| Search by name | Type a rule name in this search box to filter all the rule violations on the basis of rule name. |

To sort the rule violations, click on the header row of any column. You can sort in both ascending and descending order.

### Working with Rule List

The application lists all the rules that you create in the Rule list page. The following table provides you details of all the actions possible in the Rule violations page:

| Action | Description |
|---|---|
| **Create rule** | Use this action button to create a new rule. For more information, see "Creating a Rule" on page 720. |
| ✎ | To edit a rule, click this action button on the row of the corresponding rule.<br>**Note:**<br>You cannot modify the rule name. |
| 🗑 | To delete a rule, click this action button on the row of the corresponding rule. |
| ─● | Select this toggle button in the **Status** column to enable or disable a rule. ─● indicates that a rule is enabled and ●─ indicates that a rule is disabled.<br>**Note:**<br>When you use this button, you will see corresponding system messages on screen to indicate if a rule is enabled or disabled successfully. |
| ⚙ | This is the column view settings button. Click this to customize the columns you want to view for the listed rules. In the Show Columns dialog, select the columns you want to view and click **Save**. |

| Action | Description |
|---|---|
|  | Select this drop-down list to filter the rules on the basis of group name. You also get a search box when you click this drop-down list. Use this search box to type a group name that you want to search.<br>**Note:**<br>You can select more than one group. |
| **Search by name** | Type a rule name in this search box to filter the rules on the basis of rule name. |

To sort the rules, click on the header row of any column. You can sort in both ascending and descending order.

## Business Transaction Details Page

The business transaction details page provides the following information about the transaction:

| Parameter | Description |
|---|---|
| Name | Transaction name. |
| Trace ID | Distinct identifier for the transaction generated internally by the application. |
| Status | Status of the transaction. |
| Start period | Time at which the transaction started. |
| End period | Time at which the transaction ended. |
| Duration (ms) | Total time taken by the transaction to execute. This includes the time taken by the transaction to process within each component and any network latency to or from the client. For more information, see “ Duration” on page 725 |

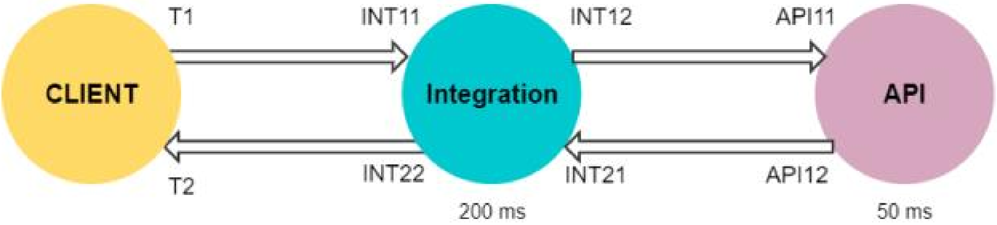
### Duration

Duration is the total time taken by a transaction from the time a client initiates a request to the time a response is received by the client.

Let us consider the following scenario where the duration for a transaction to complete is *200ms*. In the following image, *200ms* is the time difference between INT22 and INT11. The time taken by the transaction within the API component which is *50ms* is already taken into account as part of this *200ms* value. End to End Monitoring records the duration only from the time a client request enters the first component of the cloud platform (INT11) and the time the response exits the last component of the cloud platform (INT22).

**Note:**

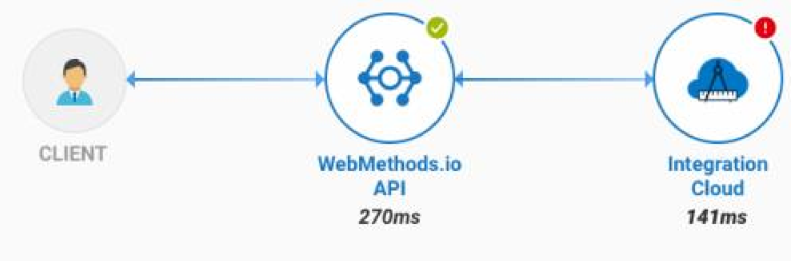
The application does not note the time the client sends the request (T1) and the time the client receives the response (T2), as this happens outside the Software AG environment.



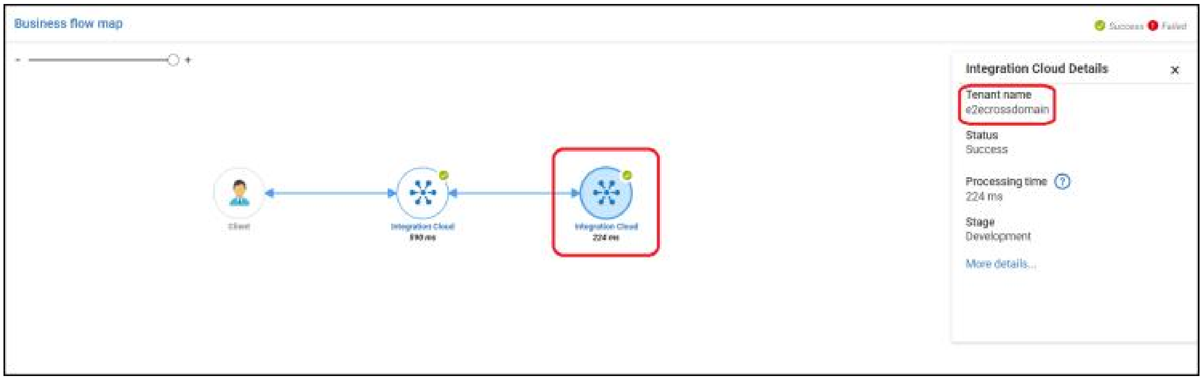
## Business Flow Map

This provides a logical representation of the business flow showing the path taken by the transaction through the various cloud components. Also, the processing time for the transaction within each component is visible.

A legend is available to identify the status of a transaction. An example of a business flow is as follows:



The Business Flow Map also shows details when there is a cross-domain transaction. A cross-domain transaction is one where the client makes a call to a supported component in one tenant and then the call moves to a component in another tenant. For example, let us consider the following image. Here, a client calls an Integration Cloud instance in one tenant and then the transaction moves to call another Integration Cloud instance in a different tenant. When you select the component, you can view the corresponding tenant name in the component details pane. In the following example, the second Integration Cloud instance is highlighted along with its tenant name.



Integration Cloud Details

Tenant name
e2ecrossdomain

Status
Success

Processing time
224 ms

Stage
Development

More details...

## Viewing the Transaction Details

### To view the transaction details

1. On the Application dashboard, select the transaction group for which you want to view the details.

2. On the Business Transaction Details page, click the component from the business flow map for which you want to view the details.

   The details open in a separate pane and includes the following information:

| Field | Description |
|---|---|
| Tenant name | Provides the name of the tenant where the transaction happened. |
| Status | Provides the status of the transaction. Status can be:<br>■ Success<br>■ Failed |
| Cause of failure | Provides the reason for the error to occur. If a transaction is successful, then this parameter is not visible. |
| Processing time | Time spent by the transaction within this component. |
| Stage | Phase at which the failure has occurred during the execution of a transaction. Stage represents an activity in the life cycle of the transaction. Possible stages are:<br>■ Development<br>■ Live<br>■ Test<br>■ Pre-Live<br>**Note:**<br>Stage is visible only if you are viewing the details for Integration Cloud. |
| More details... | Click this URL to open the details of the API or the integration that is associated with the business transaction. For an Integration, you are redirected to the instance of webMethods Integration Cloud in a new tab and for an API, you are redirected to the instance of webMethods API Gateway in a new tab. The More Details link also show the product-specific details for the products involved in a cross-product transaction even if the tenant for that product was configured in a different domain or a custom domain. |

## User Journey

### Summary

This section provides details about the journey of a user using End to End Monitoring. It is used to view the end-to-end details of a business transaction as it moves through the various webMethods applications.

### Actors

As part of this user journey, we have two actors here who play their respective roles:

- Jim - A Software AG customer who owns an enterprise called **XYSsales** and sells many products on his business platform. To run his business better, he uses products like webMethods API Gateway and webMethods Integration Cloud. **XYSsales** has a large customer base where every day new customers create their accounts on his business platform.
- Hayley - A prospective customer of Jim. She wants to create an account in **XYSsales**.

### Challenge

Jim reviews many business transactions in a day. These transactions move across the various Software AG applications. Reviewing the transactions helps Jim identify areas of improvement and any errors. In this scenario, XYSsales uses both webMethods API Gateway and webMethods Integration Cloud. Jim can view the details of the transactions within the individual systems. For example, on an API execution, he can view its details using the logs in API Gateway. In the same manner, when an Integration is run, he can view its details using the logs in webMethods Integration Cloud. However, if an API calls an Integration in webMethods Integration Cloud, the log information does not tell Jim which Integration in webMethods Integration Cloud corresponds to which API execution in webMethods API Gateway.

To summarize, Jim is unable to correlate the information when a business transaction traverses through multiple systems. He is unable to view the end to end details of a business transaction. We are helping Jim solve this problem by introducing End to End Monitoring.

Hayley sends a request to create a user account using a client application. This triggers a business transaction. Jim wants to view the complete flow of this business transaction. He wants to know the details from the time Hayley sent the request to the time she received the response.

### Preconditions

As part of the prerequisites, Jim performs the following actions:

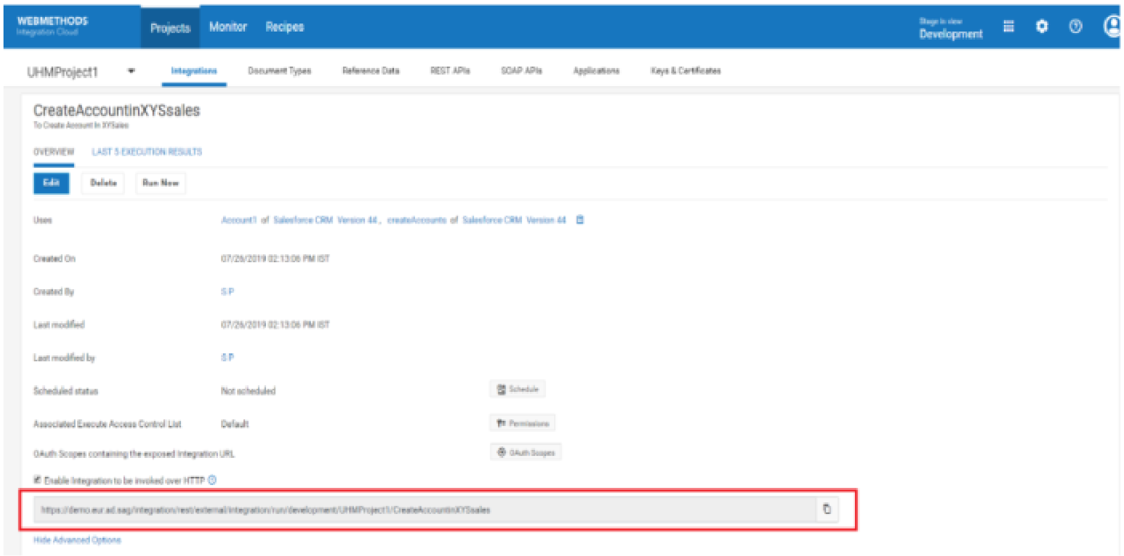
- Creates a connection from **XYSsales** to webMethods Integration Cloud by creating an account in webMethods Integration Cloud.
- Creates an Integration called **CreateAccountinXYSsales** in webMethods Integration Cloud. He exposes this Integration as a REST endpoint such that any client can access it.

- Creates an API called APItoCreateAccountinXYSsales in webMethods API Gateway. This allows him to make a connection with the REST endpoint created in the earlier step.

## Basic Flow

When Hayley sends a request to create a user account in XYSsales using the client application, following events take place:

1. The client application calls the REST API APItoCreateAccountinXYSsales. The system applies all the policies defined for this API at this stage of the transaction.
2. After successful authentication of Hayley's account credentials with webMethods API Gateway, the API invokes the Integration **CreateAccountinXYSsales** which is the REST endpoint.



CreateAccountinXYSsales

To Create Account In XYSales

Uses: Account1 of Salesforce CRM Version 44 , createAccounts of Salesforce CRM Version 44

Created On: 07/26/2019 02:13:06 PM IST

Created By: S P

Last modified: 07/26/2019 02:13:06 PM IST

Last modified by: S P

Scheduled status: Not scheduled

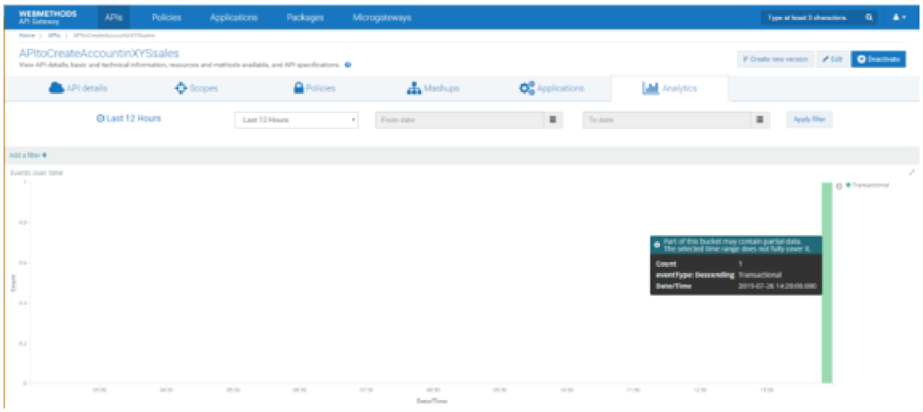
Associated Execute Access Control List: Default

OAuth Scopes containing the exposed Integration URL

☑ Enable Integration to be invoked over HTTP

https://demo.eur.ad.sag/integration/rest/external/integration/run/development/UHMProject1/CreateAccountinXYSsales

3. The Integration is run and it creates the user account for Hayley in XYSsales. On successful creation of the account, the system sends a confirmation message as a response to Hayley through the client application.
4. Jim can view the details of the API execution through the **Analytics** tab of the webMethods API Gateway cloud instance as shown here:



APItoCreateAccountinXYSsales

View API details, basic and technical information, resources and methods available, and API specifications.

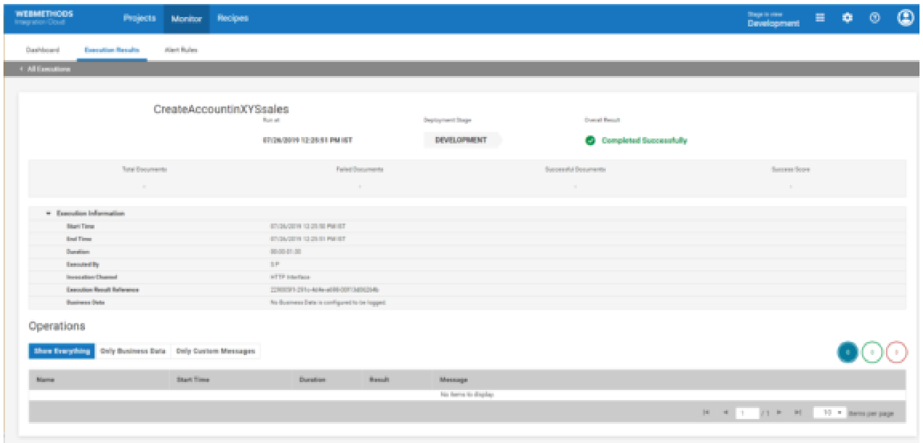
Part of this bucket may contain partial data. The selected time range does not fully cover it.

Count: 1

eventType: Descending: Transactional

Date/Time: 2019-07-26 14:20:08.000

5. Jim can also view the details of the integration using the **Monitor** tab of the webMethods Integration Cloud instance as shown here:



CreateAccountinXYSsales

Run at: 07/26/2019 12:25:51 PM IST

Deployment Stage: DEVELOPMENT

Overall Result: Completed Successfully

Total Documents | Failed Documents | Successful Documents | Success Score

Execution Information

| | |
|---|---|
| Start Time | 07/26/2019 12:25:50 PM IST |
| End Time | 07/26/2019 12:25:51 PM IST |
| Duration | 00:00:01:30 |
| Executed By | S P |
| Invocation Channel | HTTP Interface |
| Execution Result Reference | [illegible] |
| Business Data | No Business Data is configured to be logged |

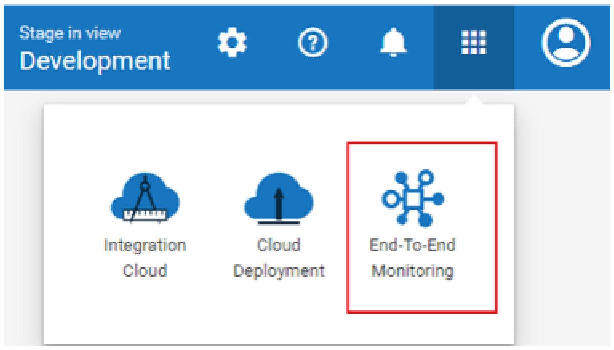
Operations

| Name | Start Time | Duration | Result | Message |
|---|---|---|---|---|
| | | | | No items to display |

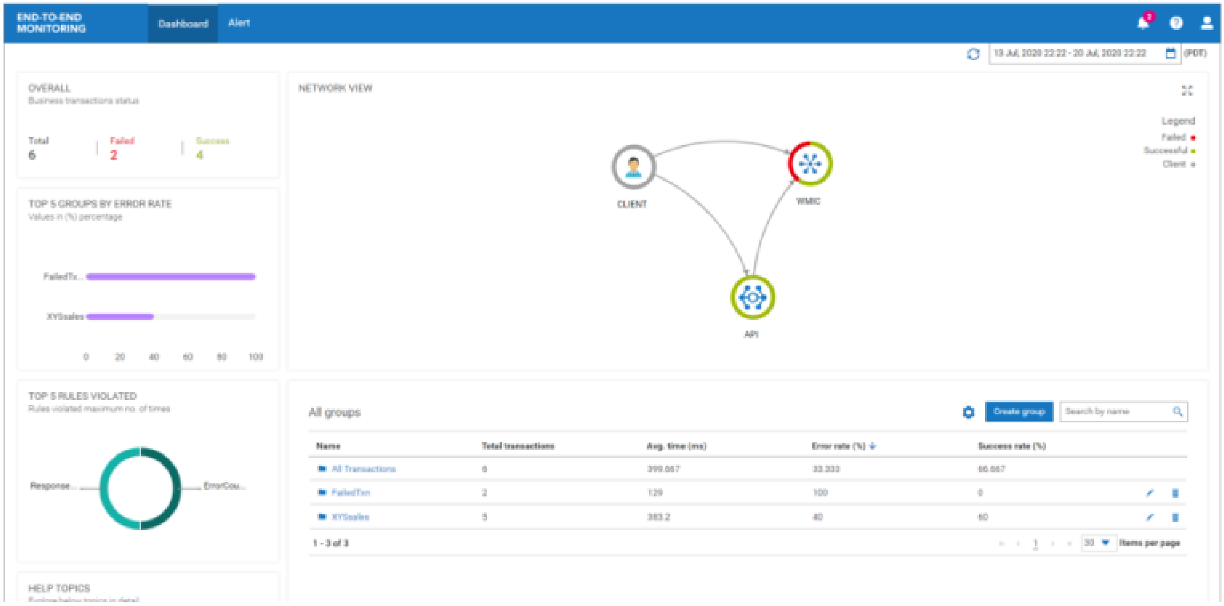
## How does Jim Monitor the Transactions?

