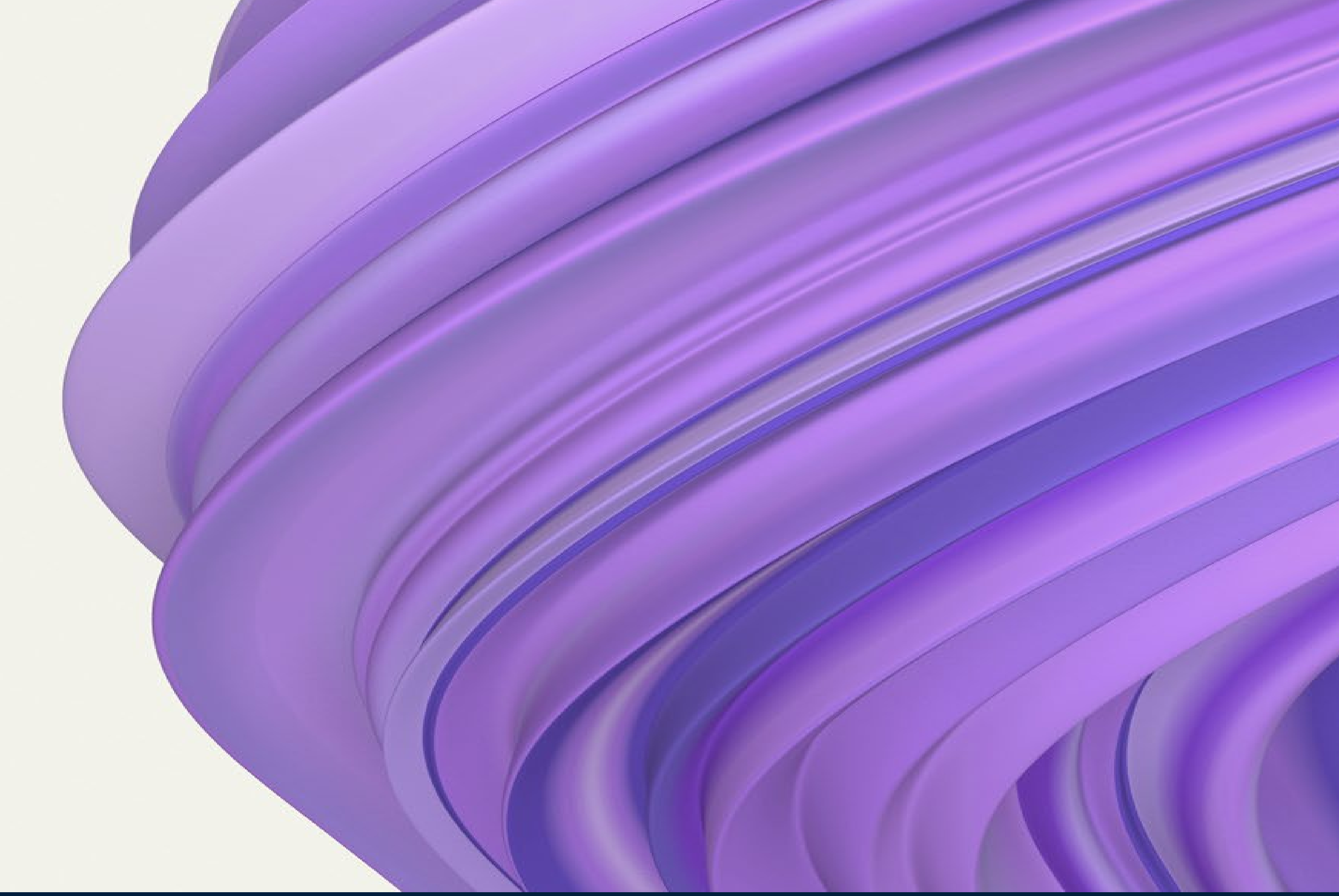


WEBMETHODS CLOUDSTREAMS API REFERENCE GUIDE

October 2022

Contents

1 Overview

CloudStreams provides a robust set of APIs that allow developers to programmatically integrate their own applications with CloudStreams in a simple and secure manner. CloudStreams APIs permit access to CloudStreams functionalities and data in an authorized way to accomplish integration of systems and applications. The APIs are based on the REST architectural style and can be communicated via HTTP requests.

2 Authentication

CloudStreams uses the HTTP basic authentication method to authenticate each HTTP request made to the API.

The basic authentication scheme used for CloudStreams API is the same as Integration server. This authentication method verifies the API consumer’s authentication credentials contained in an Authentication Header against the list of users registered in the Integration Server on which CloudStreams is running.

