

webMethods Integration Cloud Help

Version 5.1.0

January 2019

This document applies to webMethods Integration Cloud Version 5.1.0 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 © 2014-2019 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

Welcome to webMethods Integration Cloud	15
Document Conventions	17
Online Information and Support	19
Data Protection	21
What's new	23
Version 5.1.0 (February 2019).....	24
Version 5.0.0 (October 2018).....	27
Version 4.6.0 (July 2018).....	30
Version 4.5.0 (April 2018).....	33
Version 4.1.0 (January 2018).....	37
Registration	41
Registration Overview.....	42
Creating an Account.....	42
Securing your Account.....	45
Settings	47
Users.....	48
Adding Users.....	48
Updating Users.....	52
Resetting Passwords.....	53
User Profile.....	53
Security Question.....	54
Change Password.....	54
My Certificate.....	54
Access Profiles.....	55
Adding or Updating Access Profiles.....	56
Access Control Lists.....	61
Adding or Updating Access Control Lists.....	62
Single Sign-On.....	63
Configuring SAML Settings for Single Sign-On.....	65
Company Information.....	70
Updating Company Information.....	70
Password Policy.....	72
Updating Password Policy Settings.....	72
Client Certificate.....	73
Adding Client Certificates.....	75
OAuth 2.0.....	76
About OAuth 2.0.....	76

Configuring OAuth 2.0.....	77
Registering Clients.....	78
Managing Scopes.....	81
Generating Tokens.....	82
Refreshing Access Tokens Using Refresh Tokens.....	85
Managing Tokens.....	85
Running Services Using OAuth 2.0.....	86
Capability.....	87
Connect.....	89
Applications.....	90
Accounts.....	90
Adding or Editing Accounts.....	91
Account Configuration Details.....	93
Alfabet.....	93
Apache Solr Search.....	95
Applicability Statement 2 (AS2).....	96
Amazon DynamoDB.....	100
Amazon Kinesis.....	102
Amazon Simple Notification Service(SNS).....	103
Amazon Simple Queue Service (SQS).....	105
Amazon Simple Storage Service (S3).....	106
Atlassian Jira.....	108
Avalara AvaTax.....	110
Cloud Deployment.....	112
CloudStreams Connector for Anaplan®.....	112
CloudStreams Connector for Microsoft Azure Cosmos DB.....	114
CloudStreams Connector for Microsoft Azure Storage.....	115
CloudStreams Connector for NetSuite™.....	116
CloudStreams Connector for Salesforce® Bulk v2 Data Loader.....	118
Concur.....	120
Coupa.....	122
Cumulocity.....	123
DocuSign.....	125
File Transfer Protocol (FTP/FTPS).....	126
Google Analytics.....	127
Google Apps Admin.....	129
Google BigQuery.....	130
Google Calendar.....	132
Google Contacts.....	133
Google Drive.....	135
Google Cloud Pub/Sub.....	136
Google Cloud Storage.....	138
Google Prediction.....	139
Google Sheets.....	141

IBM Watson Tone Analyzer.....	142
Magento eCommerce Platform.....	144
Marketo.....	145
Microsoft Dynamics CRM.....	147
OData v2.0.....	151
OData v4.0.....	152
On-Premises Applications.....	154
REST Applications - Account Configuration Details.....	154
Salesforce.....	158
SAP Cloud for Customer(C4C) OData v2.0.....	162
SAP S/4HANA Marketing Cloud.....	164
Secure File Transfer Protocol (SFTP).....	166
ServiceNow Enterprise Service Management.....	169
Siemens MindSphere.....	171
Slack.....	173
SOAP Applications - Account Configuration Details.....	174
Strikelron Contact Verification.....	176
SuccessFactors HCM.....	178
SugarCRM.....	180
Twilio.....	181
Zendesk.....	183
Operations.....	185
Adding or Editing Operations.....	186
AS2 Predefined Operations.....	192
receive.....	192
send.....	195
Configuring the Auto Detect Option.....	198
Creating an Endpoint URL.....	198
FTP Predefined Operations.....	199
getFile.....	200
listFiles.....	200
deleteFiles.....	201
putFile.....	201
renameFile.....	202
SFTP Predefined Operations.....	203
cd.....	204
chgrp.....	204

chmod.....	205
chown.....	205
get.....	205
ls.....	206
mkdir.....	207
put.....	207
pwd.....	208
rename.....	208
rm.....	209
rmdir.....	209
symlink.....	210
Cloud Deployment Operations.....	210
Upgrade.....	211
Creating and updating SOAP Applications.....	212
SOAP Signature.....	214
REST Applications.....	219
Creating and updating REST Applications.....	221
Keys and Certificates.....	227
Add Keystore.....	227
Add Truststore.....	228
Add Partner Certificate.....	229
Develop.....	231
Integrations.....	232
Point-to-Point Integrations.....	233
Orchestrated Integrations.....	235
Pipeline and Signatures.....	253
Built-In Services.....	257
Date.....	257
Summary of Date services.....	261
.....	
calculateDateDifference.....	262
compareDates.....	263
currentNanoTime.....	264
dateBuild.....	265
dateTimeBuild.....	266

dateTimeFormat.....	267
elapsedNanoTime.....	268
formatDate.....	269
getCurrentDate.....	270
getCurrentDateString.....	270
incrementDate.....	271
Document.....	273
Summary of Document Services.....	273
.....	
findDocuments.....	274
insertDocument.....	275
deleteDocuments.....	275
documentListToDocument.....	276
documentToDocumentList.....	278
groupDocuments.....	279
documentToBytes.....	280
bytesToDocument.....	281
searchDocuments.....	282
sortDocuments.....	283
List.....	284
Summary of List services.....	284
.....	
addItemToVector.....	285
appendToDocumentList.....	286
appendToStringList.....	287
sizeOfList.....	287
stringListToDocumentList.....	288
vectorToArray.....	289
Math.....	290
Summary of Math services.....	290
.....	

addObjects.....	291
divideObjects.....	292
min.....	293
multiplyObjects.....	293
subtractObjects.....	294
toNumber.....	295
absoluteValue.....	295
addFloatList.....	295
addFloats.....	296
addIntList.....	297
addInts.....	298
divideFloats.....	298
divideInts.....	300
max.....	300
multiplyFloatList.....	301
multiplyFloats.....	302
multiplyIntList.....	303
multiplyInts.....	303
randomDouble.....	304
roundNumber.....	304
subtractFloats.....	305
subtractInts.....	306
Storage.....	306
Summary of Storage services.....	309
.....	
add.....	310
get.....	311
keys.....	312
lock.....	312

put.....	314
remove.....	315
unlock.....	316
String.....	316
Summary of String services.....	317
.....	
HTMLDecode.....	319
HTMLEncode.....	320
base64Decode.....	321
base64Encode.....	321
bytesToString.....	322
concat.....	322
indexOf.....	323
length.....	323
lookupDictionary.....	324
makeString.....	324
messageFormat.....	325
numericFormat.....	325
objectToString.....	327
padLeft.....	327
padRight.....	328
replace.....	329
stringToBytes.....	330
substring.....	330
tokenize.....	331
toLowerCase.....	331
toUpperCase.....	332
trim.....	332
URLDecode.....	333

URLEncode.....	333
fuzzyMatch.....	334
isNumber.....	335
isAlphanumeric.....	335
isNullOrBlank.....	336
isDate.....	336
substitutePipelineVariables.....	337
compareStrings.....	338
Flow.....	338
Summary of Flow services.....	338
.....	
clearPipeline.....	339
getLastError.....	340
getSessionInfo.....	341
getHTTPRequest.....	343
setHTTPResponse.....	343
countProcessedDocuments.....	344
logCustomMessage.....	344
sleep.....	345
Hashtable.....	345
Summary of Hashtable services.....	345
.....	
containsKey.....	346
createHashtable.....	346
get.....	346
listKeys.....	347
put.....	347
remove.....	348
size.....	348
Flat File.....	348

Summary of Flat File services.....	349
.....	
delimitedDataBytesToDocument.....	349
delimitedDataStreamToDocument.....	352
delimitedDataStringToDocument.....	355
documentToDelimitedDataBytes.....	358
documentToDelimitedDataStream.....	360
documentToDelimitedDataString.....	362
JSON.....	364
Summary of JSON services.....	364
.....	
documentToJSONBytes.....	365
documentToJSONStream.....	365
documentToJSONString.....	366
jsonBytesToDocument.....	366
jsonStreamToDocument.....	367
jsonStringToDocument.....	368
XML.....	368
Summary of XML services.....	368
.....	
documentToXMLBytes.....	369
documentToXMLStream.....	373
documentToXMLString.....	377
xmlBytesToDocument.....	381
xmlStreamToDocument.....	388
xmlStringToDocument.....	394
IO.....	401
Summary of IO services.....	401
.....	
bytesToStream.....	402
streamToBytes.....	403

streamToString.....	403
stringToStream.....	403
Utils.....	404
Summary of Utils services.....	404
.....	
generateUUID.....	404
Integration Details.....	405
Exporting Integrations.....	410
Importing Integrations.....	411
REST APIs.....	412
Creating REST APIs from scratch.....	412
Modifying Resource Operations.....	414
Creating REST APIs with Swagger.....	416
Copying REST APIs.....	418
Exporting REST APIs.....	418
Importing REST APIs.....	419
Document Types.....	420
Reference Data.....	422
Reference Data Signature.....	425
Recipes.....	426
Monitor.....	429
Monitor Overview.....	430
Dashboard.....	430
Execution Results.....	432
Audit Log.....	434
Alert Rules.....	435
Stage in view.....	437
Manage.....	438
Applying Access Profiles to a Stage.....	439
Deploy.....	440
Change Stage To View.....	440
Containers.....	443
Containers Overview.....	444
Managing Repositories.....	445
Viewing Tag Details.....	446
Managing Services.....	447
Cloud Deployment.....	449
About Cloud Deployment.....	450
Solutions.....	454
Solution List.....	454

Exploring a Solution.....	457
Assets.....	458
Deploy.....	461
Asset Repository.....	462
Manage.....	463
Landscape.....	463
Service Access Settings.....	464
Administration.....	466
Restart.....	467
Asset Repository.....	468
Capability.....	468
Usage.....	470
Monitoring.....	470
Dashboard.....	471
Solutions.....	472
Runtimes.....	473
Viewing Adapter KPIs.....	475
Services.....	475
Uptime.....	476
Alerts.....	477
Alert Types.....	478
Configuring the Alerts.....	479
Logs.....	480

Welcome to webMethods Integration Cloud

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

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

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

Integration Cloud enables:

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

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

Document Conventions

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

Online Information and Support

Software AG Documentation Website

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

Software AG Empower Product Support Website

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

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

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

To submit feature/enhancement requests, get information about product availability, and download products, go to "[Products](#)".

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the "[Knowledge Center](#)".

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

Software AG TECHcommunity

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

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

Data Protection

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

1 What's new

■ Version 5.1.0 (February 2019)	24
■ Version 5.0.0 (October 2018)	27
■ Version 4.6.0 (July 2018)	30
■ Version 4.5.0 (April 2018)	33
■ Version 4.1.0 (January 2018)	37

Version 5.1.0 (February 2019)

This section describes the enhancements and changes made in Version 5.1.0 for **Integration Cloud**:

Item	Description
“Support for multi-part request body” on page 186	<p>For some Applications and Operations, for example, for the <i>CloudStreams Connector for Salesforce(R) Bulk v2 Data Loader Application</i> and <i>Create Job and Upload Job Data Operation</i>, Integration Cloud supports multipart request body.</p>
<p>User Interface changes due to performance enhancements</p>	<p>The following user interface changes are made to enhance the performance and reduce the wait time for user actions:</p> <ul style="list-style-type: none"> ■ Applications are now listed under different categories such as Predefined Applications, REST Applications, On-Premises Applications, and SOAP Applications. ■ The Integrations list page (Develop > Integrations) now shows a basic view of all the Integrations. To view the references (Uses column), select the <i>Show Advanced View</i> check box available on the Integrations list page.
<p>Mapping enhancement</p>	<p>While doing pipeline mapping, you can now view only the mapped fields by selecting the <i>Show Only Mapped Fields</i> check box.</p>
“Support for viewing Integration Cloud Capabilities” on page 87	<p>The new <i>Capabilities</i> page allows you to view the status of some of the system capabilities, based on your license offering. To access this page, from the Integration Cloud navigation bar, click  and select <i>Licensing > Capability</i>.</p>
“Support for invoking deployed services from Integration Cloud” on page 210	<p>Integration Cloud Integrations can now invoke Cloud Deployment webMethods Integration Server Flow and Java services for the same tenant. A new pre-defined Application, <i>Cloud Deployment</i>, is added in Integration Cloud. Using this Application, you can select the solution webMethods Integration Server services that you want to call from Integration Cloud.</p>

Item	Description
"OAuth 2.0 enhancements" on page 76	<p>The following OAuth 2.0 enhancements have been implemented in this release:</p> <ul style="list-style-type: none"><li data-bbox="727 415 1541 552">■ From the <i>Client Registration</i> page, you can now associate scopes with a client as well as create a new scope and associate it with a client by clicking the <i>Associate Scopes</i> option.<li data-bbox="727 573 1541 667">■ Support is added for <i>Client Credentials Grant</i> and <i>Resource Owner Password Credential Grant</i> types as part of OAuth spec for OAuth token generation flow.<li data-bbox="727 688 1541 972">■ After you select the <i>Enable Integration to be invoked over HTTP</i> option on the <i>Integration Details</i> page, <i>OAuth Scopes containing the exposed Integration URL</i> appears. Clicking the <i>OAuth Scopes</i> option displays the OAuth Scopes which contain the exposed URL of the Integration. You can add the exposed URL of the Integration to an existing scope or add a new Scope which will contain the exposed URL of the Integration.<li data-bbox="727 993 1541 1224">■ From the REST API Resources page for a REST API which is created from scratch, you can now click the <i>OAuth Scopes</i> option to view the OAuth Scopes which contain the REST Resource path with Method. You can add the REST Resource path with Method to an existing scope or add a new scope which will contain the REST Resource path with Method.<li data-bbox="727 1245 1541 1350">■ While creating or updating a scope, you can now select the exposed Integrations and REST Resources that you want to add as <i>Service URLs</i> from the <i>Services</i> dialog box.<li data-bbox="727 1371 1541 1434">■ Enhanced the OAuth Approval page generated in the OAuth token generation flow.<li data-bbox="727 1455 1541 1518">■ SSO support for authentication during OAuth token generation flow.<li data-bbox="727 1539 1541 1606">■ Support for auditing OAuth Client, Scope, and Token CRUD operations.

This section describes the enhancements and changes made in Version 5.1.0 for **Cloud Deployment**:

Item	Description
Support for usage based license monitoring of CPU and Memory capacity	<p>You can now view the cumulative usage of CPU and Memory for all the active solutions in all the stages from the <i>Usage</i> page.</p> <p>To access this page, from the Cloud Deployment navigation bar, click  and select <i>Licensing > Usage</i>.</p> <p>Further, on the Cloud Deployment landing page, you can see the current usage (CPU and Memory) for the tenant.</p>
Continuous Integration/Continuous Deployment (CI/CD)	<p>You can now build user-created assets and configurations using webMethods Asset Build Environment (ABE), retrieve those assets and configurations from a VCS by using ABE or an automated tool like Jenkins, and then deploy those assets and configurations to the Integration Cloud<i>Development Stage</i> using ABE.</p> <div data-bbox="732 919 1513 1050" style="background-color: #f0f0f0; padding: 10px;"> <p>Note: For information on CI/CD and how to promote assets across stages, see the <i>Deploying to webMethods Integration Cloud</i> document.</p> </div>
Landscape component status	<p>You can now see the status of a solution landscape component whether it is Ready or Not Ready on the landscape page.</p>
Support for stateless cluster of webMethods Integration Servers	<p>In addition to supporting stateful webMethods Integration Server clusters, you can now define that a cluster is <i>stateless</i> from the <i>Landscape</i> page for a solution. The deployment of a solution will therefore not need a Terracotta Server Array.</p>
Support for viewing more assets	<p>You can now view more on-premises assets for Cloud Deployment.</p>
“Deployment Status Tracking” on page 461	<p>You can now type a message to describe the rollback. The rollback message appears on the <i>History</i> page.</p> <p>The History page shows the <i>Trace ID</i>, that is, the tracking ID, which is automatically generated on every successful or unsuccessful promotion or rollback, or deletion, the Deployment, Rollback, or Deletion <i>Action</i>, <i>Date</i> when the asset was promoted, rolled back, or deleted, the <i>User</i> who promoted, rolled back, or deleted the asset, and the</p>

Item	Description
	commit <i>Message</i> for the selected instance. You can click on a Trace ID to see the <i>Track History</i> for the specific action.
“Support for deleting assets” on page 461	You can now delete assets from all stages from the <i>Deploy</i> page for a solution.
“Support for viewing Cloud Deployment Capabilities” on page 87	The new <i>Capabilities</i> page allows you to view the status of some of the Cloud Deployment capabilities, based on your license offering. To access this page, from the Cloud Deployment navigation bar, click  and select <i>Licensing > Capability</i> .
Support for webMethods Adapter KPI monitoring	You can now monitor the following details of all the installed Adapters: <ul style="list-style-type: none"> ■ Total number of enabled and disabled connection pools ■ Total number of enabled and disabled listeners ■ Total number of enabled and disabled notifications
Duplicate low severity alerts	In earlier releases, along with critical alerts, redundant alerts of lower severity like warning and info alerts were also displayed. From this release, if there are critical alerts, lower severity alerts will not be displayed.

Version 5.0.0 (October 2018)

This section describes the enhancements and changes made in Version 5.0.0 for Integration Cloud:

Item	Description
Debug Orchestrated Integrations	You can now debug an orchestrated Integration and can inspect the data flow during the debugging session. You can inspect and edit the pipeline data before and after executing the individual blocks. Further, you can step over each block one at a time, or can specify breakpoints where you want to halt the execution.
Support for OAuth 2.0	You can now protect Integration services and REST APIs using the OAuth 2.0 protocol.

Item	Description
Support for enabling dynamic account configuration details at runtime	For REST Application operations, you can now provide account configurations at runtime during integration execution. When you create or edit an operation of a REST application, you can select the fields you want to pass at runtime in the <i>Dynamic Account Configuration Fields</i> dialog box.
Support for configuring two-way SSL for inbound connections	Integration Cloud now supports two-way SSL for inbound connections. You can store client certificates and associate a certificate with a user account.
Support for refreshing OAuth 2.0 Access Tokens	To refresh the access tokens for accounts which use the OAuth 2.0 protocol, you can now specify a call-back Integration which will execute when the access token expires. Select <i>Custom ESB Service</i> in the <i>Refresh URL Request</i> field and provide the Integration name in the <i>Refresh Custom ESB Service</i> field.
New Applications	<p><i>Applicability Statement 2 (AS2)</i> - Applicability Statement 2 (AS2) allows you to transport EDI and other business data securely and reliably using the HTTP transport protocol. The application supports the AS2 protocol versions 1.1 and 1.2.</p> <p><i>Microsoft Dynamics CRM 365</i> - Microsoft Dynamics CRM 365 allows you to manage CRM data and access metadata that defines the specific CRM instance to which you are connecting using the OData API Version 4.0. This Application performs standard CRUD operations on business objects by connecting to the OData service endpoint.</p>
Support for configuring Headers and Parameters for REST-based Applications	You can now add and configure the Headers and Parameters for REST-based Applications.
Support for viewing associated ACLs for an Access Profile	You can now view ACLs associated with an Access Profile from the Access Profile details page.
Support for editing the default Access Control List	You can now edit the default Access Control List. Further, the name of the default ACL has been changed from <i>Everybody</i> to <i>Default</i> .
Service changes	<i>New services</i>

Item	Description
	<p>sleep: This service has been added under the Flow category.</p> <p><i>Updated Services</i></p> <p>xmlStringToDocument: New input parameters have been added to this service.</p>
Enhancements for Interactions and Business Objects for Operations	You can now edit interactions in a single request and also edit multiple business objects in a single request. Further, you can drag and drop the interactions or business objects to change the order in which they will execute.

This section describes the enhancements and changes made in Version 5.0.0 for Cloud Deployment:

Item	Description
Cloud Deployment	Cloud Deployment allows you to deploy on-premises assets and configurations, which are developed using Software AG Designer and on-premises webMethods Integration Server, to Integration Cloud. You can choose from predefined solution landscapes to deploy your on-premises assets from Software AG Designer. The assets and configurations can be promoted from one stage to another stage. During promotion, the asset field values can be changed. You can monitor the health and availability of the solutions and run-time instances, alerts, and alert statuses for all the deployed solutions.
New Administrative Permissions for Solutions	Permissions for creating, updating, and deleting solutions have been added under <i>Access Profile > Administrative Permissions > Functional Controls</i> . The permissions allow you to add, modify, and delete solutions available on Integration Cloud.
Solution Permissions	A new <i>Solution Permissions</i> page in Access Profiles is now available which allows you to map webMethods Integration Server user groups to an Access Profile. With this mapping, the Integration Cloud user who has that Access Profile can perform tasks on the solution webMethods Integration Server(s) by being present in that particular user group.

Item	Description
Customized message while promoting assets	You can now type a message while promoting assets. The promotion message appears on the new <i>History</i> page.
Infrastructure provisioning status for Cloud Deployment	You can now see the status when a solution landscape is provisioning or has been successfully provisioned. The user interface pages will be disabled when a solution is being provisioned.

Version 4.6.0 (July 2018)

This section describes the enhancements and changes made in Version 4.6.0:

Item	Description
Reference Data enhancements	<p>You can now view, edit, delete, and download Reference Data in all stages. The new <i>Status</i> column in the Reference Data table displays <i>Configured</i> if the Reference Data is available in the current stage and displays <i>Not Configured</i> if the Reference Data is not available in the current stage but available in any other stage.</p>
Support for nested and multiple Business Objects and Interactions	<p>You can now create or update <i>multiple business objects</i>, for example, Contact, Opportunities, and Account in a single request for the Salesforce v42 Application. The following operations have been added for the Salesforce v42 Application:</p> <ul style="list-style-type: none"> ■ <i>createMultiple</i> - This operation allows you to add one or more records of different business object types. For example, Account and Contact business objects can be created in a single invocation. ■ <i>updateMultiple</i> - This operation allows you to update one or more records of different business object types. For example, Account and Contact business objects can be updated in a single invocation. <p>You can also add <i>interactions</i> (sub-operations), for example, Create, Update, Upsert, and Delete, and then <i>associate those interactions with business objects</i> in a single request for the OData 4.0 Application. The following operations are available for the OData 4.0 Application:</p> <ul style="list-style-type: none"> ■ <i>Batch</i> - Batch requests allow grouping multiple interactions into a single HTTP request payload. Batch

Item	Description
	<p>allows you to create, update, read, and/or delete entities of same or different entity types in a single request.</p> <ul style="list-style-type: none"> ■ <i>ChangeSet</i> - A change set is an atomic unit of work consisting of an unordered group of one or more data modification requests. ChangeSet allows you to create, update and/or delete entities of same or different entity types in a single request. <p>Note: Business Objects and Interactions appear only for certain Applications and Operations.</p> <p>Further, for some operations, for example, for the <i>Retrieve Contained Or Derived Entity</i> operation in the OData 4.0 Application, Integration Cloud displays nested business objects. You can expand the nested business objects to display the child-level objects.</p>
<p>Controlling Integration Executions using Access Control Lists (ACLs)</p>	<p>You can now use Access Control Lists (ACLs) to control the execution permission of an Integration. An ACL can be assigned to an Integration and a user can be associated with the ACL through the Access Profile. Therefore using ACLs, you can control users who can execute an Integration.</p>
<p>Users associated with an Access Profile</p>	<p>You can now view the list of users associated with an Access Profile by clicking the Access Profile link on the Access Profiles page. You can also associate multiple Access Profiles with an ACL while creating or modifying an ACL.</p>
<p>Ability to Copy, Export, Import, and Refresh REST APIs</p>	<p>You can now copy, export, import, and also refresh a REST API.</p>
<p>New Applications</p>	<p>The following Applications are added in this release:</p> <ul style="list-style-type: none"> ■ <i>Amazon Kinesis</i> - Amazon Kinesis is a managed service that scales elastically for real-time processing of streaming big data. The most common Amazon Kinesis use case scenario is rapid and continuous data intake and aggregation. ■ <i>Twilio</i> - Using the REST interface, Twilio allows you to programmatically make and receive phone calls and send and receive text messages.

Item	Description
New Services	<p data-bbox="701 321 1541 394">New services have been added under the following categories:</p> <p data-bbox="701 415 1541 447"><i>Date</i></p> <ul data-bbox="701 468 1541 762" style="list-style-type: none"><li data-bbox="701 468 1541 541">■ <code>currentNanoTime</code> - Returns the current time returned by the most precise system timer, in nanoseconds.<li data-bbox="701 552 1541 625">■ <code>elapsedNanoTime</code> - Calculates the time elapsed between the current time and the given time, in nanoseconds.<li data-bbox="701 636 1541 667">■ <code>formatDate</code> - Formats a <code>Date</code> object as a string.<li data-bbox="701 678 1541 762">■ <code>getCurrentDate</code> - Returns the current date as a <code>Date</code> object. <p data-bbox="701 783 1541 814"><i>Document</i></p> <ul data-bbox="701 835 1541 909" style="list-style-type: none"><li data-bbox="701 835 1541 909">■ <code>searchDocuments</code> - Searches a set of documents for entries matching a set of <code>Criteria</code>. <p data-bbox="701 930 1541 961"><i>List</i></p> <ul data-bbox="701 982 1541 1129" style="list-style-type: none"><li data-bbox="701 982 1541 1056">■ <code>addItemToVector</code> - Adds an item or a list of items to a <code>java.util.Vector</code> object.<li data-bbox="701 1066 1541 1129">■ <code>vectorToArray</code> - Converts a <code>java.util.Vector</code> object to an array. <p data-bbox="701 1150 1541 1182"><i>Math</i></p> <ul data-bbox="701 1203 1541 1675" style="list-style-type: none"><li data-bbox="701 1203 1541 1276">■ <code>addObjects</code> - Adds one <code>java.lang.Number</code> object to another and returns the sum.<li data-bbox="701 1287 1541 1360">■ <code>divideObjects</code> - Divides one <code>java.lang.Number</code> object by another (<code>num1/num2</code>) and returns the quotient.<li data-bbox="701 1371 1541 1444">■ <code>min</code> - Returns the smallest number from a list of numbers.<li data-bbox="701 1455 1541 1528">■ <code>multiplyObjects</code> - Multiplies one <code>java.lang.Number</code> object by another and returns the product.<li data-bbox="701 1539 1541 1612">■ <code>subtractObjects</code> - Subtracts one <code>java.lang.Number</code> object from another and returns the difference.<li data-bbox="701 1623 1541 1675">■ <code>toNumber</code> - Converts a string to numeric data type. <p data-bbox="701 1696 1541 1728"><i>String</i></p> <ul data-bbox="701 1749 1541 1896" style="list-style-type: none"><li data-bbox="701 1749 1541 1822">■ <code>HTMLDecode</code> - Replaces HTML character entities with native characters.<li data-bbox="701 1833 1541 1896">■ <code>HTMLEncode</code> - Replaces HTML-sensitive characters with equivalent HTML character entities.

Item	Description
	<p><i>Utils</i></p> <ul style="list-style-type: none"> ■ generateUUID - Generates a random Universally Unique Identifier (UUID).
Help navigation and user interface changes	Page titles and help icons have now been removed from the specific user interface pages. To access context-sensitive help information, from the navigation bar, click on the help  icon.
Recipes page enhancements	<p><i>Pagination</i></p> <p>The Recipes page is now paginated to identify the sequential order of the pages. You can also select the number of recipes to be viewed per page.</p>  <p><i>Search</i></p> <p>You can now search Recipes by Application names.</p>
Registration page changes	The field name of the email address provided during registration has been changed to <i>Work Email Address</i> .
Submit option is deprecated while executing Integrations from an external system	You can use the <i>run</i> option in the Request URL while executing an Integration from an external system, that is, if the Integration is enabled to be invoked over HTTP.

Version 4.5.0 (April 2018)

This section describes the enhancements and changes made in Version 4.5.0:

Item	Description
Create REST APIs	<p>Integration Cloud allows you to write integration logic to integrate different types of applications. This logic can be exposed to the external world using REST APIs.</p> <p>These REST APIs can be created by using an existing set of Integrations (from scratch) or by using a file containing the Open API specification</p>

Item	Description
	<p>(formerly known as the Swagger specification) as a template.</p> <p>A REST API consists of many Resource Operations and each Resource Operation has a Path, one or more HTTP Methods, and an associated Integration.</p> <p>A REST Resource Operation can be tried out from the Swagger screen of a REST API. When the Resource Operation is invoked using the HTTP Method, the associated Integration gets executed.</p>
Application enhancements	<p>Following are the Application enhancements:</p> <ul style="list-style-type: none"> ■ Some Applications, which includes custom REST Applications, now allow two-way SSL authentication by providing keystore and truststore aliases in the Account Configuration section. ■ The Account configuration field <i>Use Chunking</i> has been added in many Applications. ■ For custom REST Applications, <i>binary</i> has been added as the content type for the Request/Response body. Binary data can be sent as an input to a REST operation.
Refresh SOAP Applications	<p>You can now update a SOAP Application by specifying a new WSDL URL or by uploading a new WSDL file.</p>
New Blocks and Expressions in Orchestrated Integrations	<p>Following are the new and modified blocks and expressions:</p> <p>New Blocks</p> <ul style="list-style-type: none"> ■ <i>switch</i> block in <i>Control Flow</i> category. The <i>switch</i> block can be mutated for multiple <i>cases</i> and one <i>default</i>. ■ <i>Throw error "..."</i> block in <i>Control Flow</i> category. ■ <i>Field exists</i> expression in <i>Expressions</i> category. <p>Modified Blocks</p> <ul style="list-style-type: none"> ■ <i>if</i> block has been enhanced and can now be mutated.

Item	Description
	<ul style="list-style-type: none"> ■ <i>if else</i> block has been deprecated and is not available from this release but Integrations currently using the <i>if else</i> block will continue to work successfully. ■ <i>Exit Integration</i> has been renamed to <i>Exit Integration signaling success</i> and <i>Exit Integration with failure</i> has been renamed to <i>Exit Integration signaling failure</i>.
Ability to delete assets used by an Integration	<p>You can now delete custom Applications, Accounts, Operations, Integrations, Reference Data, and Document Types assets available in the <i>Development</i> stage, even if those assets are referenced by other assets. Note that if assets used by an Integration are deleted, you will not be able to pull the Integration into subsequent stages or export the Integration. After deleting an asset, the deleted asset reference is highlighted () in the user interface.</p>
Ability to delete users	<p>You can now delete users provided you have the <i>User Management</i> permission. Note that you cannot delete your own user profile. If a user is deleted, then the user cannot be recovered and all assets created or modified by the user will appear in the <i>Created By</i> and <i>Modified By</i> columns as <i>Unknown User{first two characters of the first name and last name}</i>.</p>
New Services	<p>Following new services have been added under the <i>IO</i> category:</p> <ul style="list-style-type: none"> ■ <i>stringtoStream</i> - Converts a string to a binary stream. ■ <i>streamToString</i> - Creates a string from data that is read from an <i>InputStream</i>. <p>The following new services have been added under the <i>Flow</i> category:</p> <ul style="list-style-type: none"> ■ <i>clearPipeline</i> - Removes all fields from the pipeline. ■ <i>getHTTPRequest</i> - Gets information about the HTTP request, received by Integration Cloud.

Item	Description
	<ul style="list-style-type: none"> ■ <i>setHTTPResponse</i> - Sets the HTTP response information to be returned by Integration Cloud.
Terminate in-progress Integration executions	<p>You can now click the <i>Terminate</i> option available in the Execution Results details page to terminate <i>in-progress</i> Integration executions provided you have the Execute Integration permission. The <i>Terminate</i> audit log entry is created.</p>
New Application	<p>A new Application, <i>Anaplan</i>, has been added in this release. Anaplan allows you to interact with data in your models and securely upload files, download files, import and export data, and run actions programmatically.</p>
Viewing assets and services in various stages	<p>Integration Cloud now displays the current active stage on the navigation bar.</p>  <p>The Stage in View can be changed in the Stages > Change Stage To View page. You can view assets and services in each stage and <i>view only those assets that are available in the selected stage</i>.</p>
API Management permission	<p>The Manage Promotions permission has been added under Functional Controls and allows you to add, modify, and delete API Gateway stages, or move API Gateway assets from the source stage to one or more target stages, or to rollback an asset promotion that is already available in the target stage at any time.</p>
Redesigned user interface	<p>The user interface has been redesigned and has a new look and feel in this release. The functions that can be accomplished have been reorganized for improved usability.</p> <ul style="list-style-type: none"> ■ <i>Settings</i> menu can be accessed by clicking on the  icon. ■ <i>User Profile, My Profile, and Logout</i> menu items can be accessed by clicking on the  icon.

Item	Description
	<ul style="list-style-type: none"> <li data-bbox="669 323 1308 426">■ <i>Help Topics</i>, <i>TECHcommunity</i> website, and the <i>About</i> page can be accessed by clicking on the  icon. <li data-bbox="669 447 1297 510">■ <i>Applications</i> and <i>Keys & Certificates</i> have been moved under the <i>Connect</i> menu. <li data-bbox="669 531 1325 667">■ <i>Stages</i> tab has been newly added in this release and contains <i>Change Stage To View</i>, <i>Deploy</i>, and <i>Manage</i>. The <i>Deploy</i> option is also available on the <i>Integrations</i> page. <li data-bbox="669 688 1268 720">■ <i>Recipes</i> tab has been moved under <i>Develop</i>. <li data-bbox="669 741 1320 877">■ The <i>Pull</i> functionality has been moved from the <i>Integration Details</i> page to the <i>Deploy Assets (Stages > Deploy)</i> page. This page allows you to move assets from one stage to another stage. <li data-bbox="669 898 1287 1001">■ Stage selection drop-down list box has been removed from the <i>Dashboard</i> and <i>Execution Results</i> pages.

Version 4.1.0 (January 2018)

This section describes the enhancements and changes made in Version 4.1.0:

Item	Description
Export and Import Integrations	Integration Cloud now allows you to export Integrations from one tenant and import those Integrations to another tenant. You can import only those Integrations that are exported from Integration Cloud. You must have the <i>Export</i> permission to export Integrations.
Home page enhancements	The Home page has been redesigned and has a new look and feel in this release. The functions that can be accomplished from the Home page have been reorganized for improved usability.
Customizing Integration Cloud	You can now customize the Integration Cloud user interface by changing the logo, colors, font face, copyright information, and the <i>About</i> page contents.

Item	Description
	The behavior or functionality of Integration Cloud cannot be changed.
Email opt-in during new tenant registration	The <i>Email opt-in</i> field is now added that will enable you to receive marketing materials from Software AG. You can unsubscribe from the updates at any time.
SSO login capability	If you have already configured SSO, the <i>SSO Login</i> option now appears in the login page. Clicking this option redirects you to the Identity Provider (IdP) login page. After you provide the IdP login credentials, you will be logged into Integration Cloud.
Redesigned test results and Orchestration layouts	The test results panel has been redesigned to provide a larger work space in this release. Also, the <i>Cancel</i> button in the Orchestrated Integration page has been changed to <i>Exit</i> and the Help icon has been moved to the lower left-corner of the page.
Logged fields for business data	You can now select any number of fields to log business data from the Operation and Integration signatures.
New Applications	The following Applications are added in this release: <ul style="list-style-type: none"> ■ DocuSign ■ Amazon DynamoDB
New Services	The new <i>Storage</i> block is added in this release and the following services are added under the Storage block: <i>add</i> , <i>get</i> , <i>keys</i> , <i>lock</i> , <i>put</i> , <i>remove</i> , and <i>unlock</i> . The <i>objectToString</i> service is added under the String block and it is also available as a transformer service.
New Recipes	39 new Recipes integrating Coupa, NetSuite, ServiceNow, Salesforce, Amazon S3, FTP server, Jira, Sugar CRM, Concur, and Zendesk have been added in this release.
Prevent concurrent executions	You can now skip the next scheduled Integration execution if the previous scheduled execution is still

Item	Description
	running by selecting the <i>Prevent concurrent executions</i> option in the Schedule Execution window.

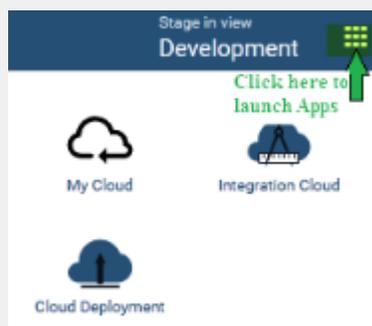
1 Registration

- Registration Overview 42

Registration Overview

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

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



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

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

Creating an Account

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

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

To create a new User Account

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

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

Field	Description
First Name	Type your first name.

Field	Description
	You can change the value after the user is created from the Settings  > Users screen.
Last Name	Type your last name. You can change the value after the user is created from the Settings  > Users screen.
Company	Type your company name. You can change that later from the Settings  > Company Information screen.
Country	Select your country from the drop-down list box. You can change that later from the Settings  > Users screen.
State or Province	Type your State or Province. You can change that later from the Settings  > Users screen.
Phone	Type your phone number. You can change that later from the Settings  > Users screen.
Your Role	Select your role in your current organization from the drop-down list box.
Company Size	Select the number of employees range in your current organization from the drop-down list box.
Is your interest based on	Select at what stage is your current project from the drop-down list box.
Sub-Domain	Provide a unique sub-domain, typically your company name. For example, suppose you are at ABC Company and you decide to use “abc” as your unique sub-domain. With that setting, you will access your instance of the platform at https://abc.webmethodscloud.com.

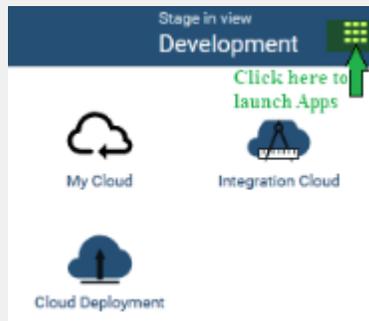
Note: You must log in with the correct sub-domain. Some functionalities may not work properly if you log in with a generic sub-domain.

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

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

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

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



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

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

Securing your Account

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

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

To secure your account

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

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

2 Settings

■ Users	48
■ Access Profiles	55
■ Access Control Lists	61
■ Single Sign-On	63
■ Company Information	70
■ Password Policy	72
■ Client Certificate	73
■ OAuth 2.0	76

Users

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

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

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

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

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

Adding Users

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

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

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

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

1. From the Integration Cloud navigation bar, go to **Settings**  > **Users**.
2. From the upper right part of the Users screen, click **Add New User**.
3. On the **Basic** tab, complete the following fields. Required fields are marked with an asterisk on the screen.

Field	Description
First name	User's first name as it should appear in the platform.
Last name	User's last name as it should appear in the platform.
Title	User's professional title.
Access Profile	<p>The access profile assigned to the User. Each User is assigned an access profile, which can be shared by other users. An Access Profile specifies the network locations (IP addresses) from where it is possible to login and administrative permissions. Select one of the following Access Profiles:</p> <ul style="list-style-type: none"> ■ Administrator - Provides permissions needed by the System Administrator. ■ Regular User - Provides permissions that are more appropriate for normal users. <p>By default, the system administrator can change the Administrative Permissions associated with each Access Profile (except the above mentioned Administrator Access Profile), and can add additional Access Profiles, as needed.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p>Note: By default, the Administrator and Regular User Access Profiles are associated with the Development Stage. If you have created a new Access Profile, ensure that the Access Profile you have created is associated with the Development Stage. See Adding or Updating Access Profiles for more information and for information on API Management Access Profiles and permissions.</p> </div>
Employee Number	Optional identification number for each employee.
Email	Email address of the user. User credentials will be sent to the specified email address.

Field	Description
	As soon as you add a new user, two different emails will be sent to the email address. One email will contain the user ID and the other email will contain the temporary password. Use the temporary password to log in. You will be asked to change your password.
User Name	User name is a unique name associated with each user and is required to log in. It can be an email address or an alphanumeric text string. Note: If you are a Software AG Cloud user, you will not be able to update the User Name .
Federation ID	Enter the Federation ID if your Identity Provider passes the Federation ID for Single Sign-On . See the "Single Sign-On" Help page for more information. The Federation ID acts as a user's authentication across multiple IT systems or organizations. A federated identity means linking a person's electronic identity and attributes stored across multiple distinct identity management systems.
Partner	Select this option if the user is a Partner user. If Allow User Interface Access permission available under Access Profile > Administrative Permissions > Account Controls is not enabled, a Partner User can still perform on-premises tasks.
Active	Select this option to indicate that the user account is active. You can use this option to reactivate a locked or disabled user account.

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

Field	Description
Time Zone	Choose a Time Zone Code from the drop down list.
Date Format	Choose a Date Format from the drop down list. "mm" is "Month", "dd" is "Day", and "yyyy" is Year. Dates and Times are used throughout the platform, in Appointments, as Start/End Dates in Tasks, Expected Close Date, Estimated Start/End Date, Date Due, and so on. Default formats are specified under the Settings  > Company

Field	Description
	Information > Advanced Information tab. Administrators and Users can change the default selection in the Users screen.
Locale	Select the user's locale setting. This setting determines the format for numbers, decimal fields, and percentages.
Time Format	Select a 12-hour clock (hh:mm a) with AM/PM, or a 24-hour clock (HH:mm).

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

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

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

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

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

Updating Users

To edit or update the user information

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

1. From the Integration Cloud navigation bar, click **Settings**  > **Users**.
2. Select a user from the list, and then click **Edit**.
3. Make necessary modifications. See [“Adding Users” on page 48](#) for information on the relevant fields. You can also enter or update the following information on the **Address and Contact** tab. Required fields are marked with an asterisk on the screen.

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

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

4. Click **Apply**.

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

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

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

Resetting Passwords

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

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

To reset a user's password

1. From the Integration Cloud navigation bar, go to **Settings  > Users**.
2. For the user whose password is to be reset, select the user and click **Reset Password**.

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

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

User Profile

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

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

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

Click **Edit** to update the information.

Security Question

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

To update the Security Question and Answer

1. From the Integration Cloud navigation bar, go to  > **My Profile** > **Security Question**.
2. Select a **Security Question** and type a **Security Answer**. You can change the **Security Question** associated with your Account Login/Password.
3. Click **Submit**.

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

Change Password

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

To change your password

1. From the Integration Cloud navigation bar, go to  > **My Profile** > **Change Password**.
2. Type your current password in the **Old Password** field, your new password in the **New Password** field, and again retype your new password in the **Retype New Password** field.
3. Click **Submit**. You will receive a confirmation email about your changed password.

My Certificate

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

To view the certificate

1. From the Integration Cloud navigation bar, click  > **My Profile** > **My Certificate**.
2. If a certificate is configured for the user, the **View Certificate** panel displays the configured certificate. You can click **Download** to download the user certificate or click **Delete** to delete the user certificate. The downloaded file is named as <username>.crt.
3. In the **Upload New Certificate** field, click **Browse** to upload a new client certificate signed by a trusted Certificate Authority (CA).
4. In the **Generate Private Key and Certificate** field, click **Generate** if you want Integration Cloud to generate a private key and a new Integration Cloud signed client certificate.

Integration Cloud validates it against the issuer of the certificate. The generated certificate is named as <username>.txt.

Access Profiles

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

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

An Access Profile specifies:

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

The default Access Profiles are:

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

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

- API Gateway Administrators - By default, all API Management permissions are assigned to the Administrators access profile and these privileges cannot be modified.
- API Gateway Providers - By default, the following permissions are assigned to the API Gateway Providers access profile and these privileges cannot be modified:
 - Manage APIs
 - Manage Applications
 - Manage policy templates
 - Manage packages and plans
 - Publish to API Portal
 - Export assets
 - Execute service result cache APIs

- Activate/Deactivate APIs/Packages
- Manage aliases
- Import assets
- API Portal Administrators - The API Portal Administrator can perform all the functions in API Portal.
- API Portal Providers - The API Portal Provider can manage APIs and packages in API Portal.

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

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

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

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

Adding or Updating Access Profiles

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

To add or update an Access Profile

1. From the Integration Cloud navigation bar, go to **Settings**  > **Access Profiles**.
2. Click **Add New Access Profile** to add a custom access profile or click **Edit** to modify an existing Access Profile.
3. On the **Add New Access Profile** or **Update Access Profile** > **Access Profile Information** tab, complete the following fields. Required fields are marked with an asterisk on the screen.

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

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

Field	Description
IP Address Ranges	<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>Note: A maximum of 25 IP address ranges can be specified. You can add, modify, and delete the entries. Accepted format is xxx.xxx.xxx.xxx - yyy.yyy.yyy.yyy, where xxx and yyy are numbers in the range 0-255 and xxx.xxx.xxx.xxx is less than or equal to yyy.yyy.yyy.yyy. To specify a single IP address, use the same IP address for the start and endpoint of the range: 192.168.1.1 - 192.168.1.1</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** page, select the operations a user can perform in order to access, view, create, update, upgrade, administer, execute, export, deploy, and delete and to allow the user to customize selected aspects of the platform.

Field	Description
User and Ownership Controls	<p>User Management - Select this option if you want to add, update, delete users, or assign users to Access Profiles.</p> <p>Access Control - Select this option if you want to allow a user to modify Access Profiles, edit ACLs, specify user application access rights, manage Access Profiles, specify the password policy, create, edit, and delete OAuth 2.0 clients and scopes, and delete OAuth 2.0 tokens.</p> <p>Manage Personal Setup - Select this option if you want to allow a user to modify the personal information, and generate or edit the user's own certificate.</p>
Account Controls	<p>Manage Company Capabilities - Select this option if you want to allow users to modify the company information.</p>

Field	Description
	<p>Allow User Interface Access - Select this option if you want to allow users to log in to Integration Cloud and access the user interface. Clear this option if you want to deny users to access the user interface. Further, even if you clear this option, all users can still interact with Integration Cloud using REST interface calls.</p> <p>Note: If the Allow User Interface Access permission is not enabled for a user but if the user is a Partner user, that user will still be able to perform on-premises tasks.</p>
Data Management Controls	<p>Manage Audit Log - Select this option if you want to allow users to view the Audit Log. If this option is enabled, the Audit Log page will be displayed. If not selected, the user will not be able to view the Audit log page. To view the Audit Log screen, from the Integration Cloud navigation bar, click Monitor > Audit Log.</p>
Functional Controls	<p>Select the required options under Assets, Stages, Advanced Security, Application, Accounts, Operations, Reference Data, Document Type, Integrations, REST APIs, Container, and Solution. You must select the required permissions to administer, access, create, update, upgrade, delete, execute, export, restart, resume, or deploy those functions.</p> <p>Note: The Container tab and Container related Administrative permissions are available only if you have the required license for Containers.</p>
	<p>6. On the Container page, enter the names of the webMethods Integration Server Access Control List (ACL) groups separated by a comma, for example, Administrators, Developers, and so on. Users who are assigned to this Access Profile will be now part of the webMethods Integration Server container user group (s) and can perform tasks allowed for those user groups. If you do not map an Access Profile to an webMethods Integration Server group, you will not be able to invoke webMethods Integration Server services. For information about user groups, see the <i>Managing Users and Groups</i> section in the <i>webMethods Integration Server Administrator's Guide</i>.</p> <p>Note: Integration Cloud Administrator profiles are <i>not automatically</i> assigned to the webMethods Integration Server Administrators ACL group. If you do not enter any user groups in the Container User Groups field, but have configured webMethods Integration Server in a way such that it needs to verify the ACL groups you have entered in the Container User Groups field while invoking services, you will not be able to run or invoke webMethods Integration Server services from Integration Cloud.</p>
	<p>7. The API Management tab displays the API management permissions.</p>

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

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

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

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

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

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

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

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

- Administrators ACL
- Developers ACL
- Replicators ACL

- Click **Apply**.

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

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

Access Control Lists

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

Example 1

You have three users U1, U2, and U3. U1 is assigned to Access Profile AP1, U2 is assigned to Access Profile AP2, and U3 is assigned to Access Profile AP3. Each user has the Integration execution permission. There are also four Integrations IN1, IN2, IN3, and IN4 in your tenancy. Initially, U1, U2, and U3 can run all the four Integrations IN1,

IN2, IN3, and IN4. Now you want IN1 to be executed only by U1 and *not* by U2 and U3. To do that, create an Access Control List, ACL1. Associate ACL1 to IN1. Then associate ACL1 to AP1. As U1 has already been assigned to AP1, IN1 can be executed by *only* U1. If you want IN1 to be executed also by U2, then associate ACL1 with AP2.

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

Example 2

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

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

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

Adding or Updating Access Control Lists

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

To add or update an ACL

1. From the Integration Cloud navigation bar, go to **Settings**  **> Access Control Lists**.
2. Click **Add New Access Control List** to add an ACL or click **Edit** to modify an existing ACL.
3. On the **Add New Access Control List** or **Update Access Control List** page, complete the following fields. Required fields are marked with an asterisk on the screen.

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

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

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

5. Click **Apply**.

Single Sign-On

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

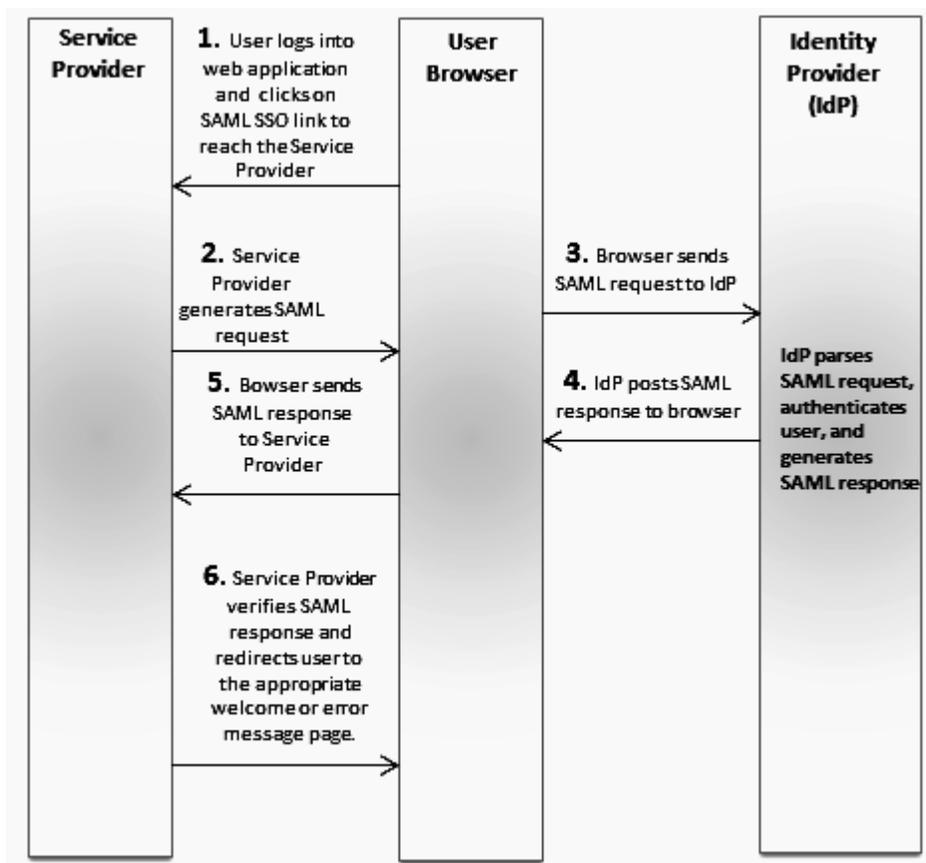
Single sign-on is a process that allows users to access all authorized network resources without having to log in separately to each resource.

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

Note: Currently, only SAML 2.0 is supported.

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

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



Service Provider: Integration Cloud

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

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

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

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

See [Configuring SAML Settings for Single Sign-On](#) on how to configure SAML settings for single sign-on.

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

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

Configuring SAML Settings for Single Sign-On

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

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

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

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

To configure SAML 2.0 settings for single sign-on

1. From the Integration Cloud navigation bar, click **Settings**  **> Single Sign-On**.
2. Click **Edit**.
3. On the **Update Single Sign-On Configuration** screen, select **SAML 2.0** in the **Sign-On Using** field and make the necessary modifications. Required fields are marked with an asterisk on the screen.