1. Jim logs into webMethods Integration Cloud.

2. He selects **End to End Monitoring** from the App Switcher as shown here:



3. Jim opens the Dashboard of End to End Monitoring to view all the transaction groups. By default, the duration selected for view is **Last 30 mins** . Jim views the following details:



OVERALL
Business transactions status

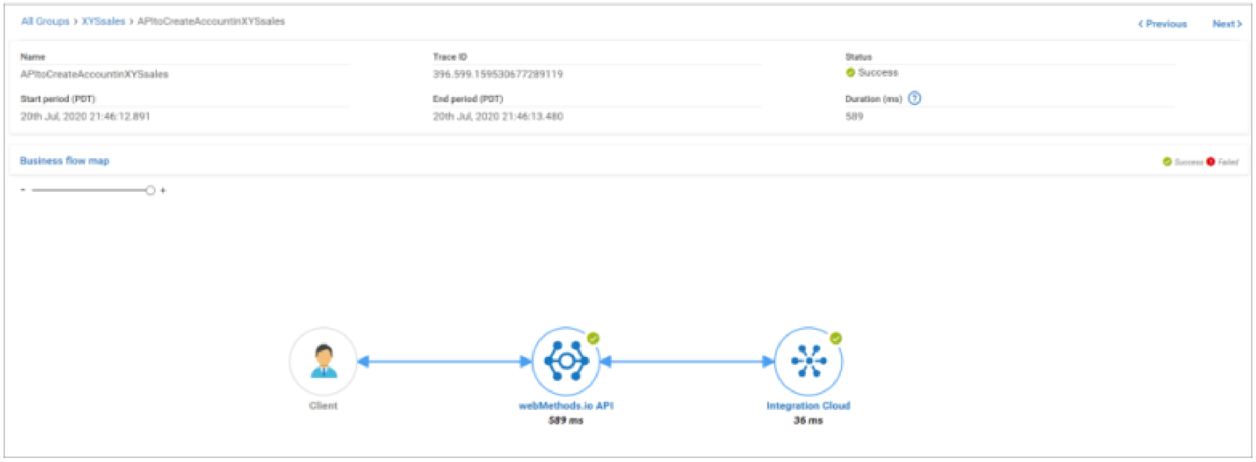
| Total | Failed | Success |
|---|---|---|
| 6 | 2 | 4 |

All groups

| Name | Total transactions | Avg. time (ms) | Error rate (%) | Success rate (%) |
|---|---|---|---|---|
| All Transactions | 6 | 399.667 | 33.333 | 66.667 |
| FailedTxn | 2 | 129 | 100 | 0 |
| XYSsales | 5 | 383.2 | 40 | 60 |

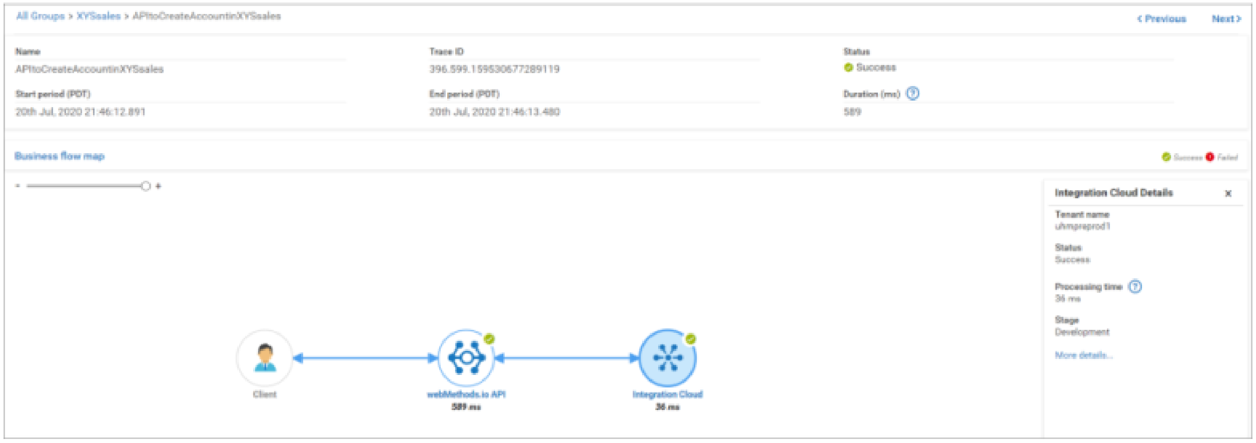
1 - 3 of 3   1   30 Items per page

4. He selects the group **XYSsales** which lists all the transactions associated with his enterprise.
5. He selects the transaction which he wants to view.
6. On the business transaction details page, he can view all the information for that business transaction as shown here:



| Name | Trace ID | Status |
|---|---|---|
| APItoCreateAccountinXYSsales | 396.599.1595306677289119 | Success |
| **Start period (PDT)** | **End period (PDT)** | **Duration (ms)** |
| 20th Jul, 2020 21:46:12.891 | 20th Jul, 2020 21:46:13.480 | 589 |

7. On the business transaction details page, he can get more details for each component of the transaction by selecting the component from the **Business flow map**. When he selects the component, the details are shown in a new pane. For example, Jim selects the **Integration Cloud** component and he can view the details as shown in the following image:



All Groups > XYSsales > APItoCreateAccountinXYSsales

‹ Previous Next ›

| Name | Trace ID | Status |
|---|---|---|
| APItoCreateAccountinXYSsales | 396.599.1595306772891119 | Success |
| **Start period (PDT)** | **End period (PDT)** | **Duration (ms)** |
| 20th Jul, 2020 21:46:12.891 | 20th Jul, 2020 21:46:13.480 | 589 |

Business flow map

Integration Cloud Details

Tenant name: uhmpreprod1

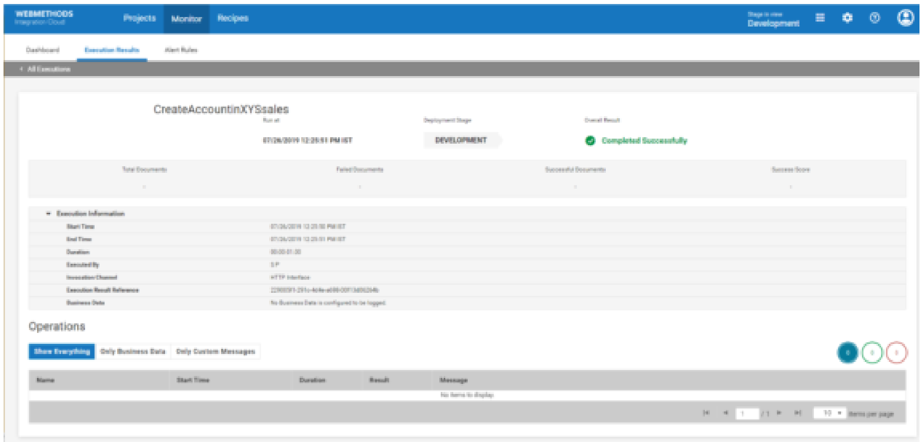
Status: Success

Processing time: 36 ms

Stage: Development

More details...

8. Jim selects the **More details..** URL to open the webMethods Integration Cloud instance. Here, he finds information about the integration that was executed as part of this business transaction as shown in the following image:



CreateAccountinXYSsales

| Run at | Deployment Stage | Overall Result |
|---|---|---|
| 07/26/2019 12:25:51 PM IST | DEVELOPMENT | Completed Successfully |

| Total Documents | Failed Documents | Successful Documents | Success Score |
|---|---|---|---|

Execution Information

| | |
|---|---|
| Start Time | 07/26/2019 12:25:50 PM IST |
| End Time | 07/26/2019 12:25:51 PM IST |
| Duration | 00:00:01:30 |
| Executed By | S P |
| Invocation Channel | HTTP Interface |
| Execution Result Reference | 22900591-291c-4d4e-a08-00f13d0264b |
| Business Data | No Business Data is configured to be logged |

Operations

| Name | Start Time | Duration | Result | Message |
|---|---|---|---|---|
| | | | | No items to display |

9. This completes the user journey for Jim.

# 20 Switch to Cloud Deployment

# Overview

**The Cloud Deployment section is applicable only if you are an existing tenant and have already provisioned Cloud Deployment.**

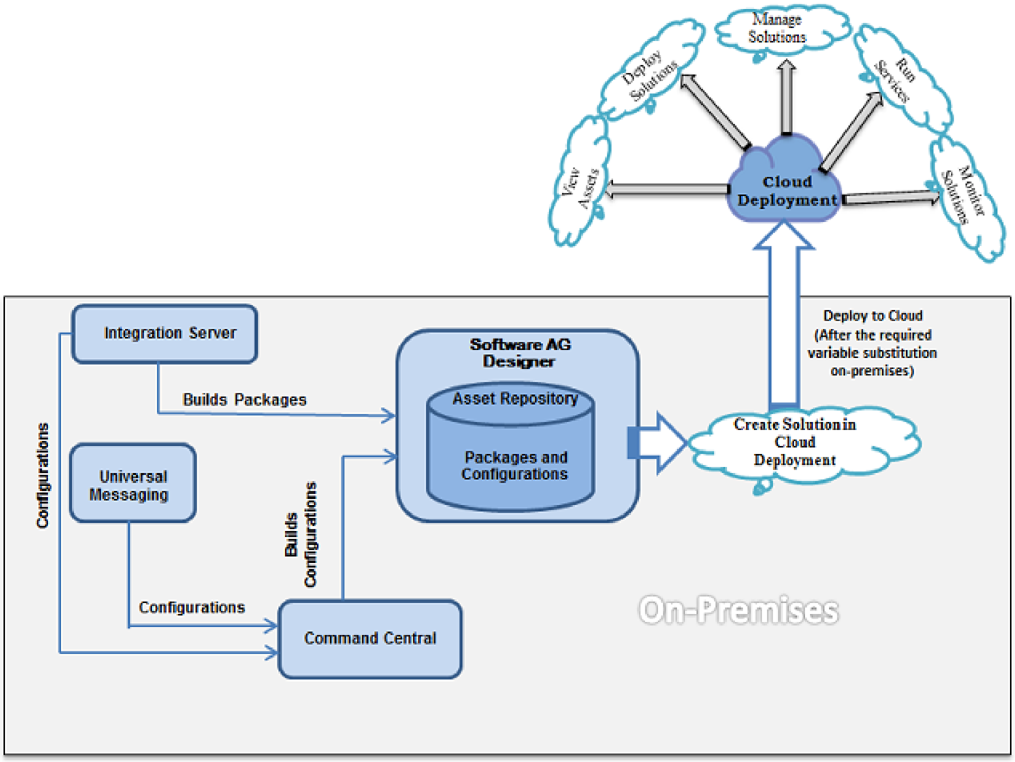
**Note:**
See the article on Software AG TECHcommunity website to get started on Cloud Deployment.

A **Solution** consists of packages bundled together into one coherent service. It is a logical combination of webMethods Integration Server packages, Adapter packages, Services, webMethods CloudStreams packages, and webMethods Integration Server and Universal Messaging configuration assets or configurations.

**Note:**
After deploying the configurations, due to webMethods Integration Server restart, the assets will appear after a short delay.

Integration Cloud Integrations can invoke Cloud Deployment webMethods Integration Server services for the same tenant. Using the predefined *Cloud Deployment Application* available in Integration Cloud, you can select the solution webMethods Integration Server services that you want to invoke from Integration Cloud.

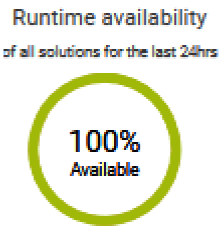
The following figure provides a high-level overview of the process involved in deploying on-premises webMethods Integration Server packages and configuration assets to Integration Cloud.

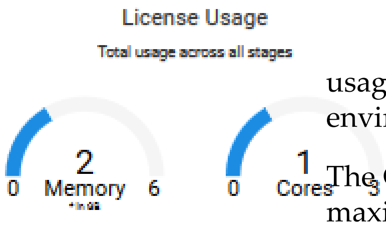
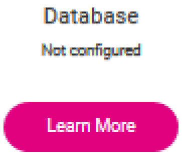
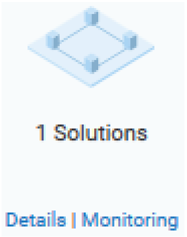
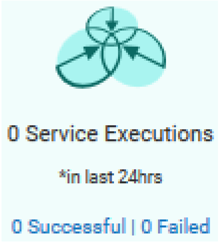


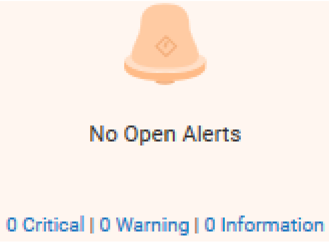
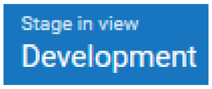
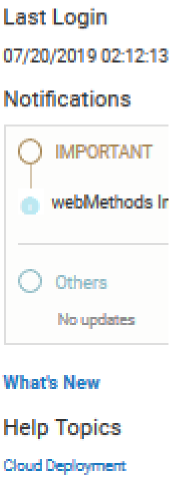
With Software AG Designer, you can deploy the webMethods Integration Server packages or configuration assets that you have created, verified, and tested on on-premises webMethods Integration Server or Universal Messaging to Integration Cloud. When you initiate the deployment from Software AG Designer, webMethods Integration Server packages and configuration assets are built from webMethods Integration Server and Software AG Command Central respectively, and are published to the Asset Repository available in Software AG Designer. After performing variable substitutions to make the on-premises configuration data compatible, you can publish the packages and configurations to an asset repository provisioned for the tenant on Integration Cloud.

### User Interface elements

The following table describes the various user interface elements that appear on the **Cloud Deployment** workspace:

| Page Elements | Icon | Description |
|---|---|---|
| webMethods Cloud Deployment | WEBMETHODS<br>Cloud Deployment | Access the home page. |
| Solutions | Solutions | View and create solutions, deploy solutions to environments, view the Asset Repository, and manage webMethods Integration Server instances. |
| Monitoring | Monitoring | View the overall status of the solutions, landscapes, system and runtime alerts, KPI graphs of the runtimes, service executions of the webMethods Integration Server instances, availability of the run times, alert status of the solutions, and logs. |
| Runtime availability |  Runtime availability<br>of all solutions for the last 24hrs<br>100%<br>Available | View the overall run-time availability for all the solutions in the last 24 hours. The doughnut chart displays the periods of availability (green color), unavailability (red color), and maintenance (yellow color). |

| **Page Elements** | **Icon** | **Description** |
|---|---|---|
| Usage Statistics | <br>License Usage<br>Total usage across all stages<br>0 2 Memory 6 *in GB<br>0 1 Cores 3 | sage statistics shows PU and Memory usage for all solutions in environments.<br>The CPU bar shows the maximum CPU that is licensed for the tenant. The colored part of the bar shows how much of the allowed CPU is currently used by the tenant.<br>The Memory bar shows the memory in Gb that is licensed for the tenant. The colored part of the bar shows how much of the allowed memory is currently used by the tenant. |
| Database | <br>Database<br>Not configured<br>Learn More | Add a database to your subscription. This enables you to configure, store, and monitor your database directly in the cloud instead of using external systems. |
| Solutions section | <br>1 Solutions<br>Details \| Monitoring | Total number of solutions created. View the *Solution List* page by clicking the Details link and the *Monitoring Dashboard* by clicking the Monitoring link. |
| Service Executions section | <br>0 Service Executions<br>*in last 24hrs<br>0 Successful \| 0 Failed | Number of service executions and their status in the last 24 hours for all the solutions that are currently available in the selected *Default Environment*. To view the Services page under Monitoring, click the service execution link. |

| Page Elements | Icon | Description |
|---|---|---|
| Active Alerts section | <br>No Open Alerts<br>0 Critical \| 0 Warning \| 0 Information | Number of currently open alerts and their severity. Click on the links to go to the *Alerts* page. |
| Default Environment | <br>Stage in view<br>Development | Click to view and link environments. |
| Help | ? | Access Help topics, Software AG TECHcommunity, Licensing capabilities, and the About page. The About page displays the version information, Global Support information, Copyright information, Impressum and Privacy Policy, and the release readme. |
| Profile | 👤 | Name of the logged in user, profile information of the logged in user, and Logout option. |
| Last Login, Notifications, and Help topics | <br>Last Login<br>07/20/2019 02:12:13<br>Notifications<br>IMPORTANT<br>webMethods Ir<br>Others<br>No updates<br>What's New<br>Help Topics<br>Cloud Deployment | When was the last login, important and other notifications, what is new in this release, and context-sensitive help topics. |

# Solutions

A **Solution** consists of packages bundled together into one coherent service. It is a logical combination of webMethods Integration Server packages, Adapter packages, Services, webMethods CloudStreams packages, and webMethods Integration Server and Universal Messaging configuration assets or configurations.

**Note:**
After deploying the configurations, due to webMethods Integration Server restart, the assets will appear after a short delay.

Integration Cloud Integrations can invoke Cloud Deployment webMethods Integration Server services for the same tenant. Using the predefined *Cloud Deployment Application* available in Integration Cloud, you can select the solution webMethods Integration Server services that you want to invoke from Integration Cloud.

## Creating Solutions

The **Solution List** page displays the solutions created in Integration Cloud.

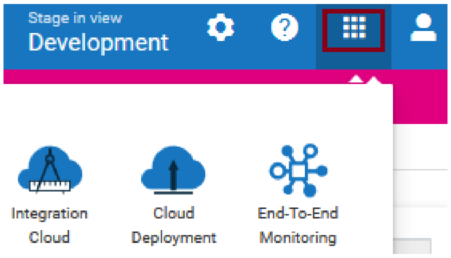
**Note:**
You must have the required permissions under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Solution** to create, update, or delete Solutions.

**Note:**
You must first create a solution in Integration Cloud before deploying the on-premises assets and configurations.

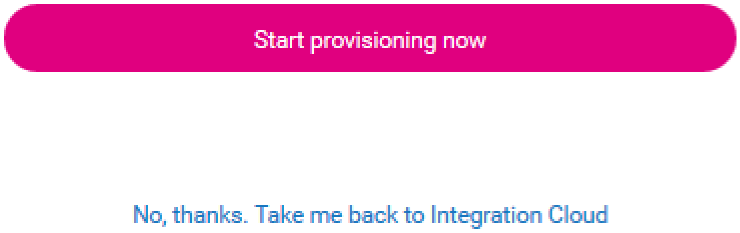
All solutions created are initially copied to the Development stage. You can create a new solution in any stage, for example, development, test, live, and prelive. Further, you cannot modify the solution name and clustering details after it is created. You can configure the solution in subsequent stages but after you configure a solution in a stage, you cannot modify the solution again in that stage. In a stage, you can configure only those solutions that are marked as **Not Configured**.

### To create a solution

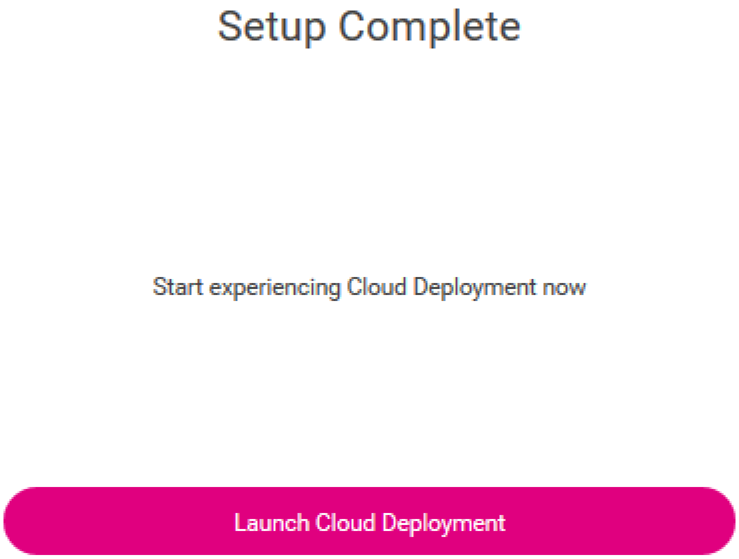
1. Switch to the **Cloud Deployment** perspective.



