

webMethods Integration Cloud Help

Version 3.6.0

July 2017

This document applies to webMethods Integration Cloud Version 3.6.0 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.

Copyright © 2014-2017 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at <http://softwareag.com/licenses>.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at <http://softwareag.com/licenses> and/or in the root installation directory of the licensed product(s).

Table of Contents

Welcome to webMethods Integration Cloud.....	9
Registration.....	11
Creating an Account.....	12
Securing your Account.....	14
Users.....	15
Adding Users.....	16
Updating Users.....	19
Resetting Passwords.....	20
User Profile.....	20
Security Question.....	21
Change Password.....	21
Access Profiles.....	23
Adding or Updating Access Profiles.....	24
Containers.....	29
Managing Repositories.....	31
Viewing Tag Details.....	32
Managing Services.....	33
Single Sign-On.....	35
Configuring SAML Settings for Single Sign-On.....	37
Advanced Security.....	43
Add Keystore.....	44
Add Truststore.....	45
Add Partner Certificate.....	46
Stage Management.....	47
Applying Access Profiles to a Stage.....	48
Company Information.....	51
Updating Company Information.....	52
Password Policies.....	55
Updating Password Policy Settings.....	56
Applications.....	59
Upgrade.....	60
Creating SOAP Applications.....	62
REST Applications Overview.....	63
Creating REST Applications.....	65

Accounts	73
Adding or Editing Accounts.....	74
Account Configuration Details	77
Alfabet.....	79
Apache Solr Search.....	80
Amazon Simple Notification Service (SNS).....	81
Amazon Simple Queue Service (SQS).....	82
Amazon Simple Storage Service (S3).....	83
Avalara AvaTax.....	84
Concur.....	85
Coupa.....	87
Cumulocity.....	88
CloudStreams Connector for NetSuiteTM.....	89
File Transfer Protocol (FTP/FTPS).....	90
Google Apps Admin.....	91
Google BigQuery.....	92
Google Calendar.....	93
Google Contacts.....	95
Google Drive.....	96
Google Cloud Pub/Sub.....	97
Google Cloud Storage.....	98
Google Sheets.....	99
Magento eCommerce Platform.....	100
Marketo.....	101
Microsoft Dynamics CRM.....	102
OData v2.0.....	103
OData v4.0.....	105
On-Premises Applications.....	106
REST Application Account Configuration Details.....	106
Salesforce.....	108
SAP Cloud for Customer(C4C) OData v2.0.....	111
SAP S/4HANA Marketing Cloud.....	112
Secure File Transfer Protocol (SFTP).....	114
ServiceNow Enterprise Service Management.....	117
Slack.....	118
SOAP Application Account Configuration Details.....	119
Strikelron Contact Verification.....	121
SuccessFactors HCM.....	122
SugarCRM.....	124
Zendesk.....	125
Operations	127
Adding or Editing Operations.....	129
FTP Predefined Operations.....	131

getFile.....	131
listFiles.....	132
deleteFiles.....	132
putFile.....	133
renameFile.....	134
SFTP Predefined Operations.....	134
cd.....	135
chgrp.....	136
chmod.....	136
chown.....	137
get.....	137
ls.....	138
mkdir.....	139
put.....	139
pwd.....	140
rename.....	140
rm.....	141
rmdir.....	141
symlink.....	141
Develop.....	143
Point-to-Point Integrations.....	144
Orchestrated Integrations.....	146
Pipeline and Signatures.....	159
Built-In Services.....	162
Date.....	162
Summary of Date services.....	166
calculateDateDifference.....	167
compareDates.....	168
dateBuild.....	169
dateTimeBuild.....	170
dateTimeFormat.....	172
getCurrentDateString.....	173
incrementDate.....	174
Document.....	176
Summary of Document services.....	176
findDocuments.....	177
insertDocument.....	178
deleteDocuments.....	178
documentListToDocument.....	179
documentToDocumentList.....	180
groupDocuments.....	182
documentToBytes.....	183
bytesToDocument.....	184
sortDocuments.....	185

List.....	186
Summary of List services.....	186
appendToDocumentList.....	186
appendToStringList.....	187
sizeOfList.....	188
stringListToDocumentList.....	189
Math.....	190
Summary of Math services.....	190
absoluteValue.....	191
addFloatList.....	191
addFloats.....	192
addIntList.....	193
addInts.....	194
divideFloats.....	194
divideInts.....	195
max.....	196
multiplyFloatList.....	196
multiplyFloats.....	197
multiplyIntList.....	198
multiplyInts.....	199
randomDouble.....	200
roundNumber.....	200
subtractFloats.....	201
subtractInts.....	202
String.....	202
Summary of String services.....	202
base64Decode.....	204
base64Encode.....	205
bytesToString.....	206
concat.....	206
indexOf.....	207
length.....	207
lookupDictionary.....	208
makeString.....	208
messageFormat.....	208
numericFormat.....	209
padLeft.....	211
padRight.....	212
replace.....	213
stringToBytes.....	213
substring.....	214
tokenize.....	214
toLowerCase.....	215
toUpperCase.....	215
trim.....	216

URLDecode.....	216
URLEncode.....	217
fuzzyMatch.....	217
isNumber.....	218
isAlphanumeric.....	219
isNullOrBlank.....	219
isDate.....	220
substitutePipelineVariables.....	220
compareStrings.....	221
Flow.....	222
Summary of Flow services.....	222
getLastError.....	222
getSessionInfo.....	224
countProcessedDocuments.....	225
logCustomMessage.....	225
Hashtable.....	226
Summary of Hashtable services.....	226
containsKey.....	226
createHashtable.....	227
get.....	227
listKeys.....	228
put.....	228
remove.....	228
size.....	229
Flat File.....	229
Summary of Flat File services.....	229
delimitedDataBytesToDocument.....	230
delimitedDataStreamToDocument.....	233
delimitedDataStringToDocument.....	236
documentToDelimitedDataBytes.....	239
documentToDelimitedDataStream.....	241
documentToDelimitedDataString.....	243
JSON.....	245
Summary of JSON services.....	245
documentToJSONBytes.....	246
documentToJSONStream.....	246
documentToJSONString.....	247
jsonBytesToDocument.....	247
jsonStreamToDocument.....	248
jsonStringToDocument.....	249
XML.....	249
Summary of XML services.....	249
documentToXMLBytes.....	250
documentToXMLStream.....	254
documentToXMLString.....	258

xmlBytesToDocument.....	262
xmlStreamToDocument.....	269
xmlStringToDocument.....	275
IO.....	281
Summary of IO services.....	281
bytesToStream.....	282
streamToBytes.....	282
Integration Details.....	283
Recipes.....	286
Document Types.....	287
Reference Data.....	290
Reference Data Signature.....	291
Monitor.....	295
Dashboard.....	296
Execution Results.....	297
Audit Log.....	298

Welcome to webMethods Integration Cloud

webMethods Integration Cloud is Software AG's Integration Platform as a Service (iPaaS) offering. It is a part of the *Software AG Live* family of products. The solution 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-premise 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-premise 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. You can 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 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 package and price tiers.

1 Registration

■ Creating an Account	12
■ Securing your Account	14

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

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

To create a new User Account

1. From the Integration Cloud login screen, click **New Registration**.
2. 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. You can change the value after the user is created from the Settings > Users screen.
Last Name	Type your last name. You can change the value after the user is created from the Settings > Users screen.
Company	Type your company name. You can change that later from the Settings > Company Information screen.
Country	Select your country from the drop-down list box. You can change that later from the Settings > Users screen.
State or Province	Type your State or Province. You can change that later from the Settings > Users screen.
Phone	Type your phone number.

Field	Description
	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. 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.
Email Address	Type your email address. 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.
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 .
Promo Code	Enter a valid promotion code if you have one, for availing subscription benefits.

- 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: 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.

Securing your Account

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.

2 Users

■ Adding Users	16
■ Updating Users	19
■ Resetting Passwords	20
■ User Profile	20

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, update existing user information, and reset a user's password.

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

You can add users for accessing the platform. The operations that a user can perform is determined by their *Access Profile*.

To add a user

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	<p>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:</p> <ul style="list-style-type: none"> ■ Administrator - Provides permissions needed by the System Administrator. ■ Regular User - Provides permissions that are more appropriate for normal users. <p>By default, the system administrator can change the Administrative Permissions associated with each Access</p>

Field	Description
	<p>Profile (except the above mentioned Administrator Access Profile), and can add additional Access Profiles, as needed.</p> <p>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 for more information.</p>
Employee Number	Optional identification number for each employee.
Email	<p>Email address of the user. User credentials will be sent to the specified email address.</p> <p>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.</p>
Username	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.
Federation Id	<p>Enter the Federation ID if your Identity Provider passes the Federation ID for Single Sign-On.</p> <p>Note: 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.</p>
Partner	<p>Select this option if the user is a Partner user.</p> <p>Note: If Allow User Interface Access permission available under Access Profile > Administrative Permissions > Account Controls is not enabled, a Partner User can still perform on-premise tasks.</p>
Force password change on first login	Select this option to force the user to change the login password when logging in for the first time.

Field	Description
	Note: This option is available only when creating a new user.
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. 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.
Time Format	Select a 12-hour clock (hh:mm a) with AM/PM, or a 24-hour clock (HH:mm).
Locale	Select the user's locale setting. This setting determines the format for numbers, decimal fields, and percentages.

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.

Field	Description
State/Province	State or province for the user.
Postal/Zip Code	Postal or ZIP Code for the user.
Country	Country for the user.

- 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 **<User name> > 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

- From the Integration Cloud navigation bar, click **Settings > Users**.
- Select a user from the list, and then click **Edit**.
- Make necessary modifications. See "[Adding Users](#)" on page 16 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.

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.

Field	Description
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.

Resetting Passwords

To reset a User password

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 (*<Logged in User>* > **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

To update the Security Question and Answer

1. From the Integration Cloud navigation bar, go to **<User name> > 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 Information** screen.

Change Password

To change your password

1. From the Integration Cloud navigation bar, go to **<User name> > 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.

3 Access Profiles

- Adding or Updating Access Profiles 24

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.

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.

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** tab, 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 computer on a network outside of the specified range, access to the platform is denied.

Note: 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

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** tab, select the operations a user can perform in order to view, create, update, upgrade, administer, and delete permissions and to allow the user to customize selected aspects of the platform.

Field	Description
User and Ownership Controls	<p>User Management - Select this option if you want to add, update, delete users, or assign users to Access Profiles.</p> <p>Access Control - Select this option if you want to allow a user to modify Access Profiles, specify user application access rights, manage Access Profiles, or specify the password policy.</p>

Field	Description
	Manage Personal Setup - Select this option if you want to allow a user to modify the personal information.
Account Controls	<p>Manage Company Capabilities - Select this option if you want to allow users to modify the company information.</p> <p>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.</p> <p>Note: If the Allow User Interface Access permission is not enabled for a user and if the user is a Partner user, that user will still be able to perform on-premise tasks.</p>
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 Monitor > Audit Log .
Functional Controls	Select the required options under Stages, Advanced Security, Application, Accounts, Operations, Reference Data, Document Types, Integrations, Dashboard, and Container . You must select the required permissions to administer, access, create, update, upgrade, delete, execute, or deploy those functions.
	Note: The Container tab and Container related Administrative permissions are available only if you have the required license for Containers.

- On the **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 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 Integration Server group, you will not be able to invoke Integration Server services. 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 not automatically assigned to the Integration Server Administrators ACL group. If you do not enter any user groups in the **Container User Groups** field, but have configured 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 Integration Server services from Integration Cloud.

7. Click **Apply**.

4 Containers

■ Managing Repositories	31
■ Viewing Tag Details	32
■ Managing Services	33

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. You can create and upload images by running the `is_container` scripts available with the Integration Server installation from `Integration_Server_directory/docker`, store the images in a registry on Integration Cloud, and manage those images from the Integration Cloud user interface. Images or a repository can be 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.

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 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 now 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 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 Integration Server Administrators ACL group. If you do not map an Access Profile to an Integration Server group, you will not be able to invoke Integration Server services.

Note: You must 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. Enabling the CSRF security feature will prevent CSRF attacks. 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 a 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](#) 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**, number of **Docker Containers** to instantiate, and the **Docker Container Port**.
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, 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 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 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 a stage, the stage name, 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 **Edit** 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, the deployment stage, the repository, 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**. 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 **Docker Services** (webMethods Integration Server Administrator > **webMethods Cloud > Docker Services**) exposed by webMethods Integration Server packages. The **Base URL** is the **webMethods Cloud Docker Service**.

Note: You must 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. Enabling the CSRF security feature will prevent CSRF attacks. 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.


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. Volume name 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.
- **Docker Container Port** - Enter the container port number.

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.

5 Single Sign-On

- [Configuring SAML Settings for Single Sign-On](#) 37

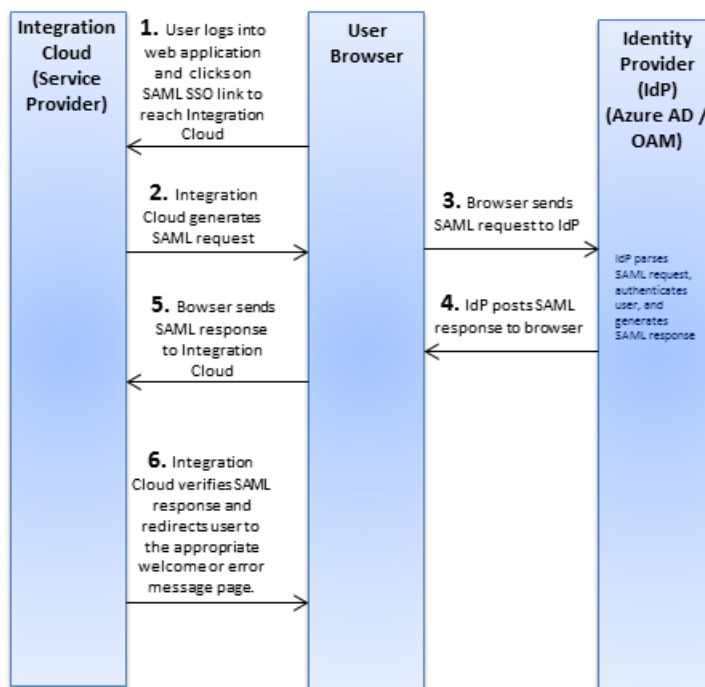
Single sign-on is a process that allows users to access all authorized network resources without having to log in separately to each resource.