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

Field	Description
	<p>For Microsoft Azure, send the RelayState value to Microsoft Azure AD to configure the RelayState for your application instance. See Microsoft Azure website for more information.</p> <p>For Okta, apply it in the Default RelayState field.</p>
Identity Provider Configuration	
SAML Request Issuer URL	<p>This is the Integration Cloud (Service Provider) URL used to access this tenant. This URL acts as the Service Provider ID.</p> <p>For Oracle Access Manager (OAM), apply it in the Provider ID field.</p> <p>For Microsoft Azure, apply it in the Identifier field.</p> <p>For Okta, apply it in the Audience URI (SP Entity ID) field.</p>
Identity Provider Details	<p>Specify how you want to define the Identity Provider details.</p> <p>Select Enter Manually if you want to manually enter the URL that uniquely identifies Integration Cloud in your SAML Identity Provider in the Issuer field.</p> <p>Select Load From Identity Provider Metadata and select the metadata file to upload the IdP details.</p>
Issuer	<p>A URL that uniquely identifies Integration Cloud in your SAML Identity Provider. Integration Cloud (Service Provider) must enroll with an Identity Provider and obtain an Issuer URL.</p> <p>If you have selected Enter Manually for Identity Provider Details, copy the URL provided by the IdP here after setting up Integration Cloud configuration in the IdP.</p> <p>If you have selected Load From Identity Provider Metadata for Identity Provider Details and uploaded the IdP file, the Issuer field will be automatically populated.</p> <p>For Microsoft Azure, copy the URL from the Issuer URL field.</p> <p>For Oracle Access Manager (OAM), copy the URL from the Provider Id field under Federation Settings.</p> <p>For Okta, copy the URL from the Identity Provider Issuer field.</p>

Field	Description
Identity Provider Certificate	<p>This is the authentication certificate (a valid x509 issuer certificate) issued by your Identity Provider and is required to sign and verify SAML messages.</p> <p>If you have selected Enter Manually for Identity Provider Details, select Browse and upload a file that contains the Identity Provider's certificate.</p> <p>If you have selected Load From Identity Provider Metadata for Identity Provider Details and uploaded the IdP file, the IdP certificate will be automatically uploaded.</p>
Identity Provider Login URL	<p>This is the URL used to log in to the Identity Provider.</p> <p>If you have selected Enter Manually for Identity Provider Details, type the URL that will be used to log in to the Identity Provider.</p> <p>If you have selected Load From Identity Provider Metadata for Identity Provider Details and uploaded the IdP file, the IdP login URL will be automatically populated.</p> <p>For Oracle Access Manager (OAM), the URL is <i>http://<oamserverhost name>:14100/oamfed/idp/samlv20</i>.</p> <p>For Microsoft Azure, copy the URL from the Single sign-on service URL field.</p> <p>For Okta, copy the URL from the Identity Provider Single Sign-On URL field.</p>
User ID Type	<p>Determines the type of identifier.</p> <p>Assertion contains user's Integration Cloud username - Select this option if your Identity Provider passes the username ( > <i>User Profile</i> > Basic tab) in the SAML assertion to identify the user.</p> <p>Assertion contains the Federation ID from the User Object - The Federation ID acts as a user's authentication across multiple IT systems or organizations. A federated identity means linking a person's electronic identity and attributes stored across multiple distinct identity management systems. Select this option if your Identity Provider passes the Federation ID ( > <i>User Profile</i> > Basic tab), to identify the user. You can add the Federation ID ( > <i>User Profile</i> > Basic tab) to each user's profile after you have configured single sign-on.</p>

Field	Description
User ID Location	<p>Specifies an attribute tag that defines the location of the User ID. This is the location in the assertion where a user should be identified.</p> <p>Select Subject if the User ID is located in the <Subject> statement of the assertion.</p> <p>Select Attribute if the User ID is specified in an <AttributeValue>, located in the <Attribute> of the assertion. If you have selected Attribute, specify the attribute that contains the User ID in the Attribute for User ID field. If the User ID attribute is empty or does not match an existing user, then either login fails or a new user is created, depending on the Create Users setting.</p>
Attribute for User ID	<p>This field appears if you have selected Attribute in the User ID Location field. Specify the attribute that contains the User ID. If the User ID attribute is empty or does not match an existing user, then either login fails or a new user is created, depending on the Create Users setting.</p>
Create Users	<p>Select this option to create a new user when the User ID is not recognized. When selected, additional options appear where you can specify the attribute to use for the First Name, Last Name, Email, and Access Profile.</p> <p>Attribute for First Name - The name of the SAML attribute that designates the user's first name.</p> <p>Attribute for Last Name - The name of the SAML attribute that designates the user's last name.</p> <p>Attribute for Email - The name of the SAML attribute that designates the user's email address.</p> <p>Default Access Profile - This field is used to specify the default Access Profile for the created user.</p> <p>Attribute for Access Profile - The name of the SAML attribute that designates the user's access profile. The attribute must contain the ID of the Access Profile. You can get the ID of the Access Profile from the Access Profiles screen (Settings  > Access Profiles).</p>

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

Company Information

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

See “[Updating Company Information](#)” on page 70 for information on the fields.

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

Updating Company Information

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

To update the Company Information

1. From the Integration Cloud navigation bar, go to **Settings**  > **Company Information**.
2. Click **Edit**.
3. On the **Basic** tab, complete the following fields. Required fields are marked with an asterisk on the screen.

Field	Description
Tenant ID	<p>This is the unique ID assigned to your organization's tenancy on the platform.</p> <p>Note: This field cannot be edited and appears in view only mode under the Basic tab.</p>
Sub Domain	<p>This is the unique sub domain that you specified during registration. A sub domain is a domain that is part of a main domain. For example, suppose you are at ABC Company and you decide to use “abc” as your unique sub domain. With that setting, you will access your instance of the platform at https://abc.webmethodscloud.com.</p> <p>Note: This field cannot be edited and appears in view only mode under the Basic tab.</p>
Company Name	<p>The name of the company. This field accepts only alphanumerics, spaces, and hyphens (-). The company name is automatically populated from the Registration screen.</p>

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

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

Field	Description
Time Zone	Choose your time zone from the drop down list.
Time Format	Choose a time format from the drop down list. You can choose a 12-hour clock with AM/PM or a 24-hour clock. hh:mm a - 12-hour clock - 3:30 AM, 3:30 PM HH:mm - 24-hour clock - 3:30, 15:30
Date Format	Choose a date format from the drop down list. mm is "Month", dd is "Day", yyyy is Year and the delimiters are:(/) slash or stroke(-) dash or hyphen(.) period, dot, or full stop.
Default Locale	The field displays the user's initial locale setting and determines the format for numbers, decimal fields, and percentages. Choose any other locale from the drop down list.
Last Modified	This field displays the date and time when the company information record was last updated. This field cannot be edited and appears in view only mode .

Password Policy

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

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

You can view the password policies for the Integration Cloud instance in this screen. See [“Updating Password Policy Settings” on page 72](#) for information on the fields.

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

Updating Password Policy Settings

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

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

To update the Password Policy

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

Field	Description
Minimum Length	Select the minimum number of characters in the password.
Required Character types	This option defines the level of security for passwords, which can be simple and allow any character combination, or very secure, requiring upper and lower case characters, as well as special characters.
Expires in	Select the number of days the password will remain valid before the user will be prompted to change it. 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> .

Field	Description
Password Never Expires for	Select the users for whom the password will never expire. Only active users appear in the list. You can make an user account active by selecting the Settings  > Users > Update User > Basic tab > Active option.
New Password cannot match	The new password cannot match the number of previous passwords.
Minimum Age	Select the number of days that must pass before a user can change passwords.
Session Timeout	Select the length of time the session will remain active without any user activity. The session will end when it reaches the selected timeout. The user will need to log in again.
Account Lockout Threshold	Select the number of login attempts before the account is locked out. 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> .
Account Lockout Duration	Select the length of time that an account is locked out.
Record Information	For audit purposes, the following information is displayed after you save the record: <i>Last Modified By <username> on {date} <time> Created by System.</i>

- Click **Apply**.

Client Certificate

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

In a *one-way* SSL connection, an anonymous client authenticates the credentials of a server in preparation for setting up a secure transaction. In most cases, the server

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

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

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

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

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

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

Task	Activities	Notes
Create keys and certificates	<ul style="list-style-type: none"> ■ Generate a public key/private key pair. ■ Generate a certificate signing request (CSR) and send it to the certificate authority (CA) for signing. ■ Receive validated certificate from the CA. 	Two-way SSL connection requires a valid client certificate.

Task	Activities	Notes
Upload client certificate or generate a certificate	Upload the CA signed client certificate for the user in the Client Certificate page or generate a private key and a new Integration Cloud signed client certificate.	Required for two-way SSL connections.
Connect to Integration Cloud using the client certificate.	Configure the REST client with the private key and certificate. Optionally you can also pass the basic authentication credentials.	Integration Cloud support two-way SSL on port 8443.

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

See [“Adding Client Certificates” on page 75](#) for information on how to add client certificates.

Adding Client Certificates

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

To add a client certificate

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

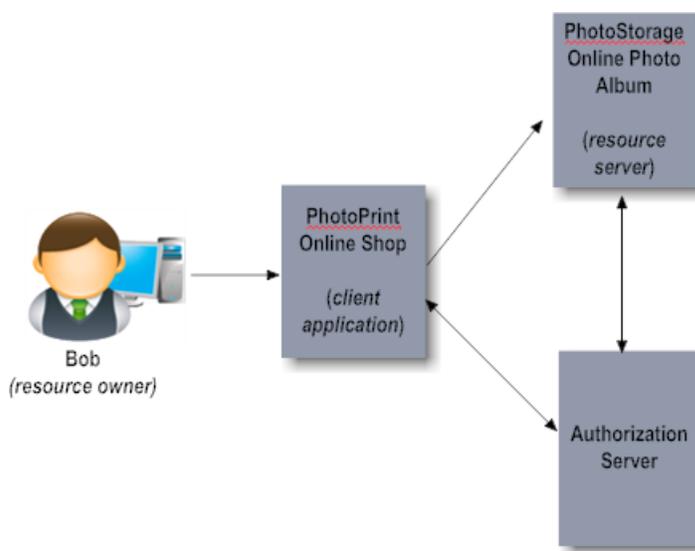
OAuth 2.0

About OAuth 2.0

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

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

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



Integration Cloud services can be accessed through REST APIs from any REST client. In OAuth 2.0, the client obtains an access token issued by an authorization server on

approval of the resource owner. The client uses the access token to access the protected resources.

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

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

Configuring OAuth 2.0

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

Steps	Description
“Define Clients” on page 78	Define the clients that are authorized to invoke services in Integration Cloud. Specify the client name, version number of the client, client type, redirection URLs, allowed grants, expiration interval, and the refresh count. See “Registering Clients” on page 78 for more information.
“Define Scopes” on page 81	A scope defines the services the client can access on behalf of the resource owner. A scope consists of a name and one or more services. If access is granted for a scope, then access is granted for all the services in that scope.
“Associate scopes to a client” on page 78	Associate defined scopes with the registered clients. When you associate scopes, you authorize the scopes that each client can access.
“Generate Tokens” on page 82	Generate tokens (Access Token and Refresh Token) by using a REST Client. See “Generating Tokens” on page 82 for more information. Integration Cloud supports the Authorization Code Grant, Implicit Grant, Client Credentials Grant, and the Resource Owner Password Credentials Grant to generate the access tokens. Clients use the tokens to execute REST URLs for running the Integrations. After you generate the tokens, the

Steps	Description
	tokens are available in the Token Management page in Integration Cloud.

Registering Clients

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

See [“About OAuth 2.0” on page 76](#) for information on the high-level steps for configuring OAuth 2.0.

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

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

To add a Client

1. From the Integration Cloud navigation bar, go to **Settings**  **> OAuth 2.0 > Client Registration > Add New Client**.
2. On the **Add New Client** page, complete the following fields. Required fields are marked with an asterisk on the screen.

Field	Description
Name	Type the name of the client. You cannot create clients with the same Name and Version combination. You cannot modify the client name after the client is saved. Client names are not case-sensitive.
Description	Type a description of the client.
Client ID	The Client ID field appears only when you update a client. This is a client identifier issued to the client to identify itself to the

<u>Field</u>	<u>Description</u>
	authorization server, and is used while generating tokens.
Client Secret	The Client Secret field appears only when you update a client. This is a secret matching to the client identifier and is used while generating tokens. It will not be generated if the Client Type is Public .
Authorization Endpoint	View the authorization URL that has to be provided while generating tokens. See the <i>Generating Tokens</i> section for more information.
Token Endpoint	View the Access Token URL that has to be provided while generating tokens. See the <i>Generating Tokens</i> section for more information.
Refresh Token Endpoint	View the Refresh Token URL that has to be provided while refreshing Access Tokens. See the <i>Refreshing Access Tokens Using Refresh Tokens</i> section for more information.
Version	Type the version number of the client. You cannot create clients with the same Name and Version combination.
Type	Select the type of the client according to its ability to communicate with Integration Cloud. Confidential - Select Confidential when the OAuth session uses the Authorization Code Grant . This client is capable of maintaining secure client authentications. When you select client type as Confidential , Integration Cloud generates a client secret. This client secret will be required by Integration Cloud when the client makes requests to the OAuth services. Public - Select Public when the OAuth session uses the Implicit Grant type. This client is

Field	Description
	not capable of maintaining secure client authentications.
Redirection URLs	<p>Specify the URLs that Integration Cloud will use to redirect the resource owner's browser during the grant process.</p> <p>You can add more than one redirection URL.</p> <p>If you select the Authorization Code Grant or the Implicit Grant types, you must enter at least one Redirection URL for the client.</p>
Allowed Grants	Select the type of grant flow required by the client.
Expiration Interval	<p>Select the length of time (in seconds) that the access token is valid.</p> <p>Never Expires - Indicates that the access token never expires. The Token Management page displays Lifetime for that token.</p> <p>Expires In - Specify the number of seconds the access token is valid.</p>
Refresh Count	<p>Select the number of times the access token can be refreshed.</p> <p>Unlimited - Refresh the access token an unlimited number of times using the refresh token. The Token Management page displays Unlimited for that refresh token.</p> <p>Limited - Specify the number of times to refresh the access token. The Token Management page will display the Refresh Count for that refresh token. If you specify 0 or leave the field empty, a refresh token will not be issued.</p> <div data-bbox="748 1644 1360 1738"><p>Note: Tokens can be refreshed only when using the Authorization Code Grant flow.</p></div>
Active	This option appears only when you update a client. Clear this option to deactivate the client. When you deactivate a client, all the

Field	Description
	access tokens and refresh tokens for the client become invalid.

3. Click **Add** to add the client in the **Client Registration** page.
4. On the **Client Registration** page, if you want to associate scopes with a client, select a client and then click **Associate Scopes**. The **Associate Scopes with <ClientName(Version)>** page appears. The **Associate Scopes with <ClientName(Version)>** page displays the already associated scopes with the selected client.
 - a. On the **Associate Scopes with <ClientName(Version)>** page, to associate existing scopes with the client, select **Associate Existing Scopes**.
 - b. On the **Select Scopes to Associate** dialog box, select the existing scopes to associate with the client and then select **Associate Scopes**. The newly associated scopes will appear in the **Associate Scopes with <ClientName(Version)>** page.
 - c. To create a new scope and associate it with the selected client, select **Associate New Scope**. Create the new scope as described in the [“Managing Scopes” on page 81](#) section. The new scope will be associated with the selected client.
 - d. To disassociate a scope from a client, select the scope on the **Associate Scopes with <ClientName(Version)>** page and then click **Disassociate**.

Managing Scopes

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

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

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

See [“About OAuth 2.0” on page 76](#) for information on the high-level steps for configuring OAuth 2.0.

To add a scope

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

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

Field	Description
Name	Type a unique name for the scope. You cannot modify the scope name after a scope is saved. Scope names are not case-sensitive.
Description	Type a description of the scope.
Services	Specify the list of services that the client can access on behalf of the resource owner for executing the integrations. Click Select Services and in the Services dialog box, select the exposed Integrations or REST Resources that you want to add as Service URLs. <div data-bbox="748 884 1365 1230" style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Note: Service URL is a relative URL and it must start with <i>/integration</i>. For example, if the absolute URL is <code>https://<sub-domain>.webmethodscloud.com/integration/rest/external/integration/run/development/restintegration</code>, then the Service URL is <code>/integration/rest/external/integration/run/development/restintegration</code>.</p> </div>
Service URLs	This field appears once you have added the exposed Integrations and REST Resources and shows the selected services.

- See the [“Registering Clients” on page 78](#) section on how to associate scopes with a client.

Generating Tokens

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

See [“About OAuth 2.0” on page 76](#) for information on the high-level steps for configuring OAuth 2.0.

The following example shows how to generate tokens using Postman

1. Add the Postman extension to your Google Chrome web browser.
2. Open the Postman application.
3. On the Postman **Authorization** page, select the **Type** as **OAuth 2.0**, and then click **Get New Access Token**. The **Get New Access Token** page appears.

The screenshot shows the Postman interface with the 'Authorization' tab selected. The 'Type' dropdown is set to 'OAuth 2.0'. Below this, there is a section for 'Existing Tokens' with a 'Get New Access Token' button. To the right, there is a 'Token Details' section with the instruction 'Select a token from the list to view details'.

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

The screenshot shows the 'GET NEW ACCESS TOKEN' dialog box. It contains the following fields and options:

- Callback URL:** `https://www.getpostman.com/oauth2/callback` (with a note: 'Set this as the callback URL in your app settings page.')
- Token Name:** Text input field with placeholder 'Token Name'.
- Auth URL:** Text input field.
- Access Token URL:** Text input field.
- Client ID:** Text input field.
- Client Secret:** Text input field.
- Scope (Optional):** Text input field.
- Grant Type:** Dropdown menu set to 'Authorization Code'.
- Request access token locally**

At the bottom, there are two buttons: 'Cancel' and 'Request Token'.

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

5. Click **Request Token**.

Integration Cloud login page appears.

6. Login to Integration Cloud with your credentials.

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

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

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

Refreshing Access Tokens Using Refresh Tokens

You can refresh Access Tokens using Refresh Tokens.

To refresh access tokens using Postman

1. Add the Postman extension to your Google Chrome web browser.
2. Open the Postman application.
3. Make a HTTP POST call with the following details:

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

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

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

Managing Tokens

You can use this page to delete the *active tokens* issued by Integration Cloud. Client applications use these tokens to access the resources on Integration Cloud. When you delete the tokens, the client application can no longer access the resources owned by the resource owners. See the [“Generating Tokens” on page 82](#) section on how to generate tokens (Access Token and Refresh Token) by using a REST Client. See [“About](#)

[OAuth 2.0](#) on page 76 for information on the high-level steps for configuring OAuth 2.0.

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

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

To delete a token

1. From the Integration Cloud navigation bar, go to **Settings**  > **OAuth 2.0** > **Token Management**.
2. Select a token and click **Delete**.

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

Running Services Using OAuth 2.0

To invoke service URLs using Postman

1. Add the Postman extension to your Google Chrome web browser.
2. Open the Postman application.
3. Type the **Request URL** and change the **HTTP method** to **POST**.



4. On the **Headers** tab, add *Authorization* as the **Key** and *Bearer <token ID>* as the **Value**.
5. Click **Send** to run the Integration.

2 Capability

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

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

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

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

Field	Description
Max allowed cores	Maximum number of CPU cores allowed across all active solutions and all stages for the tenant. You will not be able to create additional solutions if you exceed this capability.
Max allowed memory	Maximum memory capacity allowed across all active solutions and all stages for the tenant. You will not be

Field	Description
	able to create additional solutions if you exceed this capability.
Allowed number of stages	Maximum number of staging environments allowed for the tenant.
Trial account	If Yes , then the account is a trial account.
Trial end date	The trial period end date. This field appears only if the account is a trial account.

3 Connect

- Applications 90
- Keys and Certificates 227

Applications

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

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

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

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

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

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

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

Accounts

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

If you select an Account for an FTP, SFTP, custom SOAP, or on-premises Application and click **Test Connection**, the screen displays the status of the connection. If you have configured the Account details incorrectly in any stage, the stage appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the stage appears in green color and if an Account is not configured in a particular stage, that stage appears in white color.

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

Note: Only enabled or active Accounts are listed in the drop down list of the Operation wizard, Integration wizard, Look up Transformer, and Manage Stages page.

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

Note: Users who have the required permissions under **Settings**  **> 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 **Connect > Applications > <Application category>**.
2. Select an Application from the Application category page and then click **Accounts**.
To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.
3. From the Accounts screen, click **Add New Account** to add an Account or click **Edit** to update an existing Account.

Adding or Editing Accounts

Use the **Accounts** page to add, edit, or delete Accounts. The options available may vary according to the selected Application.

Note: See the [“Account Configuration Details” on page 93](#) section for information on the Account configuration fields for each Application.

To add or edit an Account

1. From the Integration Cloud navigation bar, click **Connect > Applications > <Application category>**.
2. Select an Application from the page, and then click **Accounts**.

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

3. From the **Accounts** page, click **Add New Account** to add an Account or click **Edit** to change any field in an existing Account.
4. On the **New Account** or **Edit Account** page, complete the following fields. Required fields are marked with an asterisk on the page.

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

Field	Description
Save As	<p>Provide a valid name for the Account. This field is common for all Applications. Names can contain alphanumeric characters, underscores (_), and hyphens (-). The name must not be null and cannot be an empty string. The following characters are also not allowed:</p> <ul style="list-style-type: none"> \\ (double backward slashes) / (forward slash) : (colon) * (asterisk) ? (question mark) " (double quote) < (Less Than symbol) > (Greater Than symbol) (vertical bar)
Description	<p>Provide a description for the Account. This field is common for all Applications.</p>

The Account configuration section allows you to provide details to connect with the Application. The fields available may vary according to the selected Application. See the [“Account Configuration Details” on page 93](#) section for information on the Account configuration fields for each Application. 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

Field	Description
	are listed in the drop down list of the Operation wizard, Integration wizard, Look up Transformer, and Manage Stages page.

See “[Manage](#)” on page 438 for more information.

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

- Click **Save** or **Update** to save your settings.

A new Account will be created.

Account Configuration Details

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

Alfabet

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the Account configuration. The URL depends on where the required instance of Alfabet is installed. It is possible to either include or omit the endpoint suffix “/Alfabet/api/vXX” in the URL. For example, both these options are equivalent: <ul style="list-style-type: none"> ■ https://myalfabet.com ■ https://myalfabet.com/Alfabet/api/v1
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.

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

Apache Solr Search

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

Field	Description
Server URL	<p>Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: https://<hostName>.</p> <p>Replace <hostName> with your actual back end system server URL hosting Apache Solr as the search engine.</p>
Response Timeout	<p>The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.</p>
Retry Count on Response Failure	<p>The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.</p>
Retry on Response Failure	<p>Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.</p>
Username	<p>Username received from the back end system hosting Apache Solr as the search engine.</p>
Password	<p>This is the password received from the back end system hosting Apache Solr as the search engine.</p>
Authorization Type	<p>Apache Solr REST APIs use Basic Authentication. The Username and Password is passed when you invoke any of the REST API endpoints. This is the type of HTTP authorization scheme to use for the connection. If you select none, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value</p>

Field	Description
	for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Applicability Statement 2 (AS2)

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

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

■ Data integrity

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

Field	Description
Recipient Endpoint	The endpoint URL of the recipient.
Authorization Type	<p>The type of HTTP authorization scheme to use for the connection. You can choose one of the following options:</p> <ul style="list-style-type: none"> ■ none: No additional authorization scheme will be executed at run time. For example, when you specify a user name and password, but do not specify a value for the authorization type, the user credentials are not inserted into an authorization header. ■ basic: When the application requires or supports HTTP basic authentication for user name and password.
From	The AS2 ID of the sender.
To	The AS2 ID of the recipient.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even

<u>Field</u>	<u>Description</u>
	though the request was sent successfully. Select this option if you want to re-establish the connection.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration from the list. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. Select Add New Truststore to add a new trust store from this list.
Keystore Alias	Select the alias for the Integration Cloud keystore configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. Select Add New Keystore to add a new keystore from this list.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both, Keystore Alias and Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation for guards against man-in-the-middle (MITM) attacks from the list. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier . This enables hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier from the list to disable hostname verification.
Username	The name of the user account that the AS2 connection will use to connect to the AS2 provider.
Password	The password for the user name provided in the Username field.
Compression	Select this option to compress an outbound AS2 message.
Sign Message	Select this option to sign an outbound AS2 message.
Signing Algorithm	The signing algorithm to use for an outbound AS2 message. The available options are:

Field	Description
	<ul style="list-style-type: none"> ■ MD5 ■ SHA-1 ■ SHA-256 ■ SHA-384 ■ SHA-512
Signing Keystore and Key Aliases	The keystore aliases and the key aliases in the keystore to use for signing an outbound AS2 message.
Receive Signed Message	Select this option to receive a signed inbound AS2 message. If you select this option and the incoming AS2 message is not signed, then an <code>Insufficient message security</code> error is encountered and shared with the sender if MDN is requested by the sender.
Signature Verification Certificate	The certificate to use for verifying an inbound signed AS2 message.
Encrypt Message	Select this option to encrypt an outbound AS2 message.
Encryption Algorithm	<p>The encryption algorithm to use for an outbound AS2 message. The available options are:</p> <ul style="list-style-type: none"> ■ RC2 40 ■ RC2 64 ■ RC2 128 ■ DES ■ TripleDES ■ AES 128 ■ AES 192 ■ AES 256
Encryption Certificate	The certificate to use for encrypting an outbound AS2 message.
Receive Encrypted Message	Select this option to receive an encrypted inbound AS2 message. If you select this option and the

Field	Description
	incoming AS2 message is not encrypted, then an <code>Insufficient message security error</code> is encountered and shared with the sender if MDN is requested by the sender.
Decryption Keystore and Key Aliases	The keystore aliases the key aliases in the keystore to use for decrypting an inbound AS2 message.
Request MDN	<p>Whether you want the recipient to return an MDN to the sender.</p> <p>You can select one of the following options:</p> <ul style="list-style-type: none"> ■ None: The recipient of the AS2 message does not return an MDN to the sender. ■ Synchronous: The recipient of the AS2 message returns an MDN to the sender through the same HTTP connection used to send the original AS2 message. ■ Asynchronous: The recipient of the AS2 message returns an MDN to the sender through a different HTTP connection instead of the one used to send the original AS2 message.
Request Signed MDN	<p>Select this option if you want the recipient to sign an AS2 MDN.</p> <p>Ensure that you also select an option in the Request MDN field if you want the recipient to sign and return an AS2 MDN.</p>
Asynchronous MDN Endpoint	Type your endpoint URL that accepts an inbound AS2 MDN if you selected the Asynchronous option for Request MDN .
AS2 Version	Select the AS2 protocol version to use from the list.

Amazon DynamoDB

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://dynamodb.<instance>.amazonaws.com.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Access Key	This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key.

Field	Description
Secret Key	This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots.
Region	An area specific value.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.

Amazon Kinesis

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://kinesis.<Region>.amazonaws.com.</code>
Access Key	This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key.
Secret Key	This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.

Field	Description
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Region	An area specific value.

Amazon Simple Notification Service(SNS)

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

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

Field	Description
	string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key.
Secret Key	Secret key obtained from AWS Identity and Access Management (IAM) Console. This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots.
Region	An area specific value. The region is different for different users.
Signing Algorithm	Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.

Amazon Simple Queue Service (SQS)

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

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

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

Amazon Simple Storage Service (S3)

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

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

Field	Description
	keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Use Expect Continue	Whether to use the Expect/Continue HTTP/1.1 handshake and send the Expect request header. When the client sends the Expect request header, the client waits for the server to confirm that it will accept the request, before the client sends the request body. Enable this option to use the Expect/Continue handshake.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Access Key	This is a username. It is an alphanumeric text string that uniquely identifies the user who owns the account. No two accounts can have the same Access Key.
Secret Key	This key plays the role of a password. It is called secret because it is assumed to be known only by the owner. When you type the secret key, it is displayed as an asterisk or dots.
Region	An area specific value.
Signing Algorithm	Explicitly select the signing algorithm, for example, HMAC-SHA1 Signatures used to sign the message.

Atlassian Jira

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

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

Field	Description
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Avalara AvaTax

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://<instance_name>.avalara.net</code> , where <code><instance_name></code> represents the actual instance name.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.

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

Field	Description
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Cloud Deployment

Use this Application to invoke Java and Flow services for any [“Cloud Deployment” on page 454](#) solution.

Field	Description
Run As	Select the user name you want Integration Cloud to use while running the cloud deployment service. Integration Cloud runs the service as if the user you specify is the authenticated user that invoked the service. If the service is governed by an Access Control List, be sure to specify a user that is allowed to invoke the service. <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p>Note: You must map the selected user's Access Profile to a webMethods Integration Server user group (Settings  > Access Profiles > Select an Access Profile > Solution Permissions), else you will not be able to view, edit, or run webMethods Integration Server services in a solution.</p> </div>
Stage	Select the stage where the cloud deployment services are available.

CloudStreams Connector for Anaplan[®]

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format: http://host:port. Replace < host:port > with your actual JIRA instance.

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

Field	Description
	org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

CloudStreams Connector for Microsoft Azure Cosmos DB

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

Field	Description
Server URL	The login endpoint to initiate communication with the SaaS provider. For example, for the CloudStreams Connector for Microsoft Azure Cosmos DB, the end point URL will be of the format: https://<accountName>.documents.azure.com:443/. Replace <accountName> with the name of your Microsoft Azure Cosmos DB account.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.

Field	Description
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Access Key	The Access key token that contains the secret of the account required by the SaaS provider.

CloudStreams Connector for Microsoft Azure Storage

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

Field	Description
Server URL	<p>The login endpoint to initiate communication with the SaaS provider. For example, for the CloudStreams Connector for Microsoft Azure Storage, the end point URL will be of the format: <code>https://<accountName></code>.</p> <p>Replace <code><accountName></code> with the name of your Microsoft Azure Storage account.</p>

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

CloudStreams Connector for NetSuite™

Integration Cloud connects to NetSuite™ SuiteTalk platform using the SuiteTalk web services. It provides programmatic access to NetSuite™ data related to accounting,

order management/inventory, CRM, professional services automation (PSA), and eCommerce applications through operations like `addList`, `get`, `updateList`, `upsertList`, and `deleteList`.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. This is the native provider endpoint target for the Account configuration. You may need to specify the correct URL for your exact instance, for example: <code>https://webservices.na1.netsuite.com/services/NetSuitePort_2016_2</code> , where <code>na1</code> is the instance name.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Email	The user email account that the connection will use to connect to the SaaS provider.
Password	The password of the user email account.
Authorization Type	This is the type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.

Field	Description
Account	Specify the account number issued to you by NetSuite™.
Role	Specify the role with which you want to execute the web services, for example, Administrator.
ApplicationID	When you create the NetSuite™ Account, it sends a login request to the back end using email, password, Account, and the Application Id. Application Id is required for requests using end point 2015.2 or later.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

CloudStreams Connector for Salesforce® Bulk v2 Data Loader

Using the Bulk API 2.0, Salesforce supports Job resource, and allows you to create, update, delete, upsert jobs, and operate on large number of records asynchronously by submitting jobs which are processed in the background.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, https://<instance>.salesforce.com. Replace <instance> with your actual Salesforce instance.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token, for example, https://<instance>.salesforce.com/services/oauth2/token.
Access Token	This token is used for authentication and is issued by the Authorization Server.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote

Field	Description
	server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Concur

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: <code>https://<instance>/api</code> . Replace <code><instance></code> with your actual Concur instance.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.

Field	Description
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server. Concur REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.

Field	Description
Refresh URL	This is the provider specific URL to refresh an Access Token.

Coupa

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, for Coupa, the end point URL would be of the format: https://<instance>.com. Replace <instance> with your actual Coupa instance.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a

Field	Description
	client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
X-COUPA-API-KEY	<p>The API Key received from the user account.</p> <p>Coupa REST APIs authentication requests require a unique API key generated in Coupa. All API requests must pass an X-COUPA-API-KEY header with an API key. A key can be created from the API Keys section of the Administration tab by an administrator. The key is a 40-character long case-sensitive alphanumeric code. The API key is associated with an API user who is the equivalent of an administrator in Coupa. Any changes to resources through the API are attributed to the API user.</p>

Cumulocity

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

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

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

Field	Description
	identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

DocuSign

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

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, for the DocuSign connector version 2, the end point URL is of the format: <code>https://demo.DocuSign.net/restapi</code> .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a

Field	Description
	client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server. DocuSign REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints. The Access Token is valid in all future API calls to authenticate the user, until the token is revoked. It is not affected by password changes.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token.

File Transfer Protocol (FTP/FTPS)

Integration Cloud connects to an FTP server using the FTP protocol and provides operations to list, download, upload, and delete files. It also supports FTPS (FTP over SSL).

Note: FTP is not a secure file transfer protocol and it has security vulnerabilities. It does not provide any encryption for data transfer. It is recommended to use secured protocols such as HTTPS and FTPS for securing the data transmitted over the network.

Note: See this ["video"](#) on how to create an Account for an FTP Application and test the connection.

Field	Description
Host	Host name or IP address or the domain name of the FTP server.

Field	Description
Port	FTP port defined on the FTP server.
User	Valid user name on the FTP server.
Password	Password of the FTP user.

SSL Configuration - Select this option for secured FTP connection.

Secure Data	Select True to secure the data channel. Select False if you do not want to secure the data channel.
Keystore Alias	Alias to the keystore that contains the private key used to connect to the host securely. You can also add a new Keystore from this field.

Note: Users who have the **Administer** permission under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Advanced Security** can add, edit, and delete Keystores.

Key Alias	Alias to the key in the keystore that contains the private key used to connect to the host securely. The key must be in the keystore specified in the Keystore Alias field.
-----------	--

Truststore Alias	The alias for the truststore, which contains the trusted root of a certificate or signing authority (CA). You can also add a new Truststore from this field.
------------------	--

Note: Users who have the **Administer** permission under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Advanced Security** can add, edit, and delete Truststores.

Google Analytics

Integration Cloud connects to Google Analytics using the API for Management services. You can use Management services to retrieve, create, update, and delete analytics configuration data (accounts, metrics, dimensions, and custom data sources).

Integration Cloud also connects to Google Analytics using the API v4 for Core Reporting services. You can use Reporting services to generate customized reports based on dimensions, date range, and metrics.

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

Field	Description
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google Apps Admin

Integration Cloud connects to Google Apps Admin and supports the functionality to create and list users.

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

Field	Description
	is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google BigQuery

Integration Cloud connects to Google BigQuery using the Google BigQuery API and allows you to create, update, and delete data sets and tables. You can also load, copy, extract, and query data from BigQuery's Bigtable.

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

Field	Description
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google Calendar

Integration Cloud connects to Google Calendar using Google Calendar APIs. It enables you to manage calendar data such as Secondary Calendars, Events, and Quick Event Add.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/calendar/v3 .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.

Field	Description
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: <code>https://www.googleapis.com/oauth2/v4/token</code>

Google Contacts

Integration Cloud connects to Google Contacts using Google Contacts APIs. It enables you to manage a user's contact list. The contacts are usually stored in the user's Google Account.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.google.com/m8/feeds .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.

Field	Description
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google Drive

Integration Cloud connects to Google Drive using the Google Drive API. It provides functionality of file storage and access to list, upload, and delete files.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> ,

Field	Description
	which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: <code>https://www.googleapis.com/oauth2/v4/token</code> .

Google Cloud Pub/Sub

Integration Cloud connects to Google Cloud Pub/Sub and allows you to create, get, delete, set policy, and get policy on topics and subscription resources.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://pubsub.googleapis.com</code> .

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

Field	Description
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google Cloud Storage

Integration Cloud connects to Google Cloud Storage using the Google Cloud Storage API and allows you to create and manage Buckets, Objects, and AccessControls.

Field	Description
Server URL	Provide the login endpoint to initiate communication with Google Cloud Storage. Example: https://www.googleapis.com .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.

Field	Description
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

Google Prediction

Integration Cloud connects to Google Prediction using the Google Prediction API. It includes the Hosted Model and Trained Model services that are used to predict data by using machine learning.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://www.googleapis.com/prediction/v1.6 .

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

Field	Description
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token

Google Sheets

Integration Cloud connects to Google Sheets and allows you to create, update, and get a spreadsheet, as well as append values to a spreadsheet.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://sheets.googleapis.com .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.

Field	Description
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Consumer ID	Also referred to as the Client ID, specify the Client ID you obtained from the Google Developer Console. This is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, specify the Client Secret you obtained from the Google Developer Console. This is a secret matching to the client identifier.
Access Token	Specify the access token you obtained from the OAuth Playground. This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token. Example: https://www.googleapis.com/oauth2/v4/token .

IBM Watson Tone Analyzer

Integration Cloud connects to IBM Watson Tone Analyzer using the REST interface to detect emotional, social, and language tones in written text. You can use the Application to learn the tone of your customer's communications and to respond to each customer appropriately, or to understand and improve customer conversations.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider.

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

Field	Description
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Magento eCommerce Platform

Integration Cloud connects to Magento using the Magento REST API. You can use it to manage customers, customer addresses, sales orders, inventory, products, and so on, without having to directly work on Magento.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>http://<yourhost>/api/rest</code> .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a

Field	Description
	client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server.
Access Token Secret	A secret used by the Consumer to establish ownership of a given Access Token.

Marketo

Integration Cloud connects to Marketo using the Marketo REST API and allows you to create, retrieve, and remove entities and data stored within Marketo.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: <code>https://<instance>.mktoreset.com</code> . Replace <code><instance></code> with your actual Marketo instance.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is

Field	Description
	recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is org.apache.http.conn.ssl.DefaultHostnameVerifier, which will enable hostname verification. Select org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.

Field	Description
Refresh URL	This is the provider specific URL to refresh an Access Token.

Microsoft Dynamics CRM

Integration Cloud connects to **Microsoft Dynamics CRM** using the Microsoft Dynamics CRM SOAP API. You can manage CRM data and access metadata that defines the specific CRM instance to which you are connecting.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: <code>https://<organization>.api.crm.dynamics.com/XRMServices/2011/Organization.svc</code> , where <code><organization></code> must be replaced with your actual Microsoft Dynamics CRM organization.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider.
Password	Provide a password for the user name to initiate communication with the SaaS provider.

Field	Description
Authorization Type	This is the type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Microsoft Dynamics CRM 365

Integration Cloud connects to **Microsoft Dynamics CRM 365** using the OData API Version 4.0 and allows you to manage CRM data and access metadata that defines the specific CRM instance to which you are connecting. This Application performs standard CRUD operations on business objects by connecting to the OData service endpoint.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, for Microsoft Dynamics CRM 365, the end point URL is the OData service endpoint.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token.
Access Token	This token is used for authentication and is issued by the Authorization Server.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though

Field	Description
	the request was sent successfully. Select this option if you want to re-establish the connection.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Caching	Select this option if you want the Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

OData v2.0

Integration Cloud connects to any cloud application that exposes its services using the OData Version 2.0 Specification. It supports only those OData providers, which strictly adhere to the OData Version 2.0 Specification and allows you to perform standard CRUD operations on business objects by connecting to the OData service endpoint.

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

Field	Description
	password. If you enter the username and password, then set the authorization type as basic .
Caching	Select this option if you want the OData v2.0 Application to cache the backend metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

OData v4.0

Integration Cloud connects to any cloud application that exposes its services using the OData Version 4.0 Specification. It supports only those OData providers, which strictly adhere to the OData Version 4.0 Specification and allows you to perform standard CRUD operations on business objects by connecting to the OData service endpoint.

Field	Description
Server URL	This is the OData service endpoint to initiate communication with the OData provider.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Caching	Select this option if you want the OData v4.0 Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent. By default, the cache

Field	Description
	will be refreshed every 12 hours. It is recommended to enable the cache to increase the performance.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.

On-Premises Applications

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

If you select an Account for an on-premises Application and click **Test Connection**, the screen displays the status of the connection. If you have configured the Account details incorrectly in any stage, the stage appears in red color in the **Connectivity Status** column. If an Account is configured correctly in a particular stage, the stage appears in green color and if an Account is not configured in a particular stage, that stage appears in white color. For on-premises Applications, the Account can be used to execute services on the on-premises webMethods Integration Server.

REST Applications - Account Configuration Details

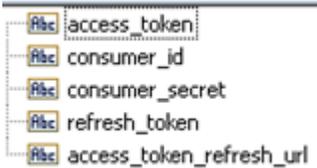
Integration Cloud allows you to create custom REST Applications. REST (Representational State Transfer) is an architectural style that requires web applications

to support the HTTP GET, POST, PUT, and DELETE methods and to use a consistent, application-independent interface.

Field	Description
Server URL	This is the login Endpoint URL you have specified in the Define Application Details page while creating the REST Application.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Consumer ID	Also referred to as the Client ID, in OAuth 2.0, this is a client identifier issued to the client to identify itself to the authorization server. For Auth 1.0a, it is the Consumer Key issued by the Service Provider and used by the consumer to identify itself to the Service Provider.
Access Token	This token is used for authentication and is issued by the Authorization Server. For OAuth 1.0a, it is a value used by the Consumer to gain access to the Protected Resources

Field	Description
	on behalf of the User, instead of using the User's Service Provider credentials.
Access Token Secret	A secret used by the Consumer to establish ownership of a given Access Token. For OAuth 1.0a, it is the secret used by the Consumer to establish ownership of a given Access Token.
Refresh URL	The provider specific URL to refresh an Access Token.
Session Timeout (min)	The maximum number of minutes a session can remain active, in other words, how long you want the server to wait before terminating a session. The value should be equal to the session timeout value specified at the SaaS provider back end.
Username	The username credentials for the current Account configuration.
Password	The password credentials for the current Account configuration.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field. This option is available only if Credentials is selected as the Authentication Type while creating the Application.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification. This option is available only if Credentials is selected as the Authentication Type while creating the Application.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field. This option is available

Field	Description
	only if Credentials is selected as the Authentication Type while creating the Application.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields. This option is available only if Credentials is selected as the Authentication Type while creating the Application.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier. For Auth 1.0a, it is the secret used by the Consumer to establish ownership of the Consumer Key.
Refresh Token	Issued with the OAuth v2.0 access token only. A token used by the client to obtain a new access token without having to involve the resource owner.
Refresh URL Request	Options for sending the parameters in the Access Token refresh request. The options are Body Query String , URL Query String , and Custom Integration . Body Query String - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the body of the POST request. URL Query String - The refresh request parameters, for example, refresh_token, grant_type, and so on, and their values are sent as query strings in the URL of the POST request. Custom ESB Service - Use this option if the back end requires refresh requests in a custom format. If you select this option, you must specify the name of your Integration in the Refresh Custom ESB Service field. You can also create an Integration by clicking the link.
Refresh Custom ESB Service	To refresh the access tokens for accounts which use the OAuth 2.0 protocol, you can specify a call-back Integration which will execute when the access token expires. This is a user implemented service for refreshing the <i>OAuth 2.0</i> Access Token. This option allows you to create an Integration, which will be executed when the Access Token has expired or is not valid.

Field	Description
	<p>To create this Integration, create a custom REST Application with an Operation that invokes the back end to fetch the refresh token. You can use this custom REST Application along with its Operation to create this Integration.</p> <p>Click the <i>Integration link</i> to create a new Integration.</p> <p>The Integration must have a specification whose input parameters are:</p>  <p>and the output parameters are:</p>  <p>Note: The integration will be pre-populated with the input/output adhering to the above mentioned specification.</p> <p>The newly created Integration will generate an access token that is mapped to the access token field in the output signature.</p> <p>Note: While editing the Account, this new Integration appears in the Refresh Custom ESB Service field in the Account Configuration page.</p>

Salesforce

Salesforce Bulk Data Loader

Integration Cloud connects to Salesforce using the Salesforce Bulk API and supports Job and Batch resources. You can use it to create, update, delete, query jobs and batches, and operate on large number of records asynchronously by submitting batches which are processed in the background by Salesforce.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: https://login.salesforce.com/services/Soap/u/31.0 .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider.
Password	The password for the user name provided in the Username field. When you access Salesforce.com from outside your company's trusted networks, you must add a security token (provided by Salesforce) to your password. For more information about logging on Salesforce.com, see the Salesforce.com documentation.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.

Field	Description
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Salesforce CRM

Integration Cloud connects to Salesforce using the Partner SOAP API. It supports all business objects (for example, Account) and operations including any customizations done on the Salesforce instance. It also supports Salesforce analytics using wave.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://login.salesforce.com/services/Soap/u/31.0</code> .
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is

Field	Description
	recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Username	This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider.
Password	The password for the user name provided in the Username field. When you access Salesforce.com from outside your company's trusted networks, you must add a security token (provided by

Field	Description
	Salesforce) to your password. For more information about logging on Salesforce.com, see the Salesforce.com documentation.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.

SAP Cloud for Customer(C4C) OData v2.0

Integration Cloud connects to SAP Cloud for Customer (C4C) including SAP Cloud for Sales, SAP Cloud for Service, and SAP Cloud for Social Engagement solutions using the REST interface, and allows you to do standard CRUD operations on business objects by connecting to the OData Service endpoint.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. This is the OData service endpoint to initiate communication with the SAP C4C OData provider.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.

Field	Description
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SAP C4C OData provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider.
Password	Provide the password for the user name provided in the Username field to initiate communication with the SaaS provider.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password , and also select none , you do not specify a value for the Authorization Type , so the user credentials are not inserted into an Authorization header. If you enter the Username and Password , then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Metadata Caching	Select this option if you want the SAP C4C Application to cache the backend metadata. Caching of the metadata significantly increases the performance of a request sent through SAP C4C. If this option is selected, the cache will be refreshed every 12 hours. It is recommended to enable the metadata cache to increase the performance.
Use CSRF Token	To prevent cross site request forgery, SAP C4C protects its resources by using a CSRF token. Select this option if you want Integration Cloud to use the CSRF token key, received in the response from SAP C4C, to perform any state changing requests on SAP C4C. By default, the CSRF token is enabled by the SAP C4C back end. You must enable this option particularly when the entity state changing operation is invoked.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used

Field	Description
	to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

SAP S/4HANA Marketing Cloud

Integration Cloud connects to SAP S/4HANA Marketing Cloud using the OData based REST interface, which allows you to do only bulk imports. You can create or update Interaction Contacts, Interactions, Interests, Corporate Accounts, Product Categories, Products, and so on.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. This is the OData service endpoint to initiate communication with the SAP S/4HANA Marketing Cloud provider.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.

Field	Description
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SAP S/4HANA Marketing Cloud provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider.
Password	Provide the password for the user name provided in the Username field to initiate communication with the SaaS provider.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password , and also select none , you do not specify a value for the Authorization Type , so the user credentials are not inserted into an Authorization header. If you enter the Username and Password , then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Metadata Caching	Select this option if you want the SAP S/4HANA Marketing Cloud Application to cache the back end metadata. Caching of the metadata significantly increases the performance of a request sent through SAP S/4HANA Marketing Cloud. If this option is selected, the cache will be refreshed every 12 hours. It is recommended to enable the metadata cache to increase the performance.
Validate Metadata	Whether to validate the \$metadata xml during edm object creation. Select this option to enable the metadata validation.
Use CSRF Token	To prevent cross site request forgery, SAP S/4HANA Marketing Cloud protects its resources by using a CSRF token. Select this option if you want Integration Cloud to use the CSRF token key, received in the response from SAP S/4HANA Marketing

Field	Description
	Cloud, to perform any state changing requests on SAP S/4HANA Marketing Cloud. By default, the CSRF token is enabled by the SAP S/4HANA Marketing Cloud back end. You must enable this option particularly when the entity state changing operation is invoked.
Use Chunking	Enable this option if you want to send or receive a large binary stream with a chunk size of 8192 bytes. This is applicable only if the back end supports HTTP/1.1 chunking.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Secure File Transfer Protocol (SFTP)

The SSH File Transfer Protocol (SFTP) is a network protocol that is based on the Secure Shell protocol (SSH). SFTP facilitates secure file access, file transfer, and file management over any reliable data stream. The Secure File Transfer Protocol (SFTP) Application downloads files from or uploads files to an SFTP-enabled server using the secure file transport channel.

Integration Cloud connects to an SFTP server using the SSH File Transfer Protocol (SFTP) and provides operations to retrieve, transfer, rename, and delete files or

directories in the SFTP server. You can also change the permission or ownership of files in the SFTP server.

You can configure Integration Cloud to connect to an SFTP server to perform the following tasks using the SFTP protocol:

- Transfer files between Integration Cloud and the SFTP server. You can get a file from the SFTP server or upload a file to the SFTP server.
- Access files in the SFTP server. You can view the directories and files in the SFTP server and also view their permissions and ownership information.
- Manage directories or files in the SFTP server. You can create, rename, or delete files or directories in the SFTP server. You can also change the permissions or ownership of files in the SFTP server.

For this parameter...	Specify...
Host Name or IP Address	The host name or IP address of the SFTP server.
Port Number	The port number of the SFTP server. The port number must be within the range of 1 and 65535 (inclusive).
Host Public Key	The public key of the SFTP server. Select Auto Retrieve if you want Integration Cloud to automatically retrieve the public key of the SFTP server. Select Upload if you have the public key of the SFTP server. Integration Cloud will use the uploaded public key.
Finger Print	<p>The host public key fingerprint of the SFTP server.</p> <div style="background-color: #f0f0f0; padding: 5px; margin: 5px 0;"> <p>Note: This field is visible only after you have established a connection to an SFTP server.</p> </div> <p>Before establishing a connection, the SFTP server sends an encrypted fingerprint of its host public keys to ensure that the SFTP connection will be exchanging data with the correct server. Save the fingerprint information locally. This enables you to check the fingerprint information against the data you have saved every time you establish a new connection.</p>
User Name	The user name for the SFTP user account.
Authentication Type	The type of authentication Integration Cloud uses to authenticate itself to the SFTP server. Client authentication can be either by password or by public and private keys. Select Password if you want to use

<u>For this parameter...</u>	<u>Specify...</u>
	password authentication. If you are using password authentication, enter the password for the specified user to connect to the SFTP server. Select Public Key if you want to authenticate Integration Cloud by using public and private keys. To use this authentication type, the SFTP server and Integration Cloud must each have access to their own private key and each other's public key.
Private Key	If you selected Public Key as the authentication type, select the private key file of the specified SFTP user.
PassPhrase	If you selected Public Key as the authentication type and if the private key you specified requires a passphrase, enter the passphrase for the private key file of the specified user.
Advanced Options	
Maximum Retries	The number of times Integration Cloud attempts to connect to the SFTP server. The maximum allowed value is 6. The minimum allowed value is 1.
Response Timeout	The amount of time (measured in seconds) Integration Cloud waits for a response from the SFTP server before timing out and terminating the request. A value of 0 indicates that the session will never time out. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value.
Session Timeout	The number of minutes you want Integration Cloud to wait before terminating an idle session. The session timeout value must be within the range of 10 and 60 minutes.
Preferred Key Exchange Algorithms	The algorithms that Integration Cloud presents to the SFTP server for key exchange. You can specify the order in which Integration Cloud presents the algorithms to the SFTP server by moving the available algorithms up or down by clicking Move Up or Move Down . The SFTP server has its own set of preferred algorithms configured. At the time of key exchange, one of the algorithms supported by both Integration Cloud and the SFTP server will be chosen.

For this parameter...	Specify...
Compression	Whether or not to compress the data to reduce the amount of data that is transmitted. Integration Cloud supports compression using the compression algorithm zlib . You can use compression only if the SFTP server that you are connecting to supports compression. Select None if you do not want to compress the data. Select zlib if you want to compress the data that is transmitted between the SFTP server and Integration Cloud.
Compression Level	The compression level to use if you selected the compression algorithm zlib in the Compression field. The minimum allowed value is 1 (fast, less compression) and the maximum allowed value is 9 (slow, most compression).

ServiceNow Enterprise Service Management

Integration Cloud connects to different areas (Incident, Problem, and Change management) of ServiceNow using the Geneva version of the ServiceNow API. You can create incidents, get details of created incidents, and update and delete them. Similar operations are available for problem and change management cloud applications.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://<instance_name>.service-now.com</code> , where <code><instance_name></code> represents the actual instance name.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent

Field	Description
	successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider.
Password	Provide a password for the user name provided in the Username field to initiate communication with the SaaS provider.
Authorization Type	The type of HTTP authorization scheme to use for the connection. If you select none , no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Siemens MindSphere

Integration Cloud connects to Siemens MindSphere using the API version 2.0 and allows you to create aspects and post data into a MindSphere asset.

Note: The fields displayed may vary according to the version of the Application.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, https://mindconnectcom.apps.mindsphere.io/
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field.

Field	Description
	The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
MSU Limit	MindSphere Unit (MSU) limit for an asset. MSU is the basis for fees invoiced monthly per asset and as per a used application.
Security Profile	The type of security profile/algorithm used for token generation. Default is <code>SHARED_SECRET</code> .
Agent Id	The ID of the agent managing a MindSphere asset.
Schemas	Schemas used by the MindSphere platform. Default is <code>urn:siemens:mindsphere:v1</code> .
IAT	Initial access token for agent onboarding.
Tenant	The tenant ID of the onboarded agent.
Links	Links returned upon successful agent onboarding.
JWT Signing Algorithm	The JSON Web Token (JWT) signing algorithm. Default is HS256 (HMAC with SHA 256).
JWT Token Expiration (msec)	The JWT token expiration time in milliseconds. Default is 600000.

Field	Description
JWT Audience	The JWT Audience. Default is MindSphere AS.

Slack

Integration Cloud connects to Slack using the Slack REST API. You can use it to collaborate in your team within persistent chat rooms, private groups, and direct messaging, where all the content is searchable.

Field	Description
Server URL	<p>Provide the login endpoint to initiate communication with the SaaS provider.</p> <p>The URL for Slack REST Application depends on the team name: <code>https://YOURTEAM.slack.com</code></p> <p>Example: <code>https://exampleteam.slack.com</code></p>
Response Timeout	<p>The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.</p>
Retry Count on Response Failure	<p>The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.</p>
Retry on Response Failure	<p>Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.</p>
Keystore Alias	<p>Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.</p>
Client Key Alias	<p>Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's</p>

Field	Description
	identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Access Token	The token used for authentication and issued by the Authorization Server.

SOAP Applications - Account Configuration Details

Integration Cloud allows you to create Custom SOAP Applications. Custom SOAP Applications enable you to access third party web services hosted in the cloud or on-premises environment. The Custom SOAP Application uses a WSDL that is accessible through publicly or locally accessible URLs.

