

# webMethods Integration Cloud Help

Version 2.1.0

January 2016

This document applies to webMethods Integration Cloud Version 2.1.0 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 © 2014-2016 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).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

---

## Table of Contents

<b>New Registration</b> .....	<b>7</b>
Creating an Account.....	8
Securing your Account.....	9
<b>User Information</b> .....	<b>11</b>
Editing Security Question and Answer.....	12
<b>Managing Users</b> .....	<b>13</b>
Adding Users.....	14
Updating User Information.....	17
Resetting Passwords.....	18
<b>Managing Access Profiles</b> .....	<b>19</b>
Adding or Updating Access Profiles.....	20
<b>Advanced Security</b> .....	<b>23</b>
Add Keystore.....	24
Add Truststore.....	25
Add Partner Certificate.....	26
<b>Stages Management</b> .....	<b>27</b>
Applying Access Profiles to a Stage.....	29
<b>Audit Log</b> .....	<b>31</b>
<b>Company Information</b> .....	<b>33</b>
Updating Company Information.....	34
<b>Managing Password Policies</b> .....	<b>37</b>
Updating Password Policy Settings.....	38
<b>Applications</b> .....	<b>41</b>
Adding Custom SOAP Applications.....	43
<b>Managing Accounts</b> .....	<b>45</b>
Adding or Editing Accounts.....	47
Account Configuration.....	48
<b>Managing Operations</b> .....	<b>57</b>
Adding or Editing Operations.....	58
<b>Integrations</b> .....	<b>61</b>
Creating Point-to-Point Integrations.....	62
Creating Orchestrated Integrations.....	65

---

Pipeline and Signatures.....	74
Built-In Services.....	78
Date.....	78
Summary of Date services.....	82
calculateDateDifference.....	83
compareDates.....	84
dateBuild.....	85
dateTimeBuild.....	86
dateTimeFormat.....	88
getCurrentDateString.....	89
incrementDate.....	90
Document.....	92
Summary of Document services.....	92
findDocuments.....	92
insertDocument.....	93
deleteDocuments.....	94
documentListToDocument.....	95
documentToDocumentList.....	96
List.....	98
Summary of List services.....	98
appendToDocumentList.....	98
appendToStringList.....	99
sizeOfList.....	100
stringListToDocumentList.....	101
Math.....	102
Summary of Math services.....	103
absoluteValue.....	103
addFloatList.....	103
addFloats.....	104
addIntList.....	105
addInts.....	106
divideFloats.....	106
divideInts.....	107
max.....	108
multiplyFloatList.....	108
multiplyFloats.....	109
multiplyIntList.....	110
multiplyInts.....	111
randomDouble.....	112
roundNumber.....	112
subtractFloats.....	113
subtractInts.....	114
String.....	114
Summary of String services.....	116
base64Decode.....	116

---

base64Encode.....	116
bytesToString.....	117
concat.....	118
indexOf.....	118
length.....	119
lookupDictionary.....	119
makeString.....	120
messageFormat.....	120
numericFormat.....	121
padLeft.....	122
padRight.....	123
replace.....	124
stringToBytes.....	125
substring.....	125
tokenize.....	126
toLowerCase.....	126
toUpperCase.....	127
trim.....	128
URLDecode.....	128
URLEncode.....	128
fuzzyMatch.....	129
Flow.....	130
Summary of Flow services.....	130
getLastError.....	130
Hashtable.....	132
Summary of Hashtable services.....	132
containsKey.....	132
createHashtable.....	133
get.....	133
listKeys.....	134
put.....	134
remove.....	134
size.....	135
Flat File.....	135
Summary of Flat File services.....	136
delimitedDataBytesToDocument.....	136
delimitedDataStreamToDocument.....	139
delimitedDataStringToDocument.....	142
documentToDelimitedDataBytes.....	145
documentToDelimitedDataStream.....	147
documentToDelimitedDataString.....	149
JSON.....	151
Summary of JSON services.....	152
documentToJSONBytes.....	152
documentToJSONStream.....	152

---

documentToJSONString.....	153
jsonBytesToDocument.....	153
jsonStreamToDocument.....	154
jsonStringToDocument.....	155
XML.....	155
Summary of XML services.....	156
documentToXMLBytes.....	156
documentToXMLStream.....	161
documentToXMLString.....	165
xmlBytesToDocument.....	170
xmlStreamToDocument.....	175
xmlStringToDocument.....	181
IO.....	186
Summary of IO services.....	186
bytesToStream.....	187
streamToBytes.....	187
Document Types.....	188
Reference Data.....	190
Reference Data Signature.....	191
Integration Details.....	193
Execution Results.....	196

# 1 New Registration

---

- Creating an Account ..... 8
- Securing your Account ..... 9

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).

### Related Topics

[Creating an Account](#)

[Securing your Account](#)

[User Information](#)

[Editing Security Question and Answer](#)

## 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:

Field	Description
<b>First Name</b>	Provide your first name. You can change the value after the user is created from the <b>Settings &gt; Users</b> screen.
<b>Last Name</b>	Provide your last name. You can change the value after the user is created from the <b>Settings &gt; Users</b> screen.
<b>Company</b>	Provide your company name. You can change that later from the <b>Settings &gt; Company Information</b> screen.
<b>Country</b>	Provide your country name. You can change that later from the <b>Settings &gt; Users</b> screen.



Field	Description
<b>State or Province</b>	Provide your State or Province name. You can change that later from the <b>Settings &gt; Users</b> screen.
<b>Phone</b>	Provide your phone number. You can change that later from the <b>Settings &gt; Users</b> screen.
<b>Sub-Domain for Service Portal</b>	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.
<b>Email Address</b>	Provide 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 <b>Settings &gt; Users</b> screen.
<b>Promo Code</b>	Enter a valid promotion code if you have one, for availing subscription benefits.
<b>I agree to the Terms of Service</b>	Select this option to agree to the webMethods Integration Cloud Terms of Service.

- Click "Submit" to continue to the next step to activate and secure your account or click "Cancel" to go back to the Login screen. After you click "Submit" and as soon as the registration process is complete, an email will be sent to the email address you provided during registration. Use the temporary password sent in the email to log in. You will be asked to change your password.

### Related Topics

[Securing your Account](#)

[User Information](#)

[Editing Security Question and Answer](#)

## 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, and then click **Submit**. 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.

### Related Topics

[Creating an Account](#)

[User Information](#)

[Editing Security Question and Answer](#)

## 2 User Information

---

- Editing Security Question and Answer ..... 12

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.

#### **Related Topics**

[Managing Users](#)

[Resetting Passwords](#)

[Adding Users](#)

[Updating Password Policy Settings](#)

---

## **Editing Security Question and Answer**

---

#### **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.

#### **Related Topics**

[Creating an Account](#)

[Securing your Account](#)

[User Information](#)

# 3 Managing Users

---

- Adding Users ..... 14
- Updating User Information ..... 17
- Resetting Passwords ..... 18

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.

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

### Related Topics

[User Information](#)

[Resetting Passwords](#)

[Updating User Information](#)

[Adding Users](#)

[Managing Password Policies](#)

## 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 in the screen.

Field	Description
<b>First name</b>	User's first name as it should appear in the platform.
<b>Last name</b>	User's last name as it should appear in the platform.
<b>Title</b>	User's professional title.
<b>Access Profile</b>	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. Specify one of the following Access Profiles:

Field	Description
	<ul style="list-style-type: none"> <li>■ <b>Administrator</b> - Provides permissions needed by the System Administrator.</li> <li>■ <b>Regular User</b> - Provides permissions that are more appropriate for normal users.</li> </ul>
	<p>By default, the system administrator can change the Administrative Permissions associated with each Access Profile (except the above mentioned <b>Administrator</b> Access Profile), and can add additional Access Profiles, as needed.</p>
	<p><b>Note:</b> By default, the <b>Administrator</b> and <b>Regular User</b> 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 <a href="#">Adding or Updating Access Profiles</a> for more information.</p>
<b>Employee Number</b>	Optional identification number for each employee.
<b>Email</b>	Email address of the user. The email address is used to send the welcome and login information to the user.
<b>User name</b>	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.
<b>Partner</b>	Select this option if the user is a Partner user.
	<p><b>Note:</b> If <b>Allow User Interface Access</b> permission available under <b>Access Profile &gt; Administrative Permissions &gt; Account Controls</b> is not enabled, a Partner User can still perform on-premise tasks.</p>
<b>Active</b>	Select this option to indicate that the user account is active. You can use this option to reactivate a locked or disabled user account.
<b>Force password change on first login</b>	This option is available only when creating a new user. Select this option to force the user to change the login password when logging in for the first time.

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

Field	Description
<b>Time Zone</b>	Choose a Time Zone Code from the drop down list.
<b>Date Format</b>	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 <b>Settings &gt; Company Information &gt; Advanced Information</b> tab. Administrators and Users can change the default selection in the <b>Users</b> screen.
<b>Time Format</b>	Select a 12-hour clock (hh:mm a) with AM/PM, or a 24-hour clock (HH:mm).
<b>Locale</b>	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
<b>Phone</b>	Primary phone number for the user.
<b>Mobile Phone</b>	Mobile phone number for the user.
<b>Fax</b>	Fax number for the user.
<b>Street Address</b>	Street address for the user.
<b>City</b>	City for the user.
<b>State/Province</b>	State or province for the user.
<b>Postal/Zip Code</b>	Postal or ZIP Code for the user.
<b>Country</b>	Country for the user.

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



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

## Related Topics

[Managing Users](#)

[Resetting Passwords](#)

[Updating User Information](#)

## Updating User Information

### To edit or update the user information

1. From the Integration Cloud navigation bar, click Settings > Users.
2. Select a user from the list and then click Edit.
3. Make necessary modifications. See "[Adding Users](#)" on page 14 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 in 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.
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.

