Oracle® Cloud

Using the Rapid Adapter Builder with Oracle Integration 3

F79578-06
February 2025

Oracle Cloud Using the Rapid Adapter Builder with Oracle Integration 3,

F79578-06

Copyright © 2024, 2025, Oracle and/or its affiliates.

Primary Author: Oracle Corporation

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Audience x
Documentation Accessibility x
Diversity and Inclusion x
Related Resources x
Conventions xi

Part I Get Started with the Rapid Adapter Builder

1 Learn About the Rapid Adapter Builder in Oracle Integration

What Is the Rapid Adapter Builder? 1-1
Reasons to Build an Adapter 1-1
Where Your Adapters Appear 1-2
How Your Adapters Differ from Oracle Adapters 1-3
Explore What You Can Do 1-4
- Features 1-5
- Security 1-6
- Extendability 1-7
- Limitations 1-8
- Limits 1-9

2 Use VS Code as an Integrated Development Environment

About the Adapter Definition Document 2-1
Data Mapping in the Adapter Definition Document 2-2
- Data from an OpenAPI Document 2-2
  - Info Component 2-3
  - Connection Component 2-4
  - Actions Component 2-6
  - Flows Component 2-8
  - Schemas Component 2-9
- Data from a Postman Collection 2-9

Info Component 2-10
Use Postman as an API-Test Environment 2-10
Explore the VS Code Extension's Features 2-11
Build an Adapter in an Hour 2-12

Part II Build an Adapter Using the Rapid Adapter Builder

3 Plan and Design an Adapter

Guidelines for Planning Your Adapter 3-1
Identify the Problem 3-1
Understand the Target Application 3-2
Define the Requirements 3-3
Best Practices for Building an Adapter 3-3
General Guidelines 3-4
Connections Guidelines 3-5
Guidelines for Adapter's Designtime, Runtime, and Life Cycle Management 3-5
Security Guidelines 3-7
Versioning Guidelines 3-7
Guidelines for Limits and Constraints 3-8
Complete the Prerequisites 3-8
Create a Client Application and Obtain Credentials 3-9
Assign the Integration-Instance Developer Role to the Client Application 3-9
Download and Install the Required Software 3-10
Install the VS Code Extension for Rapid Adapter Builder 3-10
Configure the VS Code Extension for Rapid Adapter Builder 3-10

4 Build and Test an Adapter

Workflow for Building and Testing an Adapter 4-1
Obtain the Details for an API Call 4-3
Create a Postman Collection Using API Requests 4-3
Generate an Adapter Definition Document 4-5
Review and Update an Adapter Definition Document 4-6
Validate an Adapter Definition Document 4-7
Validation Rules for the Adapter Definition Document 4-7
Register an Adapter for Testing 4-11
Test an Adapter 4-12
Checklist for Testing an Adapter 4-12
Add Functionality to an Adapter Definition Document 4-15

5 Deploy an Adapter to Production

Determine an Adapter's Version Number 5-1
- Check the Version Number Recommendation 5-2
- Rules for the Semantic Version Check 5-2
- Set the Version Number of an Adapter 5-4

Register an Adapter on Oracle Integration 5-5
Publish the Adapter Documentation 5-5

6 Download, Import, or Delete Adapters

Download or Delete Adapters 6-1
Create or Import Adapter Bundles 6-1

7 Guidelines for Oracle Partners

Introduction 7-1
Build Adapters for the Integration Store 7-1
- Lifecycle Overview 7-2
- Get Started 7-2
- Build and Test Your Adapter 7-3
- Publish Your Adapter 7-3
- Support and Maintain Your Adapter 7-4

Build Private Adapters 7-5

Part III Work with the Adapter Definition Document

8 Use the VS Code Extension's Authoring Features

Validate Documents Against a Schema 8-1
- View Error Highlights 8-2
- View Descriptions for Valid Keys 8-2
- View Code Hints 8-3

Use Code Templates 8-4
- Use Authentication Scheme Templates 8-4
- Use the Test Connection Template 8-5
- Use the Action Code Template 8-5
- Use the Trigger Code Template 8-5

View the VS Code Extension Logs 8-6

9 Update Info Definitions

Learn About Info Definitions 9-1
- Introduction to Info Definitions 9-1
- Where Info Definitions Appear in Oracle Integration 9-2
- Info Properties and Sample Code 9-2

Work with Info Definitions 9-6
- Add Info Definitions 9-6
  - Restrict Outbound Invocations to Specific Domains 9-7
  - Allow Non-HTTPS Traffic 9-8
  - Debug Integration Flows in Designtime 9-8
- Change the Logo of an Adapter 9-98

10 Review Schemas

Learn About Schemas 10-1
- Introduction to Schemas 10-1
- Where Schemas Appear in Oracle Integration 10-1
- Schemas Properties and Sample Code 10-2

11 Update Connection Definitions

Learn About Connection Definitions 11-1
- Introduction to Connection Definitions 11-1
- Where Connection Definitions Appear in Oracle Integration 11-3
- Connection Properties and Sample Code 11-3

Work with Connection Definitions 11-13
- Review a Connection Definition 11-14
- Implement a New Connection Definition 11-14
- Learn About Common Updates to Connection Definitions 11-15
  - Allow Integration Developers to Test Connections 11-16
  - Hide a Connection Property 11-16
  - Update the Name or Description of a Connection Property 11-17
  - Create a Connection Property with a Drop-Down List 11-18
  - Hide a Security Property 11-19
  - Change the Display Name or Description of a Security Property 11-20

Understand Security Policies in Connection Definitions 11-21
- Learn About Security Policies for Invoke Connections 11-21
  - HTTP Basic Authentication 11-21
  - API Key Based Authentication 11-23
  - OCI Signature Version 1 11-24
  - OAuth 1.0 One-Legged Authentication 11-26

OAuth Resource Owner Password Credentials 11-28
OAuth Client Credentials 11-36
OAuth Authorization Code Policy 11-41
OAuth JWT User Assertion 11-52
OAuth JWT Client Assertion 11-56
Learn About Security Policies for Trigger Connections 11-60
Basic Authentication 11-61
OAuth 2.0 Access Tokens 11-62
HMAC Signature Validation 11-63
RSA Signature Validation 11-67
JWT Validation 11-70

12 Update Action Definitions

Learn about Action Definitions 12-1
Introduction to Action Definitions 12-1
Design Considerations to Implement an Action 12-2
Identity of an Action Definition 12-3
Input and Output Schemas of an Action Definition 12-4
Configuring an Action Definition 12-11
Runtime Implementation of an Action Definition 12-13
Action Definition Properties and Sample Code 12-14
Work with Action Definitions 12-17
Implement an Action Definition 12-17
Design User Interface Components of an Adapter 12-18
Design of Configuration Fields 12-28
Combine Field and Value Dependencies 12-33
Dynamic Configuration of Options 12-42
Use Postman Conversion of Flows to Design Configuration Fields 12-45
Use Postman Conversion of Flows to Model Dynamic Schema 12-48

13 Update Trigger Definitions

Learn About Trigger Definitions 13-1
Introduction to Trigger Definitions 13-1
What Is a Webhook Trigger 13-2
What Is a Polling Trigger 13-7
Understand Event Subscription 13-9
Understand Trigger Security 13-13
What Is Webhook Security 13-14
Understand Configuration Scenarios 13-24
Runtime Implementation of Triggers 13-25

Workflow to Implement a Trigger 13-30
Trigger Properties and Sample Code 13-32
Work with Trigger Definitions 13-35
Define a Pass-Through Trigger 13-35
Define a Trigger with Webhook Transformation 13-36
Define a Trigger That Registers Itself with the Producer Application 13-37
Define Validation-Request Handling for a Trigger 13-37
Work with Security Policies in Triggers 13-39
Create a Trigger Connection Definition to Invoke Protected Endpoints 13-39
Implement a Trigger Connection Definition With JWT Signatures Security Policy 13-41
Create a Trigger Connection Definition Using OAuth2.0 Security Policy 13-45
Create a Trigger Connection Definition Using Basic Authentication 13-46
Create a Trigger Connection Definition Using HMAC Signatures 13-47
Create a Trigger Connection Definition Using RSA Signatures 13-49