Field	Description
Port Binding	Select the bind address from the drop-down list, that is, the concrete protocol and data format specification for the web service.
URL	This is the URL for the web service. You can edit the URL to specify a different web service endpoint.
Response Timeout	<p>The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the response timeout value. It is recommended to specify a value other than 0.</p> <p>If you specify 0, Integration Cloud will wait indefinitely for a response.</p> <p>If you do not specify a timeout, Integration Cloud will consider 5 minutes as the response timeout.</p> <p>If the connection to the host provider ends before Integration Cloud receives a response, the web service operation ends with an exception and a status code of 408.</p>

Field	Description
User	User name used to authenticate the consumer at the HTTP or HTTPS transport level on the host server.
Password	The password used to authenticate the consumer on the host server.
Keystore Alias	Alias to the keystore that contains the private key used to connect to the Web Service host securely. You can also add a new Keystore from this field.

Note: Users who have the **Administer** permission under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Advanced Security** can add, edit, and delete Keystores.

Key Alias	Alias to the key in the keystore that contains the private key used to connect to the Web Service host securely. The key must be in the keystore specified in the Keystore Alias field.
------------------	--

Show Advanced Options - WS-Security properties are used by the SOAP processor to provide security information in the WS-Security header of the SOAP message.

Security Credentials

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

Keystore / Truststore

Keystore Alias	The alias for the keystore, which contains private keys and certificates associated with those private keys. You can also add a new Keystore from this field.
-----------------------	---

Note: Users who have the **Administer** permission under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Advanced Security** can add, edit, and delete Keystores.

Key Alias	The text identifier for the private key associated with the Keystore Alias .
------------------	---

Field	Description
Truststore Alias	<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>Note: Users who have the Administer permission under Settings  > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security can add, edit, and delete Truststores.</p>
Partner Certificate Alias	<p>The file that contains the partner's self-signed certificate. You can also add a new Partner Certificate from this field.</p> <p>Note: Users who have the Administer permission under Settings  > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security can add, edit, and delete Partner Certificates.</p>
Timestamp	
Timestamp Precision	<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>
Timestamp TTL	<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>
Timestamp Max Skew	<p>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.</p>

StrikeIron Contact Verification

Integration Cloud connects to StrikeIron using the StrikeIron Contact Verification APIs, and provides access to email verification and hygiene services, along with the North America address verification service.

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

Field	Description
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

SuccessFactors HCM

Integration Cloud connects to SuccessFactors using the SuccessFactors web service SFAPI, and performs SuccessFactors operations (Create, Read, Update, Delete, Fetch, Insert, Query, queryMore, and Upsert) over HTTP using synchronous SOAP protocols. This Application has been tested with the following business objects: GOAL\$1, GOAL\$2, GOAL\$3, GoalMilestone\$2, GoalMilestone\$3, GoalTask\$2, GoalTask\$3, MatrixManager, and CustomManager.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. Example: <code>https://api.successfactors.com/sfapi/v1/soap</code> <code>https://<instance_name>.successfactors.com</code> , where <code><instance_name></code> represents the actual instance name.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.

Field	Description
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Username	This is the user account name on the SaaS provider that the Account will use to connect to the SaaS provider. The Account will use this credential to connect to the SaaS provider.
Password	Provide a password for the user name to initiate communication with the SaaS provider.
Authorization Type	<p>The type of HTTP authorization scheme to use for the Account. The SuccessFactors Application does not use Authorization headers.</p> <p>If you specify none, no additional authorization scheme will be executed at run time. If you specify a Company ID, Username, and Password, but do not specify a value for Authorization Type, the user credentials are not inserted into an Authorization header. If you enter the username and password, then set the authorization type as basic. Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password.</p>
Company ID	The company ID that SuccessFactors provided, when your company registered with them.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select

Field	Description
	org.apache.http.conn.ssl.NoopHostnameVerifier to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

SugarCRM

Integration Cloud connects to SugarCRM using the Interface for RESTful Web Services v10 and manages the CRM data. You can use it to retrieve, query, create, update, and delete business objects of any type.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: https://<instance>/rest/v10. Replace <instance> with your actual SugarCRM instance.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.

Field	Description
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server. SugarCRM REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.
Refresh URL	This is the provider specific URL to refresh an Access Token.
Trust store Alias	Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Twilio

Using the REST interface, Twilio allows you to programmatically make and receive phone calls and send and receive text messages.

Field	Description
Server URL	<p>Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL would be of the format:</p> <p>https://api.twilio.com/2010-04-01</p>
Authorization Type	<p>The type of HTTP authorization scheme to use for the connection. If you select none, no additional authorization scheme will be executed at run time. For example, when you specify a Username and Password, but do not specify a value for the Authorization Type, the user credentials are not inserted into an Authorization header.</p> <p>If you enter the username and password, then set the authorization type as basic . Basic refers to HTTP Basic Authentication. This option can be used if the Application requires or supports HTTP Basic authentication using a username and password. Select the Authorization Type as basic.</p>
Response Timeout	<p>The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.</p>
Retry Count on Response Failure	<p>The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.</p>
Retry on Response Failure	<p>Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.</p>
Username	<p>The name of the user account on the SaaS provider that the connection will use to connect to the SaaS provider.</p>
Password	<p>The password for the user name provided in the Username field.</p>
Trust store Alias	<p>Select the alias name of the Integration Cloud trust store configuration. The trust store contains trusted certificates used</p>

Field	Description
	to determine trust for the remote server peer certificates. You can also add a new Truststore from this field.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.

Zendesk

Integration Cloud connects to Zendesk using the Zendesk API v2. It includes ticketing system, self-service options, and customer support features, and allows you to create, update, and solve customer support tickets and also track problems and questions.

Field	Description
Server URL	Provide the login endpoint to initiate communication with the SaaS provider. For example, the end point URL is of the format: <code>https://<domain>.zendesk.com</code> . Replace <code><domain></code> with your actual Zendesk instance.
Response Timeout	The number of milliseconds Integration Cloud waits for a response before canceling its attempt to connect to the back end. In case the network is slow or the back end processing takes longer than usual, increase the Response Timeout value. It is recommended to specify a value other than 0. If you specify 0, Integration Cloud will wait indefinitely for a response.

Field	Description
Retry Count on Response Failure	The number of times Integration Cloud attempts to connect to the back end to read a response if the initial attempt fails. If an I/O error occurs, it will retry only if you have selected the Retry on Response Failure option.
Retry on Response Failure	Whether Integration Cloud should attempt to resend the request when the response has failed, even though the request was sent successfully. Select this option if you want to re-establish the connection.
Keystore Alias	Select the alias for the Integration Cloud key store configuration. This is a text identifier for the keystore alias. A keystore file contains the credentials (private key/signed certificate) that a client needs for authentication. You can also add a new Keystore from this field.
Client Key Alias	Alias to the private key in the keystore file specified in the Keystore Alias field. The outbound connections use this key to send client credentials to a remote server. To send the client's identity to a remote server, you must specify values in both the Keystore Alias and the Client Key Alias fields.
Hostname verifier	Select a hostname verifier implementation. Guards against man-in-the-middle (MITM) attacks. The default is <code>org.apache.http.conn.ssl.DefaultHostnameVerifier</code> , which will enable hostname verification. Select <code>org.apache.http.conn.ssl.NoopHostnameVerifier</code> to disable hostname verification.
Consumer ID	Also referred to as the Client ID, this is a client identifier issued to the client to identify itself to the authorization server.
Consumer Secret	Also referred to as the Client Secret, this is a secret matching to the client identifier.
Access Token	This token is used for authentication and is issued by the Authorization Server. Zendesk REST APIs use OAuth 2.0. The Access Token is passed when you invoke any of the REST API endpoints.
Refresh Token	A token used by the client to obtain a new access token without involving the resource owner.

Field	Description
Refresh URL	This is the provider specific URL to refresh an Access Token.

Operations

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

Note: Users who have the required permissions under **Settings**  **> 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 page lists all the available Operations for a selected application including predefined Operations.

See [“FTP Predefined Operations” on page 199](#) for information on the predefined FTP operations.

See [“SFTP Predefined Operations” on page 203](#) for information on the predefined SFTP operations.

See [“AS2 Predefined Operations” on page 192](#) for information on the predefined Applicability Statement 2 (AS2) operations.

See [“Cloud Deployment Operations” on page 210](#) for information on how to import Cloud Deployment services.

To create or edit a custom Operation

1. From the Integration Cloud navigation bar, click **Connect > Applications**.
2. Select an Application category and then click **Operations**.

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

3. From the Operations page for the selected Application, click **Add New Operation** to create a new Operation or select an Operation and click **Edit** to update an existing Operation. You can click **Delete** to delete an existing Operation, click **Show Signature** to view the input and output signature of the Operation, or click **Test** to test the Operation. The **Add New Operation** option is not available for some Applications.

Select an Operation and click **Show Signature** to view the input and output signature of the operation. The input and output fields cannot be edited. This option is available for all predefined and custom operations. Click the input and output fields

to view the field properties. From the **Input** or **Output** pane, click the  icon to copy a field. Depending on the context, you can either paste the field or the field path.

Click the **Test** option and in the test dialog box, specify the **Account** name and the **Input** data. If an operation does not have an input signature, the input fields are not displayed. The **Test** option is available for all predefined and custom operations.

Click **Run** to test the Operation and view the test results in the test results window.

Click the  icon beside **Result** if you want to go back to the test dialog box and enter another set of values. The last 5 test results are also displayed and are applicable only for the same test operation run, that is, if you close the test results window, you will not be able to view the test results later. Further, a test result appears in red color if the test run is unsuccessful and appears in green color for a successful test run.

Adding or Editing Operations

Use the **Operations** page to add, edit, or delete custom operations.

To add or edit a custom operation

1. From the Integration Cloud navigation bar, click **Connect > Applications**.
2. Select an Application from an Application category and then select **Operations**.
To use an Application, you are required to agree to the summary of terms. Click **I agree** to use the Application. Click **I do not agree** if you disagree with the summary of terms and do not want to use the Application. Click **Cancel** to go back to the **Applications** page.
3. From the Operations page for the selected Application, click **Add New Operation** to add a custom operation or click **Edit** to update an existing custom operation. The **Add New Operation** option is not available for some Applications.
4. On the **Select your <...> account** page, complete the following fields. Required fields are marked with an asterisk on the page.

Field	Description
Save as	Provide a name for the custom operation.
Description	Provide a description for the custom operation.
Select account	Select an Account created for the Application from the drop-down list. Only active or enabled Accounts are listed in the drop-down list.
Select functional area	Select the Application service from the drop-down list.

Note: This option is available only for certain Applications.

Field	Description
Enable dynamic account configuration at runtime	<p data-bbox="602 321 1341 632">For REST Application operations, you can now provide account configurations at <i>runtime</i> during integration execution. Select the Enable dynamic account configuration at runtime check box and then click Configure Fields to run the operation by passing authentication details that may be <i>different</i> from the authentication details configured in the Account configuration page. Dynamic authentication overwrites the Account configuration details.</p> <p data-bbox="602 674 1341 814">For example, suppose you have configured authentication details X for Account A. You can use different authentication details Y to run the service, by overwriting the authentication details X of Account A.</p> <div data-bbox="602 831 1365 1035" style="background-color: #f0f0f0; padding: 10px;"><p data-bbox="618 846 1325 1014">Note: Enable dynamic account configuration at runtime is currently supported only for custom REST Applications. Further, the option is enabled only after you have selected an Account in the Select account field.</p></div> <p data-bbox="602 1052 1341 1430">While creating an operation, if you select the Enable dynamic account configuration at runtime check box and click Configure Fields, the Dynamic Account Configuration Fields window appears. Select the fields you want to include in the operation input signature. When you select the fields, the operation input signature will be automatically updated based on your selections. Only the fields that are part of the signature gets mapped as the input request. You can override the field values at run time. Default Account configuration values will be used for other fields while creating the operation.</p> <div data-bbox="602 1446 1365 1890" style="background-color: #f0f0f0; padding: 10px;"><p data-bbox="618 1461 1341 1629">Note: ■ You cannot clear encrypted fields like Access Token and Consumer Secret. Further, you cannot enter a value for encrypted fields. You can enter the values for encrypted fields only while running the operation.</p><p data-bbox="691 1650 1341 1890">■ While running the operation, if you have provided details in both the Account configuration page and in the Dynamic Account Configuration Fields dialog box, Integration Cloud reads the data from the Account configuration page and merges the data with the data provided in the Dynamic Account</p></div>

Field	Description
	<p>Configuration Fields dialog box. Existing Account configuration details will not be modified.</p> <ul style="list-style-type: none"> ■ Details specified in the Dynamic Account Configuration Fields dialog box take precedence over the details specified in the Account configuration page. ■ If you pass incorrect values in the Dynamic Account Configuration Fields dialog box, service execution will fail while running the operation.

5. Click **Next**.

The **Select the Operation** page appears. If you do not want to add **Headers** and **Parameters**, skip to Step 9.

6. For a REST-based Application, select an operation and click **Headers** to add input Headers, if required. Integration Cloud displays the default HTTP transport headers for the operation, along with their default values. At run time, while processing the headers, Integration Cloud substitutes the values as necessary. In order to customize the headers, do the following:
 - a. Click **Add Header** to add a custom Header. Specify the **Header** name. To specify an optional default value for the header variable, click the **Default Value** box and type or paste the new value. If the variable is null in the input pipeline, this value will be used at run time. If the variable already has an existing default value defined, this value will overwrite the existing value at run time.
 - b. If headers appear in the signature, select **Active** to activate the headers in the signature.
 - c. To delete a custom header that you have added, select the header and click **Delete**.

Note: You cannot delete the required headers.

7. For a REST-based Application, select an operation and click **Parameters**. To customize the parameters, do the following:
 - a. Review the operation parameter details. Integration Cloud displays the parameter **Name** and **Description**, the **Data Type** used to represent the kind of information the parameter can hold, the parameterization **Type** of the request, and the default value needed to access the operation.
 - b. To specify a default value for the parameter, click the **Default Value** box and then type or paste the default value. The default value is used at run time.

You cannot add or delete request parameters. You can only modify the default value of a parameter. If parameters appear in the signature, select **Active** to activate the parameters in the signature.

8. For some Applications and Operations, for example, for the *CloudStreams Connector for Salesforce(R) Bulk v2 Data Loader Application* and *Create Job and Upload Job Data Operation*, Integration Cloud supports multipart request body. For example, if you want to create a user as well as upload a photo, the request has to be a multipart request where one part is an image file while the other part is a JSON request body.

Though application/x-www-form-urlencoded is a more natural way of encoding, it becomes inefficient for encoding binary data or text containing non-ASCII characters. The media type *multipart/form-data* is the preferred media type for request payloads that contain files, non-ASCII, and binary data. Integration Cloud supports this media type using which you can embed binary data such as files into the request body.

A multipart/form-data request body contains a series of parts separated by a boundary delimiter, constructed using Carriage Return Line Feed (CRLF), "--", and also the value of the boundary parameters. The boundary delimiter must not appear inside any of the encapsulated parts.

Each part has the Type, Name, Content-Type, and Part ID fields.

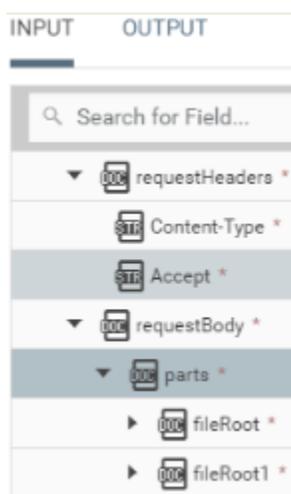
Example of a multipart request

```
--BOUNDARY
Content-Type: application/json
Content-Disposition: form-data; name="job"
{
  "object": "Contact",
  "contentType": "CSV",
  "operation": "insert"
}
--BOUNDARY
Content-Type: text/csv
Content-Disposition: form-data; name="content"; filename="content"
(Content of your CSV file)
--BOUNDARY--
```

Part Configuration

The **Part Configuration** window lists all the configured parts to be sent to the service provider. To view the **Part Configuration** window, on the **Select the Operation** page, select **Attachments**.

The configured parts appear in the input signature.



- **Name** - The **Name** field displays the name of the file part as documented in the SaaS provider API documentation.
 - **Content Type** - The **Content Type** field displays the content type of the file you are uploading.
 - **Type - Document** - Some back ends expect application/json or application/xml content in the part body. This represents a part where the content of the part is of type application/json or application/xml. **File** - Represents a binary part of a multipart/form-data payload where the content of the part is binary or the content of the file itself. For file upload kind of use cases, this part is the main or mandatory part required by the service provider.
9. On the **Select the Operation** page, select the operation to be performed and then click **Next**. Depending on whether the operation is Complex, Simple, or Dynamic, the **Business Object** or the **Interactions** page appears. Examples of **Business Objects** are Contact, Account, and so on and examples of **Interactions** are Create, Update, Upsert, Delete, and so on.

Note: For some operations, Integration Cloud displays appropriate Business Object pages, for example, for the createMultiple and updateMultiple operations in the Salesforce v42 Application, or Interactions (sub-operations) pages, for example, for the Batch and ChangeSet operations in the OData 4.0 Application, depending on whether the selected operation requires metadata, such as a business object, fields, and data types of fields. You can add or edit multiple business objects in a single request. You can also add or edit interactions and then associate those interactions with business objects in a single request. Interactions or multiple business objects will be executed in the same sequence as they appear in the table. You can drag and drop the interactions or multiple business objects to change the order in which they will be executed.

Different pages appear based on the scenarios mentioned in the following *Single or Multiple Interactions with Single or Multiple Business Objects with dependencies* section.

Single or Multiple Interactions with Single or Multiple Business Objects with dependencies

- **Single operation has a single Business Object** - An operation has only a single business object. The operation has neither multiple interactions nor has records. An example of a single operation and a single object can be a "create" operation that contains only the "contact" business object.
- **Single operation has multiple Business Objects** - A single operation has multiple business objects. An example of a single operation with multiple business objects can be a "create" operation that contains two business objects, "contact" and "account".
- **Single operation has multiple Business Objects with dependencies** - A single operation has multiple business objects and some of the business objects may have dependencies on other business objects.
- **Multiple Interactions have multiple Business Objects** - Multiple Interactions have multiple business objects. For example, the "create" and "update" Interactions can act on the "account" and "contact" business objects respectively.
- **Multiple Interactions have multiple Business Objects with dependencies** - Multiple Interactions have multiple business objects and some of the business objects may have dependencies on other business objects. For example, the "create" and "update" Interactions can act on the "account" and "contact" business objects respectively.

Further, for some operations, for example, for the *Retrieve Contained Or Derived Entity* operation in the OData 4.0 Application, Integration Cloud displays nested, hierarchical, or multi-level business objects if the operation is designed to support nested business objects. You can expand the nested business objects to display the child-level objects.

10. Select the Business Object to be associated with the operation you have selected in the previous step and then click **Next**. For some operations, you must select the Interaction and then the Business Object. Business Objects and Interactions appear only for certain Applications and operations.
11. In the **Data Fields** page, select the data fields for the Business Object you have chosen in the previous step and then click **Next**. Data fields appear only for certain Applications and operations. Mandatory data fields for the Business Object are selected by default and cannot be cleared.

Note: For some operations of certain Applications, for example, Coupa, you can add your own fields. Such fields are called custom fields. Custom fields are marked by *custom* on the page. You can add, edit, or delete only custom fields. Click the **+** icon to add a custom field. The **+** icon appears only if you are allowed to add custom fields for the selected operation. After you have added a custom field, you can click on the custom field to edit the field properties.

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

- **Unchecked** - For unchecked complex fields, only the mandatory child fields are selected but those fields will not be added to the signature, unless you select the parent field.
- **Checked** - If a complex field is selected by default, then only the mandatory child fields are selected. You can select the optional fields, if required. If you select a complex field, then all the child fields are selected.

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

12. In the **Confirm Operation** page, verify the details. You can click the **Data Fields** link to view the data fields you have added. For some operations, you can click the Business Object link to view the data fields added.
13. Click **Finish** to create the custom operation.

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

AS2 Predefined Operations

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

Operation	Description
receive	Receives an AS2 message from a recipient.
send	Sends an AS2 message to a recipient's defined endpoint.

receive

Receives an AS2 message from a recipient.

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

- [“Configuring the Auto Detect Option” on page 198](#)
- [“Creating an Endpoint URL” on page 198](#)

Input Parameters

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

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

Output Parameters

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

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

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

stream **Object** Raw output stream received by an application.

headers **Document** AS2 message headers.

AS2-To **String** AS2 ID of the recipient.

AS2-From **String** AS2 ID of the sender.

Message-ID **String** Message ID of the inbound message.

AS2-Version **String** AS2 protocol version used for the inbound message.

Content-Type **String** MIME content type of the inbound message.

EDIINT-Features **String** Optional features supported by the application.

Receipt-Delivery-Option **String** Optional. Sender's asynchronous MDN endpoint URL.

Disposition-Notification-To **String** Optional. Acknowledgment request for the inbound message.

	<i>Disposition-Notification-Options</i>	String Optional. Acknowledgment request to be signed for the inbound message.
<i>payload</i>		Document Extracted payload that you receive.
	<i>stream</i>	Object Extracted payload stream.
	<i>contentType</i>	String Content type assigned to the payload.
	<i>headers</i>	Document Headers assigned to the payload.
<i>attachments</i>		Document Array Optional. Attachments you receive with an inbound message, if any.
	<i>stream</i>	Object Output stream of the attachment.
	<i>contentType</i>	String Content type assigned to the attachment.
	<i>headers</i>	Document Headers assigned to the attachment.
<i>response</i>		Document Sent MDN or received asynchronous MDN response.
	<i>status</i>	String Status of the sent or received MDN.
	<i>statusMessage</i>	String Status message of the sent or received MDN.
	<i>receipt</i>	Document Optional. Sent or received MDN.
	<i>stream</i>	Object Object stream of the sent or received MDN.
	<i>headers</i>	Document Headers of the sent or received MDN.

<i>AS2-To</i>	String AS2 ID of the recipient.
<i>AS2-From</i>	String AS2 ID of the sender.
<i>Message-ID</i>	String Message ID of the inbound or outbound MDN.
<i>AS2-Version</i>	String AS2 protocol version used for the inbound or outbound MDN.
<i>Content-Type</i>	String MIME content type of the inbound or outbound message.

send

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

Input Parameters

<i>data</i>	Document Payload you want to send.
<i>stream</i>	Object java.io.InputStream object you want map from EDI or XML data.
<i>contentType</i>	<p>String Content type to assign to an outbound message. The following options are available by default:</p> <ul style="list-style-type: none"> ■ application/edi-x12

		<ul style="list-style-type: none"> ■ application/edifact ■ application/xml <p>You can also type a custom value.</p>
	<i>otherHeaders</i>	Document Optional. <i>key</i> and <i>value</i> strings of the header for an outbound message.
<i>attachments</i>	Document Array	Optional. Attachments for a message, if any.
	<i>stream</i>	Object java.io.InputStream object you want add to the attachment.
	<i>contentType</i>	String Content type of the attachment. For example, <code>application/zip</code> if the attachment is a <code>.zip</code> file.
	<i>otherHeaders</i>	Document Optional. <i>key</i> and <i>value</i> strings of the header you want to add to the attachment.
<i>customHeaders</i>	Document	Optional. Custom headers you want to include in an AS2 message.
	<i>key</i>	String Key for the custom header.
	<i>value</i>	String Value for the customer header.
Output Parameters		
<i>status</i>	String	Status of an outbound message.
<i>statusMessage</i>	String	Processing status of an outbound message.
<i>request</i>	Document	AS2 message sent to a recipient.
	<i>stream</i>	Object AS2 message stream.
	<i>headers</i>	Document Optional. AS2 message headers.
	<i>AS2-To</i>	String AS2 ID of the recipient.
	<i>AS2-From</i>	String AS2 ID of the sender.

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

<i>AS2-From</i>	String AS2 ID of the sender.
<i>Message-ID</i>	String Message ID of the inbound MDN.
<i>AS2-Version</i>	String AS2 protocol version used for the inbound MDN.
<i>Content-Type</i>	String MIME content type of the inbound message.

Configuring the Auto Detect Option

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

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

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

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

Creating an Endpoint URL

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

To create an endpoint URL

1. Create an orchestrated integration. For instructions, see [“Orchestrated Integrations” on page 235](#). Ensure that you specify a signature with *contentStream* and *node* as input parameters of type **Object**. For instructions, see [“Pipeline and Signatures” on page 253](#). Ensure you specify a name for the integration.

Alternatively, you can define a Document Type as a signature with *contentStream* and *node* as input parameters of type **Object**. For instructions, see [“Document Types” on page 420](#).

2. Configure the AS2 application with the **receive** operation to work with a new or existing account, or select the **Auto Detect** option for the application. For information about accounts and configuring an account using the **Auto Detect** option, see [“Adding or Editing Accounts” on page 91](#) and [“Configuring the Auto Detect Option” on page 198](#) respectively.
3. Map the *contentStream* and *node* parameters of the Pipeline Input signature defined in step 1 with the AS2 application's *receiveInput* parameter. For more information, see [“Pipeline and Signatures” on page 253](#).
4. Select the **Enable Integration to be invoked over HTTP** option for the integration. For more information, see [“Integration Details” on page 405](#).

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

FTP Predefined Operations

The following predefined FTP operations are available:

Operation	Description
getFile	Retrieves a file from a remote FTP server.
listFiles	Returns a list of file names in a specified remote directory. If path is not specified, the operation retrieves the file listing of the current remote directory. The operation also retrieves additional details such as permissions and ownership information.
deleteFiles	Delete file(s) from a remote FTP server.
putFile	Transfers a file to a remote FTP server.
renameFile	Renames a file on a remote FTP server.

getFile

Retrieves a file from a remote FTP server.

Input Parameters

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

Output Parameters

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

listFiles

Returns a list of file names in a specified remote directory. If path is not specified, the operation retrieves the file listing of the current remote directory. The operation also retrieves additional details such as permissions and ownership information.

Input Parameters

<i>remotePath</i>	String Optional. Absolute or relative path of the remote directory. If <i>remotepath</i> is not specified, the listFiles operation retrieves the directory listing of the current remote directory. You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path.
<i>listFilter</i>	String Optional. Filter that specifies the names of the files to include in the list (for example, *.txt).

Output Parameters

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

deleteFiles

Delete file(s) from a remote FTP server.

Input Parameters

<i>remotePath</i>	String Optional. Absolute or relative path of the remote directory. If <i>remotePath</i> is not specified, the deleteFiles operation deletes the directory listing of the current remote directory. You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path.
<i>deleteFilter</i>	String Optional. Filter that specifies the names of the files to be deleted (for example, *.txt).

Output Parameters

<i>filesDeleted</i> []	String List List of deleted files that match the <i>deleteFilter</i> .
<i>filesNotDeleted</i> []	String List List of files not deleted.
<i>statusCode</i>	String Standard FTP protocol status code.
<i>statusmsg</i>	String Standard FTP protocol status message.

putFile

Transfers a file to a remote FTP server.

Input Parameters

<i>remoteFile</i>	String The name of the remote file.
<i>transferType</i>	String FTP file transfer mode (<code>ascii</code> or <code>binary</code>). The default is <code>ascii</code> .
<i>writeOption</i>	String Optional. Indicates whether to send a <code>STOR</code> or a <code>STOU</code> (Store as Unique File) command to the remote FTP server. Set to: <ul style="list-style-type: none"> ■ <code>true</code> to send a <code>STOU</code> (Store as Unique File) command. ■ <code>false</code> to send a <code>STOR</code> command. This is the default.
<i>contentStream</i>	java.io.InputStream, byte[], or String Data to be transferred to the remote file.

Output Parameters

<i>statusCode</i>	String Standard FTP protocol status code.
<i>statusMessage</i>	String Standard FTP protocol status message.

Usage Notes

Some FTP servers do not support “putting” a unique file. When using the `putFile` operation to put a unique file to an FTP server that does not support putting a unique file, you may encounter the following error:

```
com.wm.app.b2b.server.ServiceException: 500 'STOU': command not understood.
```

renameFile

Renames a file on a remote FTP server.

Input Parameters

<i>oldFileName</i>	String Fully qualified name of the file you want to rename (for example, <code>temp/oldfilename.txt</code>).
<i>newFileName</i>	String Fully qualified name of the new file (for example, <code>temp/newfilename.txt</code>).

Output Parameters

StatusCode **String** Standard FTP protocol status code.

StatusMessage **String** Standard FTP protocol status message.

SFTP Predefined Operations

The following predefined SFTP operations are available:

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

Operation	Description
rmdir	Deletes one or more remote directories on the SFTP server.
symlink	Creates a symbolic link between the old path and the new path of a file.

cd

Changes the working directory on the remote SFTP server.

Input Parameters

path **String** Absolute or relative path of the directory that you want as the working directory on the remote SFTP server.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

chgrp

Changes the group ownership of one or more remote files.

Input Parameters

groupId **String** Numeric group identifier of the group to which you want to transfer ownership of the remote files.

path **String** Absolute or relative path of the remote files.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

chmod

Changes permissions of one or more remote files.

Input Parameters

<i>mode</i>	String The permission mode to apply to the remote file (for example, <i>777</i>).
<i>path</i>	String Absolute or relative path of the remote files.

Output Parameters

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

chown

Changes the owning user of one or more remote files.

Input Parameters

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

Output Parameters

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

get

Retrieves a file from a remote SFTP server.

Input Parameters

remoteFile **String** Absolute or relative path of the remote file.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

contentStream **Object** A java.io.InputStream object.

ls

Retrieves the remote directory listing of the specified path. If path is not specified, the ls service retrieves the file listing of the current remote directory. The ls service also retrieves additional details such as permissions and ownership information.

Input Parameters

path **String** Optional. Absolute or relative path of the remote directory. If no *path* is specified, the ls service retrieves the directory listing of the current remote directory.

You can use the wildcard characters asterisk (*) and question mark (?) after the last slash mark (/) to view all remote directories that match the specified path.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

dirList **Document** List of directories matching the pattern specified in the *path* parameter. This document has the following parameters:

fileName: **String** Specifies the name of the remote file. .

fileSize: **String** Specifies the size of the remote file.

permissions: **String** Specifies the access permission of the file (read, write, or execute).

lastAccessTime: **String** Specifies the time when the file was last accessed.

lastModifiedTime: **String** Specifies the time when the file was last modified.

uid: **String** Specifies the user ID of the file owner.

gid: **String** Specifies the group ID associated with the file.

longName: **String** Specifies the long name of the *ls* entry. It contains all the parameters separated by a space.

mkdir

Creates a new remote directory.

Input Parameters

path **String** Absolute or relative path of the remote directory where you want to create a new directory.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

put

Transfers a file to a remote SFTP server.

Input Parameters

contentStream **java.io.InputStream** Data to be transferred to the remote file.

remoteFile **String** Absolute or relative path of the remote file to which the *contentStream* would be written based on the *mode*.

<i>mode</i>	String Optional. Specifies how the local file is to be transferred to the remote SFTP server. Set to: <ul style="list-style-type: none">■ <code>overwrite</code> to overwrite the contents of the remote file with the contents of the <i>contentStream</i> . This is the default.■ <code>append</code> to append the entire contents of the <i>contentStream</i> to the remote file.■ <code>resume</code> to resume writing the contents of the <i>contentStream</i> to the remote file from the point the writing was stopped during previous SFTP sessions.
-------------	---

Output Parameters

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

pwd

Displays the remote working directory in the SFTP server.

Input Parameters

None.

Output Parameters

<i>returnCode</i>	String Standard SFTP protocol return code.
<i>returnMsg</i>	String Text message describing the return code.
<i>path</i>	String Absolute or relative path of the working directory on the remote SFTP server.

rename

Renames a file or directory on a remote SFTP server.

Input Parameters

oldPath **String** Fully qualified name of the file you want to rename (for example, `temp/oldname.txt`).

newPath **String** New fully qualified name for the file (for example, `temp/newname.txt`).

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

rm

Deletes one or more remote files on the SFTP server.

Input Parameters

path **String** Absolute or relative path of the file you want to delete.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

rmdir

Deletes one or more remote directories on the SFTP server.

Input Parameters

path **String** Absolute or relative path of the directory you want to delete.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

Usage Notes

The remote directories that you want to delete must be empty.

symlink

Creates a symbolic link between the old path and the new path of a file.

Input Parameters

oldPath **String** Old path of the file for which you want to create a symbolic link.

newPath **String** New path of the file to which the symbolic link should point.

Output Parameters

returnCode **String** Standard SFTP protocol return code.

returnMsg **String** Text message describing the return code.

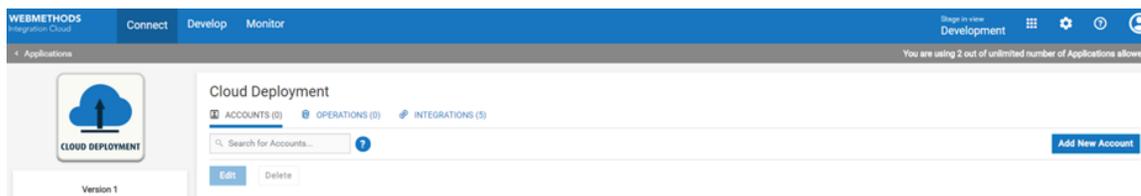
Cloud Deployment Operations

Integration Cloud Integrations can invoke Cloud Deployment solution webMethods Integration Server Java and Flow services for the same tenant, by using the “[Cloud Deployment](#)” on page 112 Application. You can use these services along with other Integrations and services, both in Orchestration and Point-to-Point Integrations.

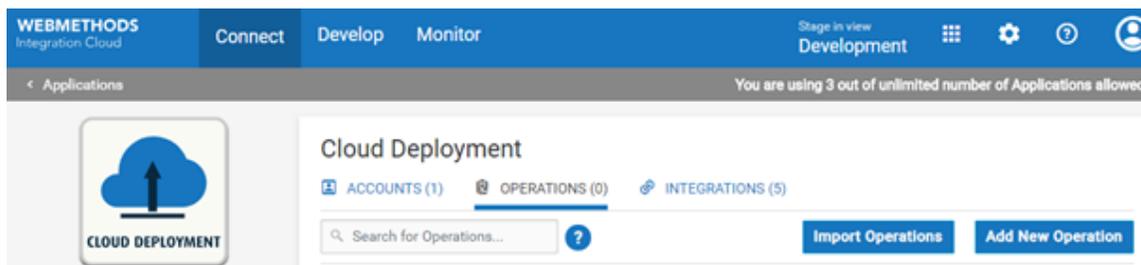
Note: The Cloud Deployment Application appears under the **Predefined Applications** category. You must have the Cloud Deployment license to use this Application.

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

1. From the Integration Cloud navigation bar, click **Connect > Applications > Predefined Applications > Cloud Deployment**. The **Cloud Deployment** Application details page appears.



2. Select **Accounts** > **Add New Account** to create a new “Cloud Deployment account” on page 112.
3. Click **Operations**.



4. Click **Import Operations** if you want to import multiple Java or Flow services. If any of the services have a *Document Reference* in its input/output signature, then those services will not be imported. To import those services, select **Add new Operation** and import those services individually.
5. If you click **Import Operations**, the **Import Operations** window will appear. Select the cloud deployment account, and then select the services that you want to import in Integration Cloud.
6. Click **Import**. The services will appear on the **Operations** list page for the Cloud Deployment Application.

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

Upgrade

The upgrade feature in Integration Cloud allows you to upgrade assets, for example, Accounts, Operations, and the associated Integrations which uses those assets from a lower version to a higher version. When an upgrade is available for a version, the upgrade notification text: *Upgrade available for this version*, appears beside the relevant Application on the **Applications** screen, else the message *This is the latest version* appears. Currently, upgrade functionality is available only for the Salesforce CRM Application.

Note: Users who have the **Upgrade** permission under **Settings** ⚙️ > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Application** can perform the upgrade task.

If an upgrade is available for a version, and if you click the **Add New Account** button on the Application specific Accounts screen, a dialog box appears where you can either select **Upgrade** to start the asset upgrade process or select **Skip** to add a new Account.

If you click the **Upgrade** button, the upgrade confirmation screen appears which displays the number of assets (Accounts, Operations, and the associated Integrations) that will be upgraded. The screen also displays details of all the conflicting assets. Conflicting assets are those assets that exist in the higher version with the same name.

On the upgrade confirmation screen, select **Skip** if you do not want to upgrade the conflicting assets to the higher version. You can also select to **Overwrite** if you want the conflicting assets in the higher version to be replaced with the lower version assets. Here, the higher version assets will be deleted and will be replaced with the lower version assets.

Note: The upgrade process upgrades Accounts from a lower version to a higher version only in the Development stage. If you want to reflect the upgraded Accounts in other stages in the higher version, you must *manually* configure the Accounts in the different stages from the Account configuration screen, and then **Pull** the Integration in the respective stages.

Integration Cloud performs the following tasks if you click **Upgrade** on the upgrade confirmation page:

- Migrates all the Accounts from the lower version to the higher version only in the development stage.
- Migrates all the custom Operations and predefined Operations to the latest version.
- Updates those Integrations which uses the upgraded Accounts and Operations.
- Updates Integrations only in the development stage.
- Displays the upgraded Accounts, Operations, and Integrations in the Integration Cloud **Audit Log**.
- Displays the upgrade results along with a list of all the modified Accounts, Operations, and Integrations.
- Displays a message in case of an upgrade failure and performs rollback of the Accounts, Operations, and Integrations in case of an error in the upgrade process.

Creating and updating SOAP Applications

The SOAP Application enables you to access third party Web Services hosted in the cloud. The SOAP Application uses a WSDL to create consumer operations.

The following features are supported for SOAP Applications:

- A SOAP Application implementation that follows the WS-I Basic Profile 1.1 specification.

- SOAP Applications can be created by uploading a WSDL file or by using a valid WSDL URL that can be accessed over a network.
- SOAP Applications can be created with WSDLs that are annotated with WS-Security Policy/Policies.
- SOAP Applications with SOAP version 1.1 and 1.2 and Style/Use as Document/Literal and RPC/Literal (RPC/Encoded model is not supported for SOAP version 1.2).
- The following SOAP Binding types are supported:
 - SOAP over HTTP.
 - SOAP over HTTPS.
- Authentication type: HTTP Basic Token.

SOAP Applications have the following restrictions:

- The WSDL and associated schema(s) must be accessible through a publicly or locally accessible URL.
- Only WSDLs with WS-Security policies are supported. Any other policies, for example, WS-Addressing, WS-Reliable Messaging, and so on, are not supported. If you create SOAP Applications with WSDLs having non-WS-Security Policies, exceptions may appear while executing Integrations.
- Manual addition of WS-Security Policies in a SOAP Application is not supported. SOAP Applications with WS-Security can be created with only policy-annotated WSDLs, that is, WSDLs that already have WS-Security Policies annotated in them.
- SOAP over JMS is not supported.
- Only Basic Authentication is supported. Other authentication types such as Digest, NTLM, and Kerberos are not supported.
- You will not be able to attach or upload a file while executing an Integration.

To add a SOAP Application

1. From the Integration Cloud navigation bar, click **Connect > Applications > SOAP Applications > Add New Application**.
2. Provide a name and description of your SOAP Application. The description you enter here will appear in the **SOAP Applications** page. Required fields are marked with an asterisk on the screen.
3. Click **Browse** next to the **Application Icon** if you want to select a different icon for your SOAP Application. The icon must be a PNG file and the size cannot exceed 50 KB, else the default image is displayed.
4. Click **Next** and specify the **WSDL Source**. Select **URL** if you want to specify the URL of the WSDL. The URL should begin with http:// or https://. The URL is used to retrieve the WSDL for the SOAP Application. Select **File** and then click **Browse** if you want to select the WSDL from your local file system. You can click the  icon beside

the **Browse** button if you want to add separate elements of a service definition after import, such as WSDLs or XSDs, to the primary WSDL.

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

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

Editing SOAP Applications

From the **SOAP Applications** page, click the SOAP Application link, and then click **Edit Application**. You can change the **Description** and the **Application Icon**.

In the Application details page, **Update WSDL** section, select **No, keep existing WSDL** if you do not want to modify the WSDL URL or the WSDL file. Select **Yes, override WSDL** if you want to specify a new WSDL URL or upload a new WSDL file in the **WSDL Source** section.

Confirm the updated Application. After you click **Finish**, the **Update SOAP Application** window appears, which provides a summary of the impacted Accounts, Operations, and Integrations. Click **Update** to update the SOAP Application. Updating the WSDL may result in addition or removal of Operations or fields in the Input/Output signature of an Operation. This may lead to incorrect mappings if you have used that Operation in an Integration.

Note: To delete a SOAP Application, click **Delete Application**.

SOAP Signature

All SOAP Application operations have an identical input and output signature with the exception of the variables used to represent the input and output messages. For information about how a SOAP Application operation represents the input and output messages in the signature, see *How a SOAP Application Operation represents the Input and Output Messages*.

How a SOAP Application Operation represents the Input and Output Messages

How a SOAP Application operation represents the contents of the input and output message in the signature depends on the style/use of the binder for the SOAP Application operation.

- For a SOAP Application operation that uses a style/use of Document/Literal:
 - The input signature contains an optional document reference to a document type created to represent the operation input message. At run time, if you do not specify any input for the document reference variable or any of its child variables, Integration Cloud sends an empty SOAP body in the SOAP message.

- The output signature contains a document reference to a document type created to represent the operation output message. This document reference is conditional and is only returned by the SOAP Application operation if the SOAP Application operation executes successfully. If returned at run time, this document reference contains the response from a successful invocation of a SOAP Application operation. If the SOAP Application operation receives a SOAP fault, it is converted to an exception that can be caught with the try catch block in an Orchestrated Integration.
- For a SOAP Application operation that uses a style/use of RPC/Encoded or RPC/Literal:
 - The input signature contains variables that represent the top-level elements in the operation input message. All of these variables are optional. At run-time, if you do not specify any input for the variable (or variables) that represent the input message, Integration Cloud sends an empty SOAP body in the SOAP message.
 - The output signature contains variables that represent the top-level elements in the operation output message. All of these variables are conditional and are only returned by the SOAP Application operation if the SOAP Application operation executes successfully. If returned, these variables contain the response from a successful invocation of a SOAP Application operation.

Input Parameters

transportHeaders

Document Optional. Transport-specific header fields that you want to explicitly set in the request. Specify a key in *transportHeaders* for each header field that you want to set, where the key's name represents the name of the header field and the key's value represents the value of that header field.

The names and values supplied to *transportHeaders* must be of type String. For information about using *transportHeaders* with HTTP/S requests including a description of the default behavior, see *Setting Transport Headers for HTTP/S*.

Output Parameters

transportInfo

Document Conditional. Headers from response and request messages.

The contents of the *transportInfo* vary depending on the actual transport (HTTP or HTTPS) used.

transportInfo contains the following keys:

Key	Value
-----	-------

<i>requestHeaders</i>	<p>Document Conditional. Header fields from the request message. The contents of the <i>requestHeaders</i> document are not identical to <i>transportHeaders</i> used as input. The transport can add, remove, or alter specific headers while processing the request.</p> <p>Whether or not the SOAP Application operation returns the <i>requestHeaders</i> parameter depends on the success or failure of the operation. In the case of failure, the point at which the failure occurs determines the presence of the <i>requestHeaders</i> parameter. For more information, see <i>Transport and Exceptions Returned by a SOAP Application Operation</i>.</p> <p>For the HTTP or HTTPS transports, the <i>requestHeaders</i> parameter will not contain any HTTP headers that the transport mechanism added or modified when sending the request.</p>
<i>responseHeaders</i>	<p>Document Conditional. Header fields from the response. Each key in <i>responseHeaders</i> represents a field (line) of the response header. Key names represent the names of header fields. The keys' values are Strings containing the values of the fields.</p> <p>Whether or not the SOAP Application operation returns the <i>responseHeaders</i> parameter depends on the success or failure of the operation. In the case of failure, the point at which the failure occurs determines the presence of the <i>responseHeaders</i> parameter. For more information, see <i>Transport and Exceptions Information Returned by a SOAP Application Operation</i>.</p> <p>For the HTTP or HTTPS transports, the <i>responseHeaders</i> parameter contains any HTTP/HTTPS headers present in the response.</p>
<i>status</i>	<p>String Conditional. Status code from the request, returned by the underlying transport.</p>

For more information about status codes and status messages returned by a SOAP Application operation, see *Transport and Exceptions Information Returned by a SOAP Application Operation*.

statusMessage

String Conditional. Description of the status code returned by the transport.

For more information about status codes and status messages returned by a SOAP Application operation, see *Transport and Exceptions Information Returned by a SOAP Application Operation*.

Setting Transport Headers for HTTP/S

When creating a service that executes a SOAP Application operation, you can pass transport header information directly into the SOAP Application operation by passing name/value pairs in to the *transportHeaders* input parameter. When creating the SOAP request, Integration Cloud adds a transport header for each name/value pair.

Keep the following information in mind when setting *transportHeaders* for an HTTP/S request:

- Specify a key in *transportHeaders* for each header field that you want to set, where the key's name represents the name of the header field and the key's value represents the value of that header field.
- The names and values supplied to *transportHeaders* must be of type String. If a transport header has a name or value that is not of type String, the header will not be included in the message.
- For any header name/value pair supplied in *transportHeaders* for an HTTP/S request, Integration Cloud simply passes through the supplied headers and does not perform any validation for the headers beyond verifying that the name and value are of type String.
- If you do not set *transportHeaders* or do not specify the following header fields in *transportHeaders*, Integration Cloud adds and specifies values for the following standard header fields:
 - Accept
 - Authorization
 - Connection
 - Content-Type
 - SOAPAction (Added when *soapProtocol* is SOAP 1.1 only)
 - User-Agent

Note: Pass in the preceding headers to *transportHeaders* only if you are an experienced SOAP Application developer. Incorrect header values can result in failure of the request.

- For a SOAP Application operation, Integration Cloud sets the value of the `Host` header and overwrites any supplied value.
- If you specify `Authorization` in *transportHeaders*, the values specified for the *auth/transport* document and its children will not be used in the `Authorization` header.
- If you specify `Content-Type` in *transportHeaders* and the SOAP Protocol is SOAP 1.2, Integration Cloud ignores the value of `soapAction` obtained from the WSDL used to create the SOAP Application operation.
- If you specify the `SOAPAction` header in *transportHeaders* and the SOAP Protocol is SOAP 1.1, Integration Cloud ignores the value of `SOAPAction` obtained from the WSDL used to create the SOAP Application operation.
- Integration Cloud sets the value of `Content-Length` automatically and overwrites any value passed in to *transportHeaders*.
- Integration Cloud automatically adds the `Cookie` header to the HTTP header and supplies any cookies established between Integration Cloud and the HTTP server with which it is interacting. If you supply the `Cookie` header to *transportHeaders*, Integration Cloud prepends the values you supply to the already established `Cookie` header value.
- The following headers are considered to be standard and require the specified capitalization: `Accept`, `Authorization`, `Connection`, `Content-Type`, `Cookie`, `Host`, `SOAPAction`, `User-Agent`.

Important: Using capitalization other than that which is specified results in undefined behavior.

Important: Supplying duplicate entries for any standard header results in undefined behavior.

Transport and Exceptions Returned by a SOAP Application Operation

The transport information, such as headers, status codes, and status messages, returned by a SOAP Application operation varies depending on the following:

- The transport used to send and receive the SOAP message
- The success or failure of the SOAP Application operation
- The point at which failure occurs
- The message exchange pattern (MEP) for the operation

Note: Transport information is returned in the *transportInfo* output parameter.

If the SOAP Application operation receives a SOAP fault, it is converted to an exception that can be caught with the try catch block in an Orchestrated Integration.

REST Applications

REST (Representational State Transfer) is an architectural style that requires web applications to support the HTTP GET, POST, PUT, and DELETE methods and to use a consistent, application-independent interface.

Endpoint URL

The endpoint of an API is a unique URL, which represents an object or collection of objects. The endpoint is a reference to a URI that accepts web requests. It is the login endpoint URL to initiate communication with the SaaS provider. To get the endpoint, go through the SaaS provider documentation available on the internet. For example, <https://api.twitter.com/1.1/> is the Twitter endpoint.

Authentication Type

Every back end provides its own Authentication mechanism to provide authorized access to its APIs. You need to get the authentication details from the SaaS provider documentation. For example, for Twitter, go to <https://apps.twitter.com>, create a new application, and then get the credentials. For Twitter, the authentication is OAuth V1.0a, which you can get from <https://apps.twitter.com>.

Resource

A resource refers to some object or set of objects that are exposed at an API end point. It is a representation of a thing (a noun) on which the REST APIs (verbs) operate. A resource has a type, one or more parameters, and some standard operations that allow you to manipulate or retrieve it from a remote location if you know its endpoint URL. Each resource derives its path from the namespace of the resource. For example, if the REST resource is named `myREST.myRESTResource`, the path is `"/myREST.myRESTResource"`.

Action

Actions are tasks that act on a Resource. You must create at least one Action for a Resource after you have created the Resource. You can add a Method, Request Parameters, Request and Response Headers, and a Request and Response body to an Action.

HTTP Method

The primary or most-commonly-used HTTP verbs (or methods, as they are properly called) are POST, GET, PUT, and DELETE. These correspond to create, read/retrieve, update, and delete (or CRUD) operations, respectively. You use the following HTTP methods to map the CRUD operations to HTTP requests. In a REST request, the resource that you are working with is specified in the URL – Uniform Resource Locator. The URL is a special case of the URI – Uniform Resource Identifier.

- **GET** - Used to read or retrieve a representation of a resource. For example, GET <endpointurl>/addresses/2 will retrieve an address with an ID of 2.
- **POST** - Creates a resource. For example, POST <endpointurl>/addresses will create a new address.
- **PUT** - Updates an existing resource. For example, PUT <endpointurl>/addresses/3 will modify the address with an ID of 3.
- **DELETE** - Used to delete a resource identified by a URI. For example, DELETE <endpointurl>/addresses/4 will delete an address with an ID of 4.

Resource	GET	PUT	POST	DELETE
http://example.com/api/resource/	Lists details and perhaps URIs of the resources in this collection.	Replaces the entire collection.	Creates a new item in the collection.	Deletes a collection.
http://example.com/api/resource/123/	Retrieves a specific item in the collection.	Updates the item in the collection and possibly creates an item if it does not exist.	Creates a new item in the collection.	Deletes an item from the collection.

Headers and Parameters

REST is not a standard in itself but instead makes use of the HTTP standard. HTTP headers allow the client and the server to pass additional information with the request or the response. For example, the *Accept* and *Content-Type* HTTP headers can be used to describe the content being sent or requested within an HTTP request. The client may set *Accept* to *application/json* if it is requesting a response in JSON or *application/xml* if it is requesting a response in XML, that is, when sending data, setting the *Content-Type* to *application/xml* tells the client that the data being sent in the request is XML.

REST calls (requests) and responses are sent over the HTTP protocol, hence REST requests are in the form of URLs that point to the resource(s) on the server. Required parameters are attached to the end of the URL. For example, in the resource URL `http://<name>.com/user/789`, `user` is the resource and `789` is the parameter that is passed to the URL of the resource. You can use any REST client to make REST calls.

REST parameters specify the variable parts of your resources, that is, the data that you are working with. QUERY parameters are the most common type of parameters, which is appended to the path of the URI when submitting a request. For example, `https://api.twitter.com/1.1/users/show.json?screen_name=twitterdev` is an example of a QUERY parameter URI where `screen_name` is the name of the parameter and `twitterdev` is the value of the parameter.

HTTP Status Codes

HTTP Status Codes indicate the status of the HTTP response:

- 1XX - Informational
- 2XX - Success
- 3XX - Redirection
- 4XX - Client error
- 5XX - Server error

Creating and updating REST Applications

These screens allow you to define a REST Application, define Resources and Actions, and then create a REST Application. See [“REST Applications” on page 219](#) for conceptual information on REST Resources, HTTP Methods, HTTP Status Codes, HTTP Headers, and Parameters.

To create a REST Application

1. From the Integration Cloud navigation bar, click **Connect > Applications > REST Applications > Add New Application**.
2. In the **Define Application Details** page, complete the following fields. Required fields are marked with an asterisk on the screen.

Field	Description
Name	Type a name for the REST Application.
Description	Type an optional description for the REST Application. The description you enter here will appear in the Applications page.
Default Endpoint URL	Specify the Endpoint for the Application. It is the login endpoint URL to initiate communication with the SaaS provider. To get the end point, see the back end documentation available on the internet for the SaaS provider.
Authentication Type	Every back end provides its own authentication mechanism. Get the authentication details from the back end documentation and select the supported Authentication Type from the drop-down list.
Application Icon	Click Browse and select another icon for the REST Application, if necessary.

- Click **Next**.

The **Define Resources and Actions** page appears.

- In the **Define Resources and Actions** page, click **Add Resource** to create a new REST Resource.

The **Add Resource** dialog box appears. In the **Add Resource** dialog box, complete the following fields:

Field	Description
Name	Type the Resource name.
Path	<p>Type the path to the Resource. The Resource path is relative to the endpoint specified. Each REST Resource derives its path from the namespace of the REST Resource. For example, if the REST Resource is named myREST.myRESTRResource, the path is “/myREST.myRESTRResource”.</p> <p>You can define dynamic parameters in the resource path by enclosing each parameter within { } brackets. For example, to get the employee data corresponding to a dynamic parameter called employeeID, specify the resource path as /employee/{employeeID}. To get item information from a particular department in a store, specify the resource path as /store/{departmentID}/{itemID}.</p>

Note: While adding an **Action**, if your Resource path contains { } brackets, for example, /user/{userID}, you must add a request parameter having the same name, that is, "userID", and set the **Parameter Type** to **URI_CONTEXT**.