#### Related Topics

[Managing Users](#)

[Resetting Passwords](#)

[Adding Users](#)

## 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 an email notification to the user regarding the password reset.

**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.

#### Related Topics

[Managing Users](#)

[Adding Users](#)

[Updating User Information](#)

# 4 Managing Access Profiles

---

■ Adding or Updating Access Profiles .....	20
--	----

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**. To create a new Access Profile, click **Add New Access Profile**.

### Related Topics

[Adding or Updating Access Profiles](#)

[Managing Users](#)

[Managing Password Policies](#)

---

## 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 change any field in 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 in the screen.

Field	Description
<b>Name</b>	Provide a name for the Access Profile. You can reference the profile by name when assigning it to a user.
<b>Description</b>	Provide a general description for the Access Profile.

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

Field	Description
<b>IP Address Ranges</b>	<p>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.</p> <p><b>Note:</b> A maximum of 25 IP address ranges can be specified. 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</p> <p>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.</p>

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

Field	Description
<b>User and Ownership Controls</b>	<p><b>User Management</b> - Select this option if you want to add, update, delete users, or assign users to Access Profiles.</p> <p><b>Access Control</b> - 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
	<p><b>Manage Personal Setup</b> - Select this option if you want to allow a user to modify the personal information.</p>
<b>Account Controls</b>	<p><b>Manage Company Capabilities</b> - Select this option if you want to allow users to modify the company information.</p> <p><b>Allow User Interface Access</b> - 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><b>Note:</b> If the <b>Allow User Interface Access</b> permission is not enabled for a user and if the user is a Partner user, then that user will still be able to perform on-premise tasks.</p>
<b>Data Management Controls</b>	<p><b>Manage Audit Log</b> - 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 <b>Audit Log</b> page, from the Integration Cloud navigation bar, click <b>Settings &gt; Audit Log</b>.</p>
<b>Functional Controls</b>	<p>Select the required options under <b>Stages, Advanced Security, Accounts, Operations, Reference Data, Document Types, and Integrations</b>. You must select the required permissions to create, update, delete, or administer those functions.</p>

**Related Topics**[Managing Access Profiles](#)[Audit Log](#)[Managing Users](#)[Managing Password Policies](#)

# 5 Advanced Security

---

- Add Keystore ..... 24
- Add Truststore ..... 25
- Add Partner Certificate ..... 26

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 Custom 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**.

### Related Topics

[Add Keystore](#)

[Add Truststore](#)

[Add Partner Certificate](#)

[Adding or Editing Accounts](#)

[Custom SOAP Application](#)

## 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 Custom 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 Custom SOAP Application.

**Related Topics**

[Advanced Security](#)

[Custom 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 Custom 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 Custom SOAP Application.

#### Related Topics

[Advanced Security](#)

[Custom 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 Custom 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 Custom SOAP Application.

#### Related Topics

[Advanced Security](#)

[Custom SOAP Application](#)

# 6 Stages Management

---

- Applying Access Profiles to a Stage ..... 29

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**.

### Related Topics

[Advanced Security](#)

[Applying Access Profiles to a Stage](#)

[Managing Access Profiles](#)

[Managing Accounts](#)

[Managing Operations](#)

[Integrations](#)

[Managing Users](#)

---

## 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, then 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, then 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 > Stages Management. All stages added including the Development stage is displayed. Initially, before any other stages are added, the Development stage is displayed.
2. Click the Access Profiles icon and select the Access Profiles you want to apply to the desired 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, then only those will be pulled into the next stage.

**Related Topics**[Stages Management](#)[Managing Users](#)[Managing Access Profiles](#)[Managing Operations](#)[Integrations](#)[Managing Accounts](#)

# 7 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**, select **Settings > Audit Log** from the Integration Cloud navigation bar.

**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 > Manage Audit Log**.

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, 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 log entries for a specified period. You can download Audit logs only up to 30 days.

## Related Topics

[Managing Access Profiles](#)





# 8 Company Information

---

■ Updating Company Information .....	34
--------------------------------------	----

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.

### Related Topics

[New Registration](#)

[Creating an Account](#)

[User Information](#)

[Managing Users](#)

[Managing Password Policies](#)

[Managing Access Profiles](#)

## 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 in the screen.

Field	Description
<b>Tenant ID</b>	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 <b>Basic</b> tab.
<b>Company Name</b>	The name of the company. This field accepts only alphanumerics, spaces, and hyphens (-). The company name is automatically populated from the <b>Registration</b> screen.
<b>Street</b>	The street address of the company.
<b>City</b>	The city where the company is located.

Field	Description
<b>State/Province</b>	The state or Province where the company is located. The state or province name is automatically populated from the <b>Registration</b> screen.
<b>Postal/Zip Code</b>	The postal or zip code for the company.
<b>Country</b>	The country where the company is located. The country name is automatically populated from the <b>Registration</b> screen.
<b>System Notification Email Addresses</b>	Enter an address or comma-separated email addresses to receive system notifications. Such notifications can occur, for example, when a connection to the system mailbox fails after repeated attempts. This field displays the information from the <b>Registration</b> screen but you can change that later using the <b>Edit</b> button.

4. On the **Advanced Information** tab, complete the following fields:

Field	Description
<b>Time Zone</b>	Choose your time zone from the drop down list.
<b>Time Format</b>	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
<b>Date Format</b>	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.
<b>Default Locale</b>	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.
<b>Last Modified</b>	This field displays the date and time when the company information record was last updated and cannot be edited.

#### Related Topics

[Company Information](#)

[Creating an Account](#)

User Information

Managing Users

Managing Password Policies

Managing Access Profiles

# 9 Managing Password Policies

---

■ Updating Password Policy Settings .....	38
---	----

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.

### Related Topics

[Updating Password Policy Settings](#)

[Managing Users](#)

[Managing Access Profiles](#)

## 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
<b>Minimum Length</b>	Specify the minimum number of characters in the password. Default: 6 characters. Range: 6-10 characters.
<b>Required Character types</b>	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.
<b>Expires in</b>	Specify 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> .
<b>New Password cannot match</b>	The new password cannot match the number of previous passwords. Default: Last Password. Range: Last 2-5 passwords.

Field	Description
<b>Minimum Age</b>	Specify the number of days that must pass before a user can change passwords. Default: No Minimum. Range: 1-5 Days.
<b>Account Lockout Threshold</b>	<p>Specify 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>
<b>Account Lockout Duration</b>	Specify the length of time that an account is locked out. Default: 15 minutes. Choices: 5, 10, 15, 30 minutes, 1 hour, Disable.
<b>Record Information</b>	<p>For audit purposes, the following information is displayed after you save the record:</p> <p>Last Modified By &lt;username&gt; on {date} &lt;time&gt; Created by System.</p>

4. Click Apply.

#### Related Topics

[Managing Password Policies](#)

[Managing Users](#)





# 10 Applications

---

- Adding Custom SOAP Applications ..... 43

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, which enable you to connect to the particular SaaS providers.

Custom SOAP Applications, which are used to access third party Web Services are also listed in this page. To create a Custom SOAP Application, click **Add New Application**. A File Transfer Protocol (FTP/FTPS) Application is also available that allows Integration Cloud to connect to an FTP server using the FTP protocol and provides operations to list, download, upload, and delete files. It also supports FTP over SSL connection.

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.

You can also create Accounts and Operations for an Application and Integrations between different SaaS applications from the **Applications** page. For any Application, you can select **Accounts**, **Operations**, or **Integrations** if you want to create or edit them for that Application.

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.

## Related Topics

[Advanced Security](#)

[Adding Custom SOAP Applications](#)

[Managing Accounts](#)

[Managing Operations](#)

[Integrations](#)

[Managing Access Profiles](#)

[Stages Management](#)

---

## Adding Custom SOAP Applications

---

The “Applications” screen allows you to create a Custom SOAP Application. The Custom SOAP Application enables you to access third party Web Services hosted in the cloud or on-premise 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".
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 in the screen.
4. In the "Type" field, select the Application type. For example, select "SOAP" if you want to create a SOAP-based Application.
5. Click "Browse" next to the "Application Icon" if you want to select a different icon for your custom Application. The icon must be an svg file and the size cannot exceed 50 KB.
6. 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.
7. Enter the user name and password in the "Authentication" section if authentication is required to access the WSDL URL.
8. Click "Next" to review the details you have entered and then click "Finish" to create your Custom SOAP Application.

**Related Topics**[Advanced Security](#)[Applications](#)

# 11 Managing Accounts

---

- Adding or Editing Accounts ..... 47

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

If you select an Account for an FTP, 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 will appear in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, then the stage will appear in green color and if an Account is not configured in a particular stage, then that stage will appear 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 Stages 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.

### Related Topics

[Adding or Editing Accounts](#)

[Managing Operations](#)

[Integrations](#)

[Applications](#)

[Stages Management](#)

## 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 in the screen.

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

Field	Description
<b>Save As</b>	Provide a valid name for the Account. This field is common for all Applications.
<b>Description</b>	Provide a description for the Account. This field is common for all Applications.

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 "Stages Management" page, 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 Stages Management.

See "[Stages Management](#)" on page 27 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.

---

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

---

Click on the following links for information on the Account configuration fields:

### Applications

- ["File Transfer Protocol \(FTP/FTPS\)" on page 48](#)
- ["Custom SOAP Application" on page 49](#)
- ["Amazon Simple Storage Service \(S3\)" on page 52](#)
- ["Amazon Simple Queue Service \(SQS\)" on page 52](#)
- ["Salesforce CRM" on page 54](#)
- ["Salesforce Bulk Data Loader" on page 54](#)
- ["Microsoft Dynamics CRM" on page 53](#)
- ["SuccessFactors HCM" on page 55](#)
- ["ServiceNow Enterprise Service Management" on page 55](#)
- ["StrikeIron Contact Verification" on page 56](#)

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.