As soon as you register, Cloud Deployment capability is enabled by default for all tenants. By default, 3 CPU core and 6 GB memory are allocated for all tenants. When you access Cloud Deployment for the first time using the application launcher, you need to start provisioning.



After provisioning, your setup will be complete, and you can launch Cloud Deployment.



Setup Complete

Start experiencing Cloud Deployment now

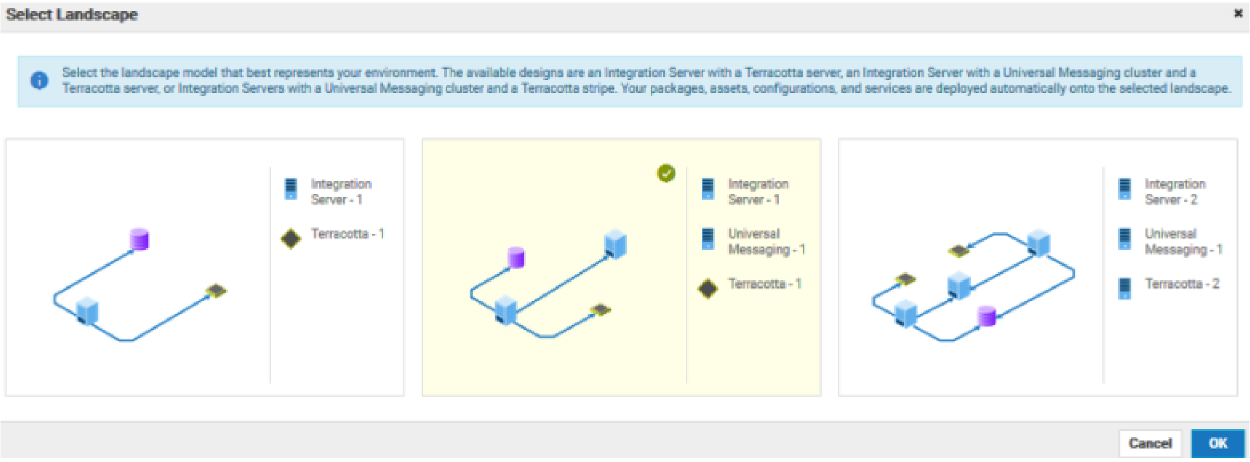
**Note:**
Solutions created using a trial account are deactivated daily. After you log in, you need to reactivate the solutions. All assets will be available after a short delay.

2. After you launch Cloud Deployment, from the **Cloud Deployment** navigation bar, click **Solutions > Solution List**.

   The **Solution List** page appears.

3. From the **Solution List** page, click **Create New Solution** to create a solution. You can create a solution in any stage.

4. Select the landscape model that best represents your environment. The available designs are an webMethods Integration Server with a Terracotta server, an webMethods Integration Server with Universal Messaging and a Terracotta server, or webMethods Integration Servers with Universal Messaging and a Terracotta stripe. Your packages, assets, configurations, and services are deployed automatically onto the selected landscape.



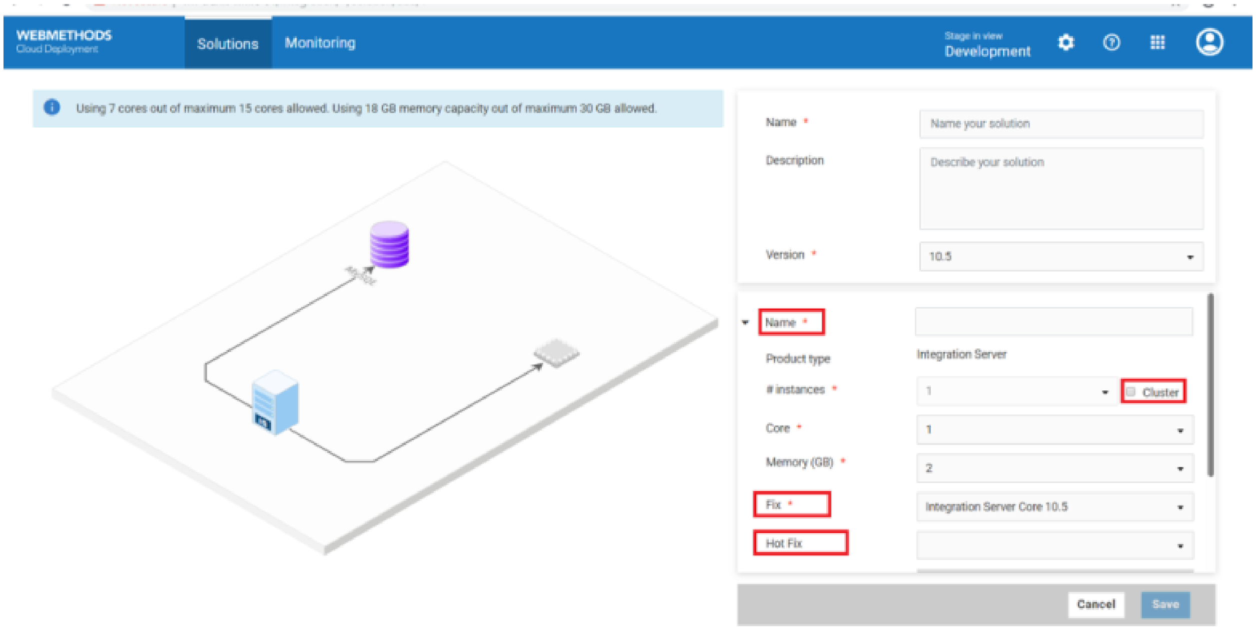
Select Landscape

Select the landscape model that best represents your environment. The available designs are an Integration Server with a Terracotta server, an Integration Server with a Universal Messaging cluster and a Terracotta server, or Integration Servers with a Universal Messaging cluster and a Terracotta stripe. Your packages, assets, configurations, and services are deployed automatically onto the selected landscape.

5. Click **OK**.

6. In the **New Solution** page, fill in the solution **Name**, the solution **Description**, the name of the webMethods Integration Server and Universal Messaging instances, number of CPU **Cores**, and **Memory** characteristics of the hardware to support each service in the solution landscape.

> **Note:**
> You can also select the webMethods Integration Server and Universal Messaging icons to highlight the sections in the **New Solution** page. Terracotta is available only when webMethods Integration Server runs in a clustered mode. Further, clustering and Terracotta options are not available in the **Development** stage.



Using 7 cores out of maximum 15 cores allowed. Using 18 GB memory capacity out of maximum 30 GB allowed.

Name * Name your solution

Description Describe your solution

Version * 10.5

Name *

Product type Integration Server

# instances * 1 □ Cluster

Core * 1

Memory (GB) * 2

Fix * Integration Server Core 10.5

Hot Fix

The **Version** list box lists all the available major versions. The solution will be created based on the available **Fix** version. webMethods Integration Server packages such as WmCloudStreams and WmJDBCAdapter will appear in the **Packages** group box based on the **Fix** version. Select the **Packages** you want to enable. After you save the solution, you will not be able to change the major **Version**. You can change only the number of CPU Cores, Memory, Fix version if available, and Packages.

You can deploy CloudStreams provider packages, CloudStreams connector services, CloudStreams connection, and CloudStreams connector listeners to a solution in Cloud Deployment only if you have selected WmCloudStreams as the package option while creating the solution.

On the **Landscape** page, select the **Cluster Type** as **Stateless** if the group of webMethods Integration Servers function in a manner similar to a cluster but are not part of a configured cluster. A stateless cluster of webMethods Integration Servers does not use a Terracotta Server Array. Select **Stateful** to add the Terracotta section. The Terracotta icons will be activated.

The **Hot Fix** list box lists all available fixes that include enhancements to versions. **Hot fix** field is optional. Selecting **None** option will remove the hotfixes applied on the solution, and the solution will point to the base fix version. Typically, hotfixes are created for specific tenants.

**Note:**
After creating a solution, if you modify the package list options in the **Packages** group box and save the solution, due to webMethods Integration Server restart, the assets will appear after a short delay.

**Updating products to a higher fix version in a solution**

You can update any product in a solution to the available higher fix version after you create the solution. The **Update Available** option appears if a higher fix version is available for any of the products in the solution. The latest fix version appears in the **Fix** drop-down list. For example, if you have selected Fix 3 of webMethods Integration Server while creating a solution, and if Fix 4 is now available, you can select Fix 4 and save the solution.

**Note:**
While configuring a solution in a higher stage, the fix version of the products in the solution will be taken from the lower stage. If a solution is updated in a lower stage, you will be given an option to update the solution in the higher stage, to the fix versions of the products in the lower stage.

For example, if you configure a solution in a higher stage (Test stage) and then go back to a lower stage (Development stage), and If a higher fix version is available for a product in the Development stage, you can click **Update Available**, and in the **Edit Solution** screen, you can select the higher fix version for the product in the solution. Now if you go to the higher stage (Test), the **Update Available** option will appear in the Test stage. You can update products in a solution at every stage, to the fix versions of the products in the solution available in the immediate lower stage. Note that after you update or update the products in the solution, the products in the solution will not be accessible for sometime.

**Updating products to a higher version in a solution**

You can update any product in a solution to the available higher version after you create the solution. The **Update Available** option appears if a higher version is available for any of the products in the solution. The latest version appears in the **Version** drop-down list. For example, if you have selected 10.3 of webMethods Integration Server while creating a solution, and if

10.5 is now available, you can select version 10.5 and save the solution. Further, if you update to v10.5, the latest available fix for v10.5 will be automatically selected.

**Note:**
While configuring a solution in a higher stage, the version of the products in the solution will be taken from the lower stage.

If a higher version is available for a product in a stage, you can click **Update Available**, and in the **Edit Solution** screen, you can select the higher version for the product in the solution.

The **Schedule** option appears if you select a higher version available for any of the products in the solution. Click **Schedule** to schedule the update process by specifying the date and time on which you want the update process to execute. Once the solution is scheduled for update, you can click **Cancel Schedule** to cancel the schedule or click **Modify Schedule** to modify the schedule. If the update process is scheduled, the status of the solution on the **Solution List** page displays **Update scheduled**. The status of the solution on the **Solution List** page displays **Update in progress** if the scheduled update process is under way.

**Note:**
After a solution is updated successfully, ensure that you click **Confirm Update** within seven days to complete the update process or click **Rollback** to rollback the solution to the previous version. After seven days, except for the **Confirm Update** and **Rollback** options, the solution page will not be available. If you rollback, the status of the solution on the **Solution List** page displays **Rollback in progress** until the solution is rolled back to the previous version.

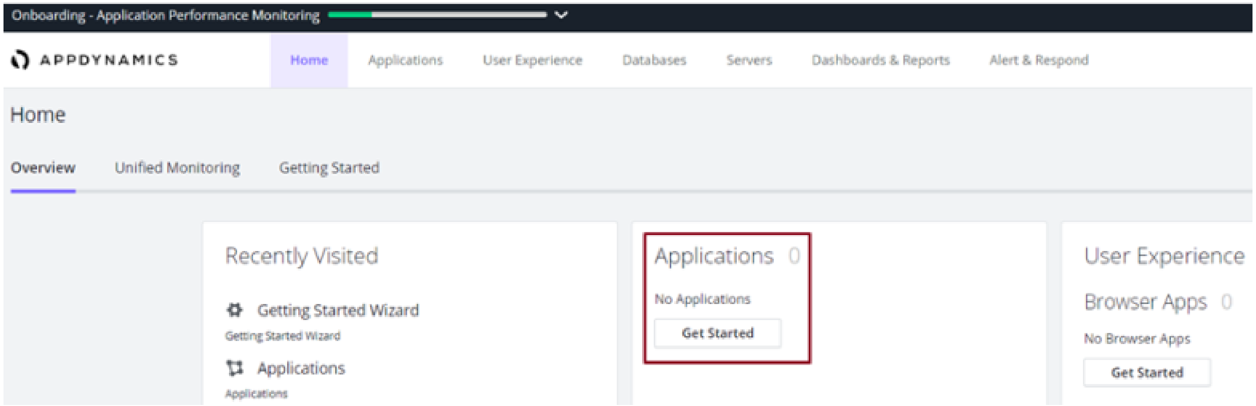
7. The **Tracing** list box appears only if you have the App Dynamics capability and if you are currently using AppDynamics to trace end to end business flows. It allows you to trace logs after you create or update a solution for an webMethods Integration Server runtime. Currently, tracing support is provided only for the webMethods Integration Server runtime. Select **AppDynamics** from the drop-down list to provide the AppDynamics tracing support and upload a valid Controller XML and Config XML file containing the Appdynamics details. You can update the controller file for each webMethods Integration Server runtime. You also have the option to download the controller and config files and then upload the files after modifying them.

**Note:**
If you select AppDynamics tracing and provide a valid controller file, the tracing data will appear on the AppDynamics cloud application.

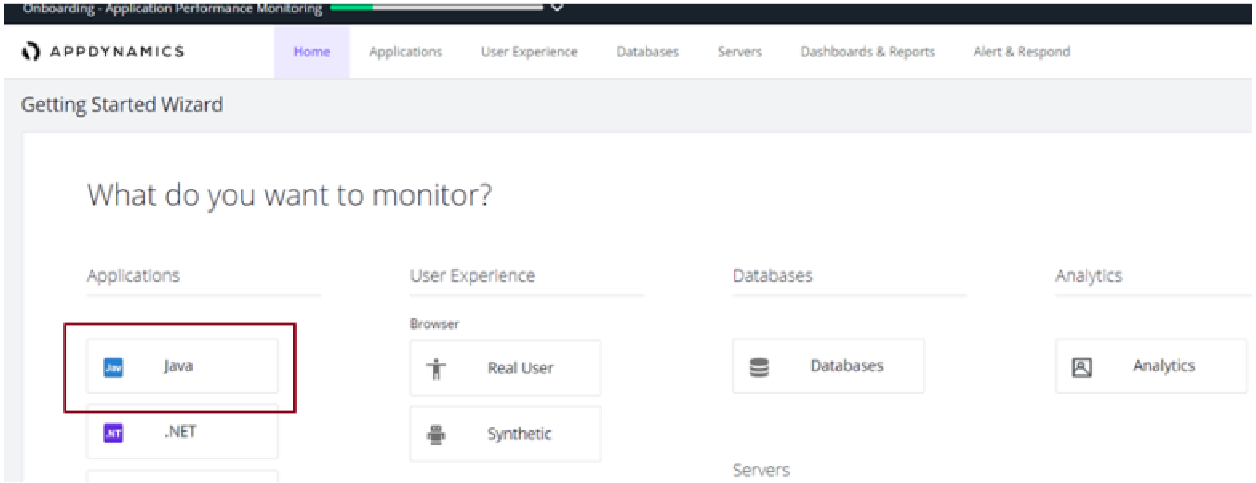
Let us see an example of how the tracing data appears on Appdynamics when you create a solution in Cloud Deployment.

a. Go to https://www.appdynamics.com/ and register in AppDynamics.

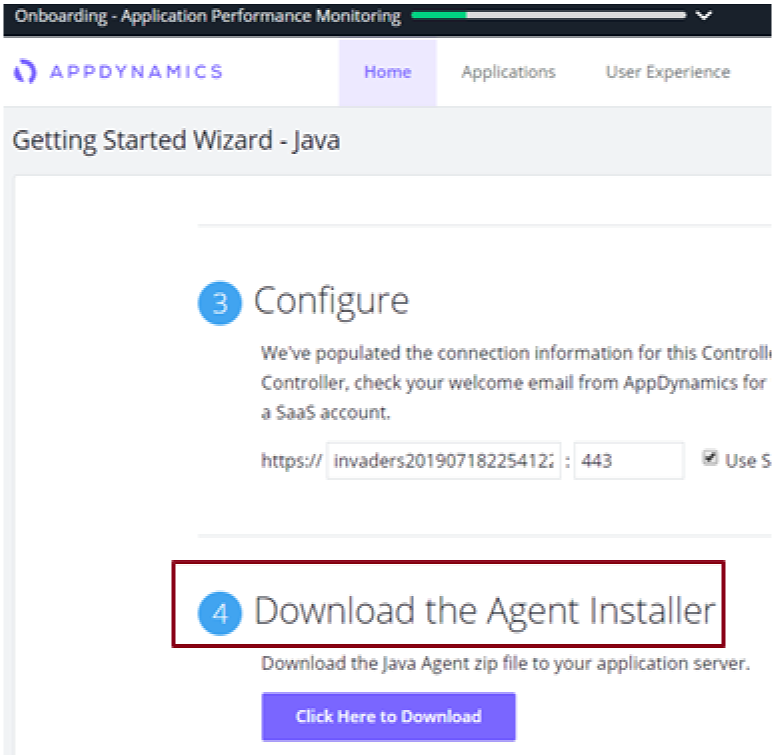
b. On the Overview > Applications panel, click **Get Started**.



c. On the Getting Started wizard, click **Java** under the Applications panel.



d. Download the Java Agent Installer.



Getting Started Wizard - Java

3 Configure

We've populated the connection information for this Controll
Controller, check your welcome email from AppDynamics for
a SaaS account.

https:// invaders20190718225412: : 443 ☑ Use S

4 Download the Agent Installer

Download the Java Agent zip file to your application server.

e. Go to the ver<xxx> > **conf** folder on your local system and view the controller-info.xml and app-agent-config.xml files.

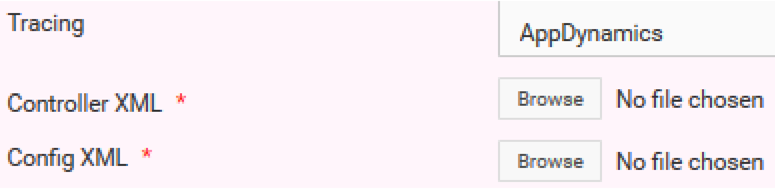
f. Open the controller-info.xml file and provide the following values:

  <application-name>MyApplication</application-name

  <node-name>AppIS</node-name>

g. Log in to Integration Cloud and switch to Cloud Deployment using the App switcher.