Note: Currently, CloudStreams uses the HTTP basic authentication method to authenticate each HTTP request made to the API. We will introduce new authentication mechanisms in a future release.

3 CloudStreams APIs

CloudStreams APIs are categorized into the following domains:

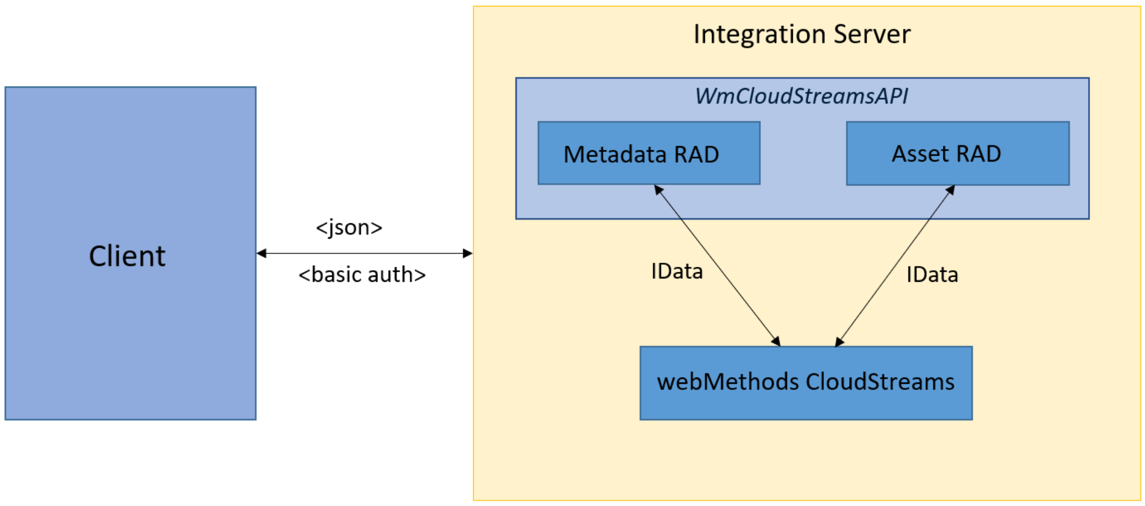
API Category	**Used for…**
Metadata APIs	Querying the platform for available connectors, operations, authentication schemes, lookups to use connectors.
Admin APIs	Managing connectors and other administrative operations.
Asset Management APIs	Managing customer-defined assets like accounts (connections), services (operations), listeners, and other relevant assets.
Runtime APIs	Managing runtime state of connections, listeners, analytics, runtime logs, and executing connector services.

CTK APIs	Connector development

4 Architectural approach for CloudStreams APIs

The architectural style followed for exposing CloudStreams APIs is *Rapid Application Development (RAD).* A separate RAD service is designed for each category of CloudStreams APIs within the webMethods Integration Server. The RAD service is developed using the OpenAPI specification. Inside the *Integration Server*, each RAD service resides in a common hidden package called *WmCloudStreamsAPI* and internally communicates with *webMethods CloudStreams* via *IData*.

The following diagram shows a high-level overview of the architectural approach followed to expose CloudStreams API.



5 Product Setup

Currently, we have introduced three API categories – *Metadata APIs, Asset Management APIs, and Runtime APIs* of CloudStreams APIs. We will upgrade the feature in the upcoming releases.

The download URL for OpenAPI specification for each category of CloudStreams APIs is provided in the respective sections.

5.1 Metadata APIs

Use Metadata APIs to query connectors and CloudStreams server metadata.

The download URL for OpenAPI specification for Metadata APIs has the following format: *http//:<integration_server_hostname:port>rad/wm.c10s.api:metadata?openapi.yaml*

5.1.1.1 List all connectors

Allows you to retrieve a list of all connectors from the Integration Server.
Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the provider, for which you want to retrieve the connectors.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors* path.

   Path: /metadata/connectors

   Method: GET

   Query Parameter:

Name	Required	Description
includeDisabled	false	Set this query parameter to *true* if you want to display disabled connectors in the response.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "version": "string",
    "packageName": "string",
    "description": "string",
    "connectorGroupId": "string",
    "type": "string",
    "providerName": "string",
    "versions": [
      {
        "connectorGroupVersion": "string",
        "id": "string",
        "description": "string",
        "latest": true
      }
    ]
  }
]
```

5.1.1.2 List all connection types

Allows you to retrieve a list of all supported connection types for a specific connector.
Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector, for which you want to retrieve the connection types.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/connections* path.

   Path: /metadata/connectors/{connectorId}/connections

   Method: GET

   Path Parameter:

Name	Required	Description
connectorId	true	ID of the connector

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "type": "string",
    "authenticationType": "string",
    "testable": true
  }
]
```

5.1.1.3 Fetch connection metadata for a specific type

Allows you to retrieve metadata of a connection for the specified connection type.
Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector, to retrieve the connection metadata.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/connections/{connectionType}* path.