14 Update Flow Definitions

Learn About Flow Definitions 14-1
Introduction to Flow Definitions 14-1
Supported CNCF and jq Functions 14-3
CNCF Functions Support 14-3
Supported jq Features 14-5
Supported Custom CNCF Serverless Workflow 14-11
Supported Custom jq Functions 14-33
Supported Custom jq Functions for Encoding and Decoding 14-36
Flows Properties and Syntax for Dot (.) Notation 14-39
Dot (.) Notation Syntax for Validation Request Condition in Trigger 14-39
Dot (.) Notation Syntax for Validation Request-Response Generation Flow 14-40
Dot (.) Notation Syntax for Trigger Subscription Register and Deregister Flows 14-42
Dot (.) Notation Syntax for Action Runtime 14-43
Dot (.) Notation Syntax to Authenticate Inbound Signature Policies 14-43
Dot (.) Notation Syntax for Request Transformation in Trigger Runtime 14-45
Dot (.) Notation Syntax for Generate Schema and Configuration Flows 14-45
Dot (.) Notation Syntax for the Init Flow 14-46
Dot (.) Notation Syntax for the Poll Flow 14-47
Dot (.) Notation Syntax for the Success Flow 14-48
Work with Flow Definitions 14-48
Implement the Test-Connection Behavior Using Flows 14-49
Switch States Conditionally in a Flow 14-51
Invoke a GET API 14-59
Invoke a POST API 14-61
Response Transformation in Post-processing 14-63

Send or Receive Form URL Encoded Content 14-66
Receive Binary Content 14-68
Send Multi-part Content 14-70
Send Binary Content 14-72
Override Default Content Handling in connectivity::rest 14-74
Pre and Post Processing Expressions and Multiple Outbound Requests 14-75
Fetch Design Time Configuration Values Using a Flow 14-80

15 Update Category Definitions

Learn About Categories 15-1
Introduction to Categories 15-1
Where Categories Appear in Oracle Integration 15-2
Categories Properties and Sample Code 15-2

Preface

This guide describes how to create and publish an adapter using the Rapid Adapter Builder in Oracle Integration.

Topics:

- Audience
- Documentation Accessibility
- Diversity and Inclusion
- Related Resources
- Conventions

Audience

This guide is for developers who want to build custom adapters using the Rapid Adapter Builder framework in Oracle Integration.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

See these Oracle resources:

- Oracle Cloud at http://cloud.oracle.com
- *Using Integrations in Oracle Integration 3*
- *Using the Oracle Mapper with Oracle Integration 3*
- Oracle Integration documentation on the Oracle Help Center.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I

Get Started with the Rapid Adapter Builder

Using the Rapid Adapter Builder in Oracle Integration, you can build your own adapters for applications to facilitate your integration scenarios.

The Rapid Adapter Builder is available as a Visual Studio Code (VS Code) extension, which helps you use VS Code as an Integrated Development Environment (IDE) for adapter development.

Topics:

- Learn About the Rapid Adapter Builder in Oracle Integration
- Use VS Code as an Integrated Development Environment

1

Learn About the Rapid Adapter Builder in Oracle Integration

Learn about the Rapid Adapter Builder and its features, the reasons to build your own adapter, where your adapters appear when you publish them, and how they may differ from Oracle-provided adapters.

Topics:

- What Is the Rapid Adapter Builder?
- Reasons to Build an Adapter
- Where Your Adapters Appear
- How Your Adapters Differ from Oracle Adapters
- Explore What You Can Do

What Is the Rapid Adapter Builder?

With the Rapid Adapter Builder in Oracle Integration, you can build an adapter for any application that exposes RESTful APIs, without having to write complex code from scratch.

In Oracle Integration, an adapter provides an interface to connect an integration flow with a specific application. Adapters simplify integrations by eliminating the complexity associated with connectivity methods. You can use an adapter to create connections, configure invoke and trigger endpoints, and support runtime activities while designing an integration.

Oracle Integration has an ever-growing library of application-specific adapters that you can readily use for your integration scenarios. However, when an Oracle-provided adapter is not available for your purpose, you can build your own adapter using the Rapid Adapter Builder. It provides all the necessary infrastructure to build adapters for Oracle Integration. An adapter built using the Rapid Adapter Builder can offer the same capabilities as an Oracle-provided adapter. You can implement behaviors similar to those available in the existing adapters on Oracle Integration.

The Rapid Adapter Builder is available as a Visual Studio Code (VS Code) extension, which helps you generate the code required to build an adapter. Using this extension, you can iteratively develop your adapter and publish it to Oracle Integration.

If you're new to Oracle Integration, familiarize yourself with its key concepts, such as, integrations, connections, mappings, and more. See Integration Concepts and About Connections in *Using Integrations in Oracle Integration 3*.

Reasons to Build an Adapter

You can build application-specific adapters for your own use or make it available for others. Any Oracle Integration customer, partner, or independent software vendor can build an adapter. The location where your adapter appears in Oracle Integration varies, depending on your organization.

When an Oracle-provided adapter is not available for your purpose, you can either use a technology adapter (such as the REST Adapter, SOAP Adapter, FTP Adapter, or File Adapter), or you can build a custom adapter using the Rapid Adapter Builder.

The following table lists the different scenarios that may require building an adapter using the Rapid Adapter Builder.

Organization	Reasons to build an adapter
Oracle Integration customers	If you're an Oracle Integration customer, you can build an adapter for use within your organization. When you build an adapter that is tailored to your organization's business processes and requirements, you simplify the integration developers' experience during design time. For example, integration developers can create connections, configure endpoints, and map data without having to research and understand an application's APIs and security policies.
Oracle partners	A partner organization of Oracle can build adapters for third-party applications and make it available for any Oracle Integration customer to download, potentially for a fee. Oracle partners build adapters to better support their own customers. In addition, they can make their adapters available for any Oracle Integration customer and monetize the adapters, thereby creating new revenue streams.
Independent software vendors	Independent software vendors can build adapters for their own applications and make it available for any Oracle Integration customer to download, potentially for a fee. Independent software vendors typically build their own adapters to drive adoption of their software and improve the user experience of integrating with their applications. Vendors who monetize their adapters can also create new revenue streams.

Where Your Adapters Appear

When you build and publish an adapter for your own organization to use, the adapter appears on the Adapters page, within **Design** on your Oracle Integration instance. You can find the adapters that Oracle partners have published from the Home page.

Adapters You Built or Installed

If you publish an adapter, or if you install an adapter that an Oracle partner or independent software vendor has published, you can see it on the Adapters page: In the navigation pane, click **Design**, then **Adapters**.

On the Adapters page, the source indicates who built the adapter:

- **Preinstalled:** Oracle-provided adapters.
- **Local:** Adapters that you built.
- **Third party:** Adapters built by Oracle partners.

Adapters from Oracle Partners

When an Oracle partner or independent software vendor publishes an adapter for use by any Oracle Integration customer, the adapter appears in the Integration store. To access the store: On the Home page, in the Get started section, click **Browse store**.

How Your Adapters Differ from Oracle Adapters

Oracle adapters and non-Oracle adapters offer nearly identical experiences to integration developers. For example, the type of information that integration developers specify when creating connections is the same, and the monitoring capabilities are the same. However, be aware of some key differences.

Area	Oracle adapters	Non-Oracle adapters
Functionality	Oracle adapters offer a rich experience with complete functionality. They cater to a wide breadth of use cases and are generally suitable for most audiences.	Non-Oracle adapters typically offer a more targeted functionality for specific use cases.
User experience for endpoint configuration	The wizard that an integration developer uses to configure an endpoint has three or more sections, depending on the adapter's complexity and supported functionality.	The wizard that an integration developer uses to configure an endpoint always has three sections: Basic Information, Configuration, and Summary.
Version numbers	The version numbers for Oracle adapters correspond to the version numbers of functional releases. For example, 23.12 or 24.02.	The version numbers for non-Oracle adapters follow semantic versioning rules. Version number typically have 3 places, such as 1.2.3.

Note:

Currently, the Rapid Adapter Builder doesn't allow a major version change.

Area	Oracle adapters	Non-Oracle adapters
Backward compatibility	Adapter updates are backward compatible.	Adapter updates are backward compatible.
Maintenance and support	Oracle maintains and supports its own adapters.	Oracle does not maintain or support non-Oracle adapters.

Note:

Although the Rapid Adapter Builder attempts to detect violation of backward compatibility by analyzing the adapter definition document's updates (such as, changes to set of actions, triggers, their contracts, and the security-policy definitions), the updates are not guaranteed to be backward compatible.

Explore What You Can Do

Before building an adapter, familiarize yourself with the features and limitations of the Rapid Adapter Builder.

- Features
- Security
- Extendability
- Limitations

- Limits

Features

The Rapid Adapter Builder provides numerous features to support your business requirements.