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:



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: 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

The **Single Sign-On Configuration** screen allows you to configure SAML 2.0 settings for single sign-on.

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**.

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: None . 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. You need to open this link in a browser to view the Identity Provider authentication page.
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. Note: For Oracle Access Manager (OAM), apply it in the Assertion Consumer Service URL field. For Microsoft Azure, apply it in the Reply URL field. 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. Note: In an Identity Provider initiated single sign-on scenario, you must set the RelayState value in the Identity Provider. You can test the Identity Provider initiated SSO only after configuring the RelayState. Note: For Oracle Access Manager (OAM), apply the "RelayState" value as the "Return URL" in the Identity Provider initiated URL. 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.

Field	Description
	For Okta, apply it in the Default RelayState field.
Identity Provider Configuration	
SAML Request Issuer URL	The Integration Cloud (Service Provider) URL used to access this tenant. This URL acts as the Service Provider ID.
	<p>Note: For Oracle Access Manager (OAM), apply it in the Provider ID field.</p> <p>For Microsoft Azure, apply it in the Identifier field.</p> <p>For Okta, apply it in the Audience URI (SP Entity ID) field.</p>
Identity Provider Details	<p>Specify how you want to define the Identity Provider details.</p> <p>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.</p> <p>Select Load From Identity Provider Metadata and select the metadata file to upload the IdP details.</p>
Issuer	<p>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. 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.</p>
	<p>Note: For Oracle Access Manager (OAM), copy the URL from the Provider Id field under Federation Settings.</p> <p>For Microsoft Azure, copy the URL from the Issuer URL field.</p> <p>For Okta, copy the URL from the Identity Provider Issuer field.</p>
	<p>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.</p>
Identity Provider Certificate	<p>This is the authentication certificate (a valid x509 issuer certificate) issued by your Identity Provider and is required to sign and verify SAML messages. If you have selected Enter Manually for Identity Provider Details, select Browse and upload a</p>

Field	Description
	<p>file that contains the Identity Provider's certificate. 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.</p>
Identity Provider Login URL	<p>This is the URL used to log in to the Identity Provider. If you have selected Enter Manually for Identity Provider Details, type the URL that will be used to log in to the Identity Provider.</p> <p>Note: For Oracle Access Manager (OAM), the URL is <code>http://<oamserverhost name>:14100/oamfed/idp/samlv20</code>.</p> <p>For Microsoft Azure, copy the URL from the Single sign-on service URL field.</p> <p>For Okta, copy the URL from the Identity Provider Single Sign-On URL field.</p> <p>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.</p>
User ID Type	<p>Determines the type of identifier.</p> <p>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.</p> <p>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.</p>
User ID Location	<p>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. Select Subject if the User ID is located in the <Subject> statement of the assertion. 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</p>

Field	Description
	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 either login fails or a new user is created, depending on the Create Users setting.
Create Users	<p>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.</p> <p>Attribute for First Name - The name of the SAML attribute that designates the user's first name.</p> <p>Attribute for Last Name - The name of the SAML attribute that designates the user's last name.</p> <p>Attribute for Email - The name of the SAML attribute that designates the user's email address.</p> <p>Default Access Profile - This field is used to specify the default Access Profile for the created user.</p> <p>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).</p>

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

6 Advanced Security

- Add Keystore 44
- Add Truststore 45
- Add Partner Certificate 46

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 in the **Settings > Advanced Security** tab and can use them to secure your SOAP Application Account. 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 **Settings > Advanced Security > Keystores > Add Keystore**.

To add a Truststore, from the Integration Cloud navigation bar, click **Settings > Advanced Security > Truststores > Add Truststore**.

To add a Partner Certificate, from the Integration Cloud navigation bar, click **Settings > Advanced Security > 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 a SOAP Application.

To add a Keystore

1. From the Integration Cloud navigation bar, click **Settings > Advanced Security > 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 a SOAP 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 a SOAP Application.

To add a Truststore

1. From the Integration Cloud navigation bar, click **Settings > Advanced Security > 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 a SOAP 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 a SOAP Application.

To add a Partner Certificate

1. From the Integration Cloud navigation bar, click **Settings > Advanced Security > 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 a SOAP Application.

7 Stage Management

- Applying Access Profiles to a Stage 48

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.

Integration Cloud provides ways to manage the life cycle of an Integration development. 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 life cycle development. To aide these activities, Integration Cloud provides you with **Stages**.

Your organization is a tenant in the platform. When you log in to the platform, you log into your organization's tenancy. When you set up a stage, an environment is created for testing and executing the Integrations in the production environment. You can also schedule the Integrations to be executed in a specific stage.

A predefined set of stages is allowed, each representing an activity in the Integration life cycle development. They are:

- Development
- Test
- Pre-Live
- Live

You can create or delete stages only in a particular order. By default, every user gets a **Development Stage**. In the Development Stage, you can create, update, delete, or view Integrations. 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: Users who have the **Administer** permission under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Stages** can add or delete the Stages information.

To add a new stage, click **Add New Stage**.

Applying 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** to add the next stage. Multiple boundary arrows indicate that more stages can be added.

Note: The Accounts drop down list in the Live stage lists all the Accounts defined in the Development stage. Accounts that are not present in the Live stage are highlighted. Click on such an Account in the Live stage to view the **Edit Accounts** page. Only active or enabled Accounts are listed in the drop down list.

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 **Settings > Stage Management**. All stages added including the Development stage are displayed. Initially, before any other stages are added, the Development stage is displayed.
2. Click the Access Profiles icon on a stage and select the Access Profiles you want to apply to the stage.

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 stage.

Note: You can pull Integrations from all other stages except from the **Development** stage. An Integration depends on the Action, Trigger, and Account. When an Integration is pulled, all its dependents will also be pulled and copied to that stage. If you update only the action or trigger, only those will be pulled into the next stage.

8 Company Information

■ Updating Company Information	52
--------------------------------------	----

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

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. On the **Basic** tab, 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. 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	Enter an address or comma-separated email addresses to receive system notifications. Such notifications can occur, for example,

Field	Description
Email Addresses	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. On the **Advanced Information** tab, 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. hh:mm a - 12-hour clock - 3:30 AM, 3:30 PM HH:mm - 24-hour clock - 3:30, 15:30
Date Format	Choose a date format from the drop down list. mm is "Month", dd is "Day", yyyy is Year and the delimiters are:(/) slash or stroke(-) dash or hyphen(.) period, dot, or full stop.
Default Locale	The field displays the user's initial locale setting and determines the format for numbers, decimal fields, and percentages. Choose any other locale from the drop down list.
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 .

9 Password Policies

- Updating Password Policy Settings 56

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.

Click **Edit** to modify Password Policy information.

Updating Password Policy Settings

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

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** screen, make the necessary modifications.

Field	Description
Minimum Length	Select the minimum number of characters in the password. Default: 6 characters. Range: 6-10 characters.
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. Default: No Restrictions.
Expires in	Select the number of days the password will remain valid before the user will be prompted to change it. Default: 90 Days. Range: 15, 30, 60, 90, 120 days, Never. 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 <i>Never</i> .
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. Default: Last Password. Range: Last 2-5 passwords.

Field	Description
Minimum Age	Select the number of days that must pass before a user can change passwords. Default: No Minimum. Range: 1-5 Days.
Inactive 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. Default: 30 Minutes. Range: 15 minutes, 30 minutes, and 1 hour.
Account Lockout Threshold	<p>Select the number of login attempts before the account is locked out. Default: 5 failed tries. Choices: 3-10 failed tries, No limit.</p> <p>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 <i>Account Lockout Duration</i> field. The login limit is defined by the <i>Password Policy</i>.</p>
Account Lockout Duration	Select the length of time that an account is locked out. Default: 15 minutes. Choices: 5, 10, 15, 30 minutes, 1 hour, Disable.
Record Information	<p>For audit purposes, the following information is displayed after you save the record:</p> <p>Last Modified By <username> on {date} <time> Created by System.</p>

4. Click **Apply**.

10 Applications

■ Upgrade	60
■ Creating SOAP Applications	62
■ REST Applications Overview	63
■ Creating REST Applications	65

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

You can also create SOAP and REST Applications from this page. To create a SOAP Application, click **Add New Application**, select **Create SOAP Application**, and then click **OK**. To create a REST Application, click **Add New Application**, select **Create REST Application**, and then click **OK**. FTP and SFTP Applications are available that allow Integration Cloud to connect to FTP and SFTP servers.

On-Premise applications loaded from on-premise systems are also listed in the **Applications** page but you will not be able to create Accounts or Operations for on-premise applications. Those can be uploaded only from the webMethods Integration Server. Further, when you upload services as part of an application from the on-premise 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 **Applications** page, you can create Accounts and Operations for an Application and Integrations between different SaaS applications. For an Application, you can click **Accounts**, **Operations**, or **Integrations** if you want to create or edit them for that Application. For REST Applications, the **Document Types** link allows you to create new Document Types. Document Types created for a REST Application appears only in the **Document Types** panel for the selected REST Application.

If you have the required access privileges, you can also click the **Upgrade** button to upgrade Application assets (Accounts, Operations, and the associated Integrations) from a lower version to a higher version.

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 access privileges under **Settings > Access Profiles > Administrative Permissions > Functional Controls** can create, update, administer, execute, deploy, or delete the Accounts, Operations, Integrations, Stages, Advanced Security, Document Types, and Reference Data information.

Upgrade

The upgrade feature in Integration Cloud allows you to 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. Currently, upgrade functionality is available only for the Salesforce CRM Application.

Note: Users who have the **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.

Creating SOAP Applications

The Custom SOAP Application enables you to access third party Web Services hosted in the cloud or on-premises environment. The Custom SOAP Application uses a WSDL to create consumer operations and can be a trigger or action or both.

The following features are supported for Custom SOAP Applications:


- A Web Service implementation that follows the WS-I Basic Profile 1.1 specification.
- Custom SOAP Applications can be created by uploading a WSDL file or by using a valid WSDL URL that can be accessed over a network.
- Web Services that use WS-Security. Custom SOAP Applications can be created with WSDLs that are annotated with WS-Security Policy/Policies.
- Web Services 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.

Custom 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 Custom SOAP Applications with WSDLs having non-WS-Security Policies, exceptions may appear while executing Integrations.
- Manual addition of WS-Security Policies in a Custom SOAP Application is not supported. Custom 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 Custom SOAP Application

1. From the Integration Cloud navigation bar, click **Applications**.
2. Click **Add New Application > Create SOAP Application**.

3. Provide a name and description of your Custom SOAP Application. The description you enter here will appear in the **Applications** page. Required fields are marked with an asterisk on the screen.
4. Click **Browse** next to the **Application Icon** if you want to select a different icon for your custom SOAP Application. The icon must be a PNG file and the size cannot exceed 50 KB.
5. 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 Web Service. Select **File**, and then click **Browse** if you want to select the WSDL from your local file system. You can click the  icon beside the **Browse** button if you want to add separate elements of a service definition after import, as WSDLs or XSDs to the primary WSDL.

Note: Ensure that you add the primary WSDL as the first WSDL, and then add the separate elements of the service definition, for example, dependent WSDLs and XSDs to the primary WSDL.

6. Enter the user name and password in the **Authentication** section if authentication is required to access the WSDL URL.
7. Click **Next** to review the details you have entered.
8. Click **Finish** to create the Custom SOAP Application.

REST Applications Overview

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 a 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/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/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 REST Applications

These screens allow you to define a REST Application, define Resources and Actions, and then create a REST Application. See ["REST Applications Overview" on page 63](#) 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 **Applications**.
The **Applications** page appears.
2. From the **Applications** page, click **Add New Application**, select **Create REST Application**, and then click **OK**.

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.
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, go through 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	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

Field	Description
	<p>myREST.myRESTResource, the path is “/myREST.myRESTResource”.</p> <p>You can define parameters in the resource path by enclosing each parameter in { } brackets. For example, to define the path parameter for a user, specify the resource path as /user/{userID}. To define multiple path parameters like "department" and "item", specify the resource path as /store/{departmentID}/{itemID}.</p>

Note: 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**.

- Click **Add** to create the Resource. You can **Edit** or **Delete** the Resource from the **Define Resources and Actions** page.
- 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	<p>Select an HTTP Method.</p> <ul style="list-style-type: none"> ■ GET - Reads or retrieves a representation of a resource. For example, GET <endpointurl>/addresses/2 will retrieve an address with an ID of 2. ■ PUT - Updates an existing resource. For example, PUT <endpointurl>/addresses/3 will modify the address with an ID of 3. ■ POST - Creates a resource. For example, POST <endpointurl>/addresses will create a new address. ■ DELETE - Deletes a resource identified by an URI. For example, DELETE <endpointurl>/addresses/4 will delete an address with an ID of 4.