Path: /metadata/connectors/{connectorId}/connections/{connectionType}

Method: GET

Path Parameters:

Name	**Required**	**Description**
connectorId	true	ID of the connector
connectionType	true	Connection type to fetch the metadata

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "providerName": "string",
  "connectorID": "string",
  "connectionType": "string",
  "authenticationScheme": "string",
  "connectionGroups": [
    {
      "groupType": "string",
      "groupDisplayName": "string",
      "fields": [
        {
          "displayName": "string",
          "defaultValue": "string",
          "propertyValue": "string",
          "schemaType": "string",
          "propertyKey": "string",
          "isEncrypted": true,
          "isHidden": true,
          "isRequired": true,
          "isArray": true,
          "modelType": "string",
          "isContextProperty": true,
          "isOverride": true,
          "allowedValues": [
            "string"
          ]
        }
      ]
    }
  ]
}
```

5.1.1.4 List all domains for a connector

Allows you to retrieve a list of all domains for a specific connector.
Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector, to retrieve a list of all domains.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains* path.

   Path: /metadata/connectors/{connectorId}/domains

   Method: GET

   Path Parameter:

Name	Required	Description
connectorId	true	ID of the connector to fetch domains.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "displayName": "string"
  }
]
```

5.1.1.5 Retrieve basic information of all REST Interactions

Allows you to retrieve basic information of all REST Interactions for a specific domain.
Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the basic information of all REST Interactions for a specific domain.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions

   Method: GET

   Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector to fetch Interactions
domainId	true	ID of the domain for which you want to fetch Interactions
connectionType	false	Interactions and their associated properties vary based on the connection type

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "displayName": "string",
    "description": "string",
    "type": "string",
    "method": "string",
    "path": "string"
  }
]
```

5.1.1.6 Fetch all headers for a REST Interaction

Allows you to retrieve a list of all headers for a specific REST Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve a list of all headers, for a specific REST Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/headers* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/headers

   Method: GET

   Path Parameters:

Name	**Required**	**Description**
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction. Domain is the parent group for Interactions.
context	false	The value for this field should be either IN or OUT.
connectionType	false	Headers will vary based on the connection type.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "defaultValue": "string",
    "required": true,
    "mandatory": true,
    "readOnly": true
  }
]
```

5.1.1.7 Fetch all parameters for a REST Interaction

Allows you to retrieve a list of all parameters for a specific REST Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve a list of all parameters associated with a specific REST Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/paramete*rs path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/parameters

   Method: GET

   Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch parameters.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "dataType": "string",
    "type": "string",
    "defaultValue": "string",
    "description": "string",
    "isRequired": true,
    "isActive": true,
    "isFixed": true,
    "isCustom": "string"
  }
]
```

5.1.1.8 Get the request/response body for a specific REST Interaction

Allows you to retrieve the request and/or response body for a specific REST Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the request and response body for a specific REST Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/body* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/body

   Method: GET

   Path Parameters:

Name	**Required**	**Description**
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch the request and/or response body.
type	true	Fetch either request or response body. By default, the request and response body will be returned.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "name": "string",
  "displayName": "string",
  "description": "string",
  "type": "string",
  "method": "string",
  "path": "string",
  "request": {
    "name": "string",
    "messageFormatterType": "string",
    "builderType": "string",
    "documentRef": "string",
    "excludeRoot": "string"
  },
  "responses": [
    {
      "name": "string",
      "code": "string",
      "messageFormatterType": "string",
      "builderType": "string",
      "documentRef": "string",
      "excludeRoot": "string"
    }
  ]
}
```

5.1.1.9 Get the abstract entity information for a specific REST Interaction

Allows you to retrieve the abstract entity information for a specific REST Interaction. This API is applicable for complex operations only.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the abstract entity information for a specific REST Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/abstract* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-rest/{interactionId}/abstract

   Method: GET

   Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch the abstract entity information.

Response:
If the request is successful, you will receive *the HTTP 200 OK* success status response code.

```
[
  {
    "path": "string",
    "context": "string",
    "entityType": "string",
    "lookupType": "string",
    "fixed": true,
    "defaultValue": "string"
  }
]
```

5.1.1.10 List all SOAP Interactions

Allows you to retrieve a list of all SOAP Interactions.
Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve a list of all SOAP Interactions.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap

   Method: GET

   Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "displayName": "string",
    "description": "string",
    "type": "string",
    "method": "string",
    "path": "string"
  }
]
```

5.1.1.11 Fetch all headers for a SOAP Interaction

Allows you to retrieve a list of all headers for a specific SOAP Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve a list of all headers for a specific SOAP Interaction.