Feature	More information
Outbound and inbound integration capabilities	Your adapter supports the following types of integration activities: • Outbound integration from Oracle Integration to another application. For example, you can send requests to external APIs, data sources, or applications. The requests can create, modify, delete, or query resources. • Inbound integration from another application to Oracle Integration. For example, an integration starts running when a state change occurs, such as when a person creates an order.
Supported APIs	The adapter that you build can integrate with an application's: • Public REST APIs • Private REST APIs You can use the connectivity agent to integrate with any APIs that are hosted in your data center or private cloud.
Creation and life cycle management	The creation and lifecycle management of an integration is the same, regardless of whether you or Oracle builds the adapters that the integration uses. For example, the tasks to export, import, and promote integrations to higher environments are the same, regardless of the author of the adapter.
Support for data model extension	Your adapter can support the dynamic retrieval of custom attributes for the standard business objects that are present in an application. Therefore, if you've extended an application, integration developers can refresh the model and get the latest schemas when configuring a connection.
Support for content types while invoking the REST API services	Support for the following content types while invoking the REST API services: • application/JSON • application/octet-stream • multipart/mixed • multipart/form-data
Support for webhook triggers	• Development of webhook triggers to facilitate automatic starting of integrations when specific state changes occur in the event producer application. • Support for automatic and manual registration of webhook triggers during flow activation. • Support for verifying the digitally signed messages, such as HMAC and RSA, that are posted to Oracle Integration. • Support for modeling the pre-flight requests to allow for event producer applications to validate the integrity of webhook. • Support for filtering the events. See Update Trigger Definitions.
Language support	You can define the following in the adapter definition document: • Cloud Native Computing Foundation (CNCF) functions • JQ expressions See Supported CNCF and jq Functions.

Feature	More information
Constructs from the CNCF Serverless Workflow specification	In the adapter definition document, you can model the flow implementation by using a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
Higher-order functions that use the Domain Specific Language pattern	The Rapid Adapter Builder provides the following higher-order functions, which use the Domain Specific Language pattern for security, message format translation domains, and so on: • Custom CNCF Serverless Workflow functions • Custom JQ functions See Supported CNCF and jq Functions.
Inbound concurrent rate limit	Service limits with respect to concurrent rate limit (on the inbound) are automatically enforced for the inbound requests posted to the endpoints stood up using Rapid Adapter Builder based adapters.
Modeling of endpoint user experience	Declaratively modeling various types of user interfaces in the adapter definition document for both action and trigger specific endpoint configuration. See Update Action Definitions.
Extension for Visual Studio (VS) Code	The VS Code extension for Rapid Adapter Builder enables you to use VS Code as an Integrated Development Environment (IDE) for your adapter-development work. The extension facilitates conversion of a Postman collection or an OpenAPI document into an adapter definition document. Additionally, the extension provides several other features, like workspace initialization, validation, code snippets, and so on. The extension integrates with Oracle Integration for seamless creation of adapters. See Use VS Code as an Integrated Development Environment.

Security

The Rapid Adapter Builder provides native support for several authentication mechanisms in the form of security policies for both inbound and outbound invocations.

Security feature	Supported schemes
Outbound authentication	The following authentication and authorization schemes are supported natively using managed security policies for authenticating against the external endpoints and APIs: • HTTP Basic Authentication • API Key Based Authentication • OCI Signature Version 1 • OAuth 1.0 One-Legged Authentication • OAuth Resource Owner Password Credentials • OAuth Client Credentials • OAuth Authorization Code • OAuth JWT User Assertion • OAuth JWT Client Assertion Each connection definition in the adapter can support one or more outbound authentication schemes. See Learn About Security Policies for Invoke Connections.

Security feature	Supported schemes
Inbound authentication	The following schemes, including authentication and authorization schemes and Digital Signature verification schemes, are supported natively using managed security policies for processing inbound requests: • Basic Authentication • OAuth 2.0 Access Tokens • HMAC Signature Validation • RSA Signature Validation • JWT Validation Each connection definition in the adapter can support one or more inbound authentication schemas. See Learn About Security Policies for Trigger Connections.
Extension of OAuth authentication	When an external OAuth provider flow is not RFC 6749 compatible, you can extend the following OAuth authentication schemes: • OAuth Resource Owner Password Credentials • OAuth Client Credentials • OAuth Authorization Code Policy
Additional security features	The adapters that you build using the Rapid Adapter Builder have the following capabilities: • In OCI Vault, automatically securing customer-sensitive information that is captured by a connection. • Calling TLS-protected APIs (TLS v1.3) • Calling external APIs protected using self-signed certificates or lesser known CA signed certificates • Guardrails to ensure that all the outbound requests made on behalf of adapters are only to the domains advertised in the adapter definition document • Automatic GET retries for 502, 503, and 504 • Sending requests in clear text without transport level security By default, the non-SSL support is disabled **Note:** Adapters that send requests in clear text without transport level security have lower security.

Extendability

The Rapid Adapter Builder provides native support for plugging in additional behaviors for the adapters that you build.

The adapter definition document includes the following extension points for plugging in additional behaviors.

Extension point	Supported extendability
Input/Output	Allows you to define the input and output schema of the action or trigger. You can either use a static value for the JSON schema or dynamically determine the JSON schema by calling a flow.

Extension point	Supported extendability
Actions execute	Allows you to extend the default implementation of the flow (as pass-through), and perform more complicated tasks, such as: • Call JQ and CNCF functions. • Execute calls to third parties. • Convert a message from Oracle Integration into a form that the API expects. • Collate and enrich the incoming data into another form that the API expects (which may require accessing other sources to retrieve data). In the same flow, the extension point allows you to perform the following tasks: • Parse the response into a more user-friendly form. • Enrich the response with additional data. • Transform the response to a form required for the integration in Oracle Integration.
Triggers execute	Allows you to perform the following tasks: • Perform post-processing on the information to send to the next activity in the integration. • Transform a message, where a webhook message schema is different from the input message schema (required by Oracle Integration).
Activate integrations	Allows automatic registration of subscription and webhook for a specific event in the event producer or the source application.
Deactivate integrations	Allows automatic de-registration of subscription or webhook for a specific event from the event producer or source application.
Test a connection while designing an integration	Allows you to extend the default ping behavior, with advanced and custom logic. For example, call any idempotent API and ensure success. See Implement the Test-Connection Behavior Using Flows.
Authorize and authenticate during design time and runtime	By default, the OAuth-2 policies are RFC 6749 compatible. However, many implementations differ from RFC standards. Extensible, managed security policies allow you to extend some security policies and override one or more steps.

Limitations

When building an adapter using the adapter definition document and creating connections based on your adapter, be aware of the restrictions.

Area	Limitations
Adapter definition document	The adapter definition document has the following limitations. • Some constructs of the CNCF Serverless Workflow specification aren't supported for modeling the flows in the adapter definition document. • Some JQ functions used in JQ expressions aren't supported. • Oracle doesn't provide version support for adapter definition documents. After pushing an adapter to Oracle Integration, when you need to update an adapter definition document, you are responsible for ensuring backward compatibility for each of the adapters that you build.

Area	Limitations
Types of external endpoints and APIs that are not supported	An adapter that you build using the Rapid Adapter Builder has the following limitations: • The adapter can't use private endpoints to access resources that are hosted in a private cloud or in your organization's data center. • The adapter can't interface with the following: – SOAP APIs, gRPC APIs, GraphQL APIs, and WebSockets. – Endpoints that use a non-HTTP protocol, such as JDBC, AMQP, MQTT, or FTP. • The adapter can't use REST APIs that accept Application/XML as the content type for requests. Similarly, the adapter can't receive responses of Application/XML Content type from the external APIs. • The adapter can't implement custom security schemes for authorization and authentication. • The adapter can't use triggers that are based on synchronous request and response patterns. • The adapter can't call external REST APIs that are protected using mTLS (2-way SSL). • The adapter can't use HTTP-based or non-HTTP-based polling messages and events as triggers to start integrations.

Limits

The Rapid Adapter Builder has limits for several areas. A limit is the quota or allowance for a resource. You cannot change the limits.