### Related Topics

[Advanced Security](#)

[Account Configuration](#)

[Managing Accounts](#)

[Stages Management](#)

[Managing Operations](#)

[Integrations](#)

## Account Configuration

### File Transfer Protocol (FTP/FTPS)

The File Transfer Protocol (FTP/FTPS) Application can download files from or upload files to an FTP-enabled server.



Field	Description
<b>Host</b>	Host name or IP address or the domain name of the FTP server.
<b>Port</b>	FTP port defined on the FTP server.
<b>User</b>	Valid user name on the FTP server.
<b>Password</b>	Password of the FTP user.
<b>SSL Configuration - Select this option for secured FTP connection.</b>	
<b>Secure Data</b>	Select <b>True</b> to secure the data channel. Select <b>False</b> if you do not want to secure the data channel.
<b>Keystore Alias</b>	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.  <b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Keystores.
<b>Key Alias</b>	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.
<b>Truststore Alias</b>	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.  <b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Truststores.

### Custom SOAP Application

The Custom SOAP Application enables you to access third party Web Services hosted in the cloud or on-premise environment.

Field	Description
<b>Protocol</b>	The protocol to use for the port (HTTP or HTTPS). If you select HTTPS, the <b>Keystore Alias</b> and <b>Key Alias</b> fields appear.
<b>Host</b>	Host name or IP address of the server on which the Web Service resides.
<b>Port</b>	An active HTTP or HTTPS type listener port defined on the host server specified in the <b>Host</b> field.
<b>Port Binding</b>	The specific bind address to which to bind this port.
<b>User</b>	User name used to authenticate the consumer at the HTTP or HTTPS transport level on the host server.
<b>Password</b>	The password used to authenticate the consumer on the host server.
<b>Keystore Alias</b>	Alias to the keystore that contains the private key used to connect to the Web Service host securely. This field applies to the HTTPS transport type only. You can also add a new Keystore from this field.
	<b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Keystores.
<b>Key Alias</b>	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. This field applies to the HTTPS transport type only.

**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

<b>User Name</b>	Name to include with the Username Token, if the Web Service's security policy requires one.
<b>Password</b>	The password to include with the UsernameToken (must be plain text).

Field	Description
<b>Keystore / Truststore</b>	
<b>Keystore Alias</b>	<p>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.</p> <p><b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Keystores.</p>
<b>Key Alias</b>	<p>The text identifier for the private key associated with the "Keystore Alias".</p>
<b>Truststore Alias</b>	<p>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> <p><b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Truststores.</p>
<b>Partner Certificate Alias</b>	<p>The file that contains the partner's self-signed certificate. You can also add a new Partner Certificate from this field.</p> <p><b>Note:</b> Users who have the <b>Administer</b> permission under <b>Settings &gt; Access Profiles &gt; Administrative Permissions &gt; Functional Controls &gt; Advanced Security</b> can add, edit, and delete Partner Certificates.</p>
<b>Timestamp</b>	
<b>Timestamp Precision</b>	<p>Whether the timestamp placed in the Timestamp element of the security header of an outbound message is precise to seconds or milliseconds. If the precision is set to milliseconds, the timestamp format yyyy-MM-dd'T'HH:mm:ss:SSS'Z' is used. If the precision is set to seconds, the timestamp format yyyy-MM-dd'T'HH:mm:ss'Z' is used.</p>
<b>Timestamp TTL</b>	<p>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.</p>

Field	Description
<b>Timestamp Max Skew</b>	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.

### Amazon Simple Storage Service (S3)

Field	Description
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Access Key</b>	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.
<b>Secret Key</b>	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.
<b>Region</b>	An area specific value.
<b>Signing Algorithm</b>	Explicitly specify the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.

### Amazon Simple Queue Service (SQS)

Field	Description
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.

Field	Description
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Access Key</b>	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.
<b>Secret Key</b>	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.
<b>Region</b>	An area specific value.
<b>Signing Algorithm</b>	Explicitly specify the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.

#### Microsoft Dynamics CRM

Field	Description
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>User Name</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.

---

## Salesforce CRM

<b>Field</b>	<b>Description</b>
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Username</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.

## Salesforce Bulk Data Loader

<b>Field</b>	<b>Description</b>
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Username</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.

**SuccessFactors HCM**

<b>Field</b>	<b>Description</b>
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Username</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.
<b>Company ID</b>	The companyID that SuccessFactors provided, when your company registered with them.

**ServiceNow Enterprise Service Management**

<b>Field</b>	<b>Description</b>
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Username</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.

## Strikelron Contact Verification

<b>Field</b>	<b>Description</b>
<b>Server URL</b>	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the connection configuration.
<b>Connection Timeout</b>	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. Default: 30000 ms.
<b>Username</b>	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.
<b>Password</b>	Provide a password for the user name to initiate communication with the SaaS provider.

### Related Topics

[Advanced Security](#)

[Adding or Editing Accounts](#)

[Adding Custom SOAP Applications](#)



# 12 Managing Operations

---

■ Adding or Editing Operations .....	58
--------------------------------------	----

Integration Cloud provides pre-configured applications. The Applications contain SaaS provider-specific information that enables you to connect to a particular SaaS provider. Further, each Application uses an Account to connect to the provider's back end and perform Operations.

**Note:** Users who have the required permissions under **Settings > 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.

---

### 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 add an Operation, click Edit to update an existing Operation, or click Delete to delete an existing Operation.

### Related Topics

[Adding or Editing Operations](#)

[Stages Management](#)

[Managing Accounts](#)

[Integrations](#)

## 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, click Add to add a custom Operation or click Edit to update an existing custom Operation. You can edit the Operation or the Business Parameter.
4. On the Select your <...> account screen, complete the following fields. Required fields are marked with an asterisk in the screen.

Field	Description
<b>Name</b>	Provide a name for the custom Operation.
<b>Description</b>	Provide a description for the custom Operation.
<b>Type</b>	<p>Select the type of Operation you want to perform for the selected application.</p> <p>A <b>Trigger</b> 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 <b>Action</b> is an Operation which uploads data to an application and which can be used in the Target section while creating an Integration.</p> <p><b>Note:</b> You will not be allowed to change these options after the Operation is created.</p>
<b>Select Account</b>	Select an Account created for the Application from the drop down list. Only active or enabled Accounts are listed in the drop down list.
<b>Select functional area</b>	Select the Application service from the drop down list. This option is applicable only for certain applications.

5. Click Next.
6. Select the Operation to be performed and then click Next.
7. Select the business object to be processed by the Operation and then click Next. Business objects will appear only for certain applications and Operations.
8. Select the data fields to be added if any, and then click Next. Data fields will appear only for certain applications and Operations.
9. Verify the details in the Confirm Operation page and then 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. Currently, Business Parameters Configuration screen appears only for Upsert Operations for Salesforce.

**Related Topics**[Managing Operations](#)[Managing Accounts](#)[Stages Management](#)[Integrations](#)

---


# 13 Integrations

---

■ Creating Point-to-Point Integrations .....	62
■ Creating Orchestrated Integrations .....	65
■ Pipeline and Signatures .....	74
■ Built-In Services .....	78
■ Document Types .....	188
■ Reference Data .....	190
■ Integration Details .....	193
■ Execution Results .....	196

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. 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.

You can also see when the Integration was created and who created it. You can delete, edit, copy, or create a new Integration from this screen.

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 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**.

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 audit trail of all the executions that happened in a stage for an Integration, click **Execution Results**.

### Related Topics

[Creating Orchestrated Integrations](#)

[Built-In Services](#)

[Integration Details](#)

[Creating Point-to-Point Integrations](#)

[Execution Results](#)

[Managing Accounts](#)

[Managing Operations](#)

[Stages Management](#)

## Creating 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 ["Creating Orchestrated Integrations" on page 65](#) 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 in 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.

Click the drop down arrow beside the *fx* icon and select **Add** to view the **Transform Data - Select Transformer** 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. The **Transform Data - Select Transformer** screen also allows you to look up and use data from another source Operation to transform the data. You can click the drop down arrow beside the *fx* icon and select **Edit** or **Delete** 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.

### Related Topics

[Creating Orchestrated Integrations](#)

[Built-In Services](#)

[Integrations](#)

[Execution Results](#)

[Managing Accounts](#)

[Managing Operations](#)

[Stages Management](#)



## Creating 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

Block category	Icons	Description
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

Block category	Icons	Description
		<p>can sequence services and manage the flow of data among them.</p>
		<p><b>Note:</b> For information on the different services, see <a href="#">Built-In Services</a>.</p> <p>The <b>Reference Data</b> block appears only if a Reference Data service is available at <b>Develop &gt; Reference Data &gt; Reference Data</b> page. See <a href="#">Reference Data</a> for more information.</p>
Integrations		<p>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>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>
Control Flow		<p>Conditional expressions, looping structures, and transform pipeline.</p> <p>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>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 time, the Loop executes one pass of the loop for each member in the specified array. For example, if</p>


Block category	Icons	Description
		<p>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>A Loop takes as input an array field that is in the pipeline. It loops over the members of an input array, executing its child steps each time through the loop. For example, if you have a Integration that takes a string as input and a string list in the pipeline, use Loops to invoke the Integration one time for each string in the string list. You identify a single array field to use as input when you set the properties for the Loop. You can also designate a single field for the output. Loop collects an output value each time it runs through the loop and creates an output array that contains the collected output values.</p> <p>Use the <i>Transform Pipeline</i> block to make pipeline modifications. See <a href="#">Pipeline and Signatures</a> 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>

Block category	Icons	Description
		You can also type a text value, select a field on which to build an expression, or select a block with no inputs.





**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:** 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. You can also click **Test** to test the Integration execution in real time and view the execution results in the **Test Results** pane.