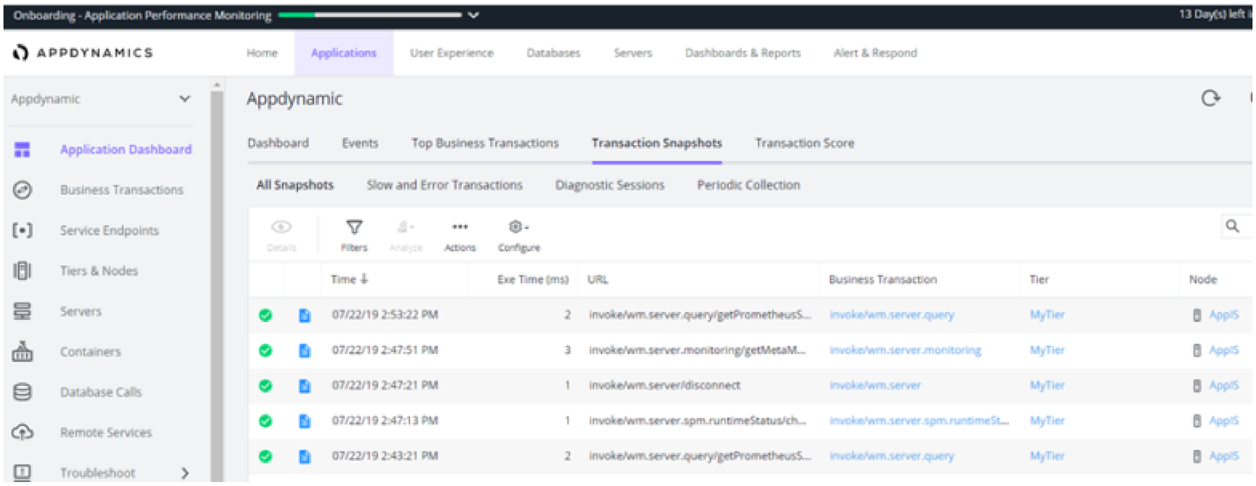
h. Create a new solution in Cloud Deployment and select **AppDynamics** from the drop-down list to provide the AppDynamics tracing support. Upload the controller-info.xml and app-agent-config.xml files, and save the solution.



| | |
|---|---|
| Tracing | AppDynamics |
| Controller XML * | Browse No file chosen |
| Config XML * | Browse No file chosen |

i. Log in to AppDynamics. Your application, that is, **MyApplication** appears on the Applications panel.

j. Click on MyApplication and view the service execution status and execution logs.



| | | Time ↓ | Exe Time (ms) | URL | Business Transaction | Tier | Node |
|---|---|---|---|---|---|---|---|
| ✓ | | 07/22/19 2:53:22 PM | 2 | invoke/wm.server.query/getPrometheusS... | invoke/wm.server.query | MyTier | AppIS |
| ✓ | | 07/22/19 2:47:51 PM | 3 | invoke/wm.server.monitoring/getMetaM... | invoke/wm.server.monitoring | MyTier | AppIS |
| ✓ | | 07/22/19 2:47:21 PM | 1 | invoke/wm.server/disconnect | invoke/wm.server | MyTier | AppIS |
| ✓ | | 07/22/19 2:47:13 PM | 1 | invoke/wm.server.spm.runtimeStatus/ch... | invoke/wm.server.spm.runtimeSt... | MyTier | AppIS |
| ✓ | | 07/22/19 2:43:21 PM | 2 | invoke/wm.server.query/getPrometheusS... | invoke/wm.server.query | MyTier | AppIS |

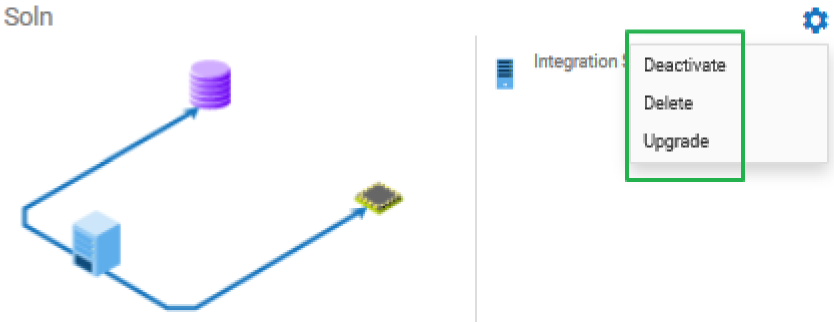
8. Click **Save** to save the solution.

   The new solution is created and appears in the **Solution List** page. You cannot modify the solution after the solution is created. You can configure the solution in subsequent stages but once a solution is configured in a stage, you cannot modify the solution again in that stage.

   **Note:**
   You can now deploy the solution to the next stage.

### Deactivate, Activate, and Delete a solution

Click the ⚙ icon and select **Deactivate** to deactivate a solution. All packages and assets will be permanently deleted and cannot be recovered. Click the ⚙ icon and select **Activate** to activate an inactive solution.



Click the ⚙ icon and select **Delete** to permanently delete the solution. If a solution is configured in a subsequent stage, it will be permanently deleted from the current stage and cannot be recovered. Further, if you delete the solution, you will not be able to promote assets from the current stage.

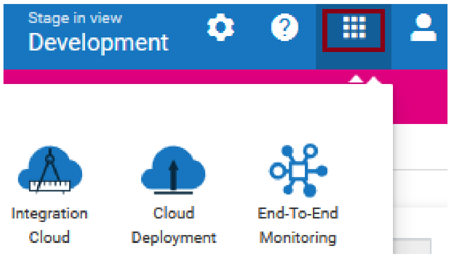
## Exploring Solutions

The solution details page allows you to view the packages, assets, configurations and services for different runtimes in the solution, deploy the solution to another stage, view the Asset Repository, and manage the solution, that is, view the landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances.

**Note:**
You can create a new solution in any stage. You can configure the solution in subsequent stages but after you configure the solution in a stage, you cannot modify it again in that stage.

### To view the Solution Explorer

1. **Switch to the Cloud Deployment perspective.**

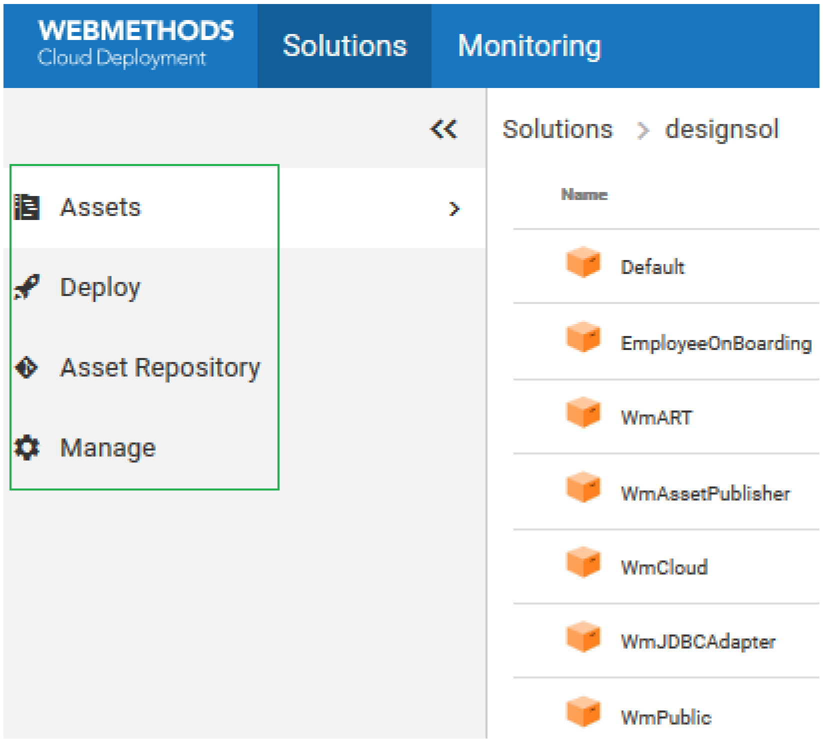


2. From the Cloud Deployment navigation bar, click **Solutions > Solution List**.

   The **Solution List** page appears listing all the solutions.

3. Click on an existing solution. The Solution Explorer page appears.

   **Solution Explorer**



The following table provides a high-level overview of the Solution Explorer page:

| Component | Description |
|---|---|
| "Assets" on page 747 | View the Packages, Folders, Assets, and Services for webMethods Integration Server, Adapters, webMethods CloudStreams packages, and configurations for the Universal Messaging runtime. |
| "Deploy" on page 757 | Deploy the solution to another stage. |
| "Asset Repository" on page 756 | View the Asset Repository which displays the contents of the on-premises packages published to Integration Cloud. |
| "Manage" on page 762 | View the landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances. |

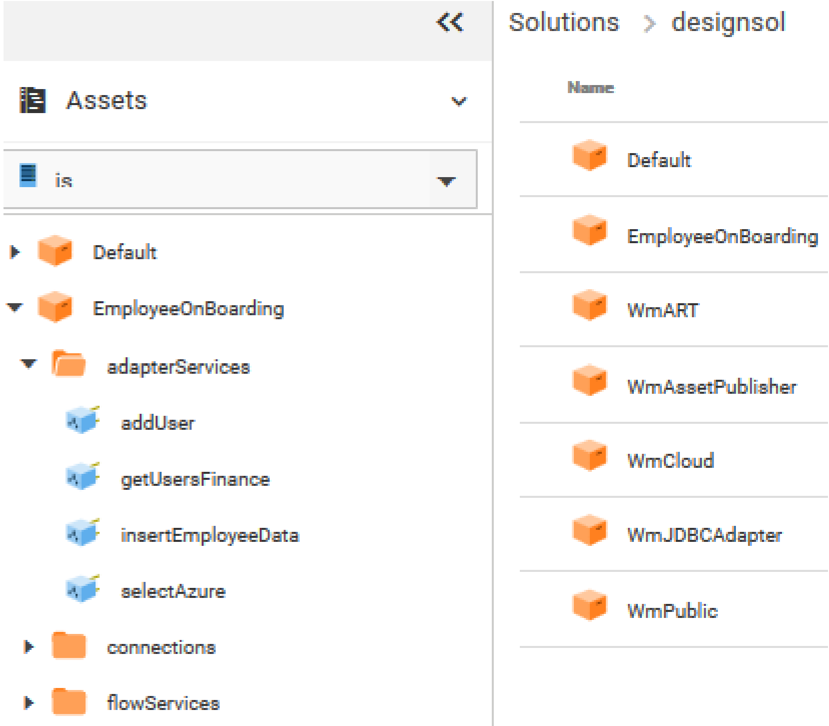
## Assets

A package is a container that is used to bundle services and related elements, such as specifications, webMethods Integration Server document types, webMethods Integration Server schemas, and triggers. When you create a folder, service, webMethods Integration Server document type, or any element, you save it in a package.

**Note:**
To view and access webMethods Integration Server packages in Integration Cloud, you must assign any custom user groups created in webMethods Integration Server, which are assigned to Access Profiles in the **Solution Permissions** page, to the following Access Control Lists in webMethods Integration Server:

- Administrators ACL
- Developers ACL
- Replicators ACL

The following figure depicts the package structure in a solution.



Packages are designed to hold all of the components of a logical unit in an integration solution. For example, you might group all the services and files specific to a particular marketplace in a single package. All the components that belong to a package reside in the package's subdirectory.

**Note:**
Click the arrow beside a folder or package to view its contents.

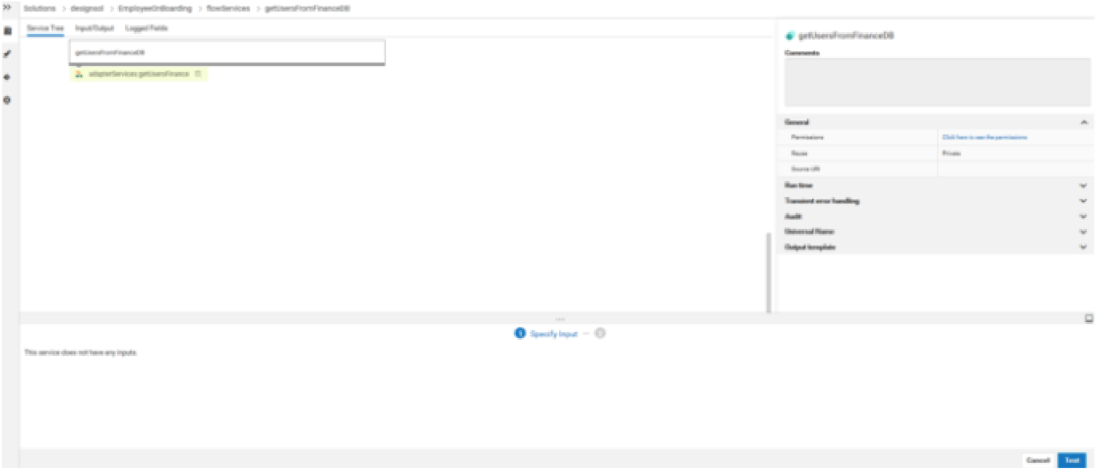
### Downloading assets

You can download user deployed packages and configurations from the **Asset** page. To download assets, point to an asset, click the ellipses ⋮ icon, and then click **Download**. The assets will be zipped and downloaded to your local storage space.

**Note:**
You cannot download default packages such as packages that come with webMethods Integration Server installation, for example, Default, WmART, WmCloud, WmJDBCAdapter, WmPublic, and so on.

### Services

Services are method-like units of logic that operate on documents. You build services to carry out work such as extracting data from documents, interacting with back-end resources (for example, submitting a query to a database or executing a transaction on a mainframe computer), and publishing documents. Adapters and other add-on packages provide additional services that you use to interact with specific resources or applications. The service editor allows you to view and run the services.



This service does not have any inputs.

### Service Signature

Input and output parameters are the names and types of fields that the service requires as input and generates as output. These parameters are also collectively referred to as a *signature*. You declare a signature for all types of services: flow services, Java services, and services written in other supported programming languages.

For a flow service, the input side describes the initial contents of the pipeline. In other words, it specifies the variables that this flow service expects to find in the pipeline at run time. The output side identifies the variables produced by the flow service and returned to the pipeline. An webMethods Integration Server document type can also be used to define the input or output parameters for a service.

Click **Test** to run the service after providing the data to pass into the service.

### Service Editor

Use the service editor to view the services. The source code, properties, inputs, and outputs are read only. The editor has the following tabs:

- **Source** tab contains the code or flow for the service.
- **Input/Output** tab contains the input and output signature of the service.
- **Logged Fields** tab indicates the input and output parameters for which the data is logged. You define the data to pass into the service by defining the input parameters on the lower panel of the editor.

### Load pipeline for testing services

The pipeline is the general term used to refer to the data structure in which input and output values are maintained for a service in Software AG Designer. The pipeline holds the input and output for a service. The pipeline starts with the input to the service and collects inputs and outputs from subsequent services. When a service runs, it has access to all data in the pipeline at that point.

When you run a service in Software AG Designer, you can click **Save** and save the pipeline data as an XML document to your local file system. After you deploy the service, you can click the **Load Data** option in the service editor to select the XML file, and load or update the pipeline data to test the service.

**Note:**
Integration Cloud Integrations can now invoke Cloud Deployment webMethods Integration Server services for the same tenant. A new pre-defined Application, Cloud Deployment, is added in Integration Cloud. Using this Application, you can select the solution webMethods Integration Server services that you want to call from Integration Cloud.

**Note:**
See the *webMethods Service Development Help*, *webMethods Integration Server Administrator's Guide*, and the *webMethods Adapter for JDBC Installation and User's Guide* for detailed descriptions of all the services and document types including Adapter services.

### Displaying the API details of an executable service

After deploying assets, on the Asset explorer page, click the *API Details* option to view the API details of the service such as the HTTP Method, URL, Input structure, and the parameters that are required to invoke this service from an external system, for example, a REST client. You can copy the required API details to execute the service from the external system.

The *API Details* option appears only for executable services. Some examples of executable services are Adapter services, Flow services, Java services, Flat File Schema, and Map services.

You will be able to execute a Flat File Schema by using *only* the SoapUI tool. You must specify the following settings in the SoapUI tool:

- Set the following query parameters in the **Request** section:

  skipWhiteSpace = true

  encoding = UTF8

  file = file:file1

- In the **Attachments** section, browse for the source file and update the following column values:

  Name = file1

  ContentID = file

## Support for GraphQL Assets

### Overview

GraphQL is a query language designed to build client applications by providing a flexible syntax and system for describing their data requirements and interactions. Using GraphQL service, you query a specific data to the server and get the response in a predictable way.

You can deploy GraphQL assets, which are developed using Software AG Designer and on-premises webMethods Integration Server, to Cloud Deployment. You choose from predefined solution landscapes to deploy your on-premises assets from Software AG Designer.

Let us see an example of how we deploy GraphQL assets which are developed using Software AG Designer and on-premises webMethods Integration Server to Cloud Deployment.

### Preconditions

### Create a GraphQL Schema

Create a GraphQL schema using SDL (Schema Definition Language). A sample schema is shown below:

```
type Query {
  dog: Dog
  findDog(complex: ComplexInput): Dog
  booleanList(booleanListArg: [Boolean!]): Boolean
  arguments: Arguments
  human: Human
  pet: Pet
  name: String!
  Name: String!
  catOrDog: CatOrDog
}
enum DogCommand { SIT, DOWN, HEEL }
type Dog implements Pet {
  name: String!
  nickname: String
  barkVolume: Int
  doesKnowCommand(dogCommand: DogCommand!): Boolean!
  isHousetrained(atOtherHomes: Boolean): Boolean!
  owner: Human
}
interface Sentient {
  name: String!
}
```

### Implementing GraphQL Descriptor in Integration Server and Designer

You use the Service Development perspective to create a new GraphQL descriptor based on the schema.

- In the Designer perspective, select **File > New > GraphQL Descriptor**.

- In the New GraphQL Descriptor dialog box, select the folder in which you want to create the *GraphQL descriptor*.
- In the **Element name** field, type a name for the *GraphQL descriptor* using any combination of letters, numbers, and/or the underscore character.
- Click **Next**.
- In the Select the GraphQL Schema Location wizard, browse and select the schema file.

  **Note:**
  Ensure that the format of the schema file is .txt, .json, .graphql, or .graphqls.

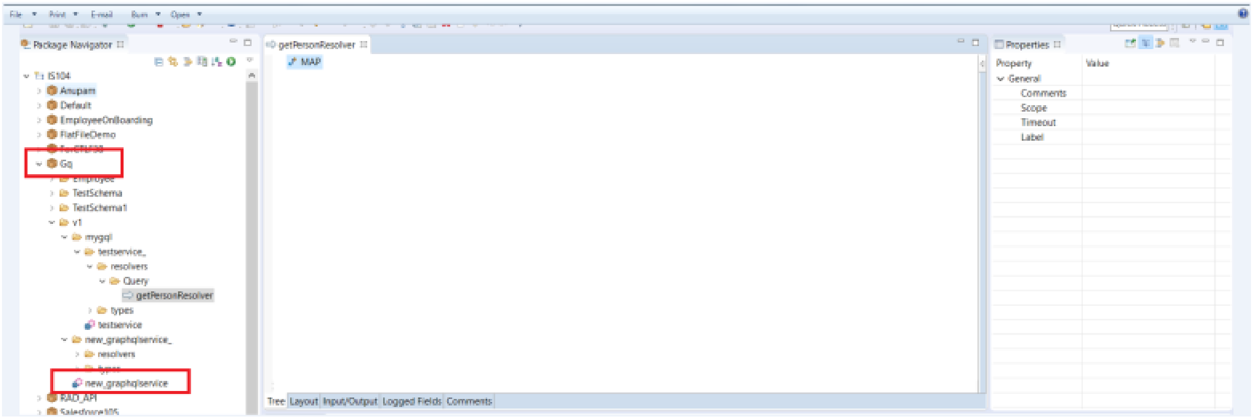
- Click **Finish**.
- Designer creates the *GraphQL descriptor* and displays the details in the *GraphQL descriptor* editor.

## Build Resolver Services for the GraphQL Descriptor

Integration Server creates the following assets when you import a GraphQL schema:

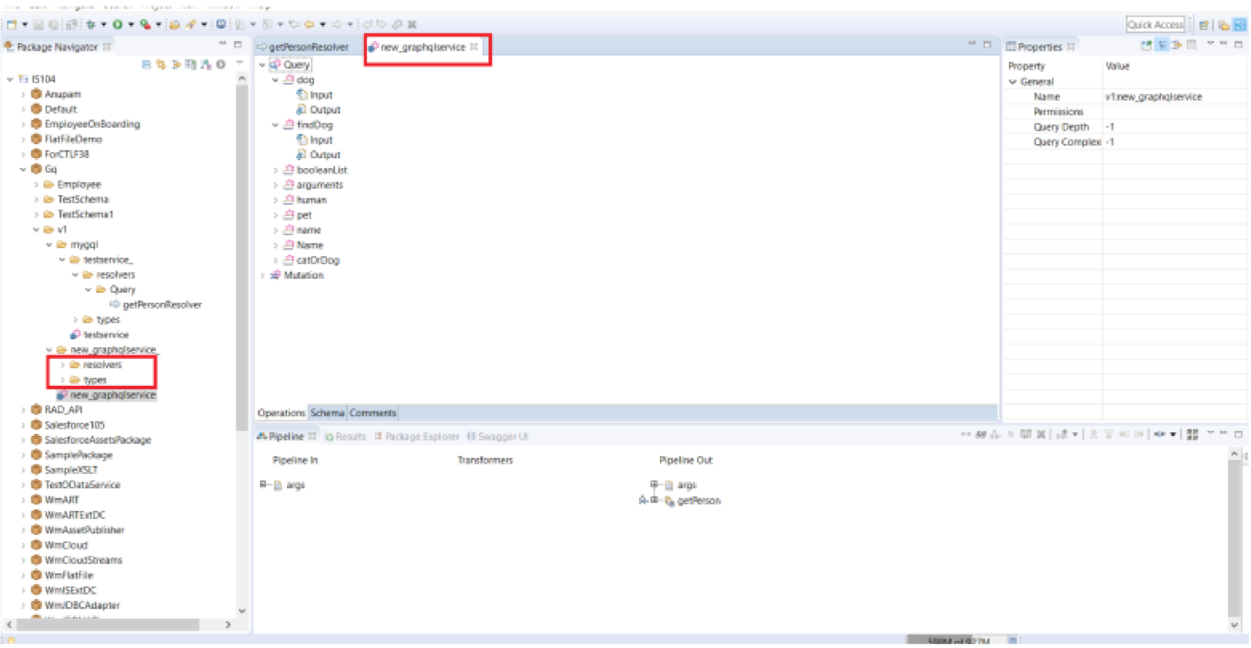
- A GraphQL descriptor with the name as provided by the user.