Limit categories: General | Schemas | Connections and Security Policies | Triggers | Actions | Actions Runtime | Categories | Triggers Runtime | CNCF Flows

General

Resource	Minimum required	Maximum allowed
The size of the adapter definition document you can include in the adapter bundle, created using the Rapid Adapter Builder.	0	10 MB
The size of the OpenAPI document you can include in the adapter bundle, created using the Rapid Adapter Builder.	0	10 MB
The size of the adapter icon you can include in the adapter bundle, created using the Rapid Adapter Builder.	0	100 KB
Rapid Adapter Builder based adapters in a service instance	0	10
Allowable domains defined for an adapter	0 **Note:** An adapter with 0 allowedDomains is not secure.	10
Schemas defined for an adapter within the adapter definition document	1	100

Schemas

Resource	Minimum required	Maximum allowed
Schemas defined in the adapter definition document	0	500

Connections and Security Policies

Resource	Minimum required	Maximum allowed
Security policies defined within a trigger/action connection definition	1	6
Properties including hidden connection properties defined within a connection definition	0	10
Non-hidden connection properties defined within a connection definition	0	10
Security properties including hidden properties defined within a connection definition	1	10
Security properties excluding the hidden properties defined within a connection definition	0	5

Triggers

Resource	Minimum required	Maximum allowed
Static triggers defined for an adapter within the adapter definition document	0	100
Trigger configurations within a single trigger	0	20
Webhooks defined within a trigger within the adapter definition document	0	1

Actions

Resource	Minimum required	Maximum allowed
Static actions defined for an adapter within the adapter definition document	1	100
Action configurations within a single action	0	20

Actions Runtime

Resource	Maximum allowed
Size of the structured payload that can be sent by Rapid Adapter Builder based adapter to the external endpoint	100 MB
Size of the structured payload that can be received as a response by the Rapid Adapter Builder based adapter from an external endpoint	100 MB

Resource	Maximum allowed
Size of the attachment that can be received as part of the response by the Rapid Adapter Builder based adapter from an external endpoint	1 GB

Categories

Resource	Minimum required	Maximum allowed
Categories defined	0	10

Triggers Runtime

Resource	Maximum allowed
Maximum size of the structured payload that can be received by the Rapid Adapter Builder based adapter endpoint	100 MB

CNCF Flows

Resource	Minimum required	Maximum allowed
Cloud Native Computing Foundation (CNCF) actions within a state	1	10
Actions in a state	1	10
States within a flow	1	10
Flows defined within an adapter	1	100
Outbound invocations (network calls) made using connectivity::rest within any flow, regardless of whether the flow is used in designtime or runtime	0	4

2

Use VS Code as an Integrated Development Environment

The VS Code extension for Rapid Adapter Builder enables you to use VS Code as an Integrated Development Environment (IDE) for your adapter-development work.

Learn how you can use VS Code to author, manage, validate, and publish your adapters.

Topics:

- About the Adapter Definition Document
- Data Mapping in the Adapter Definition Document
- Use Postman as an API-Test Environment
- Explore the VS Code Extension's Features
- Build an Adapter in an Hour

About the Adapter Definition Document

To publish an adapter to Oracle Integration, you must define the specifications and functions of your adapter in a JSON-metadata file, termed as the adapter definition document.

An adapter definition document typically describes the design and implementation of an adapter. It contains information about the adapter's behavior and properties. You don't have to create this document manually. The VS Code extension for Rapid Adapter Builder can quickly create an adapter definition document from either of the following sources:

- A Postman collection of an application's APIs
- An OpenAPI document that describes a set of APIs of an application

The VS Code extension automatically populates various components of the adapter definition document from the data in the Postman collection or OpenAPI file as best it can. However, you must review the document and make necessary changes for the adapter to implement the desired behavior. To understand how data is populated from a Postman collection (or an OpenAPI document) into an adapter definition document, see Data Mapping in the Adapter Definition Document.

An adapter definition document offers the following capabilities, which you can customize as needed:

- Connection definition
- Endpoint configuration
- Flow activation
- Runtime execution

An adapter definition document consists of the following sections. You can edit them in VS Code:

Section	Description
`info`	Describes the general information and branding of the adapter.
`schemas`	Defines the structure that appears in the mapper in Oracle Integration.
`connection`	Defines the connection parameters and security policies supported by adapter.
`triggers`	Defines the events that the adapter can receive to start an integration.
`actions`	Defines the operations that can be implemented against the target application. Actions provide a user-centric encapsulation of the operations available and exposed in the target application.
`categories`	Defines grouping of actions and triggers for a better user experience.
`flows`	Defines the implementation logic that drives the adapter. Flows are modeled using a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.

To learn more about updating different sections of an adapter definition document, see Work with the Adapter Definition Document.

Data Mapping in the Adapter Definition Document

The VS Code extension for Rapid Adapter Builder populates the sections in an adapter definition document with the data from the respective components in a Postman collection or an OpenAPI document. This section provides the data-mapping details between the source document and the generated adapter definition document.

- Data from an OpenAPI Document
- Data from a Postman Collection

Data from an OpenAPI Document

This section provides the mapping of data between an OpenAPI document and an adapter definition document, which is generated using the OpenAPI file.

> **Note:**
>
> If a property is marked *Not Applicable (NA)*, it means the corresponding property is absent from the Postman collection that is used to generate the adapter definition document. In these instances, a default or fixed value is set in the adapter definition document.

Info Component

This topic provides the mapping of data for the Info component of the adapter definition document, generated using an OpenAPI file.

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Notes
`"develop:" + info.title`, where any non-alphanumeric character (including spaces) is replaced with a hyphen (-), and then truncated to the maximum character length (100) for an ID of the adapter definition document.	`id`	Yes	If the title in the OpenAPI document is `"Basemap styles service"`, the value for the ID field in the adapter definition document is: `"develop:basemap-styles-service"`.
NA	`type`	No	User-defined value.
`info.title`	`displayName`	Yes	Title mapping.
`info.description`	`description`	Yes	Description mapping.
NA	`version`	Yes	Default value filled in is 1.0.0.
NA	`specVersion`	Yes	Default value filled in is 1.0.
NA	`categories`	No	User-defined value.
`info.title`	`appInfo.name`	Yes	Mapping for the `appInfo` object in the adapter definition document; child objects can be empty.
`info.contact.email`	`appInfo.contactUs`	Yes	
`info.description`	`appInfo.description`	Yes	
`info.contact.url`	`appInfo.supportURL`	Yes	
`externalDocs.url`	`appInfo.documentationUrl`	Yes	
NA	`publisherInfo.name`	No	User-defined value.
NA	`publisherInfo.description`	No	
NA	`publisherInfo.contactUS`	No	
NA	`publisherInfo.supportURL`	No	
NA	`publisherInfo.documentationURL`	No	
NA	`publisherInfo.releaseNotes`	No	

Connection Component

This topic provides the mapping of data for the Connection component of the adapter definition document, generated using an OpenAPI file.

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Example or Notes
NA	`connectionProperties[].name`	Yes	Fixed value. Set it to `baseURL`.
NA	`connectionProperties[].type`	Yes	Fixed value: `STRING`.
NA	`connectionProperties[].displayName`	Yes	Fixed value: `Base URL`.
NA	`connectionProperties[].required`	Yes	Fixed value: `true`.
NA	`connectionProperties[].hidden`	Yes	The value is `true` if there is an URI in `openapi.servers`; otherwise, it is `false`.
NA	`connectionProperties[].scope`	Yes	Fixed value: single element array containing `"ACTION"`.
NA	`connectionProperties[].tokenized`	Yes	Fixed value: `false`.
The first URI in openapi.servers	`connectionProperties[].default`	Conditionally	If `openapi.servers` is empty, this field won't be filled in.