- 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.	Add or edit the description of a block.  Comments for all blocks.
	Application.	Map the input of the operation from the Pipeline and also map the output of the operation into the pipeline.
	Transform Pipeline.	Make pipeline modifications.  Edit data mapping, Set Transformer, Set a value, Remove a field, Restore a field, Add a field, Clear all mappings, and so on.
	Application.	Select an Account and an Operation for the Application.

Icons	Applicable for...	Action/Description
	Orchestrated Integrations.	Click to view or modify an Orchestrated Integration after it is moved to the workspace. The Orchestrated Integration will open up for editing in a new tab.
	Application/Service.	The block is not configured. Select an Account and an Operation for the Application or select a Service.
	Start/End Integration block	<p>Click to define the input/output signature of an Integration.</p> <p>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>

Icons	Applicable for...	Action/Description
		<p>You can select a Document Type from the <b>Document Reference</b> drop down list. To create a Document Type, from the Integration Cloud navigation bar, select <b>Develop &gt; Document Types &gt; Add New Document Type</b>.</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>See <a href="#">Document Types</a> for more information.</p>
	Orchestrated Integrations.	An Orchestrated Integration has been modified but not saved.

8. 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.
9. To map the Pipeline Input to the Input Signature, click .
10. Map the Output Signature to the Pipeline Output in the pipeline data window and then click Finish.

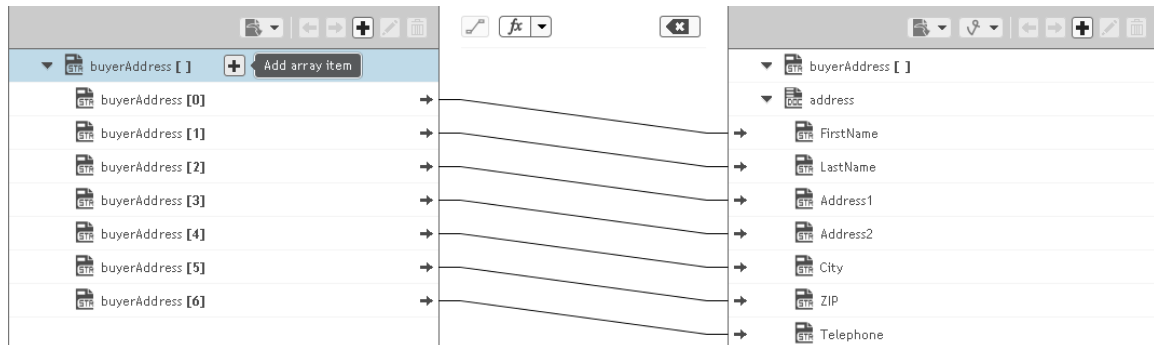
### Indexed Mapping

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, then at run time, the String List length is increased from 3 to 5.

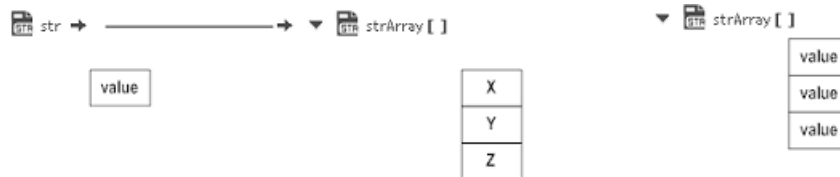
When you link a Document or Document List variable to another Document or Document List variable, the structure of the source variable determines the structure of the target variable.

#### *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.

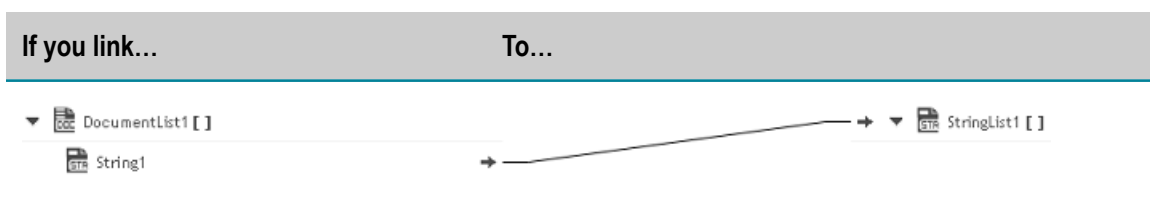


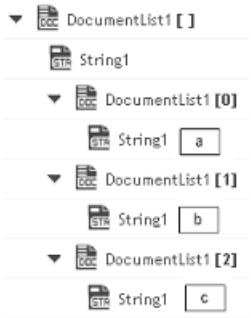



If you link...	To...	Then...
An array variable	An array variable that has a defined length	The length of the source array variable <i>must</i> equal the length of the target array variable. If the lengths do not match, the link will not occur. If the lengths are equal, the elements in the target array 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:



Where the value of DocumentList1 is...	Then the value of StringList1 is...
	

- 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.
- Click “Save” to save your Integration or click “Save All” to save all modified Integrations.

The new Integration will appear in the **Integrations** page. You can click on the Integration link in the “Integrations” page to view the Integration details.

### Related Topics

[Creating Point-to-Point Integrations](#)

[Reference Data](#)

[Document Types](#)

[Integration Details](#)

[Built-In Services](#)

[Pipeline and Signatures](#)

## 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/End block of an Orchestrated Integration, you can click the  icon and 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 \*  
stringArray

Type \*  
String [ ]

XML Namespace  
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.

### Related Topics

[Creating Orchestrated Integrations](#)

[Creating Point-to-Point Integrations](#)

## 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.

### Related Topics

[Creating Orchestrated Integrations](#)

[Reference Data](#)

## Date

The services under Date can be used 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
M	month in year	Text or Number	July or Jul or 07

<b>Symbol</b>	<b>Meaning</b>	<b>Presentation</b>	<b>Example</b>
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).

<b>ID</b>	<b>Raw Offset</b>	<b>Name</b>
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
ECT	+1	European Central Time
CAT	+2	Central Africa Time



ID	Raw Offset	Name
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
<a href="#">calculateDateDifference</a>	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**

<i>startDate</i>	<b>String</b> Starting date and time.
<i>endDate</i>	<b>String</b> Ending date and time.
<i>startDatePattern</i>	<b>String</b> Format in which the <i>startDate</i> parameter is to be specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the “Pattern String Symbols” section.
<i>endDatePattern</i>	<b>String</b> Format in which the <i>endDate</i> 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>	<b>String</b> The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of seconds.
<i>dateDifferenceMinutes</i>	<b>String</b> The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of minutes.
<i>dateDifferenceHours</i>	<b>String</b> The difference between the <i>startingDateTime</i> and <i>endingDateTime</i> , truncated to the nearest whole number of hours.
<i>dateDifferenceDays</i>	<b>String</b> 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>	<b>String</b> Starting date to compare against <i>endDate</i> .
<i>endDate</i>	<b>String</b> Ending date to compare against <i>startDate</i> .
<i>startDatePattern</i>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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>	<p><b>String</b> Date/time that you want to convert.</p> <div data-bbox="483 594 1367 762" style="background-color: #f0f0f0; padding: 10px;"><p><b>Important:</b> 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.</p></div>
<i>currentPattern</i>	<p><b>String</b> Pattern string that describes the format of <i>inString</i>. For pattern-string notation, see the “Pattern String Symbols” section.</p>
<i>newPattern</i>	<p><b>String</b> Pattern string that describes the format in which you want <i>inString</i> returned. For pattern-string syntax, see the “Pattern String Symbols” section.</p>
<i>locale</i>	<p><b>String</b> 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>.</p>
<i>lenient</i>	<p><b>String</b> 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:</p> <ul style="list-style-type: none"><li>■ <code>true</code> to perform a lenient check. This is the default. <p>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.</p></li><li>■ <code>false</code> to perform a strict check. <p>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.</p></li></ul>



---

## 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>	<b>String</b> Starting date and time.
<i>startDatePattern</i>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> Optional. Number of years to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMonths</i>	<b>String</b> Optional. Number of months to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addDays</i>	<b>String</b> Optional. Number of days to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addHours</i>	<b>String</b> Optional. Number of hours to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMinutes</i>	<b>String</b> Optional. Number of minutes to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.

---

<i>addSeconds</i>	<b>String</b> Optional. Number of seconds to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>addMilliseconds</i>	<b>String</b> Optional. Number of milliseconds to add to <i>startDate</i> . The value must be an integer between -2147483648 and 2147483647.
<i>timezone</i>	<b>String</b> 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>	<b>String</b> 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>	<b>String</b> 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.

### ***findDocuments***

Searches a set of documents for entries matching a set of criteria.

---

## Input Parameters

---

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

---

## Output Parameters

---

- result*      **Document List** List of documents that match the retrieve criteria.  
*documents*

---

## *insertDocument*

Inserts a new document in a set of documents at a specified position.

---

### Input Parameters

---

<i>documents</i>	<b>Document List</b> Set of documents in which a new document is to be inserted.
<i>insertDocument</i>	<b>Document</b> The new document to be inserted to the set of documents specified in the <i>documents</i> parameter.
<i>index</i>	<b>String</b> 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>	<b>Document List</b> Document list after inserting the new document.
------------------	--

---

## ***deleteDocuments***

Deletes the specified documents from a set of documents.