```
Headers:
{
```

```
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/headers* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap

   Method: GET

   Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch headers.
context	false	The value for this field should be either IN or OUT

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
[
  {
    "name": "string",
    "defaultValue": "string",
    "required": true,
    "mandatory": true,
    "readOnly": true
  }
]
```

5.1.1.12 Fetch parameters for a SOAP Interaction

Allows you to retrieve a list of all parameters for a specific SOAP Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve a list of all parameters for a specific SOAP Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/parameters* path.

Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/parameters

Method: GET

Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch parameters.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "dataType": "string",
    "type": "string",
    "defaultValue": "string",
    "description": "string",
    "isRequired": true,
    "isActive": true,
    "isFixed": true,
    "isCustom": "string"
  }
]
```

5.1.1.13 Get the request/response body for a specific SOAP Interaction

Allows you to retrieve the request and/or response body for a specific SOAP Interaction.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the request and response body for a specific SOAP Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/body* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/body

   Method: GET

Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch the request and/or response body.
type	false	Fetch either request or response body. By default, the request and response body will be returned.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "name": "string",
  "displayName": "string",
  "description": "string",
  "type": "string",
  "input": {
    "uri": "string"
  },
  "output": {
    "uri": "string"
  }
}
```

5.1.1.14 Get the abstract entity information for a specific Interaction

Allows you to retrieve the abstract entity information for a specific SOAP Interaction. This API is applicable for complex operations only.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the abstract entity information for a specific SOAP Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/abstract* path.

   Path: /metadata/connectors/{connectorId}/domains/{domainId}/interactions-soap/{interactionId}/abstract

   Method: GET

Path Parameters:

Name	Required	Description
connectorId	true	ID of the connector. Connector ID is required to fetch any child information like Interaction.
domainId	true	ID of the domain. Domain is the parent group for Interactions.
interactionId	true	ID of the Interaction to fetch the abstract entity information.

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "path": "string",
    "context": "string",
    "entityType": "string",
    "lookupType": "string",
    "fixed": true,
    "defaultValue": "string"
  }
]
```

5.1.1.15 Get the schema for a specific URI

Allows you to retrieve the schema for a specific request/response reference.
Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the schema for a specific request/response reference.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/connectors/schema/{docTypeRef}* path.

   Path: /metadata/connectors/schema/{docTypeRef}

   Method: GET

   Path Parameter:

Name	Required	Description
docTypeRef	true	Document reference name to fetch the schema

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "schema": "string"
}
```

5.1.1.16 Get the schema for a specific document reference link

Allows you to retrieve the schema for a specific document reference.

Steps:

1. In a REST client platform, add the authentication details of Integration Server, to retrieve the schema for a specific document reference.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */metadata/global/schema/{docTypeRef}* path.

   Path: /metadata/global/schema/{docTypeRef}

   Method: GET

   Path Parameter:

Name	**Required**	**Description**
docTypeRef	true	Document reference name to fetch the schema

Response:
If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "schema": "string"
}
```

5.2 Asset Management APIs

Use Asset APIs to perform CRUD (create, read/retrieve, update, delete) operations on CloudStreams assets.

The download URL for OpenAPI specification for Asset APIs has the following format:
http//:<integration_server_hostname:port>rad/wm.c10s.api:asset?openapi.yaml

5.2.1.1 List all connections for a specific connector

Allows you to retrieve a list of all connections associated with a connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector, for the connections you want to retrieve.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/connections* path.

Path: /asset/connections

Method: GET

Query Parameters:

Name	Required	Description
providerName	false	Provider name associated with the connector for the connections you want to retrieve.
connectorId	false	ID of the connector to fetch connections.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "connectionList": [
    {
      "name": "string",
      "state": "string",
      "authenticationScheme": "string"
    }
  ]
}
```

5.2.1.2 Create a connection for a specific connector

Allows you to create a new connection for a specific connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector to create a new connection.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *POST* request to the */asset/connections* path.

This endpoint allows you to create the connection asset.

Path: /asset/connections

Method: POST

Request Body (Required):

```
{
  "name": "string",
  "providerName": "string",
  "connectorId": "string",
  "connectionType": "string",
  "packageName": "string",
  "namespace": "string",
  "connectionProperties": {}
}
```

To retrieve input data for the *connectionProperties* object, do the following:

1. Perform a *GET* request to the connection metadata API - */metadata/connectors/{connectorId}/connections/{connectionType}*
2. Fetch the *propertyKey* and other *default values* from the response.
3. Add the values as key-value pairs to the *connectionProperties* object.

Sample request data:

```
{
      "connectorId": "",
      "providerName": "",
      "name": "soap_connector",
      "connectionType": "sagcloud",
      "description": "",
      "packageName":"Package name where the connection should be created",
      "namespace":"a.b.c", //folder hierarchy where connection asset will be created
      "connectionProperties": {
         "cn.providerUrl": "<server_URL>",
          "cr.username": "<user_name>",
          "cr.password": "<password>",
          "cr.authSchemeType": "",
          "cm.poolable": <boolean>,
          "cm.timeoutType": "",
          "cm.sessionExpiry": <integer_value>
        }
          }
```

Response:

If the request is successful, you will receive the *HTTP 201 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.3 Get the connection configuration for a given connection ID

Allows you to retrieve the connection configuration details for the specified connection ID.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the connection configuration details for the specified connection ID.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/connections/{connectionId}* path.

   Path: /asset/connections/{connectionId}

   Method: GET

Query Parameters:

Name	Required	Description
connectionId	true	ID of the connection. Connection ID is required to retrieve the connection configuration details.
providerName	false	Provider name associated with the specified connection.
connectorId	false	ID of the connector associated with the specified connection.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "connectionType": "string",
  "authenticationScheme": "string",
  "connectionGroups": [
    {
      "groupDisplayName": "string",
      "groupType": "string",
      "fields": [
        {
          "displayName": "string",
          "defaultValue": "string",
          "propertyValue": "string",
          "schemaType": "string",
          "propertyKey": "string",
          "isEncrypted": true,
          "isHidden": true,
          "isRequired": true,
          "isArray": true,
          "modelType": "string",
          "isContextProperty": true,
          "isOverride": true,
          "allowedValues":
          [
           "string"
          ]
        }
      ]
    }
  ]
}
```

5.2.1.4 Update an existing connection

Allows you to update an existing connection associated with a specific connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector for the connection you want to update.

```
Headers:
```

```
{
Authorization: BasicAuth
}
```

2. Perform a *PUT* request to the */asset/connections/{connectionId}* path.

   Path: /asset/connections/{connectionId}

   Method: PUT

   Request body (Required):

```
{
  "connectorId": "string",
  "connectionType": "string",
  "connectionProperties": {}
}
```

   Query parameter:

Name	Required	Description
connectionId	true	ID of the connection you want to update.

To retrieve input data for the *connectionProperties* object, do the following:

1. Perform a *GET* request to the connection asset API - */asset/connections/{connectionId}*.
2. Fetch the *propertyKey* and other *default values* from the response.
3. Add the values as key-value pairs to the *connectionProperties* object.

Sample request data:

```
{
      "connectorId": "",
      "providerName": "",
      "connectionType": "<connection type can be fetched from the retrieve call>",
      "description": "",
      "packageName":" Package name where the connection should be created",
      "namespace":"a.b.c", //folder hierarchy where the connection asset will be created
      "connectionProperties": {
         "cn.providerUrl": "<server_URL>",
          "cr.username": "<user_name>",
          "cr.password": "<password>",
          "cr.authSchemeType": "",
          "cm.poolable": <boolean>,
          "cm.timeoutType": "",
          "cm.sessionExpiry": <integer_value>
        }
          }
```

Response Body:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.5 Delete a specific connection

Allows you to delete a specific connection associated with a connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector for the connection you want to delete.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *DELETE* request to the */asset/connections/{connectionId}* path.

Path: /asset/connections/{connectionId}

Method: DELETE

Query parameters:

Name	**Required**	**Description**
connectionId	true	ID of the connection you want to delete.
providerName	true	Provider name associated with the specified connection.
connectorId	true	ID of the connector associated with the specified connection.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.6 Update the status of a connection

Allows you to either enable or disable an existing connection associated with a specific connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector for the connection you want to enable or disable.

```
Headers:
{
```

```
Authorization: BasicAuth
}
```

2. Perform a *PATCH* request to the */asset/connections/{connectionId}/state* path.

   Path: /asset/connections/{connectionId}/state

   Method: PATCH

   Request Body (Required):

```
{
  "providerName": "string",
  "connectorId": "string",
  "enable": true
}
```

Query parameter:

Name	Required	Description
connectionId	true	ID of the connection for the state you want to update.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.7 Fetch all services for a connector

Allows you to retrieve a list of all services associated with a specific connector.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector for the services you want to retrieve.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/services* path.