- Click **Add** to create the Resource. You can **Edit** or **Delete** the Resource from the **Define Resources and Actions** page.
- In the **Define Resources and Actions** page, select the Resource and click **Add Action**.

Note: Every Resource must have an Action associated with it.

In the **Add Action to Resource** dialog box, complete the following fields:

Field	Description
Method	<p>Select an HTTP Method.</p> <ul style="list-style-type: none"> ■ GET - Reads or retrieves a representation of a resource. For example, GET <endpointurl>/addresses/2 will retrieve an address with an ID of 2. ■ PUT - Updates an existing resource. For example, PUT <endpointurl>/addresses/3 will modify the address with an ID of 3. ■ POST - Creates a resource. For example, POST <endpointurl>/addresses will create a new address. ■ DELETE - Deletes a resource identified by an URI. For example, DELETE <endpointurl>/addresses/4 will delete an address with an ID of 4.
Description	Type an optional description for the Action.
Request Parameter	<p>You can set parameters that become part of the outgoing request. Parameters specify the variable parts of your resources. Click Add Parameter to add a parameter to the request. Complete the following fields:</p> <p>Name - Type the parameter name.</p> <p>Value - Type a value for the parameter.</p> <p>Parameter Type - Select the parameter's type, which determines how the parameter should be used.</p> <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Note: If you select an AWS authentication type, then you must add a <i>mandatory request parameter</i> in all the Actions you create. The parameter name must be <i>aws.service</i> and the parameter type must be <i>CFG_PARAM</i>. Type the service name in the endpoint URL as the parameter value. For example, if the endpoint URL is <code>https://<instance>.s3.com/</code>, type the parameter value as <code>s3</code>.</p> </div> <p>REST services rely on HTTP methods (GET, POST, PUT, and DELETE) to make requests to a SaaS provider. Thus the parameters are closely tied to these HTTP methods, as they are sent as part of</p>

Field	Description
	<p>these HTTP method requests. The parameters are part of the HTTP URI.</p> <p><i>URI_CONTEXT</i> parameters are passed as the path component of a REST Resource URI, and the parameter names correspond to the URI path variable names specified in the {} annotation. For example, if the URI is <code>https://api.twitter.com/1.1/users/{id}</code>, the Resource path will be <code>/users/{id}</code>, the parameter type will be <code>uriContext</code>, the parameter name will be <code>id</code>, and the value could be the user id, for example, either 1, or 2, or 3.</p> <p><i>QUERYSTRING_PARAM</i> parameters are passed as the query component of a REST resource invocation request. For example, if the URI is <code>https://api.twitter.com/1.1/users/show.json?screen_name=twitterdev</code>, the resource path will be <code>/users/show.json</code>, <code>screen_name</code> is the name of the parameter, <code>twitterdev</code> is the value of the parameter, and the parameter type is <code>query</code>.</p> <p><i>CFG_PARAM</i> is an internal configuration parameter.</p> <p>Required - Select this option if you want this parameter to be made mandatory while creating an Integration.</p>
Request Header	<p>HTTP headers allow the client and the server to pass additional information with the request or the response.</p> <div data-bbox="618 1367 1300 1465" style="background-color: #f0f0f0; padding: 5px;"> <p>Note: Do not add an authorization header if you use credentials as the mode of authentication.</p> </div> <p>Click Add Header to add a request HTTP header. In the Add Header dialog box, complete the following fields:</p> <p>Name - Type the Header name.</p> <p>Value - Type a value for the Header.</p> <p>Required - Select this option if you want this Header to be made mandatory while creating an Integration.</p>

Field	Description
Request Body	<p>In the Request Body pane, complete the following fields:</p> <p>Content Type If the documentation of the SaaS provider specifies that the content type of the request body is JSON, select application/json as the content type. If the documentation of the SaaS provider specifies that the content type of the request body is XML, select application/xml as the content type. If the documentation of the SaaS provider specifies that the content type of the request body is binary, select Binary as the content type. These options allow you to control the content in an HTTP request body.</p> <p>Document Type - Select a Document Type for the request body or click Create Document Type to create a new Document Type. See “Document Types” on page 420.</p> <p>Note: Document Types created for a REST Application do not appear in the Develop > Document Types screen but appears only in the Document Types panel for the selected REST Application.</p>
Response Header	<p>In the Response Header pane, click Add Header to add a Response HTTP header.</p> <p>Note: Do not add an authorization header if you use credentials as the mode of authentication.</p> <p>Complete the following fields:</p> <p>Name - Type the Header name.</p> <p>Value - Type a value for the Header.</p> <p>Required - Select this option if you want this Header to be made mandatory while creating an Integration.</p>
Response Body	<p>In the Response Body pane, complete the following fields:</p> <p>HTTP Code - Type a single HTTP status code or a code range to indicate the status of the response.</p>

Field	Description
	<p>Valid values are 100, 101, 102...599 or a range from 100-599.</p> <p>Content Type If the documentation of the SaaS provider specifies that the content type of the response body is JSON, select application/json as the content type. If the documentation of the SaaS provider specifies that the content type of the response body is XML, select application/xml as the content type. If the documentation of the SaaS provider specifies that the content type of the response body is binary, select binary as the content type. These options allow you to control the content in an HTTP response body.</p> <p>Document Type - Select a Document Type for the Response Body or click Create Document Type to create a new Document Type. See “Document Types” on page 420.</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>Note: Document Types created for a REST Application do not appear in the Develop > Document Types screen but appears only in the Document Types panel for the selected REST Application.</p> </div>

7. Click **Save**.

The Action appears in the **Define Resources and Actions** page. You can **Edit** or **Delete** the Action from the **Define Resources and Actions** page.

Note: Do not edit or delete an Action if it is already used in an Operation. If the Action is edited or deleted, the Operations that are dependent on the Action including the Integrations that are dependent on the affected Operations, will not function properly.

8. Click **Next**.

The **Confirm REST Application** page appears.

9. Click **Finish** to create the REST Application.

The new REST Application appears in the **REST Applications** page.

Note: To edit the REST Application, click the REST Application link and then click **Edit Application**. You can change the **Description** and **Application Icon**. After you click **Finish**, the **Update REST Application** window appears, which provides a summary of the impacted Accounts, Operations, and Integrations. Click **Update** to update the REST Application.

To delete the REST Application, click **Delete Application**

Keys and Certificates

Keystores and truststores are files that function as repositories for storage of keys and certificates necessary for SSL authentication, encryption/decryption, and digital signing/verification services. Keystores and truststores provide added layers of security and ease of administration, compared to maintaining the keys and certificates in separate files.

Integration Cloud stores its private keys and SSL certificates in keystore files and the trusted roots for the certificates in truststore files. Keystores and truststores are secure files with industry-standard file formats.

If you want to run services that submit HTTPS requests to other resources on the Internet, your server will be acting as a client and will receive certificates from these resources. In order for these transactions to work, your server must have copies of their public keys and signing CA certificates.

To identify a particular keystore or truststore file, or private key within a keystore, aliases are used. The use of aliases simplifies keystore and truststore management, because you do not need to enter path information when specifying a keystore, truststore, or the private key.

Note: You can add, edit, or view keystore and truststore aliases and partner's self-signed certificates from **Connect > Keys & Certificates** and can use them to secure your Application Accounts. Some Applications, including custom REST Applications allow two-way SSL authentication by providing keystore and truststore aliases in the Account Configuration section. Users who have the **Administer** permission under **Settings  > Access Profiles > Administrative Permissions > Functional Controls > Advanced Security** can add, edit, and delete Keystores, Truststores, and Partner Certificates.

To add a Keystore, from the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Keystores > Add Keystore**.

To add a Truststore, from the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Truststores > Add Truststore**.

To add a Partner Certificate, from the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Partner Certificates > Add Certificate**.

Add Keystore

Integration Cloud allows you to upload a Keystore file to store SSL certificates and keys. A Keystore file contains one or more pairs of a private key and signed certificate for its corresponding public key. From this screen, you can create aliases for the Keystore, so that they can be referenced while creating an Account for an Application.

To add a Keystore

1. From the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Keystores > Add Keystore**.
2. Provide a name and description for the **Keystore file**.
3. In the **Type** field, select the Keystore file format. The default file format is **JKS**. You can also use **PKCS12**, a commonly used, standardized, certificate file format that provides a high degree of portability.
4. In the **Provider** field, select the provider from the list of available providers. The corresponding provider will be available in the provider list for a selected Keystore type.
5. Click **Browse** to select the Keystore file.
6. In the **Passphrase** field, enter the passphrase for the Keystore file. The passphrase must have been defined at the time the Keystore was created.
7. Click **Next** to protect the Key Aliases with passphrases. A key alias is a label for specific key within a Keystore. Enter a passphrase for each Key Alias found in the Keystore file, and then click "Finish" to upload the Keystore file.

The uploaded Keystore file can be used while creating an Account for an Application.

Add Truststore

Integration Cloud allows you to upload a Truststore file, which contains the trusted root of the certificate or signing authority (CA). From this screen, you can create aliases for the Truststore, so that they can be referenced while creating an Account for an Application.

To add a Truststore

1. From the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Truststores > Add Truststore**.
2. Provide a name and description for the Truststore file.
3. In the **Type** field, select the Truststore file format. The default file format is **JKS**. You can also use **PKCS12**, a commonly used, standardized, certificate file format that provides a high degree of portability.
4. In the **Provider** field, select the provider from the list of available providers. The corresponding provider will be available in the provider list for a selected Truststore type.
5. Click **Browse** to select the Truststore file.
6. In the **Passphrase** field, enter the passphrase for the Truststore file. The passphrase must have been defined at the time the Truststore was created and is used to protect the contents of the Truststore.

7. Click **Save** to upload the Truststore file.

The uploaded Truststore file can be used while creating an Account for an Application.

Add Partner Certificate

Integration Cloud allows you to upload the Partner's certificate which contains its public key. The Partner's certificate with the public key is required to encrypt outbound request messages and verify the signature of inbound messages.

From this screen, you can create aliases for Partner Certificates, so that they can be referenced while creating an Account for an Application.

To add a Partner Certificate

1. From the Integration Cloud navigation bar, click **Connect > Keys & Certificates > Partner Certificates > Add Certificate**.
2. Provide a name and description for the Partner Certificate file.
3. Click **Browse** to select the Partner Certificate file.
4. Click **Save** to upload the Partner Certificate file.

The uploaded Partner Certificate can be used while creating an Account for an Application.

4 Develop

■ Integrations	232
■ REST APIs	412
■ Document Types	420
■ Reference Data	422
■ Recipes	426

Integrations

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-premises applications.

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

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

Note: If assets used by an Integration are deleted, you will not be able to deploy the Integration into subsequent stages or export the Integration. See ["Deploy" on page 440](#) for information on how to deploy assets.

To edit an Integration, select the Integration, and then click **Edit**. The Integration opens up for editing in the **Design** panel. To delete an Integration, select the Integration, and then click **Delete**. The Integration will be permanently deleted and cannot be recovered. To copy an Integration, select the Integration, and then click **Copy** to save the Integration with a different valid name. This way you can have different names for the same Integration for different stages.

To export an Integration, select the Integration, and then click **Export**.

Note: Users who have the required permissions under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Assets** can export assets.

To import Integrations, select the Integration, and then click **Import Integrations**. See "Importing Integrations" and "Exporting Integrations" for more information. To create a new Integration, click **Add New Integration**, and then select **Synchronize two applications** to create a Point to Point Integration. To create an Orchestrated Integration, select **Orchestrate two or more applications**.

Point-to-Point Integrations

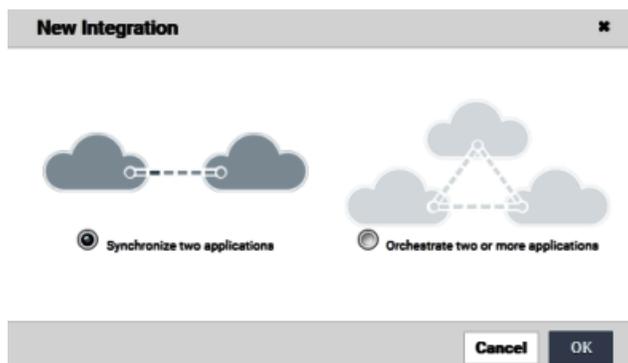
Integration Cloud enables you to integrate your cloud-based Software as a Service (SaaS) applications with other cloud-based SaaS applications. It also integrates your SaaS applications with on-premises applications.

Integration between two cloud providers includes the following steps:

- Invoking a source Operation on an application, which fetches data from it
- Invoking a target Operation on an application, which uploads data into it
- Filtering the data fetched from an application, before it is passed on to the target application
- Mapping the data fetched from an application, to the structure needed by the target application to which you want to upload the data.

To add or edit an existing Point-to-Point Integration

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



Note: See [“Orchestrated Integrations” on page 235](#) for information on how to create an orchestrated Integration.

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

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

5. Drag and drop your applications to the **Source** and **Target** sections. You can also double-click an Application to move it to the required section.
6. Select an Account, and then select a custom or a predefined Operation in both the Source and Target sections. Only active or enabled Accounts are listed in the drop down list.

Note: If you had already done the mapping for a source and target Operation, and you want to change any of the source and target Operations, all the mappings you had performed before will be removed.

7. Click **Next** to filter the data fetched by the application selected in the source section, before it is passed on to the application selected in the target section. Click **Load Data** to preview the data as well as view the data filters. The source Operation fetches the data and displays a sample of the data in the preview pane. Out of all the records fetched, you may want to upload only selected records. To do this, you can have a selection or a filter criteria so that you can view only a few records. A **sample preview** of only a few records can be viewed to analyze the kind of data that exists in the system. After you analyze the records, you can set filters, to upload, for example only Accounts that are based out of California to the target application. After you set the filters, whenever you run the Integration, all records will be fetched from the source application, but only the filtered records will be moved to the target application after mapping and transformation.
8. Click **Next** to map the data fetched by the application selected in the source section, to the structure needed by the application selected in the target section.

Select a field from the source section and drag and drop it on to a relevant field in the target section. Select a mapping and then click the **Unmap** icon to unmap only the selected mapping. Click the **Clear All** icon to unmap all the mapped elements, values set for fields, and transformers. To view only the mapped fields, select the **Show Only Mapped Fields** option.

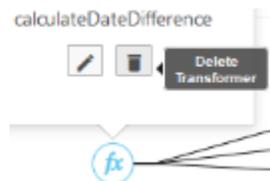
You can select a field in the target Operation table, and then choose to set a new value of the selected field in the target Operation. You can assign a value to a field when the field is not linked or when the field is only implicitly linked to another value in the pipeline. You cannot assign values to fields that are explicitly linked to another value in the pipeline or fields that have been dropped from the pipeline.

You can copy a field from the fields panel by clicking the  icon. Depending on the context, you can either paste the field or the field path. For example, if you copy a field and paste the field in the **Set Value** window in an Integration (double-click a field to set a value), the field path will be pasted. If you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index [10]. But if it is pasted in the document tree, it will appear as an array, like A[]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B.

Note: The paste option is not applicable for Point-to-Point Integrations.

Click the **Add Transformer** icon to add a transformer in the **Transform Data** page. This page allows you to transform the source Operation data, for example, concatenate two strings and map it to a single field. Several *built-in services* specifically designed to translate values between formats are provided. You can transform time and date information from one format to another, perform simple arithmetic calculations (add, subtract, multiply, and divide) on integers and decimals contained in String fields, or transform String values in various ways. Reference Data is also available while transforming the data.

The **Transform Data** screen also allows you to look up and use data from another source Operation to transform the data. After adding a transformer, you can click the  icon and select **Edit Transformer** or **Delete Transformer** to either modify the transformer or delete it.



9. Click **Next** to review your Integration.
10. Click **Save** and then click **Finish** to create your Integration.

The new Integration appears in the **Integrations** page.

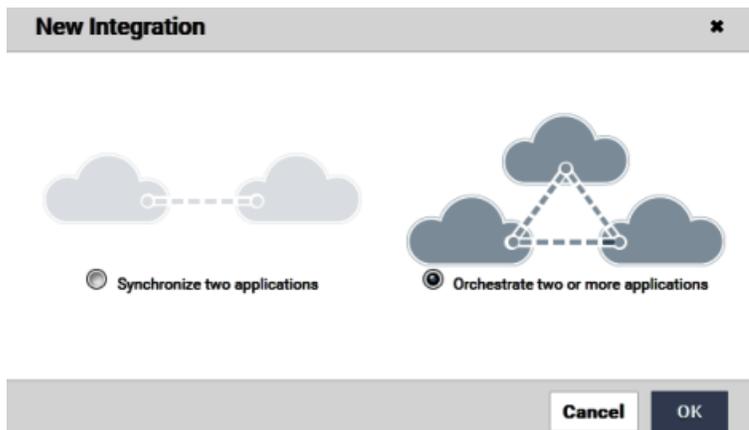
Orchestrated Integrations

Orchestrated Integration is the process of integrating two or more applications together, to automate a process, or synchronize data in real-time. Orchestrated Integration enables you to integrate applications and provides a way to manage and monitor your integrations.

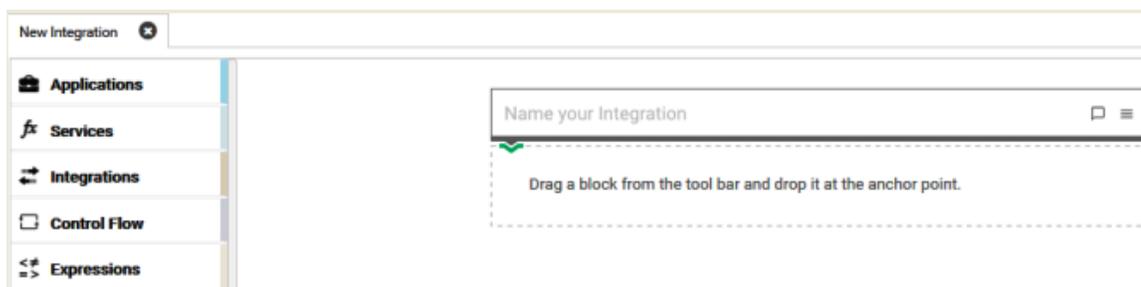
Integration Cloud supports advanced integration scenarios involving multiple application endpoints, complex routing, and Integrations involving multiple steps. Using a graphical drag and drop tool, you can create complex, orchestrated integrations and run them in the Integration Cloud environment.

To create an orchestrated integration

1. From the Integration Cloud navigation bar, click **Develop**. The **Integrations** screen appears.
2. To create a new Integration, from the **Integrations** screen, click **Add New Integration**.
3. Select **Orchestrate two or more applications**, and then click **OK**.



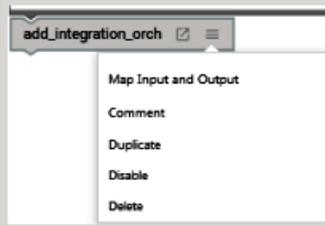
The user interface consists of a *tool bar* and a *workspace*. The tool bar holds all the available categories with blocks. You can browse through the menu of blocks and can set up your own Integration by plugging blocks together in the workspace. The menu of blocks comes with a large number of predefined blocks from Applications, Services, Integrations, conditions to looping structures. You can drag relevant blocks from the tool bar and drop them at the anchor point.



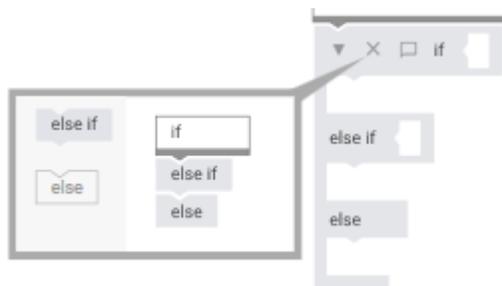
The tool bar has a large number of blocks for common instructions and the blocks are divided into the following categories:

- Applications
- Services
- Integrations
- Control Flow
- Expressions

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

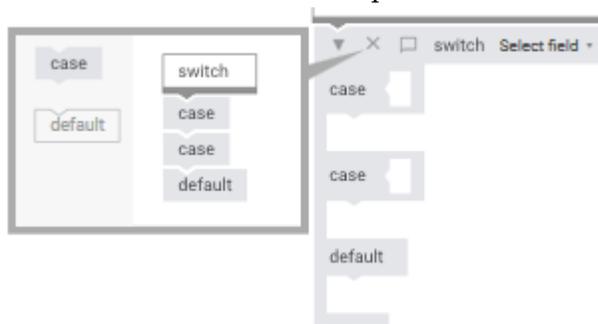
Block category	Icons	Description
Integrations		<p>can sequence services and manage the flow of data among them.</p> <p>Note: For information on the different services, see Built-In Services.</p> <p>The Reference Data block appears only if a Reference Data service is available at Develop > Reference Data > Reference Data page. See Reference Data for more information.</p> <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 dropped at the anchor point. The Integration will open up for editing in a new tab.</p> <p>Click the  icon and select Map Input and Output if you want to map the input of the operation from the Pipeline and also map the output of the operation into the pipeline.</p>  <p>Click Duplicate to repeat a block, click Collapse to flatten a block, click Delete to remove a block from the workspace, or click Disable to disable a block and all blocks within that block. If you disable blocks, those blocks will not be considered for execution, test, or debug operations.</p>
Control Flow		<p>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</p>

Block category	Icons	Description
		<p>or false. The if block is used to evaluate a boolean condition and if the condition is true, statements inside the <i>if</i> block are executed. The <i>if</i> statement can be followed by an optional <i>else</i> statement, which executes when the boolean expression is false.</p>



The *if* statements are executed from the top towards the bottom. You can use one *if* or **else if** statement inside another *if* or *else if* statement(s). You cannot have multiple else statements.

Switch allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case, that is, Switch evaluates a variable and skips to the value that matches the case. For example, if the Switch variable evaluates as "A", then case "A" is executed. A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases are true. You cannot insert multiple default statements.



Note: You can include case steps that match null or empty switch values. A switch value is considered to be null if the variable does not exist in the pipeline or is explicitly set to null. A switch value is considered to be an empty string

Block category	Icons	Description
		<p>if the variable exists in the pipeline but its value is a zero length string.</p> <p>Note: Switch executes the first case that matches the value, and <i>exits the block</i>.</p>
		<p>The try catch block is used to handle errors and exceptions. If you have a statement in the try block that has thrown an error, the error will be caught in the catch statement.</p> <p>Note: If an error is thrown inside the catch section of the try catch block, the error will be ignored and the next statements in the Integration will be executed.</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 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>The while loop is used to iterate a part of the program several times. If the number of iterations are not fixed, it is recommended to use the while loop.</p>
		<p>The do-until loops are similar except that they repeat their bodies until some condition is true.</p>
		<p>The for-each block traverses items in a collection. Unlike other for loop constructs, for-each loops usually maintain no explicit counter: they essentially say "do this to everything in this set", rather than "do this x times".</p>
		<p>The Exit Integration signaling success block allows you to successfully terminate and exit from the currently</p>

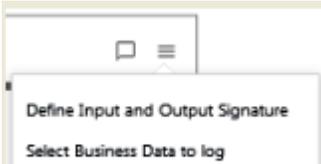
Block category	Icons	Description
		<p>running Integration. You cannot attach child blocks to the Exit Integration signaling success block.</p> <p>The Exit Integration signaling failure "..." block abnormally terminates the currently running integration with an error message. You can specify the text of the error message that is to be displayed. If you want to use the value of a pipeline variable for this error message, type the variable name between % symbols, for example, <code>%mymessage%</code>. The variable you specify must be a String. You cannot attach child blocks to the Exit Integration signaling failure "..." block.</p> <p>The Throw error "..." block can be attached inside any block <i>except the catch section of the try catch block</i>, and allows you to <i>explicitly</i> throw an exception with a custom error message. If it is used inside the try section of the try catch block, the error will be caught in the catch section. If you want to use the value of a pipeline variable for this custom error message, type the variable name between % symbols, for example, <code>%mymessage%</code>. The variable you specify must be a String. You cannot attach child blocks to the Throw error "..." block.</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p>Note: If you add a Throw error "..." block inside a try catch block, any changes done to the pipeline variables inside the try block will be reset to the previous values existing in the pipeline.</p> </div> <p>The Break out of loop block should be used only within a loop and allows you to break out of the containing loop, that is, it allows you to break the program execution out of the loop it is placed in. You cannot attach child blocks to the Break out of loop block.</p> <p>A Loop takes as input an array field that is in the pipeline. It loops over the members of an input array, executing its child steps each time through the loop. For example, if you have a Integration that takes a string as input and a string list in the pipeline, use Loops to invoke the Integration one time for each string in the string list. You identify a single array field to use as input when you set the properties for the Loop. You can also designate a</p>

Block category	Icons	Description
Expressions		<p>single field for the output. Loop collects an output value each time it runs through the loop and creates an output array that contains the collected output values.</p> <p>Use the Transform Pipeline block to make pipeline modifications. See Pipeline and Signatures for more information.</p> <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 and block will return true only if both of its two inputs are also true. The or block will return true if either of its two inputs are true. The not block converts its Boolean input into its opposite.</p> <p>You can also type a text value, select a field on which to build an expression (Select field), or select a block with no inputs.</p> <p>The Field exists block allows you to check if a variable exists or not and can be used with other Control Flow blocks, for example, the <i>if</i> block. The <i>Field exists</i> block validates the existence of a particular field in the pipeline.</p> <p>Note: It is recommended not to leave an input empty.</p>

4. Provide a valid name and description for the Integration.
5. Click **Applications**. The list of supported Applications appears.
6. Drag and drop an Application to the root block.
7. To select the Account and Operation for the Application, click 

The following table depicts the block interactions:

Icons	Applicable for...	Action/Description
	Only for the Control Flow block and the Root block.	Comments for the Control Flow block and the Root block.

Icons	Applicable for...	Action/Description
☰	Applications, Services, Integrations, and the Root block	Root Block > Define Input and Output Signature
		
		<p>Click to define the input and output signature of an Integration. You can declare the input and output parameters for an Integration using the Input and Output tabs. Input and output parameters are the names and types of fields that the Integration requires as input and generates as output. These parameters are also collectively referred to as a signature. For example, an Integration can take two string values, an account number (AcctNum) and a dollar amount (OrderTotal) as inputs and produces an authorization code (AuthCode) as the output. On the Output tab, specify the fields that you want the Integration to return.</p>
		<p>You can use a Document Reference to define the input or output parameters for an Integration. If you have multiple Integrations with identical input parameters but different output parameters, you can use a Document Type to define the input parameters rather than manually specifying individual input fields for each Integration. When you assign a Document Type to the Input or Output side, you cannot add, modify, or delete the fields on that part of the tab.</p>
		<p>You can select a Document Type from the Document Reference drop-down list. To create a Document</p>

Icons	Applicable for...	Action/Description
		<p>Type, from the Integration Cloud navigation bar, select Develop > Document Types > Add New Document Type.</p> <p>You can click Load XML and then paste the XML content to generate a Document Type from the XML structure or click Load JSON and then paste the JSON content to generate a Document Type from the JSON structure.</p> <p>You can create pipeline variables as document references, create document types comprising of document references, and also define the signature of Integrations comprising of document references.</p> <p>You can also copy a field from the fields panel by clicking the  icon. Depending on the context, you can either paste the field or the field path by clicking the  icon. For example, if you copy a field and paste the field in the Set Value window in an Integration, (double-click a field to set a value), the field path will be pasted.</p> <p>See Document Types for more information.</p> <div data-bbox="865 1423 1365 1556" style="background-color: #f0f0f0; padding: 5px;"> <p>Note You cannot modify or paste the child fields of a Document Reference.</p> </div> <p>Select Business Data to Log</p> <p>Integration Cloud allows you to log select business data from the Operation and Integration signatures either always, or only when errors occur. Values of logged fields can be viewed in the Only Business Data section in the Execution Results screen. You can</p>

Icons	Applicable for...	Action/Description
		<p>also create aliases for the logged fields.</p> <div data-bbox="867 407 1367 592" style="background-color: #f0f0f0; padding: 10px;"> <p>Note User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.</p> </div> <p>To select input or output fields for logging, click Select Business Data to Log, and in the Select Business Data to Log dialog box, choose whether you want to log business data only when errors occur (On Failure) or choose (Always) to always log business data. The default setting is On Failure. Then expand the Input Fields and Output Fields trees to display the fields available in the signature, and select the check boxes next to the fields you want to log. If you want to define an alias for a field, type an alias name beside the field. The alias defaults to the name of the selected field, but it can be modified.</p> <p>When selecting fields for logging, you can create the same alias for more than one field, but this is not recommended. Having the same alias might make monitoring the fields at run time difficult.</p> <p>Map Input and Output</p> <p>Map the input of the operation from the Pipeline and also map the output of the operation into the pipeline.</p> <p>You can copy a field from the fields panel by clicking the  icon. Depending on the context, you can either paste the field or the field path by clicking the  icon. If</p>

Icons	Applicable for...	Action/Description
		<p>you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index [10]. But if it is pasted in the document tree, it will appear as an array, like A[]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B.</p> <p>You can select a block, other than the root block, and click Duplicate to repeat a block, click Collapse to flatten a block, click Delete to remove a block from the workspace, or click Disable to disable a block and all blocks within that block. If you disable blocks, those blocks will not be considered for execution, test, or debug operations.</p>
	Control Flow > Transform Pipeline	Make pipeline modifications. Edit data mapping, add Transformer, clear all mappings, add, delete, edit, or discard a field, set a value for a field and perform pipeline variable substitutions.
	Applications	Select an Account and an Operation for the Application.
	Applications	The block is not configured. Select an Account and an Operation for the Application.
	Services	The block is not configured. Select a service.

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.
	Orchestrated Integrations	An Orchestrated Integration has been modified or newly created but not saved.

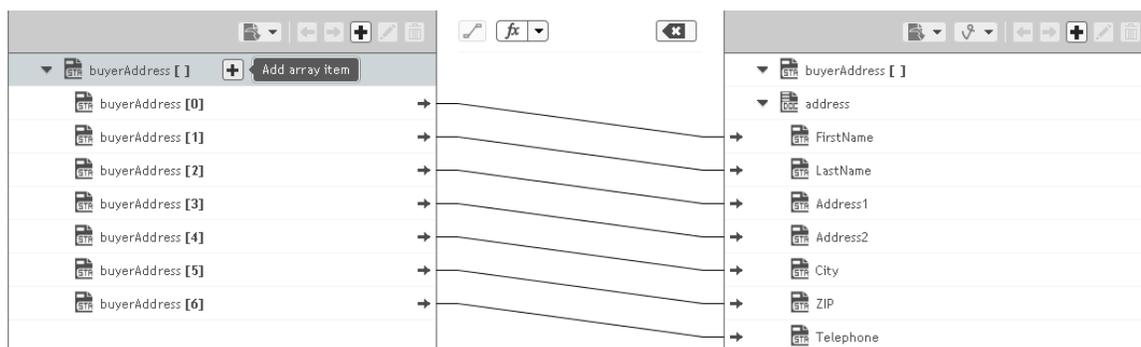
8. Create the Integration using the available constructs by inserting the blocks, setting properties, declaring the input and output parameters, setting values, performing pipeline variable substitutions (if you want to replace the value of a pipeline field at run time), and mapping the pipeline data.
9. Click the  icon and then select **Map Input and Output** to map the Pipeline Input to the Input Signature. To view only the mapped fields, select the **Show Only Mapped Fields** option.
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.



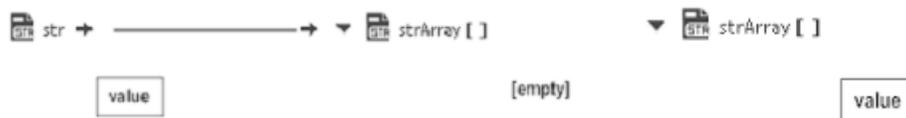
Suppose a String List has length 3 and if you link index 4 of the String List, at run time, the String List length is increased from 3 to 5.

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.

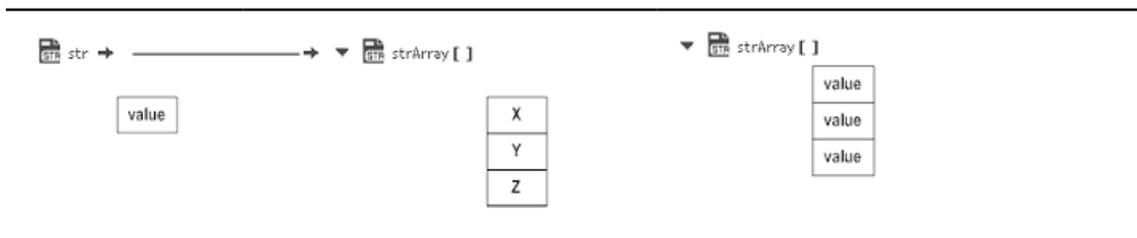
If you link...	To...	Then...
A scalar variable	An array variable that is empty (the variable does not have a defined length)	The link defines the length of the array variable; that is, it contains one element and has length of one. The first (and only) element in the array is assigned the value of the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

If you link...	To...	Then...
----------------	-------	---------

the array is assigned the value of the scalar variable.

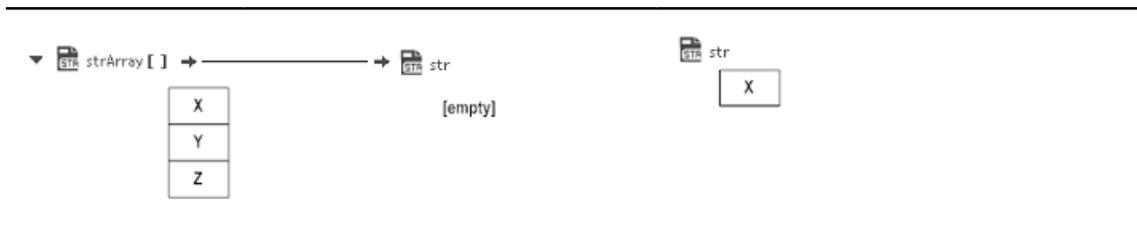


If you link...	To...	Then...
----------------	-------	---------

An array variable

A scalar variable

The scalar variable is assigned the first element in the array.

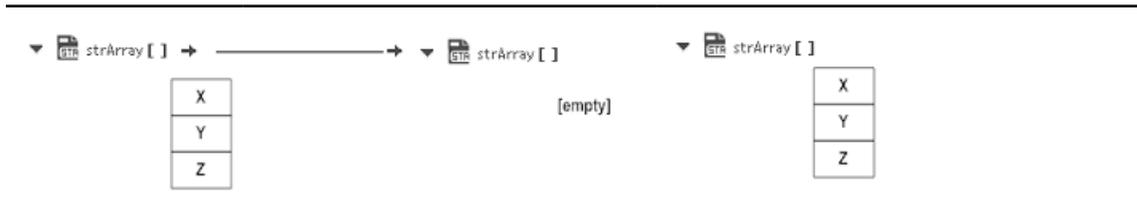


If you link...	To...	Then...
----------------	-------	---------

An array variable

An array variable that does not have a defined length

The link defines the length of the target array variable; that is, it will be the same length as the source array variable. The elements in the target array variable are assigned the values of the corresponding elements in the source array variable.



If you link...	To...	Then...
----------------	-------	---------

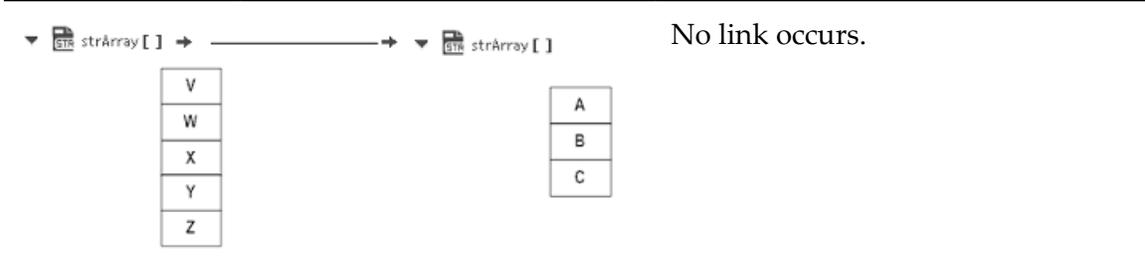
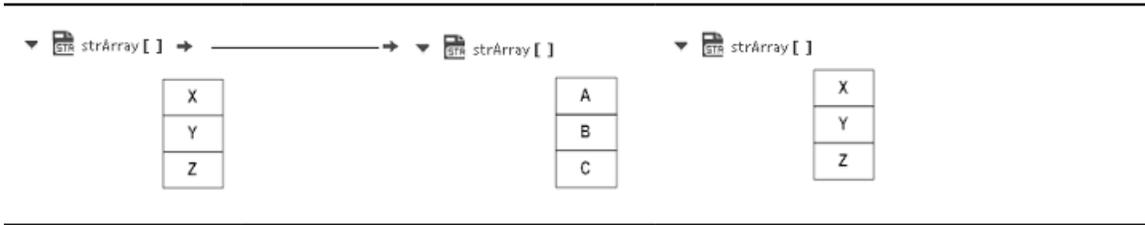
An array variable

An array variable that has a defined length

The length of the source array variable *must* equal the length

If you link...	To...	Then...
----------------	-------	---------

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.



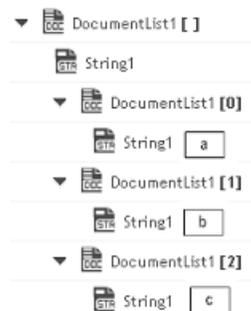
A source variable that is the child of a Document List is treated like an array because there is one value of the source variable for each Document in the Document List. For example:

If you link...	To...
----------------	-------



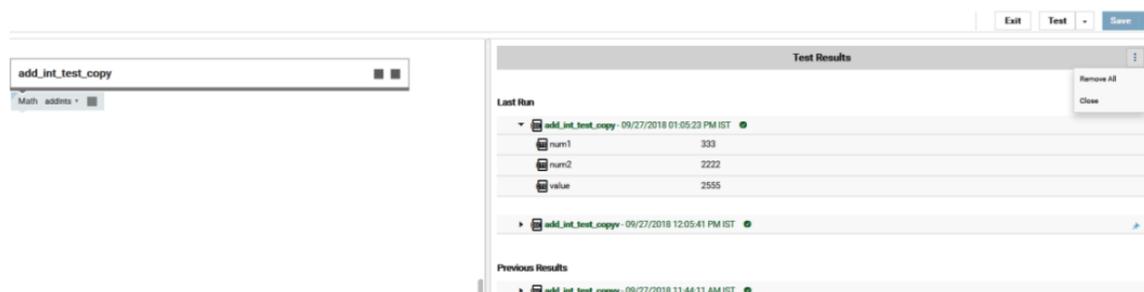
Where the value of DocumentList1 is...

Then the value of StringList1 is...

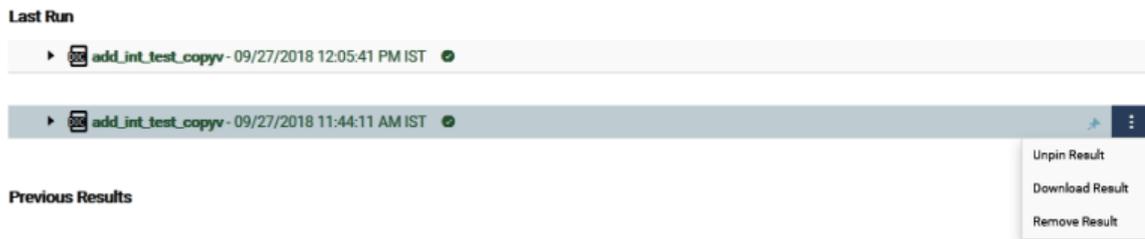


- Use the **Transform Pipeline** block under the **Control Flow** category to adjust the pipeline at any point in the Integration and make pipeline modifications. Within this step, you can discard or remove an existing pipeline input field, (once you discard a field from the pipeline, it is no longer 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. Click **Exit** to cancel your changes. The new Integration appears in the **Integrations** page. You can click on the Integration link in the **Integrations** page to view the Integration details.
- Integration Cloud allows you to test or debug an Integration after you have created it.

After saving an Integration, in the edit Integration page, click **Test** to test the Integration execution in real time and view the execution results on the **Test Results** panel.



The **Test Results** panel displays up to 25 test entries and the most recent test entry is located at the top of the panel. Click the  icon on the **Test Results** panel header and click **Remove All** to delete the test results permanently or click **Close** to close the test results panel.



Point to a test result entry, click the  icon, and click **Download Result** to save the entry locally in JSON format. Click **Remove Result** to remove the selected entry. Click **Pin Result** if you want to prevent a previous result from getting deleted as more results fill the test results panel. Click **Unpin Result** to move the result to the **Previous Results** panel.

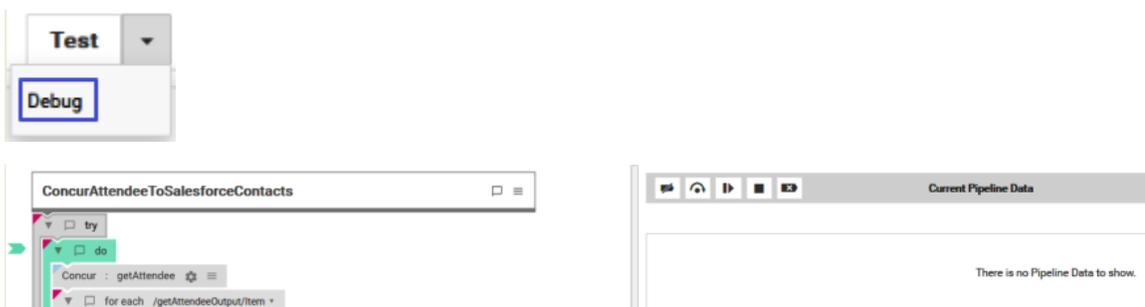
Debugging Orchestrated Integrations

You can debug an orchestrated Integration and can inspect the data flow during the debugging session. You can debug an orchestrated Integration only in the Development stage and after the Integration has been saved.

You can do the following in debug mode:

- Start an Integration in debug mode, specify the input values, and inspect the results.
- Examine and edit the pipeline data before and after executing the individual blocks.
- Monitor the execution path, execute the blocks one at a time, or specify breakpoints where you want to halt the execution.

To start the **Debug** mode, click the drop-down arrow beside **Test** and then click **Debug**.



The following table describes the options available in the **Debug** panel:

Icons	Applicable for...	Action/Description
	Inserting Breakpoints	A breakpoint is a point in an Integration where you want processing to halt when you debug that

Icons	Applicable for...	Action/Description
		<p>Integration. Breakpoints can help you isolate a section of code or examine data values at a particular point in the execution path. For example, you might want to set a pair of breakpoints before and after a particular block so that you can examine the pipeline before and after that block executes.</p> <p>Breakpoints are recognized only when you execute an Integration in debug session. To insert a breakpoint, in debug mode, click the top-left corner of the block. To remove a breakpoint, click on the inserted breakpoint.</p> <p>When you execute an Integration that contains a breakpoint, the Integration is executed up to, but not including the designated breakpoint. At this point, processing stops and the debug session suspends. To resume processing, select Resume. After you resume the debug session, the Integration flow stops at the next breakpoint.</p>
	Ignore All Breakpoints	Ignores all breakpoints inserted in the Integration blocks. You cannot insert breakpoints for variables in the Expressions category.
	Stepover	Executes the current block. Integration Cloud suspends the debug session immediately before

Icons	Applicable for...	Action/Description
	Resume	<p>executing the next block in the Integration.</p> <p>The debug session resumes but suspends at the next breakpoint.</p>
	Stop	<p>Terminates the debug session.</p> <p>A debug session may also stop by itself for the following reasons:</p> <ul style="list-style-type: none"> ■ The Integration that you are debugging executes to completion (error or success). ■ You select Stepover for the last step in the Integration. ■ You Exit the Integration.
	Clear All Breakpoints	Removes all breakpoints inserted in the Integration.

Modifying the current pipeline data while Debugging

During debugging, you can modify the contents of the pipeline. The changed values will be applied when you perform a **Stepover** or **Resume**.

While modifying the pipeline, keep the following points in mind:

- You can modify the pipeline data only during an active debug session.
- When you modify values in the pipeline, the changes apply only to the current debugging session. The Integration is not permanently changed.
- You can only modify existing variables. You cannot add new variables to the pipeline.

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

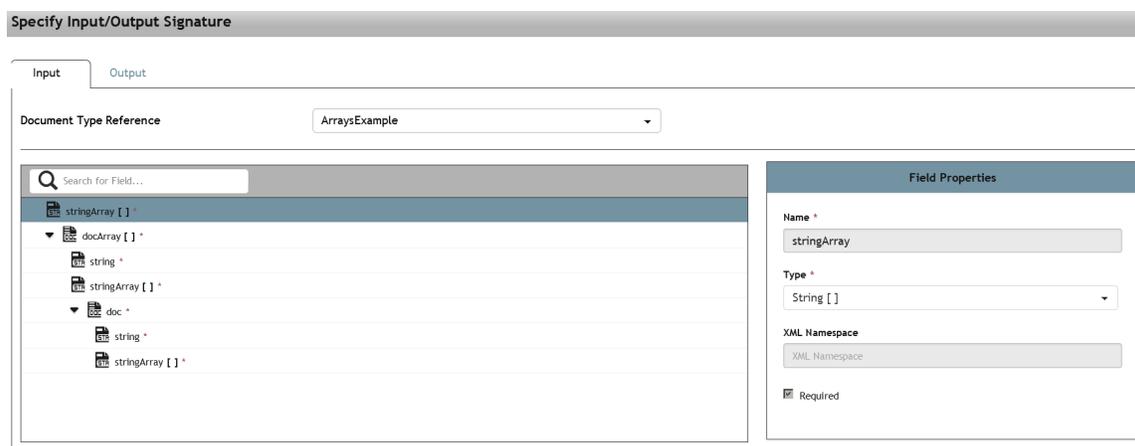
On the output side of the Input/Output tab, you specify the variables that you want the Integration to return to the calling program or client. The guidelines for defining the output parameters are similar to those for defining input parameters:

- **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 root block of an Orchestrated Integration, click the  icon > **Define Input and Output Signature** to define the input and output parameters. On the Input tab, you define the variables that the Integration requires as input. On the Output tab, you define the variables the Integration returns to the client or calling program.



For an Integration, the input side describes the initial contents of the pipeline. In other words, it specifies the variables that this Integration expects to find in the pipeline at run time. The output side identifies the variables produced by the Integration and returned to the pipeline.

Note: You can create pipeline variables as document references, create document types comprising of document references, and also define the signature of Integrations comprising of document references.

You can declare a signature in one of the following ways:

- **Reference a document type.** You can use a document type to define the input or output parameters for an Integration. When you assign a document type to the Input or Output side, you cannot add, modify, or delete the variables on that half of the tab.
- **Manually insert input and output variables.** Click the **+** icon to manually insert variables to the Input or Output sides.

Using a Document Type to Specify Integration Input or Output Parameters

You can use a document type as the set of input or output parameters for an Integration. If you have multiple Integrations with identical input parameters but different output parameters, you can use a document type to define the input parameters rather than

manually specifying individual input fields for each Integration. When a document type is assigned to the input or output of an Integration, you cannot add, delete, or modify the fields on that tab.

Built-In Services

This section describes the services provided with Integration Cloud. Services are method-like units of logic that clients can invoke.

Integration Cloud has an extensive library of built-in services for performing common integration tasks such as transforming data values, performing simple mathematical operations, and so on. Related services are grouped in blocks. Input and output parameters are the names and types of fields that the service requires as input and generates as output. These parameters are also collectively referred to as a signature.

Date

Use **Date** services to generate and format date values.

Pattern String Symbols - Many of the Date services require you to specify pattern strings describing the data's current format and/or the format to which you want it converted. For services that require a pattern string, use the symbols in the following table to describe the format of your data. For example, to describe a date in the January 15, 1999 format, you would use the pattern string `MMMM dd, yyyy`. To describe the format `01/15/99`, you would use the pattern string `MM/dd/yy`.

Symbol	Meaning	Presentation	Example
G	era designator	Text	AD
y	year	Number	1996 or 96
M	month in year	Text or Number	July or Jul or 07
d	day in month	Number	10
h	hour in am/pm (1-12)	Number	12
H	hour in day (0-23)	Number	0
m	minute in hour	Number	30
s	second in minute	Number	55
S	millisecond	Number	978

Symbol	Meaning	Presentation	Example
E	day in week	Text	Tuesday or Tue
D	day in year	Number	189
F	day of week in month	Number	2 (2nd Wed in July)
w	week in year	Number	27
W	week in month	Number	2
a	am/pm marker	Text	PM
k	hour in day (1-24)	Number	24
K	hour in am/pm (0-11)	Number	0
z	time zone	Text	Pacific Standard Time or PST or GMT-08:00
Z	RFC 822 time zone (JVM 1.4 or later)	Number	-0800 (offset from GMT/UT)
'	escape for text	Delimiter	
''	single quote	Literal	'

Time Zones - When working with date services, you can specify time zones. The Earth is divided into 24 standard time zones, one for every 15 degrees of longitude. Using the time zone including Greenwich, England (known as Greenwich Mean Time, or GMT) as the starting point, the time is increased by an hour for each time zone east of Greenwich and decreases by an hour for each time zone west of Greenwich. The time difference between a time zone and the time zone including Greenwich, England (GMT) is referred to as the *raw offset*.

The following table identifies the different time zones for the Earth and the raw offset for each zone from Greenwich, England. The effects of daylight savings time are ignored in this table.

Note: Greenwich Mean Time (GMT) is also known as Universal Time (UT).

ID	Raw Offset	Name
MIT	-11	Midway Islands Time
HST	-10	Hawaii Standard Time
AST	-9	Alaska Standard Time
PST	-8	Pacific Standard Time
PNT	-7	Phoenix Standard Time
MST	-7	Mountain Standard Time
CST	-6	Central Standard Time
EST	-5	Eastern Standard Time
IET	-5	Indiana Eastern Standard Time
PRT	-4	Puerto Rico and U.S. Virgin Islands Time
CNT	-3.5	Canada Newfoundland Time
AGT	-3	Argentina Standard Time
BET	-3	Brazil Eastern Time
GMT	0	Greenwich Mean Time
ECT	+1	European Central Time
CAT	+2	Central Africa Time
EET	+2	Eastern European Time
ART	+2	(Arabic) Egypt Standard Time
EAT	+3	Eastern African Time
MET	+3.5	Middle East Time

ID	Raw Offset	Name
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
<code>calculateDateDifference</code>	Calculates the difference between two dates and returns the result as seconds, minutes, hours, and days.
<code>compareDates</code>	Compares two dates and returns the result as integer.
<code>currentNanoTime</code>	Returns the current time returned by the most precise system timer, in nanoseconds.
<code>dateBuild</code>	Builds a date String using the specified pattern and the specified date services.

Service	Description
dateTimeBuild	Builds a date/time string using the specified pattern and the specified date services.
dateTimeFormat	Converts date/time (represented as a String) string from one format to another.
elapsedNanoTime	Calculates the time elapsed between the current time and the given time, in nanoseconds.
formatDate	Formats a Date object as a string.
getCurrentDate	Returns the current date as a Date object.
getCurrentDateString	Returns the current date as a String in a specified format.
incrementDate	Increments a date by a specified period.

calculateDateDifference

Calculates the difference between two dates and returns the result as seconds, minutes, hours, and days.

Input Parameters

<i>startDate</i>	String Starting date and time.
<i>endDate</i>	String Ending date and time.
<i>startDatePattern</i>	String 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>	String 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

dateDifferenceSeconds **String** The difference between the *startingDateTime* and *endingDateTime*, truncated to the nearest whole number of seconds.

dateDifferenceMinutes **String** The difference between the *startingDateTime* and *endingDateTime*, truncated to the nearest whole number of minutes.

dateDifferenceHours **String** The difference between the *startingDateTime* and *endingDateTime*, truncated to the nearest whole number of hours.

dateDifferenceDays **String** The difference between the *startingDateTime* and *endingDateTime*, truncated to the nearest whole number of days.

Usage Notes

Each output value represents the same date difference, but in a different scale. Do not add these values together. Make sure your subsequent Integration steps use the correct output, depending on the scale required.

compareDates

Compares two dates and returns the result as an integer.

Input Parameters

startDate **String** Starting date to compare against *endDate* .

endDate **String** Ending date to compare against *startDate* .

startDatePattern **String** Format in which the *startDate* parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.

endDatePattern **String** Format in which the *endDate* parameter is specified (for example, yyyyMMdd HH:mm:ss.SSS). For pattern-string notation, see the "Pattern String Symbols" section.

Output Parameters

result **String** Checks whether *startDate* is before, the same, or after the *endDate*.

<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.

currentNanoTime

Returns the current time returned by the most precise system timer, in nanoseconds.

Input Parameters

None.

Output Parameters

nanoTime **java.lang.Long** Current time returned by the most precise system timer, in nanoseconds.

dateBuild

Builds a date String using the specified pattern and the specified date services.