### Input Parameters

---

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

### Output Parameters

---

<i>documents</i>	<b>Document List</b> List of documents whose indices do <i>not</i> match the values in <i>indices</i> parameter.
<i>deletedDocuments</i>	<b>Document List</b> 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

---

<i>documentList</i>	<p><b>Document List</b> Set of documents that you want to transform into a single document.</p> <p><b>Note:</b> If the <i>documentList</i> parameter contains a single document instead of a Document List, the <code>documentListToDocument</code> service does nothing.</p>
<i>name</i>	<p><b>String</b> Name of the element in the <i>documentList</i> parameter whose value provides the name of each key in the resulting document.</p> <p><b>Important:</b> The data type of the element that you specify in the <i>name</i> parameter must be String.</p>
<i>value</i>	<p><b>String</b> Name of the element in the <i>documentList</i> parameter whose values will be assigned to the keys specified in <i>name</i>. This element can be of any data type.</p>

### Output Parameters

---

<i>document</i>	<p><b>Document</b> Document containing the key/value pairs generated from the <i>documentList</i> parameter.</p>
-----------------	--

## 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:

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

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

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

Would be converted to a document containing these three key:

<b>Key</b>	<b>Value</b>
<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**

---

*document*      **Document** Document to transform.



<i>name</i>	<b>String</b> 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>	<b>String</b> 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>	<b>Document List</b> 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:

<b>Key</b>	<b>Value</b>
<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:

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

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

Key	Value
<i>pValue</i>	2500

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

## 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
<a href="#">appendToDocumentList</a>	Adds documents to a document list.
<a href="#">appendToStringList</a>	Adds Strings to a String list.
<a href="#">sizeOfList</a>	Returns the number of elements in a list.
<a href="#">stringListToDocumentList</a>	Converts a String list to a document list.

## ***appendToDocumentList***

Adds documents to a document list.

### Input Parameters

<i>toList</i>	<b>Document List</b> 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>	<b>Document List</b> Optional. Documents you want to append to the end of <i>toList</i> .
<i>fromItem</i>	<b>Document</b> 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>	<b>Document List</b> 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>	<b>String List</b> 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>	<b>String List</b> Optional. List of Strings to add to <i>toList</i> . Strings are added after the entries of <i>toList</i> .
<i>fromItem</i>	<b>String</b> 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 <i>fromItem</i> after the Strings specified in <i>fromList</i> .

---

## 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>	<b>String List</b> Optional. List of Strings (a <code>String[]</code> ) 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>	<b>String</b> Optional. Key name to use in the generated document list.

### **Output Parameters**

<i>toList</i>	<b>Document List</b> 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:

▼  <i>toList</i> [ ]	
▼  <i>toList</i> [0]	
 myKey	a
▼  <i>toList</i> [1]	
 myKey	b
▼  <i>toList</i> [2]	
 myKey	c

## Math

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 services are available in this folder:

Service	Description
<a href="#">absoluteValue</a>	Returns the absolute value of the input number.
<a href="#">addFloatList</a>	Adds a list of floating point numbers (represented in a string list) and returns the sum.
<a href="#">addFloats</a>	Adds one floating point number (represented as a String) to another and returns the sum.
<a href="#">addIntList</a>	Adds a list of integers (represented in a String list) and returns the sum.
<a href="#">addInts</a>	Adds one integer (represented as a String) to another and returns the sum.
<a href="#">divideFloats</a>	Divides one floating point number (represented as a String) by another ( $num1/num2$ ) and returns the quotient.
<a href="#">divideInts</a>	Divides one integer (represented as a String) by another ( $num1/num2$ ) and returns the quotient.
<a href="#">max</a>	Returns the largest number from a list of numbers.
<a href="#">multiplyFloatList</a>	Multiplies a list of floating point numbers (represented in a String list) and returns the product.
<a href="#">multiplyFloats</a>	Multiplies one floating point number (represented as String) by another and returns the product.
<a href="#">multiplyIntList</a>	Multiplies a list of integers (represented in a String list) and returns the product.

Service	Description
<code>multiplyInts</code>	Multiplies one integer (represented as a String) by another and returns the product.
<code>randomDouble</code>	Returns the next pseudorandom, uniformly distributed double between 0.0 and 1.0.
<code>roundNumber</code>	Returns a rounded number.
<code>subtractFloats</code>	Subtracts one floating point number (represented as a String) from another and returns the difference.
<code>subtractInts</code>	Subtracts one integer (represented as a String) from another and returns the difference.

## Summary of Math services

### ***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:

<b>Value</b>	<b>Description</b>
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

---

*num1* **String** Number to add.

*num2* **String** Number to add.

*precision* **String** Optional. Number of decimal places to which the sum will be rounded. The default value is null.



---

## Output Parameters

---

*value*      **String** Sum of the numbers in *num1* and *num2* . If a sum cannot be produced, *value* contains one of the following:

<b>Value</b>	<b>Description</b>
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:

Value	Description
Infinity	The computation produces a positive value that overflows the representable range of a float type.
-Infinity	The computation produces a negative value that overflows the representable range of a float type.
0.0	The computation produces a value that underflows the representable range of a float type (for example, subtracting a number from infinity).
NaN	The computation produces a value that cannot be represented as a number (for example, the result of an illegal operation such as multiplying zero by zero or any operation that uses NaN as input, such as $10.0 - \text{NaN} = \text{NaN}$ ).

### 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.

---

## 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

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

Service	Description
<a href="#">base64Decode</a>	Decodes a Base-64 encoded string into a sequence of bytes.
<a href="#">base64Encode</a>	Converts a sequence of bytes into a Base64-encoded String.
<a href="#">bytesToString</a>	Converts a sequence of bytes to a String.

Service	Description
<a href="#">concat</a>	Concatenates two strings.
<a href="#">indexOf</a>	Returns the index of the first occurrence of a sequence of characters in a string.
<a href="#">length</a>	Returns the length of a string.
<a href="#">lookupDictionary</a>	Looks up a given key in a hash table and returns the string to which that key is mapped.
<a href="#">makeString</a>	Builds a single string by concatenating the elements of a String List.
<a href="#">messageFormat</a>	Formats an array of strings into a given message pattern.
<a href="#">numericFormat</a>	Formats a number into a given numeric pattern.
<a href="#">padLeft</a>	Pads a string to a specified length by adding pad characters to the beginning of the string.
<a href="#">padRight</a>	Pads a string to a specified length by adding pad characters to the end of the string.
<a href="#">replace</a>	Replaces all occurrences of a specified substring with a substitute string.
<a href="#">stringToBytes</a>	Converts a string to a byte array.
<a href="#">substring</a>	Returns a substring of a given string.
<a href="#">tokenize</a>	Tokenizes a string using specified delimiter characters and generates a String List from the resulting tokens.
<a href="#">toLowerCase</a>	Converts all characters in a given string to lowercase.
<a href="#">toUpperCase</a>	Converts all characters in a given string to uppercase.

Service	Description
<a href="#">trim</a>	Trims leading and trailing white space from a given string.
<a href="#">URLDecode</a>	Decodes a URL-encoded string.
<a href="#">URLEncode</a>	URL-encodes a string.
<a href="#">fuzzyMatch</a>	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.

## Summary of String services

### ***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

---

<i>bytes</i>	<b>byte[ ]</b> Sequence of bytes to encode into a Base64-encoded String.
<i>useNewLine</i>	<p><b>String</b> Optional. Flag indicating whether to retain or remove the line breaks. Set to:</p> <ul style="list-style-type: none"> <li>■ <code>true</code> to retain the line breaks. This is the default.</li> <li>■ <code>false</code> to remove the line breaks.</li> </ul>
<i>encoding</i>	<b>String</b> Optional. Specifies the encoding method. Default value is <code>ASCII</code> .

---

## Output Parameters

---

<i>value</i>	<b>String</b> 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>	<b>byte[ ]</b> Sequence of bytes to convert to a String.
<i>encoding</i>	<p><b>String</b> Optional. Name of a registered, IANA character set (for example, <code>ISO-8859-1</code>). If you specify an unsupported encoding, the system throws an exception.</p> <p>To use the default encoding, set <i>encoding</i> to <code>autoDetect</code>.</p>
<i>ignoreBOMChars</i>	<b>String</b> Optional. Flag indicating whether or not the byte order mark (BOM) characters in the input sequence of bytes are removed before converting the byte array to string. Set to:

- `true` to remove the byte order mark (BOM) characters before converting the input sequence of bytes to string, if the byte array contains BOM characters.
- `false` to include the byte order mark (BOM) characters while converting the input sequence of bytes to string. The default is `false`.

---

### Output Parameters

---

*string*                      **String** String representation of the contents of *bytes* .

---

## ***concat***

Concatenates two strings.

---

### Input Parameters

---

*inString1*                      **String** String to which you want to concatenate another string.

*inString2*                      **String** String to concatenate to *inString1* .

---

### Output Parameters

---

*value*                              **String** Result of concatenating *inString1* with *inString2* (*inString1* + *inString2*).

---

## ***indexOf***

Returns the index of the first occurrence of a sequence of characters in a string.

---

### Input Parameters

---

*inString*                              **String** String in which you want to locate a sequence of characters.

*subString*                              **String** Sequence of characters to locate.

---

*fromIndex*      **String** Optional. Index of *inString* from which to start the search. If no value is specified, this parameter contains 0 to indicate the beginning of the string.

#### Output Parameters

---

*value*      **String** Index of the first occurrence of *subString* in *inString* . If no occurrence is found, this parameter contains -1.

---

## ***length***

Returns the length of a string.

#### Input Parameters

---

*inString*      **String** String whose length you want to discover.

#### Output Parameters

---

*value*      **String** The number of characters in *inString* .

---

## ***lookupDictionary***

Looks up a given key in a hash table and returns the string to which that key is mapped.

#### Input Parameters

---

*hashtable*      **java.util.Hashtable** Hash table that uses String objects for keys and values.

*key*      **String** Key in *hashtable* whose value you want to retrieve.

**Note:** The key is case sensitive.

---

## Output Parameters

---

*value* **String** Value of the string to which *key* is mapped. If the requested key in *hashtable* is null or if *key* is not mapped to any value in *hashtable*, the service returns null.

---

## **makeString**

Builds a single string by concatenating the elements of a String List.

---

### Input Parameters

---

*elementList* **String List** Strings to concatenate.

*separator* **String** String to insert between each non-null element in *elementList*.