Field	Description
Description	Type an optional description for the Action.
Request Parameter	<p>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:</p> <p>Name - Type the parameter name.</p> <p>Value - Type a value for the parameter.</p> <p>Parameter Type - Select the parameter's type, which determines how the parameter should be used, for example, <code>QUERYSTRING_PARAM</code> or <code>URI_CONTEXT</code>.</p> <p>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.</p> <p><code>QUERYSTRING_PARAM</code> parameters are passed as the query component of a REST resource invocation request.</p> <p>For example, if the URI is <code>https://api.twitter.com/1.1/users/show.json?screen_name=twitterdev</code>, the resource path will be <code>/users/show.json</code>, <code>screen_name</code> is the name of the parameter, <code>twitterdev</code> is the value of the parameter, and the parameter type is <code>QUERYSTRING_PARAM</code>.</p> <p><code>URI_CONTEXT</code> 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 <code>{}</code> annotation.</p> <p>For example, if the URI is <code>https://api.twitter.com/1.1/users/{id}</code>, the Resource path will be <code>/users/{id}</code>, the parameter type will be <code>URI_CONTEXT</code>, the parameter name will be <code>id</code>, and the value could be the user id, for example, either 1, or 2, or 3.</p>

Field	Description
Request Header	<p>Required - Select this option if you want this parameter to be made mandatory while creating an Integration.</p> <p>HTTP headers allow the client and the server to pass additional information with the request or the response. Click Add Header to add a request HTTP header. In the Add Header dialog box, complete the following fields:</p> <p>Name - Type the Header name.</p> <p>Value - Type a value for the Header.</p> <p>Required - Select this option if you want this Header to be made mandatory while creating an Integration.</p>
Request Body	<p>In the Request Body pane, complete the following fields:</p> <p>Document Type - Select a Document Type for the request body or click Create Document Type to create a new Document Type. See "Document Types" on page 287.</p> <p>Note: Document Types created for a REST Application does not appear in the Develop > Document Types screen but appears only in the Document Types panel for the selected REST Application.</p> <p>Document 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 document 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 document content type. These options allow you to control the content in an HTTP request body.</p> <p>Note: Currently, only application/json and application/xml are the supported document content types.</p>

Field	Description
Response Header	<p>In the Response Header pane, click Add Header to add a Response HTTP header. Complete the following fields:</p> <p>Name - Type the Header name.</p> <p>Value - Type a value for the Header.</p> <p>Required - Select this option if you want this Header to be made mandatory while creating an Integration.</p>
Response Body	<p>In the Response Body pane, complete the following fields:</p> <p>HTTP Code - Type an HTTP status code to indicate the status of the response.</p> <p>Document Type - Select a Document Type for the Response Body or click Create Document Type to create a new Document Type. See "Document Types" on page 287.</p> <div data-bbox="618 1031 1304 1230" style="background-color: #f0f0f0; padding: 5px;"> <p>Note: Document Types created for a REST Application does not appear in the Develop > Document Types screen but appears only in the Document Types panel for the selected REST Application.</p> </div> <p>Document 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 document 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 document content type. These options allow you to control the content in an HTTP response body.</p> <div data-bbox="618 1577 1304 1703" style="background-color: #f0f0f0; padding: 5px;"> <p>Note: Currently, only application/json and application/xml are the supported document content types.</p> </div>

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 **Applications** page.

Note: To edit the REST Application, click the REST Application link and then click **Edit Application**. After editing the Application, the **Update REST Application** window appears, which provides a summary of the impacted Accounts, Operations, and Integrations. To delete the REST Application, click **Delete Application**.

11 Accounts

- Adding or Editing Accounts 74

This screen lists all the available Accounts created for an Application.

If you select an Account for an FTP, SFTP, custom SOAP, or on-premise 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 stage appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the stage appears in green color and if an Account is not configured in a particular stage, that stage appears in white color.

For on-premise Applications, the Account can be used to execute services on the on-premise Integration Server. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for information on how to configure 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 Stage Management.

You can create, edit, or delete an Account for a particular application from this screen.

Note: Users who have the required permissions under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Accounts** can create, update, or delete the Accounts information.

To create or edit an Account

1. From the Integration Cloud navigation bar, click **Applications**.
2. Select an Application from the list, 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** screen to add, edit, or delete Accounts. The options available may vary according to the selected Application.

To add or edit an Account

1. From the Integration Cloud navigation bar, click **Applications**.
2. Select an Application from the list, 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 change any field in an existing Account.
4. On the **New Account** or **Edit Account** screen, complete the following fields. Required fields are marked with an asterisk on the screen.

Note: Based on the Application you had selected, applicable fields are displayed.

Field	Description
Save As	<p>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. The following characters are also not allowed:</p> <ul style="list-style-type: none"> \\ (double backward slashes) / (forward slash) : (colon) * (asterisk) ? (question mark) " (double quote) < (Less Than symbol) > (Greater Than symbol) (vertical bar)
Description	<p>Provide a description for the Account. This field is common for all Applications.</p>

The Account configuration section allows you to provide details to connect with the Application. The fields available may vary according to the selected Application. If you have added any stage in the **Stage Management** screen, the stages will appear as tabs in the Account configuration section. Enter the Account configuration details for each stage. If you have configured the Account details incorrectly in any stage, the stage will appear in red text and the Account will be inactive. If an Account is configured correctly in a particular stage, then the stage appears in green text and is active. Only active or enabled Accounts are listed in the drop down list of the Operation wizard, Integration wizard, Look up Transformer, and Stage Management.

Field	Description
-------	-------------

See "[Stage Management](#)" on page 47 for more information.

You must have the permission to administer stages (**Access Profiles > Administrative Permissions > Functional Controls > Stages**) if you want to create or delete stages.

See the "[Account Configuration Details](#)" on page 77 section for information on the Account configuration fields for each Application.

5. Click **Save** or **Update** to save your settings or click **Save All Stages** to save the changes done in all the stages.

A new Account will be created.

12 Account Configuration Details

■ Alfabet	79
■ Apache Solr Search	80
■ Amazon Simple Notification Service (SNS)	81
■ Amazon Simple Queue Service (SQS)	82
■ Amazon Simple Storage Service (S3)	83
■ Avalara AvaTax	84
■ Concur	85
■ Coupa	87
■ Cumulocity	88
■ CloudStreams Connector for NetSuite™	89
■ File Transfer Protocol (FTP/FTPS)	90
■ Google Apps Admin	91
■ Google BigQuery	92
■ Google Calendar	93
■ Google Contacts	95
■ Google Drive	96
■ Google Cloud Pub/Sub	97
■ Google Cloud Storage	98
■ Google Sheets	99
■ Magento eCommerce Platform	100
■ Marketo	101
■ Microsoft Dynamics CRM	102
■ OData v2.0	103
■ OData v4.0	105
■ On-Premises Applications	106
■ REST Application Account Configuration Details	106

■ Salesforce	108
■ SAP Cloud for Customer(C4C) OData v2.0	111
■ SAP S/4HANA Marketing Cloud	112
■ Secure File Transfer Protocol (SFTP)	114
■ ServiceNow Enterprise Service Management	117
■ Slack	118
■ SOAP Application Account Configuration Details	119
■ Strikelron Contact Verification	121
■ SuccessFactors HCM	122
■ SugarCRM	124
■ Zendesk	125

This section provides information on the Account configuration fields.

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	<p>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 <code>"/Alfabet/api/vXX"</code> in the URL. For example, both these options are equivalent:</p> <ul style="list-style-type: none"> ■ <code>https://myalfabet.com</code> ■ <code>https://myalfabet.com/Alfabet/api/v1</code>
Connection TimeOut	<p>The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. Specify a value other than 0 to avoid using a socket with no timeout.</p>
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
Retry on Response Failure	<p>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.</p>
Authorization Token	<p>The Alfabet Authorization Token as defined in the <code>web.config</code> file of the Alfabet Web Application on the server side, under the <code><alfaSection></code> element.</p>

Note: See the *Authorization* chapter in the Alfabet Interface for RESTful Web Services reference manual for required

Field	Description
	configurations in the server side for Alfabet and for details about the different authorization modes.

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	<p>Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: <code>https://<hostName></code>.</p> <p>Replace <code><hostName></code> with your actual back end system server URL hosting Apache Solr as the search engine.</p>
Connection TimeOut	<p>The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. Specify a value other than 0 to avoid using a socket with no timeout.</p>
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
Retry on Response Failure	<p>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.</p>
Username	<p>Username received from the back end system hosting Apache Solr as the search engine.</p>

Field	Description
Password	This is the password received from the back end system hosting Apache Solr as the search engine.
Authorization Type	<p>Apache Solr REST APIs use Basic Authentication. The Username and Password is passed when you invoke any of the REST API endpoints.</p> <p>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.</p>

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 <code>https://</code> , for example, <code>https://sns.(Region).amazonaws.com</code> . This is the native provider endpoint target for the Account configuration.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>

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.
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.
Signing Algorithm	Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.

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/ .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response

Field	Description
	back from the backend, it will retry only if the Retry on Response Failure option is selected.
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	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.
	Note: This field is not applicable for Amazon SQS Version 4.

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/ .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.

Field	Description
	Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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	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.

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.

Field	Description
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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.

Concur

Integration Cloud connects to Concur using the Concur API v3 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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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, 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. 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.

Coupa

Integration Cloud connects to Coupa using the Coupa API v17 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	<p>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.</p> <p>Replace <instance> with your actual Coupa instance.</p>
Connection TimeOut	<p>The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. Specify a value other than 0 to avoid using a socket with no timeout.</p>
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
Retry on Response Failure	<p>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.</p>
X-COUPA-API-KEY	<p>The API Key received from the user account.</p> <p>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.</p>

Cumulocity

Integration Cloud connects to Cumulocity using the API version 0.9 and allows you to manage assets and Internet of Things (IoT) devices.

Field	Description
Server URL	<p>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>.</p> <p>Replace <instance> with your actual Cumulocity instance.</p>
Connection TimeOut	<p>The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the connection waits indefinitely. Specify a value other than 0 to avoid using a socket with no timeout.</p>
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
Retry on Response Failure	<p>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.</p>
Username	<p>This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider.</p>
Password	<p>Provide a password for the user name provided in the Username field to initiate communication with the SaaS provider.</p>
Authorization Type	<p>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</p>

Field	Description
	Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.

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`.

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: <code>https://webservices.na1.netsuite.com/services/NetSuitePort_2013_2</code> , where <code>na1</code> is the instance name.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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.

Field	Description
Password	The password of the user email account.
Authorization Type	The type of HTTP authorization scheme to use for the connection. The CloudStreams Connector for NetSuite™ supports HTTP Basic authentication.
Account	Specify the account number issued to you by NetSuite™.
Role	Specify the role with which you want to execute the web services, for example, Administrator.

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: 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. 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.

Field	Description
	<p>Note: Users who have the Administer permission under Settings > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security can add, edit, and delete Keystores.</p>
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.
	<p>Note: Users who have the Administer permission under Settings > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security can add, edit, and delete Truststores.</p>

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: <code>https://www.googleapis.com/admin</code> .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>

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.
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 .

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.

Field	Description
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
Retry on Response Failure	<p>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.</p>
Consumer ID	<p>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.</p>
Consumer Secret	<p>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.</p>
Access Token	<p>Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.</p>
Refresh Token	<p>A token used by the client to obtain a new access token without involving the resource owner.</p>
Refresh URL	<p>This is the provider specific URL to refresh an Access Token.</p> <p>Example: https://www.googleapis.com/oauth2/v4/token.</p>

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the back end. If an I/O error occurs when Integration Cloud is reading a response back from the back end, it will retry only if the Retry on Response Failure option is selected.</p>
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, 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

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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, 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 .

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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 .

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 the SaaS provider. Example: https://pubsub.googleapis.com .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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 .

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.

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.
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 .

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.

Field	Description
	Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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 .

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: <a href="http://<yourhost>/api/rest">http://<yourhost>/api/rest .

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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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.

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>.mktoreset.com. Replace <instance> with your actual Marketo instance.

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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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, 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.

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/XRMServices/2011/Organization.svc, where <organization> must be replaced with your actual Microsoft Dynamics CRM organization.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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.
User Name	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.

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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 .
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

Field	Description
	will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance.

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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.
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 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.

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.
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.

On-Premises Applications

On-Premises applications loaded from on-premises systems are listed in the **Applications** page, but you will not be able to create Accounts or Operations for on-premises applications. Those can be uploaded only from the webMethods Integration Server. Further, when you upload services as part of an application from the on-premises 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 stage appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the stage appears in green color and if an Account is not configured in a particular stage, that stage appears in white color. For on-premises Applications, the Account can be used to execute services on the on-premises webMethods Integration Server.

REST Application Account Configuration Details

Integration Cloud allows you to create 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.

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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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, 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.
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.
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 Token	A token used by the client to obtain a new access token without having to involve the resource owner.
Refresh URL	The provider specific URL to refresh an Access Token.

Field	Description
Refresh URL Request	<p>Options for sending the parameters in the Access Token refresh request. The options are Body Query String and URL Query String. Default is Body Query String.</p> <p>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.</p> <p>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.</p>
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.
Authorization Type	Select the type of HTTP authorization scheme to use for the Account. 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.

Salesforce

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.

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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.com from outside your company's trusted networks, you must add a security token (provided by Salesforce) to your password. For more information about logging on Salesforce.com, see the Salesforce.com 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.

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.

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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.com from outside your company's trusted networks, you must add a security token (provided by Salesforce) to your password. For more information about logging on Salesforce.com, see the Salesforce.com 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

Field	Description
	Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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.

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 , 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.
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.

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.
Connection TimeOut	The number of milliseconds a connection waits before canceling its attempt to connect to the resource. If you specify 0, the

Field	Description
	connection waits indefinitely. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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 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.

Field	Description
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.