Input Parameters

<i>pattern</i>	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 <i>pattern</i> , <code>dateBuild</code> returns null. If <i>pattern</i> contains a time zone and <i>timezone</i> is not specified, the default time zone is used.
<i>year</i>	String Optional. The year expressed in <i>yyyy</i> or <i>yy</i> format (for example, <i>01</i> or <i>2001</i>). If you do not specify <i>year</i> or you specify an invalid value, <code>dateBuild</code> uses the current year.
<i>month</i>	String Optional. The month expressed as a number (for example, <i>1</i> for January, <i>2</i> for February). If you do not specify <i>month</i> or you specify an invalid value, <code>dateBuild</code> uses the current month.
<i>dayofmonth</i>	String Optional. The day of the month expressed as a number (for example, <i>1</i> for the first day of the month, <i>2</i> for the second day of the month). If you do not specify <i>dayofmonth</i> or you specify an invalid value, <code>dateBuild</code> uses the current day.
<i>timezone</i>	String Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the "Time Zones" section, for example, <i>EST</i> for Eastern Standard Time. If you do not specify <i>timezone</i> , the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.
<i>locale</i>	String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <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>value</i>	String The date specified by <i>year</i> , <i>month</i> , and <i>dayofmonth</i> , in the format of <i>pattern</i> .
--------------	--

dateTimeBuild

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

Input Parameters

<i>pattern</i>	String Pattern representing the format in which you want the time returned. For pattern-string notation, see the “Pattern String Symbols” section. If you do not specify <i>pattern</i> , <code>dateTimeBuild</code> returns null. If <i>pattern</i> contains a time zone and the <i>timezone</i> parameter is not set, the default time zone is used.
<i>year</i>	String Optional. The year expressed in <i>yyyy</i> or <i>yy</i> format (for example, <i>01</i> or <i>2001</i>). If you do not specify <i>year</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current year.
<i>month</i>	String Optional. The month expressed as a number (for example, <i>1</i> for January, <i>2</i> for February). If you do not specify <i>month</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current month.
<i>dayofmonth</i>	String Optional. The day of the month expressed as a number (for example, <i>1</i> for the first day of the month, <i>2</i> for the second day of the month). If you do not specify <i>dayofmonth</i> or you specify an invalid value, <code>dateTimeBuild</code> uses the current day.
<i>hour</i>	String Optional. The hour expressed as a number based on a 24-hour clock. For example, specify <i>0</i> for midnight, <i>2</i> for 2:00 A.M., and <i>14</i> for 2:00 P.M. If you do not specify <i>hour</i> or you specify an invalid value, <code>dateTimeBuild</code> uses <i>0</i> as the <i>hour</i> value.
<i>minute</i>	String Optional. Minutes expressed as a number. If you do not specify <i>minute</i> or you specify an invalid value, <code>dateTimeBuild</code> uses <i>0</i> as the <i>minute</i> value.
<i>second</i>	String Optional. Seconds expressed as a number. If you do not specify <i>second</i> or you specify an invalid value, <code>dateTimeBuild</code> uses <i>0</i> as the <i>second</i> value.
<i>millis</i>	String Optional. Milliseconds expressed as a number. If you do not specify <i>millis</i> or you specify an invalid value, <code>dateTimeBuild</code> uses <i>0</i> as the <i>millis</i> value.

<i>timezone</i>	<p>String Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the “Time Zones” section, for example, <code>EST</code> for Eastern Standard Time.</p> <p>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.</p>
<i>locale</i>	<p>String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM YYYY</code> will produce <code>Friday 23 August 2002</code>, and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code>.</p>

Output Parameters

<i>value</i>	String Date and time in format of <i>pattern</i> .
--------------	---

dateTimeFormat

Converts date/time (represented as a String) string from one format to another.

Input Parameters

<i>inString</i>	<p>String Date/time that you want to convert.</p> <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p>Important: If <i>inString</i> contains a character in the last position, that character is interpreted as 0. This can result in an inaccurate date. For information about invalid dates, see the “Notes on Invalid Dates” section.</p> </div>
<i>currentPattern</i>	String Pattern string that describes the format of <i>inString</i> . For pattern-string notation, see the “Pattern String Symbols” section.
<i>newPattern</i>	String Pattern string that describes the format in which you want <i>inString</i> returned. For pattern-string syntax, see the “Pattern String Symbols” section.
<i>locale</i>	<p>String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM YYYY</code> will produce <code>Friday 23 August 2002</code>, and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code>.</p>

<i>lenient</i>	<p>String Optional. A flag indicating whether an exception will appear if the <i>inString</i> value does not adhere to the format specified in <i>currentPattern</i> parameter. Set to:</p> <ul style="list-style-type: none">■ <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> <ul style="list-style-type: none">■ <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>
----------------	--

Output Parameters

<i>value</i>	<p>String The date/time given by <i>inString</i>, in the format of <i>newPattern</i>.</p>
--------------	--

Usage Notes

As described in the “Notes on Invalid Dates” section, if the pattern *yy* is used for the year, `dateTimeFormat` uses a 50-year moving window to interpret the value of the year.

If *currentPattern* does not contain a time zone, the value is assumed to be in the default time zone.

If *newPattern* contains a time zone, the default time zone is used.

elapsedNanoTime

Calculates the time elapsed between the current time and the given time, in nanoseconds.

Input Parameters

<i>nanoTime</i>	<p>java.lang.Long Time in nanoseconds. If <i>nanoTime</i> is less than zero, then the service treats it as zero.</p>
-----------------	---

Output Parameters

<i>elapsedNanoTime</i>	java.lang.Long The difference between the current time in nanoseconds and <i>nanoTime</i> . If <i>nanoTime</i> is greater than the current nano time, the service returns zero.
<i>elapsedNanoTimeStr</i>	<p>String The difference between the current time in nanoseconds and <i>nanoTime</i> . The difference is expressed as a String, in this format:</p> <p>[years] [days] [hours] [minutes] [seconds] [millisec] [microsec] <nanosec></p> <p>If <i>nanoTime</i> is greater than the current nano time, the service returns zero.</p>

formatDate

Formats a Date object as a string.

Input Parameters

<i>date</i>	java.util.Date Optional. Date/time that you want to convert.
<i>pattern</i>	String Pattern string that describes the format in which you want the date returned. For pattern-string notation, see the <i>Pattern String Symbols</i> section.
<i>timezone</i>	<p>String Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the <i>Time Zones</i> section, for example, <code>EST</code> for Eastern Standard Time.</p> <p>If you do not specify <i>timezone</i> , the user's time zone is used, else GMT is used.</p>
<i>locale</i>	String Optional. Locale in which the date is to be expressed. For example, if <i>locale</i> is <code>en</code> (for English), the pattern <code>EEE d MMM yyyy</code> will produce <code>Friday 23 August 2002</code> , and the <i>locale</i> of <code>fr</code> (for French) will produce <code>vendredi 23 août 2002</code> .

Output Parameters

<i>value</i>	String The date/time given by <i>date</i> in the format specified by <i>pattern</i> .
--------------	--

getCurrentDate

Returns the current date as a Date object.

Input Parameters

None.

Output Parameters

date **java.util.Date** Current date.

getCurrentDateString

Returns the current date as a String in a specified format.

Input Parameters

pattern **String** Pattern representing the format in which you want the date returned. For pattern-string notation, see the “Pattern String Symbols” section.

timezone **String** Optional. Time zone in which you want the output date and time expressed. Specify a time zone code as shown in the “Time Zones” section, for example, `EST` for Eastern Standard Time.

If you do not specify *timezone*, the value of the server's "user timezone" property is used. If this property has not been set, GMT is used.

locale **String** Optional. Locale in which the date is to be expressed. For example, if *locale* is `en` (for English), the pattern `EEE d MMM yyyy` will produce `Friday 23 August 2002`, and the *locale* of `fr` (for French) will produce `vendredi 23 août 2002`.

Output Parameters

value **String** Current date in the format specified by *pattern*.

incrementDate

Increments a date by a specified amount of time.

Input Parameters

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

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

Output Parameters

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

Usage Notes

The *addYears*, *addMonths*, *addDays*, *addHours*, *addMinutes*, *addSeconds*, and *addMilliseconds* input parameters can take positive or negative values. For example, If *startDate* is 10/10/2001, *startDatePattern* is MM/dd/yyyy, *addYears* is 1, and *addMonths* is -1, *endDate* will be 09/10/2002.

If you specify only the *startDate*, *startDatePattern*, and *endDatePattern* input parameters and do not specify any of the optional input parameters to increment the period,

the `incrementDate` service just converts the format of `startDate` from `startDatePattern` to `endDatePattern` and returns it as `endDate`.

Note: The format of the date specified in the `startDate` parameter must match the format specified in the `startDatePattern` and the format of the date specified in the `endDate` parameter must match the `endDatePattern` format.

Document

Use **Document** services to perform operations on documents.

Summary of Document Services

The following **Document** services are available:

Service	Description
findDocuments	Searches a set of documents for entries matching a set of Criteria.
insertDocument	Inserts a new document in a set of documents at a specified position.
deleteDocuments	Deletes the specified documents from a set of documents.
documentListToDocument	Constructs a document from a document list by generating key/value pairs from the values of two elements that you specify in the document list.
documentToDocumentList	Expands the contents of a document into a list of documents. Each key/value pair in the source document is transformed to a single document containing two keys (whose names you specify). These two keys will contain the key name and value of the original pair.
groupDocuments	Groups a set of documents based on specified criteria.
documentToBytes	Converts a document to an array of bytes.
bytesToDocument	Converts an array of bytes to a document.

Service	Description
searchDocuments	Searches a set of documents for entries matching a set of Criteria.
sortDocuments	Sorts a set of input documents based on the specified sortCriteria.

findDocuments

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

Input Parameters

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

Output Parameters

result **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

documents **Document List** Set of documents in which a new document is to be inserted.

insertDocument **Document** The new document to be inserted to the set of documents specified in the *documents* parameter.

index **String** Optional. The position in the set which the document is to be inserted.

The *index* parameter is zero-based. If the value for the *index* parameter is not specified, the document will be inserted at the end of the document list specified in the *documents* parameter.

Output Parameters

documents **Document List** Document list after inserting the new document.

deleteDocuments

Deletes the specified documents from a set of documents.

Input Parameters

documents **Document List** Set of documents that contain the documents you want to delete.

indices **String List** Index values of documents to be deleted from the *documents* parameter document list.

Output Parameters

documents **Document List** List of documents whose indices do *not* match the values in *indices* parameter.

deletedDocuments **Document List** List of deleted documents.

Usage Notes

The deleteDocuments service returns an error if the *indices* parameter value is less than zero or more than the number of documents in the *documents* input parameter.

documentListToDocument

Constructs a document from a document list by generating key/value pairs from the values of two elements that you specify in the document list.

Input Parameters

documentList **Document List** Set of documents that you want to transform into a single document.

Note: If the *documentList* parameter contains a single document instead of a Document List, the documentListToDocument service does nothing.

name **String** Name of the element in the *documentList* parameter whose value provides the name of each key in the resulting document.

Important: The data type of the element that you specify in the *name* parameter must be String.

value **String** Name of the element in the *documentList* parameter whose values will be assigned to the keys specified in *name*. This element can be of any data type.

Output Parameters

document **Document** Document containing the key/value pairs generated from the *documentList* parameter.

Usage Notes

The following example illustrates how the `documentListToDocument` service would convert a document list that contains three documents to a single document containing three key/value pairs. When you use the `documentListToDocument` service, you specify which two elements from the source list are to be transformed into the keys and values in the output document. In the following example, the values from the `pName` elements in the source list are transformed into key names, and the values from the `pValue` elements are transformed into the values for these keys.

A `documentList` containing these three documents:

Key	Value
<i>pName</i>	<code>cx_timeout</code>

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

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

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

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

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

Would be converted to a document containing these three key:

Key	Value
<i>cx_timeout</i>	1000
<i>cx_max</i>	2500
<i>cx_min</i>	10

documentToDocumentList

Expands the contents of a document into a list of documents.

Each key/value pair in the source document is transformed to a single document containing two keys (whose names you specify). These two keys will contain the key name and value of the original pair.

Input Parameters

<i>document</i>	Document Document to transform.
<i>name</i>	String Name to assign to the key that will receive the key name from the original key/value pair. In the example above, this parameter was set to <code>pName</code> .
<i>value</i>	String Name to assign to the key that will receive the value from the original key/value pair. In the example above, this parameter was set to <code>pValue</code> .

Output Parameters

<i>documentList</i>	Document List List containing a document for each key/value pair in the <i>document</i> parameter. Each document in the list will contain two keys, whose names were specified by the <i>name</i> and <i>value</i> parameters. The values of these two keys will be the name and value (respectively) of the original pair.
---------------------	--

Usage Notes

The following example shows how a document containing three keys would be converted to a document list containing three documents. In this example, the names *pName* and *pValue* are specified as names for the two new keys in the document list.

A document containing these three keys:

Key	Value
<i>cx_timeout</i>	1000
<i>cx_max</i>	2500
<i>cx_min</i>	10

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

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

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

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

groupDocuments

Groups a set of documents based on specified criteria.

Input Parameters

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

Note: If *key* is not found in all the input documents, the documents that do not match the *groupCriteria* are grouped together as a single group.

Output Parameters

documentGroups **Document List** List of documents where each element represents a set of documents grouped based on the criteria specified.

Usage Notes

The following example illustrates how to specify the values for the *groupCriteria* parameter:

key	compareStringsAs	pattern
name	string	
age	numeric	
birthdate	datetime	yyyy-MM-dd

The input documents will be grouped based on name, age, and birth date.

documentToBytes

Converts a document to an array of bytes.

Input Parameters

document **Document** Document to convert to bytes.

- If *document* is null, the service does not return an output or an error message.
- If *document* is not a document, the service throws an exception.
- If *document* contains no elements, the service produces a zero-length byte array.

Output Parameters

documentBytes **Object** A serialized representation of the document as an array of bytes (byte[]).

Usage Notes

Use the `documentToBytes` service with the `bytesToDocument` service, which converts the byte array created by this service back into the original document.

The `documentToBytes` service is useful when you want to write a document to a file, an input stream, or a cache.

In order for the document-to-bytes-to-document conversion to work, the entire content of the document must be serializable. Every object in the document must be of a data type known to Integration Cloud, or it must support the `java.io.Serializable` interface. If Integration Cloud encounters an unknown object in the document that does not support the `java.io.Serializable` interface, that object's value will be lost. Integration Cloud will replace it with a string containing the object's class name.

bytesToDocument

Converts an array of bytes to a document. This service can only be used with byte arrays created by executing the `documentToBytes` service.

Input Parameters

documentBytes **Object** An array of bytes (byte[]) to convert to a document.

- If *documentBytes* is null, the service does not return a document or an error message.
- If *documentBytes* is not a byte array, the service throws an exception.
- If *documentBytes* is zero-length, the service produces an empty document.

Output Parameters

document **Document** A document.

Usage Notes

Use this service with the `documentToBytes` service, which converts a document into a byte array. You can pass the resulting byte array to the `bytesToDocument` service to convert it back into the original document.

In order for the document-to-bytes-to-document conversion to work, the entire content of the document must be serializable. Every object in the document must be of a data type known to Integration Cloud, or it must support the `java.io.Serializable` interface.

Note: If Integration Cloud encounters an unknown object in the document that does not support the `java.io.Serializable` interface, that object's value will be lost. It will be replaced with a string containing the object's class name.

searchDocuments

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

Input Parameters

<i>documents</i>	Document List Set of documents from which the documents meeting the search criteria are to be returned.
<i>searchCriteria</i>	<p>Document Criteria on which the documents in the <i>documents</i> parameter are to be searched.</p> <p>Valid values for <i>searchCriteria</i> parameters are:</p> <ul style="list-style-type: none"> ■ <i>key</i>. Name of the element in <code>documentList</code> whose value provides the value for the search text. The value for <i>key</i> can be a path expression. For example, "Family/Chidren[0]/BirthDate" retrieves the birthday of the first child from the input Family document list. ■ <i>value</i>. Optional. Any search text. If no value is specified, the service searches for null in the document list. ■ <i>compareStringsAs</i>. Optional. Allowed values are <code>string</code>, <code>numeric</code>, and <code>datetime</code>. The default value is <code>string</code>. ■ <i>pattern</i>. Optional. <i>pattern</i> will be considered only if the <i>compareStringsAs</i> value is of type <code>datetime</code>. For information about using patterns, see the <i>Time Zones</i> section.
<i>sorted</i>	String Optional. The value of the <i>sorted</i> parameter is <code>true</code> if the document list is already sorted based on the search criteria and same search key; otherwise <code>false</code> .

If the value for the *sorted* parameter is set to `true`, the required documents are searched faster.

Output Parameters

<i>resultdocuments</i>	Document List List of documents which are matching the search criteria.
<i>documentListIndices</i>	String List Positions of search documents in the document list.
<i>documents</i>	Document List List of documents that were input.

Usage Note

For example, if you want to search a set of documents for documents where BirthDate is 10th January 2008, the values for the *searchCriteria* parameter would be:

<u>key</u>	<u>value</u>	<u>compareStringsAs</u>	<u>pattern</u>
Birthdate	2008-01-10	datetime	yyyy-MM-dd

sortDocuments

Sorts a set of input documents based on the specified sortCriteria.

Input Parameters

<i>documents</i>	Document List Set of documents that are to be sorted.
<i>sortCriteria</i>	<p>Document List Criteria based on which the documents in the <i>documents</i> parameter are to be sorted.</p> <p>Valid values for <i>sortCriteria</i> parameters are:</p> <ul style="list-style-type: none"> ■ <i>key</i>. Name of the element in documentList whose value provides the value based on which the documents are to be sorted. The value for <i>key</i> can be a path expression. For example, "Family/Children[0]/BirthDate" retrieves the birthday of the first child from the input Family document list. ■ <i>order</i>. Optional. Allowed values are <code>ascending</code> and <code>descending</code>. The default value is <code>ascending</code>.

- *compareStringsAs* . Optional. Allowed values are `string`, `numeric`, and `datetime`. Default value is `string`.
- *pattern* . Optional. The value for *pattern* will be considered only if the *compareStringsAs* value is of type `datetime`.

Note: If *key* is not found in all the input documents, the sorted list of documents appears at the end or start of the list based on the *order* specified. If the order is ascending, then all the documents that do not match the sort criteria appears at the top of the list, followed by the sorted list. If the order is descending, the sorted list will appear at the top, followed by the documents that do not match the sort criteria.

Output Parameters

documents **Document List** The documents sorted based on the sort criteria specified in the *sortCriteria* parameter.

Usage Notes

For example, if you want to sort a set of documents based on name, age, and then on birth date, the values for *sortCriteria* parameter would be:

<u>key</u>	<u>order</u>	<u>compareStringsAs</u>	<u>pattern</u>
Name	ascending	string	
Age	descending	numeric	
Birthdate	ascending	datetime	yyyy-MM-dd

List

Use **List** services to retrieve, replace, or add elements in an Object List, Document List, or String List, including converting String Lists to Document Lists.

Summary of List services

The following **List** services are available:

Service	Description
addItemToVector	Adds an item or a list of items to a <code>java.util.Vector</code> object.
appendToDocumentList	Adds documents to a document list.
appendToStringList	Adds Strings to a String list.
sizeOfList	Returns the number of elements in a list.
stringListToDocumentList	Converts a String list to a document list.
vectorToArray	Converts a <code>java.util.Vector</code> object to an array.

addItemToVector

Adds an item or a list of items to a `java.util.Vector` object.

Input Parameters

<i>vector</i>	java.util.Vector Optional. The vector object to which you want to add an item or list of items. If no value is specified, the service creates a new <code>java.util.Vector</code> object to which the item(s) will be added.
<i>item</i>	Object Optional. Item to be added to the vector object.
	Note: You can use either <i>item</i> or <i>itemList</i> to specify the input object. If both <i>item</i> and <i>itemList</i> input parameters are specified, the item as well as the list of items will be added to the vector object.
<i>itemList</i>	Object[] Optional. List of items to be added to the vector object.
<i>addNulls</i>	String Optional. Specifies whether a null item can be added to the vector object. Set to: <ul style="list-style-type: none"> ■ <code>false</code> to prevent null values from being added to the vector object. This is the default. ■ <code>true</code> to allow null values to be added to the vector object.

Output Parameters

vector **java.util.Vector** Updated vector object with the list of items added or an empty vector in case no items are added.

Usage Notes

Either of the optional input parameters, *item* or *itemList*, is required.

appendToDocumentList

Adds documents to a document list.

Input Parameters

toList **Document List** Optional. List to which you want to append documents. If you do not specify *toList*, the service creates a new list.

fromList **Document List** Optional. Documents you want to append to the end of *toList*.

fromItem **Document** Optional. Document you want to append to the end of *toList*. If you specify both *fromList* and *fromItem*, the service adds the document specified in *fromItem* after the documents in *fromList*.

Output Parameters

toList **Document List** The *toList* document list with the documents in *fromList* and *fromItem* appended to it.

Usage Notes

The documents contained in *fromList* and *fromItem* are not actually appended as entries to *toList*. Instead, references to the documents in *fromList* and *fromItem* are appended as entries to *toList*. Consequently, any changes made to the documents in *fromList* and *fromItem* also affect the resulting *toList*.

appendToStringList

Adds Strings to a String list.

Input Parameters

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

Output Parameters

<i>toList</i>	String List The <i>toList</i> String list with the Strings from <i>fromList</i> and <i>fromItem</i> 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

<i>fromList</i>	Document List, String List, or Object List Optional. List whose size you want to discover. If <i>fromList</i> is not specified, the service returns a <i>size</i> of 0.
-----------------	--

Output Parameters

<i>size</i>	String Number of entries in <i>fromList</i> .
<i>fromList</i>	Document List, String List, or Object List Original list.

Usage Notes

For example, if *fromList* consists of:

```
fromList [0] = "a"
```

```
fromList [1] = "b"
```

```
fromList [2] = "c"
```

The result would be:

```
size = "3"
```

stringListToDocumentList

Converts a String list to a document list.

Input Parameters

<i>fromList</i>	String List Optional. List of Strings (a <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>	String Optional. Key name to use in the generated document list.

Output Parameters

<i>toList</i>	Document List Resulting document list.
---------------	---

Usage Notes

Creates a document list containing one document for each element in the *fromList* . Each document will contain a single String element named *key* .

For example, if *fromList* consists of:

```
fromList [0] = "a"
```

```

fromList [1] = "b"
fromList [2] = "c"
key = "myKey"

```

The result would be:

▼  toList []	
▼  toList[0]	
 myKey	a
▼  toList[1]	
 myKey	b
▼  toList[2]	
 myKey	c

vectorToArray

Converts a java.util.Vector object to an array.

Input Parameters

<i>vector</i>	java.util.Vector The object to be converted to an array.
<i>stronglyType</i>	<p>String Optional. If this option is specified, the service expects all items in the vector to have the same Java type as the first non-null item in the vector. If the service detects an item of a different type, an error appears.</p> <p>Set to:</p> <ul style="list-style-type: none"> ■ <code>false</code> to convert the vector to an object array. This is the default. ■ <code>true</code> to convert the vector to a strongly typed array holding the same type of objects.

Output Parameters

<i>array</i>	Object[] Converted object array.
--------------	--

Math

Use **Math** services to perform mathematical operations on string-based numeric values. Services that operate on integer values use Java's long data type (64-bit, two's complement). Services that operate on float values use Java's double data type (64-bit IEEE 754). If extremely precise calculations are critical to your application, you should write your own Java services to perform math functions.

Summary of Math services

The following **Math** services are available:

Service	Description
addObjects	Adds one java.lang.Number object to another and returns the sum.
divideObjects	Divides one java.lang.Number object by another (<i>num1/num2</i>) and returns the quotient.
min	Returns the smallest number from a list of numbers.
multiplyObjects	Multiplies one java.lang.Number object by another and returns the product.
subtractObjects	Subtracts one java.lang.Number object from another and returns the difference.
toNumber	Converts a string to numeric data type.
absoluteValue	Returns the absolute value of the input number.
addFloatList	Adds a list of floating point numbers (represented in a string list) and returns the sum.
addFloats	Adds one floating point number (represented as a String) to another and returns the sum.
addIntList	Adds a list of integers (represented in a String list) and returns the sum.
addInts	Adds one integer (represented as a String) to another and returns the sum.

Service	Description
divideFloats	Divides one floating point number (represented as a String) by another (<i>num1/num2</i>) and returns the quotient.
divideInts	Divides one integer (represented as a String) by another (<i>num1/num2</i>) and returns the quotient.
max	Returns the largest number from a list of numbers.
multiplyFloatList	Multiplies a list of floating point numbers (represented in a String list) and returns the product.
multiplyFloats	Multiplies one floating point number (represented as String) by another and returns the product.
multiplyIntList	Multiplies a list of integers (represented in a String list) and returns the product.
multiplyInts	Multiplies one integer (represented as a String) by another and returns the product.
randomDouble	Returns the next pseudorandom, uniformly distributed double between 0.0 and 1.0.
roundNumber	Returns a rounded number.
subtractFloats	Subtracts one floating point number (represented as a String) from another and returns the difference.
subtractInts	Subtracts one integer (represented as a String) from another and returns the difference.

addObjects

Adds one `java.lang.Number` object to another and returns the sum.

Input Parameters

num1 **java.lang.Number** Number to add. See the Usage Notes for supported subclasses.

num2 **java.lang.Number** Number to add. See the Usage Notes for supported sub-classes.

Output Parameters

value **java.lang.Number** Sum of the numeric values of *num1* and *num2*.

Usage Notes

This service accepts the following sub-classes of `java.lang.Number`: `java.lang.Byte`, `java.lang.Double`, `java.lang.Float`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Short`.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type `Double`, the other is converted to `Double`.
- Otherwise, if either operand is of type `Float`, the other is converted to `Float`.
- Otherwise, if either operand is of type `Long`, the other is converted to `Long`.
- Otherwise, both operands are converted to type `Integer`.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

divideObjects

Divides one `java.lang.Number` object by another ($num1/num2$) and returns the quotient.

Input Parameters

num1 **java.lang.Number** Number that is the dividend. See the Usage Notes for supported sub-classes.

num2 **java.lang.Number** Number that is the divisor. See the Usage Notes for supported sub-classes.

Output Parameters

value **java.lang.Number** Quotient of $num1 / num2$.

Usage Notes

This service accepts the following sub-classes of `java.lang.Number`: `java.lang.Byte`, `java.lang.Double`, `java.lang.Float`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Short`.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type Double, the other is converted to Double.
- Otherwise, if either operand is of type Float, the other is converted to Float.
- Otherwise, if either operand is of type Long, the other is converted to Long.
- Otherwise, both operands are converted to type Integer.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

min

Returns the smallest number from a list of numbers.

Input Parameters

numList **String List** List of numbers from which the smallest number is to be returned.

Output Parameters

minValue **String** Smallest number from the list of numbers.

multiplyObjects

Multiplies one `java.lang.Number` object by another and returns the product.

Input Parameters

num1 **java.lang.Number** Number to multiply. See the Usage Notes for supported sub-classes.

num2 **java.lang.Number** Number to multiply. See the Usage Notes for supported sub-classes.

Output Parameters

value **java.lang.Number** Product of *num1* and *num2* .

Usage Notes

This service accepts the following sub-classes of `java.lang.Number`: `java.lang.Byte`, `java.lang.Double`, `java.lang.Float`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Short`.

This service applies the following rules for binary numeric promotion to the operands in order:

- If either operand is of type `Double`, the other is converted to `Double`.
- Otherwise, if either operand is of type `Float`, the other is converted to `Float`.
- Otherwise, if either operand is of type `Long`, the other is converted to `Long`.
- Otherwise, both operands are converted to type `Integer`.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

subtractObjects

Subtracts one `java.lang.Number` object from another and returns the difference.

Input Parameters

num1 **java.lang.Number** Number. See the Usage Notes for supported sub-classes.

num2 **java.lang.Number** Number to subtract from *num1* . See the Usage Notes for supported sub-classes.

Output Parameters

value **java.lang.Number** Difference of *num1* - *num2* .

Usage Notes

This service accepts the following sub-classes of `java.lang.Number`: `java.lang.Byte`, `java.lang.Double`, `java.lang.Float`, `java.lang.Integer`, `java.lang.Long`, `java.lang.Short`.

This service applies the following rules for binary numeric promotion to the operands. The following rules are applied in order:

- If either operand is of type `Double`, the other is converted to `Double`.
- Otherwise, if either operand is of type `Float`, the other is converted to `Float`.
- Otherwise, if either operand is of type `Long`, the other is converted to `Long`.
- Otherwise, both operands are converted to type `Integer`.

These promotion rules mirror the Java rules for numeric promotion of numeric types.

toNumber

Converts a string to numeric data type.

Input Parameters

<i>num</i>	String Number (represented as a string) to be converted to numeric format.
<i>convertAs</i>	String Optional. Specifies the Java numeric data type to which the <i>num</i> parameter is to be converted. Valid values for the <i>convertAs</i> parameter are <code>java.lang.Double</code> , <code>java.lang.Float</code> , <code>java.lang.Integer</code> , <code>java.math.BigDecimal</code> , <code>java.math.BigInteger</code> , <code>java.lang.Long</code> . The default value is <code>java.lang.Double</code> .

Output Parameters

<i>num</i>	java.lang.Number Converted numeric object.
------------	---

absoluteValue

Returns the absolute value of the input number.

Input Parameters

<i>num</i>	String Number whose absolute value is to be returned.
------------	--

Output Parameters

<i>positiveNumber</i>	String Absolute value of the input number.
-----------------------	---

addFloatList

Adds a list of floating point numbers (represented in a string list) and returns the sum.

Input Parameters

numList **String List** Numbers (floating point numbers represented in a string list) to add.

Output Parameters

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

Value	Description
Infinity	The computation produces a positive value that overflows the representable range of a float type.
-Infinity	The computation produces a negative value that overflows the representable range of a float type.
0.0	The computation produces a value that underflows the representable range of a float type (for example, adding a number to infinity).
NaN	The computation produces a value that cannot be represented as a number (for example, any operation that uses NaN as input, such as $10.0 + \text{NaN} = \text{NaN}$).

Usage Notes

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern `-####.##`). Passing locally formatted strings may result in unexpected results. For example, calling `addFloats` in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

addFloats

Adds one floating point number (represented as a String) to another and returns the sum.

Input Parameters

num1 **String** Number to add.

<i>num2</i>	String Number to add.
<i>precision</i>	String Optional. Number of decimal places to which the sum will be rounded. The default value is null.

Output Parameters

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

<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, 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

<i>numList</i>	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

<i>num1</i>	String Number (floating point number represented as a String) to multiply.
<i>num2</i>	String Number (floating point number represented as a String) to multiply.
<i>precision</i>	String Optional. Number of decimal places to which the product will be rounded. The default value is null.

Output Parameters

value **String** Product of the numeric values of *num1* and *num2* . If a product cannot be produced, *value* contains one of the following:

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

Usage Notes

Make sure the strings that are passed to the service in *num1* and *num2* are in a locale-neutral format (that is, using the pattern -####.##). Passing locally formatted strings may result in unexpected results. For example, calling `addFloats` in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

multiplyIntList

Multiplies a list of integers (represented in a String list) and returns the product.

Input Parameters

numList **String List** Numbers (floating point numbers represented as Strings) to multiply.

Output Parameters

value **String** Product of the numbers in *numList*.

Usage Notes

Make sure the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Make sure the strings that are passed to the service in *numList* are in a locale-neutral format (that is, using the pattern -####.##). Passing locally formatted strings may result in unexpected results. For example, calling `addFloats` in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

multiplyInts

Multiplies one integer (represented as a String) by another and returns the product.

Input Parameters

num1 **String** Number (integer represented as a String) to multiply.

num2 **String** Number (integer represented as a String) to multiply.

Output Parameters

value **String** Product of *num1* and *num2*.

Usage Notes

Make sure the result of your calculation is less than 64 bits in width (the maximum width for the long data type). If the result exceeds this limit, it will generate a data overflow.

Make sure the strings that are passed to the service in *num1* and *num2* are in a locale-neutral format (that is, using the pattern -####.##). Passing locally formatted strings may result in unexpected results. For example, calling `addFloats` in a German locale with the arguments `1,23` and `2,34` will result in the value `357`, not `3.57` or `3,57`.

randomDouble

Returns the next pseudorandom, uniformly distributed double between 0.0 and 1.0.

Random number generators are often referred to as pseudorandom number generators because the numbers produced tend to repeat themselves over time.

Input Parameters

None.

Output Parameters

number **String** Generated random number.

roundNumber

Returns a rounded number.

Input Parameters

num **String** Number to be rounded.

numberOfDigits **String** Specifies the number of digits to which you want to round the number.

roundingMode **String** Optional. Specifies the rounding method.

Valid values for the *roundingMode* parameter are `RoundHalfUp`, `RoundUp`, `RoundDown`, `RoundCeiling`, `RoundFloor`, `RoundHalfDown`, and `RoundHalfEven`. The default value is `RoundHalfUp`.

Output Parameters

roundedNumber **String** The rounded number.

subtractFloats

Subtracts one floating point number (represented as a `String`) from another and returns the difference.

Input Parameters

num1 **String** Number (floating point number represented as a `String`).

num2 **String** Number (floating point number represented as a `String`) to subtract from *num1*.

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

Output Parameters

value **String** Difference of *num1* - *num2*. If a difference cannot be produced, *value* contains one of the following:

<u>Value</u>	<u>Description</u>
<code>Infinity</code>	The computation produces a positive value that overflows the representable range of a float type.
<code>-Infinity</code>	The computation produces a negative value that overflows the representable range of a float type.
<code>0.0</code>	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`.

Storage

Use **Storage** services to insert, retrieve, update, and remove entries from a data store.

When using the storage services, keep in mind that the short-term store is not intended to be used as a general-purpose storage engine. Rather, it is primarily provided to support shared storage of application resources and transient data in Integration Cloud.

It is recommended not to use the short-term store to process high volumes, large data records, or to permanently archive records.

Note: User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.

Note: These services are a tool for maintaining state information in the short-term store. It is up to the developer of the Integration to make sure that the Integration keeps track of its state and correctly handles restarts.

Locking Considerations

The following sections describe in general how the storage services handle locking requests.

Entry Locking

To maintain data integrity, the short-term store uses locking to ensure that multiple threads do not modify the same entry at the same time. For insertions and removals, the short-term store sets and releases the lock. For updates, the client must set and release the lock. Using locking improperly, that is, creating a lock but not releasing it, can cause deadlocks in the short-term store.

The following guidelines can help you avoid short-term store deadlocks:

- Release locks in the thread through which they were set. In other words, you cannot set a lock in one thread and release it in another. The safest way to do this is to release each lock in the Integration that acquired it.
- Unlock entries before the Integration completes. Entries remain locked until released using a put or an explicit unlock. To accomplish this, always pair a call to get or lock with a call to put or unlock so that every lock is followed by an unlock. In addition, use a try-catch pattern in your Integration so that an exception does not prevent the Integration from continuing and releasing the lock.

Data Store Locking

When a storage service locks an entry, the service also implicitly locks the data store in which the entry resides. This behavior prevents another thread from deleting the entire data store and the entries it contains while your thread is working with the entry. When the locked entry is unlocked, the implicit lock on the data store is also released.

Be careful when explicitly unlocking data stores. Consider the following example:

1. User_A locks an item. This creates two locks: an explicit lock on the entry, and an implicit lock on the data store.
2. User_A later unlocks the data store explicitly while still holding the lock on the entry.
3. User_B locks, then deletes the data store, including the entry locked by User_A in the first step.

When User_A explicitly unlocked the data store in step 2, User_B was able to delete the entry the User_A was working with.

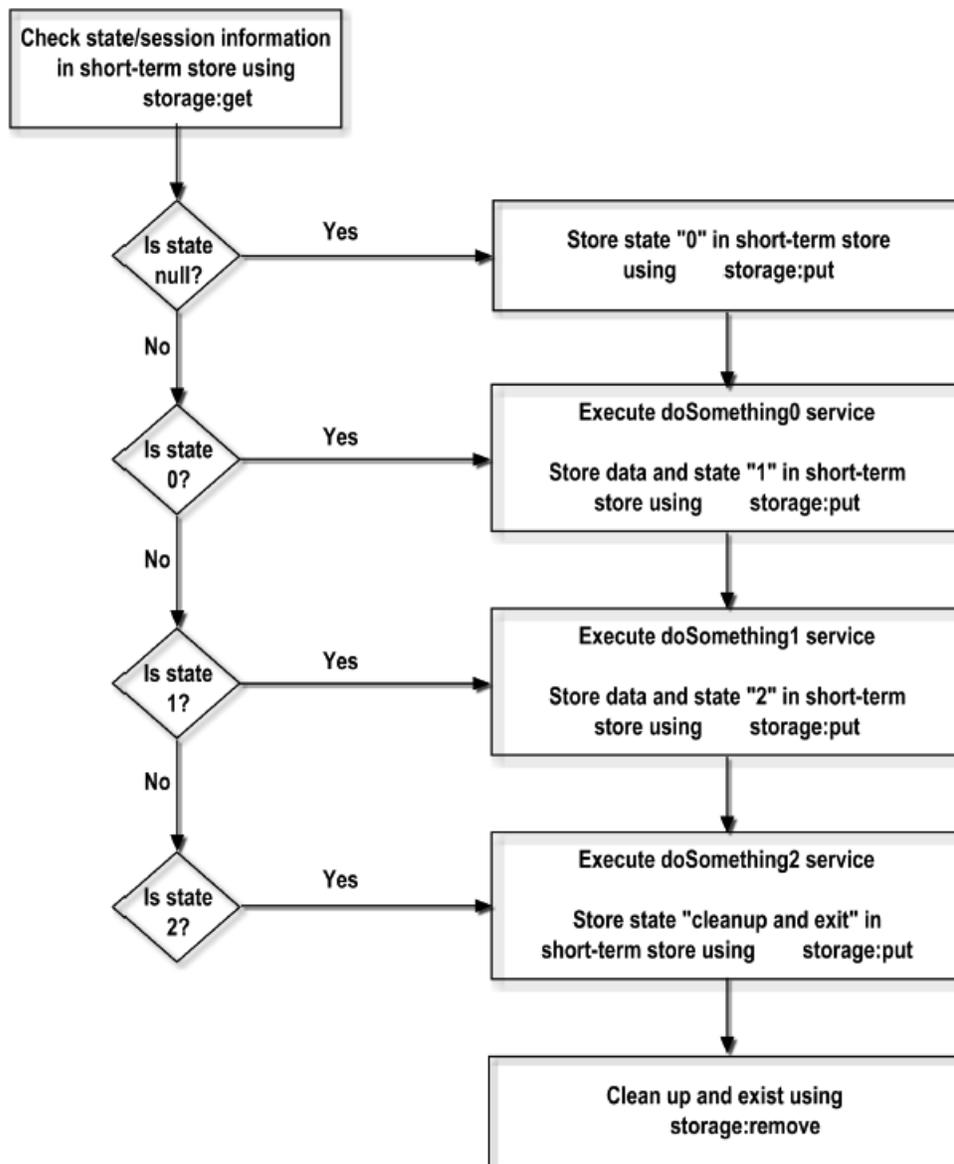
Automatic Promotion to Exclusive Lock

If a storage service tries to acquire an exclusive lock on an object, but finds a shared lock from the same thread already in place on the object, the service will try to promote the lock to an exclusive lock.

If a storage service that requires an exclusive lock encounters a shared or exclusive lock held by another thread, it will wait until the object becomes available. If the object remains locked for the period specified by the *waitlength* parameter passed by the service, the service will fail.

Sample Integration for Checkpoint Restart

The following diagram shows how to create checkpoint restarts into your Integrations. The following diagram explains the logic of an Integration and shows where the various storage services are used to achieve checkpoint restarts.

Logic to achieve checkpoint restart**Summary of Storage services**

The following **Storage** services are available:

Element	Package and Description
add	Inserts a new entry into a data store.

Element	Package and Description
get	Retrieves a value from a data store and locks the entry and the data store on behalf of the thread that invoked the service.
keys	Obtains a list of all the keys in a data store.
lock	Locks an entry and/or data store on behalf of the thread invoking this service.
put	Inserts or updates an entry in a data store. If the key does not exist in the data store, the entry is inserted.
remove	Removes an entry from a data store.
unlock	Unlocks an entry or a data store.

add

Inserts a new entry into a data store.

If the key already exists in the data store, the service does nothing.

Input Parameters

<i>storeName</i>	String Name of the data store in which to insert the entry.
<i>key</i>	String Key under which the entry is to be inserted.
<i>value</i>	Document Value to be inserted.

Output Parameters

<i>result</i>	<p>String Flag indicating whether the entry was successfully added. A value of:</p> <ul style="list-style-type: none"> ■ <code>true</code> indicates that the new entry was inserted successfully. ■ <code>false</code> indicates that the entry was not inserted (usually because an entry for <i>key</i> already exists).
---------------	--

error **String** Error message generated while inserting the new entry into the data store.

get

Retrieves a value from a data store and locks the entry and the data store on behalf of the thread that invoked the service.

Important: This service does not automatically release the lock on the data store or entry after performing the get operation, so you need to ensure that the lock is released by calling the put or unlock services. If you do not release the lock, Integration Cloud will release the lock at the end of the Integration execution.

Input Parameters

<i>storeName</i>	String Name of the data store from which you want to retrieve the entry.
<i>key</i>	String Key of the entry whose value you want to retrieve.
<i>waitLength</i>	String Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread.
<i>lockMode</i>	String Optional. Type of lock you want to place on the entry. Set to: <ul style="list-style-type: none">■ Exclusive to prevent other threads from reading or updating the entry while you are using it. The service also obtains a shared lock on the data store. An exclusive lock on an entry allows you to modify the entry.■ Read is obsolete. If this value is specified, the service obtains a shared lock.■ Share to prevent other threads from obtaining an exclusive lock on the entry. The service also obtains a shared lock on the data store. A shared lock on an entry allows you to read, but not modify, the entry. This is the default.

Output Parameters

value **Document** Retrieved entry. If the requested entry does not exist, the value of this parameter is null.

Usage Notes

If you request an exclusive lock and the service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock on the entry to an exclusive lock.

When this service locks an entry, it also acquires a shared lock on the associated data store to prevent another thread from deleting the data store, and the entries it contains, while your thread has the entry locked.

When storing and retrieving the flow state in the short-term store for checkpoint restart purposes, ensure that the value of *key* is unique to the transaction.

keys

Obtains a list of all the keys in a data store.

Input Parameters

storeName **String** Name of the data store from which you want to obtain a list of keys.

Output Parameters

keys **String List** Keys for the data store specified in *storeName* .

lock

Locks an entry and/or data store on behalf of the thread invoking this service.

Important: When you lock an entry or data store using this service, you must release the lock by using a put or an explicit unlock. If you do not release the lock, Integration Cloud will release the lock at the end of the Integration execution.

Important: Be careful when releasing locks with the unlock service. If you release a lock on a data store, another thread can obtain a lock on the data store and delete

it, and the entries it contains, even if your thread still has locks on one or more of the entries.

Input Parameters

<i>storeName</i>	String Name of the data store containing the entry.
<i>key</i>	<p>String Optional. Key of the entry that you want to lock.</p> <p>If <i>key</i> is not supplied and you request:</p> <ul style="list-style-type: none"> ■ A shared lock, the service obtains a shared lock on the data store, allowing other threads to read and modify entries, but not to delete them. ■ An exclusive lock, the service obtains an exclusive lock on the data store, preventing other threads from locking the data store and the entries, thereby preventing those threads from reading, modifying, or deleting the entries or the data store. <p>If both <i>storeName</i> and <i>key</i> are specified and you request:</p> <ul style="list-style-type: none"> ■ A shared lock, the service obtains a shared lock on the data store and the entry. ■ An exclusive lock, the service obtains a shared lock on the data store and an exclusive lock on the entry.
<i>waitLength</i>	String Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread.
<i>lockMode</i>	<p>String Optional. Type of lock you want to place on the entry or data store. Set to:</p> <ul style="list-style-type: none"> ■ Exclusive to prevent other threads from obtaining a lock on the data store or entry. <p>An exclusive lock on an entry allows you to modify the entry, and prevents other threads from reading or modifying the entry.</p> <p>An exclusive lock on a data store also locks the entries in the data store. In addition, an exclusive lock on a data store allows you to delete the data store.</p> <ul style="list-style-type: none"> ■ Read is obsolete. If this value is specified, the service obtains a shared lock. ■ Share to prevent other threads from obtaining an exclusive lock on an entry or a data store. A shared lock on

an entry allows you to read, but not modify, the entry. A shared lock on a data store prevents another thread from deleting the data store. This is the default.

Output Parameters

None.

Usage Notes

If you have not specified a *key*, and your Integration does not invoke `put` or `unlock`, or your Integration throws an exception before invoking `put` or `unlock`, the entire data store remains locked.

If the key does not exist in the data store at the time your Integration executes, the lock service inserts the key with an empty value and takes the lock on the entry.

If you request an exclusive lock on an entry, the service obtains an exclusive lock on the entry and a shared lock on the data store. If this service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock on the entry to an exclusive lock.

If you request a shared lock on an entry, the service obtains a shared lock on the entry and a shared lock on the data store.

If you request a shared lock on an entry or a data store and this service finds an exclusive lock from the same thread, the existing exclusive lock will be reused. The exclusive lock will not be demoted to a shared lock.

If you request an exclusive lock on a data store, and this service finds a shared lock from the same thread on the data store, the service will automatically promote the shared lock on the data store to an exclusive lock.

put

Inserts or updates an entry in a data store. If the key does not exist in the data store, the entry is inserted.

If the requested entry is not currently locked by the thread that invoked this service, the `put` service will automatically attempt to lock the entry for the duration of the `put` operation.

The service obtains an exclusive lock on the entry and a shared lock on the data store. If the service finds a shared lock from the same thread on the entry, the service will automatically promote the shared lock to an exclusive lock.

This service releases the lock when the `put` operation has completed.

Input Parameters

<i>storeName</i>	String Name of the data store into which you want to insert or update the entry.
<i>value</i>	Document Value to be inserted or updated.
<i>waitLength</i>	String Optional. Length of time, in milliseconds, that you want to wait for this entry to become available if it is already locked by another thread. If the wait length expires before a lock is obtained, the service fails and throws an exception. This parameter is used only when your service did not explicitly lock the entry beforehand.
<i>key</i>	String Key where you want to insert or update the entry.

Output Parameters

<i>error</i>	String Error message generated while inserting the new entry into the data store.
--------------	--

Usage Notes

When storing and retrieving the flow state in the short-term store for checkpoint restart purposes, ensure that the value of *key* is unique to the transaction.

remove

Removes an entry from a data store. This service obtains an exclusive lock on the entry and a shared lock on the data store.

Input Parameters

<i>storeName</i>	String Name of the data store from which to remove an entry.
<i>key</i>	String Key of the entry that you want to remove.
<i>waitLength</i>	String Optional. Length of time, in milliseconds, that you want to wait for this entry to become available for deletion if it is already locked by another thread.

Output Parameters

result **String** Flag indicating whether the entry was successfully removed. A value of:

- `true` indicates that the entry was removed successfully.
- `false` indicates that the entry was not removed (usually because an entry for key does not exist).

unlock

Unlocks an entry or a data store.

When an Integration retrieves an entry using the get service, the entry is locked to prevent modification by other users before the Integration completes. The entry remains locked until the lock owner invokes a put service. To unlock a service without using the put service, use the unlock service.

In addition, if an Integration uses the lock service to lock an entry or data store, you must use the unlock or put service to release the lock.

Important: Be careful when releasing locks with this service. If you release a lock on a data store, another thread can obtain a lock on the data store and delete it, and the entries it contains, even if the original thread still has locks on one or more of the entries.

Input Parameters

storeName **String** Name of the data store in which to unlock an entry.

key **String** Optional. Key of the entry that you want to unlock. If *key* is not supplied, the lock will be removed from the data store specified in *storeName*, but any locks on entries in the data store will remain.

Output Parameters

None.

String

Use **String** services to perform string manipulation and substitution operations.

Summary of String services

The following **String** services are available:

Service	Description
HTMLDecode	Replaces HTML character entities with native characters.
HTMLEncode	Replaces HTML-sensitive characters with equivalent HTML character entities.
base64Decode	Decodes a Base-64 encoded string into a sequence of bytes.
base64Encode	Converts a sequence of bytes into a Base64-encoded String.
bytesToString	Converts a sequence of bytes to a String.
concat	Concatenates two strings.
indexOf	Returns the index of the first occurrence of a sequence of characters in a string.
length	Returns the length of a string.
lookupDictionary	Looks up a given key in a hash table and returns the string to which that key is mapped.
makeString	Builds a single string by concatenating the elements of a String List.
messageFormat	Formats an array of strings into a given message pattern.
numericFormat	Formats a number into a given numeric pattern.
objectToString	Converts an object to string representation using the Java <code>toString()</code> method of the object.

Service	Description
padLeft	Pads a string to a specified length by adding pad characters to the beginning of the string.
padRight	Pads a string to a specified length by adding pad characters to the end of the string.
replace	Replaces all occurrences of a specified substring with a substitute string.
stringToBytes	Converts a string to a byte array.
substring	Returns a substring of a given string.
tokenize	Tokenizes a string using specified delimiter characters and generates a String List from the resulting tokens.
toLowerCase	Converts all characters in a given string to lowercase.
toUpperCase	Converts all characters in a given string to uppercase.
trim	Trims leading and trailing white space from a given string.
URLDecode	Decodes a URL-encoded string.
URLEncode	URL-encodes a string.
fuzzyMatch	A given string is not exactly matched against a set of strings. If the match is above <code>similarityThreshold</code> , it returns the <code>matchedValue</code> . If more than one string has not exactly matched, then the first matched string is returned.
isNumber	Determines whether the contents of a string can be converted to a float value.

Service	Description
isAlphanumeric	Determines whether a string consists entirely of alphanumeric characters (in the ranges A–Z, a–z, or 0–9).
isNullOrBlank	Checks a string for a null or a blank value.
isDate	Determines whether a string follows a specified date pattern.
substitutePipelineVariables	Replaces a pipeline variable with its corresponding value.
compareStrings	Performs a case-sensitive comparison of two strings, and indicates whether the strings are identical.

HTMLDecode

Replaces HTML character entities with native characters.

Specifically, the service:

Replaces this HTML character entity...	With...
>	>
<	<
&	&
"	"

Input Parameters

inString **String** An HTML-encoded String.

Output Parameters

value **String** Result from decoding the contents of *inString*. Any HTML character entities that existed in *inString* will appear as native characters in *value*.

HTMLEncode

Replaces HTML-sensitive characters with equivalent HTML character entities.

Specifically, this service:

<u>Replaces this native language character...</u>	<u>With...</u>
>	>
<	<
&	&
"	"
'	'

These translations are useful when displaying text in an HTML context.

Input Parameters

inString **String** The character you want to encode in HTML.

Output Parameters

value **String** Result from encoding the contents of *inString*. Any HTML-sensitive characters that existed in *inString*, for example, > or &, will appear as the equivalent HTML character entities in *value*.

base64Decode

Decodes a Base-64 encoded string into a sequence of bytes.

Input Parameters

string **String** A Base64-encoded String to decode into bytes.

Output Parameters

value **byte[]** The sequence of bytes decoded from the Base64-encoded String.

encoding **String** Optional. Specifies the encoding method. Default value is ASCII.

base64Encode

Converts a sequence of bytes into a Base64-encoded String.

Input Parameters

bytes **byte[]** Sequence of bytes to encode into a Base64-encoded String.

useNewLine **String** Optional. Flag indicating whether to retain or remove the line breaks. Set to:

- `true` to retain the line breaks. This is the default.
- `false` to remove the line breaks.

encoding **String** Optional. Specifies the encoding method. Default value is ASCII.

Output Parameters

value **String** Base64-encoded String encoded from the sequence of bytes.

Usage Notes

By default, the `base64Encode` service inserts line breaks after 76 characters of data, which is not the canonical lexical form expected by implementations such as MTOM. You can use the `useNewLine` parameter to remove the line breaks.

bytesToString

Converts a sequence of bytes to a String.

Input Parameters

<i>bytes</i>	byte[] Sequence of bytes to convert to a String.
<i>encoding</i>	String Optional. Name of a registered, IANA character set (for example, <code>ISO-8859-1</code>). If you specify an unsupported encoding, the system throws an exception. To use the default encoding, set <i>encoding</i> to <code>autoDetect</code> .
<i>ignoreBOMChars</i>	String Optional. Flag indicating whether or not the byte order mark (BOM) characters in the input sequence of bytes are removed before converting the byte array to string. Set to: <ul style="list-style-type: none">■ <code>true</code> to remove the byte order mark (BOM) characters before converting the input sequence of bytes to string, if the byte array contains BOM characters.■ <code>false</code> to include the byte order mark (BOM) characters while converting the input sequence of bytes to string. The default is <code>false</code>.

Output Parameters

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

concat

Concatenates two strings.

Input Parameters

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

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

Output Parameters

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

indexOf

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

Input Parameters

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

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

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

Output Parameters

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

length

Returns the length of a string.

Input Parameters

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

Output Parameters

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

lookupDictionary

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

Input Parameters

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

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

Note: The key is case sensitive.

Output Parameters

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

makeString

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

Input Parameters

elementList **String List** Strings to concatenate.

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

Output Parameters

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

messageFormat

Formats an array of strings into a given message pattern.

Input Parameters

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

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

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

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

Output Parameters

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

numericFormat

Formats a number into a given numeric pattern.

Input Parameters

num **String** The number to format.

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

This symbol...

Indicates...

0

A digit.