When you create a GraphQL descriptor using the above GraphQL schema with name Gq, then Designer creates folders with name types and resolvers under *Gq* folder like this:



All the document types are created under the *types* folder and all the resolver services are created under the *resolvers* folder.

If you import the above schema in Integration Server with name *Gq*, then Integration Server creates the resolver services for Query operation under `Gq_/resolvers/Query` folder, and the resolver services for Mutation operation under `Gq_/resolvers/Mutation` folder.



| Property | Value |
|---|---|
| General | |
| Name | v1:new_graphqlservice |
| Permissions | |
| Query Depth | -1 |
| Query Complexi | -1 |

The resolver will have the input/output signature set.

## Deploy Assets to Cloud Deployment

Using Designer, you deploy your on-premises Integration Server GraphQL packages and assets to Cloud Deployment. Before deploying the packages and assets, you must configure a connection to Integration Cloud.

Before you configure a connection to Integration Cloud, ensure that the following criteria are met:

- A valid URL exists to connect to Integration Cloud.
- A valid user account is created on Integration Cloud.

### To add a connection configuration for Integration Cloud

- In Designer, select **Window** > **Preferences**.
- In the preferences navigation tree, select **Software AG** >**Integration Cloud.**
- Click **Add**.
- In the **Add connection configuration** dialog box, enter the following information:

| Field | Description |
|---|---|
| **Name** | The name to use for the Cloud Deployment connection configuration.<br>**Note:** |

| Field | Description |
|---|---|
| | The name cannot contain control characters, special characters, and characters outside of the basic ASCII character set, such as multi-byte characters. |
| **URL** | URL of the Cloud Deployment host to which Designer is to connect. For example, `https://<sub-domain>.<domain-name>.<domain-suffix>.` |
| **User** | The user name for an account on Cloud Deployment. |
| **Password** | The password for the specified **User**. |
| **Save password (in the Eclipse secure storage)** | Indicates whether the password for the specified user account should be saved in Eclipse secure storage. Cloud Deployment uses this password from the Eclipse secure storage whenever user authorization is required. If you want to save the password in Eclipse secure storage, select this check box. |

A connection configuration is now added to the Connections page with the specified details.

**To deploy assets to Cloud Deployment**

1. In Package Navigator view, select the Integration Server package that you want to deploy to Cloud Deployment.
2. Right-click the GraphQl package and click **Deploy to Cloud**.

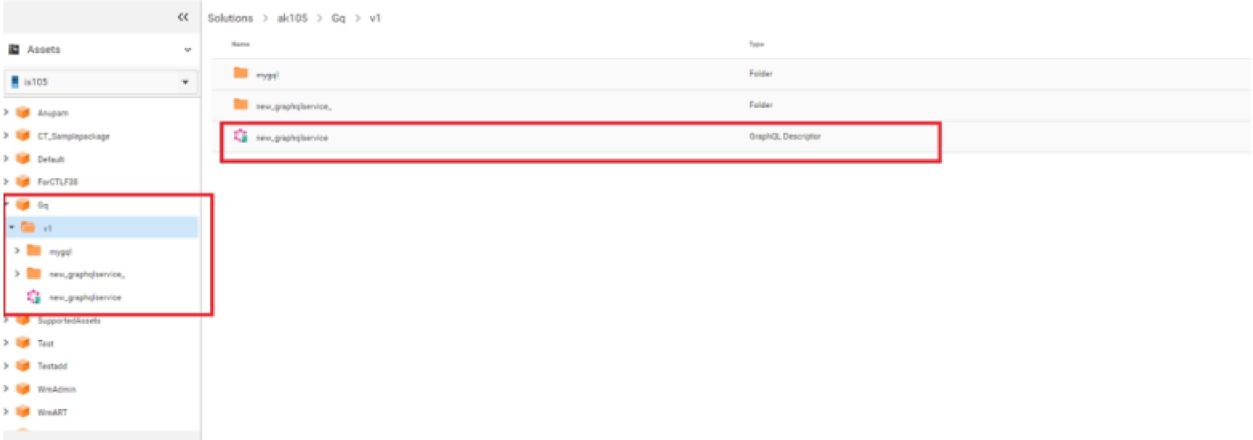
## Verify the Deployed Assets in Cloud Deployment

After publishing the GraphQL packages to an asset repository provisioned for the tenant on Cloud Deployment, you can view and access the webMethods Integration Server packages in Cloud Deployment. The solution details page allows you to view the packages, assets, and services for GraphQL runtimes in the solution.

To view the GraphQl packages in the Solution Explorer:

1. Switch to the Cloud Deployment perspective.
2. From the Cloud Deployment navigation bar, click **Solutions** > **Solution List**.
3. Click on an existing solution. The Solution Explorer page appears.

The Solution List page appears listing GraphQL solutions.

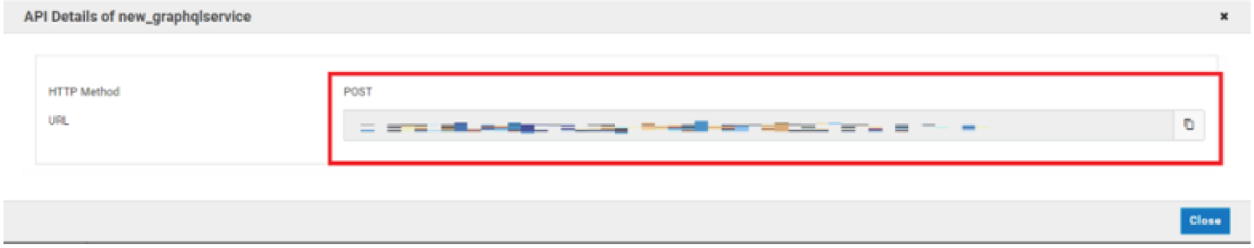


Solutions > ak105 > Gq > v1

| Name | Type |
|---|---|
| mygql | Folder |
| new_graphqlservice_ | Folder |
| new_graphqlservice | GraphQL Descriptor |

## Displaying the API details of an executable service

After deploying assets, on the Asset explorer page, click the ellipses ⋮ icon to view the API details of the service such as the HTTP Method and URL that are required to invoke this service from an external system. You copy the required API details to execute the service from the external client.

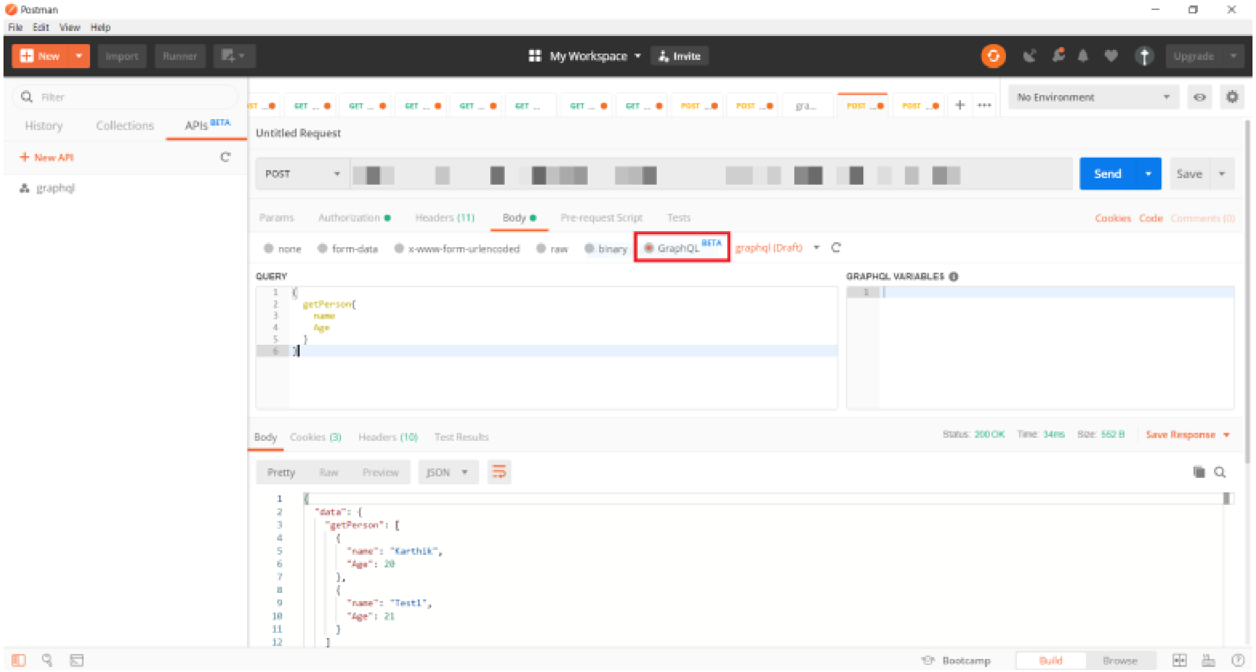


API Details of new_graphqlservice

| | |
|---|---|
| HTTP Method | POST |
| URL | [illegible] |

## Service Execution from External Client

To execute the service from a third party tool, let us use Postman as the GraphQL client. So in this example, when we open the GraphQL client, and invoke the URL as shown below, we get the output as per the fields passed in the query.



POST [redacted]

```
{
  getPerson{
    name
    Age
  }
}
```

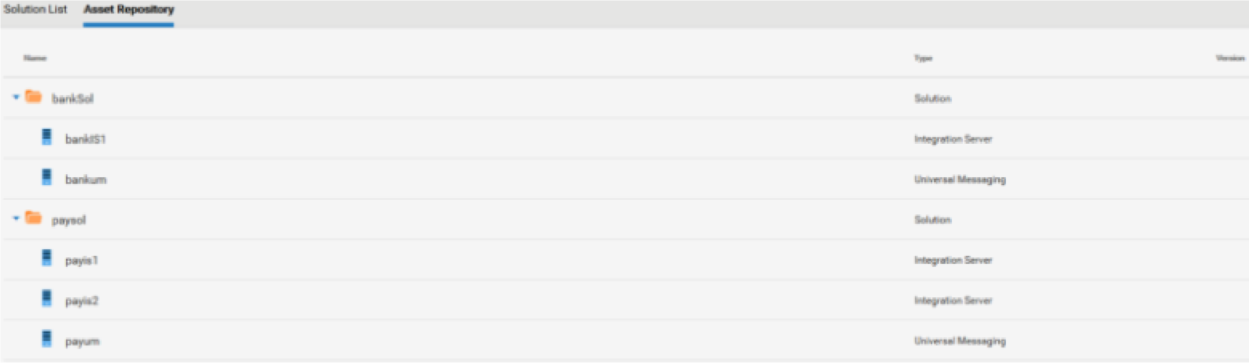
```
{
  "data": {
    "getPerson": [
      {
        "name": "Karthik",
        "Age": 20
      },
      {
        "name": "Test1",
        "Age": 21
      }
    ]
```

## Asset Repository

The **Asset Repository** page **(Cloud Deployment > Solutions > Asset Repository)** displays the list of all solutions and the user-created assets. You can also view the asset type, version of the assets, and services. Before deploying packages and configurations from Software AG Designer, you must create a solution in Integration Cloud to which you want to deploy the configuration assets. Software AG Designer deploys the assets and configurations to the Asset Repository in Integration Cloud.

### Asset Repository for all solutions

The following figure depicts the Asset Repository structure for all solutions.

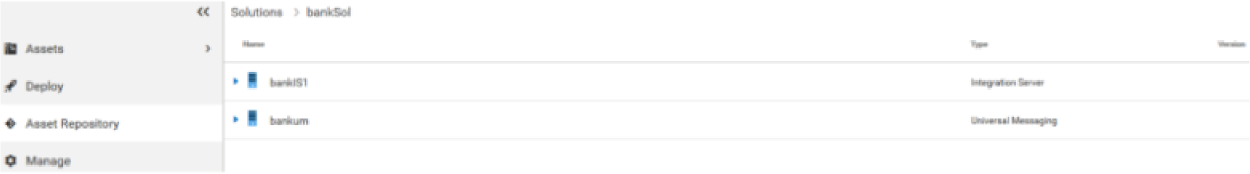


| Name | Type | Version |
|---|---|---|
| bankSol | Solution | |
| bankIS1 | Integration Server | |
| bankum | Universal Messaging | |
| paysol | Solution | |
| payis1 | Integration Server | |
| payis2 | Integration Server | |
| payum | Universal Messaging | |

### Asset Repository for a selected solution

The **Asset Repository** page for a solution **(Cloud Deployment > Solutions > Select a Solution > Asset Repository)** displays the assets for the selected solution, including the asset type and version of the assets.

The following figure depicts the Asset Repository structure for a selected solution.



| Name | Type | Version |
|---|---|---|
| bankIS1 | Integration Server | |
| bankum | Universal Messaging | |

### Downloading packages and assets

You can download user deployed packages and configurations from the **Asset Repository** page. You can either download individual packages or the whole repository for each product. To download assets, point to an asset or product and click the ⬇ icon. The assets including ACDL files will be zipped and downloaded to your local storage.

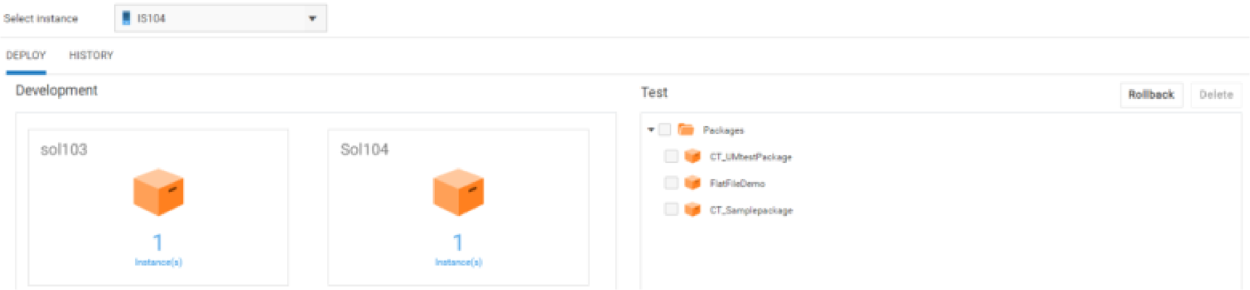
## Deploy

After publishing the assets and configurations that reside within on-premises runtimes or repositories to webMethods Integration Cloud, you can promote them from the previous stage to the current stage. Within a tenant, you can promote assets from a solution to another solution, from a previous stage to the current stage, for the same runtime type. You can promote assets if the source runtime version is lesser than or same as the target runtime version.

If you are in the **Development** stage, click the **Change Stage To View** link to change the stage. Only active and accessible stages appear in the drop-down list for selection in the **Stage to view** field. Select a different stage other than Development and click **Submit**.

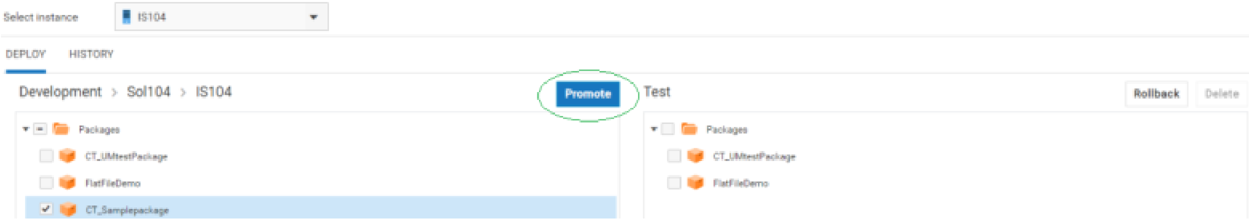
In the next stage, select a solution and then click **Deploy**. Select a runtime instance. All *solutions* of the previous stage will be listed on the left panel. The right panel will list all the assets of the selected solution in the current stage for the selected runtime instance.

**Note:**
You will *not* be able to promote assets from a higher version solution to a lower version solution. For example, if you have a v10.4 solution in the source stage and a v10.3 solution in the current stage, you will not be able to promote the v10.4 assets to the current stage.



**Note:**
You can access a stage only if your Access Profile is assigned to the stage.

Click on a solution and select the runtime package folder. Then click on the runtime instance. The assets of the solution corresponding to the selected runtime instance will appear. Select the assets and then click **Promote** to promote the assets to the current stage (right panel).



Select instance: IS104

After you click **Promote**, the **Promote Assets** dialog box appears for the selected asset.

Select an asset and change the values for the variable substitution properties, if needed. The variable substitution properties appear for the selected asset, only if the asset has properties. If there are any similar type of assets for which you want the same values, then select the **Show similar assets to apply values** option. Then select the assets in the lower panel. Click **Apply** to apply the property values to the selected assets. The changed values will be applied to all the selected similar assets during promotion.

On the **Promote** dialog box, you can type a message to describe the promotion. The promotion message will appear on the **History** page.

Click **Simulate Promote** to check the consistency of the assets and their dependencies. If there are dependencies, then for a successful promotion, you have to select all the dependent assets. Select **Save and Promote** to save the variable substitution and promote the assets to the next stage.

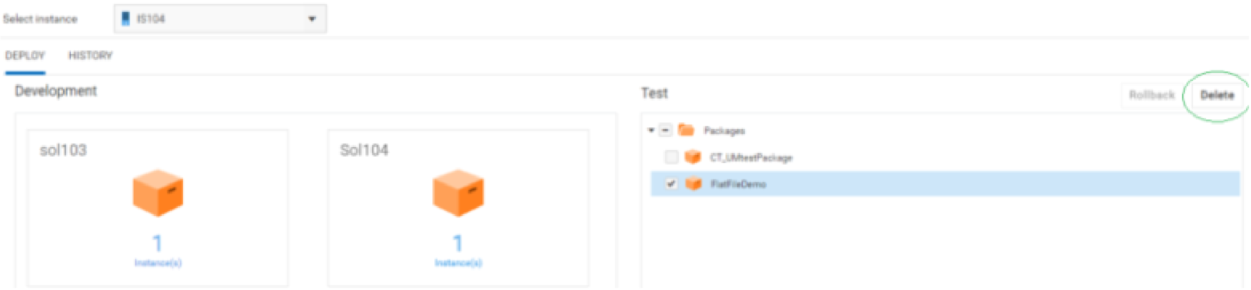
On the **Deploy** page, click **Rollback** to rollback *all* promoted assets to their previous state.

You can type a message to describe the rollback. The rollback message will appear on the **History** page.

## Deleting assets

On the **Deploy** page, you can delete an asset from the current stage (right panel). The asset will be deleted from the asset repository in that stage as well as from the runtime.

**Note:**
Currently you can delete only webMethods Integration Server packages and not webMethods Integration Server and Universal Messaging configurations.