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Example or Notes
`components.securitySchemes`. See Security Scheme Object.	`connectionProperties[].securityPolicies`	Yes	The following security policies are supported: • No Auth (if no security policy is provided) • Basic Authentication • Bearer Token Authentication • API Key • Implement OAuth 2.0 Resource Owner Password Flow • Implement OAuth 2.0 Client Credentials • Implement OAuth 2.0 Authorization Code For information on how the security policies are implemented, see Learn About Security Policies for Invoke Connections. During conversion, the Rapid Adapter Builder sets a default value for some fields for which it can infer values (such as, `accessTokenUsage`, `oauth.access.token.uri`, `auth.auth.code.uri`), and it sets those fields as `hidden`. However, you can modify the values in the adapter definition document. Further, the following OpenAPI security schemes are implemented using the API-Key Based Authentication security policy: • HTTP Bearer Auth • API key auth Here are some additional notes about the implementation and mapping of the above security schemes: • If there is only Bearer auth, the `accessTokenUsage` security property is resolved to "`-H Authorization: Bearer ${api-key}`" and is hidden. • If there is only API Key auth, the `accessTokenUsage` security property is resolved according to whether it is in the header or query, and then it's hidden. • If both Bearer auth and Api Key auth are present, the `accessTokenUsage` property is visible with the name `Access token authentication`. – If the API Key auth is in the header, the header field is combined (Example: "`-H api_key: ${access_token} -H Authorization: Bearer ${api-key}`"). As it is visible, you can choose which scheme to use. – If the API Key auth is a query parameter, the field is left blank. Regarding the OAuth 2.0 security policies, here are some points to note with respect to OpenAPI documents:

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Example or Notes
			• OpenAPI lists all scopes available, so this is left blank and visible, and you can set your own scope. • Fields such as `oauth.client.id`, `oauth.client.secret`, `username`, `password` are left visible, and you can fill the same. • The fields `oauth.access.token.uri` and `oauth.auth.code.uri` exhibit the following behavior: – For context, all URIs in an OpenAPI document can be relative, so there is a complex behavior for the resolution of access token URI or authorization code URI. OpenAPI documents have a server URI that can be used to resolve relative URIs within the document. However, there can be multiple server URIs and they can be relative. – If a fully qualified URI can be unambigously resolved, it is set as the default value and this security property is hidden. This is true for the following scenarios: Either the provided access or auth URI is absolute, or there is one absolute base-server URI and the relative access or auth URI can be resolved against it. – If a fully qualified URI cannot be determined for access or authorization, the security property is set to visible with no default value that you can fill in. This is true for the following scenarios: Either the server is relative (to the URI where the OpenAPI document is hosted), or there are multiple servers, or the server invalid, or the access or auth URI is invalid.
NA	`securityPolicies[].type`	Yes	Fixed value: managed

Actions Component

This topic provides the mapping of data for the Actions component of the adapter definition document, generated using an OpenAPI file.

One `OperationObject` from an OpenAPI document is mapped to one action in the adapter definition document. For information on operation objects, see Operation Object.

The following example illustrates how an operation object in an OpenAPI document is converted to an action in an adapter definition document:

- The operation-object data in an OpenAPI document: A `GET` operation to the object path `"/webmaps/arcgis/imagery"`
- The corresponding action in an adapter definition document: `webmapsArcgisImageryGetAction`

Other action properties are mapped as follows:

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Example or Notes
`OperationObject.summary` or `OperationObject.operationId`	`displayName`	Yes	If the summary isn't present, the operation ID is used as a fallback.
`OperationObject.description`	`description`	Yes	NA
NA	`execute`	Yes	Fixed value: `flow:generalActionFlow`.
NA	`input`	No	Not necessary as an action can contain a reference to the OpenAPI path that maps to the action. From this reference, the input and output schema are derived. Example: `"$refOpenapi": "POST:/pet"` However, actions that are not using OpenAPI can live alongside actions that use OpenAPI.
NA	`output`	No	Not necessary as an action can contain a reference to the OpenAPI path that maps to the action. From this reference, the input and output schema are derived. Example: `"$refOpenapi": "POST:/pet"` However, actions that are not using OpenAPI can live alongside actions that use OpenAPI.
`HTTPMethod:pathName`	`$refOpenapi`	Yes	Example: `GET:/styles/arcgis/imagery`
`servers[0]+pathName`	`configuration[0].default` (path configuration field)	Yes	Every field (except `default`) is fixed. `{ "name": "path", "type": "TEXT_BOX", "displayName": "Path", "required": true, "readOnly": true, "hidden": true, "default": "/pet" }`

Property in OpenAPI Document	Property in Adapter Definition Document	Filled In During Conversion?	Example or Notes
`servers[1]+methodName`	`configuration[1].default` (method configuration field)	Yes	Every field (except `default`) is fixed. `{ "name": "method", "type": "TEXT_BOX", "displayName": "Method", "required": true, "readOnly": true, "hidden": true, "default": "POST" }`

Flows Component

This topic provides information for the Flows component of the adapter definition document, generated using an OpenAPI file.

During OpenAPI conversion, no information from an OpenAPI document is added to the Flows section of an adapter definition document. However, after conversion, the following default/ generic flow is present in the adapter definition document.

```
"flows": {
    "generalActionFlow": {
        "id": "generalActionFlow",
        "version": "0.1",
        "start": "startState",
        "specVersion": "0.8",
        "functions": [
            {
                "name": "generalRestFunc",
                "operation": "connectivity::rest",
                "type": "custom"
            }
        ],
        "states": [
            {
                "actions": [
                    {
                        "name": "generalActionFlow",
                        "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                                "uri": "${ .connectionProperties.baseURL
+ .configuration.path }",
                                "method": "${ .configuration.method }",
                                "parameters": "${ .input.parameters }",
                                "body": "${ .input.body }"
                            }
                        },
                        "actionDataFilter": {
```

```
                                "results": "${ { body: .body,
headers: .headers } }",
                                "toStateData": "${ .output }"
                            }
                        }
                    ],
                    "name": "startState",
                    "type": "operation",
                    "end": true
                }
            ]
        }
}
```

Schemas Component

This topic provides information for the Schemas component of the adapter definition document, generated using an OpenAPI file.

As the adapter definition document references the OpenAPI schema (`$refOpenAPI`), no schemas are generated.

You can identify if an adapter definition document is driven by OpenAPI using the following characteristics:

- The presence of `"$refOpenapi"` in some actions.
- The parameter `info.settings.supportsRegeneration` set to `true`.

Data from a Postman Collection

This section provides the mapping of data between a Postman collection and an adapter definition document, which is generated using the collection.

> **Note:**
>
> If a property is marked *Not Applicable (NA)*, it means the corresponding property is absent from the Postman collection that is used to generate the adapter definition document. In these instances, a default or fixed value is set in the adapter definition document.

Info Component

This topic provides the mapping of data for the Info component of the adapter definition document, generated using a Postman collection.

Property in Postman Collection	Property in Adapter Definition Document	Filled In During Conversion?	Notes
`"develop:" + info.name.`	`id`	Yes	During conversion, any non-alphanumeric character (including spaces) is replaced with a hyphen (-), and then truncated to the maximum character length (100) for an ID of the adapter definition document.
`info.name`	`displayName`	Yes	Title mapping.
`info.description`	`description`	Yes	Description mapping.
NA	`version`	Yes	Default value filled in is 1.0.0.
NA	`specVersion`	Yes	Default value filled in is 1.0.
NA	`categories`	No	User-defined value.
NA	`appInfo.name`	Yes	Hard-coded null.
NA	`appInfo.contactUs`	Yes	
NA	`appInfo.description`	Yes	
NA	`appInfo.supportURL`	Yes	
NA	`appInfo.documentationUrl`	Yes	
NA	`publisherInfo.name`	Yes	Hard-coded null.
NA	`publisherInfo.description`	Yes	
NA	`publisherInfo.contactUS`	Yes	
NA	`publisherInfo.supportURL`	Yes	
NA	`publisherInfo.documentationURL`	Yes	
NA	`publisherInfo.releaseNotes`	Yes	

Use Postman as an API-Test Environment

Using the Postman application, you can model API requests to a target application, and test the requests for a response.

To create API requests, you must refer to the target application's API documentation and understand the public APIs available. Subsequently, you can use the Postman application to model and test your requests.

Among other things, the Postman application allows you to model an authentication scheme, save a REST API contract, and save a response as an example. In addition, you can group

requests into a collection, and edit the name or description for the requests. You can use all this information to model several aspects for your adapter.

You must create a Postman collection of API requests (to the target application) in order to generate an adapter definition document using the VS Code extension for Rapid Adapter Builder. For the detailed steps on how to create a collection and export it, see Create a Postman Collection Using API Requests.

Explore the VS Code Extension's Features

The VS Code extension for Rapid Adapter Builder is designed to help you easily author an adapter definition document as a JSON file, and efficiently manage your adapter-development life cycle.

In addition to converting a Postman collection of API requests into an adapter definition document, the extension provides several other features, like workspace initialization, validation, code snippets, and so on, to make the authoring process easy.

The following table lists all the features of the VS Code extension for Rapid Adapter Builder.

Feature	Description
Workspace initialization	Provides the ability to use a structured workspace on your system and organize adapter-related artifacts within it. See Configure the VS Code Extension for Rapid Adapter Builder.
Publisher-profile management	Provides the ability to communicate with an Oracle Integration instance to publish and manage adapters. See Configure the VS Code Extension for Rapid Adapter Builder.
Conversion of Postman collections	Provides the ability to convert a Postman collection of API requests into an adapter definition document. The extension automatically generates several sections of the adapter definition document, including: • Connection definitions, including security policies • Actions for the tests in the Postman collection • Specific design-time for rendering endpoint-configuration UI for each action • Action-specific pass-through implementation for invoking the external API at runtime Also, it supports addition of new requests to an existing adapter definition document. See Generate an Adapter Definition Document and Add Functionality to an Adapter Definition Document.
Conversion of OpenAPI documents	Provides the ability to convert an OpenAPI document (describing a set of APIs of an application) into an adapter definition document. See Generate an Adapter Definition Document.