#	A digit. Leading zeroes will not be shown.
.	A placeholder for a decimal separator.
,	A placeholder for a grouping separator.
;	A separation in format.
-	The default negative prefix.
%	That <i>num</i> will be multiplied by 100 and shown as a percentage.
x	Any character used as a prefix or suffix (for example, A, \$).
'	That special characters are to be used as literals in a prefix or suffix. Enclose the special characters within " (for example, '#').

The following are examples of pattern strings:

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

denotes a digit and the period is a placeholder for decimal separator.

Output Parameters

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

objectToString

Converts an object to string representation using the Java toString() method of the object.

Input Parameters

object **Object** The object to be converted to string representation.

Output Parameters

string **String** String representation of the input object converted using the Java toString() method of the object.

padLeft

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

Input Parameters

inString **String** String that you want to pad.

padString **String** Characters to use to pad *inString* .

length **String** Total length of the resulting string, including pad characters.

Output Parameters

value **String** Contents of *inString* preceded by as many pad characters as needed so that the total length of the string equals *length* .

Usage Notes

If *padString* is longer than one character and does not fit exactly into the resulting string, the beginning of *padString* is aligned with the beginning of the resulting string. For example, suppose *inString* equals `shipped` and *padString* equals `x9y`.

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

If *inString* is longer than *length* characters, only the last *length* characters from *inString* are returned. For example, if *inString* equals `acct1234` and *length* equals 4, *value* will contain `1234`.

padRight

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

Input Parameters

inString **String** String that you want to pad.

padString **String** Characters to use to pad *inString* .

length **String** Total length of the resulting string, including pad characters.

Output Parameters

value **String** Contents of *inString* followed by as many pad characters as needed so that the total length of the string equals *length* .

Usage Notes

If *padString* is longer than one character and does not fit exactly into the resulting string, the end of *padString* is aligned with the end of the resulting string. For example, suppose *inString* equals `shipped` and *padString* equals `x9y`.

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

Output Parameters

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

stringToBytes

Converts a string to a byte array.

Input Parameters

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

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

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

Output Parameters

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

substring

Returns a substring of a given string.

Input Parameters

inString **String** String from which to extract a substring.

beginIndex **String** Beginning index of the substring to extract (inclusive).

endIndex **String** Ending index of the substring to extract (exclusive). If this parameter is null or empty, the substring will extend to the end of *inString*.

Output Parameters

value **String** Substring from *beginIndex* and extending to the character at *endIndex* - 1.

tokenize

Tokenizes a string using specified delimiter characters and generates a String List from the resulting tokens.

This service does not return delimiters as tokens.

Input Parameters

inString **String** String you want to tokenize, that is, break into delimited chunks.

delim **String** Delimiter characters. If null or empty, the service uses the default delimiters `\t\n\r`, where t, n, and r represent the white space characters tab, new line, and carriage return.

Output Parameters

valueList **String List** Strings containing the tokens extracted from *inString*.

toLower

Converts all characters in a given string to lowercase.

Input Parameters

inString **String** String to convert.

language **String** Optional. Lowercase, two-letter ISO-639 code. If this parameter is null, the system default is used.

country **String** Optional. Uppercase, two-letter ISO-3166 code. If this parameter is null, the system default is used.

variant **String** Optional. Vendor and browser-specific code. If null, this parameter is ignored.

Output Parameters

value **String** Contents of *inString*, with all uppercase characters converted to lowercase.

toUpper

Converts all characters in a given string to uppercase.

Input Parameters

inString **String** String to convert.

language **String** Optional. Lowercase, two-letter ISO-639 code. If this parameter is null, the system default is used.

country **String** Optional. Uppercase, two-letter ISO-3166 code. If this parameter is null, the system default is used.

variant **String** Optional. Vendor and browser-specific code. If null, this parameter is ignored.

Output Parameters

value **String** Contents of *inString*, with all lowercase characters converted to uppercase.

trim

Trims leading and trailing white space from a given string.

Input Parameters

inString **String** String to trim.

Output Parameters

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

URLDecode

Decodes a URL-encoded string.

Input Parameters

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

Output Parameters

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

URLEncode

URL-encodes a string.

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

Input Parameters

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

Output Parameters

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

fuzzyMatch

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

Input Parameters

<i>inString</i>	String (Required) Text to be matched. Text should not be empty or null.
<i>matchData</i>	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.
<i>similarityThreshold</i>	String (Optional) If the inexact match score is above the given threshold, then service output contains the <i>matchedValue</i> 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.
<i>algorithm</i>	String (Optional) The algorithm used for an inexact match. Default value is Levenshtein. Supported algorithms are Levenshtein and JaroWinkler.

Output Parameters

<i>matchedValue</i>	String (Optional) If the inexact match is above <i>similarityThreshold</i> , then the returned value contains the matched string.
<i>similarity</i>	String (Optional) If the inexact match is above <i>similarityThreshold</i> , then it contains a similarity score. It provides the measure of how close the match is. The returned value can be between 0.0 and 1.0. Value 0.0 represents no match and value 1.0 represents an exact match.

Usage Notes

Search the web for more information about Levenshtein and JaroWinkler algorithms.

isNumber

Determines whether the contents of a string can be converted to a float value.

Input Parameters

inString **String** Optional. String to be checked for conversion to float.

Output Parameters

isNumber **String** Indicates whether or not *inString* can be converted to a float value.

- `true` indicates that *inString* can be converted to a float value.
- `false` indicates that *inString* cannot be converted to a float value.

The service returns `false` if *inString* is not specified.

isAlphanumeric

Determines whether a string consists entirely of alphanumeric characters (in the ranges A–Z, a–z, or 0–9).

Input Parameters

inString **String** Optional. String to be checked for alphanumeric characters.

Output Parameters

isAlphanumeric **String** Indicates whether or not all the characters in *inString* are alphanumeric.

- `true` indicates that all the characters in *inString* are alphanumeric.
- `false` indicates that *not all* the characters in *inString* are alphanumeric.

The service returns `false` if *inString* is not specified.

isNullOrBlank

Checks a string for a null or a blank value.

Input Parameters

inString **String** Optional. String to be checked for a null or a blank value.

Output Parameters

isNullOrBlank **String** Indicates whether or not *inString* has a null or a blank value.

- `true` indicates that *inString* has either a null or a blank value.
- `false` indicates that *inString* contains a value that is not null.

Note: If *inString* is not specified, the service considers the string to be blank and returns `true`.

isDate

Determines whether a string follows a specified date pattern.

Input Parameters

inString **String** Optional. String to be checked for adherence to the specified date *pattern*.

pattern **String** Date format for specifying the *inString* parameter (for example, yyyyMMdd HH:mm:ss.SSS).

For more information about the pattern strings that can be specified for the date, see the “Pattern String Symbols” section.

Output Parameters

isDate **String** Indicates whether or not *inString* follows the specified date pattern.

- `true` indicates that *inString* follows the specified date pattern.
- `false` indicates that *inString* does not follow the specified date pattern.

The service returns `false` if *inString* is not specified.

Usage Notes

The service returns an error if both *inString* and *pattern* are not specified.

You can specify any random string (for example, 111212) as both *inString* and *pattern*. The service returns `true` if the same user-defined string is specified as both *inString* and *pattern*. This is because the `java.text.SimpleDateFormat` class parses the user-defined input string and pattern to a valid date when the particular input values are identical.

substitutePipelineVariables

Replaces a pipeline variable with its corresponding value.

Input Parameters

inString **String** Optional. String containing the pipeline variable to replace. Specify the name of the pipeline variable between the % symbols (for example, %phone%).

Output Parameters

value **String** Contents of *inString* with the pipeline variable replaced.

Usage Notes

The service returns an error if *inString* is not specified.

If *inString* does not contain any variable between the % symbols, or contains a value other than the pipeline variable between the % symbols, the service does not perform any variable substitution from the pipeline.

If you want to include the % symbol in the output, you can specify it as \% in *inString*. To specify the value of the pipeline variable as a percentage in the output, append \% after the variable name in *inString*. For example, suppose a pipeline variable *revenueIncreasePercent* has a value of 100.

If <i>inString</i> equals...	Then <i>value</i> will contain...
%revenueIncreasePercent%\%	100%

The service cannot be used for substitution of global variables.

compareStrings

Performs a case-sensitive comparison of two strings and indicates whether the strings are identical.

Input Parameters

<i>inString1</i>	String Optional. String to compare against <i>inString2</i> . This input variable can be null.
<i>inString2</i>	String Optional. String to compare against <i>inString1</i> . This input variable can be null.

Output Parameters

<i>isEqual</i>	<p>String Indicates whether or not <i>inString1</i> and <i>inString2</i> are identical.</p> <ul style="list-style-type: none"> ■ <code>true</code> indicates that <i>inString1</i> and <i>inString2</i> are identical. ■ <code>false</code> indicates that <i>inString1</i> and <i>inString2</i> are not identical.
----------------	--

Note: If both *inString1* and *inString2* are null, the service considers the strings to be identical and returns `true`.

Flow

Use **Flow** services to perform utility-type tasks.

Summary of Flow services

The following **Flow** services are available:

Service	Description
clearPipeline	Removes all fields from the pipeline. You may optionally specify fields that should not be cleared by this service.
getLastError	Obtains detailed information about the last error that was trapped within an Integration.
getSessionInfo	Obtains detailed information about the current logged-in user session. Also provides the current Integration name and the execution result reference identifier.
getHTTPRequest	Gets information about the HTTP request, received by Integration Cloud.
setHTTPResponse	Sets the HTTP response information to be returned by Integration Cloud.
countProcessedDocuments	Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen.
logCustomMessage	Logs a message, which can be viewed in the Execution Results screen.
sleep	Causes the currently executing Integration to pause for the specified number of seconds.

clearPipeline

Removes all fields from the pipeline. You may optionally specify fields that should not be cleared by this service.

Input Parameters

preserve **String List** Optional. Field names that should not be cleared from the pipeline.

Output Parameters

None

getLastError

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

Input Parameters

None

Output Parameters

lastError

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

<u>Key</u>	<u>Description</u>
time	String. Date and time the event occurred, in the format <i>yyyy/MM/dd HH:mm:ss.SSS</i>
error	String. Optional. Error message of the exception.
localizedError	String. Optional. Error message in the language that corresponds to the server locale.
user	String. User who executed the Integration.
block	Document. Contains the following fields:
<u>Key</u>	<u>Description</u>
name	String. Integration, Operation, or Service name.

<code>type</code>	String. Application, Integration, or Service.
<code>details</code>	String. Optional. Account and Application name if the Block Type is "Application".
callStack	Document List. The call stack information describing where the error occurred including details of the block. Each document represents a block on the call stack. The first document in the list represents the block that threw the error and the last document in the list represents the top level block. It contains the following fields:

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

Usage Notes

You can use this service in the *catch* section of the *try-catch* block. Each execution of an Integration or a service (whether the Integration or the service succeeds or fails) updates the value returned by `getLastError`. Consequently, `getLastError` itself resets the value of `lastError`. Therefore, if the results of `getLastError` will be used as input to subsequent Integrations, map the value of `lastError` to a variable in the pipeline.

If a map has multiple transformers, then a subsequent call to `getLastError` will return the error associated with the last failed transformer in the map, even if it is followed by successful transformers.

getSessionInfo

Obtains detailed information about the current logged-in user session. Also provides the current Integration execution result reference identifier.

Input Parameters

None

Output Parameters

\$session **Document** Returns information about the current logged-in user session. Also provides the current Integration name and the execution result reference identifier.

<u>Key</u>	<u>Description</u>
<i>tenantId</i>	String Tenant Identifier.
<i>stageId</i>	String The stage ID where the current integration resides.
<i>user</i>	Document Returns user details.
<u>Key</u>	<u>Description</u>
<i>name</i>	String Name of the user who is executing the service.
<i>integrationName</i>	String The name of the Integration. <div style="background-color: #f0f0f0; padding: 5px; margin-top: 5px;"> <p>Note If Integration A has a referenced Integration B, and if the getSessionInfo service is called in Integration B, then <i>integrationName</i> will be A but if Integration B is executed independently, then <i>integrationName</i> will be B.</p> </div>
<i>executionResultReference</i>	String Returns the current Integration execution result reference identifier. For example, you can pass the identifier to an on-premises operation and trace the Integration execution.

getHTTPRequest

Gets information about the HTTP request, received by Integration Cloud.

Parameters

<i>headers</i>	Document Contains the header fields from the HTTP request.
<i>requestURL</i>	String URL used by the client to invoke the service.
<i>method</i>	String HTTP method used by the client to request the top-level service. Possible values are GET, PUT, POST, PATCH, and DELETE.

setHTTPResponse

Sets the HTTP response information to be returned by Integration Cloud.

Parameters

<i>headers</i>	Document Optional. Contains the header fields to be returned in the HTTP response.
<i>responseCode</i>	String Optional. HTTP status code to be returned to the client. The response codes and phrases are defined in " https://tools.ietf.org/html/rfc7231#section-6 ". If you provide a value for <i>responseCode</i> that is not listed in RFC 7321, Section 6, you must also provide a value for <i>reasonPhrase</i> .
<i>responsePhrase</i>	String Optional. HTTP reason phrase to be returned to the client. If no reason is provided, the default reason phrase associated with <i>responseCode</i> will be used. You must provide a <i>reasonPhrase</i> for any <i>responseCode</i> that is not listed in RFC 7321, Section 6.
<i>responseString</i>	String Optional. Response to be returned to the client, specified as a string.
<i>responseBytes</i>	byte[] Optional. Response to be returned to the client, specified as a byte array.

responseStream **java.io.InputStream** Optional. Response to be returned to the client, specified as an InputStream.

countProcessedDocuments

Counts the number of documents processed by an Integration. Details about the processed documents can be viewed in the Execution Results screen.

Input Parameters

status **String** Optional valid values are "success" or "fail". Set status to "success" to count the number of successfully processed documents, else set it to "fail". Default value is "success".

incrementBy **String** Optional. Increment the number of documents processed by an Integration. Every time the service is used, successful or failed documents are incremented by the given value. Default value is 1.

Output Parameters

None

Usage Notes

To increment the number of documents processed by a list, use the **sizeOfList** service in the **List** service block.

logCustomMessage

Logs a message, which can be viewed in the Execution Results screen.

Input Parameters

message **String** Custom message to be logged, which can be viewed in the Execution Results screen.

Output Parameters

None

sleep

Causes the currently executing Integration to pause for the specified number of seconds.

Input Parameters

seconds **String** The number of seconds to pause the currently executing Integration. The value must be an integer between 1 second and 60 seconds.

Output Parameters

None

Hashtable

Use **Hashtable** services to create, update, and obtain information about the hashtable.

Summary of Hashtable services

The following **Hashtable** services are available:

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

containsKey

Checks for the existence of a hashtable element.

Input Parameters

hashtable **java.util.Hashtable** Hashtable in which to check for the existence of a hashtable element.

key **String** Hashtable element to be checked for.

Output Parameters

containsKey **String** Indicates whether the specified hashtable element exists. A value of:

- `true` indicates that the element exists.
- `false` indicates that the element does not exist.

createHashtable

Creates a hashtable object.

Input Parameters

None.

Output Parameters

hashtable **java.util.Hashtable** The new hashtable object.

get

Gets the value for a specified key in the hashtable.

Input Parameters

hashtable **java.util.Hashtable** Hashtable from which to retrieve the specified value.

key **String** Key of the hashtable element whose value is to be retrieved.

Output Parameters

value **Object** Value of the input hashtable element.

listKeys

Lists all the keys stored in the hashtable.

Input Parameters

hashtable **java.util.Hashtable** Hashtable from which the keys are to be listed.

Output Parameters

keys **String[]** List of keys stored in the input hashtable.

put

Adds a key/value pair in the hashtable.

Input Parameters

hashtable **java.util.Hashtable** Hashtable to which the key/value pair is to be added.

key **String** Key of the element to be added to the hashtable.

value **Object** Value of the element to be inserted into the hashtable.

Output Parameters

hashtable **java.util.Hashtable** Hashtable object after the insertion of the key/value pair.

remove

Removes a key/value pair from the hashtable.

Input Parameters

hashtable **java.util.Hashtable** Hashtable from which to remove the key/value pair.

key **String** Key of the hashtable element to be removed.

value **Object** Value of the hashtable element to be removed.

Output Parameters

hashtable **java.util.Hashtable** Hashtable object after the key/value pair is removed.

value **Object** Value of the hashtable element that was removed. Returns `null` if the input *key* is not found in the hashtable.

size

Gets the number of elements in the hashtable.

Input Parameters

hashtable **java.util.Hashtable** Hashtable from which the number of elements stored in it is to be retrieved.

Output Parameters

size **String** Number of elements in the hashtable.

Flat File

Use **Flat File** services to convert data bytes, data stream, and data string to a document and vice versa.

Summary of Flat File services

The following **Flat File** services are available:

Service	Description
delimitedDataBytesToDocument	Converts delimited data bytes (byte array) to a document.
delimitedDataStreamToDocument	Converts delimited data stream to a document.
delimitedDataStringToDocument	Converts delimited data string to a document.
documentToDelimitedDataBytes	Converts a document to delimited data bytes (byte array object).
documentToDelimitedDataStream	Converts a document to a delimited data stream.
documentToDelimitedDataString	Converts a document to a delimited data string.

delimitedDataBytesToDocument

Converts delimited data bytes (byte array) to a document.

This service will convert the following delimited data from byte array:

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Jane
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

or to a document that looks like: (useHeaderRowForFieldNames=false)

▼  document	
▼  rows []	
▼  rows[0]	
 column1	Date
 column2	Pupil
 column3	Grade
▼  rows[1]	
 column1	25 May
 column2	Bloggs, Fred
 column3	C
▼  rows[2]	
 column1	25 May
 column2	Doe, Joe
 column3	B
▼  rows[3]	
 column1	15 July
 column2	Bloggs, Fred
 column3	A

Input Parameters

delimitedDataBytes

java.lang.Byte[]. Delimited data in bytes (Byte array) to convert to a document.

fieldQualifier

String Optional. The delimiter to use for separating entries in *delimitedDataBytes*. Default is comma (,).

textQualifier

String Optional. The character to use for quoted elements. Default is double quote (").

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

Output Parameters

<i>document</i>	Document. Document resulting from the conversion of <i>delimitedDataBytes</i> . This document contains document array rows[] corresponding to the delimited data.
-----------------	--

delimitedDataStreamToDocument

Converts delimited data stream to a document. The permissible size of the content stream is based on your tenancy. The permissible size of the content stream is based on your tenancy.

This service converts the following delimited data in a stream:

```
"Date","Pupil","Grade"
```

```
"25 May","Bloggs, Fred","C"
```

```
"25 May","Doe, Jane","B"
```

```
"15 July","Bloggs, Fred","A"
```

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Joe
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

or to a document that looks like: (useHeaderRowForFieldNames=false)

▼  document	
▼  rows []	
▼  rows[0]	
 column1	Date
 column2	Pupil
 column3	Grade
▼  rows[1]	
 column1	25 May
 column2	Bloggs, Fred
 column3	C
▼  rows[2]	
 column1	25 May
 column2	Doe, Joe
 column3	B
▼  rows[3]	
 column1	15 July
 column2	Bloggs, Fred
 column3	A

Input Parameters

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

fieldQualifier **String** Optional. The delimiter to use for separating entries in *delimitedDataStream*. Default is comma (,).

textQualifier **String** Optional. The character to use for quoted elements. Default is double quote (").

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

Output Parameters

<i>document</i>	Document. Document resulting from the conversion of <i>delimitedDataStream</i> . This document contains document array rows[] corresponding to the delimited data.
-----------------	---

delimitedDataStringToDocument

Converts delimited data string to a document.

This service will convert the following delimited data string:

```
"Date","Pupil","Grade"
```

```
"25 May","Bloggs, Fred","C"
```

```
"25 May","Doe, Jane","B"
```

```
"15 July","Bloggs, Fred","A"
```

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

to a document that looks like: (useHeaderRowForFieldNames=true)

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Jane
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

or to a document that looks like: (useHeaderRowForFieldNames=false)

▼  document	
▼  rows []	
▼  rows[0]	
 column1	Date
 column2	Pupil
 column3	Grade
▼  rows[1]	
 column1	25 May
 column2	Bloggs, Fred
 column3	C
▼  rows[2]	
 column1	25 May
 column2	Doe, Joe
 column3	B
▼  rows[3]	
 column1	15 July
 column2	Bloggs, Fred
 column3	A

Input Parameters

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

-
- useHeader* **String** Optional. Consider first line as header row and use the delimited data of this line as property names in the output document.
- RowForFieldNames* Set to:
- *true* . The delimited data of first line will be used as the property name in the output document. This is the default.
 - *false* . *column1, column2...columnN* will be used as the property name in the output document.
- encoding* **String** Optional. The encoding to use while parsing the delimited data.

Output Parameters

- document* **Document**. Document resulting from the conversion of *delimitedDataString* . This document contains document array rows[] corresponding to the delimited data.

documentToDelimitedDataBytes

Converts a document to delimited data bytes (byte array object).

This service will convert the following document:

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Jane
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

To bytes (byte array object) containing the following delimited data:
(useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

To the byte (byte array object) containing the following delimited data:
(useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

Input Parameters

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

Output Parameters

<i>delimitedDataBytes</i>	Object. Delimited data byte array object resulting from the conversion of a document.
---------------------------	--

documentToDelimitedDataStream

Converts a document to a delimited data stream.

This service will convert the following document:

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Jane
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

To the stream containing the following delimited data:
(useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

or to the stream containing the following delimited data:
(useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Note: Here the fieldQualifier = Comma(,) and textQualifier= double quote(")

Input Parameters

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

Output Parameters

<i>delimitedDataStream</i>	java.io.InputStream. Delimited data stream resulting from the conversion of a document.
----------------------------	--

documentToDelimitedDataString

Converts a document to a delimited data string.

This service will convert the following document:

▼  document	
▼  rows []	
▼  rows[0]	
 Date	25 May
 Grade	C
 Pupil	Bloggs, Fred
▼  rows[1]	
 Date	25 May
 Grade	B
 Pupil	Doe, Jane
▼  rows[2]	
 Date	15 July
 Grade	A
 Pupil	Bloggs, Fred

To the string containing the following delimited data:
(useHeaderRowForFieldNames=true)

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote("")

To the string containing the following delimited data:
(useHeaderRowForFieldNames=false)

"column1","column2","column3"

"Date","Pupil","Grade"

"25 May","Bloggs, Fred","C"

"25 May","Doe, Jane","B"

"15 July","Bloggs, Fred","A"

Here the fieldQualifier = Comma(,) and textQualifier= double quote("")

Input Parameters

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

Output Parameters

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

JSON

Use **JSON** services to convert JSON content into a document and to convert a document into JSON content.

Summary of JSON services

The following **JSON** services are available:

Service	Description
documentToJSONBytes	Converts a document to JSON bytes (byte array).
documentToJSONStream	Converts a document to a JSON stream.

Service	Description
documentToJSONString	Converts a document to a JSON string.
jsonBytesToDocument	Converts JSON content in bytes (byte array) to a document.
jsonStreamToDocument	Converts content from the JSON content stream to a document.
jsonStringToDocument	Converts content from the JSON string to a document.

documentToJSONBytes

Converts a document to JSON bytes (byte array).

Input Parameters

document **Document.** The document to be converted to JSON bytes (byte array).

Output Parameters

jsonBytes **Object.** JSON bytes (byte array) resulting from the conversion of a document.

documentToJSONStream

Converts a document to a JSON stream.

Input Parameters

document **Document.** The document to be converted to a JSON stream.

Output Parameters

jsonStream **java.io.InputStream.** JSON stream resulting from the conversion of a document.

documentToJSONString

Converts a document to a JSON string.

Input Parameters

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

Output Parameters

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

jsonBytesToDocument

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

Input Parameters

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

- *true* to convert integers to Long Java wrapper types
- *false* to convert integers to Integer Java wrapper types

Default value is *true*.

Output Parameters

document **Document.** Document resulting from the conversion of *jsonBytes*.

jsonStreamToDocument

Converts content from the JSON content stream to a document. The permissible size of the content stream is based on your tenancy.

Input Parameters

jsonStream **java.io.InputStream.** JSON content in an input stream to convert to a document.

decodeRealAsDouble **String.** Optional. Converts real numbers from *jsonStream* to either a Float or Double Java wrapper type. Set to:

- *true* to convert real numbers to Double Java wrapper types
- *false* to convert real numbers to Float Java wrapper types

Default value is *true*.

decodeIntegerAsLong **String.** Optional. Converts integers from *jsonStream* to either a Long or Integer Java wrapper type. Set to:

- *true* to convert integers to Long Java wrapper types
- *false* to convert integers to Integer Java wrapper types

Default value is *true*.

Output Parameters

document **Document.** Document resulting from the conversion of *jsonStream*.

jsonStringToDocument

Converts content from the JSON content string to a document.

Input Parameters

- jsonString* **String.** JSON content string to convert to a document.
- decodeRealAsDouble* **String.** Optional. Converts real numbers from *jsonString* to either a Float or Double Java wrapper type. Set to:
- *true* to convert real numbers to Double Java wrapper types
 - *false* to convert real numbers to Float Java wrapper types
- Default value is *true* .
- decodeIntegerAsLong* **String.** Optional. Converts integers from *jsonString* to either a Long or Integer Java wrapper type. Set to:
- *true* to convert integers to Long Java wrapper types
 - *false* to convert integers to Integer Java wrapper types
- Default value is *true* .

Output Parameters

- document* **Document.** Document resulting from the conversion of *jsonString* .

XML

Use **XML** services to convert a document to XML content and XML content to a document.

Summary of XML services

The following **XML** services are available:

Service	Description
documentToXMLBytes	Converts a document to XML content bytes, as a byte array object.

Service	Description
documentToXMLStream	Converts a document to XML stream, as a <code>java.io.InputStream</code> object.
documentToXMLString	Converts a document to XML content string.
xmlBytesToDocument	Converts XML content bytes (byte array) to a document.
xmlStreamToDocument	Converts an XML content stream to a document.
xmlStringToDocument	Converts an XML string to a document.

documentToXMLBytes

Converts a document to xml content bytes, as a byte array object. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements, and the key values are turned into the contents of those elements.

This service will convert the following document:

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

To XML document bytes (byte array object), whose content looks like:

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

```

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

```

Input Parameters

document **Document.** Document that is to be converted to XML. Note that if you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

nsDecls [] **Document.** Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:

The screenshot shows a configuration interface for namespace declarations. It features a tree view with a root node 'nsDecls []' and two child nodes: 'nsDecls [0]' and 'nsDecls [1]'. Each child node contains two input fields: 'prefix' and 'uri'. The 'prefix' field for 'nsDecls [0]' is 'GSX' and for 'nsDecls [1]' is 'TxMon'. The 'uri' field for 'nsDecls [0]' is 'http://www.gsx.com' and for 'nsDecls [1]' is 'http://www.acrtrak/txmonitor'. Each input field has a small 'STR' icon and a red asterisk, and a three-dot menu icon to its right.

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

```
xmlns:gsx="http://www.gsx.com"
```

```
xmlns:TxMon="http://www.acrtrak/txmonitor"
```

Alternatively, you can declare a namespace by including an `@xmlns` key in document. If you were not using the `@` character to designate attributes, use the correct attribute prefix in your code.

Parameters for `nsDecls []` are:

prefix: Key name.

uri: Key value.

addHeader **String.** Optional.

Flag specifying whether the header element `<?xml version="1.0"?>` is to be included in the resulting XML String.

Set to:

true to include the header. This is the default.

false to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

Output Parameters

xmlBytes **Object.** XML content bytes (byte array) produced from document.

Usage Notes

If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in the document as shown below:

 name Midwest Extreme Sports

If you want to generate an element that contains children, represent with a document in the document as shown below:

 address1

 @country	USA
 street1	10211 Brook Road
 city	closed
 state	OH
 postalCode	22130

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named `*body` that contains the element's value.

For example, if you want to produce the following element:

`<phoneNum cc=011>216-741-7566</phoneNum>`, you would include the following in document:

▼  phoneNum	
 @cc	011
 *body	216-741-7566

To include namespaces, ensure that you do the following:

Include the appropriate namespace prefix in the key names in document. For example, to produce an element called `acctNum` that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named `GSX:acctNum` in document.

Define the URIs for the prefixes that appear in document. You can do this through `nsDecls` or by including an `@xmlns` key in the element where you want the `xmlns` attribute to be inserted.

documentToXMLStream

Converts a document to xml stream, as a `java.io.InputStream` object. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements and the key values are turned into contents of those elements.

This service will convert the following document:

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

To an XML document stream, whose content looks like:

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

```

<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>

```

Input Parameters

document **Document.** Document that is to be converted to XML. Note that if you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

nsDecls [] **Document.** Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:

nsDecls []	
▼ nsDecls [0]	
prefix *	GSX ...
uri *	http://www.gsx.com ...
▼ nsDecls [1]	
prefix *	TxMon ...
uri *	http://www.acrtrak/txmonitor ...

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

```
xmlns:gsx="http://www.gsx.com"
```

```
xmlns:TxMon="http:www.acrtrak/txMonitor"
```

Alternatively, you can declare a namespace by including an `@xmlns` key in document. If you were not using the `@` character to designate attributes, use the correct attribute prefix in your code.

Parameters for `nsDecls []` are:

prefix: Key name.

uri: Key value.

addHeader **String.** Optional.

Flag specifying whether the header element `<?xml version="1.0"?>` is to be included in the resulting XML String.

Set to:

true to include the header. This is the default.

false to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

Output Parameters

xmlStream **java.io.InputStream.** XML content stream produced from document.

Usage Notes

If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in document as shown below:

```
 name Midwest Extreme Sports
```

If you want to generate an element that contains children, represent with a document in the document as shown below:

▼  address1	
 @country	USA
 street1	10211 Brook Road
 city	closed
 state	OH
 postalCode	22130

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named `*body` that contains the element's value.

For example, if you want to produce the following element:

```
<phoneNum cc=011>216-741-7566</phoneNum>
```

You would include the following in document:

▼  phoneNum	
 @cc	011
 *body	216-741-7566

To include namespaces, ensure that you do the following:

Include the appropriate namespace prefix in the key names in document. For example, to produce an element called `acctNum` that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named `GSX:acctNum` in document.

Define the URIs for the prefixes that appear in document. You can do this through `nsDecls` or by including an `@xmlns` key in the element where you want the `xmlns` attribute to be inserted.

documentToXMLString

Converts a document to xml content string. This service will recurse through a given document and build an XML representation from the elements within it. Key names are turned into XML elements, and the key values are turned into the contents of those elements.

This service will convert the following document:

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

To an XML document string, whose content looks like:

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

```

<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>

```

Input Parameters

document **Document.** Document that is to be converted to XML. If you want to produce a valid XML document (one with a single root node), document must contain only one top-level document that is, a single document. The name of that document will serve as the name of the XML document's root element. If you need to produce an XML fragment, for example, a loose collection of elements that are not encompassed within a single root element, then document can contain multiple top level elements.

nsDecls [] **Document.** Optional. Namespaces associated with any namespace prefixes that are used in the key names in document. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:

nsDecls []	
▼ nsDecls [0]	
prefix *	GSX ...
uri *	http://www.gsx.com ...
▼ nsDecls [1]	
prefix *	TxMon ...
uri *	http://www.acrtrak/txmonitor ...

For each prefix specified in nsDecls, this service generates an xmlns attribute and inserts it into the top-most element of the resulting XML String. For example, if nsDecls had the two keys shown above, this service would insert the following attribute into the root element of the XML String:

```
xmlns:gsx="http://www.gsx.com"
```

```
xmlns:TxMon="http:www.acrtrak/txMonitor"
```

Alternatively, you can declare a namespace by including an `@xmlns` key in document. If you were not using the `@` character to designate attributes, use the correct attribute prefix in your code.

Parameters for `nsDecls []` are:

prefix: Key name.

uri: Key value.

addHeader **String.** Optional.

Flag specifying whether the header element `<?xml version="1.0"?>` is to be included in the resulting XML String.

Set to:

true to include the header. This is the default.

false to omit the header. Omit the header to generate an XML fragment or to insert a custom header.

Output Parameters

xmlString **Object.** XML document string produced from document.

Usage Notes

If you are building a Document that will be converted to an XML String, keep the following points in mind:

If you want to generate a simple element that contains only a character value, represent it with a String element in document as shown below:

```
 name Midwest Extreme Sports
```

If you want to generate an element that contains children, represent with a document in the document as shown below:

▼  address1	
 @country	USA
 street1	10211 Brook Road
 city	closed
 state	OH
 postalCode	22130

If you want to generate a simple element that contains a character value and one or more attributes, you must represent it as a document that has one key for each attribute and a key named `*body` that contains the element's value.

For example, if you want to produce the following element:

```
<phoneNum cc=011>216-741-7566</phoneNum>
```

You would include the following in document:

▼  phoneNum	
 @cc	011
 *body	216-741-7566

To include namespaces, ensure that you do the following:

Include the appropriate namespace prefix in the key names in document. For example, to produce an element called `acctNum` that belongs to a namespace that is represented by the "GSX" prefix, you would include a key named `GSX:acctNum` in document.

Define the URIs for the prefixes that appear in document. You can do this through `nsDecls` or by including an `@xmlns` key in the element where you want the `xmlns` attribute to be inserted.

xmlBytesToDocument

Converts XML content bytes (byte array) to a document. This service transforms each element and attribute in XML content bytes to an element in a Document.

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

```
<?xml version="1.0" ?>
<tns:AcctInfo>
xmlns:tns="http://localhost/DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
<name>Midwest Extreme Sports</name>
<rep>Laura M. Sanchez</rep>
<acctNum type=platinum>G97041A</acctNum>
<phoneNum cc=011>216-741-7566</phoneNum>
<address country=USA>
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
</address>
<address country=USA xsi:type="tns:DerivedAddress">
<street1>10211 Brook Road</street1>
<city>Cleveland</city>
<state>OH</state>
<postalCode>22130</postalCode>
<landMark>Besides Ohio River-Bank Square</landMark>
<telNo>001222555</telNo>
</address>
<serialNum>19970523A</serialNum>
<serialNum>20001106G</serialNum>
<serialNum>20010404K</serialNum>
</tns:AcctInfo>
```

To a Document that looks like:

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

Input Parameters

xmlBytes **Object.** XML content bytes that is to be converted to a document.

nsDecls [] **Document.** Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in nsDecls . This is important because incoming XML documents can use any prefix for a given

namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in `nsDecls` also define the prefixes used by the arrays, documents, `documentTypeName`, and `collect` parameters. Each entry in `nsDecls` represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called `GSX` and `TxMon`, you would set `nsDecls` as follows:

nsDecls []	
nsDecls [0]	
prefix *	GSX ...
uri *	http://www.gsx.com ...
nsDecls [1]	
prefix *	TxMon ...
uri *	http://www.acrtrak/txmonitor ...

Parameters for `nsDecls []` are:

prefix: Key name.

uri: Key value.

`preserveUndeclaredNS`

String Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the `nsDecls` input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the `nsDecls` parameter, Integration Cloud adds the `xmlns` attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- `False` to ignore namespace declarations in the XML document that are not specified in the `nsDecls` parameter. This is the default.

`preserveNSPositions`

String Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the `nsDecls` parameter in the resulting document.

Set to:

- `True` to preserve the position of namespaces declared in `nsDecls` in the resulting document. For each namespace specified in the `nsDecls` parameter, Integration Cloud adds the `xmlns` attribute to the document as a String variable named "`@xmlns:NSprefix`" where "`NSprefix`" is the prefix name specified in `nsDecls`. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the `xmlns` attribute declaration within the XML document.
- `False` to not maintain the position of the namespace declarations specified in `nsDecls` in the resulting document. This is the default.

Output Parameters

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

Usage Notes

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

XML Document	Document
<pre><myDoc><e1>e1Value</e1> </myDoc></pre>	<pre> document └── myDoc └── e1 e1Value </pre>
<pre><?xml version="1.0" encoding="UTF-8" standalone="no"?><myDoc><e1>e1Value </e1></myDoc></pre>	<pre> document ├── @encoding UTF-8 ├── @standalone no ├── @version 1.0 └── myDoc └── e1 e1Value </pre>

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1
e1Attr="attrValue">
e1Value</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	
*body	e1Value
@e1Attr	attrValue

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>e1Value
</e1><e2>e2Value</e2></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	e1Value
e2	e2Value

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>e1Value1
</e1><e2>e2Value</e2><e1>e1Value2
</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	e1Value1
e2	e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">e1Value1
</e1><e2>e2Value</e2><e1 e1Attr=
"attrValue2">
e1Value2</e1></myDoc>
```

document	
@encoding	UTF-8
@version	1.0
myDoc	
e1 []	
e1[0]	
*body	e1Value1
@e1Attr	attrValue1
e1[1]	
*body	e1Value2
@e1Attr	attrValue2
e2	e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1
</e1><e2>e2Value</e2><e1
e1Attr="attrValue2">
e1Value2</e1></myDoc>
```

document	
@encoding	UTF-8
@version	1.0
myDoc	
e1 []	
e1[0]	
*body	e1Value1
@e1Attr	attrValue1
e1[1]	
*body	e1Value2
@e1Attr	attrValue2
e2	e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">e1Value1
</e1><e2>e2Value</e2><e1 e1Attr=
"attrValue2">
e1Value2</e1></myDoc>
```

document	
@encoding	UTF-8
@version	1.0
myDoc	
e1	
*body	e1Value2
@e1Attr	attrValue2
e2	e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">e1Value1
</e1><e2><e3>e3Value</e3>
<e4 e4Attr="attrValue4"e4Attrb=
"attrValue4b">
e4Value</e4></e2></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1		
*body	e1Value2	
@e1Attr	attrValue2	
e2		
e3	e3Value	
e4		
*body	e4Value	
@e4Attr	attrValue4	
@e4Attrb	attrValue4b	

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><tns:AcctInfo>
xmlns:tns="http://localhost/
DerivedAddress/schema.xsd"
xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance">
<myDoc>
<e1>e1Value</e1></myDoc>
<myDoc xsi:type="tns:DerivedDoc">
<e1>e1Value</e1><e2>
e2Value</e2></myDoc>
</tns:AcctInfo>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc []		
myDoc[0]		
e1	e1Value	
myDoc[1]		
*docType	DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc	
e1	e1Value	
e2	e2Value	

xmlStreamToDocument

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

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

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

```

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

```

To a Document that looks like:

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

Input Parameters

- xmlStream* **java.io.InputStream**. XML content stream that is to be converted to a document.
- nsDecls []* **Document**. Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in *nsDecls* . This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in *nsDecls* also define the prefixes used by the arrays, documents, documentTypeName, and collect parameters. Each entry in *nsDecls* represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set *nsDecls* as follows:

▼ nsDecls []

- ▼ nsDecls [0]

prefix *	GSX	...
uri *	http://www.gsx.com	...
- ▼ nsDecls [1]

prefix *	TxMon	...
uri *	http://www.acrtrak/txmonitor	...

Parameters for *nsDecls []* are:

prefix: Key name.

uri: Key value.

- preserveUndeclaredNS* **String** Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- `True` to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the `nsDecls` parameter, Integration Cloud adds the `xmlns` attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- `False` to ignore namespace declarations in the XML document that are not specified in the `nsDecls` parameter. This is the default.

`preserveNSPositions` **String** Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the `nsDecls` parameter in the resulting document.

Set to:

- `True` to preserve the position of namespaces declared in `nsDecls` in the resulting document. For each namespace specified in the `nsDecls` parameter, Integration Cloud adds the `xmlns` attribute to the document as a String variable named "@xmlns:NSprefix" where "NSprefix" is the prefix name specified in `nsDecls`. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the `xmlns` attribute declaration within the XML document.
- `False` to not maintain the position of the namespace declarations specified in `nsDecls` in the resulting document. This is the default.

Output Parameters

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

Usage Notes

Following are examples of XML documents and the documents that `xmlStreamToDocument` will produce:

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

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

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	e1Value

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc>
<e1 e1Attr="attrValue">
e1Value</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	
body	e1Value
@e1Attr	attrValue

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc>
<e1>e1Value</e1><e2>e2Value</e2>
</myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	e1Value
e2	e2Value

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>
e1Value1</e1><e2>
e2Value</e2><e1>
e1Value2</e1></myDoc>
```

document	
@encoding	UTF-8
@standalone	no
@version	1.0
myDoc	
e1	e1Value1
e2	e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">e1Value2
</e1></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1 []		
e1[0]		
*body		e1Value1
@e1Attr		attrValue1
e1[1]		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>
e2Value</e2><e1 e1Attr="attrValue2">
e1Value2</e1></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1 []		
e1[0]		
*body		e1Value1
@e1Attr		attrValue1
e1[1]		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">
e1Value2</e1>
</myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0" encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1
</e1><e2><e3>e3Value</e3>
<e4 e4Attr=
"attrValue4" e4Attrb="attrValue4b">
e4Value
</e4></e2></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1		
*body		e1Value2
@e1Attr		attrValue2
e2		
e3		e3Value
e4		
*body		e4Value
@e4Attr		attrValue4
@e4Attrb		attrValue4b

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><tns:AcctInfo>
xmlns:tns="http://localhost/
DerivedAddress/
schema.xsd" xmlns:xsi="http://www
.w3.org/
2001/XMLSchema-instance"><myDoc>
<e1>e1Value</e1></myDoc>
<myDoc xsi:type="tns:DerivedDoc">
<e1>e1Value</e1><e2>e2Value</e2>
</myDoc>
</tns:AcctInfo>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc []		
myDoc[0]		
e1		e1Value
myDoc[1]		
*docType		DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc
e1		e1Value
e2		e2Value

xmlStringToDocument

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

This service will convert the following XML string:

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

```

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

```

To a Document that looks like:

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

Input Parameters

xmlString

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

nsDecls []

Document. Optional. Namespace prefixes to use for the conversion. This parameter specifies the prefixes that will be used when namespace-qualified elements are converted to key names in the resulting document object. For example, if you want elements belonging to a particular namespace to have the prefix GSX in the resulting document, for example, GSX:acctNum, you would associate the prefix GSX with that namespace in nsDecls . This is important because incoming XML documents can use any prefix for a given namespace, but the key names expected by a target service will have a fixed prefix. Namespace prefixes in nsDecls also define the prefixes used by the arrays, documents, documentTypeName, and collect parameters. Each entry in nsDecls represents a namespace prefix/URI pair, where a key name represents a prefix and the value of the key specifies the namespace URI. For example, to define the URIs associated with two prefixes called GSX and TxMon, you would set nsDecls as follows:

nsDecls []	
nsDecls [0]	
prefix *	GSX
uri *	http://www.gsx.com
nsDecls [1]	
prefix *	TxMon
uri *	http://www.acrtrak/tx

Parameters for *nsDecls []* are:

prefix: Key name.

uri: Key value.

preserveUndeclaredNS

String Optional. Flag indicating whether or not Integration Cloud keeps undeclared namespaces in the resulting document. An undeclared namespace is one that is not specified as part of the *nsDecls* input parameter.

Set to:

- **True** to preserve undeclared namespaces in the resulting document. For each namespace declaration in the XML document that is not specified in the *nsDecls* parameter, Integration Cloud adds the *xmlns* attribute as a String variable to the document. Integration Cloud gives the variable a name that begins with "@xmlns" and assigns the variable the namespace value specified in the XML document. Integration Cloud preserves the position of the undeclared namespace in the resulting document.
- **False** to ignore namespace declarations in the XML document that are not specified in the *nsDecls* parameter. This is the default.

preserveNSPositions

String Optional. Flag indicating whether or not Integration Cloud maintains the position of namespaces declared in the *nsDecls* parameter in the resulting document.

Set to:

- **True** to preserve the position of namespaces declared in *nsDecls* in the resulting document. For each namespace specified in the *nsDecls* parameter, Integration Cloud adds the *xmlns* attribute to the document as a String variable named "@xmlns:NSprefix" where "NSprefix" is the prefix name specified in *nsDecls*. Integration Cloud assigns the variable the namespace value specified in the XML document. This variable maintains the position of the *xmlns* attribute declaration within the XML document.
- **False** to not maintain the position of the namespace declarations specified in *nsDecls* in the resulting document. This is the default.

arrays []

String List Optional. Names of elements that are to be generated as arrays, regardless of whether

they appear multiple times. For example, if *arrays* contained the following values for the XML document shown in the example in the description for this service:

```
rep
address
```

`xmlStringToDocument` would generate element `rep` as a String List and element `address` as a Document List.

Important If you include namespace prefixes in the element names that you specify in *arrays*, you must define the namespaces associated with those prefixes in *nsDecls*.

makeArrays

String Optional. Flag indicating whether you want `xmlStringToDocument` to automatically create an array for every element that appears more than once. Set to:

- `true` to automatically create arrays for every element that appears more than once. This is the default.
- `false` to create arrays for only those elements specified in *arrays*.

Output Parameters

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

Usage Notes

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

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

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

document		
@encoding	UTF-8	
@standalone	no	
@version	1.0	
myDoc		
e1	e1Value	

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc>
<e1 e1Attr="attrValue">
e1Value</e1></myDoc>
```

document		
@encoding	UTF-8	
@standalone	no	
@version	1.0	
myDoc		
e1		
body	e1Value	
@e1Attr	attrValue	

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>
e1Value</e1><e2>e2Value</e2>
</myDoc>
```

document		
@encoding	UTF-8	
@standalone	no	
@version	1.0	
myDoc		
e1	e1Value	
e2	e2Value	

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><myDoc><e1>e1Value1
</e1><e2>e2Value</e2><e1>
e1Value2</e1></myDoc>
```

document		
@encoding	UTF-8	
@standalone	no	
@version	1.0	
myDoc		
e1	e1Value1	
e2	e2Value	

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">
e1Value2</e1></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1 []		
e1[0]		
*body		e1Value1
@e1Attr		attrValue1
e1[1]		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">
e1Value2</e1></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1 []		
e1[0]		
*body		e1Value1
@e1Attr		attrValue1
e1[1]		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2>e2Value</e2>
<e1 e1Attr="attrValue2">
e1Value2</e1></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1		
*body		e1Value2
@e1Attr		attrValue2
e2		e2Value

```
<?xml version="1.0"encoding="UTF-8"?>
<myDoc><e1 e1Attr="attrValue1">
e1Value1</e1><e2><e3>e3Value
</e3><e4 e4Attr="attrValue4"
e4Attrb=
"attrValue4b">e4Value</e4>
</e2></myDoc>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc		
e1		
*body		e1Value2
@e1Attr		attrValue2
e2		
e3		e3Value
e4		
*body		e4Value
@e4Attr		attrValue4
@e4Attrb		attrValue4b