   Path: /asset/services

   Method: GET

   Query parameters:

Name	Required	Description

providerName	false	Provider name associated with the connector to fetch services.
connectorId	false	Connector ID to fetch services.
packageName	false	Package name where the services exist.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "serviceList": [
    {
      "name": "string",
      "nsName": "string",
      "displayName": "string",
      "description": "string"
    }
  ]
}
```

5.2.1.8 Create a custom operation

Allows you to create a custom operation for a specific Interaction.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to create a custom operation for a specific Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *POST* request to the */asset/services* path.

   Path: /asset/services

   Method: POST

   Request Body (Required):

```
{
  "name": "string",
  "displayName": "string",
  "description": "string",
  "packageName": "string",
  "namespace": "string",
  "providerName": "string",
  "connectorId": "string",
  "domainId": "string",
  "interactionId": "string",
  "connectionType": "string",
  "request": {
    "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "parameters": [
      {
        "name": "string",
        "dataType": "string",
        "type": "string",
        "defaultValue": "string",
        "description": "string",
        "isRequired": true,
        "isActive": true,
        "isFixed": true,
        "isCustom": "string"
      }
    ],
    "body": {}
  },
  "response": {
    "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "body": {}
  },
  "entity": {
    "name": "string",
    "fields": [
      {
        "name": "string",
        "dataType": "string",
        "required": true,
        "custom": true
      }
    ],
    "relatedEntities": [
      {
        "name": "string",
        "fields": [
          {
            "name": "string",
            "dataType": "string",
            "required": true,
            "custom": true
          }
        ],
        "relatedEntities": [
          "string"
        ]
      }
    ]
  },
  "customObject": [
    {
      "name": "string"
    }
  ]
}
```

Query parameter:

Name	Required	Description
connectionId	false	ID of the connection to create a custom operation.

To create a custom operation, do the following:

1. Create a connection for the respective connector.
2. Fetch Interaction details from metadata APIs.

Notes:

- If the Interaction type is *Simple*, create the custom operation by passing the *Interaction ID* and *Connection details*.
- If the Interaction type is *Complex*, you will need to retrieve the *abstract information* for the *Interaction*.
- If the Interaction type is *businessObject*, retrieve the *schemaObjects* from the respective asset API and prepare the *customObjects* list by adding the name of *schemaObjects*.
- *entity* and *customObject* are related to *Complex* type operations only.
- *entity* and *customObject* do not exist in the same context for any Interaction.
- In the *release 10.15*, *relatedEntities* is not supported.

Sample request data:

```
{
  "name": " ",
  "displayName": "Saleforce REST operation creation",
  "description": "Create resource for SF",
  "connectorId": "<connector_id>",
  "providerName": "<Provider name>",
  "interactionId": "<Interaction_name>",
  "description": "My custom operation for Salesforce",
  "domainId": "<domainId from the connector>",
  "packageName":" <Package name where the asset should be created>",
  "namespace":"<folder hierarchy where the asset should be created>",
  "request": {},
  "response": {},
   "entity":
         {
         "name": "Account",
         "fields": [
                   {
                   "name": "Name",
                   },
                   {
                   "name": "Type"
                   },
                   {
                   "name":"BillingStreet"
                   }
                   ]
         },

         "customObject": ["<name>"]
    }
```

Response:

If the request is successful, you will receive the *HTTP 201 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.9 Fetch custom operation data

Allows you to retrieve data associated with a specific service.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve data associated with a specific service.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/services/{serviceId}* path.

   Path: /asset/services/{serviceId}

   Method: GET

   Query parameters:

Name	Required	Description
serviceId	true	ID of the service for the data you want to retrieve.
connectionId	true	ID of the connection you want to update.
providerName	false	Provider name associated with the specified service.
connectorId	false	Connector ID for the specified service.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "name": "string",
  "displayName": "string",
  "description": "string",
  "providerName": "string",
  "connectorId": "string",
  "domainId": "string",
  "interactionId": "string",
  "request": {
  "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "parameters": [
      {
        "name": "string",
        "dataType": "string",
        "type": "string",
        "defaultValue": "string",
        "description": "string",
        "isRequired": true,
        "isActive": true,
        "isFixed": true,
        "isCustom": "string"
      }
    ]
  },
  "response": {
   "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "body": {}
  },
  "entity": {
    "name": "string",
    "displayName": "string",
    "path": "string",
    "fields": [
      {
        "name": "string",
        "displayName": "string",
        "dataType": "string",
        "required": true,
        "relationship": {
        "name": "string",
        "relatesTo": "string",
        "type": "string"
        }
      }
    ],
    "relatedEntities": [
      {
        "name": "string",
        "fields": [
          {
            "name": "string",
            "dataType": "string",
            "required": true,
            "custom": true
          }
        ],
        "relatedEntities": [
        "string"
        ]
      }
    ]
  },
  "customObject": [
    {
      "name": "string"
    }
  ]
}
```

5.2.1.10 Update an existing custom operation

Allows you to update an existing service associated with a specific Interaction.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to update an existing service associated with a specific Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *PUT* request to the */asset/services/{serviceId}* path.