---

### Output Parameters

---

*value* **String** Result from concatenating the strings in *elementList*. Strings are separated by the characters specified in *separator*.

---

## **messageFormat**

Formats an array of strings into a given message pattern.

---

### Input Parameters

---

*pattern* **String** Message that includes "placeholders" where elements from *argumentList* are to be inserted. The message can contain any sequence of characters. Use the `{n}` placeholder to insert elements from *argumentList*, where *n* is the index of the element that you want to insert. For example, the following pattern string inserts elements 0 and 1 into the message:

```
Test results: {0} items passed, {1} items failed.
```

**Note:** Do not use any characters except digits for *n*.



*argumentList* **String List** Optional. List of strings to use to populate *pattern*. If *argumentList* is not supplied, the service will not replace placeholders in *pattern* with actual values.

## Output Parameters

---

*value* **String** Result from substituting *argumentList* into *pattern*. If *pattern* is empty or null, this parameter is null.

## ***numericFormat***

Formats a number into a given numeric pattern.

## Input Parameters

---

*num* **String** The number to format.

*pattern* **String** A pattern string that describes the way in which *num* is to be formatted:

<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:

Pattern	Description
#,###	Use commas to separate into groups of three digits. The pound sign denotes a digit and the comma is a placeholder for the grouping separator.
#,####	Use commas to separate into groups of four digits.
\$.00	Show digits before the decimal point as needed and exactly two digits after the decimal point. Prefix with the \$ character.
'#'#.0	Show digits before the decimal point as needed and exactly one digit after the decimal point. Prefix with the # character. The first character in a pattern is the dollar sign (\$). The pound sign denotes a digit and the period is a placeholder for decimal separator.

## Output Parameters

*value* **String***num* formatted according to *pattern*. If *pattern* is an empty (not null) string, the default pattern of comma separators is used and the number of digits after the decimal point remains unchanged.

## ***padLeft***

Pads a string to a specified length by adding pad characters to the beginning of the string.

---

### Input Parameters

---

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

### Output Parameters

---

<i>value</i>	<b>String</b> Contents of <i>inString</i> preceded 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 beginning of *padString* is aligned with the beginning of the resulting string. For example, suppose *inString* equals *shipped* and *padString* equals *x9y*.

<b>If <i>length</i> equals...</b>	<b>Then <i>value</i> will contain...</b>
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>	<b>String</b> String that you want to pad.
-----------------	--

---

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

### Output Parameters

---

<i>value</i>	<b>String</b> 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`.

<b>If <i>length</i> equals...</b>	<b>Then <i>value</i> will contain...</b>
7	<code>shipped</code>
10	<code>shippedx9y</code>
12	<code>shippedx9y9y</code>

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>	<b>String</b> String containing the substring to replace.
<i>searchString</i>	<b>String</b> Substring to replace within <i>inString</i> .
<i>replaceString</i>	<b>String</b> 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> .

*useRegex*

**String** Optional. Flag indicating whether *searchString* is a regular expression. When regular expressions are used to specify a search string, *replaceString* may also contain interpolation fields (for example, "\$1") that match parenthetical subexpressions in *searchString*.

Set to:

- `true` to indicate that *searchString* is a regular expression.
- `false` to indicate that *searchString* is not a regular expression. This is the default.

### Output Parameters

---

*value*

**String** Contents of *inString* with replacements made.

## ***stringToBytes***

---

Converts a string to a byte array.

### Input Parameters

---

*string*

**String** String to convert to a `byte[]`.

*encoding*

**String** Optional. Name of a registered, IANA character set that specifies the encoding to use when converting the String to an array of bytes (for example: `ISO-8859-1`).

To use the default encoding, set this value to `autoDetect`. If you specify an unsupported encoding, an exception will be thrown.

### Output Parameters

---

*bytes*

**byte[]** Contents of *string* represented as a `byte[]`.

## ***substring***

---

Returns a substring of a given string.

---

### Input Parameters

---

<i>inString</i>	<b>String</b> String from which to extract a substring.
<i>beginIndex</i>	<b>String</b> Beginning index of the substring to extract (inclusive).
<i>endIndex</i>	<b>String</b> Ending index of the substring to extract (exclusive). If this parameter is null or empty, the substring will extend to the end of <i>inString</i> .

### Output Parameters

---

<i>value</i>	<b>String</b> Substring from <i>beginIndex</i> and extending to the character at <i>endIndex</i> - 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

---

<i>inString</i>	<b>String</b> String you want to tokenize, that is, break into delimited chunks.
<i>delim</i>	<b>String</b> Delimiter characters. If null or empty, the service uses the default delimiters <code>\t\n\r</code> , where t, n, and r represent the white space characters tab, new line, and carriage return.

### Output Parameters

---

<i>valueList</i>	<b>String List</b> Strings containing the tokens extracted from <i>inString</i> .
------------------	---

---

## **toLower**

Converts all characters in a given string to lowercase.

---

### Input Parameters

---

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

### Output Parameters

---

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

---

## ***toUpper***

Converts all characters in a given string to uppercase.

### Input Parameters

---

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

### Output Parameters

---

<i>value</i>	<b>String</b> 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**

---

*inString*                      **String** String to trim.

### **Output Parameters**

---

*value*                              **String** Contents of *inString* with white space trimmed from both ends.

---

## ***URLDecode***

Decodes a URL-encoded string.

### **Input Parameters**

---

*inString*                      **String** URL-encoded string to decode.

### **Output Parameters**

---

*value*                              **String** Result from decoding *inString*. If *inString* contains plus (+) signs, they will appear in *value* as spaces. If *inString* contains *%hex* encoded characters, they will appear in *value* as the appropriate native character.

---

## ***URLEncode***

URL-encodes a string.

Encodes characters the same way that data posted from a WWW form is encoded, that is, the `application/x-www-form-urlencoded` MIME type.

### **Input Parameters**

---

*inString*                      **String** String to URL-encode.



---

## Output Parameters

*value* **String** Result from URL-encoding *inString*. If *inString* contains non-alphanumeric characters (except [-\_.\*@]), they will appear in *value* as their URL-encoded equivalents (% followed by a two-digit hex code). If *inString* contains spaces, they will appear in *value* as plus (+) signs.

---

## fuzzyMatch

A given string is not exactly matched against a set of strings. If the match is above *similarityThreshold*, it returns the *matchedValue*. If more than one string has not exactly matched, then the first matched string is returned.

---

## Input Parameters

*inString* **String (Required)** Text to be matched. Text should not be empty or null.

*matchData* **String [ ] (Required)** Array of strings, which are used for matching. If the string array value is either empty or null, it is not used for matching.

*similarityThreshold* **String (Optional)** If the inexact match score is above the given threshold, then service output contains the *matchedValue* parameter. Default value is 0.65. Valid values should be between 0.0 and 1.0. Value 0.0 represents no match and value 1.0 represents an exact match.

*algorithm* **String (Optional)** The algorithm used for an inexact match. Default value is Levenshtein. Supported algorithms are Levenshtein and JaroWinkler.

---

## Output Parameters

*matchedValue* **String (Optional)** If the inexact match is above *similarityThreshold*, then the returned value contains the matched string.

*similarity* **String (Optional)** If the inexact match is above *similarityThreshold*, then it contains a similarity score. It provides the measure of how close the match is. The

returned value can be between 0.0 and 1.0. Value 0.0 represents no match and value 1.0 represents an exact match.

### Usage Notes

For more information about the algorithms, see [http://en.wikipedia.org/wiki/Levenshtein\\_distance](http://en.wikipedia.org/wiki/Levenshtein_distance) and [http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler\\_distance](http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance)

## Flow

You can use the Flow services to perform utility-type tasks. The following services are available:

Service	Description
<a href="#">getLastError</a>	Obtains detailed information about the last error that was trapped within an Integration.

## Summary of Flow services

### *getLastError*

Obtains detailed information about the last error that was trapped within an Integration.

#### Input Parameters

None

#### Output Parameters

*lastError*

**Document.** Information about the last error, which contains details of the time, error, user, block, and call stack information.

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

<b>error</b>	<b>String.</b> Optional. Error message of the exception.
<b>localizedError</b>	<b>String.</b> Optional. Error message in the language that corresponds to the server locale.
<b>user</b>	<b>String.</b> User who executed the Integration.
<b>block</b>	<b>Document.</b> Contains the following fields:

<u>Key</u>	<u>Description</u>
name	<b>String.</b> Integration, Operation, or Service name.
type	<b>String.</b> Application, Integration, or Service.
details	<b>String.</b> Optional. Account and Application name if the Block Type is "Application".
<b>callStack</b>	<b>Document List.</b> 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	<b>String.</b> Integration, Operation or Service name.
type	<b>String.</b> Application, Integration, or Service.
details	<b>String.</b> 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.

## Hashtable

The following Hashtable services are available:

Service	Description
<code>containsKey</code>	Checks for the existence of a hashtable element.
<code>createHashtable</code>	Creates a hashtable object.
<code>get</code>	Gets the value for a specified key in the hashtable.
<code>listKeys</code>	Lists all the keys stored in the hashtable.
<code>put</code>	Adds a key/value pair in the hashtable.
<code>remove</code>	Removes a key/value pair from the hashtable.
<code>size</code>	Gets the number of elements in the hashtable.

## Summary of Hashtable services

### ***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

The following Flat File services are available:

Service	Description
<a href="#">delimitedDataBytesToDocument</a>	Converts delimited data bytes (byte array) to a document.
<a href="#">delimitedDataStreamToDocument</a>	Converts delimited data stream to a document.

Service	Description
<a href="#">delimitedDataStringToDocument</a>	Converts delimited data string to a document.
<a href="#">documentToDelimitedDataBytes</a>	Converts a document to delimited data bytes (byte array object).
<a href="#">documentToDelimitedDataStream</a>	Converts a document to a delimited data stream.
<a href="#">documentToDelimitedDataString</a>	Converts a document to a delimited data string.