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><tns:AcctInfo>
xmlns:tns="http://localhost/
DerivedAddress/
schema.xsd"xmlns:xsi="http:
//www.w3.org/2001/
XMLSchema-instance"><myDoc>
<e1>e1Value</e1></myDoc>
<myDoc xsi:type="tns:DerivedDoc">
<e1>e1Value</e1><e2>e2Value
</e2></myDoc>
</tns:AcctInfo>
```

document		
@encoding	UTF-8	
@version	1.0	
myDoc []		
myDoc[0]		
e1		e1Value
myDoc[1]		
*docType		DerivedDoc.documentLocation:docTypeRef_tns_DerivedDoc
e1		e1Value
e2		e2Value

IO

Use **IO** services to convert data between `byte[]`, characters, and `InputStream` representations. These services are used for reading and writing bytes, characters, and streamed data to the file system. These services behave like the corresponding methods in the `java.io.InputStream` class. For more information about `InputStreams`, see the Java documentation.

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.

Summary of IO services

The following **IO** services are available:

Service	Description
bytesToStream	Converts a byte[] to java.io.ByteArrayInputStream.
streamToBytes	Creates a byte[] from data that is read from an InputStream.
streamToString	Creates a string from data that is read from an InputStream.
stringToStream	Converts a string to a binary stream.

bytesToStream

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

Input Parameters

<i>bytes</i>	byte[] The byte array to convert.
<i>length</i>	String Optional. The maximum number of bytes to read and convert. If <i>length</i> is not specified, the default value for this parameter is the length of the input byte array.
<i>offset</i>	String Optional. The offset into the input byte array from which to start converting. If no value specified, the default value is zero.

Output Parameters

<i>stream</i>	java.io.ByteArrayInputStream An open InputStream 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

stream **java.io.InputStream** The `InputStream` that you want to convert.

Output Parameters

bytes **byte[]** The bytes read from *stream* .

Usage Notes

This service reads all of the bytes from *stream* until the end of file is reached, and then it closes the `InputStream`.

streamToString

Creates a string from data that is read from an `InputStream`.

Input Parameters

inputStream **java.io.InputStream** The `InputStream` to convert to a string.

encoding **String** Optional. Name of a registered, IANA character set (for example, ISO-8859-1). If you specify an unsupported encoding, the system throws an exception. If no value is specified the encoding will be UTF-8.

Output Parameters

string **String** Data read from *inputStream* and converted to a string.

stringToStream

Converts a string to a binary stream.

Input Parameters

<i>string</i>	String The string object to be converted.
<i>encoding</i>	String Optional. Name of a registered, IANA character set, for example, ISO-8859-1. If you specify an unsupported encoding, the system throws an exception. If no value is specified, the encoding will be UTF-8.

Output Parameters

<i>inputStream</i>	java.io.ByteArrayInputStream An open InputStream created from the contents of <i>string</i> .
--------------------	--

Utils

Contains utility services.

Summary of Utils services

The following **Utils** services are available:

Service	Description
generateUUID	Generates a random Universally Unique Identifier (UUID).

generateUUID

Generates a random Universally Unique Identifier (UUID).

Input Parameters

None.

Output Parameters

<i>UUID</i>	String A randomly generated Universally Unique Identifier (UUID).
-------------	--

Integration Details

This page allows you to view 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 edit, delete, or run the Integration, enable the Integration to be invoked over HTTP and view and add OAuth Scopes containing the exposed Integration URL, enable Integration executions to be restartable or resumable, associate another Access Control List (ACL) with the Integration, and also view the last five execution results.

Option	Description
Overview	This page allows you to view 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 edit, delete, or run the Integration, enable the Integration to be invoked over HTTP and view and add OAuth Scopes containing the exposed Integration URL, enable Integration executions to be restartable or resumable, and associate another Access Control List (ACL) with the Integration.
Last 5 Execution Results	Click this tab to view the last five execution results panel. This screen allows you to view the audit trail of the executions that happened in the current stage. See Execution Results for information on the Last 5 Execution Results table columns.
Uses	<p>Displays the components used to create the Integration. This field will appear only if the selected Integration has components.</p> <p>Note: If assets used by an Integration are deleted, you will not be able to pull the Integration into subsequent stages or export the Integration.</p>
Created on	Displays the date and time when the Integration was created.
Created by	Displays the user who created the Integration.
Edit	Click this option to modify the Integration.

Option	Description
Delete	Click this option to delete the Integration from the active stage. You can delete an Integration from all stages except from the <i>Development</i> stage.
Run Now	Click this option to submit the Integration for execution. You can provide inputs to the Integration based on the defined input signature.
Last modified/Last modified by	When and by whom was the Integration last modified.
Scheduled status	If the Integration in the active stage is scheduled, the status of the Integration displays Scheduled , else it appears as Not Scheduled . The Status appears as Paused if the Integration has been paused.
Last execution	When was the Integration last executed. A warning message appears if the last execution was not successful.
Next scheduled execution	When is the Integration scheduled to run again.
Schedule	<p>Click this option to define a schedule. Select Run Once if you want to schedule the Integration to run just once immediately or run once at a specified date and time.</p> <p>Select Run Recurrently if you want to define a recurrence pattern. You can define a recurrence pattern daily, weekly, monthly, and in hours. Select the frequency (Hourly, Daily, Weekly, Monthly) with which the pattern recurs, and then select the options for the frequency. Click the + icon to repeat the execution for daily, weekly, and monthly schedules.</p> <p>Click the  icon to delete the selected execution time for daily, weekly, and monthly schedules. Select Prevent concurrent executions to skip the next scheduled execution if the previous scheduled execution is still running except when the previous scheduled execution is running for more than 3 hours. In this case, the next scheduled execution will start even if the previous scheduled execution is still running. If you do not select this option, the next</p>

Option	Description
	<p>scheduled execution will start, even if the previous scheduled execution has not yet completed.</p> <p>Click Delete if you want to permanently remove the current recurrence schedule.</p> <p>Click Next to provide inputs to the Integration based on the defined input signature.</p>
Modify Schedule	Click this option to change the existing schedule.
Pause	Click this option to pause the Integration that was scheduled.
Resume	Click this option to start the scheduled Integration that was paused.
Associated Execute Access Control List	<p>Displays the Access Control List (ACL) associated with the Integration. Integration Cloud associates the default ACL, <i>Default</i>, to an Integration when the Integration is created. Click Permissions to associate the Integration with another ACL.</p> <p>Note: The ACL will be enforced only if an Integration acts as a top-level Integration. For example, if Integration A has Integration B and Integration C as sub-Integrations, then the ACL if associated, will be enforced only on Integration A.</p>
Enable executions to be restartable	<p>For an orchestrated Integration, select the Enable executions to be restartable option if you want to enable Integration executions to be restartable or resumable. If an operation fails, you can resume the Integration execution from the point where it had failed from the Execution Results page (Resume option). Resuming an execution does not execute the previous successful operations but executes only the failed operations and operations that are not yet executed.</p> <p>You can also restart an execution from the Execution Results page (Restart option). When an Integration is restarted, the execution occurs from the beginning of an Integration.</p> <p>Note: The "Restart/Resume" capability is available only if you have the required license for restarting</p>

Option	Description
	<p>and resuming Integrations. You must also have the Integration execution (Execute) permission if you want to restart or resume an execution.</p> <p>Note: You cannot resume successful Integrations but can only restart them. If an Integration has referenced Integrations, then those referenced Integrations can also be restarted or resumed. Only top level Integrations are displayed in the Execution Results page and the referenced Integrations are displayed in the Execution Details page of that top level Integration execution. The Audit Log will display the "Restart" and "Resume" entries.</p> <p>Note: User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.</p> <p>See Execution Results for more information.</p> <p>Note: Enabling this option will increase the execution time of this Integration.</p> <p>Once the Integration is updated, you cannot restart or resume its executions that have occurred before the update.</p> <p>Integrations using Operations and other Integrations, which have fields of type "Object" in their signature, may not execute properly when restarted or resumed.</p>
<p>Enable Integration to be invoked over HTTP</p>	<p>Select the Enable Integration to be invoked over HTTP option if you want to trigger the execution of an Integration from an external system. This option provides you with one more way to trigger Integration executions from a software application, for example, a REST client, apart from manual and scheduled integrations from the user interface.</p> <p>Once the Integration is enabled to be invoked over HTTP, the HTTP request URL appears. Click the Show Advanced Options link to view the HTTP Method, sample JSON input, and the parameters that are required to invoke this Integration from an external system.</p>

Option	Description
OAuth Scopes containing the exposed Integration URL	<p>You need to provide the HTTP URL, Method, required JSON body, and necessary header parameters in the external program, including the required security credentials (user name and password) for invoking the Integration. After the Integration is executed, the response will contain the pipeline data.</p> <p><i>Request URL format</i></p> <pre>https://<sub-domain>.webmethodscloud.com/integration/rest/external/integration/run/<stagename>/<integrationname></pre> <p><i>sub-domain</i> is a domain that is part of the primary domain.</p> <p><i>stagename</i> is the name of the active stage.</p> <p><i>integrationname</i> is the name of the Integration.</p> <p><i>HTTP Status Codes</i></p> <ul style="list-style-type: none"> ■ 200 - OK ■ 500 - Internal Server Error ■ 401 - Unauthorized User Error <p>Note: You must provide your user name and password to execute the Integration from the external program, else you may encounter the 401 - Unauthorized User Error.</p> <p>A “scope” on page 81 defines the services the client can access on behalf of the resource owner and consists of one or more services. If access is granted for a scope, then access is granted for all the services in that scope. When a request is made, Integration Cloud verifies that the scope is defined for a client. The client is allowed to access only the service URLs that are specified for the scope.</p> <p>The OAuth Scopes containing the exposed Integration URL option appears when you select the Enable Integration to be invoked over HTTP check box. Click OAuth Scopes to view the OAuth scopes that contain the exposed URL of the selected Integration.</p>

Option	Description
	In the OAuth Scopes dialog box, click Add URL to Another Scope . In the Add Exposed URL to OAuth Scope dialog box, select Add To Existing Scope to add the exposed Integration URL to an existing scope or select Add New Scope to create a new scope and add the exposed Integration URL to that new scope.

Exporting Integrations

Integration Cloud allows you to export Integrations from the **Integrations** page. The export capability is available only if you have the required license for exporting Integrations. You can export Integrations from one tenant and import those Integrations to another tenant. Ensure that you have the **Export Integration** permission to export Integrations.

To export Integrations

1. From the Integration Cloud navigation bar, click **Develop > Integrations**. The **Integrations** page appears.
2. Select the Integrations from the **Integrations** page and click **Export**. If the Integrations you are exporting use Reference Data, Document Types, SOAP, or REST Applications, then those Applications including the Reference Data and Document Types will also be exported along with the Integrations.

The **Confirm Export** dialog box appears.

Confirm Export ✕

The table lists the Integrations selected for export.

IMPORTANT! If exported Integrations contain on-premises Applications, same on-premises Applications must be uploaded before the import.

Name
ConcurAttendeeToSalesforceContacts12 add_integration_service

Cancel Export

3. Click **Export** to export the Integrations. The Integrations will be downloaded as a zip file to your default download folder. The zip file size must not be greater than 50 MB.

Note: Do not modify the contents of the exported zip file. If you modify the contents of the zip file, the Integrations cannot be imported back to Integration Cloud.

Exporting Integrations having on-premises Applications

After exporting an Integration that has an on-premises Application, if you want to import the Integration, then before importing the Integration, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the Integration.

Importing Integrations

Integration Cloud allows you to import Integrations from the **Integrations** page. You can import Integrations from a zip file that was earlier exported from Integration Cloud. You can export Integrations from one tenant and import those Integrations to another tenant. You can import Integrations provided you have the **Create** Integration permission.

Note: If you want to import an Integration that has an on-premises Application, before importing the Integration, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the Integration.

To import Integrations

1. From the Integration Cloud navigation bar, click **Develop > Integrations**. The **Integrations** page appears.
2. Click **Import Integrations**.
The **Import Integrations** page appears.
3. Click **Browse** and select the zip file that contains the exported Integrations. The zip file size must not be greater than 50 MB. The Integrations available in the zip file will appear in the pane.
4. Click **Preview** to view an Integration or click **Import** to import an Integration.
5. In the **Connect to Applications** screen, select the **Account** for each Application or create a new Account, and then click **Next**.

Note: If the Integrations you are importing use SOAP or REST Applications and if those Applications do not exist in your system, you will not be able to create Accounts at this step. Continue importing the Integrations. The Applications will also be imported along with the Integrations. After importing, create the Accounts and then configure them in the imported Integrations.

Note: If an Integration you are importing uses an on-premises Application and if the Application does not exist in your system, you will not be able to select the Account at this step. The Account will appear only after you have uploaded the on-premises Application. See the *Configuring On-Premise Integration Servers for webMethods Cloud* document for information on how to upload on-premises Applications.

The **Overview and Save Integration** screen appears.

6. In the **Overview and Save Integration** screen, provide a name and description for your Integration. By default, the Integration name and description appears.
7. Click **Finish**. If you have existing references (Reference Data, Document Types, and custom Operations) with the same name in the development stage, the **Copy References** screen appears. Click **Cancel** to go back to the **Overview and Save Integration** screen. By default, all references are selected in the **Copy References** screen.
8. Deselect the references that you do not want to replace and then click **Continue** to replace or overwrite all the selected references from the Integration in the development stage.

The Integration details screen appears for the newly created Integration.

REST APIs

Integration Cloud allows you to write integration logic to integrate different types of applications. This logic can be exposed to the external world using REST APIs.

These REST APIs can be created by using an existing set of Integrations (from scratch) or by using a file containing the Open API specification (formerly known as the Swagger specification) as a template.

A REST API consists of many Resource Operations and each Resource Operation has a Path, one or more HTTP Methods, and an associated Integration.

A REST Resource Operation can be tried out from the **Swagger** screen of a REST API. When the Resource Operation is invoked using the HTTP Method, the associated Integration gets executed.

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

Note: If you have created a REST API by using a file containing the Open API specification (formerly known as the Swagger specification) as a template (**Build with Swagger** approach), and have now uploaded a new file, Integrations, Services and Document Types that are created are now based on the new Swagger file.

Creating REST APIs from scratch

To create a REST API from scratch

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

- From the **REST APIs** page, click **Add New REST API**, select **Build from scratch**, and then click **OK**.
- Complete the following fields and click **Save**.

The new REST API appears in the **REST APIs** page.

<u>In this field...</u>	<u>Specify...</u>
Save As	Type a name for the REST resource using any combination of letters, numbers, and/or underscore characters.
Swagger Title	Provide a title for the REST API.
Description	Type an optional description for the REST API.
Version	A version number. The value is typically 1.0. You need to change the version number if you want to modify and republish the REST API.
Consumes	Select the MIME types that the API consumes.
Produces	Select the MIME types that the API produces. The MIME types you select here will be part of the Swagger definition and will appear in the Responses section of the Swagger editor. While executing the REST API, you can choose the format in which you want to view the response. You must specify an <i>Accept</i> header in the REST client, else the default response will conform to the Content Type <i>text/html</i> .

- Click **Save**.

The REST API is created and appears in the **REST APIs** page. When you create a REST API, Integration Cloud uses the general information that you supplied to populate the **Overview** page in the REST API. You can click **Edit** to change the information in the **Overview** page. The name cannot be modified.

- In the **Resources** page, click **Add New Resource** to add a Resource to the REST API. Type a **Path**, select a **Mapped Integration** that you want to invoke with this path, and select the **HTTP Methods** that can be used to invoke this Integration with this path. You can also provide a description for each method in the **Operation Summary** field. You can add more Resource Operations, if needed.

6. Click **Save and Continue** to add parameters and responses and then click **Save and Finish**. See “[Modifying Resource Operations](#)” on page 414 for information on how to modify a REST Resource Operation.
7. To run an Integration with a certain path and method, go to the **REST APIs** page, select a REST API, and click the REST API link. The REST API details page appears.
8. Go to the **Swagger** page from the REST API details page. Click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs. Select the path and method pair and click **Try it out**. If required, pass the parameters and click **Execute**.

Responses will be displayed on the pane.

Modifying Resource Operations

To modify a REST Resource Operation

1. From the Integration Cloud navigation bar, click **Develop > REST APIs**.
The **REST APIs** page appears.
2. Click on the link of a REST API that has been *created from scratch*, go to the **Resources** page, select a resource operation, and then click **Edit**.

1 Define RES

Operation Description

Path *

Mapped Integration *

HTTP Methods

Select one or more HTTP methods to complete the REST Operation definition

Method	Operation Summary
<input checked="" type="checkbox"/> POST	<input type="text"/>
<input checked="" type="checkbox"/> PUT	<input type="text"/>
<input checked="" type="checkbox"/> GET	<input type="text"/>
<input checked="" type="checkbox"/> DELETE	<input type="text"/>
<input checked="" type="checkbox"/> PATCH	<input type="text"/>

Modifying the Path

On the **Resources** page, select a resource operation and then click **Edit**. Change the path. If you modify the path, the Integration will run on the new path. To run the Integration, for a given method, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the new path and method pair.

Modifying the HTTP Methods

On the **Resources** page, select a resource operation and then click **Edit**. Select new HTTP Methods with respect to which you want to expose the given Integration.

To run the integration now, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the path and one of the new methods that was selected.

Changing the mapped Integration

On the **Resources** page, select a resource operation and then click **Edit**. Choose a new **Mapped Integration** from the drop-down list. To run the new Integration, go to the **Swagger** page, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and click **Try it out** that corresponds to the Integration method and path pair.

Modifying an existing Integration

Select an Integration in the **Integrations (Develop > Integrations)** page that is exposed in a REST API and click **Edit**. Change the input signature and output signature of the Integration. Now go to the **REST APIs (Develop > REST APIs)** page, select a REST API, and click on the REST API link. Go to the **Resources** page. Identify any method and path pair that corresponds to the Integration whose signature was modified. Then go to the **Swagger** page and navigate to the method and path pair that was identified. You can see that the parameters have changed and they now correspond to the new Integration input signature. Note that the responses have also changed and they correspond to the output signature of the Integration. Click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and then click **Try it out**, supply the new parameter values, and run the Integration.

Changing Parameter Types

On the **Resources** page, select a resource operation and then click **Edit**. Click **Save and Continue**. Parameters corresponding to the Integration appears.

Parameters		Responses	
Name	b	HTTP Code	200
Source	FORMDATA	HTTP Code	401
Description	Describe the Parameter		

- If the parameter type is **body**, you cannot change its type.
- If the parameter type is **formdata**, you can change it to a header or a query parameter.
- If the parameter type is **header**, you can change it to a query or formdata parameter.
- If the parameter type is **query**, you can change it to a form or a header parameter.

Adding Responses

On the **Resources** page, select a resource operation and then click **Edit**. Click **Continue**. You will see the responses corresponding to the output signature of the Integration in the panel. You can add more response codes, if needed.

Note: Click **Back to Definition** to go back to the REST Resource Operation definition page.

3. A “[scope](#)” on page 81 defines the services the client can access on behalf of the resource owner and consists of one or more services. If access is granted for a scope, then access is granted for all the services in that scope. When a request is made, Integration Cloud verifies that the scope is defined for a client. The client is allowed to access only the service URLs that are specified for the scope.

For a Resource Path with Method, click **OAuth Scopes** to view the OAuth Scopes that contain the REST Resource path with Method.

In the **OAuth Scopes** dialog box, click **Add URL to Another Scope**. In the **Add REST Resource Path to OAuth Scope** dialog box, select **Add To Existing Scope** to add the REST Resource path with Method to an existing scope or select **Add New Scope** to create a new scope and add the REST Resource path with Method to that new scope.

Path	Description	Integration	Permissions	OAuth Scopes
▼ /pet				
POST	Add a new pet to the store	wdPet	Permissions	OAuth Scopes
PUT	Update an existing pet	updatePet	Permissions	OAuth Scopes

Creating REST APIs with Swagger

To create a REST API using a Swagger file

1. From the Integration Cloud navigation bar, click **Develop > REST APIs**.

The **REST APIs** page appears.

2. From the **REST APIs** page, click **Add New REST API**, select **Build with Swagger**, and then click **OK**.

In the **New REST API** page, complete the following fields. Required fields are marked with an asterisk on the screen.

Select...

Save As

Swagger Source

To...

Type a name for the REST API using any combination of letters, numbers, and/or the underscore character.

Select **URL** and enter the URL for the Swagger document. The URL should begin with `https://`.

OR

Select...	To...
	Select File and click Browse to select a Swagger document on your local file system. Swagger documents must be in either JSON format with a .json file extension or YAML format with a .yaml or .yml file extension.

Click **Save**.

Note the following points after you click **Save**:

- The REST API is created based on the Swagger definition that is uploaded or the Swagger definition that is referred to by the given URL.
 - Integrations are generated based on the Operation IDs in the Swagger file. One Integration is created for each Operation ID in the Swagger file. A Swagger file cannot have two Operation IDs that are similar.
 - Resource Operations are created for a unique path in the Swagger file.
3. To view the Swagger definition, from the **REST APIs** page, click the REST API link. The **Overview** page provides general information on the REST API. The **Resources** page allows you to view the resource operations. The **Swagger** page allows you to view the Swagger definition.

You can use Access Control Lists (ACLs) to control the execution permission of a REST API. Integration Cloud associates the default ACL, *Default*, to a REST API when the REST API is created using a Swagger file. Click **Permissions** to associate the REST API with another ACL.

Note: The ACL will be enforced only if an Integration acts as a top-level Integration. For example, if Integration A has Integration B and Integration C as sub-Integrations, then the ACL if associated, will be enforced only on Integration A.

4. To edit the Integration, go to the **Resource Operations** page, select a Resource Operation, expand its path, and click on the icon next to the Integration. The Integration will be open for editing in the Edit Integrations page. Add necessary blocks and perform mappings.

Note: You will not be able to modify the Input/Output signature of the Integration because the signature is derived from the Swagger definition.

5. To run the Integration with the given path and method, go to the **Swagger** page for the REST API, click **Authorize** and type your Integration Cloud user name and password to authorize access to your APIs, and then click **Try it out**.

Copying REST APIs

Integration Cloud allows you to create a copy of a REST API from the **REST APIs** page. You can copy a REST API if you have the required permissions.

To copy a REST API

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

2. Select a REST API and click **Copy**.

The **Copy** dialog box appears.

3. Type a new name in the **Copy As** field and click **Copy**. A REST API with the new name will be created and appears in the **REST APIs** page.

Note: If you have created a REST API by using a file containing the Open API specification (formerly known as the Swagger specification) as a template, a copy of the REST API (including its dependencies namely its Integrations, Document Types, and Resources) is created. All resources and document types are renamed as per the new name. When Integrations encapsulated by the copied REST API are modified, the changes are reflected only in the copied REST API and not in the REST API from which it was copied. This is because, if you have created a REST API using the **Build with Swagger** approach, each REST API has its own set of Integrations.

If you have created a REST API by using an existing set of Integrations (**Build from scratch**), a copy of the REST API will be created. You can type a new name at the time of copying and a REST API with the new name will be created. The same REST APIs that are referred to in the original REST API will also be referred to in the copied REST API. If the referred to Integration is modified in the copied REST API, then the changes will reflect in the original REST API because both the REST APIs refer or point to the same Integration. However, if another path with a different Integration is added to the copied REST API, this will not show in the original REST API.

Exporting REST APIs

Integration Cloud allows you to export REST APIs from the **REST APIs** page. The export capability is available only if you have the required license for exporting REST APIs. You can export REST APIs from one tenant and import those REST APIs to another tenant. Ensure that you have the **Export** Integration permission to export REST APIs.

When a REST API is exported, all its dependencies (Integrations, Document Types, and Resources) are exported. Integrations referred to by the REST API, their dependencies, and all dependant Document Types of the referred Integrations are also exported.

Note: If assets used by a REST API are deleted, you will not be able to export the REST API.

To export REST APIs

1. From the Integration Cloud navigation bar, click **Develop > REST APIs**. The **REST APIs** page appears.
2. Select the REST APIs from the **REST APIs** page and click **Export**. If the REST APIs you are exporting use Reference Data, Document Types, SOAP, or REST Applications, then those Applications including the Reference Data and Document Types will also be exported along with the REST APIs.

The **Confirm Export** dialog box appears.

3. Click **Export** to export the REST APIs. The REST APIs will be downloaded as a zip file to your default download folder. The zip file size must not be greater than 50 MB.

Note: Do not modify the contents of the exported zip file. If you modify the contents of the zip file, the REST APIs cannot be imported back to Integration Cloud.

Importing REST APIs

Integration Cloud allows you to import REST APIs from the **REST APIs** page. You can import REST APIs from a zip file that was earlier exported from Integration Cloud. You can export REST APIs from one tenant and import those REST APIs to another tenant. You can import REST APIs provided you have the **Create** Integration permission.

Note: If you want to import a REST API that has an on-premises Application, before importing the REST API, ensure that you upload the same on-premises Application to Integration Cloud. Else, you will not be able to import the REST API.

To import REST APIs

1. From the Integration Cloud navigation bar, click **Develop > REST APIs**. The **REST APIs** page appears.
2. Click **Import REST APIs**.

The **Import REST APIs** page appears.

3. Click **Browse** and select the zip file that contains the exported REST APIs. The zip file size must not be greater than 50 MB. The REST APIs available in the zip file will appear in the pane.
4. Select the REST APIs that you want to import and then click **Import**.

The REST APIs appear in the **REST APIs** page.

Note: While importing a REST API, the REST API including its dependencies are imported. If there is a REST API with the same name, you will be asked to provide a new name. If you have Integrations that have the same name as those referred to by the REST API, you will be asked whether you want to override those integrations. If you choose to override, then the Integrations in your current tenant will be removed. If you want to retain the Integrations, you can cancel importing the REST API. If you want to retain your current Integrations and also import the REST API, create a copy of your Integrations by providing another name and then override your current Integrations at the time of import of the REST API.

Document Types

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

Integration Cloud also allows you to create Document Types for already created REST Applications from the **Connect > Applications > REST Application > Document Types** link or from the Request Body and Response Body panels while creating a REST Application. Document Types created for a REST Application does not appear in the **Develop > Document Types** page but appears in the **Document Types** panel for the selected REST Application. Document Types for REST Applications are used in the Request Body and Response Body of an **Action**.

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

Document Types can provide the following benefits:

- Using a Document Type as the input or output signature for an Integration can reduce the effort required to build an Integration.
- Using a Document Type to build document or document list fields can reduce the effort and time needed to declare input or output parameters or build other document fields.
- Document Types improve accuracy because there is less possibility to introduce a typing error while typing field names.
- Document Types make future changes easier to implement because you can make a change in one place (the Document Type) rather than everywhere the Document type is used.

You can use Document Types to define the input or output parameters for an Integration. Input and output parameters are the names and types of fields that the Integration requires as input and generates as output. These parameters are also collectively referred to as a signature. For example, an Integration can take two string

values, an account number (AcctNum) and a dollar amount (OrderTotal) as inputs and produces an authorization code (AuthCode) as the output. If you have multiple Integrations with identical input parameters but different output parameters, you can use a Document Type to define the input parameters rather than manually specifying individual input fields for each Integration.

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

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

To add or edit a Document Type

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

Note: You can create Document Types for already created REST Applications from the **Connect > Applications > REST Application > Document Types** link or from the Request Body and Response Body panels while creating a REST Application.

From the **Document Types** page, you can add, edit, delete, or copy a Document Type.

2. To edit an existing Document Type, select a Document Type from the **Document Types** page and click **Edit**. Select a field to view the **Field Properties** panel.
3. To create a new Document Type, from the **Document Types** page, click **Add New Document Type**.
4. Provide a name and description of your Document Type. Required fields are marked with an asterisk on the page.
5. Click **Load XML** to generate a Document Type from the XML structure or click **Load JSON** to generate a Document Type from the JSON structure.
6. Click the **+** icon to add a new field. You can update the field properties by using the **Field Properties** window.

Provide the **Name** and **Type** of the field in order to define the structure and content of the Document Type. A field can be a String, Document, Document Reference, Object,

Boolean, Double, Float, Integer, Long, or Short. If you select Document Reference, select the **Document Reference**. Fields are used to declare the expected content and structure of Integration signatures, document contents, and pipeline contents. In addition to specifying the name and type of a field, and whether the type is an **Array**, you can set properties that specify an **XML Namespace** and indicate whether the field is required at runtime by selecting the **Required** option.

You can copy a field from the fields panel by clicking the  icon. Depending on the context, you can either paste the field or the field path by clicking the  icon. For example, if you copy a field and paste the field in the **Set Value** window in an Integration, the field path will be pasted. If you copy an array item, the path that is pasted includes the item index. For example, if the item that is copied is A/B/C[10], then the pasted path will also include the item index [10]. But if it is pasted in the document tree, it will appear as an array, like A[]. If there are multiple fields with the same name in a document, and one of the occurrences of such a field is copied, then the path when pasted will contain the occurrence number in brackets, for example, the path will be A/B/C(5) if the copied element C is the 5th occurrence under field B.

Note: You cannot modify or paste the child fields of a Document Reference.

Note: When defining a Document type, avoid adding identically named fields to the Document. In particular, do not add identically named fields that are of the same data type.

You can assign an **XML namespace** and prefix to a field by specifying a URI for the XML namespace property and by using the *prefix:fieldName* format for the field name. For example, suppose a field is named *eg:account* and the XML namespace property is set to `http://www.example.com`. The prefix is *eg*, the localname is *account*, and the namespace name is `http://www.example.com`.

Keep the following points in mind when assigning XML namespaces and prefixes to a field:

- The field name must be in the format: *prefix:fieldName*
 - You must specify a URI in the XML namespace property.
 - Do not use the same prefix for different namespaces in the same Document Type, input signature, or output signature.
7. Click **Apply** after you have entered the details and constraints for each field, and then click **Save** to save the **Document Type**.

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

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

Integration Cloud allows you to upload Reference Data from a text file containing tabular data separated by a character, for example, a comma, semicolon, and so on. The uploaded file should not have an empty column heading or space in the first row, and the first row cannot be empty.

Note: See this [“video”](#) on how to upload Reference Data, access the uploaded Reference Data in an Orchestrated Integration, and view the input and output parameters.

The Reference Data block appears under **Services** in the Orchestrated Integration workspace, only after you have created a Reference Data. See [Reference Data Signature](#) for information on the Input and Output parameters. The Reference Data is also available in Point-to-Point Integrations while transforming data. You can access the uploaded Reference Data in Orchestrated Integrations as a list of documents by using the *Reference Data block* and providing an appropriate name. You can filter the documents returned into the pipeline by the Reference Data block.

You can create a Reference Data only in the **Development** stage but can view, edit, delete, and download the Reference Data *in all stages*. The **Status** column in the Reference Data table displays **Configured** if the Reference Data is available in the current stage (Stage in view) and displays **Not Configured** if the Reference Data is not available in the current stage but available in any other stage. The **Delete** and **Download** options are enabled if the Reference Data is available in the current stage. The **Edit** option is enabled in all stages. The **Download** option allows you to download the previously uploaded Reference Data, edit it, and then upload the modified file. In this way you can upload different sets of data in different stages.



Name	Used in	Modified On	Modified By	Status
country_code	Text	05/16/2018 02:48:05 AM PDT	Kumar S	Configured

You can pull an Integration only if the Integration is consistent in the source stage, that is, all references of the Integration are present in the source stage. While pulling an Integration from the source stage to the target stage, if the same Reference Data is not available in the target stage, then you must configure the Reference Data in the target stage. If the same Reference Data is available in the target stage with a signature mismatch, for example, the number of columns or the column names are not the same, then you can either reconfigure the Reference Data or skip the reconfiguration. If the Reference Data is already available in the target stage and the signature is same as in the source stage, then it will not be copied from the source stage to the target stage while pulling the Integration.

Note: If a Reference Data is in the Development stage, the Reference data can be deleted even it is referenced or used in an Integration. Integrations using the deleted Reference Data will be in an inconsistent state.

If a Reference Data is in a stage other than the Development stage, the Reference data can be deleted only if it is configured in that stage and not used in any Integration.

To add or edit a Reference Data

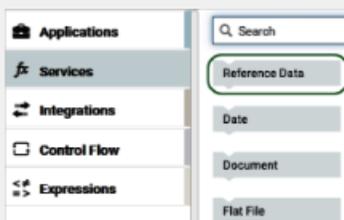
1. From the Integration Cloud navigation bar, click **Develop > Reference Data**. The **Reference Data** page appears.
2. To edit an existing Reference Data, select a configured Reference Data from the **Reference Data** page, and click **Edit**.
3. To create a new Reference Data, from the **Reference Data** page, click **Add New Reference Data**. You can create a Reference Data only in the **Development** stage.

The **Upload Reference Data** page appears.

4. Provide a name and description of your Reference Data. Required fields are marked with an asterisk on the page.
5. For **Reference Data File**, click **Browse** and select the file. Only a text file having tabular data is supported. The maximum file size you can upload is 1 MB. Further, the file should not have an empty column heading or space in the first row and the first row cannot be empty. This is because the first row of data is read as column headings.
6. Click **Next** to define and preview the Reference Data. Select the field separator and the text qualifier.
7. Determine the encoding of the Reference Data file and from the **File Encoding** drop down list, select the same encoding. Click **Load Preview** to preview the data. If you select an incorrect encoding, garbage characters will appear in the preview pane.
8. Click **Next** to review the Reference Data and then click **Finish** to create the Reference Data.

The new Reference Data appears in the **Reference Data** page with the status as **Configured**.

Note: The **Reference Data** block appears under **Services** only after you have created a Reference Data and the Reference Data service will be available while creating an Orchestrated Integration.



The Reference Data is also available in Point-to-Point Integrations while transforming data.

Reference Data Signature

Reference Data signature is derived from the column names of the uploaded text file. You can filter the Reference data by providing an appropriate **matchCriteria**. The output of Reference Data is a list of documents that match the specified **matchCriteria**.

Note: The root element in the output of Reference Data created from version 2.1.0 has the same name as the Reference Data.

Input Parameters

matchCriteria **Document** Criteria on which documents from the Reference Data will be matched.

Parameters for *matchCriteria* are:

path: Column names of the Reference Data.

compareValueAs: Optional. Allowed values are string, numeric, and datetime. The default value is string.

datePattern: Optional. Pattern will be considered only if *compareValueAs* is of type datetime. Default value is MM/dd/yyyy hh:mm:ss a.

joins: List of join criteria.

Each join criteria consists of:

operator: Allowed values are equals, doesNotEqual, greaterThan, greaterThanEqual, lessThan, lessThanEqual, equalsIgnoreCase, contains, doesNotContain, beginsWith, doesNotBeginWith, endsWith, doesNotEndWith.

value: Optional. Allowed values are string, numeric, and datetime. The default value is string.

joinType: Specifies the way two joins can be linked. Values are "and" or "or". Default value is "and".

Output Parameters

<Reference Data Name> **Document List** List of documents that match the retrieve criteria.

In the following example, the flat file contains “Type”, “Our Type”, and “Marketer” as headers and has one or more data rows.

Type,Our Type,Marketer

Existing - Growth,Growth,HUNT & SONS INC

The following graphic illustrates the generated Reference Data signature:

The image shows two side-by-side screenshots of the Reference Data signature tool. The left screenshot, titled "AccountType Input", shows a search bar at the top with the text "Search for Field...". Below it is a tree view of the signature components: "matchCriteria [] *" (expanded), "path *", "compareValueAs", "datePattern", and "joins [] *". The right screenshot, titled "AccountType Output", shows a search bar with "Search for Field...". Below it is a tree view of the signature components: "AccountType []" (expanded), "Type", "Our Type", and "Marketer".

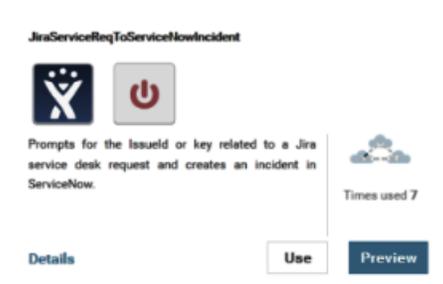
Recipes

Recipes are pre-built Orchestrated or Point-to-Point Integration templates that can be used to create an Integration. Recipes are based on the most common integration needs and can significantly reduce the effort required to build an Integration. A recipe includes associated assets, for example Applications, Operations, Reference Data, and so on, that are used to create an Integration. A detailed description of the recipe along with its assets are available for preview, which helps you to select the right recipe. All Integrations created from recipes are initially copied to the development stage. The Recipes page is paginated to identify the sequential order of the pages. You can also select the number of recipes to be viewed per page.

To view and use recipes

1. From the Integration Cloud navigation bar, click **Develop > Recipes**. The **Recipes** page appears. By default, recipes for all Applications and for all Integration types (Orchestrated and Point-to-Point) are displayed. You can search Recipes by Application names and for a specific Integration type. The **Recipes** page also displays the number of times each recipe has been used to create Integrations and

the Applications referenced in the recipe. If there are more than two Applications referenced in the recipe, the **Recipes** page also displays the incremental number. The Recipes page is paginated to identify the sequential order of the pages. You can also select the number of recipes to be viewed per page.



2. Click **Preview** to see a view-only mode of the Integration details of the recipe.
3. Click **Details** to view a detailed description of the recipe and the references in the **Recipe Details** page.
4. From the **Recipes** or **Recipe Details** page, click **Use** if you want to apply the recipe to create a new Integration. The **Connect to Applications** page appears. Depending on the number of Integrations referenced in your recipe, a message is displayed at the top of the page. You will be asked to configure all the Integrations in your recipe one after another.
5. In the **Connect to Applications** page, select the **Account** for each Application or create a new Account, and then click **Next**.

The **Overview and Save Integration** page appears.

6. In the **Overview and Save Integration** page, provide a name and description for your Integration. By default, the recipe name and recipe description is displayed.
7. Click **Finish**. If you have existing references (Reference Data, Document Types, and custom Operations) with the same name in the development stage, the **Copy References** page appears. Click **Cancel** to go back to the **Overview and Save Integration** page. By default, all references are selected in the **Copy References** page.
8. Deselect the references that you do not want to replace and then click **Continue** to replace or overwrite all the selected references from the recipe in the development stage.

The Integration details page appears for the newly created Integration.

5 Monitor

■ Monitor Overview	430
--------------------------	-----

Monitor Overview

You can view and monitor Integration executions and performance details on the **Dashboard**. The **Execution Results** screen allows you to view the audit trail of all the executions that happened in a stage for an Integration or for all Integrations. The **Audit Log** screen displays logs related to additions, deletions, updates, export, schedule, login, logout, password changes, record access attempts, access violations, deployments, and so on. You can also create **Alert Rules** and send email messages to selected users for one or more Integrations based on Integration execution results for different stages.

Dashboard

The **Dashboard** provides a centralized and intuitive way to view and monitor Integration executions and performance details. To view the **Dashboard**, click **Monitor > Dashboard**.

You can identify and diagnose problems for those Integrations that are available in the active stage. To select another active stage, click **Stages > Change Stage To View**. You can view the **Dashboard** if the Access Profile assigned to you is also specified for that stage in the **Manage Stages (Stages > Manage)** page. Further, you must be able to access the stages to view the dashboard.

The **Dashboard** displays the following details:

- When was the Dashboard last refreshed.
- Drop-down list to select an Integration or select all Integrations to view the execution details.
- Total number of documents processed by an Integration or for all Integrations in the active stage. Documents processed appear only if the Integration invokes the **countProcessedDocuments** service to count the number of documents processed by the Integration. See the **countProcessedDocuments** service available in the **Flow** block under the **Services** category for more information.

The screenshot shows the configuration interface for an integration. On the left, there is a dropdown menu for selecting a service, currently set to 'Flow'. The dropdown list includes options: 'clearPipeline', 'countProcessedDocuments' (highlighted), 'getHTTPRequest', 'getLastError', and 'getSessionInfo'. On the right, the 'Input Parameters' section is visible, showing two parameters: 'status' and 'incrementBy'. The 'status' parameter is a string with optional values 'success' or 'fail'. The 'incrementBy' parameter is a string with an optional value, used to increment the number of documents processed. Below the input parameters, the 'Output Parameters' section shows 'None'.

Input Parameters	
<i>status</i>	String Optional valid values are "success" or "fail". Set status to "success" to count the number of successfully processed documents, else set it to "fail". Default value is "success".
<i>incrementBy</i>	String Optional. Increment the number of documents processed by an Integration. Every time the service is used, successful or failed documents are incremented by the given value. Default value is 1.
Output Parameters	
None	

- Number of completed and failed Integration executions that happened during the selected time period in the active stage.
- Number of Integration executions that have completed with errors during the selected time period in the active stage.
- Completed Integration executions, failed Integration executions, and Integration executions that completed with errors displayed in a pie chart, along with the **Success Rate**, that is, the percentage of completed Integration executions compared to the total Integration executions, during the selected time period in the active stage.
- Number of in-progress Integration executions in the active stage. You can click on the **In-Progress Executions** number to view the in-progress Integration execution details in a table. You can terminate in-progress Integration executions from the Execution Results details page. The *Terminate* audit log entry is created. You can terminate an in-progress Integration execution if you have the *Execute* Integration permission.

To terminate an in-progress execution, click the **Terminate** option available in the Execution Results details page.

Run at	Deployment Stage	Overall Result	
03/21/2018 10:54:37 AM IST	DEVELOPMENT	 In-Progress	

Note: The **Terminate** option is available only if the current logged in user has the *Execute* Integration permission and the Integration status is in-progress.

- Successful Integration executions, failed Integration executions, and Integration executions that completed with errors displayed in a bar chart along with clickable links, for the selected time period in the active stage. You can click the Integration execution links available above the bar charts to display the relevant Integration execution details in the table. You can also point to each bar in the chart to view the date and time when the Integration executed and the result of the Integration execution.
- Name of the Integration, stage name, when the Integration started, the Integration run duration, documents processed details, result of the Integration execution (Completed Successfully, Failed, Completed with errors), and the Integration execution message displayed in a tabular format. The **Documents Processed** column displays the total number of documents processed by an integration, the number of documents processed successfully, and the success percentage. Values in this column appear only if the Integration invokes the `countProcessedDocuments` service to count the number of documents processed by the Integration. For more information, see the `countProcessedDocuments` service available in the **Flow** block under the **Services** category.
- If you click a row on the table, you can view the execution information as well as the operations details for the Integration . See [Execution Results](#) for more information.

Execution Results

The **Execution Results** page allows you to view the audit trail of all the Integration executions that happened in the selected active stage, during a specified time period. You can also restart or resume an Integration execution, specify the number of days to retain the entries, and download the entries.

Note: To view execution results, ensure that the Access Profile of the user is assigned to the active stage.

To view the execution results

1. From the Integration Cloud navigation bar, click **Develop > Integrations**.

The **Integrations** page appears.

2. From the **Integrations** page, select the Integration for which you want to view the execution results.
3. Click the *Integration link* to view the Integration **Overview** page. You can click **Edit** to modify the Integration, click **Delete** to delete the Integration from this page, or click **Run Now** to execute the Integration. You can also see the last five execution results in the **Last 5 Execution Results** tab.

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

4. In the **Execution Results** page, select the **Integration** and the time period for which you want to view the execution results. You can also select **All Integrations** to view the execution results for all Integrations in the active stage for a specific time period. The **Custom Range** option allows you to set a time period to view the results. The default time period is for the last 24 hours.

Execution results are displayed in a tabular form. You can filter the results in the table by clicking on the status filter circles available on the top-right corner above the table. The numbers inside the status circles indicate the sum of the execution counts for that status.



- **All** - All operations of an Integration, which have **Completed Successfully**, **Failed**, and **Completed with errors** are displayed.
- **Completed Successfully** - All operations of an Integration that completed successfully while executing are displayed.
- **Failed** - Exceptions occurred while executing an operation in an Integration.
- **Completed with errors** - Exceptions occurred while executing an operation in an Integration and caught by the try-catch block in an orchestrated Integration.

5. In the **Execution Results** page, click **Download Results** to download the execution results, or click **Modify Retention Period** and specify the number of days to retain the execution result entries. You can retain entries up to 30 days. Entries whose age exceeds the specified retention period are deleted. Default value of the Retention Period is 30 days.

Note: User specific data which may be considered as personal data will be stored and retained till the retention period defined in Execution Results.

6. In the **Execution Results** page, click an Integration in the table to view more information about the selected Integration Execution. The **Execution Details** page appears.

In the **Execution Details** page, the **Documents** row displays the total number of documents processed by the integration, the number of documents processed successfully, the number of documents that did not process successfully, and the success percentage. Values in this row appear only if the Integration invokes the **countProcessedDocuments** service to count the number of documents processed by the Integration. See the **countProcessedDocuments** service in the **Flow** block under the **Services** category for more information. You can also terminate in-progress Integration executions from the Execution Results details page. The **Terminate** option is available only if you have the *Execute* Integration permission and the Integration status is in-progress. The **Execution Information** section displays when the Integration execution was started, when it ended, the duration of the execution, who executed the Integration, through which channel the Integration was executed, for example, Scheduler, User Interface, and REST Interface, the execution result reference, that is, the Integration execution result reference identifier, and the business data details.

7. The **Execution Results** page also provides information about operations for the selected Integration. Click **Show Everything** to view all information about the operation execution including business data and custom messages. Click **Only Business Data** to view only the logged business data information. Click **Only Custom Messages** to view only custom messages. You can filter the results in the table by clicking on the status filter circles on the top-right corner of the Operations table. Click on the **All** (blue) circle to view operation information, business data, and custom messages. Click on the **Successful** (green) circle to view only successful operation information and business data. Click on the **Failed** (red) circle to view only failed operation information and business data. Custom messages appear only if you have set up log messages. See the **logCustomMessage** service in the **Flow** block under the **Services** category for more information on how to set up custom messages in an Integration.
8. In the **Execution Results** page, select a row and click **Restart** to edit the input data and restart the Integration execution from the beginning, even though the previous execution has been successful. When an Integration is restarted, the Audit Log entry displays "Restart".

Click **Resume** to edit the input data for failed operations and execute the failed and not yet executed operations. When an Integration is resumed, the Audit Log entry displays "Resume".

Note: The "Restart/Resume" capability is available only if you have the required license for restarting and resuming Integrations. You must also have the Integration execution (Execute) permission if you want to restart or resume an execution.

Note: You must select the **Enable executions to be restartable** option in the **Integration Details** page in order to enable Integration executions to be restartable or resumable. See [Integration Details](#) for more information.

The following table provides information on when you can restart or resume an Integration execution:

<u>Execution Result Status</u>	<u>Restartable</u>	<u>Resumable</u>
Completed Successfully	Yes	No
Completed with Errors	Yes	No
Failed	Yes	Yes
In-Progress	No	No

Audit Log

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

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

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

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

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

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

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

Alert Rules

Integration Cloud allows you to create alert rules and send email messages to selected users for one or more Integrations based on Integration execution results (Failed, Completed with Errors, or Completed Successfully) for the stage in view. To change the stage in view or the current active stage, click **Stages > Change Stage To View**. All executions that have occurred within a 15-minute period and have alert rules configured are sent as email messages to specified users.

Note: Email messages are sent only if there are executions that match the alert rules.