   Path: /asset/services/{serviceId}

   Method: PUT

   Request Body (Required):

```
{
  "interactionId": "string",
  "displayName": "string",
  "description": "string",
  "request": {
    "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "parameters": [
      {
        "name": "string",
        "dataType": "string",
        "type": "string",
        "defaultValue": "string",
        "description": "string",
        "isRequired": true,
        "isActive": true,
        "isFixed": true,
        "isCustom": "string"
      }
    ],
    "body": {}
  },
  "response": {
   "headers": [
      {
        "name": "string",
        "defaultValue": "string",
        "required": true,
        "mandatory": true,
        "readOnly": true
      }
    ],
    "body": {}
  },
  "entity": {
  "name": "string",
  "fields": [
      {
        "name": "string",
        "dataType": "string",
        "required": true,
        "custom": true
      }
    ]
  },
  "customObject": [
    {
      "name": "string"
    }
  ]
}
```

Query parameters:

Name	Required	Description
serviceId	true	ID of the service you want to update.
connectionId	true	ID of the connection.

To update a custom operation, do the following:

1. Have a valid connection with the respective connector.
2. Fetch Interaction details from metadata APIs.

Notes:

- If the Interaction type is *Simple*, create the custom operation by passing the *Interaction ID* and *Connection details*.
- If the Interaction type is *Complex*, you will need to retrieve the *abstract information* for the *Interaction*.
- If the Interaction type is *businessObject*, retrieve the *entities* and *entity fields* from the respective APIs and then prepare the *entity object* in the *signature*.
- You cannot edit the name of a custom operation. You can modify all other details of a custom operation.
- *entity* and *customObject* are related to *Complex* type operations only.
- *entity* and *customObject* do not exist in the same context for any Interaction.
- In the *release 10.15*, *relatedEntities* is not supported.

Sample request data:

```
{
    "displayName": "Salesforce REST operation creation",
    "description": "Create resource for Salesforce",
    "connectorId": "<connector_ID>",
    "providerName": "<Provider name>",
    "interactionId": "<Interaction_name>",
    "description": "My custom operation for Saleforce",
    "domainId": "<Domain ID from the connector>",
    "packageName":" <Package name where the asset should be created>",
    "namespace":"<folder hierarchy where the asset should be created>",
    "request": {},
    "response": {},
    "entity": {
                    "name": "Account",
                    "fields": [
                                {
                                "name": "Name",
                                },
                                {
                                "name": "Type"
                                },
                                {
                                "name":"BillingStreet"
                                }
                                ]
          },

          "customObject": ["<name>"]
    }
```

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.11 Delete the custom operation

Allows you to delete data associated with a specific service.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to delete data associated with a specific service.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *DELETE* request to the */asset/services/{serviceId}* path.

Path: /asset/services/{serviceId}

Method: DELETE

Query parameters:

Name	**Required**	**Description**
serviceId	true	ID of the service for the data you want to delete.
providerName	false	Provider name associated with the specified service.

connectorId	false	Connector ID associated with the specified service.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "status": "string",
  "assetId": "string",
  "resource": "string"
}
```

5.2.1.12 Get the input/output signature for a specific service

Allows you to retrieve the schema associated with a specific service.

Steps:

1. In a REST client platform, add the authentication details of Integration Server associated with the connector for the connection you want to update.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/services/{serviceId}/signature/schema* path.

   Path: /asset/services/{serviceId}/signature/schema

   Method: GET

   Query parameters:

Name	**Required**	**Description**
serviceId	true	ID of the service for the schema you want to retrieve.
context	true	Context value should be IN for input signature and OUT for output signature.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "schema": "string"
}
```

5.2.1.13 Get the input/output signature data for a specific service

Allows you to retrieve the input and output signature data for a specific service.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the input and output signature data for a specific service.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/services/{serviceId}/signature/data* path.