Feature	Description
Schema validation	Provides the ability to validate an adapter definition document against the JSON schema packaged with the extension. If a key or value in your document does not comply with the vocabulary described in the JSON schema, the VS Code extension highlights the violations. Also, the extension offers other essential features, such as code hinting or auto-completion of keys and values, and descriptions of valid keys as tooltips. Additionally, VS Code natively provides features, such as JSON syntax validation, color coding of keys and values, and collapsing or expanding of objects. See Validate Documents Against a Schema.
Code snippets as templates	Provides the ability to author an adapter definition document using readily-available code templates. The extension provides code snippets as templates for various sections of the adapter definition document, such as connections, triggers, actions, and so on. To author a section in a document, you can use the corresponding template and update it with your details, without having to write the code from scratch. See Use Code Templates.
Syntactic and semantic validation	Provides the ability to syntactically and semantically validate an adapter definition document. In addition, the extension provides versioning support according to the semantic versioning rules to better communicate the changes made to the document. See Validate an Adapter Definition Document and Determine an Adapter's Version Number.
Publishing-lifecycle management	Provides the ability to manage the publishing life cycle of an adapter definition document with respect to a particular Oracle Integration instance. Using this feature, you can interact with the Oracle Integration instance specified in the publisher-profile document to perform several tasks, such as registering an adapter bundle on the instance, viewing all the registered adapters on the instance in VS Code, and deleting the adapters. See Register an Adapter on Oracle Integration and Download or Delete Adapters.
Logging support	Provides the ability to view logs and debug issues while authoring an adapter definition document. See View the VS Code Extension Logs.

Build an Adapter in an Hour

To familiarize yourself with the VS Code extension for Rapid Adapter Builder, build a sample adapter for an application.

You can build a sample adapter in an hour using a quick-start tutorial. See *Get Started with the Rapid Adapter Builder in Oracle Integration 3*.

Part II

Build an Adapter Using the Rapid Adapter Builder

Learn how to design, develop, and publish your adapters using the Rapid Adapter Builder.

Whether you use agile, waterfall, or a hybrid approach in your development process, you should complete all phases of the software development life cycle (SDLC) while building an adapter.

The Rapid Adapter Builder is designed to support your work in each phase of the SDLC.

Topics:

- Plan and Design an Adapter
- Build and Test an Adapter
- Deploy an Adapter to Production
- Download, Import, or Delete Adapters
- Guidelines for Oracle Partners

3

Plan and Design an Adapter

Planning and designing is a critical step in the adapter-development process. Use the planning guidelines and development best practices provided here to save time, avoid rework, and deliver capabilities that matter to users. In addition, complete the prerequisite tasks to start building your adapter.

Topics:

- Guidelines for Planning Your Adapter
- Best Practices for Building an Adapter
- Complete the Prerequisites

Guidelines for Planning Your Adapter

Use the guidelines provided in the following topics to plan your adapter-development process.

Topics:

- Identify the Problem
- Understand the Target Application
- Define the Requirements

Identify the Problem

Your first step in planning an adapter is familiarizing yourself with the business process, including the problem that you're trying to solve by creating an adapter.

If you currently follow a software development lifecycle, you probably complete these tasks for your development work already. Adjust these tasks as needed for your organization's requirements and timeline.

Task	More information
Identify your users and stakeholders	Identify the users and stakeholders for the adapter, including: • The integration developers who will work with your adapter. • The people who work in the application for which you're building an adapter. • Anyone who is a stakeholder for the automation work that the adapter will be used for.
Understand your users' and stakeholders' use case	Gather information about the business problem that the adapter is solving. If possible, meet with your users and stakeholders to collect this information. During this phase, try not to think about the implementation. Instead, make every effort to identify all of the problems that you need to solve. Identifying and then prioritizing your requirements will help prevent scope creep during development.

Task	More information
Review the current solution	Your organization might already have a process that addresses the use cases you're addressing. While meeting with users and stakeholders, discuss the process with them. Collect their pain points, and understand why the current process doesn't meet their needs.

Understand the Target Application

Familiarize yourself with the application for which you're building an adapter. Understand its APIs and demonstrate that you can connect to them outside of Oracle Integration.

Task	More information
Understand the application	Familiarize yourself with the application for which you're creating an adapter. For instance, while interviewing users, you might ask for demos of the workflows that require automation.
Understand the application's APIs	Each API represents a feature that you can include in your adapter. Familiarize yourself with the application's APIs by reviewing their documentation.
Identify the APIs that are related to the use cases you've collected	Applications can have dozens or even hundreds of APIs. However, your adapter probably doesn't need to expose all of them. For instance: • Users might not need to automate everything. For example, an administrator might not want to automate the creation of user accounts. • Exposing many APIs increases the clutter in the adapter. An adapter with lots of options in a drop-down list can be difficult to use. • The development effort could be significant, without much return on investment. People might end up using only 10 percent of the APIs that you expose. Spend some time identifying the APIs that are related to the use cases that you collected. These APIs are the candidates for your first release, second release, and so on.
Test the APIs	Now that you understand the APIs that your adapter might need to expose, demonstrate that you can connect to the APIs outside of Oracle Integration. Having some familiarity with the APIs gives you a stronger base for your development work. Work in the environment of your choice, such as Postman or a REST client. To test the APIs, you must understand the API and its authentication scheme, which usually complies with standards such as OAuth or Basic Authentication. **Note:** This step is the most important preparation task. If you can't connect to an application's APIs, you can't build an adapter for the application. Spend as much time as you need to learn how to authenticate with the application's APIs. Testing the APIs is also part of creating a Postman collection. See Create a Postman Collection Using API Requests

Task	More information
If the application supports webhooks, test the webhooks	First, determine whether the application supports webhooks. When an event happens in an application, a webhook sends details about the event to an address, often with data. Cloud applications often support webhooks. Next, if your adapter needs to receive webhooks, test whether the webhooks are reachable. In your integrated development environment (IDE), such as Postman, test the webhook by posting something to a destination. If you don't have a server that can act as a destination and receive HTTP requests, use an online service that serves as a reachable endpoint.

Define the Requirements

After collecting information about the use cases for your adapter and identifying the APIs for each use case, define and prioritize your requirements.

Task	More information
Identify your requirements	Collect your use cases and the APIs that are used to fulfill each use case, and write requirements for your adapter. Be as specific as you prefer, and work in the format that your team usually uses. For instance, you might write a product requirement document, or you might create tickets in your issue tracking software. While you identify the requirements, focus on creating a comprehensive list, rather than prioritizing the requirements.
Prioritize the requirements	Work with your users and stakeholders to prioritize the requirements. When Oracle releases adapters to use in Oracle Integration, the adapters provide robust capabilities so that many organizations can use them to automate their business processes. However, you don't need to deliver a similarly robust adapter, especially for its first release. Instead, aim to deliver capabilities for a small set of users, and then rapidly iterate. For instance, you might aim to support four APIs for your first release, another four APIs for your second release, and so on. Whatever you plan to deliver, keep your users and stakeholders informed.
Plan your development work	For instance, you might groom your development epics, create stories, and assign your work to sprints. Or, you might identify a larger goal, such as exposing four APIs, and work toward a deadline.

Best Practices for Building an Adapter

Follow these best practices and recommendations while building adapters using the Rapid Adapter Builder.

Ensure that your adapters adhere to theses guidelines and exhibit consistent and well-established behavior (as defined by Oracle Integration) during various phases (such as, connection maintenance, endpoint configuration, activation/deactivation, runtime execution, and regeneration) in the lifecycle of an integration flow.