Note: Ensure that the Access Profile of the user is assigned to the active stage. To select another active stage, click **Stages > Change Stage To View**.

To create a new alert rule

1. From the Integration Cloud navigation bar, click **Monitor > Alert Rules**. The **Alert Rules** page appears. You can edit, delete, activate, or deactivate an existing alert rule from this page.

2. From the **Alert Rules** page, click **Add New Alert Rule** to create a new alert rule.

The **Add New Alert Rule** page appears.

3. On the **Add New Alert Rule** page, complete the following fields. Required fields are marked with an asterisk on the page.

Field	Description
Save As	Provide a name for the alert rule.
Description	Provide an optional description for the alert rule.
For the following Integrations in the stage	Click the + icon and select the Integrations in the active stage you want this alert rule to apply.

Field	Description
when their executions have	<p>Select the Integration execution results.</p> <ul style="list-style-type: none"> ■ Failed - Exceptions occurred while executing an operation in an Integration. ■ Completed with Errors - Exceptions occurred while executing an operation in an Integration and caught by the try-catch block in an orchestrated Integration. ■ Completed Successfully - All operations of an Integration that completed successfully while executing.
send an email message to the following users	<p>Click the + icon and select the active users to whom you want to send email alerts. Alerts will be sent to the email address specified in the Basic tab of the user's profile (Settings  > Users > User Profile > Basic > Email).</p>

4. Click **Save and Activate** to save and enable the new alert rule.

The new alert rule appears in the **Alert Rules** page. You can disable an alert rule by selecting the alert rule and clicking **Mark As > Inactive**. You can also enable the alert rule by selecting the alert rule and clicking **Mark As > Active**.



Note: Alerts will be sent every 5 minutes. If you do not want to receive email alerts for Integration executions, go to the **Alert Rules** page, open the Alert Rule for editing, and remove your username from the user's list.

6 Stage in view

■ Manage	438
■ Deploy	440
■ Change Stage To View	440

Manage

Stages provide safe environments for development and testing that are separated from the production environment. They allow assets to migrate from one environment to another environment. When you set up a stage, an environment is created for testing and executing the Integrations.

Integration Cloud provides ways to manage the Integration development life cycle. The typical life cycle of an Integration development involves creating Integrations, testing them, and making them production worthy. Each of these activities can be termed as different stages of an Integration development life cycle. To aide these activities, Integration Cloud provides you with **Stages**.

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

- Development
- Test
- Pre-Live
- Live

By default, every user gets a **Development Stage**. *In the Development Stage, you can create, update, delete, or view Integrations. You will not be able to create new assets in other stages.* In other stages, Integrations can be pulled from a preceding stage or deleted. Further, Integrations can be pulled into a stage only from a preceding stage.

Note: You can access a stage only if your Access Profile is assigned to the stage.

You can add stages only in the following order:

- Live
- Test
- Pre-Live

You can delete stages only in the following order:

- Pre-Live
- Test
- Live

Note: Users who have the **Administer** permission under **Settings**  **> Access Profiles > Administrative Permissions > Functional Controls > Stages** can add or delete stages.

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

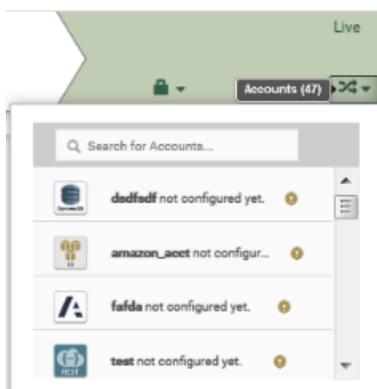
Applying Access Profiles to a Stage

The typical life cycle of an Integration development involves creating Integrations, testing them, and making them production worthy. Each of these activities can be termed as different stages of an Integration development.

Every stage can be assigned a number of Access Profiles and users who are assigned the required Access Profiles can perform activities on that stage. For example, if in an Access Profile, **Execute Integrations** permission is granted, the user assigned with that Access Profile can execute Integrations on the stages to which the Access Profile is assigned. If the Access Profile needs to perform scheduling activity on the Live stage, the Access Profile needs to have access to that stage as well. The Development stage can be accessed by everyone.

Click **Add New Stage** 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.

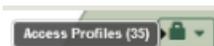


Click **Delete** to delete a stage. You cannot delete the **Development** stage.

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



Note: By default, the **Administrator** and **Regular User** Access Profiles are associated with the Development Stage. If you have created a new Access Profile, ensure that the Access Profile you have created is associated with the Development Stage.

3. Click **Apply**.

The Access Profiles are applied to the selected active stage.

Note: When an Integration is pulled, all its dependents will also be pulled and copied to that stage.

Deploy

This page displays all the assets that are available in the current stage and the previously selected stage. Click the **Change Stage To View** link to deploy assets to another stage.

Note: Users who have the **Deploy** permission under **Settings**  > **Access Profiles** > **Administrative Permissions** > **Functional Controls** > **Assets** can deploy assets.

After you change the current stage to any other stage other than the Development stage, the **Deploy** page gets populated with the relevant assets selected in the **Select Asset Type** drop-down list.

In the **Deploy** page, you can select an asset that was in the earlier stage (left panel) and click **Pull** to pull the asset to the current stage (right panel). You can pull an asset only if the asset is consistent in the source stage (left panel), that is, all references of the asset are present in the source stage. The pulled asset can be deleted from the current stage (right panel).

Note: You can access a stage only if your Access Profile is assigned to the stage.

Change Stage To View

This page allows you to change the current stage. Only active and accessible stages appear in the drop-down list for selection in the **Stage to view** field.

Note: You can access a stage only if your Access Profile is assigned to the stage.

After you change the current stage and click **Submit**, assets and services will be displayed in the user interface pages only for the selected current stage.

The **Stage In View** label on the navigation bar displays the current stage.

Stage in view
Development   

7 Containers

- Containers Overview 444

Containers Overview

Integration Cloud allows you to package existing webMethods Integration Server services as images or repositories and upload them on Integration Cloud using the Docker CLI. Docker is an open-source technology that allows you to deploy applications to software containers. A Docker container is an instance of a Docker image, where the Docker image is the application, including the file system and runtime parameters. To facilitate running webMethods Integration Server in a Docker container, webMethods Integration Server provides a script to use to build a Docker image and then push the resulting Docker image to a Docker registry hosted in Integration Cloud. You can store Docker images in a registry on Integration Cloud and manage those Docker images from the Integration Cloud user interface. Images or a repository are versioned or labeled using tags, that is, a tag is a label applied to an image or a repository. Tags help you to distinguish various images or repositories.

Note: Integration Cloud documentation assumes that you are familiar with Docker technology. An in-depth discussion of Docker and container technology is beyond the scope of this document. For information on using webMethods Integration Server with Docker, see the *webMethods Integration Server Administrator's Guide*.

Images are read-only templates from which containers are instantiated, that is, a container is a runtime instance of an image. A container also consists of an execution environment and a standard set of instructions. After uploading an image on Integration Cloud, you can create and launch services from the image/tag to the desired active stage, specify the number of containers and the container port for each service, and see details of the running instances. A service can contain one or more containers and is defined as a named group of containers created out of a single image tag.

Note: Universal Messaging (UM) can also be run as a container in Integration Cloud. For creating the UM Docker images supported by Integration Cloud, you must run the Integration Cloud UM script to modify the base image before it is uploaded into Integration Cloud.

You can access containers if you have the **Settings**  > **Access Profiles** > **Administrative Permissions** > **Container** > **Access** permission. You can administer containers if you have the **Settings**  > **Access Profiles** > **Administrative Permissions** > **Container** > **Administer** permission. On the **Settings**  > **Access Profile** > **Container** tab, enter the names of the webMethods Integration Server Access Control List (ACL) groups separated by a comma, for example, Administrators, Developers, and so on. Users who are assigned to this Access Profile will also be now part of the webMethods Integration Server container user group (s) and can perform tasks allowed for those user groups. Note that Integration Cloud Administrator profiles are not automatically assigned to the webMethods Integration Server Administrators ACL group. If you do not map an Access Profile to an webMethods Integration Server group, you will not be able to invoke webMethods Integration Server services.

Note: Enabling the CSRF security feature will prevent CSRF attacks. Enable **CSRF Guard** and configure the CSRF guard settings in webMethods Integration Server Administrator before you create the Docker image and upload it to Integration Cloud. See the *webMethods Integration Server Administrator's Guide* on the Software AG Documentation website at "<http://documentation.softwareag.com>" for information on how to enable CSRF Guard.

You can use the Docker Command Line Interface (CLI) to perform the following tasks:

Log in to the system: `#docker login -u <username> -p <password> https://<subdomain>.<domain.com>/`, for example, `docker login -u x@x.com -p test123 https://john.wmic1.com/`.

Tag an image or repository: `#docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2`.

Push or upload an image or repository: `#docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker push john.wmic1.com/john/development/is_912:withkeystore2`.

Pull or download an image or repository: `docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker pull john.wmic1.com/john/development/is_912:withkeystore2`.

Managing Repositories

Images or repositories are read-only templates from which containers are instantiated. The **Repositories** screen allows you to view, delete, and add repositories for the active stage. You can select a repository and click **Delete** to delete a repository or click **Add New Repository** to add a repository. In the **Add New Repository** screen, select the **Deployment Stage** and enter the **Repository Name**. If a stage is not enabled to access containers, contact Support to enable the stage.

To view the details of a tag from the Repositories screen

1. Select a repository and then click the repository link under the **Name** column.
The **Image Tags** screen is displayed. A list of all image tags is displayed along with their names.
2. From the **Image Tags** screen, you can select an image tag and delete it or add a new service for the tag. See [Viewing Tag Details](#) for more information.
3. Click the **Commands** tab to view and copy the commands on how to log in, tag images, push images, or pull images.

Log in to the system: #docker login -u <username> -p <password> https://<subdomain>.<domain.com>/, for example, docker login -u x@x.com -p test123 https://john.wmic1.com/.

Tag an image or repository: #docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>, for example, #docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2.

Push or upload an image or repository: #docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>, for example, #docker push john.wmic1.com/john/development/is_912:withkeystore2.

Pull or download an image or repository: docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>, for example, #docker pull john.wmic1.com/john/development/is_912:withkeystore2.

Viewing Tag Details

This screen displays details of all the tags for a repository. You can delete an image tag or add a new service. You can also click the **Commands** tab to view and copy commands on how to log in, tag images, push images, or pull images.

To view the image tag details

1. From the Integration Cloud navigation bar, click **Containers > Repositories**.
2. Select a repository and then click on the repository link. The **Image Tags** screen appears. A list of all Docker image tags is displayed along with their names.

From the **Image Tags** screen, select a tag and then click **Delete** to delete the image tag if it is not used by any service or click **Add New Service** to create a new service for the image tag. In the **New Service** window, specify the **Service Name**, the **Volume Name**, and the number of **Docker Containers** to instantiate.

3. From the **Image Tags** screen, select an image tag and click on the image tag link to view the Image Tag details screen. The Image Tag details screen displays the deployment stage of the image tag including the image tag label details. The image tag label details are as follows:
 - **Image Type** - Mandatory field. Indicates the type of image, for example, webMethods Integration Server or Universal Messaging.
 - **Description, Build Number, and Version** - Optional fields you had defined while creating the image.
 - **Pushed At** - System generated value. Date and time when the Docker image was pushed to the repository.
 - **Size** - System generated value. Indicates the size of the Docker image.

- **Exposed Port - Mandatory field.** The port you had defined while creating the image. Ensure that the exposed port is the same as defined on your application, that is, on webMethods Integration Server or Universal Messaging.

The Image Tag details screen also displays when the screen was last refreshed, used and available containers for the displayed active stage, and information on all services created for the tag. You can edit, delete, start, stop, or add a new service from the Image Tag details screen. Click on a service link to view the service details screen. See [Managing Services](#) for more information.

4. The **Commands** screen allows you to view and copy the commands on how to log in, tag images, push images, or pull images:

Log in to the system: `#docker login -u <username> -p <password> https://<subdomain>.<domain.com>/`, for example, `docker login -u x@x.com -p test123 https://john.wmic1.com/`.

Tag an image or repository: `#docker tag <imagename>:<tagname> <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker tag is_912:withkeystore john.wmic1.com/john/development/is_912:withkeystore2`.

Push or upload an image or repository: `#docker push <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker push john.wmic1.com/john/development/is_912:withkeystore2`.

Pull or download an image or repository: `docker pull <subdoamin.wmis.com>/<subdomain>/<stage name>/<image-name>:<tag-name>`, for example, `#docker pull john.wmic1.com/john/development/is_912:withkeystore2`.

Managing Services

The **Services** screen displays details of all services in the active stage, the repository name and tag name, status of the service, and number of Docker containers instantiated for a tag. You can start a service, stop a service, edit a service, delete a service, add a new service, refresh the screen, and view details on when the screen was last refreshed.

Note: You can also add a new service for a specified image tag from the **Containers > Repositories > Click a repository link > Image Tags** screen.

From the **Services** screen, select a service and then click the link to view the **Service details** screen. In the Service details screen, you can add or remove docker containers for the service for the selected repository and image tag.

The Service Details screen provides information on the service status (Stopped, Pending, Running), the repository, Deployment stage, and the image tag. You can start a service if it is in a stopped state. The **Docker Containers** pane in the Service Details screen displays information on the docker containers, their status, and the **Admin URL**, which is the webMethods Integration Server Administrator URL. You can modify Docker containers to instantiate by clicking **Manage Containers**, or click the **Admin URL** link to log into webMethods Integration Server Administrator. In the **Exposed Services** pane, you

can view the **Exposed Docker Services**. The exposed Docker service URL can be used to execute the webMethods Integration Server Administrator service running in the Docker container.

Note: Enabling the CSRF security feature will prevent CSRF attacks. Enable **CSRF Guard** and configure the CSRF guard settings in webMethods Integration Server Administrator before you create the Docker image and upload it to Integration Cloud. See the *webMethods Integration Server Administrator's Guide* on the Software AG Documentation website at "<http://documentation.softwareag.com>" for information on how to enable CSRF Guard.

Note: You can now invoke services exposed by webMethods Integration Servers running in the Docker Containers from Integrations. For more information, see the *On-Premises Applications* section and also the *Configuring On-Premise Integration Servers for webMethods Cloud* document.

To add a new service, from the **Containers > Services** screen, click **Add New Service**. The **New Service** screen appears. In the **New Service** screen, enter the following fields:

- **Repository** - Select a repository.
- **Image Tag** - Select an image tag.
- **Service Name** - Enter a valid service name. The service name can contain only alphanumeric characters and should not be more than 16 characters.
- **Volume Name** - Enter the storage volume name. The Volume name is displayed only for Universal Messaging (UM) docker images and represents the data volume for the docker container to persist the container data.
- **Docker Containers** - Enter the number of containers to instantiate for the service.

Click **Save** to create a new service.

To stop a service, from the **Services** screen, select a service that is in **Running** state and then click **Stop**. Click  and refresh the screen to view the latest service status.

To delete a service, from the **Services** screen, select a service that is not in **Running** state and then click **Delete**.

To start a service that is in **Stopped** state, select a service and then click **Start**. The launch service screen appears. Specify the number of **Docker Containers** to instantiate for the selected service and then click **Start**. Click  and refresh the screen to view the latest service status.

8 Cloud Deployment

■ About Cloud Deployment	450
--------------------------------	-----

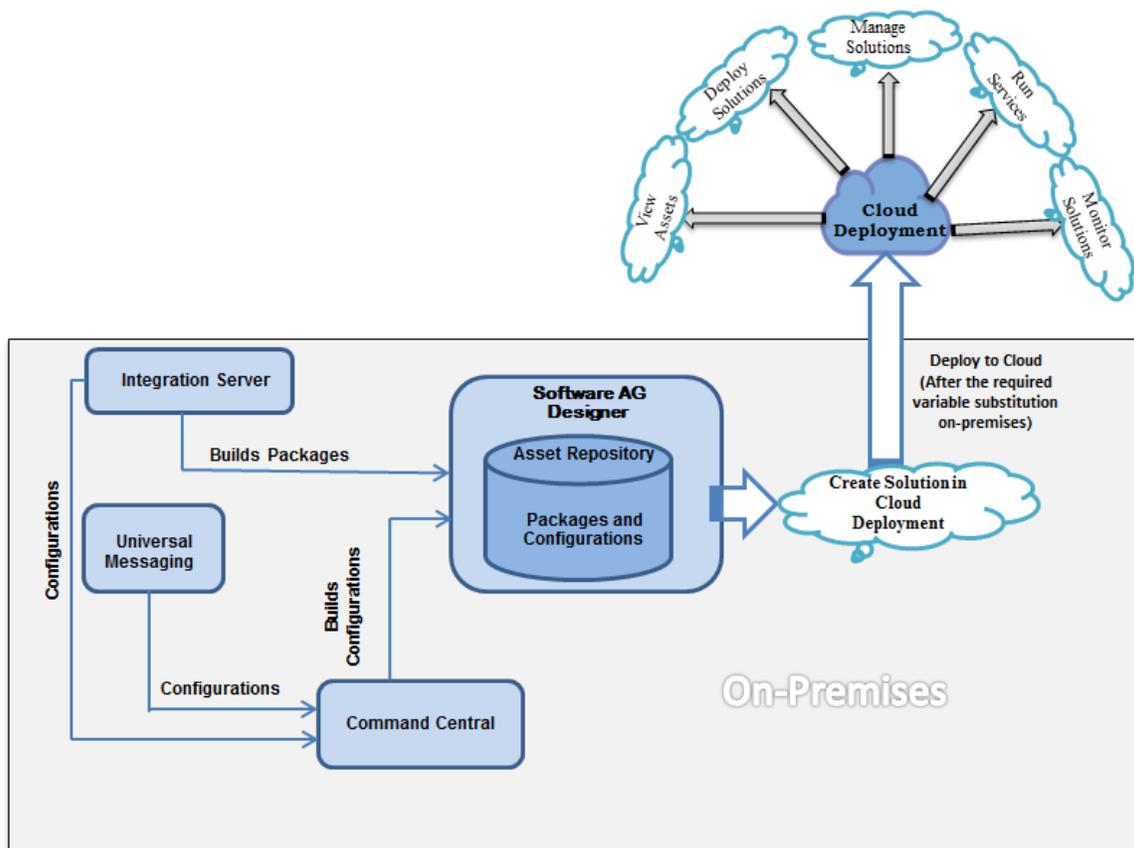
About Cloud Deployment

Cloud deployment is the process of deploying user-created packages and configurations that reside within on-premises runtimes or repositories to webMethods Integration Cloud. Using Software AG Designer or by employing the Continuous Integration (CI) approach, you can seamlessly deploy your on-premises webMethods Integration Server packages, Adapter packages, configuration assets, and customized configurations to solutions created on Integration Cloud. In the cloud deployment context, configuration assets are limited to webMethods Integration Server and Universal Messaging configurations.

Note: See this [“video”](#) for information on how to perform Cloud Deployment.

Note: See the *Deploying to webMethods Integration Cloud* document for information on how to deploy user-created packages and configurations that reside within on-premises runtimes or repositories to webMethods Integration Cloud.

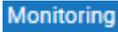
The following figure provides a high-level overview of the process involved in deploying on-premises webMethods Integration Server packages and configuration assets to Integration Cloud.



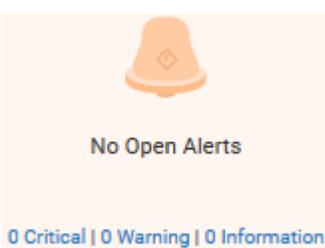
With Software AG Designer, you can deploy the webMethods Integration Server packages or configuration assets that you have created, verified, and tested on on-premises webMethods Integration Server or Universal Messaging to Integration Cloud. When you initiate the deployment from Software AG Designer, webMethods Integration Server packages and configuration assets are built from webMethods Integration Server and Software AG Command Central respectively, and are published to the Asset Repository available in Software AG Designer. After performing variable substitutions to make the on-premises configuration data compatible for cloud deployment, you can publish the packages and configurations to an asset repository provisioned for the tenant on Integration Cloud.

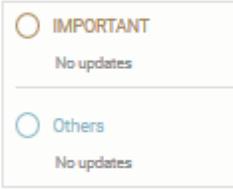
User Interface elements

The following table describes the various user interface elements that appear on the **Cloud Deployment** workspace:

Page Elements	Icon	Description
webMethods Cloud Deployment		Access the home page.
Solutions		View and create solutions, deploy solutions to stages, view the Asset Repository, and manage webMethods Integration Server instances.
Monitoring		View the overall status of the solutions, landscapes, system and runtime alerts, KPI graphs of the runtimes, service executions of the webMethods Integration Server instances, availability of the run times, alert status of the solutions, and logs.
Availability of the runtimes		View the overall runtime availability for

Page Elements	Icon	Description
Usage Statistics	 <p>The icon displays two gauges. The left gauge is labeled 'CPU' with a scale from 0 to 10 and a current value of 5. The right gauge is labeled 'Memory (in Gb)' with a scale from 0 to 24 and a current value of 16. The title 'Usage Statistics' and subtitle 'Total usage across all stages' are at the top.</p>	<p>all the solutions in the last 24 hours.</p> <p>The usage statistics shows the CPU and Memory usage for all solutions in all stages.</p> <p>The CPU bar shows the maximum CPU that is licensed for the tenant. The colored part of the bar shows how much of the allowed CPU is currently used by the tenant.</p> <p>The Memory bar shows the memory in Gb that is licensed for the tenant. The colored part of the bar shows how much of the allowed memory is currently used by the tenant.</p>
Change the stage	 <p>The icon is a blue rectangle with the text 'Stage in view' in white and 'Development' in white below it.</p>	<p>Change and manage stages.</p>
App Launcher	 <p>The icon is a blue square containing a white grid of six dots arranged in two columns and three rows.</p>	<p>Access integration Apps.</p>
Help	 <p>The icon is a blue square containing a white question mark.</p>	<p>Help topics, TECHcommunity, Licensing capabilities and usage information, and the About page. The About page displays the version information, Global Support information, Copyright information,</p>

Page Elements	Icon	Description
		Impressum and Privacy Policy, and the release readme.
Profile		Name of the logged in user, profile information of the logged in user, and Logout option.
Solutions section		Total number of solutions created. View the “Solution List” on page 454 page by clicking the Details link and the “Monitoring Dashboard” on page 471 by clicking the Monitoring link.
Service Executions section		Number of service executions and their status in the last 24 hours. View the Services page under Monitoring by clicking the service execution link.
Active Alerts section		Number of currently open alerts and their severity. Click on the links to go to the “Alerts” on page 477 page.

Page Elements	Icon	Description
Last Login, System Updates, and Help topics	<p>Last Login 12/04/2018 09:18:15 PM PST</p> <p>System Updates</p>  <p>What's New</p> <p>Help Topics</p> <ul style="list-style-type: none"> Cloud Deployment Solutions Monitoring 	When was the last login, important and other system updates, what is new in this release (“ What's New ” on page 23), and context-sensitive help topics.

Solutions

A **Solution** consists of user-created packages bundled together into one coherent service. It is a logical combination of webMethods Integration Server packages, Adapter packages, Services, and webMethods Integration Server and Universal Messaging configuration assets or configurations.

Note: Integration Cloud Integrations can now invoke Cloud Deployment webMethods Integration Server “[services](#)” on page 210 for the same tenant. A new pre-defined Application, “[Cloud Deployment](#)” on page 112, is added in Integration Cloud. Using this Application, you can select the solution webMethods Integration Server services that you want to call from Integration Cloud.

Solution List

The **Solution List** page displays the solutions created in Integration Cloud. All solutions created are initially copied to the Development stage. You can create a new solution only in the Development stage. Further, the solution cannot be modified after it is created. You can configure the solution in subsequent stages but once a solution is configured in a stage, it cannot be modified again in that stage.

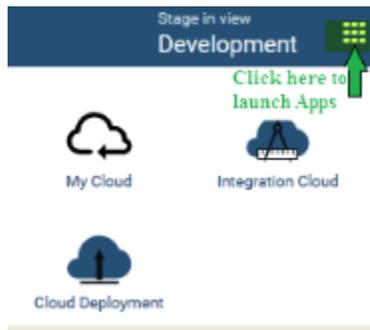
Note: You must first create a solution in Integration Cloud before deploying the assets and configurations.

Note: In a stage, you can configure only those solutions that are marked as **Not Configured**.

Click **Create New Solution** to create a new solution or click on an existing solution to view the solution details page.

To create a solution

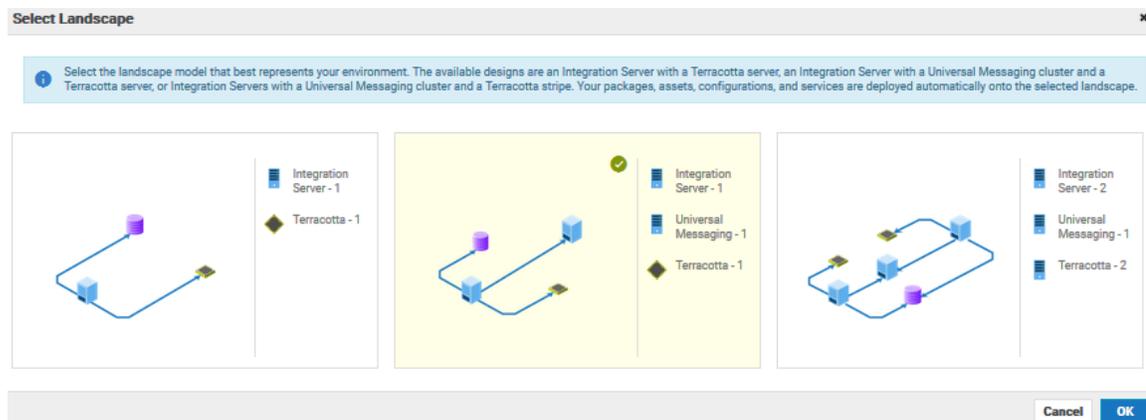
1. Switch to the **Cloud Deployment** perspective.



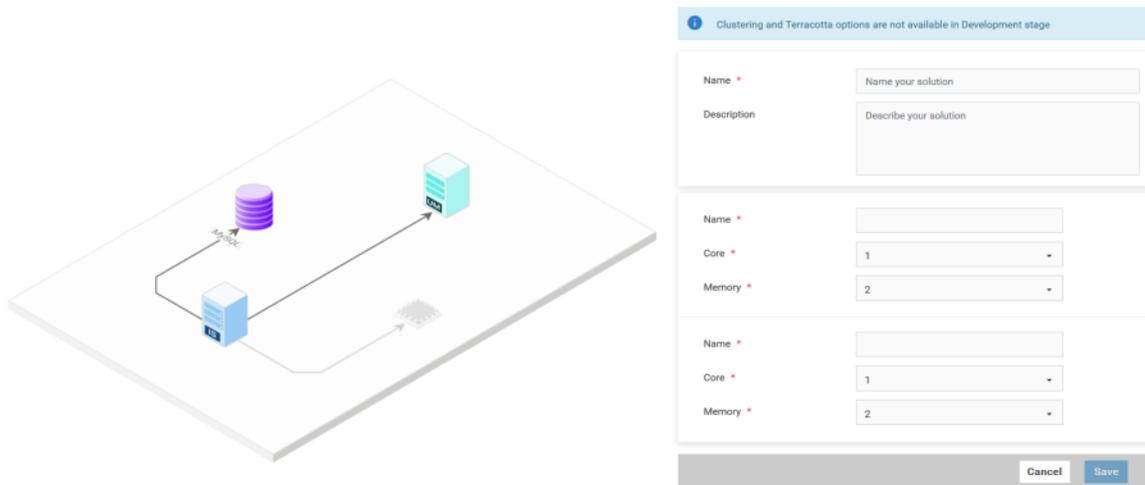
2. From the **Cloud Deployment** navigation bar, click **Solutions > Solution List**.

The **Solution List** page appears.

3. From the **Solution List** page, click **Create New Solution** to create a solution. You can create a solution only in the **Development** stage.
4. Select the landscape model that best represents your environment.



5. The available designs are an webMethods Integration Server with a Terracotta server, an webMethods Integration Server with a cluster and a Terracotta server, or webMethods Integration Servers with a Universal Messaging cluster and a Terracotta stripe. Your packages, assets, configurations, and services are deployed automatically onto the selected landscape.
6. Click **OK**.
7. In the **New Solution** page, fill in the solution **Name**, the solution **Description**, the name of the webMethods Integration Server and Universal Messaging instances, number of CPU **Cores**, and **Memory** characteristics of the hardware to support each service in the solution landscape.



Note: You can select the webMethods Integration Server and Universal Messaging icons to highlight the fields in the **New Solution** page. Terracotta is available only when webMethods Integration Server runs in a clustered mode. Further, clustering and Terracotta options are not available in the **Development** stage.

- Click **Save** to save the solution.

The new solution is created and appears in the **Solution List** page. The solution cannot be modified after the solution is created. You can configure the solution in subsequent stages but once a solution is configured in a stage, it cannot be modified again in that stage.

Note: You can now [“deploy” on page 461](#) the solution to the [“next stage” on page 440](#).

Click the  icon and select **Deactivate** to deactivate a solution. All packages and assets will be permanently deleted and cannot be recovered. A deactivated solution can be activated again.

Select **Delete** to permanently delete the solution. If a solution is configured in a subsequent stage, it will be permanently deleted from the current stage and cannot be recovered. Further, if you delete the solution, you will not be able to promote assets from the current stage.



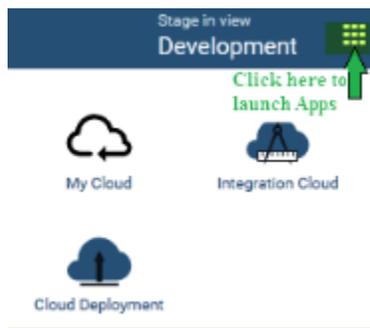
Exploring a Solution

The Solution details page allows you to view the packages, assets, configurations and services for different runtimes in the solution, deploy the solution to another stage, view the Asset Repository, and manage the solution, that is, view the landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances.

Note: You can create a new solution only in the Development stage. The solution cannot be modified after the solution is created. You can configure the solution in subsequent stages but once a solution is configured in a stage, it cannot be modified again in that stage.

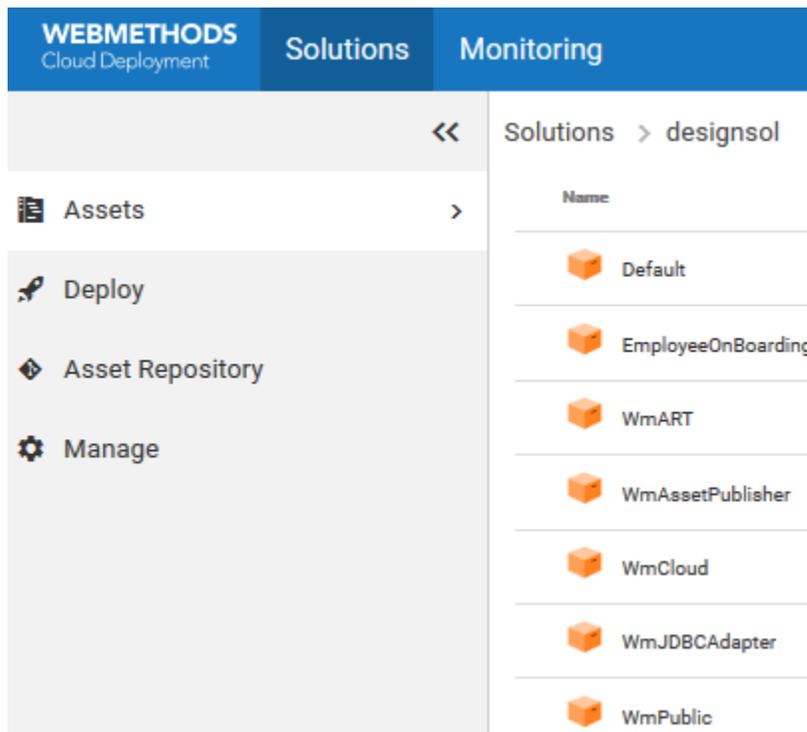
To view the Solution Explorer

1. Switch to the **Cloud Deployment** perspective.



2. From the **Cloud Deployment** navigation bar, click **Solutions > Solution List**.
The **Solution List** page appears listing all the solutions.
3. Click on an existing solution. The Solution Explorer page appears.

Solution Explorer



The following table provides a high-level overview of the Solution Explorer page:

Component	Description
“Assets” on page 458	View the Packages, Folders, Assets, and Services for webMethods Integration Server and Adapters, and configurations for the Universal Messaging runtime.
“Deploy” on page 461	Deploy the solution to another stage.
“Asset Repository” on page 462	View the Asset Repository which displays the contents of the on-premises packages published to Integration Cloud.
“Manage” on page 463	View the landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances.

Assets

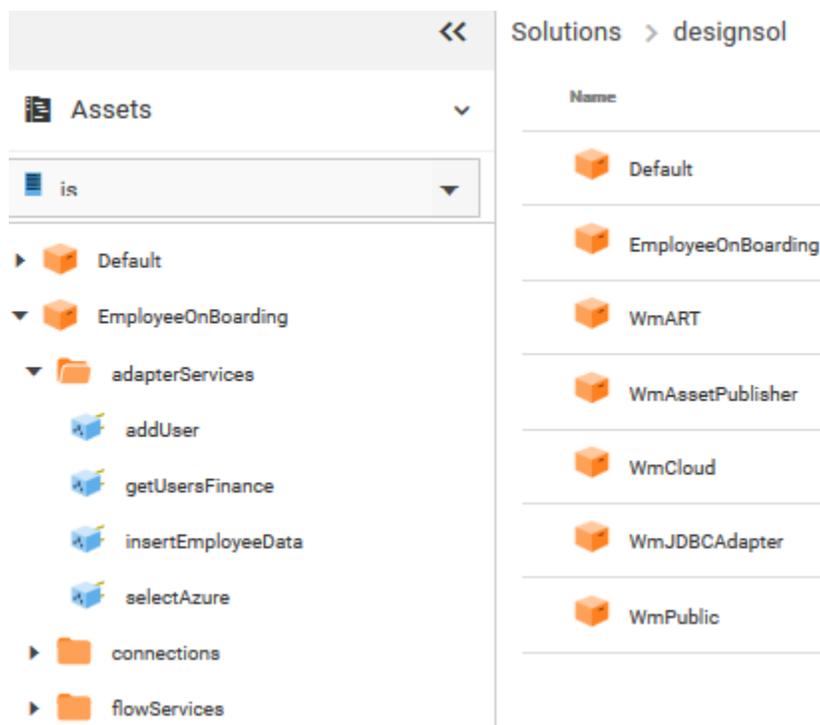
A package is a container that is used to bundle services and related elements, such as specifications, webMethods Integration Server document types, webMethods Integration

Server schemas, and triggers. When you create a folder, service, IS document type, or any element, you save it in a package.

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

- Administrators ACL
- Developers ACL
- Replicators ACL

The following figure depicts the package structure in a solution.



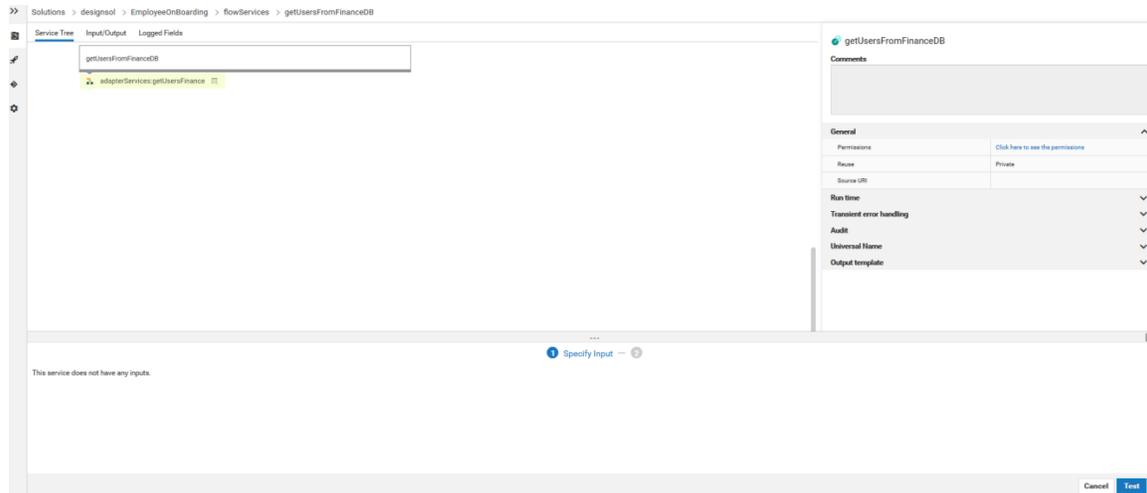
Packages are designed to hold all of the components of a logical unit in an integration solution. For example, you might group all the services and files specific to a particular marketplace in a single package. All the components that belong to a package reside in the package's subdirectory.

Note: Click the arrow beside a folder or package to view its contents.

Services

Services are method-like units of logic that operate on documents. You build services to carry out work such as extracting data from documents, interacting with back-end resources (for example, submitting a query to a database or executing a transaction on a mainframe computer), and publishing documents. Adapters and other add-on

packages provide additional services that you use to interact with specific resources or applications. The service editor allows you to view and run the services.



Service Signature

Input and output parameters are the names and types of fields that the service requires as input and generates as output. These parameters are also collectively referred to as a *signature*. You declare a signature for all types of services: flow services, Java services, and services written in other supported programming languages.

For a flow service, the input side describes the initial contents of the pipeline. In other words, it specifies the variables that this flow service expects to find in the pipeline at run time. The output side identifies the variables produced by the flow service and returned to the pipeline. An webMethods Integration Server document type can also be used to define the input or output parameters for a service.

Click **Test** to run the service after providing the data to pass into the service.

Service Editor

Use the service editor to view the services. The source code, properties, inputs, and outputs are read only. The editor has the following tabs:

- **Source** tab contains the code or flow for the service.
- **Input/Output** tab contains the input and output signature of the service.
- **Logged Fields** tab indicates the input and output parameters for which the data is logged. You define the data to pass into the service by defining the input parameters on the lower panel of the editor.

Note: Integration Cloud Integrations can now invoke Cloud Deployment webMethods Integration Server [“services” on page 210](#) for the same tenant. A new pre-defined Application, [“Cloud Deployment” on page 112](#), is added in Integration Cloud. Using this Application, you can select the solution webMethods Integration Server services that you want to call from Integration Cloud.

Note: See the *webMethods Service Development Help*, *webMethods Integration Server Administrator's Guide*, and the *webMethods Adapter for JDBC Installation and User's Guide* for detailed descriptions of all the services and document types including Adapter services.

Deploy

After publishing the assets and configurations that reside within on-premises runtimes or repositories to webMethods Integration Cloud, you can promote them to different stages.

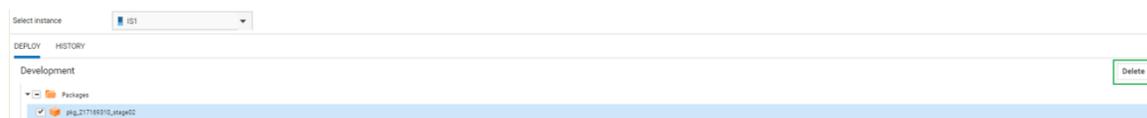
Select a solution and then click **Deploy**. Select a runtime instance. If you are in the **Development** stage, click the **Change Stage To View** link to change the stage. Only active and accessible stages appear in the drop-down list for selection in the **Stage to view** field. Select a different stage other than Development and click **Submit**.

Go back to the **Deploy** page for the selected solution. After you change the current stage to any other stage other than the Development stage, the **Deploy** page gets populated with the relevant assets.

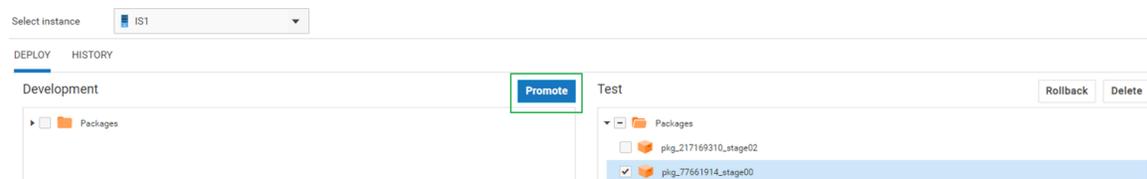
Note: You can access a stage only if your Access Profile is assigned to the stage.

On the **Deploy** page, you can delete an asset from the current stage (right panel). The asset will be deleted from the asset repository in that stage as well as from the runtime.

Note: Currently you can delete only webMethods Integration Server packages and not webMethods Integration Server and Universal Messaging configurations.



On the **Deploy** page, you can select an asset that was in the earlier stage (left panel), and click **Promote** to promote the package to the current stage (right panel).



After you click **Promote**, the **Promote Assets** dialog box appears for the selected asset.

Select an asset and change the values for the variable substitution properties, if needed. The variable substitution properties appear for the selected asset, only if the asset has properties. If there are any similar type of assets for which you want the same values, then select the **Show similar assets to apply values** option. Then select the assets in the lower panel. Click **Apply** to apply the property values to the selected assets. The changed values will be applied to all the selected similar assets during promotion.

On the **Promote** dialog box, you can type a message to describe the promotion. The promotion message will appear on the **History** page.

Click **Simulate Promote** to check the consistency of the assets and their dependencies. If there are dependencies, then for a successful promotion, you have to select all the dependent assets. Select **Save and Promote** to save the variable substitution and promote the assets to the next stage.

On the **Deploy** page, click **Rollback** to rollback *all* promoted assets to their previous state.

You can type a message to describe the rollback. The rollback message will appear on the **History** page.

History

The **History** page shows the **Trace ID**, that is, the tracking ID, which is automatically generated on every successful or unsuccessful promotion, rollback, or deletion, the Deployment, Rollback, or Deletion **Action**, **Date** when the asset was promoted, rolled back, or deleted, the **User** who promoted, rolled back, or deleted the asset, and the commit **Message** for the selected instance. You can click on a **Trace ID** to see the Track History for the specific action.

The **Track History** window displays the following details:

- **Timestamp** - The date and time when the log was generated.
- **Product** - The product that was promoted or rolled back.
- **Log Level** - Information on whether the log type is an Error, Info, or Debug.
- **Message** - Log or status message.

Note: Promotion, rollback, or deletion details appear only for the current stage.

Asset Repository

The **Asset Repository** page (**Cloud Deployment > Solutions > Asset Repository**) displays the list of all solutions and the user-created assets. You can also view the asset type, version of the assets, and services. Before deploying packages and configurations from Software AG Designer, you must create a solution in Integration Cloud to which you want to deploy the configuration assets. Software AG Designer deploys the assets and configurations to the Asset Repository in Integration Cloud.

Asset Repository for all solutions

The following figure depicts the Asset Repository structure for all solutions. You can only view the components in the Asset Repository, for all solutions.

Solution List Asset Repository		
Name	Type	Version
bankSol	Solution	
bankIS1	Integration Server	
bankum	Universal Messaging	
paySol	Solution	
payIS1	Integration Server	
payIS2	Integration Server	
payum	Universal Messaging	

Asset Repository for a selected solution

The **Asset Repository** page for a solution (**Cloud Deployment > Solutions > Select a Solution > Asset Repository**) displays the assets for the selected solution, including the asset type and version of the assets.

The following figure depicts the Asset Repository structure for a selected solution.

Solutions > bankSol		
Name	Type	Version
bankIS1	Integration Server	
bankum	Universal Messaging	

Manage

The **Manage** options allow you to view the solution landscape, configure webMethods Integration Server service access settings, administer the webMethods Integration Server, or restart the webMethods Integration Server instances.

Landscape

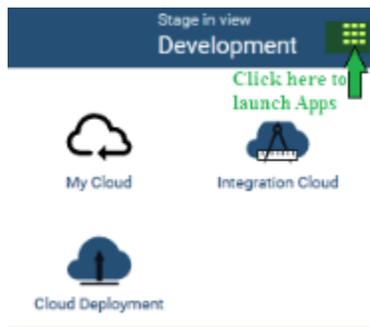
The **Landscape** page displays the landscape configuration for the selected solution.

Note: You cannot modify the solution landscape configuration in the Development stage after the solution is created. You can *configure* the solution in subsequent stages but after a solution is configured, the solution cannot be modified again in that stage. Further, clustering and Terracotta options are not available in the Development stage.

Note: In a stage, you can configure only those solutions that are marked as **Not Configured**.

To view the landscape configuration for a solution

1. Switch to the **Cloud Deployment** perspective.



- From the **Cloud Deployment** navigation bar, click **Solutions > Solution List > Select a solution > Manage > Landscape**.
- On the **Landscape** page, you can view the landscape configuration design, landscape solution name and description, and the landscape components. For each landscape component, you can view the landscape component name, product type, whether the landscape component is in a ready state, and the number of CPU cores and memory characteristics of the hardware to support each service in the solution.
- Terracotta is available only when webMethods Integration Server runs in a clustered mode. Further, clustering and Terracotta options are not available in the **Development** stage.

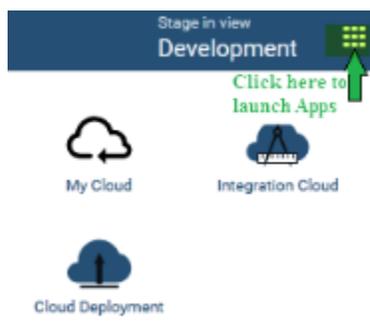
On the **Landscape** page, select the **Cluster Type** as **Stateless** if the group of webMethods Integration Servers function in a manner similar to a cluster but are not part of a configured cluster. A stateless cluster of webMethods Integration Servers does not use a Terracotta Server Array. Select **Stateful** to add the Terracotta section. The Terracotta icons will be activated.

Service Access Settings

The **Service Access Settings** page allows you to configure the webMethods Integration Server services to be called externally over HTTPS. The services will be available to consumers based on the **Allow All** and **Deny All** access modes.

To configure service access settings

- Switch to the **Cloud Deployment** perspective.



- From the **Cloud Deployment** navigation bar, click **Solutions > Solution List > Select a solution > Manage > Service Access Settings**.

- On the **Service Access Settings** page, configure the webMethods Integration Server services to be called externally over HTTPS. Required fields are marked with an asterisk on the screen.

Field	Description
Select Integration Server	Select the solution webMethods Integration Server instance where the services need to be configured.
Access Mode	<p>Ensure that the access mode of the services are properly set.</p> <p>Select Deny All if you want to deny most of the services to be invoked and allow a few. Then click ADD to add services to the Allowed Services table. In the Add Service window, the Base URL is a part of the complete service URL. In the Service URL field, type the webMethods Integration Server Service URL. The Base URL and the Service URL together forms the complete service URL.</p> <p>Select Allow All if you want to allow most of the services to be invoked and deny a few. Click ADD to add services to the Denied Services table. In the Add Service window, the Base URL is a part of the complete service URL. In the Service URL field, type the webMethods Integration Server Service URL. The Base URL and the Service URL together forms the complete service URL.</p> <div data-bbox="748 1367 1352 1499" style="background-color: #f0f0f0; padding: 5px;"> <p>Note:You can update the service URL by clicking the Edit icon beside the service URL in the services table.</p> </div> <p>The services will be available to be invoked from a software application, for example, a REST client, after you add the services to the table.</p> <p>For example, the Service URL <i>/invoke/pub.math:addInts</i> has the following components:</p> <ul style="list-style-type: none"> ■ Directive - <i>Invoke</i> ■ Namespace of the service - <i>pub.math:addInts</i>

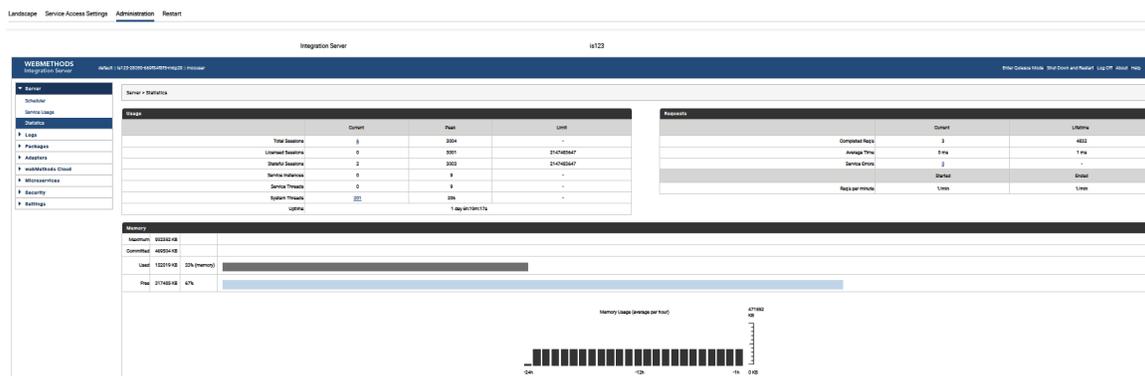
Field**Description**

Note: You can match all services to be allowed or denied by typing * at the end of the Service URL. For example, if you have two services, service url1: /invoke/pub.date:formatDate and service url2: /invoke/pub.date:getCurrentDate /invoke/, then instead of typing two entries, you can provide only one entry, service url: /invoke/pub.date:*. All services matching pub.date will be allowed or denied.

Administration

Use this page to manage a solution webMethods Integration Server Administrator instance.

The webMethods Integration Server Administrator is an HTML-based utility you use to administer the webMethods Integration Server. It allows you to monitor server activity, manage user accounts, make performance adjustments, and set operating parameters. You can run the webMethods Integration Server from any browser-equipped workstation on your network. When you click **Administration**, your browser displays the **Statistics** screen.

The webMethods Integration Server Administrator Screen

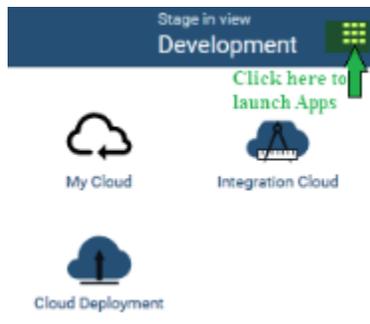
The Title bar displays the name of the host machine where webMethods Integration Server is running, the name of the webMethods Integration Server instance, and the name of the user currently logged into the webMethods Integration Server instance .

The Navigation panel on the left side of the page displays the names of menus from which you can select a task. To start a task, click a subject in the Navigation panel. The server displays a screen that corresponds to the task you select.

Note: Click **Help** to view the Help system, which provides information about the features and functionality of webMethods Integration Server.

To view the Administration page

1. Switch to the **Cloud Deployment** perspective.



2. From the **Cloud Deployment** navigation bar, click **Solutions > Solution List > Select a solution > Manage > Administration**.

The webMethods Integration Server Administrator page appears.

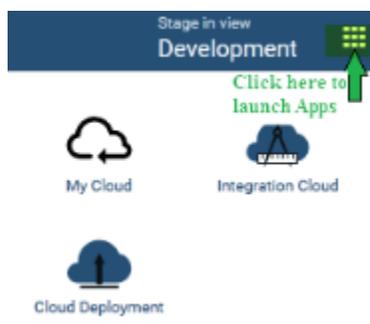
Restart

Restart the server when you need to stop and reload the server. You should restart the server when:

- You make certain configuration changes. Some configuration changes require the server to be restarted before they take effect.
- If you encounter an operational problem or the server is in an inconsistent state.

To restart the server

1. Switch to the **Cloud Deployment** perspective.



2. From the **Cloud Deployment** navigation bar, click **Solutions > Solution List > Select a solution > Manage > Restart**.

Note: webMethods Integration Servers running in a clustered mode cannot be restarted. Further, restarting servers will terminate all active sessions.

3. Click **Restart**.

Asset Repository

The **Asset Repository** page (**Cloud Deployment > Solutions > Asset Repository**) displays the list of all solutions and the user-created assets. You can also view the asset type, version of the assets, and services. Before deploying packages and configurations from Software AG Designer, you must create a solution in Integration Cloud to which you want to deploy the configuration assets. Software AG Designer deploys the assets and configurations to the Asset Repository in Integration Cloud.

Asset Repository for all solutions

The following figure depicts the Asset Repository structure for all solutions. You can only view the components in the Asset Repository, for all solutions.

Solution List Asset Repository		
Name	Type	Version
bankSol	Solution	
bankIS1	Integration Server	
bankum	Universal Messaging	
paysol	Solution	
payis1	Integration Server	
payis2	Integration Server	
payum	Universal Messaging	

Asset Repository for a selected solution

The **Asset Repository** page for a solution (**Cloud Deployment > Solutions > Select a Solution > Asset Repository**) displays the assets for the selected solution, including the asset type and version of the assets.

The following figure depicts the Asset Repository structure for a selected solution.

Solutions > bankSol		
Name	Type	Version
bankIS1	Integration Server	
bankum	Universal Messaging	

Capability

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

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

Field	Description
Allowed application count	Total number of Applications that can be utilized by the tenant.

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

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

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

Usage

The **Usage** page allows you to view the current usage of CPU cores and Memory (GB) for all the active solutions in all the stages.

To access this page, from the Cloud Deployment navigation bar, click  and select *Licensing > Usage*.

Monitoring

The Monitoring part of **Cloud Deployment** enables you to monitor the health and availability of the solutions and run-time instances, alerts and alert statuses. You receive an email whenever there is a condition that might affect the solution.

The monitoring of a new solution starts automatically 10 minutes after the creation of the solution. The data of the solution is collected and analyzed every 60 seconds.

You can access the following monitoring pages from the left-side navigation menu of the Monitoring main page:

- [“Dashboard” on page 471](#)
- [“Solutions” on page 472](#)
- [“Runtimes” on page 473](#)
- [“Services ” on page 475](#)
- [“Uptime” on page 476](#)
- [“Alerts” on page 477](#)
- [“Logs” on page 480](#)

You can filter the information on most Monitoring pages based on time. To specify the time-range, select a value in the time-range selector.

The following table describes the options in the time-range selector.

Option	Description
12h	Displays the information for the last 12 hours.
24h	Default. Displays the information for the last 24 hours.
2d	Displays the information for the last 2 days.
1w	Displays the information for the last week.
2w	Displays the information for the last 2 weeks.

Option	Description
3w	Displays the information for the last 3 weeks.
4w	Displays the information for the last 4 weeks.

To navigate to the Monitoring main page, log in to Integration Cloud, switch to the **Cloud Deployment** perspective, and select **Monitoring** in the Cloud Deployment navigation bar.

Dashboard

On the Dashboard page, you can view:

- The health of the solutions
- The number of the alerts that have been raised for all the solutions
- The landscape view of the solutions, and the number of alerts for all run-times that are part of the solutions

The following table provides more information about the panes on the Dashboard page.

Pane	Description
Overall KPI Status	<p>Shows the following information about the health of the solutions for the selected time range:</p> <ul style="list-style-type: none"> ■ Total number of solutions ■ The number of healthy solutions ■ The number of unhealthy solutions ■ The health of the solutions, as a percentage value calculated by the formula (Number of healthy solutions / total number of solutions) * 100 <p>A healthy solution is a solution without any open critical alerts.</p> <p>An unhealthy solution is a solution which has at least one open critical alert.</p> <p>To see more information about the KPI status of the solutions, click More Details.</p>
Alerts	<p>Shows the total number of open and resolved alerts that have been raised for all solutions for the selected time-range, and the number of alerts from each type: critical, warning, or information.</p>

Pane	Description
	To see more details about the alerts, click More Details . For more information about the alert types, see “Alert Types” on page 478 .
Landscapes	Displays the topology of the solutions. The number of alerts for each run-time type is displayed under the run-time type name.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see [“Monitoring” on page 470](#).

Solutions

On the Solutions page, you can check the health of the run-time instances from all the solutions. For each run-time instance, you can view the current data, and the data for the last 24 hours.

The health metrics are grouped into three categories:

- Memory - indicate the memory utilization of a run-time.
- Uptime - indicate the availability of a run-time.
- Failures- indicate failures of the run-time

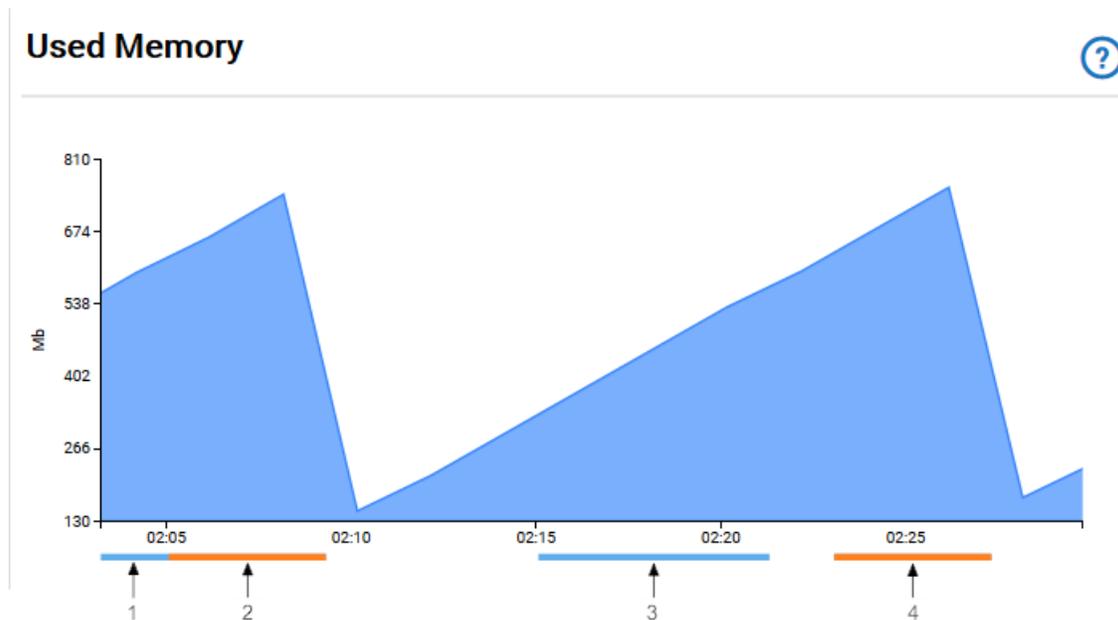
The following table describes the icons on the Solutions page.

Icon	Description
	Normal health of the run-time instance.
	The health of the run-time instance is deteriorating. Take preventive measures.
	There are critical issues with the health of the run-time instance. Your urgent attention is needed.
	The run-time instance is not available.
Note:	If the solution uses an Integration Server cluster, the number of Integration Server instances is indicated in brackets after the Integration Server instance name.

To view more details about a run-time instance on the Runtime page, click the name of the run-time instance.

Runtimes

On the Runtimes page, you can view the graphs for monitored KPIs for the selected run-time instances from all the solutions.



The example image shows the graph for the Used Memory KPI. The horizontal lines below the graph represent the severity and duration of the alerts that were raised for the KPI. The information alerts are displayed in blue, the warning alerts are in orange, and the critical alerts are in red.

The following table describes the meaning of the alert lines from the example graph for the Used Memory KPI.

Time Period	Details
1	Until 2:05 h, there had been an open information alert.
2	At 2:05 h, the severity of the information alert was changed to warning.
3	An information alert existed during that period.
4	A warning alert existed during that period.

You can change the value in the **Solutions** drop-down field to load the information about the run-time instances from a specific solution.

You can use the **INTEGRATION SERVER**, **UNIVERSAL MESSAGING**, and **TERRACOTTA** tabs to view the information related to the selected solution and runtime.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see [“Monitoring” on page 470](#).

The following table describes the monitored Integration Server KPIs.

Name	Description
Used Memory	The total used memory for the Java VM.
Service Threads	The number of active service threads.
Sessions	The number of active licensed sessions.
Stateful Sessions	The number of the current stateful HTTP sessions.

The following table describes the monitored Universal Messaging KPIs.

Name	Description
Free Memory	The amount of free memory that the Realm Server has within the Java VM. This indicates the difference between what the Java VM has currently allocated and what the Realm Server has used.
Published Events	Total number of events published on this realm from the time it started.
Subscribed Events	Total number of events that this realm has sent to clients from the time it started.

The following table describes the monitored Terracotta KPIs.

Name	Description
Off-Heap Used Memory	Shows the amount of off-heap memory that is currently used.
Live Objects	Shows the total number of live objects in the cluster, mirror group, server, or clients.

Name	Description
	If the trend for the total number of live objects goes up continuously, clients in the cluster will eventually run out of memory and applications might fail. Upward trends indicate a problem with application logic, garbage collection, or the tuning of one or more clients.

Viewing Adapter KPIs

On the **INTEGRATION SERVER** tab on the Runtimes page, you can view KPIs for adapters that are installed on an Integration Server instance.

1. Navigate to the Runtimes page.
2. Select a solution.
3. On the **INTEGRATION SERVER** tab, select an Integration Server instance.
4. Click **Show Adapter KPIs**.
5. Select an Adapter.

The Adapter KPIs are displayed. Currently you can monitor only KPIs for webMethods Adapter for JDBC.

The following table describes the monitored Adapter KPIs.

Name	Description
Connections	The number of connection pools in the adapter and how many of them are currently enabled.
Notifications	The number of adapter notifications (polling notifications) and how many of them are currently enabled.

Note: You can view Adapter KPIs only for the current time.

Services

On the Services page, you can view the number of successful and failed service executions of the Integration Server instances from the solutions.

The Services page consists of two the Service Executions pane and the History pane.

Pane	Description
Service Executions	<p>Shows the following information about the service executions of the Integration Server instances for the selected time range:</p> <ul style="list-style-type: none"> ■ Total number of service executions ■ The number of successful service executions ■ The number of failed service executions ■ The successful service execution, as a percentage value calculated by the formula (Number of successful service executions / total number of service executions) * 100

History	<p>Shows a chart with the history of successful (green) and failed (red) service executions.</p> <p>Hovering over the green and red bars displays the total number of successful and failed service executions, correspondingly.</p>
---------	--

Important: Only Integration Server services that are enabled for audit logging are included in the statistics on the Services page. You enable Integration Server services for audit logging in Software AG Designer, as described in "Configuring Service Auditing" in *webMethods Service Development Help*.

You can change the value in the **Solutions** drop-down field to view the information about a specific solution, or the information for all solutions.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see "[Monitoring](#)" on page 470.

Uptime

On the Uptime page, you can view time lines that represent the availability of all run-time instances of the solutions.

The color of the time lines changes based on the status of the run-time instances.

The following table describes the meaning of the different colors.

Time line color	Indicates that the run-time instance
green	was available during the indicated time period.

Time line color	Indicates that the run-time instance
red	was unavailable during the indicated time period.
grey	did not exist during the indicated time period.
Note:	If the solution uses an Integration Server cluster, the number of Integration Server instances is indicated in brackets after the Integration Server instance name.

By default, the time line displays the availability of the instances during the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see [“Monitoring” on page 470](#).

Alerts

On the Alerts page you can:

- View the number of critical, warning, and information alerts for all the solutions for the selected time range
- Filter the alerts by solution, runtime, severity, and status
- Change the default alert threshold and the recipient email. For more information about configuring the threshold values of the alerts, see [“Configuring the Alerts” on page 479](#).

By default, the Alerts page displays the number of alerts (critical, warning, and information) for all the solutions, and detailed information about the alerts in a tabular format.

The following table describes the information that is displayed in the table on the Alerts page.

Column	Description
Solution	Name of the solution.
Runtime	Runtime type. <ul style="list-style-type: none"> ■ Integration Server ■ Universal Messaging ■ Terracotta
Instance	Name of the run-time instance.
Date	Date and time when the alert was raised.

Column	Description
Message	Description of the alert.
Status	Status of the alert. <ul style="list-style-type: none"> ■  The alert is inactive. ■  The alert is active.

Note: The Alerts page might not display the alerts for all nodes from a cluster. For example, if you monitor an Integration Server cluster with two Integration Server instances, and both instances have alerts for the same property with different severity, the Alerts page will show the alert of lower severity only, as explained in the following table.

Integration Server instance	Alert type	Visibility on the Alerts page
Integration Server instance 1	Information. Free memory is low.	Yes
Integration Server instance 2	Warning. Free memory is low.	No

You can view all alerts for all the nodes from the cluster in the email alerts.

By default, the page displays information for the last 24 hours. To view the information for a different time period, use the time-range selector. For more information about the time-range selector, see [“Monitoring” on page 470](#).

Alert Types

The following table provides more information about the alert types.

Alert Severity	Description	Color Coding
Critical	A condition exists that is critical for the system performance.	red
Warning	A condition exists that might deteriorate the system performance.	orange
Information	A condition exists that might evolve into a warning or critical alert.	blue

Configuring the Alerts

You can change the default threshold values and the recipient email for the system alerts. Threshold values determine when the system raises an alert for the rule.

To configure the system alerts

1. Navigate to the Alerts page.
2. Select the **Configuration** tab.

The Configuration page shows information about the alerts for all solutions. The following table describes the columns on in the form.

Column	Description
Name	Alert Name.
Runtime	Integration Server, Universal Messaging, or Terracotta.
Action	The icon activates the configuration view for the alert.

3. Click the **Configuration** icon in the Action column for the alert that you want to configure.

A form with the configuration details for the alert rule is displayed. The following table describes the fields in the form.

Field	Description
Threshold	<p>The threshold values determine when the system sends the alerts.</p> <p>Drag the ends of the corresponding colored lines to change the threshold values for the following alert types:</p> <ul style="list-style-type: none"> ■ For Critical alerts, adjust the ends of the red line. ■ For Warning alerts, adjust the ends of the orange line. ■ For Information alerts, adjust the ends of blue line.
Runtime	Integration Server, Universal Messaging, or Terracotta.
Summary	Description of the alert.
Interval	Time interval at which the rule is evaluated (either 5, or 60 seconds). Read-only.

Field	Description
Email on alert	<p>Email of the user who will receive the alerts.</p> <p>Note: The email is used for all rules. In case there are alerts, the system sends emails once every 10 minutes.</p> <p>To configure more than one email, use comma-separated values.</p>

4. In the **Threshold** field, change the default threshold values for the alert.
5. In the **Email on Alert** field, type the email(s) of the user(s) who will receive the email alerts for all rules.
6. Click **Apply**.

Logs

On the Logs page, you can view the logs of the run-time instances in the solutions for a selected time period.

To view the logs for all the instances from a specific solution, select the solution in the solutions drop-down list.

To view the logs for a specific run-time instance in a solution, select the run-time instance in the run-time instance drop-down list.

By default, the page displays the logs for the last 24 hours. To view the logs for a different time period, use the time-range selector. For more information about the time-range selector, see [“Monitoring” on page 470](#).

You can view logs message statistics and logs details, such as log timestamp and message text. You can change the type of log details that you see by adding or editing filters as described in the Kibana documentation.