## Summary of Flat File services

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

---

<i>useHeaderRowForFieldNames</i>	<b>String</b> 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"> <li>■ <i>true</i> . The delimited data of first line will be used as the property name in the output document. This is the default.</li> <li>■ <i>false</i> . column1, column2...columnN will be used as the property name in the output document.</li> </ul>
<i>encoding</i>	<b>String</b> Optional. The encoding to use while parsing the delimited data.

### Output Parameters

---

<i>document</i>	<b>Document.</b> 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

<i>delimited DataStream</i>	<b>java.io.InputStream</b> . Delimited data in an input stream to convert to a document.
<i>fieldQualifier</i>	<b>String</b> Optional. The delimiter to use for separating entries in <i>delimitedDataStream</i> . Default is comma (,).
<i>textQualifier</i>	<b>String</b> Optional. The character to use for quoted elements. Default is double quote (").

---

<i>useHeaderRowForFieldNames</i>	<b>String</b> 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"> <li>■ <i>true</i> . The delimited data of first line will be used as the property name in the output document. This is the default.</li> <li>■ <i>false</i> . column1, column2...columnN will be used as the property name in the output document.</li> </ul>
<i>encoding</i>	<b>String</b> Optional. The encoding to use while parsing the delimited data.

### Output Parameters

---

<i>document</i>	<b>Document</b> . 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>	<b>String.</b> Delimited string to convert to a document.
<i>fieldQualifier</i>	<b>String</b> Optional. The delimiter to use for separating entries in <i>delimitedDataString</i> . Default is comma (,).
<i>textQualifier</i>	<b>String</b> 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>	<b>Document.</b> 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>	<b>String</b> Optional. The delimiter to use for separating entries in <i>delimitedDataBytes</i> . Default is comma (,).
<i>textQualifier</i>	<b>String</b> Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p><b>String</b> 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"> <li>■ <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> .</li> <li>■ <i>false</i> . <code>column1, column2...columnN</code> will be used as the first row in the output <i>delimitedDataBytes</i> .</li> </ul>
<i>encoding</i>	<b>String</b> Optional. The encoding to use while parsing the delimited data.

## Output Parameters

---















<i>delimitedDataBytes</i>	<b>Object.</b> 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>	<b>Document.</b> Document to be converted to delimited data stream. This document contains a document array rows[] corresponding to the delimited data.
<i>fieldQualifier</i>	<b>String</b> Optional. The delimiter to use for separating entries in <i>delimitedDataStream</i> . Default is comma (,).
<i>textQualifier</i>	<b>String</b> Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p><b>String</b> 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"> <li>■ <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> .</li> <li>■ <i>false</i> . column1, column2...columnN will be used as the first row in the output <i>delimitedDataStream</i> .</li> </ul>
<i>encoding</i>	<b>String</b> Optional. The encoding to use while parsing the delimited data.

### Output Parameters















---

<i>delimitedDataStream</i>	<b>java.io.InputStream.</b> 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>	<b>Document.</b> Document to be converted to delimited data string. This document contains document array rows[] corresponding to the delimited data.
<i>fieldQualifier</i>	<b>String</b> Optional. The delimiter to use for separating entries in <i>delimitedDataString</i> . Default is comma (,).
<i>textQualifier</i>	<b>String</b> Optional. The character to use for quoted elements. Default is double quote (").
<i>useFieldNamesForHeaderRow</i>	<p><b>String</b> 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"> <li>■ <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> .</li> <li>■ <i>false</i> . column1, column2...columnN will be used as the first row in the output <i>delimitedDataString</i> .</li> </ul>
<i>encoding</i>	<b>String</b> Optional. The encoding to use while parsing the delimited data.

## Output Parameters

---

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

## JSON

The following JSON services are available:

Service	Description
<a href="#">documentToJSONBytes</a>	Converts a document to JSON bytes (byte array).
<a href="#">documentToJSONStream</a>	Converts a document to a JSON stream.
<a href="#">documentToJSONString</a>	Converts a document to a JSON string.

---

Service	Description
<a href="#">jsonBytesToDocument</a>	Converts JSON content in bytes (byte array) to a document.
<a href="#">jsonStreamToDocument</a>	Converts content from the JSON content stream to a document.
<a href="#">jsonStringToDocument</a>	Converts content from the JSON string to a document.

---

## Summary of JSON services

---

### ***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>	<b>Document.</b> The document to be converted to a JSON string.
<i>prettyPrint</i>	<b>String.</b> 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"><li>■ <i>true</i> to format the <i>jsonString</i> output field for human readability</li><li>■ <i>false</i> to leave the <i>jsonString</i> output field in its unformed state</li></ul> The service will not add any additional carriage returns or indentation to the JSON content.

### **Output Parameters**

---

<i>jsonString</i>	<b>Object.</b> 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>	<b>java.lang.Byte[].</b> JSON content in bytes (byte array) to convert to a document.
<i>decodeRealAsDouble</i>	<b>String.</b> 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"><li>■ <i>true</i> to convert real numbers to Double Java wrapper types</li><li>■ <i>false</i> to convert real numbers to Float Java wrapper types</li></ul> Default value is <i>true</i> .
<i>decodeIntegerAsLong</i>	<b>String.</b> 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

The following XML services are available:

Service	Description
<a href="#">documentToXMLBytes</a>	Converts a document to xml content bytes, as a byte array object.
<a href="#">documentToXMLStream</a>	Converts a document to xml stream, as a <code>java.io.InputStream</code> object.
<a href="#">documentToXMLString</a>	Converts a document to xml content string.

---

Service	Description
<a href="#">xmlBytesToDocument</a>	Converts XML content bytes (byte array) to a document.
<a href="#">xmlStreamToDocument</a>	Converts an XML content stream to a document.
<a href="#">xmlStringToDocument</a>	Converts an XML string to a document.

---

## Summary of XML services

---

### ***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

*attrPrefix* **String.** Optional. Prefix that designates keys containing attributes. The default prefix is "@". For example, if you set *attrPrefix* to `ATT_` and document contains a key `ATT_currency` whose value is "dollars",

 tx	
 ATT_currency	dollars
 acct	cash
 amt	120.00

then this service will convert the `ATT_currency` key to the attribute, `currency=dollars`, in the `<tx>` element as shown below:




```

<tx currency=dollars>
<acct>cash</acct>
<amt>12</amt>
</tx>

```

*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	
 GSX	http://www.gsx.com
 TxMon	http://www.acutrak/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.acutrak/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.

*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

To produce attributes, put the attribute values in keys whose name starts with the character(s) specified in attrPrefix. For example, if you use the default attrPrefix, the names of all keys containing attributes (and only those keys containing attributes) must start with the @ character (for example, @type, @xmlns). Also, when you include attributes, ensure that keys representing attributes are direct children of the elements in which they are to be applied.

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

*attrPrefix*

**String.** Optional. Prefix that designates keys containing attributes. The default prefix is "@". For example, if you set attrPrefix to ATT\_ and document contains a key ATT\_currency whose value is "dollars",

 tx	
 ATT_currency	dollars
 acct	cash
 amt	120.00

then this service will convert the ATT\_currency key to the attribute, currency=dollars, in the <tx> element as shown below:

```

<tx currency=dollars>
<acct>cash</acct>
<amt>12</amt>
</tx>

```




*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	
 GSX	http://www.gsx.com
 TxMon	http://www.acutrak/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.

*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


To produce attributes, put the attribute values in keys whose name starts with the character(s) specified in attrPrefix. For example, if you use the default attrPrefix, the names of all keys containing attributes (and only those keys containing attributes) must start with the @ character (for example, @type, @xmlns). Also, when you include attributes, ensure that keys representing attributes are direct children of the elements in which they are to be applied.

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">
```

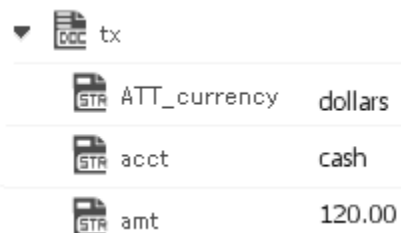
```

<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

*attrPrefix* **String.** Optional. Prefix that designates keys containing attributes. The default prefix is "@". For example, if you set *attrPrefix* to *ATT\_* and document contains a key *ATT\_currency* whose value is "dollars",



tx	
ATT_currency	dollars
acct	cash
amt	120.00

then this service will convert the *ATT\_currency* key to the attribute, *currency=dollars*, in the *<tx>* element as shown below:




```

<tx currency=dollars>
<acct>cash</acct>
<amt>12</amt>
</tx>

```

*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	
 GSX	http://www.gsx.com
 TxMon	http://www.acutrak/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.acutrak/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.

*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



To produce attributes, put the attribute values in keys whose name starts with the character(s) specified in attrPrefix. For example, if you use the default attrPrefix, the names of all keys containing attributes (and only those keys containing attributes) must start with the @ character, for example, @type, @xmlns. Also, when you include attributes, ensure that keys representing attributes are direct children of the elements in which they are to be applied.

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.

*attrPrefix* **String.** Optional. Prefix that is to be used to designate keys containing attribute values. The default is "@". For example, if you set attrPrefix to ATT\_ and node contains the following element:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
```




</tx>

Then this service will convert the currency attribute as follows:

▼  tx
 ATT_currency    dollars
 acct                cash
 amt                    120.00

*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
 GSX                http://www.gsx.com
 TxMon             http://www.acutrak/txmonitor

## Output Parameters

*document*        **Document.** Document representation of nodes and attributes in node.

## Usage Notes

Following are examples of XML documents and the documents that *xmlBytesToDocument* will produce:

XML Document

Document

```
<myDoc><e1>e1Value</e1>
</myDoc>
```

document	
myDoc	
e1	e1Value

```
<?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

## xmlStreamToDocument

Converts an XML content stream to a document. This service transforms each element and attribute in the XML content stream to an element in a Document.