Topics:

- General Guidelines
- Connections Guidelines
- Guidelines for Adapter's Designtime, Runtime, and Life Cycle Management

- Security Guidelines
- Versioning Guidelines
- Guidelines for Limits and Constraints

General Guidelines

- Ensure that the behavior of your adapter is identical to the overall behavior exhibited by Oracle-delivered adapters with respect to adapter's participation in the life cycle of an integration flow, projects, and so on.
- Develop your adapters iteratively. This mechanism of development and delivery is strongly supported in the Rapid Adapter Builder infrastructure.
  This approach offers many benefits. For example, you can better track your time, deliver more frequent releases (if needed), respond to changing requirements more quickly, and test in isolation. Additionally, you won't be overwhelmed by needing to review and test a dozen or more APIs at one time.
- In addition to starting small and iterating, Oracle recommends testing as you go. For instance, after specifying the requirements for a single API, test the adapter before you add the next API.
  The level of testing is up to you. You might focus only on functional testing and save all other testing for after you've added all the required features to the adapter for the release; or you might opt for more extensive testing.
- Keep the business process in mind while developing the adapter. The adapter should expose all capabilities needed for implementing the specific business process in Oracle Integration. For example:
  - An adapter should ensure it exposes support for calling the downstream application APIs in case the business process you are targeting calls for Oracle Integration to send real-time updates to the downstream application.
  - In case of a business process calling for create, update, or delete operations happening in the source application and relaying them to various systems, the adapter for the source application should support a trigger that has the capability to start the integration flow (in Oracle Integration) when changes occur to various resources in the source application.
- Use versioning for changes or enhancements. The Rapid Adapter Builder allows for the delivery of multiple minor versions for a specific major version. At this time, the Rapid Adapter Builder supports only one major version for an adapter. See Rules for the Semantic Version Check.
- Ensure that your adapter's latest minor version for a specific major version is validated against the latest Oracle Integration release, as and when your lower-level environment is patched with the latest Oracle Integration updates.
- If the adapter that was working against the previous version of Oracle Integration starts encountering issues against the newer release, file service requests against Oracle Integration.
- Ensure that all the appropriate tests and validations are done prior to publishing the next minor version.
  Existing artifacts built against the preceding application version, connection, and integration flows (draft as well as deployed) should continue to work against the upgraded application versions seamlessly, without the need for customer intervention.
- Ensure that your adapters can be used in/as the following artifacts:
  - All integration patterns

  - Global or project-specific adapter connections
  - Shared adapter connections
  - Any accelerator or recipe
- Create documentation that helps integration developers work with your adapter. Contact Oracle for the adapter documentation template to use and the adapter naming guidelines.

Connections Guidelines

General Connection Guidelines

- Do not impose any limits on the following items:
  - Number of endpoint configurations for a single connection—within a flow as well as across flows.
  - Number of endpoint configurations having identical configuration.

Test-Connection Guidelines

- Ensure that a Test connection invokes an idempotent (non-destructive) operation (HTTP GET) to a live endpoint with the fabricated payload, so that the service is consumable (outbound proxy, TLS versions, SSL handshake, authentication and authorization-related errors along with prescriptive recommendations needed to be rendered). See Implement the Test-Connection Behavior Using Flows.
- Ensure that a Test connection fails and throws an appropriate message when the adapter tries to make an outbound connection to a non-supported application version.
- Show the supported minimum and maximum releases in the error message.
- Ensure that the Connection page presents the application prerequisites that need to be completed.

Connection Modification Guidelines

- Ensure that the existing deployed Oracle Integration flows that leverage the your adapter continue to work without any deactivation/reactivation after the following changes to the connection:
  - Password/API Key
  - Security Policy (examples: HTTP Basic Auth to Username token; HTTP Basic Auth to OAuth Resource Owner Password Flow)
  - Fresh consent for delegated authorization in case of OAuth flows (assuming there are no scope changes)
  - Endpoint URL (assuming the new instance is identical to that of the existing instance – for example, application POD moved from 1 zone to another)
  - Username (assuming the new user has the same privileges and roles as that of the previous user)
  - Collection of additional connection metadata.

Guidelines for Adapter's Designtime, Runtime, and Life Cycle Management

- Generally, your adapters should automatically discover new publicly-consumable business objects/APIs/events as they are introduced in the applications

  - Do not introduce code changes to adapters that make the adapter consumers wait for the next adapter release to consume these objects.
  - Make sure that Oracle Integration flows created after the availability of business objects/APIs are able to discover these objects.
- You can develop adapters to target a specific business domain. You can code it to present actions specific to select business objects and operations.

Guidelines for Data Source Customizations

- Ensure that custom objects and attributes are automatically discoverable.
- Ensure that custom objects and custom attributes show the functional names in Oracle Integration (not field 1, field 2, and so on).
- Ensure that the Regenerate option is available for adapters whose applications support customizations to the data models.
- Ensure that a Re-edit of existing adapter configuration wizards automatically brings in the new custom objects and custom attributes.

Guidelines for Integration Flow Promotion or Applications Upgrade

- Ensure that the integration flow archive (IAR) files (of the newly-built adapter) work against target application instance when imported into a higher Oracle Integration environment upon activation.
  - Ensure that the adapter consumers are not asked to re-edit the adapter configurations in the higher Oracle Integration environment.
  - Do not hard-wire external application or data source-specific endpoint details, which are stored in the adapter definition document.
- Make sure that the application upgrades are transparent. Existing flows must work against an upgraded application instance without any changes, production outages, and patches.
- Ensure that adapter consumers are not forced to create new connections to work against newer application releases or newer version of external data sources.
- Ensure that adapter consumers do not have to deactivate and reactivate the existing flows (that reference your adapters) to work against the newer releases of an external application.
- Optionally, in the subsequent minor versions of the adapter release, you can deliver the new infrastructure features added to applications (such as, support for new authentication/authorization policy, bulk data import, and so on).
- Ensure that new flows work against latest application releases with no change or loss in behavior.
- You must certify the adapters against the new versions of application releases.
  - Ensure that you certify prior to the refresh of the Oracle Integration fleet with the newest application releases.
  - Ensure that the existing automation tests are run to test the regressions, if any.
- Ensure that your adapters work against all of the existing supported application versions.
- Avoid situations where adapter consumers would need to upgrade to the most recently-delivered application release for the adapter to work.
- However, your adapter may continue to have limited capabilities against older application releases.

Security Guidelines

Authentication or Authorization Guidelines

- Ensure that the connection definition modeled in the adapter definition document for invocation of actions makes an attempt to support all of the authentication and authorization schemes supported by the external application.
- Prioritize on exposing the most secured authentication or authorization schemes over the vulnerable ones.
- Note that you cannot remove the authentication schemes implemented in the preceding version of an adapter; however, you can add new authentication schemes.

Transport Level Security Guidelines

- Ensure external endpoints are secured using TLS v1.3 or v1.2.
- Avoid support for TLS v1.1 and 1.0 protected endpoints.
- Avoid calling external APIs protected using self-signed certificates or lesser-known CA-signed certificates. Oracle Integration flags adapters expecting this requirement and warns consumers subscribing to such adapters.
- Ensure that the adapter development document advertises to the domains where the outbound requests need to be sent by the adapters. This information is presented to the consumers for their understanding. Oracle Integration flags adapters not having this information and warns consumers subscribing to such adapters.
- Send requests in clear text without transport-level security. By default, the non-SSL support is disabled. Even though you can opt for supporting the transmission of data over the wire in non-encrypted mode (no TLS), it would have an adverse impact on the overall credit score, and Oracle Integration warns consumers subscribing to these adapters.

Versioning Guidelines

General Versioning Guidelines

- Develop your adapters iteratively. This mechanism of development and delivery is strongly supported in the Rapid Adapter Builder infrastructure.
- Use versioning for changes or enhancements. The Rapid Adapter Builder allows for the delivery of multiple minor versions for a specific major version. At this time, the Rapid Adapter Builder supports only one major version for an adapter. See Rules for the Semantic Version Check.
- Ensure that the minor versions are backward compatible, and the integration flows built and deployed using the preceding minor version of an adapter continue to work without any manual intervention or disruptions, after the uptake of the latest and greatest minor version for that adapter by a consumer.

Connections Versioning Guidelines

- Note that you cannot remove existing connection properties in the newer release.
- You cannot make the new connection properties mandatory.
- Do not change the functional behavior of the existing properties.

- Ensure that the connections created with the older adapter release are usable for creating integration flows after the uptake of the newer version.
- Ensure the existing deployed flows built on prior version of the adapter continue to work after the uptake of the newer version.
- Ensure that your versioning does not require adapter consumers to deactivate and reactivate the existing flows.
- Ensure that the newer adapter runtime handles the errors occurring in the existing metadata gracefully. (Ensure this even for the adapter changes done to fix the existing bugs or performance issues.)
- Ensure that the new capabilities that you introduce in the newer adapter releases are opt-ins.
- Ensure that the existing flows in draft mode (functionally complete) are deployable after the uptake of the newer version of the adapter. This should not result in adapter consumers having to configure the already-configured connections or re-edit the existing adapter configurations.