## History

The **History** page shows the **Trace ID**, that is, the tracking ID, which is automatically generated on every successful or unsuccessful promotion, rollback, or deletion, the Deployment, Rollback, or Deletion **Action**, **Date** when the asset was promoted, rolled back, or deleted, the **User** who promoted, rolled back, or deleted the asset, and the commit **Message** for the selected instance. You can click on a **Trace ID** to see the Track History for the specific action.

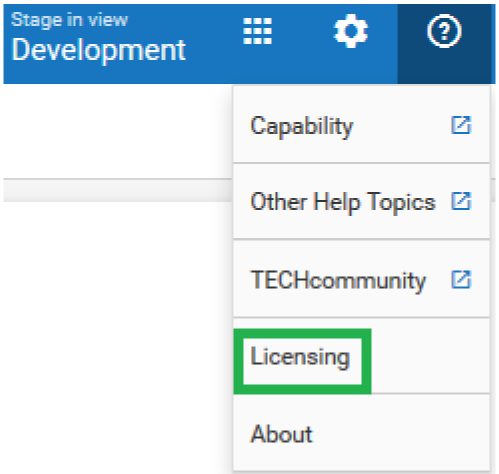
The **Track History** window displays the following details:

- **Timestamp** - The date and time when the log was generated.
- **Product** - The product that was promoted or rolled back.
- **Log Level** - Information on whether the log type is an Error, Info, or Debug.
- **Message** - Log or status message.

**Note:**
Promotion, rollback, or deletion details appear only for the current stage.

## Capability

The **Capability** (? > Licensing) page allows you to view the status of some of the system capabilities, based on your license offering.



You can view the details of the following capabilities in **Integration Cloud**:

| Field | Description |
|---|---|
| **Allowed application count** | Total number of Applications that can be utilized by the tenant. |
| **On-premises connection** | If **Yes**, then on-premises applications can be uploaded from on-premises systems. |
| **Max allowed users** | Maximum number of active users allowed for the tenant. |
| **Allowed number of stages** | Maximum number of staging environments allowed for the tenant. |
| **Integration restart and resume** | Integrations can be restarted and resumed. |
| **Integration import and export** | Integrations can be imported and exported. |
| **Trial account** | If **Yes**, then the account is a trial account. |
| **Trial end date** | The trial period end date. This field appears only if the account is a trial account. |

You can view the details of the following capabilities in Cloud Deployment:

| Field | Description |
|---|---|
| **Max allowed cores** | Maximum number of CPU cores allowed across all active solutions and all stages for the tenant. You will not be able to create additional solutions if you exceed this capability. |
| **Max allowed memory** | Maximum memory capacity allowed across all active solutions and all stages for the tenant. You will not be able to create additional solutions if you exceed this capability. |
| **Allowed number of stages** | Maximum number of staging environments allowed for the tenant. |
| **Trial account** | If **Yes**, then the account is a trial account. |
| **Trial end date** | The trial period end date. This field appears only if the account is a trial account. |

## Usage

The **Usage** page allows you to view the current usage of CPU cores and Memory (GB) for all the active solutions in all the stages.

To access this page, from the Cloud Deployment navigation bar, click ? and select *Licensing > Usage*.

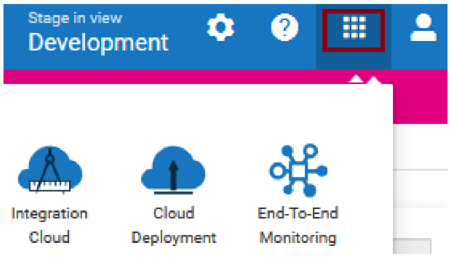
## Copying Solutions

The **Solution List** page allows you to copy solutions in any stage. Copying solutions allows you to have a back up of your solution before you make any changes and deploy your solution to production. This reduces the risk of not having a back up in case you want to revert to the original solution.

**Note:**
If you have the required permission under **Settings** ⚙ **> Access Profiles > Administrative Permissions > Functional Controls > Solution**, you can copy, update, and delete solutions.

### To copy a solution

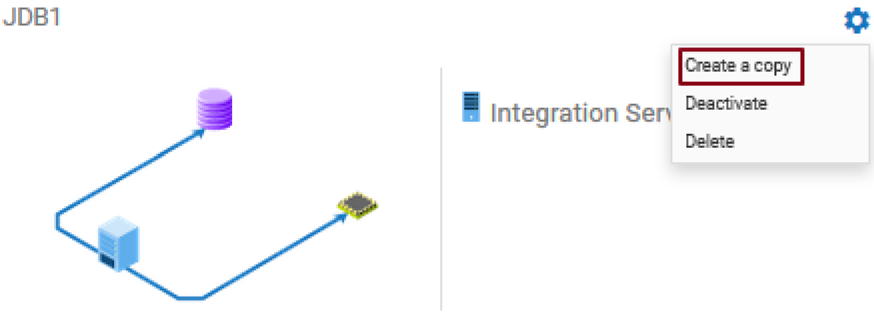
1. Switch to the **Cloud Deployment** perspective.



2. From the Cloud Deployment navigation bar, click **Solutions > Solution List**.

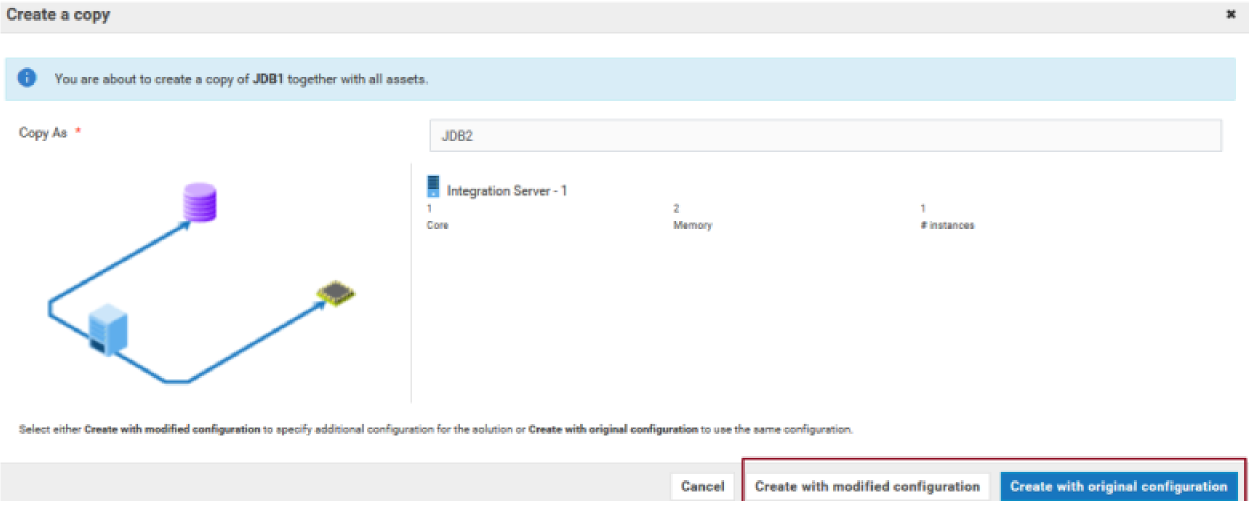
   The **Solution List** page appears.

3. On a solution, click ⚙ **> Create a copy**.



4. On the **Create a copy** page, fill in the new solution **Name**.

   You can choose to copy solutions using the same configuration and services in the solution landscape by clicking the **Create with original configuration** option or modify the configuration and services in the solution landscape by clicking the **Create with modified configuration** option.



Create a copy

You are about to create a copy of **JDB1** together with all assets.

Copy As * JDB2

Integration Server - 1

| 1 | 2 | 1 |
|---|---|---|
| Core | Memory | # instances |

Select either **Create with modified configuration** to specify additional configuration for the solution or **Create with original configuration** to use the same configuration.

Cancel
Create with modified configuration
Create with original configuration

5. Select **Create with modified configuration** and specify additional configuration of the solution, if necessary. Then click **Configure** to save the configuration details.

   The new solution is created and appears on the **Solution List** page.

   > **Note:**
   > You can now deploy the new solution to the next stage.

   **Deactivate, Activate, and Delete a solution**

   Click the ⚙ icon and select **Deactivate** to deactivate a solution. All packages and assets will be permanently deleted and cannot be recovered. Select **Activate** to activate an inactive solution. Select **Delete** to permanently delete a solution.

## Managing Solutions

The **Manage** options allow you to view the solution landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances.

### Landscape

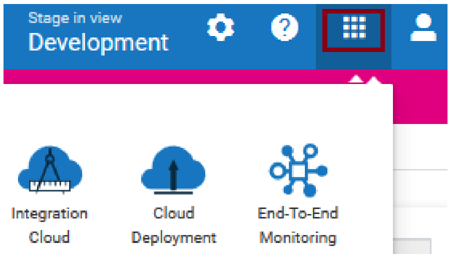
The **Landscape** page displays the landscape configuration for the selected solution.

> **Note:**
> You cannot modify the solution landscape configuration in the Development stage after the solution is created. You can *configure* the solution in subsequent stages but after a solution is configured, the solution cannot be modified again in that stage. Further, clustering and Terracotta options are not available in the Development stage.

**Note:**
In a stage, you can configure only those solutions that are marked as **Not Configured**.

### To view the landscape configuration for a solution

1. Switch to the **Cloud Deployment** perspective.



2. From the Cloud Deployment navigation bar, click **Solutions > Solution List > Select a solution > Manage > Landscape**.

3. On the **Landscape** page, you can view the landscape configuration design, landscape solution name and description, and the landscape components. For each landscape component, you can view the landscape component name, product type, whether the landscape component is in a ready state, and the number of CPU cores and memory characteristics of the hardware to support each service in the solution.

4. Terracotta is available only when webMethods Integration Server runs in a clustered mode. Further, clustering and Terracotta options are not available in the **Development** stage.

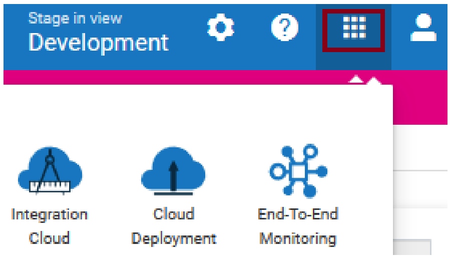
   On the **Landscape** page, select the **Cluster Type** as **Stateless** if the group of webMethods Integration Servers function in a manner similar to a cluster but are not part of a configured cluster. A stateless cluster of webMethods Integration Servers does not use a Terracotta Server Array. Select **Stateful** to add the Terracotta section. The Terracotta icons will be activated.

## Service Access Settings

The **Service Access Settings** page allows you to configure the webMethods Integration Server services to be called externally over HTTPS. The services will be available to consumers based on the **Allow All** and **Deny All** access modes.

### To configure service access settings

1. Switch to the **Cloud Deployment** perspective.



2. From the Cloud Deployment navigation bar, click **Solutions > Solution List > Select a solution > Manage > Service Access Settings**.

3. On the **Service Access Settings** page, configure the webMethods Integration Server services to be called externally over HTTPS. Required fields are marked with an asterisk on the screen.

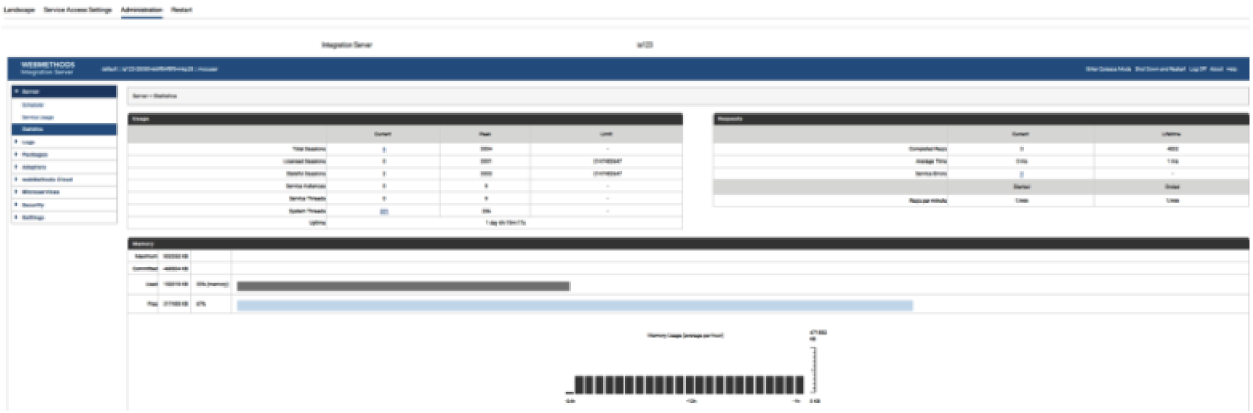
| Field | Description |
|---|---|
| **Base URL** | The base URL is a part of the complete service URL, for example,<br>https://wmic1.saglive.com/integration/<br>clouddeployment/service/<br>development/test/tt54/invoke/pub.* |
| **Solution Alias** | An alias for the solution in the base URL. The alias name for a solution is unique in that particular stage. For example, if the completed service URL is https://wmic1.saglive.com/<br>integration/clouddeployment/service/development/<br>test/tt54/invoke/pub.*<br>then the Solution Alias is test, the webMethods Integration Server instance is tt54, and the service URL is /invoke/pub.*. |
| **Select Integration Server** | Select the solution webMethods Integration Server instance where the services need to be configured. |
| **Access Mode** | Ensure that the access mode of the services are properly set.<br>Select **Deny All** if you want to deny most of the services to be invoked and allow a few. Then click **ADD** to add services to the **Allowed Services** table. In the **Add Service** window, the **Base URL** is a part of the complete service URL. In the **Service URL** field, type the webMethods Integration Server Service URL. The Base |

| Field | Description |
|---|---|
|  | URL and the Service URL together forms the complete service URL.<br>Select **Allow All** if you want to allow most of the services to be invoked and deny a few. Click **ADD** to add services to the **Denied Services** table. In the **Add Service** window, the **Base URL** is a part of the complete service URL. In the **Service URL** field, type the webMethods Integration Server Service URL. The Base URL and the Service URL together forms the complete service URL.<br>**Note:**<br>You can update the service URL by clicking the **Edit** icon beside the service URL in the services table.<br>The services will be available to be invoked from a software application, for example, a REST client, after you add the services to the table.<br>For example, the Service URL */invoke/pub.math:addInts* has the following components:<br>■ Directive - *Invoke*<br>■ Namespace of the service - *pub.math:addInts*<br>**Note:**<br>You can match all services to be allowed or denied by typing * at the end of the Service URL. For example, if you have two services, service url1: /invoke/pub.date:formatDate and service url2: /invoke/pub.date:getCurrentDate /invoke/, then instead of typing two entries, you can provide only one entry, service url: /invoke/pub.date.*. All services matching pub.date will be allowed or denied. |

## Administration

Use this page to manage a solution webMethods Integration Server Administrator instance.

The webMethods Integration Server Administrator is an HTML-based utility you use to administer the webMethods Integration Server. It allows you to monitor server activity, manage user accounts, make performance adjustments, and set operating parameters. You can run the webMethods Integration Server from any browser-equipped workstation on your network. When you click **Administration**, your browser displays the **Statistics** screen.



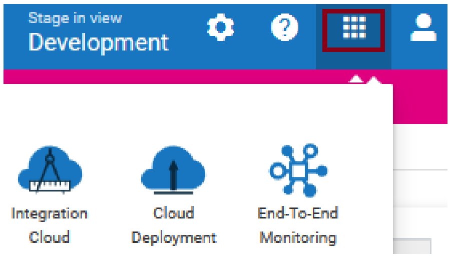
The Title bar displays the name of the host machine where webMethods Integration Server is running, the name of the webMethods Integration Server instance, and the name of the user currently logged into the webMethods Integration Server instance .

The Navigation panel on the left side of the page displays the names of menus from which you can select a task. To start a task, click a subject in the Navigation panel. The server displays a screen that corresponds to the task you select.

**Note:**
Click **Help** to view the Help system, which provides information about the features and functionality of webMethods Integration Server.

### To view the Administration page

1. Switch to the **Cloud Deployment** perspective.



2. From the Cloud Deployment navigation bar, click **Solutions > Solution List > Select a solution > Manage > Administration**.

   The webMethods Integration Server Administrator page appears.

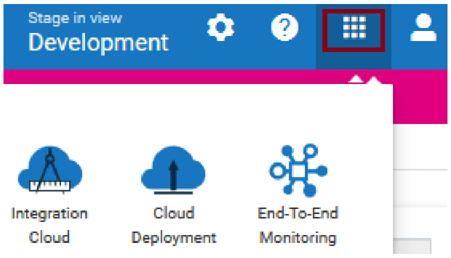
## Restart

Restart the server when you need to stop and reload the server. You should restart the server when:

- You make certain configuration changes. Some configuration changes require the server to be restarted before they take effect.
- If you encounter an operational problem or the server is in an inconsistent state.

### To restart the server

1. Switch to the **Cloud Deployment** perspective.



2. From the Cloud Deployment navigation bar, click **Solutions > Solution List > Select a solution > Manage > Restart**.

> **Note:**
> webMethods Integration Servers running in a clustered mode cannot be restarted. Further, restarting servers will terminate all active sessions.

3. Click **Restart**.

## Monitoring Solutions

The Monitoring part of Cloud Deployment enables you to monitor the health and availability of the solutions and run-time instances, alerts and alert statuses. You receive an email whenever there is a condition that might affect the solution.

The monitoring of a new solution starts automatically 10 minutes after the creation of the solution. The data of the solution is collected and analyzed every 60 seconds.

You can access the monitoring pages from the left-side navigation menu of the Monitoring main page.

You can filter the information on most Monitoring pages based on time. To specify the time-range, select a value in the time-range selector.

The following table describes the options in the time-range selector.

| Option | Description |
|---|---|
| 12h | Displays the information for the last 12 hours. |
| 24h | Default. Displays the information for the last 24 hours. |

| Option | Description |
|---|---|
| 2d | Displays the information for the last 2 days. |
| 1w | Displays the information for the last week. |
| 2w | Displays the information for the last 2 weeks. |
| 3w | Displays the information for the last 3 weeks. |
| 4w | Displays the information for the last 4 weeks. |

To navigate to the Monitoring main page, log in to Integration Cloud, switch to the Cloud Deployment perspective, and select **Monitoring** in the Cloud Deployment navigation bar.

## Dashboard

On the **Dashboard** page, you can view:

- The health of the solutions
- The number of the alerts that have been raised for all the solutions
- The landscape view of the solutions, and the number of alerts for all run-times that are part of the solutions

The following table provides more information about the panes on the Dashboard page.

| Pane | Description |
|---|---|
| Overall KPI Status | Shows the following information about the health of the solutions for the selected time range:<br>■ Total number of solutions<br>■ The number of healthy solutions<br>■ The number of unhealthy solutions<br>■ The health of the solutions, as a percentage value calculated by the formula (Number of healthy solutions / total number of solutions) * 100<br>A healthy solution is a solution without any open critical alerts.<br>An unhealthy solution is a solution which has at least one open critical alert.<br>To see more information about the KPI status of the solutions, click **More Details**. |

| Pane | Description |
|---|---|
| Alerts | Shows the total number of open and resolved alerts that have been raised for all solutions for the selected time-range, and the number of alerts from each type: critical, warning, or information.<br>To see more details about the alerts, click **More Details**.<br>For more information about the alert types, see "Alert Types" on page 776. |
| Landscapes | Displays the topology of the solutions and the number of alerts for each run-time type since the solution has been activated.<br>The **Update in progress** link indicates that the solution is under maintenance. Click **Update in progress** to see more details about the update. |

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see "Monitoring Solutions" on page 767.

## Solutions

On the Solutions page, you can check the health of the run-time instances from all the solutions. For each run-time instance, you can view the current data, and the data for the last 24 hours.