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.

<u>For this parameter...</u>	<u>Specify...</u>						
Port Number	The port number of the SFTP server. The port number must be within the range of 1 and 65535 (inclusive).						
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 SFTP server's host public key fingerprint. 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. This field is visible only after you have established a connection to an SFTP 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 that Integration Cloud uses to authenticate itself to the SFTP server. Client authentication can be either by password or by public and private keys.						
	<table border="1"> <thead> <tr> <th><u>Select...</u></th> <th><u>To...</u></th> </tr> </thead> <tbody> <tr> <td>Password</td> <td>Use password authentication. If you are using password authentication, enter the password for the specified user to connect to the SFTP server.</td> </tr> <tr> <td>Public Key</td> <td>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.</td> </tr> </tbody> </table>	<u>Select...</u>	<u>To...</u>	Password	Use password authentication. If you are using password authentication, enter the password for the specified user to connect to the SFTP server.	Public Key	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.
<u>Select...</u>	<u>To...</u>						
Password	Use password authentication. If you are using password authentication, enter the password for the specified user to connect to the SFTP server.						
Public Key	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.						

<u>For this parameter...</u>	<u>Specify...</u>						
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.						
Connection TimeOut (seconds)	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.						
Session Timeout (minutes)	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.						
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.						
	<table border="1"> <thead> <tr> <th><u>Select...</u></th> <th><u>To...</u></th> </tr> </thead> <tbody> <tr> <td>None</td> <td>Not compress the data.</td> </tr> <tr> <td>zlib</td> <td>Compress the data that is transmitted between the SFTP server and Integration Cloud.</td> </tr> </tbody> </table>	<u>Select...</u>	<u>To...</u>	None	Not compress the data.	zlib	Compress the data that is transmitted between the SFTP server and Integration Cloud.
<u>Select...</u>	<u>To...</u>						
None	Not compress the data.						
zlib	Compress the data that is transmitted between the SFTP server and Integration Cloud.						

For this parameter...	Specify...
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).

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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
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.

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. The URL for Slack REST Application depends on the team name: https://YOURTEAM.slack.com Example: https://exampleteam.slack.com
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.

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.
Access Token	The token used for authentication and issued by the Authorization Server.

SOAP Application 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
URL	This is the URL for the web service. You can edit the URL to specify a different web service endpoint.
Port Binding	Select the bind address from the drop-down list, that is, the concrete protocol and data format specification for the web service.
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.
	<p>Note: Users who have the Administer permission under Settings > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security can add, edit, and delete Keystores.</p>
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.

Field	Description
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.	
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.
	Note: 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.
	Note: 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.
	Note: 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

Field	Description
	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 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.

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 .
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.

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. 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.

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: <code>https://api.successfactors.com/sfapi/v1/soap</code> <code>https://<instance_name>.successfactors.com</code> Where <instance_name> represents the actual instance name.

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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	<p>The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails.</p> <p>Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.</p>
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	<p>The type of HTTP authorization scheme to use for the Account. The SuccessFactors Application does not use Authorization headers.</p> <p>If you specify none, no additional authorization scheme will be executed at run time.</p> <p>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.</p>
Company ID	The company ID that SuccessFactors provided, when your company registered with them.

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.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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.

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.

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: <code>https://<domain>.zendesk.com</code> . Replace <code><domain></code> with your actual Zendesk instance.
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. Specify a value other than 0 to avoid using a socket with no timeout.
Connection Retry Count	The number of times Integration Cloud should attempt to initialize the connection at startup if the initial attempt fails. Integration Cloud retries to establish a connection when an I/O error occurs while sending the request message to the backend. If an I/O error occurs when Integration Cloud is reading a response back from the backend, it will retry only if the Retry on Response Failure option is selected.
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, 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.

Field	Description
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.

13 Operations

- Adding or Editing Operations 129

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 backend and perform Operations.

Note: Users who have the required permissions under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Operations** 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 screen lists all the available Operations for a selected application including predefined Operations.

See "[FTP Predefined Operations](#)" on page 131 for information on the predefined FTP operations.


See "[SFTP Predefined Operations](#)" on page 134 for information on the predefined SFTP operations.

To create or edit a custom Operation

1. From the Integration Cloud navigation bar, click **Applications**.
2. Select an application from the list, 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** screen, click **Add** to create a new Operation. Select an Operation and click **Edit** to update an existing Operation, 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.

Click the **Show Signature** option 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. Both Trigger and Action Operations can be tested. 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** screen to add, edit, or delete custom Operations.

To add or edit a custom Operation

1. From the Integration Cloud navigation bar, click **Applications**.
2. Select an Application from the list, 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 screen for the selected Application, click **Add New Operation** to add a custom Operation or click **Edit** to update an existing custom Operation. You can edit the **Operation** or the **Business Parameter** from the **Edit** drop-down list.
4. On the **Select your <...> account** screen, complete the following fields. Required fields are marked with an asterisk on the screen.

Field	Description
Name	Provide a name for the custom Operation.
Description	Provide a description for the custom Operation.
Type	<p>Select the type of Operation you want to perform for the selected Application.</p> <p>A Trigger is an Operation which fetches data from an Application and which can be used in the Source section while creating an Integration.</p> <p>An Action is an Operation which uploads data to an Application and which can be used in the Target section while creating an Integration.</p> <p>Note: You will not be allowed to change these options after you have created an Operation.</p>
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.

Field	Description
Select functional area	Select the Application service from the drop-down list. Note: This option is available only for certain Applications.

5. Click **Next**.

The **Select the Operation** screen appears.

6. Select the Operation to be performed, and then click **Next**.

The **Select the Business Object** screen appears.




7. Select the Business Object to be associated with the Operation you have selected in the previous step and then click **Next**. Business Objects appear only for certain Applications and Operations.



The **Select the Data Fields** screen appears.

8. 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 that appear in the **Added Data Fields** pane are selected by default and cannot be cleared.


Simple fields appear with a check box while complex fields appear with an arrow mark and then followed by a check box. The following states are observed for complex fields and for the table header check box:

- Unchecked - None of the fields are selected
- Checked - When all the fields are selected
- Partially checked - When at least one field is not selected

When the state is "Partially checked", then based on the browser you are using, the check box of the complex field and the table header check box appear hyphenated in Google Chrome () , appear as a black box in Internet Explorer () , and appear shaded () in Mozilla Firefox.

Click  or  to move only the selected fields.

Click  or  to move all simple and loaded complex fields.

You can close multiple complex fields by clicking the **Collapse All** button on the top left corner of the **Select the Data Fields** pane. Point the mouse cursor to a field to see more details about that field. The informative symbol () appears beside a complex field that is not loaded.

Note: For some Applications, for example, Microsoft Dynamics CRM, you can choose the way a query can be executed in the **Confirm operation** screen. Choose the operation and then click **Finish**.

9. Verify the details in the **Confirm Operation** screen. You can click the link beside the **Data Fields Added** field to view the data fields you have added.
10. Click **Finish** to create the custom Operation.

Note: You will not be able to delete an Operation if it is used by an Integration.

After you click **Finish** or **Save**, if there are any Business Parameters, you will be asked to configure the Business Parameters.

FTP Predefined Operations

The following predefined FTP operations are available:

Operation	Description
getFile	Retrieves a file from a remote FTP server.
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.
deleteFiles	Delete file(s) from a remote FTP server.
putFile	Transfers a file to a remote FTP server.
renameFile	Renames a file on a remote FTP server.

getFile

Retrieves a file from a remote FTP server.

Input Parameters

<i>remoteFile</i>	String Name of the remote file.
<i>transferType</i>	String FTP file transfer mode (ASCII or binary). The default is ASCII.

Output Parameters

<i>contentStream</i>	Object A java.io.InputStream object.
<i>statusCode</i>	String Standard FTP protocol status code.
<i>statusMessage</i>	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

<i>remotePath</i>	String Optional. Absolute or relative path of the remote directory. If <i>remotePath</i> 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.
<i>listFilter</i>	String Optional. Filter that specifies the names of the files to include in the list (for example, *.txt).

Output Parameters

<i>fileList</i> []	String List List of file names matching <i>listFilter</i> .
<i>statusCode</i>	String Standard FTP protocol status code.
<i>statusMessage</i>	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.

SFTP Predefined Operations

The following predefined SFTP operations are available:

Operation	Description
<code>cd</code>	Changes the working directory on the remote SFTP server.
<code>chgrp</code>	Changes the group ownership of one or more remote files.
<code>chmod</code>	Changes permissions of one or more remote files.
<code>chown</code>	Changes the user of one or more remote files.
<code>get</code>	Retrieves a file from a remote SFTP server.
<code>ls</code>	Retrieves the remote directory listing of the specified path or current remote directory if path is not specified.
<code>mkdir</code>	Creates a new remote directory.
<code>put</code>	Transfers a file to a remote SFTP server.
<code>pwd</code>	Displays the remote working directory on the SFTP server.
<code>rename</code>	Renames a file or directory on a remote SFTP server.
<code>rm</code>	Deletes one or more remote files on the SFTP server.
<code>rmdir</code>	Deletes one or more remote directories on the SFTP server.
<code>symlink</code>	Creates a symbolic link between the old path and the new path of a file.

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

<i>returnCode</i>	String Standard SFTP protocol return code.
<i>returnMsg</i>	String Text message describing the return code.

chown

Changes the owning user of one or more remote files.

Input Parameters

<i>uid</i>	String Numeric user ID of the new owning user of the file.
<i>path</i>	String Absolute or relative path of the remote files.

Output Parameters

<i>returnCode</i>	String Standard SFTP protocol return code.
<i>returnMsg</i>	String Text message describing the return code.

get

Retrieves a file from a remote SFTP server.

Input Parameters

<i>remoteFile</i>	String Absolute or relative path of the remote file.
-------------------	---

Output Parameters

<i>returnCode</i>	String Standard SFTP protocol return code.
<i>returnMsg</i>	String Text message describing the return code.
<i>contentStream</i>	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.

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:

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.

14 Develop

■ Point-to-Point Integrations	144
■ Orchestrated Integrations	146
■ Pipeline and Signatures	159
■ Built-In Services	162
■ Integration Details	283
■ Recipes	286
■ Document Types	287
■ Reference Data	290

An Integration is an orchestration of a source and a target Operation with appropriate data mappings and transformations. 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-premise applications.

Note: Users who have the required permissions under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Integrations** can create, update, delete, execute, or deploy Integrations.

The **Name** column in the **Integrations** page displays the name of the Integration. If you select an Integration and click the Integration name link under the **Name** column, the Integration details **Overview** page appears for that Integration. To view the last five execution results for an Integration, click **Last 5 Execution Results** available in the Integration details page. The **Type** column shows whether the Integration is an Orchestration or a Point to Point. The **Uses** column displays the Integrations, Accounts, Operations, Applications, Reference Data, Document Types, and so on that are used or utilized to create the Integration. Click the  icon to view the components used by the Integration. The **Created On** column displays when the Integration was created and the **Created By** column displays who created it.

To edit an Integration, select the Integration, and then click **Edit**. The Integration opens up for editing in the **Design** panel. To delete an Integration, select the Integration, and then click **Delete**. The Integration will be permanently deleted and cannot be recovered. To copy an Integration, select the Integration, and then click **Copy** to save the Integration with a different valid name. This way you can have different names for the same Integration at different stages. To export an Integration, select the Integration, and then click **Export**. To import Integrations, select the Integration, and then click **Import Integrations**. See "Importing Integrations" and "Exporting Integrations" for more information. To create a new Integration, click **Add New Integration**, and then select **Synchronize two applications** to create a Point to Point Integration. To create an Orchestrated Integration, select **Orchestrate two or more applications**.

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-premise 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 **Develop**. The **Integrations** screen appears.
2. To edit an existing Integration, select an Integration from the **Integrations** screen and click **Edit**.
3. To create a new point-to-point Integration, from the **Integrations** screen, click **Add New Integration**, select **Synchronize two applications**, and click **OK**.

Note: See "[Orchestrated Integrations](#)" on page 146 for information on how to create an orchestrated Integration.

Note: 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.

4. Provide a name and description of your Integration. Required fields are marked with an asterisk on the screen.
5. 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.
6. 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.

7. 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.
8. 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. Red colored arrows indicate incorrect mappings. 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 to the fields, and transformers.

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 *fx* icon to add a transformer in the **Transform Data** screen. This screen 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. You can click the blue colored *fx* icon and select **Edit Transformer** or **Delete Transformer** to either modify the transformer or delete it.

9. Click **Next** to review your Integration.
10. 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 **Develop**. The **Integrations** screen appears.
2. To create a new Integration, from the **Integrations** screen, click **Add New Integration**.
3. Select **Orchestrate two or more applications**, and then click **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 attach them within the *Start Integration and End Integration space*.





The tool bar has a large number of blocks for common instructions and the blocks are divided into the following categories:

- Applications
- Services
- Integrations
- Control Flow
- Expressions

<u>Block category</u>	<u>Icons</u>	<u>Description</u>
Applications		Displays the Applications available in Integration Cloud.
Services		Use the <i>Service</i> blocks (date, math, string, and so on) to specify the service that will be invoked at run time. Related services are grouped in blocks. You can sequence services and manage the flow of data among them.


Note: For information on the different services, see [Built-In Services](#).