This service will convert the XML stream containing the following XML content:

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
```

```

<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>

```

To a Document that looks like:

▼ document	
@version	1.0
▼ AcctInfo	
▼ accNum	
*body	G97041A
@type	Platinum
▼ address[0]	
@country	USA
city	closed
postalCode	22130
state	OH
street1	10211 Brook Road
▼ address[1]	
*doctype	DerivedAddress.documentLocation:docTypeRef_tns_DerivedAddress
@country	USA
city	closed
landmark	Ohio River-Bank Square
postalCode	22130
state	OH
street1	10211 Brook Road
name	Midwest Extreme Sports
▼ phoneNum	
*body	216-741-7566
@cc	011
rep	Laura M. Sanchez
▼ serialNum [ ]	
serialNum[0]	19970523A



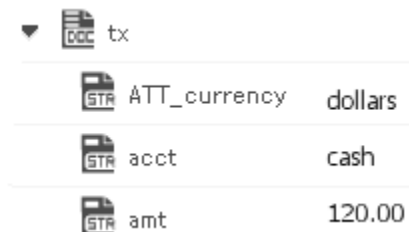
## Input Parameters

*xmlStream* **java.io.InputStream**. XML content stream that is to be converted to a document.

*attrPrefix* **String**. Optional. Prefix that is to be used to designate keys containing attribute values. The default is "@". For example, if you set attrPrefix to ATT\_ and node contains the following element:




```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
</tx>
```

Then this service will convert the currency attribute as follows:



tx	
ATT_currency	dollars
acct	cash
amt	120.00

*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	
 GSX	<a href="http://www.gsx.com">http://www.gsx.com</a>
 TxMon	<a href="http://www.acutrak/txmonitor">http://www.acutrak/txmonitor</a>

## 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>&lt;myDoc&gt;&lt;e1&gt;e1Value&lt;/e1&gt; &lt;/myDoc&gt;</pre>	<ul style="list-style-type: none"> <li>▼  document <ul style="list-style-type: none"> <li>▼  myDoc <ul style="list-style-type: none"> <li> e1 <span style="float: right;">e1Value</span></li> </ul> </li> </ul> </li> </ul>
<pre>&lt;?xml version="1.0" encoding="UTF-8" standalone="no"?&gt;&lt;myDoc&gt;&lt;e1&gt; e1Value&lt;/e1&gt;&lt;/myDoc&gt;</pre>	<ul style="list-style-type: none"> <li>▼  document <ul style="list-style-type: none"> <li> @encoding <span style="float: right;">UTF-8</span></li> <li> @standalone <span style="float: right;">no</span></li> <li> @version <span style="float: right;">1.0</span></li> <li>▼  myDoc <ul style="list-style-type: none"> <li> e1 <span style="float: right;">e1Value</span></li> </ul> </li> </ul> </li> </ul>
<pre>&lt;?xml version="1.0" encoding="UTF-8" standalone="no"?&gt;&lt;myDoc&gt; &lt;e1 e1Attr="attrValue"&gt; e1Value&lt;/e1&gt;&lt;/myDoc&gt;</pre>	<ul style="list-style-type: none"> <li>▼  document <ul style="list-style-type: none"> <li> @encoding <span style="float: right;">UTF-8</span></li> <li> @standalone <span style="float: right;">no</span></li> <li> @version <span style="float: right;">1.0</span></li> <li>▼  myDoc <ul style="list-style-type: none"> <li>▼  e1 <ul style="list-style-type: none"> <li> *body <span style="float: right;">e1Value</span></li> <li> @e1Attr <span style="float: right;">attrValue</span></li> </ul> </li> </ul> </li> </ul> </li> </ul>

```
<?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
▼ phoneNum		
*body		216-741-7566
@cc		011
rep		Laura M. Sanchez
▼ serialNum [ ]		
serialNum[0]		19970523A

## Input Parameters



*xmlString*      **String.** XML string that is to be converted to a document.

*attrPrefix*      **String.** Optional. Prefix that is to be used to designate keys containing attribute values. The default is "@". For example, if you set attrPrefix to ATT\_ and node contains the following element:

```
<tx currency=dollars>
<acct>cash</acct>
<amt>120.00</amt>
```




</tx>

Then this service will convert the currency attribute as follows:

 tx		
 ATT_currency	dollars	
 acct	cash	
 amt	120.00	

*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		
 GSX	http://www.gsx.com	
 TxMon	http://www.acutrak/txmonitor	

## Output Parameters

*document* **Document.** Document representation of nodes and attributes in node.

## Usage Notes

Following are examples of XML documents and the documents that *xmlStringToDocument* will produce:

XML Document

Document

```
<myDoc><e1>e1Value</e1>
</myDoc>
```

document	
myDoc	
e1	e1Value

```
<?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
<a href="#">bytesToStream</a>	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>	<b>byte[ ]</b> The byte array to convert.
<i>length</i>	<b>String</b> 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>	<b>String</b> 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>	<b>java.io.ByteArrayInputStream</b> 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>	<b>java.io.InputStream</b> 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.

---

## 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 input or output parameters for an Integration.

**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 needed to declare input or output parameters or the effort/time needed to build other document fields.
- Document Types improve accuracy because there is less possibility to introduce a typing error 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.  
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 in the screen.
5. 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. 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.

**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.
6. 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” page.

### Related Topics

[Creating Orchestrated Integrations](#)

## 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.

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.

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.

**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.

---

### 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 in the screen.
5. Click “Browse” and select a .txt or .csv file. Only a text file (.txt or .csv) 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. If a Reference Data is used by an Integration, you will not be able to delete the Reference Data.

### Related Topics

[Reference Data Signature](#)

[Integrations](#)

[Creating Orchestrated Integrations](#)

[Built-In Services](#)

## 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:







### AccountType Input

▼  matchCriteria [ ] \*


-  path \*
-  compareValueAs
-  datePattern
-  joins [ ] \*

### AccountType Output

▼  AccountType [ ]

-  Type
-  Our Type
-  Marketer

## 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
<b>Created on</b>	Displays the date when the Integration was created.
<b>Created by</b>	Displays the user who created the Integration.
<b>Stages tabs</b>	Displays different stages of the Integration development. You can pull Integrations from all other stages except from the <b>Development</b> stage.
<b>Overview</b>	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, or expose the Integration as a REST service from this screen.

Option	Description
<b>Last 5 Execution Results</b>	Click 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.
<b>Edit</b>	Click to modify the Integration.
<b>Delete</b>	Click to delete the Integration from a stage.
<b>Uses</b>	Displays the components used to create the Integration.
<b>Last modified/ Last modified by</b>	When and by whom was the Integration last modified.
<b>Status</b>	If the Integration in a stage is scheduled, then the status of the Integration displays <b>Scheduled</b> , else it appears as <b>Not Scheduled</b> . The Status appears as <b>Schedule Paused</b> if the Integration has been paused.
<b>Last execution</b>	When was the Integration last executed. A warning message appears if the last execution was not successful.
<b>Next scheduled execution</b>	When is the Integration scheduled to run again.
<b>Schedule</b>	Click to schedule the Integration to run once immediately or at a specified date and time. You can also define a recurrence pattern if you want to run the Integration recurrently. Click <b>Next</b> to provide inputs to the Integration based on the defined input signature.
<b>Run Now</b>	Click to submit the Integration for execution. You can provide inputs to the Integration based on the defined input signature.
<b>Resume</b>	Click to start the Integration that was paused.
<b>Pull</b>	Click 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.
<b>Remove</b>	Click to remove an Integration from a stage. You can remove an Integration from all stages except from the <i>Development</i> stage.
<b>Expose as a REST service</b>	Check this option if you want to trigger the execution of an Integration from an external system. By default, Integrations

Option	Description
	<p>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 <b>Show Advanced Options</b> 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"> <li>■ 0 - SUCCESS: Successfully submitted the Integration for execution.</li> <li>■ -1 - ERROR: Problem while submitting the Integration for execution.</li> </ul> <p><i>HTTP Status Codes</i></p> <ul style="list-style-type: none"> <li>■ 200 - OK</li> <li>■ 201 - Created</li> <li>■ 500 - Internal Server Error</li> <li>■ 401 - Unauthorized User Error</li> </ul> <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>https://&lt;sub-domain&gt;.webMethodscloud.com/integration</pre> <p><b>Example:</b></p> <pre>https://&lt;sub-domain&gt;.webmethodscloud.com/integration/rest/assembly/external/execution/result?resultReference=76fb5733-6a21-4b02-864f-5e958f698373</pre>

Option	Description
	<i>Application Status Codes</i>
	■ 0 - SUCCESS
	■ -1 - ERROR
	<i>HTTP Status Codes</i>
	■ 200 - OK
	■ 500 - Internal Server Error
	■ 401 - Unauthorized User Error
	■ 404 - Not Found
	<b>Note:</b> 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 executed or the <i>resultReference</i> is not correct.

### Related Topics

[Creating Point-to-Point Integrations](#)

[Creating Orchestrated Integrations](#)

[Integrations](#)

[Execution Results](#)

## Execution Results

The **Execution Results** screen allows you to view the audit trail of all the executions that happened in a stage.

### 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 Details screen. You can see the last five execution results in the “Last 5 Execution Results” tab.

You can also click the “Execution Results” link in the Home page or click **Develop > Execution Results** to view the **Execution Results** screen.

4. Select the **Integration**, the **Stage**, and the time period for which you want to view the execution results.
5. Click “Show Results” to view the execution results.

**Related Topics**[Integrations](#)[Creating Point-to-Point Integrations](#)[Creating Orchestrated Integrations](#)[Stages Management](#)