The health metrics are grouped into three categories:

- Memory - indicate the memory utilization of a run-time.
- Uptime - indicate the availability of a run-time.
- Failures- indicate failures of the run-time

The following table describes the icons on the Solutions page.

| Icon | Description |
|---|---|
| ✓ | Normal health of the run-time instance. |
| i | The health of the run-time instance is deteriorating. Take preventive measures. |
| ✖ | There are critical issues with the health oh the run-time instance. Your urgent attention is needed. |
| ✕ | The run-time instance is not available. |

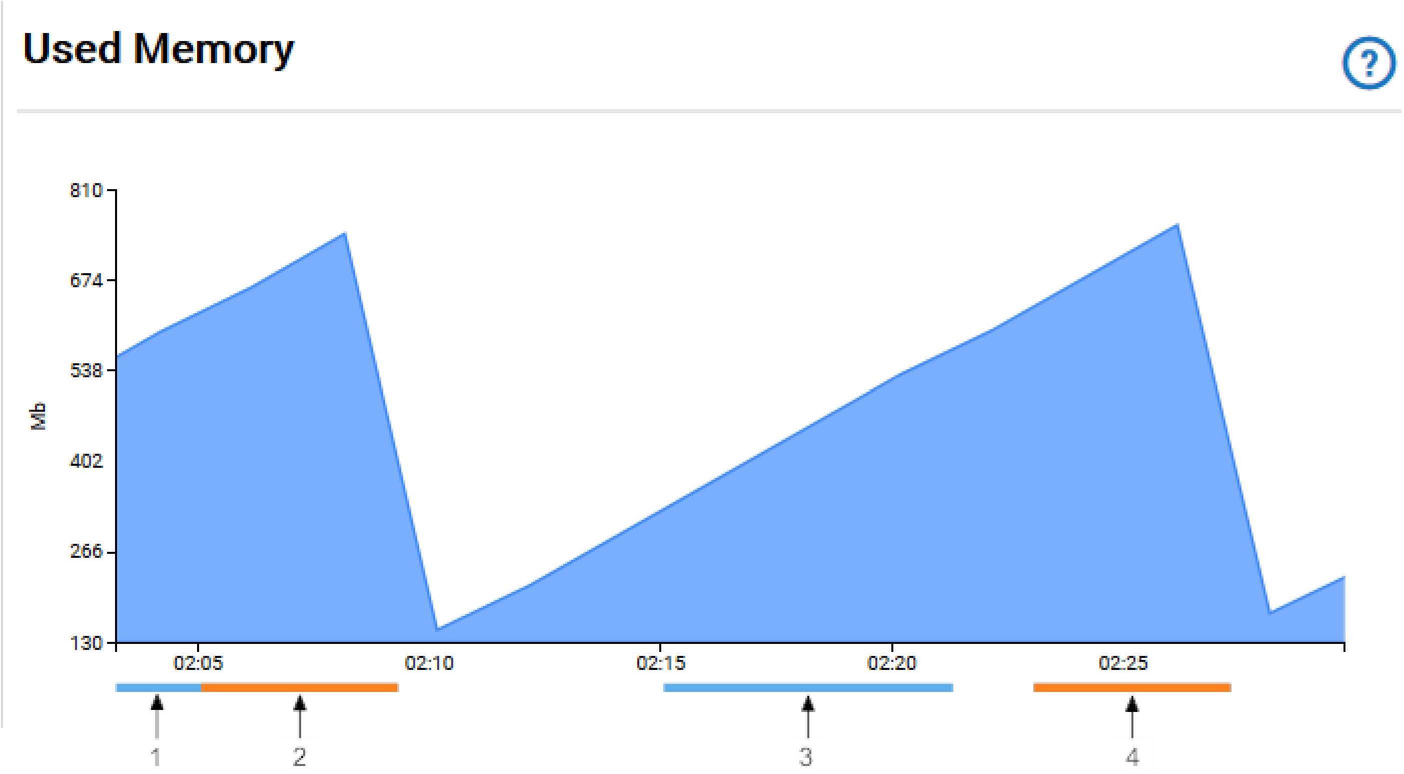
**Note:**
If the solution uses an Integration Server cluster, the number of Integration Server instances is indicated in brackets after the Integration Server instance name.

During a solution update, the Solutions page does not display status icons for the run-time instances from the solution. To view more details about the solution update, click **Update in progress**.

To view more details about a run-time instance on the **Runtime** page, click the name of the run-time instance.

## Runtimes

On the Runtimes page, you can view the graphs for monitored KPIs for the selected run-time instances from all the solutions.



The example image shows the graph for the Used Memory KPI. The horizontal lines below the graph represent the severity and duration of the alerts that were raised for the KPI. The information alerts are displayed in blue, the warning alerts are in orange, and the critical alerts are in red.

The following table describes the meaning of the alert lines from the example graph for the Used Memory KPI.

| Time Period | Details |
|---|---|
| 1 | Until 2:05 h, there had been an open information alert. |

| Time Period | Details |
|---|---|
| 2 | At 2:05 h, the severity of the information alert was changed to warning. |
| 3 | An information alert existed during that period. |
| 4 | A warning alert existed during that period. |

You can change the value in the **Solutions** drop-down field to load the information about the run-time instances from a specific solution.

You can use the Integration Server, Universal Messaging, and Terracotta tabs to view the information related to the selected solution and runtime.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see "Monitoring Solutions" on page 767.

The following table describes the monitored Integration Server KPIs.

| Name | Description |
|---|---|
| Used Memory | The total used memory for the Java VM. |
| Service Threads | The number of active service threads. |
| Sessions | The number of active licensed sessions. |
| Stateful Sessions | The number of the current stateful HTTP sessions. |

The following table describes the monitored Universal Messaging KPIs.

| Name | Description |
|---|---|
| Free Memory | The amount of free memory that the Realm Server has within the Java VM. This indicates the difference between what the Java VM has currently allocated and what the Realm Server has used. |
| Published Events | Total number of events published on this realm from the time it started. |
| Subscribed Events | Total number of events that this realm has sent to clients from the time it started. |

The following table describes the monitored Terracotta KPIs.

| Name | Description |
|---|---|
| Off-Heap Used Memory | Shows the amount of off-heap memory that is currently used. |
| Live Objects | Shows the total number of live objects in the cluster, mirror group, server, or clients. If the trend for the total number of live objects goes up continuously, clients in the cluster will eventually run out of memory and applications might fail. Upward trends indicate a problem with application logic, garbage collection, or the tuning of one or more clients. |

### Viewing Adapter KPIs

On the **Runtimes** page, you can view the KPIs for the adapters that are installed on the Integration Server instances.

1. Navigate to the Runtimes page.

2. Select a solution.

3. On the Integration Server tab, select an Integration Server instance.

4. Click **Connectivity KPIs**.

5. On the Adapters tab, select an Adapter.

   The Adapter KPIs are displayed.

   The following table describes the monitored Adapter KPIs:

| Name | Description |
|---|---|
| Connections | The number of connection pools in the adapter and how many of them are currently enabled. |
| Notifications | The number of adapter notifications (polling notifications) and how many of them are currently enabled. |

> **Note:**
> You can view Adapter KPIs only for the current time.

### Viewing Connector KPIs

On the **Runtimes** page, you can view the KPIs for the connectors that are installed on the Integration Server instances.

1. Navigate to the Runtimes page.
2. Select a solution.
3. On the Integration Server tab, select an Integration Server instance.
4. Click **Connectivity KPIs**.
5. Click the **Connectors** tab.
6. Select a provider.
7. Select a connector.

   The Connector KPIs are displayed.

   The following table describes the monitored Connector KPIs:

| Name | Description |
|---|---|
| Connections | The number of connection pools in the connector and how many of them are currently enabled. |
| Listeners | The number of connector listeners and how many of them are currently enabled. |

> **Note:**
> You can view Connector KPIs only for the current time.

## Services

On the Services page, you can view the number of successful and failed service executions of the Integration Server instances from the solutions.

The **Services** page consists of the **Service Executions** pane and the **History** pane.

| Pane | Description |
|---|---|
| Service Executions | Shows the following information about the service executions of the Integration Server instances for the selected time range:<br>- Total number of service executions<br>- The number of successful service executions<br>- The number of failed service executions |

| Pane | Description |
|---|---|
| | - The successful service execution, as a percentage value calculated by the formula (Number of successful service executions / total number of service executions) * 100 |
| History | Shows a chart with the history of successful (green) and failed (red) service executions.<br>Hovering over the green and red bars displays the total number of successful and failed service executions, correspondingly. |

The numbers of service executions on the Services page includes the public and internal services of the Integration Server instance and their child services.

You can change the value in the **Solutions** drop-down field to view the information about a specific solution, or the information for all solutions.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see "Monitoring Solutions" on page 767.

## Uptime

On the **Uptime** page, you can view time lines that represent the availability of all run-time instances of the solutions.

The color of the time lines changes based on the status of the run-time instances.

The following table describes the meaning of the different colors.

| Time line color | Indicates that |
|---|---|
| green | the run-time instance was available during the indicated time period. |
| red | the run-time instance was unavailable during the indicated time period. |
| grey | the run-time instance did not exist during the indicated time period. |
| blue | at least one node from the cluster is unavailable. |
| yellow | a solution update is in progress (under maintenance). |

**Note:**
If the solution uses an Integration Server cluster, the number of Integration Server instances is indicated in brackets after the Integration Server instance name.

By default, the time line displays the availability of the instances during the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see “Monitoring Solutions” on page 767.

## Alerts

The alert is a notification that a rule is violated.

On the **Alerts** page you can:

- View the number of critical, warning, and information alerts for all the solutions for the selected time range
- Filter the alerts by solution, runtime, severity, and status
- Configure the rules by adjusting the alert threshold values
- Configure the summary of the alerts
- Configure the recipient email for the alerts. For more information about configuring the alerts, see “Configuring the Alerts” on page 777.

By default, the Alerts page displays the number of alerts (critical, warning, and information) for all the solutions, and detailed information about the alerts in a tabular format.

**Note:**
If the duration of the rule violation is less than the time interval at which the rule is evaluated, the alert does not appear on the Alerts page. For more information about the interval, see “Configuring the Alerts” on page 777.

If you deactivate a solution, the Alerts page will not display the alerts for the solution.

If you activate a solution, the Alerts page will display both the historical alerts for the solutions that had been raised before the deactivation of the solution, and the alerts that were raised after the activation of the solution.

When a solution update starts, the existing active alerts for the solution are set to resolved. During the update period, no alerts are generated for the solution. You can disregard any email alerts that you receive during the upgrade period.

The following table describes the information that is displayed in the table on the Alerts page.

| Column | Description |
|---|---|
| **Solution** | Name of the solution. |
| **Runtime** | Run-time type.<br>■ Integration Server<br>■ Universal Messaging<br>■ Terracotta |

| Column | Description |
|---|---|
| **Instance** | Name of the run-time instance. |
| **Start Date** | Date and time when the alert was raised. |
| **Resolved On** | Date and time when the alert was resolved. The field is empty if the alert is still active. |
| **Message** | Description of the alert. |
| **Status** | Status of the alert.<br>- ✓ The alert is inactive.<br>- ✕ The alert is active. |

**Note:**
The Alerts page might not display the alerts for all nodes from a cluster. For example, if you monitor an Integration Server cluster with two Integration Server instances, and both instances have alerts for the same property with different severity, the Alerts page will show the alert of lower severity only, as explained in the following table.

| Integration Server instance | Alert type | Visibility on the Alerts page |
|---|---|---|
| Integration Server instance 1 | Information. Free memory is low. | Yes |
| Integration Server instance 2 | Warning. Free memory is low. | No |

You can view all alerts for all the nodes from the cluster in the email alerts.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see "Monitoring Solutions" on page 767.

## Alert Types

The following table provides more information about the alert types.

**Note:**
Warning alerts and information alerts are not available for KPIs that monitor the availability of a run-time instance.

| Alert Severity | Description | Color Coding |
|---|---|---|
| **Critical** | A condition exists that is critical for the system performance. | red |

| Alert Severity | Description | Color Coding |
|---|---|---|
| **Warning** | A condition exists that might deteriorate the system performance. | orange |
| **Information** | A condition exists that might evolve into a warning or critical alert. | blue |

## Configuring the Alerts

You can change the default threshold values and the recipient email for the system alerts. Threshold values determine when a rule is violated and when the system raises an alert.

### To configure the system alerts

1. Navigate to the Alerts page.

2. Select the **CONFIGURATION** tab.

   The Configuration page shows information about the alerts for all solutions. The following table describes the columns in the form.

   | Column | Description |
   |---|---|
   | **Name** | Alert Name. |
   | **Runtime** | Integration Server, Universal Messaging, or Terracotta. |
   | **Action** | The icon activates the configuration view for the alert. |

3. Click the **Edit this rule** icon in the Action column for the alert that you want to configure.

   A form with the configuration details for the alert rule is displayed. The following table describes the fields in the form.

   | Field | Description |
   |---|---|
   | **Threshold** | The KPI's boundary values. When the value of the KPI is outside the range that is specified by these boundary values, the alert is raised.<br>You can configure the threshold values of critical alerts by adjusting the ends of the red line.<br>You can configure the threshold values of warning alerts by adjusting the ends of the orange line.<br>You can configure the threshold values of information alerts by adjusting the ends of the blue line. |

| Field | Description |
|---|---|
| | **Note:**<br>The Threshold field is read-only for KPIs that monitor the availability of a runtime instance. |
| **Runtime** | Integration Server, Universal Messaging, or Terracotta. |
| **Summary** | Summary of the alert. |
| **Interval** | The scrape interval. The scrape interval is the frequency at which the system collects the data. The scrape interval is 60 seconds for all rules. Read-only.<br>**Note:**<br>The alert does not appear immediately when the corresponding rule violation occurs.<br>The time delay from the actual time of the rule violation to the system alert is the following:<br>■ up to 70 seconds for run-time availability rules of critical severity<br>■ up to 420 seconds for run-time availability rules of information severity<br>■ up to 180 seconds for the rest of the rules<br>The system will not send an alert if the rule violation condition is resolved during the corresponding delay period. |
| **Email on alert** | Email of the user who will receive the alerts.<br>**Note:**<br>The email is used for all rules. In case there are alerts, the system sends emails once every 10 minutes. The email alerts always display UTC time.<br>To configure more than one email, use comma-separated values. |

4. In the **Threshold** field, change the default threshold value(s) for the alert.

5. In the **Email on Alert** field, type the email(s) of the user(s) who will receive the email alerts for all rules.

   **Note:**
   The email(s) are stored in the local alertManager.yaml file. When you delete a tenant, the related information is deleted automatically.

6. Click **Apply**.

## Alert Actions

You can take actions and resolve the problems with the solutions that caused the alerts. The following table relates the alert with the probable cause of the problem, and the recommended actions that you can take to resolve the problem.

| Alert Name | Probable Cause | Action to Resolve the Alert |
|---|---|---|
| ISFreeMemoryLow | The memory usage is reaching the configured thresholds.<br>If the memory usage is continuously reaching 95% and above, and you do not observe any flaw in your application, then probably there is another memory-intensive application. | Allocate more memory to the solution. |
| ISRuntimeSessionUsageHigh | The solution uses too many sessions and there might not be free sessions for new requests. | Try one of the following:<br>■ Stop some unnecessary services, if any<br>■ Increase the maximum number of active licensed sessions<br>■ Move some of the workload to another solution |
| ISRuntimeStatefulSessionUsageHigh | The number of the current stateful HTTP sessions is high. There might not be enough bandwidth for new sessions. | Move some of the workload to another solution. |
| ISRuntimeUnavailable | Integration Server is down. | Try one of the following:<br>■ If the Integration Server went down |

| Alert Name | Probable Cause | Action to Resolve the Alert |
|---|---|---|
| | | because of a high workload, create an Integration Server cluster.<br>- If the Integration Server went down because of insufficient memory and you also get a memory alert, allocate more memory for Integration Server. |
| TCOffHeapMemoryLow | The heap-off memory has reached the threshold because of too much stored data. | Stop adding data or delete some data from the heap-off memory. |
| TCRuntimeUnavailable | The Terracotta server went down, or there was a human mistake (for example, somebody shut down the Terracotta server). | Restart the Terracotta server. For greater safety and security, start with the server that was shut down last. |
| UMRuntimeUnavailable | The Universal Messaging server is down. | Restart the Universal Messaging server. If the problem persists, contact the Software AG Global Support. |
| UMFreeMemoryLow | The memory usage is reaching the configured thresholds.<br>If the memory usage is continuously reaching 95% and above, and you do not observe any flaw in your application, then probably there is another | Increase the memory allocated to Universal Messaging. |

| Alert Name | Probable Cause | Action to Resolve the Alert |
|---|---|---|
| | memory-intensive application. | |

## Logs

On the **Logs** page, you can view the logs of the run-time instances in the solutions for a selected time period.

To view the logs for all the instances from a specific solution, select the solution in the solutions drop-down list.

To view the logs for a specific run-time instance in a solution, select the run-time instance in the run-time instance drop-down list.

By default, the page displays the logs for the last 24 hours. To view the logs for a different time period, use the time-range selector. For more information about the time-range selector, see "Monitoring Solutions" on page 767.

You can view logs message statistics and logs details, such as log timestamp and message text. You can change the type of log details that you see by adding or editing filters as described in the Kibana documentation.

# Database

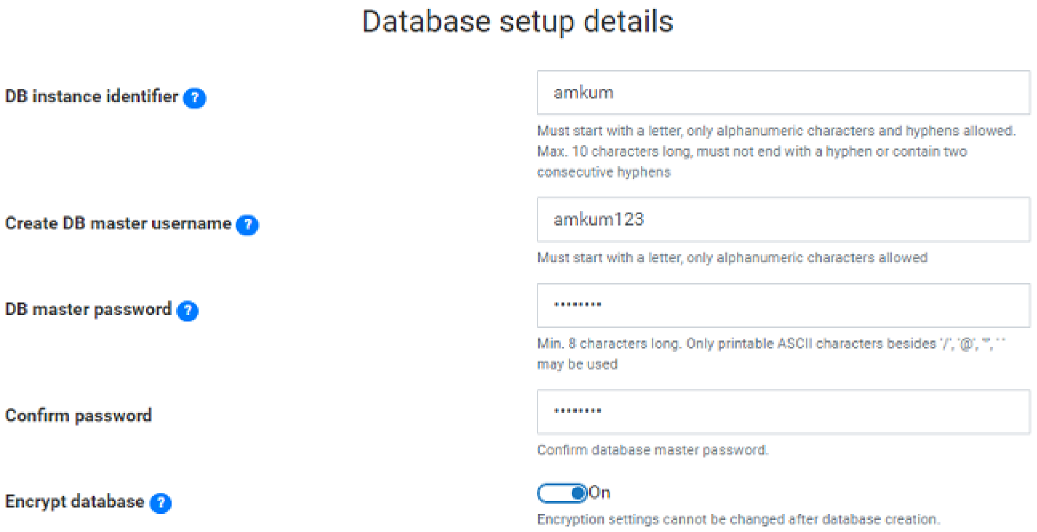
For optimal performance, you can add a MySQL database to your Cloud Deployment subscription. This enables you to configure, store, and monitor your database directly in the cloud instead of using external systems. The database endpoint can be shared by multiple solutions deployed by the tenant.

## Creating a Database

### To create a database instance in the cloud

1. On the home page in Cloud Deployment, under **Database**, click **Learn more**, and then click **Start database setup**.

2. On the **Database setup details** screen, do the following:



Database setup details

| | |
|---|---|
| DB instance identifier ? | amkum<br>Must start with a letter, only alphanumeric characters and hyphens allowed. Max. 10 characters long, must not end with a hyphen or contain two consecutive hyphens |
| Create DB master username ? | amkum123<br>Must start with a letter, only alphanumeric characters allowed |
| DB master password ? | ••••••••<br>Min. 8 characters long. Only printable ASCII characters besides '/', '@', '"', ' ' may be used |
| Confirm password | ••••••••<br>Confirm database master password. |
| Encrypt database ? | On<br>Encryption settings cannot be changed after database creation. |

a. In the **Database instance identifier** field, specify a name for the database.