The **Reference Data** block appears only if a Reference Data service is available at **Develop**

Block category	Icons	Description
Integrations		<p data-bbox="743 306 1268 369">> Reference Data > Reference Data page. See Reference Data for more information.</p> <p data-bbox="662 417 1328 621">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.</p> <p data-bbox="662 657 1289 804">Click the  icon if you want to view or modify an Integration after it is moved within the Start Integration and End Integration space. The Integration will open up for editing in a new tab.</p> <p data-bbox="662 835 1328 1150">Click the  icon and select Map Input and Output if you want to map the input of the operation from the Pipeline and also map the output of the operation into the pipeline. Click Duplicate to repeat a block, click Collapse to flatten a block, click Delete to remove a block from the workspace, or click Disable to disable a block and all blocks within that block. If you disable blocks, those blocks will not be considered for execution, test, or debug operations.</p>
Control Flow		<p data-bbox="662 1194 1289 1262">Conditional expressions, looping structures, and transform pipeline.</p> <p data-bbox="662 1283 1328 1591">Conditional expressions perform different computations or actions depending on whether a specified boolean condition evaluates to true or false. The <i>if-then</i> block is used to evaluate a boolean condition and if the condition is true, the statements following the “then” are executed. Otherwise, the execution continues in “else” if you have selected the <i>if-then-else</i> block. You can implement error handling by using the <i>try-catch</i> block.</p> <p data-bbox="662 1612 1328 1917">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 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</p>



Block category	Icons	Description
		<p>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.</p> <p><i>while-do</i> loops repeat their bodies while the conditional expression you have provided evaluates to true. <i>do-until</i> loops are similar except that they repeat their bodies until some condition is true. The <i>for-each-do</i> 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".</p> <p>The Exit Integration block allows you to successfully terminate and exit from the currently running Integration. You cannot attach child blocks to the Exit Integration block.</p> <p>The Exit Integration with 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 property, 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 with failure "..." block.</p> <p>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.</p> <p>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</p>


Block category	Icons	Description
		<p>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.</p> <p>Use the <i>Transform Pipeline</i> block to make pipeline modifications. See Pipeline and Signatures for more information.</p>

Expressions		<p>Logical operations, comparisons, and values.</p> <p>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.</p> <p>The <i>and</i> block will return true only if both of its two inputs are also true. The <i>or</i> block will return true if either of its two inputs are true. The <i>not</i> block converts its Boolean input into its opposite.</p> <p>You can also type a text value, select a field on which to build an expression, or select a block with no inputs.</p>
--------------------	---	---


Note: It is recommended not to leave an input empty.



- Provide a valid name and description for the Integration in the Start Integration and End Integration block, which is the root block for your Integration.

Note: After saving an Integration, you can click **Test** to test the Integration execution in real time and view the execution results in 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  to either open the entry in JSON format or save it locally. Click  to remove the selected entry.





- Click **Applications**. The list of supported Applications appears.
- Drag and drop an Application to the Start/End Integration block.
- To select the Account and Operation for the Application, click 


The following table depicts the block interactions:


Icons	Applicable for...	Action/Description
	All blocks except the Expressions block	<p>Add or edit the description of a block.</p> <p>Comments for all blocks except the Expressions block.</p>
	Applications, Services, Integrations	<p>Start Integration/End Integration Block > Define Input and Output Signature</p> <p>Click 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.</p> <p>You can use a Document Type 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 side, you cannot add, modify, or delete the fields on that part of the tab.</p> <p>You can select a Document Type from the Document Reference drop-</p>

Icons	Applicable for...	Action/Description
		<p>down list. To create a Document Type, from the Integration Cloud navigation bar, select Develop > Document Types > Add New Document Type.</p> <p>You can click Load XML and then paste the XML content to generate a Document Type from the XML structure or click Load JSON and then paste the JSON content to generate a Document Type from the JSON structure.</p> <p>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.</p> <p>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.</p> <p>See Document Types for more information.</p> <div data-bbox="868 1465 1367 1596" style="background-color: #f0f0f0; padding: 5px;"> <p>Note You cannot modify or paste the child fields of a Document Reference.</p> </div> <p>Select Business Data to Log</p> <p>Integration Cloud allows you to log select business data from the Operation and Integration signatures either always, or only when errors occur. Values of logged fields can be viewed in the Only Business Data section in the</p>

Icons	Applicable for...	Action/Description
		<p>Execution Results screen. You can also create aliases for the logged fields.</p> <p>To select input or output fields for logging, click Select Business Data to Log, and in the Select Business Data to Log dialog box, choose whether you want to log business data only when errors occur (Error Only) or choose (Always) to always log business data. The default setting is Error Only. 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. You can select up to three Input fields and three Output fields for logging. 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.</p> <p>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.</p> <p>Map Input and Output</p> <p>Map the input of the operation from the Pipeline and also map the output of the operation into the pipeline.</p> <p>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</p>

Icons	Applicable for...	Action/Description
		<p>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.</p> <p>You can select a block, other than the root block, and click Duplicate to repeat a block, click Collapse to flatten a block, click Delete to remove a block from the workspace, or click Disable to disable a block and all blocks within that block. If you disable blocks, those blocks will not be considered for execution, test, or debug operations.</p>
	Control Flow > Transform Pipeline	<p>Make pipeline modifications.</p> <p>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.</p>
	Applications	<p>Select an Account and an Operation for the Application.</p>
	Orchestrated Integrations	<p>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.</p>
	Applications/Services	<p>The block is not configured. Select an Account and an Operation for the Application or select a Service.</p>

Icons	Applicable for...	Action/Description
	Orchestrated Integrations	An Orchestrated Integration has been modified or newly created but not saved.

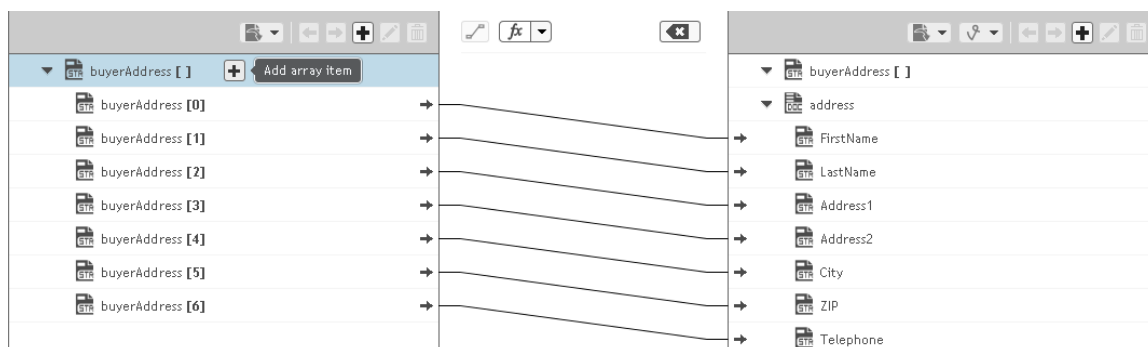
- Create the Integration using the available constructs by inserting the blocks, setting properties, declaring the input and output parameters, setting values, performing pipeline variable substitutions (if you want to replace the value of a pipeline field at run time), and mapping the pipeline data. To map the Pipeline Input to the Input Signature, click the  icon and then select **Map Input and Output**.
- Map the Output Signature to the Pipeline Output in the **Pipeline Data** window, and then click **Finish**.

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.



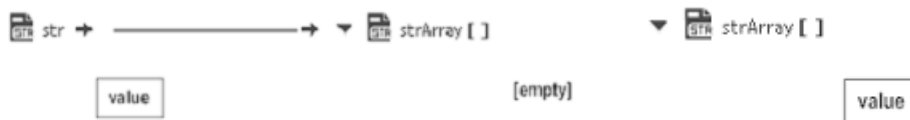
Suppose a String List has length 3 and if you link index 4 of the String List, at run time, the String List length is increased from 3 to 5.

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.

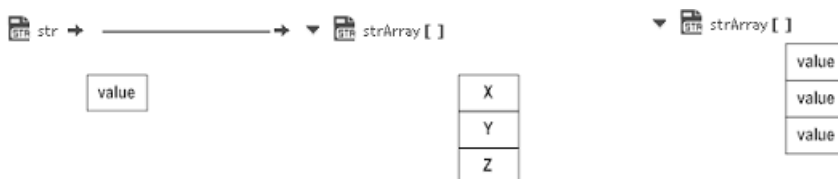
Default Pipeline Rules for Linking to and from Array Variables

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.

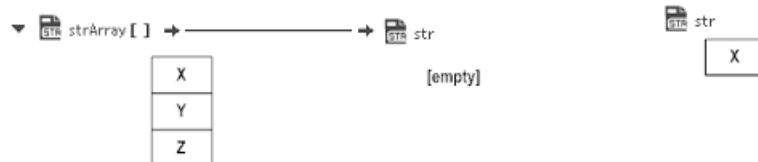
If you link...	To...	Then...
A scalar variable	An array variable that is empty (the variable does not have a defined length)	The link defines the length of the array variable; that is, it contains one element and has length of one. The first (and only) element in the array is assigned the value of the 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.



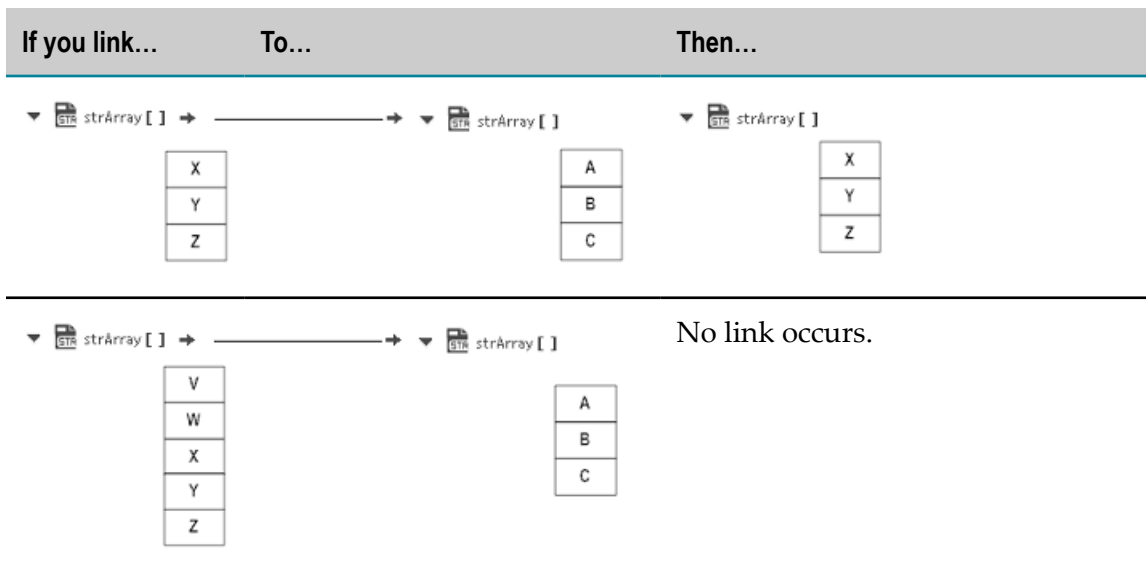
If you link...	To...	Then...
An array variable	A scalar variable	The scalar variable is assigned the first element in the array.



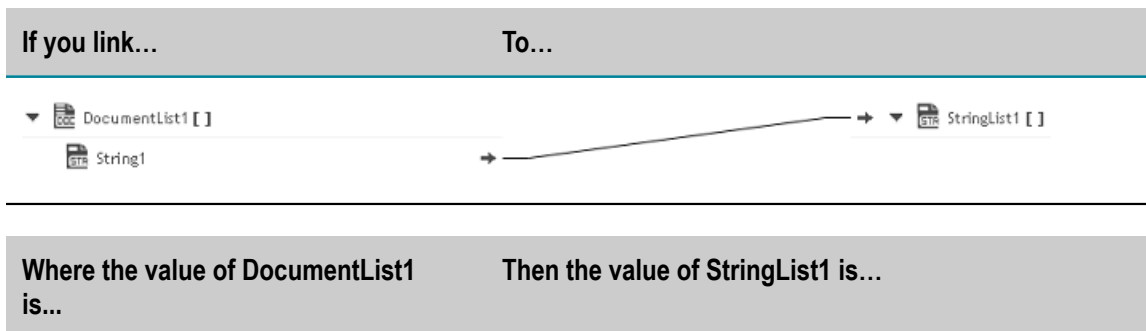
If you link...	To...	Then...
An array variable	An array variable that does not have a defined length	The link defines the length of the target array variable; that is, it will be the same length as the source array variable. The elements in the target array variable are assigned the values of the corresponding 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 <i>must</i> 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.



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:



- 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.

11. Click **Save** to save your Integration or click **Save All** to save all modified Integrations.

The new Integration appears in the **Integrations** page. You can click on the Integration link in the **Integrations** page to view the Integration details.

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

<u>Name</u>	<u>Data Type</u>	<u>Name</u>	<u>Data Type</u>
<i>AcctNum</i>	String	<i>AuthCode</i>	String
<i>OrderTotal</i>	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

In the Start Integration/End Integration block of an Orchestrated Integration, click the  icon > **Define Input/Output** 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.

Specify Input/Output Signature

Input
Output

Document Type Reference ArraysExample

Search for Field...

- stringArray [] *
- docArray [] *
 - string *
 - stringArray [] *
 - doc *
 - string *
 - stringArray [] *

Field Properties

Name *

Type *

XML Namespace

Required

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 “Add new field” 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.

Built-In Services

This section describes the services provided with Integration Cloud. 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. These parameters are also collectively referred to as a signature.

Date

Use the **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 `MMMM dd, yyyy`. To describe the format `01/15/99`, you would use the pattern string `MM/dd/yy`. For more information about these pattern string symbols, see the Oracle Java API documentation for the `SimpleDateFormat` class.

Symbol	Meaning	Presentation	Example
G	era designator	Text	AD
y	year	Number	1996 or 96

Symbol	Meaning	Presentation	Example
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
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
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

ID	Raw Offset	Name
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).