   Path: /asset/services/{serviceId}/signature/data

   Method: GET

   Query parameters:

Name	Required	Description
serviceId	true	ID of the service for the input/output signature data you want to retrieve.
context	true	Context value should be IN for input signature and OUT for output signature.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "jsonString": "string"
}
```

5.2.1.14 Get the entities for a specific Interaction

Allows you to retrieve the entities for a specific Interaction.

Note: This API is applicable for *Complex* type Interactions only. The abstract type should be *businessObject*. You can retrieve the details from the metadata APIs.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the entities for a specific Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/entities* path.

   Path: /asset/entities

   Method: GET

   Query parameters:

Name	Required	Description
connectionId	true	ID of the connection. The connection ID is required for retrieving the entity data from the back end.
connectorType	true	Connector type (REST/SOAP) is required for retrieving the entity metadata for a connector.
domainId	true	Domain ID is required for retrieving the Interaction metadata.
interactionId	true	Interaction ID is required for retrieving the entity information.
context	true	The value for this field should be either IN or OUT.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "entityList": [
    {
      "name": "string",
      "displayName": "string"
    }
  ]
}
```

5.2.1.15 Get the entity fields for a given entity name

Allows you to retrieve the fields for a specific entity.

Note: You can retrieve the entity fields, only if entities are already fetched. You can fetch the entities using the list entity API. Details can be retrieved from the metadata APIs.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the fields for a specific entity.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/entities/{entityName}* path.

Path: /asset/entities/{entityName}

Method: GET

Query parameters:

Name	Required	Description
entityName	true	Entity name is required for retrieving the entity fields from the back end.
connectionId	true	Connection ID is required for retrieving the entity data from the back end.
connectorType	true	Connector type is required for retrieving the entity metadata from a connector.
domainId	true	Domain ID is required for retrieving the Interaction metadata.
interactionId	true	Interaction ID is required for retrieving the entity information.
context	true	The value for this field should be either IN or OUT.
relationType	false	Relation type is required for retrieving the entity fields from related entities.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
[
  {
    "name": "string",
    "displayName": "string",
    "dataType": "string",
    "required": true,
    "relationship": {
    "name": "string",
    "relatesTo": "string",
    "type": "string"
    }
  }
]
```

5.2.1.16 Get schema objects for a specific Interaction

Allows you to retrieve schema objects for a specific Interaction.

Note: This API is applicable only for *Complex* type Interactions. The abstract type should be *schemaObject*. Details can be retrieved from the metadata APIs.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve schema objects for a specific Interaction.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/schemaObjects* path.

Path: /asset/schemaObjects

Method: GET

Query parameters:

Name	Required	Description
connectorId	true	Connector ID is required for fetching the schema objects.
domainId	true	Domain ID is required for retrieving the schema objects.
interactionId	true	Interaction ID is required for retrieving the schema objects.
context	true	The value for this field should be either IN or OUT.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "schemaObjects": [
    {
      "name": "string",
      "displayName": "string"
    }
  ]
}
```

5.3 Runtime APIs

Use the runtime APIs to execute a service and retrieve the current status of a connection.

The download URL for OpenAPI specification for Runtime APIs has the following format: *http//:<integration_server_hostname:port>rad/wm.c10s.api:asset?openapi.yaml*

5.3.1.1 Get the current status of a connection

Allows you to retrieve the current status of a specific connection.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to retrieve the current status of a specific connection.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *GET* request to the */asset/connections/{connectionId}/state* path.

Path: /asset/connections/{connectionId}/state

Method: GET

Query parameters:

Name	Required	Description
connectionId	true	ID of the connection. Connection ID is required to retrieve the status of a connection.
providerName	false	Provider name associated with the specified connection.
connectorId	false	Connector ID for the specified connection.

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "state": "string",
  "runtimeState": "string"
}
```

5.3.1.2 Execute a service

Allows you to execute a specific service.

Steps:

1. In a REST client platform, add the authentication details of Integration Server to execute a specific service.

```
Headers:
{
Authorization: BasicAuth
}
```

2. Perform a *POST* request to the */runtime/services/{serviceId}/execute* path.

Path: /runtime/services/{serviceId}/execute

Method: POST

Request Body (Required):

```
{
  "input": {}
}
```

Query parameters:

Name	Required	Description
serviceId	true	ID of the service you want to execute.
connectionId	true	ID of the connection associated with the service you want to execute.

To execute a custom operation, do the following:

1. Retrieve the signature data by performing the *GET* request to the *signature/data* API.
2. Add the signature data in the input request body.

Example:

```
//Example of an input body for executing a Salesforce query-related custom operation.

{
"input":
{
 "requestHeaders”:
{"tns:SessionHeader":{},"tns:QueryOptions":{"tns:batchSize":"500"}},"requestBody":{"tns:query":{}},
"$parameters":{}
}
```

Response:

If the request is successful, you will receive the *HTTP 200 OK* success status response code.

```
{
  "serviceId": "string",
  "response": {}
}
```