> **Important:**
> Each Software AG Cloud tenant can create only one database instance.

b. In the **Create DB master username** and **DB master password** fields, specify the username and password that you will use to access the database.

> **Note:**
> The master user has full privileges on the database. For more information on database privileges, see the MySQL documentation.

c. Select **Encrypt database** if you want to enable the encryption of database data and click **Continue**. Note that you cannot change the encryption settings after the database is created.

When the database configuration is complete, you are redirected to the home page of Cloud Deployment where the **Database** element shows that the database server is running.

## Connecting a Solution to the Database

After you create the database instance in the cloud, you can connect one or more solutions to it.

### To connect a solution to the database instance in the cloud

1. Enable the wMJDBCAdapter package for the solution. For more information, see the Solutions chapter in Cloud Deployment.

2. In Software AG Designer, deploy to the cloud the on-premises webMethods Integration Server packages that you want to include in the solution.

a. In the Package Navigator view, right-click any of the packages that you want to include and select **Deploy to Cloud**.

b. In the Publish Assets to Integration Cloud dialog box, select all packages that you want to include in the solution and click **Next**.

c. In the webMethods Integration Server Packages Variable Substitution dialog box, do the following for each of the packages that have JDBC connections:

- Enable the **State after Deployment** property.
- In the **user**, **password**, and **serverName** fields, specify the username, password, and host name of the database instance in the cloud.

  > **Note:**
  > You can see the host name of the database in the **DB instance endpoint** field on the Database page in Cloud Deployment.

d. In the Select the Integration Cloud Solution dialog box, select the solution that you want to connect to the database and click **Finish**.

## Securing the Connection Between a Solution and the Database

### To secure the connection between a solution and the database

1. Import the Amazon RDS Root CA certificate as a trusted certificate in a Java keystore. For more details on the certificate, see the Amazon Relational Database Service documentation.

2. Copy the truststore to the config directory of any of the packages you deploy to the solution.

3. In the web administration interface of the on-premises webMethods Integration Server, go to **Adapters > WebMethods Adapter for JDBC > Connections**. In the Other Properties field of the JDBC adapter connection, specify:

   ```
   sslMode=VERIFY_IDENTITY;
   trustCertificateKeyStoreUrl=file:packages/<package>/config/<truststore>;
   trustCertificateKeyStorePassword=<password>
   ```

   where *package* is the name of the package that contains the truststore, *truststore*is the name of the keystore that contains the trusted certificate, and *password* is the password to the truststore.

4. In Software AG Designer, deploy the updated packages to the cloud.

> **Note:**
> Amazon Relational Database Service currently does not support authentication that uses a client certificate.

## Monitoring the Database

After you create the database instance in the cloud, you can monitor its status on the Database page in Cloud Deployment.

On the **Database** dashboard of the Database page, you can view the following elements:

| Element | Description |
|---|---|
| **DB CPU usage** | CPU usage in percentage. |
| **DB storage size** | The allocated storage space and the storage space used by the database. |
| **DB server status** | Shows if the database server is running. You can start or stop the server at any time by clicking **Start server** or **Stop server**. |

In the **Configuration details** section of the Database page, you can view the following fields:

| Field | Description |
|---|---|
| **DB instance** | The size of the database, based on your license offering. |
| **DB instance identifier** | The name of the database. |
| **DB instance endpoint** | The host and port that you can use to access the database. |
| **Allowed IPs** | The list of external IPs that can access the database.<br>Software AG recommends using the database only from applications in Software AG Cloud. By default, you can access the database only in the cloud. For administrative purposes you can enable access to the database from external IPs.<br>To add external IPs, click **Edit** and type each IP on a new row. The supported IP format is Classless Inter-Domain Routing (CIDR) block format, xxx.xxx.xxx.xxx/yy. If you specify only xxx.xxx.xxx.xxx, mask /32 is automatically added. |

# Cloud Deployment CLI

The Cloud Deployment CLI performs tasks such as managing a solution, monitoring the status of all runtimes in a solution, promoting assets from one stage to another, and so on. The CLI supports the following modes:

- **Interactive Mode**: To start the CLI in interactive mode, run the following command:

  ```
  wmcc-cli --mode interactive
  ```

- **Normal Mode**: To start the CLI in normal mode, run the following command:

```
wmcc-cli <commands> [options]
```

## Install Cloud Deployment CLI

### Overview

You install the Cloud Deployment CLI (wmcc-cli) using the NPM registry. This section describes tasks such as installing the Cloud Deployment CLI and specifying credentials to connect to Cloud Deployment.

### Actors

Administrators

### Before you begin

Before you install wmcc-cli from the NPM registry, do the following:

- Install the NodeJS runtime environment version 10.13 or later on your computer along with NPM (Node's package manager).

  The Cloud Deployment CLI is available for the following operating system environments:

  - Windows
  - Linux
  - macOS

- Open a command line interface and type the following command to install wmcc-cli:

  ```
  npm install -g wmcc-cli
  ```

### Specify the credentials to connect to Cloud Deployment

To connect to Cloud Deployment, you must configure the credentials in one of the following ways:

- Specify the credentials in `config.json` file.

  - Create a `config.json` file under `%appdata%/wmcc-cli/` location for Windows or `/home/wmcc-cli/` in Linux.
  - Add new profiles in the `config.json` file.

    The following is an example of default and referenced profile in the configuration file.

    ```
    {
        "default": {
            "url": "https://{subdomain}.webmethodscloud.com",
            "userName": "userName",
            "password": "password”
        },
    ```

```
    "someotherProfile": {
        "url": "https://{subdomain}.webmethodscloud.com",
        "userName": "userName",
        "password": "password”
    }
}
```

By default, the 'default' settings are read from the configuration file. To enable the referenced profile, run the following command:

```
wmcc-cli -profile someotherProfile
```

- Specify credentials as runtime arguments.
  - When you start the CLI, credentials are passed as a runtime argument. For example:

    ```
    wmcc --mode interactive --url <url> --userName <userName>
    --password <password>
    ```

**Note:**
Runtime arguments will have the highest priority. However, If runtime arguments are not passed, then the default profile from `%appdata%/wmcc-cli/config.json` is used.

## Cloud Deployment CLI Reference

The following table describes the commands you use to perform various scenarios in the CLI interface.

### Commands for viewing alerts in Cloud Deployment

Run the following command to view alerts.

```
alert list [options]
```

where the options are:

| --Name | Description |
|---|---|
| --solutionName <solutionName> | Filter the alerts that belong to a particular solution. |
| --stageName <stageName> | Filter the alerts that belong to a particular stage. Supported values are development, test, live, prelive. |
| --alertName <alertName> | Filter the alerts by name . |
| --runtime <runtime> | Filter the alerts by Instances / Node name in a solution. |
| --severity <severity> | Filter the alerts based on the severity. Default supported values are info, warning, and critical. |
| --view <view> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format. |

| --Name | Description |
|---|---|
| | Example:<br>`alert list --severity critical` |
| --view <json> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`alert list --severity critical --view json`<br>**Note:**<br>JSON output format loads complete information that contains additional field than the table output format. |

### Commands for listing the assets in LAR

Run the following command to list all the assets available in the LAR.

```
asset-repo list-assets <solutionName> <nodeName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <view> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`asset-repo list-assets DemoSoln IS development` |

### Commands for creating solutions

Run the following command to create solutions.

```
solution create [options] <stagename>
```

where the options are:

| --Name | Description |
|---|---|
| --inputFile <fileName> | Provide the name of the file that contains the input data.<br>`solution create --inputFile /home/etc/createSolution.json <stagename>` |

| --Name | Description |
|---|---|
| | The file must contain a valid solution name, the name of the webMethods Integration Server and Universal Messaging instances, number of CPU Cores, and Memory characteristics of the hardware to support each service in the solution. |

Example 1 : Creating Type 1 solution without cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Demo",
   "description": "Demo Solution",
   "solutionType": 1,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams
.enabled=true",
 "packages.WmJDBCAdapter.enabled=true"]
     }
    }]
   }
  }
 }
}
```

Example 2: Creating Type 1 solution with stateful cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Demo",
   "description": "Demo Solution",
   "solutionType": 1,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
     "replicaCount": "2",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "dependencies": ["DB","tc"],
     "clusterType": "stateful",
     "isClustered": true,
     "statefulCluster": true,
```

| --Name | Description |
|---|---|

```
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "TERRACOTTA": [{
        "name": "tc",
     "isClustered": true,
     "statefulCluster": false,
     "replicaCount": "2",
     "enabled": true,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "4"
      }
     },
     "version": "10.4",
     "env": {
      "packages": [],
      "monitoringTools": []
     }
    }]
   }
  }
 }
}
```

Example 3: Creating Type 1 solution with stateless cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Demo",
   "description": "Demo Solution",
   "solutionType": 1,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
     "replicaCount": "2",
     "dependencies": ["DB"],
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "clusterType": "stateless",
     "isClustered": true,
     "statefulCluster": false,
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
```

| --Name | Description |
|---|---|

```
    }]
   }
  }
 }
}
```

Example 4: Creating Type 2 solution without cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution2",
   "description": "sample solution",
   "solutionType": 2,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams
.enabled=true",
 "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }]
   }
  }
 }
}
```

Example 5: Creating Type 2 solution with stateful cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution2",
   "description": "sample solution",
   "solutionType": 2,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
```

| --Name | Description |
|---|---|

```
     "clusterType": "stateful",
     "replicaCount": "2",
     "isClustered": true,
     "statefulCluster": true,
     "dependencies": ["DB", "UM", "tc"],
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "clusterType": null,
     "isClustered": false,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }],
    "TERRACOTTA": [{
        "name": "tc",
     "isClustered": true,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "4"
      }
     },
     "version": "10.4",
     "env": {
      "packages": [],
      "monitoringTools": []
     }
    }]
   }
  }
 }
}
```

Example 6: Creating Type 2 solution with stateless cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution2",
   "description": "sample solution",
```

| --Name | Description |
|---|---|

```
   "solutionType": 2,
   "productDefinitions": {
    "IS": [{
     "name": "IS",
     "replicaCount": "2",
     "clusterType": "stateless",
     "dependencies": ["DB","UM"],
     "isClustered": true,
     "statefulCluster": false,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "clusterType": null,
     "isClustered": false,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }]
   }
  }
 }
}
```

Example 7: Creating Type 3 solution without cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution3",
   "description": "Sample Solution",
   "solutionType": 3,
   "productDefinitions": {
    "IS": [{
     "name": "IS1",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
```

| --Name | Description |
|---|---|

```
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.
enabled=true"]
     }
    }, {
     "name": "IS2",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages":
["packages.WmCloudStreams.enabled=true",
 "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }]
   }
  }
 }
}
```

Example 8: Creating Type 3 solution with stateful cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution3",
   "description": "Sample Solution",
   "solutionType": 3,
   "productDefinitions": {
    "IS": [{
     "name": "IS1",
     "isClustered": true,
     "replicaCount": "2",
     "statefulCluster": true,
     "dependencies": ["UM", "DB", "tc1"],
     "clusterType": "stateful",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
```

<table>
<tr><th>--Name</th><th>Description</th></tr>
<tr><td></td><td><pre>
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
    }, {
     "name": "IS2",
     "isClustered": true,
     "statefulCluster": true,
     "replicaCount": "2",
     "dependencies": ["UM", "DB", "tc2"],
     "clusterType": "stateful",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages": ["packages.WmCloudStreams.
enabled=true", "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "isClustered": false,
     "statefulCluster": false,
     "clusterType": null,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }],
    "TERRACOTTA": [{
     "name": "tc1",
     "isClustered": true,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "4"
      }
     },
     "version": "10.4",
     "env": {
      "packages": [],
      "monitoringTools": []
     }
    }, {
     "name": "tc2",
     "isClustered": true,
     "resources": {
</pre></td></tr>
</table>

| --Name | Description |
|---|---|

```
      "limits": {
       "cpu": "1",
       "memory": "4"
      }
     },
     "version": "10.4",
     "env": {
      "packages": [],
      "monitoringTools": []
     }
    }]
   }
  }
 }
}
```

Example 9: Creating Type 3 solution with stateless cluster.

```
{
 "integration": {
  "landscapeDefinition": {
   "solutionName": "Solution3",
   "description": "Sample Solution",
   "solutionType": 3,
   "productDefinitions": {
    "IS": [{
     "name": "IS1",
     "isClustered": true,
     "replicaCount": "2",
     "statefulCluster": false,
     "dependencies": ["UM", "DB"],
     "clusterType": "stateless",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4",
     "env": {
      "packages":
["packages.WmCloudStreams.enabled=true",
 "packages.WmJDBCAdapter.enabled=true"]
     }
    }, {
     "name": "IS2",
     "isClustered": true,
     "statefulCluster": false,
     "replicaCount": "2",
     "dependencies": ["UM", "DB"],
     "clusterType": "stateless",
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
```

| --Name | Description |
|---|---|

```
     },
     "version": "10.4",
     "env": {
      "packages":
["packages.WmCloudStreams.enabled=true",
 "packages.WmJDBCAdapter.enabled=true"]
     }
    }],
    "UNIVERSALMESSAGING": [{
     "name": "UM",
     "isClustered": false,
     "statefulCluster": false,
     "clusterType": null,
     "resources": {
      "limits": {
       "cpu": "1",
       "memory": "2"
      }
     },
     "version": "10.4"
    }]

   }
  }
 }
}
```

## Commands for listing the solutions for a particular stage

Run the following command to list solutions for a particular stage.

```
solution list <stageName> [options]
```

where the valid stage name value includes development, test, prelive, and live.

Example:

```
solution list development --view json
```

## Commands for getting the solution and runtimes for a particular stage

Run the following command to get the solution and all runtimes for a particular stage.

```
solution get <solutionName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format. |

| --Name | Description |
|---|---|
| | Example:<br>`solution get DemoSoln development` |

### Commands for deleting the solution for a particular stage

Run the following command to delete the solution for a particular stage.

```
solution delete <solutionName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`solution delete DemoSoln development` |

### Commands for activating or deactivating the solution for a particular stage

Run the following command to activate and deactivate a solution for a particular stage.

```
solution update-status <solutionName> <stageName> <actionName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`solution update-stats DemoSoln development deactivate`<br>■ Allowed stage names are `development`, `test`, `live`, `prelive`.<br>■ Allowed action names are `activate` or `deactivate` |

### Commands for getting the status of all pods in a solution

Run the following command to get the status of all pods in a solution.

```
solution get-status <solutionName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`solution get-status DemoSoln development` |

## Commands for getting the license information for a particular tenant

Run the following command to get the license information for a particular tenant.

```
tenant get-license-info [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`tenant get-license-info` |

## Commands for getting the total CPU and memory utilization details for a particular tenant

Run the following command to get the total CPU and memory utilization details for a particular tenant.

```
tenant get-utilization-details [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`tenant get-license-info` |

## Commands for listing all the users

Run the following command to list all the users.

```
user list [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`tenant get-license-info` |

### Commands to get a particular user

Run the following command to get the user information.

```
user get <userId> [options]
```

where the options are:

| --Name | Description |
|---|---|
| --view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`user get <ID details>` |

### Commands for promoting Integration Server and Universal Messaging configurations from one stage to another

Run the following command to promote Integration Server and Universal Messaging configurations from one stage to another.

```
runtime promote-configuration <fromSolutionName> <fromNodeName>
<fromStageName> [toSolutioName] [toNodeName] [options]
```

**Note:**
If toSolutionName and toNodeName are provided as part of your command, then the CLI performs the cross solution promotion. However, if toSolutionName and toNodeName are not available, then the configuration will be promoted to higher stage on the sameIntegration Server or Universal Messaging node.

where the options are:

| --Name | Description |
|---|---|
| --propFile <fileName> | Properties file to perform variable substitution. |

| --Name | Description |
|---|---|
| | Example: Use the following command for promotion across same instance:<br>`runtime promote-configuration DemoSoln IS`<br>`development`<br>Example: Use the following command for cross solution promotion:<br>`runtime promote-configuration DemoSoln IS`<br>`development`<br>`DemoSoln1 IS1`<br>Promotes the configuration from the solution DemoSoln IS node to DemoSoln1 and IS1 node. |

### Commands for promoting Integration Server packages from one stage to another

Run the following command to promote Integration Server packages from one stage to another.

```
runtime promote-packages <fromSolutionName> <fromNodeName> <fromStageName>
[toSolutionName] [toNodeName] [options]
```

**Note:**
If toSolutionName and toNodeName are provided then the CLI performs the cross solution promotion. If toSolutionName and toNodeName are not available, then the packages will be promoted to higher stage on the same Integration Server instance.

where the options are:

| --Name | Description |
|---|---|
| --include <comma separated packages names> | Promotes only specified packages.<br>Example:<br>`--include package1,package2,package3` |
| --exclude <comma separated packages names> | Promotes all the packages other than the packages specified in the options.<br>Example:<br>`--exclude package1,package2,package3` |
| --propFile <fileName> | Provide properties file to perform variable substitution.<br>Examples:<br>Use the following command for promotion across same instance: |

| --Name | Description |
|---|---|
| | `runtime promote-packages DemoSoln IS development -- propFile /home/etc/var_sub.properties`<br>Use the following command for cross solution promotion:<br>`runtime promote-packages DemoSoln IS development TestSoln IS2`<br>Use the following command to replace a property in a specific composite asset.<br>`<propertyName>/<compositeAssetName>`<br>**Note:**<br>The replacement properties should be in the following format:<br>`<propertyName>/<compositeAssetName>/<assetName>`<br>Example for replacing properties of a package:<br>`activatePkgOnInstall/TestODataService=false`<br>where TestODataService is the package name whose property activatePkgOnInstall" is assigned with value "false".<br>Example for replacing properties of a service of a package:<br>`serverName/TestODataService/JDBC_Connection.ODataService=localhost`<br>where `JDBC_Connection.ODataService` is a service under package `TestODataService`, whose parameter `serverName` is assigned with value localhost |

## Commands for printing the list of exposed services

Run the following command to print the list of exposed services.

```
runtime get-exposed-is-serices <solutionName> <nodeName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example: |

| --Name | Description |
|---|---|
| | `runtime get-exposed-is-services DemoSoln IS development` |

### Commands for listing all the queues

Run the following command to list all the queues in the Universal Messaging instance.

```
um list-queues <solutionName> <nodeName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um list-queues DemoSoln UM development` |

### Commands for getting the queue information

Run the following command to retrieve pushed, popped, and memory usage of the queue.

```
um get-queue <solutionName> <nodeName> <stageName> <queueName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um get-queue DemoSoln UM development dummyQueue` |

### Commands for getting the queue details

Run the following command to retrieve the queue details.

```
um get-queue-details <solutionName> <nodeName> <stageName> <queueName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um get-queue-details DemoSoln UM development dummyQueue` |

### Commands for listing all the channels in the Universal Messaging instance

Run the following command to list all the channels in a Universal Messaging instance.

```
um list-channels <solutionName> <nodeName> <stageName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um list-channels DemoSoln UM development` |

### Commands for getting the channel information

Run the following command to get the channel information.

```
um get-channel <solutionName> <nodeName> <stageName> <channelName> [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um get-channel DemoSoln UM development dummyChannel` |

### Commands for getting the channel details

Run the following command to get the channel details.

```
um get-channel-details <solutionName> <nodeName> <stageName> <channelName>
 [options]
```

where the options are:

| --Name | Description |
|---|---|
| -view <viewType> | By default, outputs the response in a table view mode. The CLI supports both table and JSON output format.<br>Example:<br>`um get-channel-details DemoSoln UM development dummyChannel` |