Summary of Date services

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.

Service	Description
<code>compareDates</code>	Compares two dates and returns the result as integer.
<code>dateBuild</code>	Builds a date String using the specified pattern and the specified date services.
<code>dateTimeBuild</code>	Builds a date/time string using the specified pattern and the specified date services.
<code>dateTimeFormat</code>	Converts date/time (represented as a String) string from one format to another.
<code>getCurrentDateString</code>	Returns the current date as a String in a specified format.
<code>incrementDate</code>	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

<code>startDate</code>	String Starting date and time.
<code>endDate</code>	String Ending date and time.
<code>startDatePattern</code>	String Format in which the <code>startDate</code> parameter is to be specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the “Pattern String Symbols” section.
<code>endDatePattern</code>	String Format in which the <code>endDate</code> parameter is to be specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the “Pattern String Symbols” section.

Output Parameters

<i>dateDifferenceSeconds</i>	String The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of seconds.
<i>dateDifferenceMinutes</i>	String The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of minutes.
<i>dateDifferenceHours</i>	String The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of hours.
<i>dateDifferenceDays</i>	String The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , 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

<i>startDate</i>	String Starting date to compare against <i>endDate</i> .
<i>endDate</i>	String Ending date to compare against <i>startDate</i> .
<i>startDatePattern</i>	String Format in which the <i>startDate</i> parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.
<i>endDatePattern</i>	String Format in which the <i>endDate</i> 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*.

<u>A value of...</u>	<u>Indicates that...</u>
+1	The <i>startDate</i> is after the <i>endDate</i> .
0	The <i>startDate</i> is the same as the <i>endDate</i> .
-1	The <i>startDate</i> is before the <i>endDate</i> .

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.

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.

<i>month</i>	String Optional. The month expressed as a number (for example, 1 for January, 2 for February). If you do not specify <i>month</i> or you specify an invalid value, <code>dateBuild</code> uses the current month.
<i>dayofmonth</i>	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 <i>dayofmonth</i> or you specify an invalid value, <code>dateBuild</code> uses the current day.
<i>timezone</i>	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, <code>EST</code> for Eastern Standard Time. If you do not specify <i>timezone</i> , the value of the server's "user timezone" property is used. If this property has not been set, <code>GMT</code> is used.
<i>locale</i>	String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM YYYY</code> will produce <code>Friday 23 August 2002</code> , and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code> .

Output Parameters

<i>value</i>	String The date specified by <i>year</i> , <i>month</i> , and <i>dayofmonth</i> , in the format of <i>pattern</i> .
--------------	--

dateTimeBuild

Builds a date/time string using the specified pattern and the specified date services.

Input Parameters

<i>pattern</i>	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 <i>pattern</i> , <code>dateTimeBuild</code> returns null. If <i>pattern</i> contains a time zone and the <i>timezone</i> parameter is not set, the default time zone is used.
<i>year</i>	String Optional. The year expressed in <code>yyyy</code> or <code>yy</code> format (for example, <code>01</code> or <code>2001</code>). If you do not specify <i>year</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current year.

<i>month</i>	String Optional. The month expressed as a number (for example, 1 for January, 2 for February). If you do not specify <i>month</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current month.
<i>dayofmonth</i>	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 <i>dayofmonth</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current day.
<i>hour</i>	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 <i>hour</i> or you specify an invalid value, <code>dateTimeBuild</code> uses 0 as the <i>hour</i> value.
<i>minute</i>	String Optional. Minutes expressed as a number. If you do not specify <i>minute</i> or you specify an invalid value, <code>dateTimeBuild</code> uses 0 as the <i>minute</i> value.
<i>second</i>	String Optional. Seconds expressed as a number. If you do not specify <i>second</i> or you specify an invalid value, <code>dateTimeBuild</code> uses 0 as the <i>second</i> value.
<i>millis</i>	String Optional. Milliseconds expressed as a number. If you do not specify <i>millis</i> or you specify an invalid value, <code>dateTimeBuild</code> uses 0 as the <i>millis</i> value.
<i>timezone</i>	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, <code>EST</code> for Eastern Standard Time. If you do not specify <i>timezone</i> , the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.
<i>locale</i>	String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM yyyy</code> will produce <code>Friday 23 August 2002</code> , and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code> .

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

<i>inString</i>	String Date/time that you want to convert. Important: If <i>inString</i> 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.
<i>currentPattern</i>	String Pattern string that describes the format of <i>inString</i> . For pattern-string notation, see the “Pattern String Symbols” section.
<i>newPattern</i>	String Pattern string that describes the format in which you want <i>inString</i> returned. For pattern-string syntax, see the “Pattern String Symbols” section.
<i>locale</i>	String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM yyyy</code> will produce <code>Friday 23 August 2002</code> , and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code> .
<i>lenient</i>	String Optional. A flag indicating whether an exception will appear if the <i>inString</i> value does not adhere to the format specified in <i>currentPattern</i> parameter. Set to: <ul style="list-style-type: none">■ <code>true</code> to perform a lenient check. This is the default. In a lenient check, if the format of the date specified in the <i>inString</i> parameter does not match the format specified in the <i>currentPattern</i> parameter, the date in the format specified in the <i>currentPattern</i> parameter will be interpreted and returned. If the interpretation is incorrect, the service will return an invalid date.■ <code>false</code> to perform a strict check. In a strict check, an exception will appear if the format of the date specified in the <i>inString</i> parameter does not match the format specified in the <i>currentPattern</i> 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.

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

<i>startDate</i>	String Starting date and time.
<i>startDatePattern</i>	String Format in which the <i>startDate</i> parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.
<i>endDatePattern</i>	String Optional. Pattern representing the format in which you want the <i>endDate</i> to be returned. For pattern-string notation, see the "Pattern String Symbols" section. If no <i>endDatePattern</i> is specified, the <i>endDate</i> will be returned in the format specified in the <i>startDatePattern</i> parameter.
<i>addYears</i>	String Optional. Number of years to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMonths</i>	String Optional. Number of months to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addDays</i>	String Optional. Number of days to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addHours</i>	String Optional. Number of hours to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMinutes</i>	String Optional. Number of minutes to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.

<i>addSeconds</i>	String Optional. Number of seconds to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMilliseconds</i>	String Optional. Number of milliseconds to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>timezone</i>	String Optional. Time zone in which you want the <i>endDate</i> to be expressed. Specify a time zone code, for example, EST for Eastern Standard Time. If you do not specify <i>timezone</i> , the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.
<i>locale</i>	String Optional. Locale in which the <i>endDate</i> is to be expressed. For example, if <i>locale</i> is <i>en</i> (for English), the pattern <i>EEE d MMM yyyy</i> will produce <i>Friday 23 August 2002</i> , and the <i>locale</i> of <i>fr</i> (for French) will produce <i>vendredi 23 août 2002</i> .

Output Parameters

<i>endDate</i>	String The end date and time, calculated by incrementing the <i>startDate</i> with the specified years, months, days, hours, minutes, seconds, and/or milliseconds. The <i>endDate</i> will be in the <i>endDatePattern</i> format, if specified. If no <i>endDatePattern</i> is specified or if blank spaces are specified as the value, the <i>endDate</i> will be returned in the format specified in the <i>startDatePattern</i> 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

Summary of Document services

The following **Document** services are available:

Service	Description
<code>findDocuments</code>	Searches a set of documents for entries matching a set of Criteria.
<code>insertDocument</code>	Inserts a new document in a set of documents at a specified position.
<code>deleteDocuments</code>	Deletes the specified documents from a set of documents.
<code>documentListToDocument</code>	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.
<code>documentToDocumentList</code>	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.
<code>groupDocuments</code>	Groups a set of documents based on specified criteria.
<code>documentToBytes</code>	Converts a document to an array of bytes.
<code>bytesToDocument</code>	Converts an array of bytes to a document.

Service	Description
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

<i>documents</i>	Document List Set of documents from which the documents meeting the retrieve criteria are to be returned.
<i>matchCriteria</i>	<p>Document Criteria on which the documents in the <code>documents</code> parameter are to be matched. Parameters for <code>matchCriteria</code> are:</p> <p>path: Name of the element in <code>documentList</code> whose value provides the value for the search text. The value for <code>key</code> can be a path expression. For example, "Family/Chidren[0]/ BirthDate" retrieves the birthday of the first child from the input <code>Family</code> document list.</p> <p>compareValueAs: Optional. Allowed values are string, numeric, and datetime. The default value is string.</p> <p>datePattern: Optional. Pattern will be considered only if <code>compareValueAs</code> is of type datetime. Default value is MM/dd/yyyy hh:mm:ss a.</p> <p>joins: List of join criteria. Each join criteria consists of:</p> <p>operator: Allowed values are equals, doesNotEqual, greaterThan, greaterThanEqual, lessThan, lessThanEqual, equalsIgnoreCase, contains, doesNotContain, beginsWith, doesNotBeginWith, endsWith, doesNotEndWith.</p> <p>value: Optional. Allowed values are string, numeric, and datetime. The default value is string.</p> <p>joinType: Specifies the way two joins can be linked. Values are "and" or "or". Default value is "and".</p>

Output Parameters

<i>result documents</i>	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

<i>documents</i>	Document List Set of documents in which a new document is to be inserted.
<i>insertDocument</i>	Document The new document to be inserted to the set of documents specified in the <i>documents</i> parameter.
<i>index</i>	String Optional. The position in the set which the document is to be inserted. The <i>index</i> parameter is zero-based. If the value for the <i>index</i> parameter is not specified, the document will be inserted at the end of the document list specified in the <i>documents</i> parameter.

Output Parameters

<i>documents</i>	Document List Document list after inserting the new document.
------------------	--

deleteDocuments

Deletes the specified documents from a set of documents.

Input Parameters

<i>documents</i>	Document List Set of documents that contain the documents you want to delete.
<i>indices</i>	String List Index values of documents to be deleted from the <i>documents</i> parameter document list.

Output Parameters

<i>documents</i>	Document List List of documents whose indices do <i>not</i> match the values in <i>indices</i> 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
<i>pName</i>	<i>cx_timeout</i>

<i>pValue</i>	1000
---------------	------

Key	Value
<i>pName</i>	<i>cx_max</i>

<i>pValue</i>	2500
---------------	------

Key	Value
<i>pName</i>	<i>cx_min</i>

<i>pValue</i>	10
---------------	----

Would be converted to a document containing these three key:

Key	Value
<i>cx_timeout</i>	1000

<i>cx_max</i>	2500
---------------	------

<i>cx_min</i>	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

<i>document</i>	Document Document to transform.
<i>name</i>	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 <code>pName</code> .
<i>value</i>	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 <code>pValue</code> .

Output Parameters

<i>documentList</i>	Document List List containing a document for each key/value pair in the <i>document</i> parameter. Each document in the list will contain two keys, whose names were specified by the <i>name</i> and <i>value</i> 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
<i>cx_timeout</i>	1000
<i>cx_max</i>	2500
<i>cx_min</i>	10

Would be converted to a document list containing these three documents:

Key	Value
<i>pName</i>	<i>cx_timeout</i>
<i>pValue</i>	1000

Key	Value
<i>pName</i>	cx_max
<i>pValue</i>	2500

Key	Value
<i>pName</i>	cx_min
<i>pValue</i>	10

groupDocuments

Groups a set of documents based on specified criteria.

Input Parameters

<i>documents</i>	Document List Set of documents to be grouped based on the specified criteria.
<i>groupCriteria</i>	<p>Document List The criteria on which the input documents are to be grouped. Valid values for the <i>groupCriteria</i> parameter are:</p> <ul style="list-style-type: none"> ■ <i>key</i> . Key in the pipeline. The value for <i>key</i> can be a path expression. For example, "Family/Children[0]/BirthDate" retrieves the birthday of the first child from the input Family document list. ■ <i>compareStringsAs</i> . Optional. Valid values for <i>compareStringsAs</i> are <code>string</code>, <code>numeric</code>, and <code>datetime</code>. The default value is <code>string</code>. ■ <i>pattern</i> . Optional. <i>pattern</i> will be considered only if the <i>compareStringsAs</i> parameter is of type <code>datetime</code>.

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 a service 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 a service 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.

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/Children[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

You can use **List** services to retrieve, replace, or add elements in an Object List, Document List, or String List. You can also use **List** to convert String Lists to Document Lists.

Summary of List services

The following **List** services are available:

Service	Description
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.

appendToDocumentList

Adds documents to a document list.

Input Parameters

<i>toList</i>	Document List Optional. List to which you want to append documents. If you do not specify <i>toList</i> , the service creates a new list.
<i>fromList</i>	Document List Optional. Documents you want to append to the end of <i>toList</i> .
<i>fromItem</i>	Document Optional. Document you want to append to the end of <i>toList</i> . If you specify both <i>fromList</i> and <i>fromItem</i> , the service adds the document specified in <i>fromItem</i> after the documents in <i>fromList</i> .

Output Parameters

<i>toList</i>	Document List The <i>toList</i> document list with the documents in <i>fromList</i> and <i>fromItem</i> 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

<i>toList</i>	String List Optional. List to which you want to append Strings. If the value of <i>toList</i> is null, a null pointer exception error is thrown. If you do not specify <i>toList</i> , the service creates a new list.
<i>fromList</i>	String List Optional. List of Strings to add to <i>toList</i> . Strings are added after the entries of <i>toList</i> .
<i>fromItem</i>	String Optional. String you want to append to the end of <i>toList</i> . If you specify both <i>fromList</i> and <i>fromItem</i> , 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

<i>fromList</i>	String List Optional. List of Strings (a String[]) that you want to convert to a list of documents. If <i>fromList</i> is not specified, the service returns a zero length array for <i>toList</i> .
<i>key</i>	String Optional. Key name to use in the generated document list.

Output Parameters

<i>toList</i>	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

Math

Summary of Math services

You can use the **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
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.
multiplyFloats	Multiplies one floating point number (represented as String) by another and returns the product.

Service	Description
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.

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 $10.0 + \text{NaN} = \text{NaN}$).

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

<i>num1</i>	String Number to add.
<i>num2</i>	String Number to add.
<i>precision</i>	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 $10.0 + \text{NaN} = \text{NaN}$).

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 $10.0 + \text{NaN} = \text{NaN}$).

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:

<u>Value</u>	<u>Description</u>
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 $10.0 + \text{NaN} = \text{NaN}$).

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

Multiplies 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:

<u>Value</u>	<u>Description</u>
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 $10.0 + \text{NaN} = \text{NaN}$).

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.

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:

<u>Value</u>	<u>Description</u>
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`.

String

Summary of String services

You can use **String** services to perform string manipulation and substitution operations. The following **String** services are available:

Service	Description
base64Decode	Decodes a Base-64 encoded string into a sequence of bytes.
base64Encode	Converts a sequence of bytes into a Base64-encoded <code>String</code> .

Service	Description
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.
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.
toLowerCase	Converts all characters in a given string to lowercase.

Service	Description
toUpper	Converts all characters in a given string to uppercase.
trim	Trims leading and trailing white space from a given string.
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 <code>similarityThreshold</code> , it returns the <code>matchedValue</code> . 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.

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:

- `true` to retain the line breaks. This is the default.
- `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. For more information about MTOM implementations, see the *Web Services Developer's Guide*.

bytesToString

Converts a sequence of bytes to a String.

Input Parameters

<i>bytes</i>	byte[] Sequence of bytes to convert to a String.
<i>encoding</i>	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 <i>encoding</i> to <code>autoDetect</code> .
<i>ignoreBOMChars</i>	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: <ul style="list-style-type: none">■ <code>true</code> to remove the byte order mark (BOM) characters before converting the input sequence of bytes to string, if the byte array contains BOM characters.■ <code>false</code> to include the byte order mark (BOM) characters while converting the input sequence of bytes to string. The default is <code>false</code>.

Output Parameters

<i>string</i>	String String representation of the contents of <i>bytes</i> .
---------------	---

concat

Concatenates two strings.

Input Parameters

<i>inString1</i>	String String to which you want to concatenate another string.
<i>inString2</i>	String String to concatenate to <i>inString1</i> .

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:

<u>This symbol...</u>	<u>Indicates...</u>
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 <i>num</i> 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:

<u>Pattern</u>	<u>Description</u>
#,###	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.

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 <i>length</i> equals...	Then <i>value</i> will contain...
7	<i>shipped</i>
10	<i>x9yshipped</i>
12	<i>x9x9yshipped</i>

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

<i>inString</i>	String String that you want to pad.
<i>padString</i>	String Characters to use to pad <i>inString</i> .
<i>length</i>	String Total length of the resulting string, including pad characters.

Output Parameters

<i>value</i>	String Contents of <i>inString</i> followed by as many pad characters as needed so that the total length of the string equals <i>length</i> .
--------------	--

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 <i>length</i> equals...	Then <i>value</i> 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

<i>inString</i>	String String containing the substring to replace.
<i>searchString</i>	String Substring to replace within <i>inString</i> .
<i>replaceString</i>	String Character sequence that will replace <i>searchString</i> . If this parameter is null or empty, the service removes all occurrences of <i>searchString</i> from <i>inString</i> .
<i>useRegex</i>	String Optional. Flag indicating whether <i>searchString</i> is a regular expression. When regular expressions are used to specify a search string, <i>replaceString</i> may also contain interpolation fields (for example, "\$1") that match parenthetical subexpressions in <i>searchString</i> . Set to: <ul style="list-style-type: none">■ <code>true</code> to indicate that <i>searchString</i> is a regular expression.■ <code>false</code> to indicate that <i>searchString</i> is not a regular expression. This is the default.

Output Parameters

<i>value</i>	String Contents of <i>inString</i> with replacements made.
--------------	---

stringToBytes

Converts a string to a byte array.

Input Parameters

<i>string</i>	String String to convert to a byte[].
<i>encoding</i>	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*.

toLowerCase

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.

toUpperCase

Converts all characters in a given string to uppercase.

Input Parameters

inString **String** String to convert.

<i>language</i>	String Optional. Lowercase, two-letter ISO-639 code. If this parameter is null, the system default is used.
<i>country</i>	String Optional. Uppercase, two-letter ISO-3166 code. If this parameter is null, the system default is used.
<i>variant</i>	String Optional. Vendor and browser-specific code. If null, this parameter is ignored.

Output Parameters

<i>value</i>	String Contents of <i>inString</i> , with all lowercase characters converted to uppercase.
--------------	---

trim

Trims leading and trailing white space from a given string.

Input Parameters

<i>inString</i>	String String to trim.
-----------------	-------------------------------

Output Parameters

<i>value</i>	String Contents of <i>inString</i> with white space trimmed from both ends.
--------------	--

URLDecode

Decodes a URL-encoded string.

Input Parameters

<i>inString</i>	String URL-encoded string to decode.
-----------------	---

Output Parameters

<i>value</i>	String Result from decoding <i>inString</i> . If <i>inString</i> contains plus (+) signs, they will appear in <i>value</i> as spaces. If <i>inString</i>
--------------	---

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

<i>inString</i>	String Optional. String to be checked for adherence to the specified date <i>pattern</i> .
<i>pattern</i>	String Date format for specifying the <i>inString</i> 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

<i>isDate</i>	String Indicates whether or not <i>inString</i> follows the specified date pattern. <ul style="list-style-type: none">■ <code>true</code> indicates that <i>inString</i> follows the specified date pattern.■ <code>false</code> indicates that <i>inString</i> does not follow the specified date pattern. The service returns <code>false</code> if <i>inString</i> 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 <i>inString</i> equals...	Then <i>value</i> 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

Summary of Flow services

Use **Flow** services to perform utility-type tasks. The following **Flow** services are available:

Service	Description
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.
countProcessedDocuments	Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen.
logCustomMessage	Logs a message, which can be viewed in the Execution Results screen.

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.

<u>Key</u>	<u>Description</u>
time	String. Date and time the event occurred, in the format <i>yyyy/MM/dd HH:mm:ss.SSS</i>
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:
<u>Key</u>	<u>Description</u>
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:

<u>Key</u>	<u>Description</u>
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.

Input Parameters

None

Output Parameters

\$session **Document** Returns information about the current logged-in user session.

<u>Key</u>	<u>Description</u>
<i>tenantId</i>	String Tenant Identifier.

stageId **String** The stage ID where the current integration resides.

user **Document** Returns user details.

<u>Key</u>	<u>Description</u>
<i>name</i>	String Name of the user who is executing the service.

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

Hashtable

Summary of Hashtable services

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

Summary of Flat File services

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.

Service	Description
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

<i>delimitedDataBytes</i>	java.lang.Byte[] . Delimited data in bytes (Byte array) to convert to a document.
<i>fieldQualifier</i>	String Optional. The delimiter to use for separating entries in <i>delimitedDataBytes</i> . Default is comma (,).
<i>textQualifier</i>	String Optional. The character to use for quoted elements. Default is double quote (").

<i>useHeaderRowForFieldNames</i>	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: <ul style="list-style-type: none"> ■ <i>true</i> . The delimited data of first line will be used as the property name in the output document. This is the default. ■ <i>false</i> . column1, column2...columnN will be used as the property name in the output document.
<i>encoding</i>	String Optional. The encoding to use while parsing the delimited data.

Output Parameters

<i>document</i>	Document. Document resulting from the conversion of <i>delimitedDataBytes</i> . 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

delimited **java.io.InputStream**. Delimited data in an input stream to convert to a *DataStream* 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 (").

<i>useHeaderRowForFieldNames</i>	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: <ul style="list-style-type: none"> ■ <i>true</i> . The delimited data of first line will be used as the property name in the output document. This is the default. ■ <i>false</i> . column1, column2...columnN will be used as the property name in the output document.
<i>encoding</i>	String Optional. The encoding to use while parsing the delimited data.

Output Parameters

<i>document</i>	Document. Document resulting from the conversion of <i>delimitedDataStream</i> . 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

<i>delimited DataString</i>	String . Delimited string to convert to a document.
<i>fieldQualifier</i>	String Optional. The delimiter to use for separating entries in <i>delimitedDataString</i> . Default is comma (,).
<i>textQualifier</i>	String Optional. The character to use for quoted elements. Default is double quote (").

-
- useHeader* **String** Optional. Consider first line as header row and use the delimited data of this line as property names in the output document.
- RowForFieldNames* 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

<i>document</i>	Document. Document to be converted to delimited data bytes (byte array object). This document contains a document array <code>rows[]</code> corresponding to the delimited data.
<i>fieldQualifier</i>	String Optional. The delimiter to use for separating entries in <i>delimitedDataBytes</i> . Default is comma (,).
<i>textQualifier</i>	String Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p>String Optional. The first line in the output delimited data <i>delimitedDataBytes</i> will be constructed using the property names in the input document array <code>document \ rows[]</code>. Set to:</p> <ul style="list-style-type: none"> ■ <i>true</i> . Property names in the input document array <code>document \ rows[]</code> will be used as the first row in the output <i>delimitedDataBytes</i> . ■ <i>false</i> . <code>column1, column2...columnN</code> will be used as the first row in the output <i>delimitedDataBytes</i> .
<i>encoding</i>	String Optional. The encoding to use while parsing the delimited data.














Output Parameters

<i>delimitedDataBytes</i>	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

<i>document</i>	Document. Document to be converted to delimited data stream. This document contains a document array rows[] corresponding to the delimited data.
<i>fieldQualifier</i>	String Optional. The delimiter to use for separating entries in <i>delimitedDataStream</i> . Default is comma (,).
<i>textQualifier</i>	String Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p>String Optional. The first line in the output delimited data <i>delimitedDataStream</i> will be constructed using the property names in the input document array document \ rows[]. Set to:</p> <ul style="list-style-type: none"> ■ <i>true</i> . Property names in the input document array document \ rows[] will be used as the first row in the output <i>delimitedDataStream</i> . ■ <i>false</i> . column1, column2...columnN will be used as the first row in the output <i>delimitedDataStream</i> .
<i>encoding</i>	String Optional. The encoding to use while parsing the delimited data.















Output Parameters

<i>delimitedDataStream</i>	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

<i>document</i>	Document. Document to be converted to delimited data string. This document contains document array rows[] corresponding to the delimited data.
<i>fieldQualifier</i>	String Optional. The delimiter to use for separating entries in <i>delimitedDataString</i> . Default is comma (,).
<i>textQualifier</i>	String Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p>String Optional. The first line in the output delimited data <i>delimitedDataString</i> will be constructed using the property names in the input document array document \ rows[]. Set to:</p> <ul style="list-style-type: none"> ■ <i>true</i> . Property names in the input document array document \ rows[] will be used as the first row in the output <i>delimitedDataString</i> . ■ <i>false</i> . column1, column2...columnN will be used as the first row in the output <i>delimitedDataString</i> .
<i>encoding</i>	String Optional. The encoding to use while parsing the delimited data.

Output Parameters

<i>delimitedDataString</i>	String. Delimited data byte string resulting from the conversion of a document.
----------------------------	--

JSON

Summary of JSON services

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.

Service	Description
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

<i>document</i>	Document. The document to be converted to a JSON string.
<i>prettyPrint</i>	String. Formats the <i>jsonString</i> output parameter for human readability by adding carriage returns and indentation to the JSON content. Set to: <ul style="list-style-type: none">■ <i>true</i> to format the <i>jsonString</i> output field for human readability■ <i>false</i> to leave the <i>jsonString</i> output field in its unformed state The service will not add any additional carriage returns or indentation to the JSON content.

Output Parameters

<i>jsonString</i>	Object. JSON string resulting from the conversion of a document.
-------------------	---

jsonBytesToDocument

Converts JSON content in bytes (byte array) to a document.

Input Parameters

<i>jsonBytes</i>	java.lang.Byte[]. JSON content in bytes (byte array) to convert to a document.
<i>decodeRealAsDouble</i>	String. Optional. Converts real numbers from <i>jsonBytes</i> to either a Float or Double Java wrapper type. Set to: <ul style="list-style-type: none">■ <i>true</i> to convert real numbers to Double Java wrapper types■ <i>false</i> to convert real numbers to Float Java wrapper types Default value is <i>true</i> .
<i>decodeIntegerAsLong</i>	String. Optional. Converts integers from <i>jsonBytes</i> 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* .

XML

Summary of XML services

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.

Service	Description
documentToXMLString	Converts a document to xml content string.
xmlBytesToDocument	Converts XML content bytes (byte array) to a document.
xmlStreamToDocument	Converts an XML content stream to a document.
xmlStringToDocument	Converts an XML string to a document.

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.

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.

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.

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:

The screenshot shows a configuration interface for `nsDecls`. It is a tree view where the root is `nsDecls []`. Underneath, there are two array elements: `nsDecls [0]` and `nsDecls [1]`. Each array element contains two fields: `prefix *` and `uri *`. For `nsDecls [0]`, the `prefix` is `GSX` and the `uri` is `http://www.gsx.com`. For `nsDecls [1]`, the `prefix` is `TxMon` and the `uri` is `http://www.acrtrak/txmonitor`. Each field has a text input box and a three-dot menu icon to its right.

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
<pre><myDoc><e1>e1Value</e1> </myDoc></pre>	
<pre><?xml version="1.0" encoding="UTF-8" standalone="no"?><myDoc><e1>e1Value </e1></myDoc></pre>	

```
<?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.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	DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc	
e1	e1Value	
e2	e2Value	

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
phoneNumber	
*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.

`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 `xmlStreamToDocument` will produce:

XML Document	Document
<pre><myDoc><e1>e1Value</e1> </myDoc></pre>	<pre> document ├── myDoc │ └── e1 │ └── e1Value </pre>


```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>
e1Value</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
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
.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		DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc
e1		e1Value
e2		e2Value

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
phoneNumber	
*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.

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
<pre><myDoc><e1>e1Value</e1> </myDoc></pre>	<pre> graph TD document --> myDoc myDoc --> e1[e1] e1 --> e1Value[e1Value] </pre>

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>
e1Value</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
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.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		DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc
e1		e1Value
e2		e2Value

IO

You can use the 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. For more information about `InputStreams`, see the Java documentation.

Summary of IO services

Note: 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.

Service	Description
bytesToStream	Converts a <code>byte[]</code> to <code>java.io.ByteArrayInputStream</code> .

Service	Description
<code>streamToBytes</code>	Creates a <code>byte[]</code> from data that is read from an <code>InputStream</code> .

bytesToStream

Converts a `byte[]` to `java.io.ByteArrayInputStream`.

Input Parameters

<i>bytes</i>	byte[] The byte array to convert.
<i>length</i>	String Optional. The maximum number of bytes to read and convert. If <i>length</i> is not specified, the default value for this parameter is the length of the input byte array.
<i>offset</i>	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

<i>stream</i>	java.io.ByteArrayInputStream An open <code>InputStream</code> created from the contents of the input <i>bytes</i> 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

<i>stream</i>	java.io.InputStream The <code>InputStream</code> 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.

Integration Details

This screen allows you to view at which stage the Integration is running, the components used to create the Integration, when the Integration was created or last modified, who created or last modified the Integration, when was the last execution, and whether the Integration is scheduled. You can delete, edit, or expose the Integration as a REST service from this screen and also view the last five execution results.

Option	Description
Created on	Displays the date and time when the Integration was created.
Created by	Displays the user who created the Integration.
Stages tabs	Displays different stages of the Integration development. You can pull Integrations from all other stages except from the Development stage.
Overview	Provides an overview of the Integration at each stage, that is, the components used to create the Integration, when the Integration was last modified, who last modified the Integration, when was the last execution, and whether the Integration is scheduled. You can also schedule, run, pause, or expose the Integration as a REST service from this screen.
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 a stage. See Execution Results for information on the Last 5 Execution Results table columns.
Edit	Click this option to modify the Integration.
Delete	Click to this option delete the Integration from a stage.

Option	Description
Uses	Displays the components used to create the Integration.
Last modified/ Last modified by	When and by whom was the Integration last modified.
Status	If the Integration in a stage is scheduled, the status of the Integration displays Scheduled , else it appears as Not Scheduled . The Status appears as Schedule Paused if the Integration has been paused.
Last execution	When was the Integration last executed. A warning message appears if the last execution was not successful.
Next scheduled execution	When is the Integration scheduled to run again.
Schedule	<p>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.</p> <p>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. Click the  icon to delete the selected execution time for daily, weekly, and monthly schedules.</p> <p>Click Reset if you want to permanently remove the current recurrence schedule.</p> <p>Click Next to provide inputs to the Integration based on the defined input signature.</p>
Run Now	Click this option to submit the Integration for execution. You can provide inputs to the Integration based on the defined input signature.
Pause	Click this option to pause the scheduled Integration.
Resume	Click this option to start the Integration that was paused.

Option	Description
Pull	Click this option to pull an Integration from a preceding stage into this stage. You can pull an Integration after you have configured the Accounts for each stage in the Account Configuration page.
Remove	Click this option to remove an Integration from a stage. You can remove an Integration from all stages except from the <i>Development</i> stage.
Expose as a REST service	<p>Select this option if you want to trigger the execution of an Integration from an external system. By default, Integrations built in Integration Cloud are not accessible using REST. This feature provides you with one more option to trigger Integration executions from a software application, for example, a REST client, apart from manual and scheduled integrations from the user interface.</p> <p>Once the Integration is exposed as a REST service, the REST 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.</p> <p>Provide the usage URL, HTTP Method, modified or the sample JSON input, and necessary parameters in the external program, including the required security credentials (user name and password) while submitting the REST service.</p> <p>After the request is sent, the response will contain a status indicating whether the Integration has been submitted for execution. The response will also contain a reference to the execution result identifier so that a new REST call can be made later to get the execution results.</p> <p><i>Application Status Codes</i></p> <ul style="list-style-type: none">■ 0 - SUCCESS: Successfully submitted the Integration for execution.■ -1 - ERROR: Problem while submitting the Integration for execution. <p><i>HTTP Status Codes</i></p> <ul style="list-style-type: none">■ 200 - OK■ 201 - Created■ 500 - Internal Server Error■ 401 - Unauthorized User Error

Option	Description
	<p>To get the execution results, construct the URL of the new REST call from the URI field available in the <i>Response</i> section.</p> <p>To construct the URL of the new REST call, add the response URI obtained from <i>resultReference</i> in the <i>Response</i> section to:</p> <pre data-bbox="477 495 1369 520">https://<sub-domain>.webMethodscloud.com/integration</pre> <p>Example:</p> <pre data-bbox="477 590 1438 667">https://<sub-domain>.webmethodscloud.com/integration/rest/assembly/external/execution/result?resultReference=76fb5733-6a21-4b02-864f-5e958f698373</pre> <p><i>Application Status Codes</i></p> <ul style="list-style-type: none"> ■ 0 - SUCCESS ■ -1 - ERROR <p><i>HTTP Status Codes</i></p> <ul style="list-style-type: none"> ■ 200 - OK ■ 500 - Internal Server Error ■ 401 - Unauthorized User Error ■ 404 - Not Found <div data-bbox="477 1119 1369 1348" style="background-color: #f0f0f0; padding: 10px;"> <p>Note: 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 status code is 404 - Not Found, it means that either the Integration is not yet run or the <i>resultReference</i> is not correct.</p> </div>

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.

Note: All Integrations created from recipes are initially copied to the development stage.

To view and use recipes

1. From the Integration Cloud navigation bar, click **Recipes**. The **Recipes** screen appears. By default, recipes for all Applications and for all Integration types (Orchestrated and Point-to-Point) are displayed. You can filter recipes based on a specific Application and for a specific Integration type. The Application filter displays only those Applications that are used in the recipes. The **Recipes** screen also displays the number of times each recipe has been used to create Integrations and the Applications referenced in the recipe. If there are more than two Applications referenced in the recipe, the **Recipes** screen also displays the incremental number.
2. From the **Recipes** screen, for a recipe, click **Preview** to see a view-only mode of the Integration details of the recipe.
3. From the **Recipes** screen, for a recipe, click **Details** to view a detailed description of the recipe and the references in the **Recipe Details** screen.
4. From the **Recipes** or **Recipe Details** screen, click **Use** if you want to apply the recipe to create a new Integration. The **Connect to Applications** screen appears. Depending on the number of Integrations referenced in your recipe, a message is displayed at the top of the screen. You will be asked to configure all the Integrations in your recipe one after another.
5. In the **Connect to Applications** screen, select the **Account** for each Application or create a new Account, and then click **Next**.

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 recipe name and recipe description is displayed.
7. Click **Finish**. If you have existing references 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. Deselect the references that you do not want to replace. Click **Continue** to replace or overwrite all the selected references from the recipe in the development stage. The Integration details screen appears for the newly created Integration.

Document Types

A **Document Type** contains a set of fields used to define the structure and type of data in a document. You can use a Document Type to specify the input or output parameters for an Integration.

Note: Integration Cloud also allows you to create Document Types for already created REST Applications from the **Applications > REST Application > Document Types** link or from the Request Body and Response Body panels while creating a REST Application. Document Types created for a REST Application does not appear in the **Develop > Document Types** screen 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**. See "[Creating REST Applications](#)" on page 65 for information on REST Resources and Actions.

Note: Users who have the required access privileges under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Document Types** can create, update, and delete a Document Type.

Document Types can provide the following benefits:

- Using a Document Type as the input or output signature for an Integration can reduce the effort required to build an Integration.
- Using a Document Type to build document or document list fields can reduce 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 can make a change in one place (the Document Type) rather than everywhere the Document type is used.

You can 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 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. 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.

You can create a Document Type by defining the structure of the Document Type yourself by inserting fields to define its contents and structure.

Note: When you edit a Document Type, any change is automatically propagated to all Integrations that use or reference the Document Type.

To add or edit a Document Type



1. From the Integration Cloud navigation bar, click **Develop > Document Types**. The **Document Types** page appears.

Note: You can create Document Types for already created REST Applications from the **Applications > REST Application > 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** screen and click **Edit**. Select a field to view the **Field Properties** panel.
3. To create a new Document Type, from the **Document Types** page, click **Add New Document Type**.
4. Provide a name and description of your Document Type. Required fields are marked with an asterisk on the screen.
5. You can 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.
6. Click the **+** icon to add a new field. You can update the field properties by using the **Field Properties** window.

Provide the **Name** and **Type** of the fields in order to define the structure and content of the Document Type. A field can be a String, String list, Document, Document list, Document Reference, Document Reference List, Object, or Object list. If you select Document Reference or Document Reference List, choose **Document Reference** and if you select Object or Object list, choose **Object Wrapper Type**. While defining fields of a Document Type, you can specify Integer, Short, Long, Float, Double, and Boolean as types. 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 data type of a field, you can set properties that specify an **XML Namespace** and indicate whether the field is required at runtime by selecting the **Required** option.

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**.

The new Document Type appears in the **Document Types** screen.

Reference Data

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.

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: Users who have the required access privileges under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Reference Data** can create, update, or delete a Reference Data.

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.

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.

To add or edit a Reference Data

1. From the Integration Cloud navigation bar, click **Develop > Reference Data**. The **Reference Data** screen appears.

2. To edit an existing Reference Data, select a Reference Data from the **Reference Data** screen and click **Edit**.
3. To create a new Reference Data, from the **Reference Data** screen, click **Add New Reference Data**.
4. Provide a name and description of your Reference Data. Required fields are marked with an asterisk on the screen.
5. 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.

Note: The **Reference Data** block will appear 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. If a Reference Data is used by an Integration, you will not be able to delete the Reference 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:





fx AccountType Input

🔗
▼

- ▼  matchCriteria [] *
-  path *
-  compareValueAs
-  datePattern
- ▶  joins [] *

fx AccountType Output

🔗
▼

- ▼  AccountType []
-  Type
-  Our Type
-  Marketer

15 Monitor

■ Dashboard	296
■ Execution Results	297
■ Audit Log	298

You can view and monitor Integration executions and performance details on the **Dashboard**. The **Execution Results** screen allows you to view the audit trail of all the executions that happened in a stage for an Integration or for all Integrations. The **Audit Log** screen displays logs related to additions, deletions, updates, export, schedule, login, logout, password changes, record access attempts, access violations, deployments, and so on.

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 selected stage. If you have **Live** and **Development** stage permissions, Integration Cloud displays the Live stage. If you do not have the **Live** stage permission, Integration Cloud displays the **Development** stage. You can view the **Dashboard** if the Access Profile assigned to you is also specified for that stage in the **Stage Management** screen. Further, you must have the **Access** permission under **Settings > Access Profiles > Administrative Permissions > Functional Controls > Dashboard** to view the dashboard.

The **Dashboard** displays the following details:

- Total number of Integrations in the stage selected from the drop-down list based on your permissions.
- Drop-down list to select an Integration or select all Integrations to view the execution details.
- Drop-down list to select a stage to view the execution details.
- Total number of documents processed by an Integration or for all Integrations in a 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.
- Number of completed and failed Integration executions that happened during the selected time period in a stage.
- Number of Integration executions that have completed with errors during the selected time period in a 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 a stage.
- Number of in-progress Integration executions in a stage. You can click on the number to view the in-progress Integration execution details in a table.
- 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 a 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, 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** screen allows you to view the audit trail of all the executions that happened in a stage or for all stages for an Integration or for all Integrations, during a specified time period.

To view the execution results

1. From the Integration Cloud navigation bar, click **Develop**. The **Integrations** screen appears.
2. From the **Integrations** screen, select the Integration for which you want to view the execution results.
3. Click the *Integration link* to view the Integration **Overview** screen. You can click **Edit** to modify the Integration or click **Delete** to delete the Integration from this screen. You can also see the last five execution results in the **Last 5 Execution Results** tab.

You can also access the **Execution Results** link from the home page or click **Monitor > Execution Results** to view the **Execution Results** screen.

4. In the **Execution Results** screen, select the **Integration**, the **Stage**, and the time period for which you want to view the execution results. You can also select **All Integrations** and **All Stages** to view the execution results for all Integrations in all the stages for a specific 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.

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**, **Failed**, and **Completed with errors** are displayed.
 - **Completed** - 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. 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.
 6. Click on a row in the table to view more information about the selected Integration Execution. The **Execution Details** screen appears.

In the **Execution Details** screen, 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.

7. The screen 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 Message** 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.

Audit Log

The **Audit Log** feature allows you to access logs related to additions, deletions, updates, export, schedule, login, logout, password changes, record access attempts, access violations, deployments, and so on for a user.

To view the **Audit Log**, from the Integration Cloud navigation bar, click **Monitor > 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, Add, Delete, Update, Login, Logout, and so on. **Description** refers to a summary of the action performed.

Click **Update Retention Period** and specify the number of days to retain the Audit Log entries. You can retain log entries up to 365 days. Logs whose age exceeds the specified retention period are deleted. Default value of the Retention Period is 1.

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.