Guidelines for Oracle Integration Messages

- Ensure that data structures generated using the newer minor version of an adapter are backward compatible. The maps built using the preceding version of the adapter should not be lost when existing endpoint configurations is reconfigured.
- Ensure that the existing maps produced prior to the uptake of the newer minor version continue to work, both at design time and runtime after the uptake of the latest minor version.

Guidelines for Limits and Constraints

The Rapid Adapter Builder has limits for several areas. A limit is the quota or allowance for a resource. You must adhere to theses defined limits for your adapter. See Limits.

Source Control Guidelines

Ensure that Rapid Adapter Builder archive, containing the working version, is source controlled.

Complete the Prerequisites

Before you can start building an adapter, you must complete a few tasks, such as, obtaining access to Oracle Integration, installing the required software, and so on.

Order	Task	More information
1	Get access to Oracle Integration	If you don't have them already, get: • An account on Oracle Integration, including your sign-in credentials • The URL for your Oracle Integration instance Work with your Oracle Integration administrator to complete this task.
2	Create a client application in IDCS for your Oracle Integration instance and obtain the client credentials	See Create a Client Application and Obtain Credentials.

Order	Task	More information
3	Assign the integration-instance developer role to the client application you created previously	See Assign the Integration-Instance Developer Role to the Client Application.
4	Download and install the required software	See Download and Install the Required Software.
5	Install the VS Code extension for Rapid Adapter Builder	See Install the VS Code Extension for Rapid Adapter Builder.
6	Configure the VS Code extension for Rapid Adapter Builder	See Configure the VS Code Extension for Rapid Adapter Builder.

Create a Client Application and Obtain Credentials

In Oracle Identity Cloud Service (IDCS), create a client application for your Oracle Integration instance and obtain the client credentials.

You'll use these client credentials in your publisher-profile details while publishing an adapter using the Rapid Adapter Builder.

To create an IDCS application and obtain the required details, see *Prerequisites for Client Credentials* in Using OAuth 2.0 Grants in Oracle Identity Cloud Service Environments.

You must obtain the following details for your Oracle Integration instance:

- `Host`: URL of the Oracle Integration instance
- `integrationInstance`: Name of the Oracle Integration instance
- The following client credentials:
  - `tokenUrl`
  - `clientId`
  - `clientSecret`
  - `scope`

Assign the Integration-Instance Developer Role to the Client Application

To successfully use the client credentials, assign the integration-instance developer role to the client application you created previously.

To assign the required role to the client application:

1. Log in to the Oracle Identity Cloud Service (IDCS) console.
2. Click **Oracle Cloud Services**.

   The Oracle Cloud Services page appears.
3. Search for your Oracle Integration instance and select it.

   The details of the instance appear.
4. Click **Application Roles**.

   The list of all applicable roles appears.

5. Locate and click the **ServiceDeveloper** role, and then click **Assign Applications**.

   The Assign Applications dialog appears, displaying the client application you created previously.

6. Select the client application and click **OK**.

   Now, IDCS assigns the integration-instance developer role to the application.

Download and Install the Required Software

To use the Rapid Adapter Builder, you require additional applications or software installed on your system.

1. Download and install the Postman application. Visit the Download Postman page. Create an account on Postman to access full features.
2. Download and install VS Code. Visit the Download Visual Studio Code page.

   You must use VS Code version 1.73.0 or higher.

Install the VS Code Extension for Rapid Adapter Builder

Download and install the VS Code extension for Rapid Adapter Builder.

1. On Visual Studio Marketplace, visit Oracle's page.
2. Find and click the **Oracle Integration Cloud Rapid Adapter Builder** tile.

   The Details page opens.

3. Click **Install**.

   The extension file opens in VS Code.

4. Click **Install** in VS Code.

   Now, the extension is installed and appears on the sidebar.

Configure the VS Code Extension for Rapid Adapter Builder

After you install the VS Code extension for Rapid Adapter Builder, you must initialize a workspace on your system and configure the publisher profile for the extension.

Select a folder on your system to use as a structured workspace and organize adapter-related artifacts within it. Initialize this workspace using the VS Code extension.

Additionally, update the publisher profile of the extension with the details of the Oracle Integration instance that you'll publish the adapter to. You'll enter the client credentials you obtained earlier along with some additional data. The information in the publisher profile is used to communicate with the specified Oracle Integration instance for various tasks. The extension securely stores the client credentials you enter for each Oracle Integration instance in the `publisher-profiles.yaml` file. The structure of the `yaml` file is as follows:

Property Name	Definition
`active`	The active profile to use. The value must match a profile name from the profiles list.
`profiles`	The array of profile objects.
`profiles[*].name`	The unique name for a profile.

Property Name	Definition
`profiles[*].host`	The design-time URL of the Oracle Integration instance for REST API.
`profiles[*].integrationInstance`	The name of your Oracle Integration instance.
`profiles[*].auth.tokenUrl`	The IDCS get-token URL for your Oracle Integration instance.
`profiles[*].auth.clientId`	The client ID of the configured application in IDCS.
`profiles[*].auth.clientSecret`	The client secret of the configured application in IDCS.
`profiles[*].auth.scope`	The allowed scope of the configured application in IDCS.

Perform the following tasks in VS Code to make the required configuration.

1. Initialize a workspace using the VS Code extension.
   a. In VS Code, click **Explorer** on the sidebar.
   b. Click **Open Folder**, and choose a folder on your system as the workspace for adapter development.

      The folder is now listed on the left in the Explorer view.

      > **Note:**
      >
      > You can also click **File** in the top ribbon, and select **Open Folder...** to open a folder in VS Code.

   c. Press **Ctrl + Shift + P** to open the command palette, type `RAB: Initialize Workspace`, and hit **Enter**.

      The VS Code extension now initializes the workspace, and creates a few directories and template files in the workspace. The directories and files created are displayed in the Explorer view on the left. The following directories and files are created:

      - **api**: You can save all OpenAPI files in this folder.

        > **Note:**
        >
        > For the VS Code extension to package an OpenAPI document into an adapter bundle, rename the OpenAPI document as `openapi.resource.json`.

      - **definitions**: The extension creates a default adapter definition document, `main.add.json`, in this folder. You can overwrite the default document with the data from a Postman collection or an OpenAPI file. Additionally, all the other adapter definition documents you generate are saved in this folder. The file names of all adapter definition documents are appended with `.add.json`.
      - **misc**: You can save all Postman collections in this folder. Ensure that the file names of all Postman collections are appended with `.postman_collection.json`.

- **logo.svg**: A default logo file for your adapter is created under the root folder. You can replace this file with the required image.

2. Configure the publisher profile for the extension.
   a. In VS Code, click **OIC Rapid Adapter Builder** on the sidebar.
   b. In the resulting view, click **Edit** in the Publisher Profiles section.

      The `publisher-profiles.yaml` file opens in the editor.
   c. Update the file with the client credentials you obtained earlier. See Create a Client Application and Obtain Credentials. In addition, update the `active` field with the name of the profile to use.
   d. Press **Ctrl+S** on your keyboard to save the file.

   If you enter credentials for more than one Oracle Integration instance, the `.yaml` file stores them in a hierarchical structure, implemented by indenting the details.

   The following example shows the details of two Oracle Integration instances, named `Test_Lab1` and `Test_Lab2`, stored in a `.yaml` file.

```
# select which profile to use.
active: Test_Lab1

# multiple profiles can be defined.
profiles:
  - name: Test_Lab1
    host: https://<oicinstance.oraclecloud.com>
    integrationInstance: example-instance1
    auth:
      tokenUrl: https://<idcs-token>/oauth2/v1/token
      clientId: ef8xxxxxbf0c4101a4f74ae5fe301exx
      clientSecret: 0ebdae12-xxxx-4182-9051-8dbd83b2xxxx
      scope: https://
08DADxxxx9D4496F9FBAFxxCFA1B0657.integration.region-1.oraclecloud.com:443ur
n:opc:resource:consumer::all
  - name: Test_Lab2
    host: https://<oicinstance.oraclecloud.com>
    integrationInstance: example-instance2
    auth:
      tokenUrl: https://<idcs-token>/oauth2/v1/token
      clientId: ef9xxxxxbf0c4101a4f74ae5fe301exx
      clientSecret: 1ebdae12-xxxx-4182-9051-8dbd83b2xxxx
      scope: https://
18DADxxxx9D4496F9FBAFxxCFA1B0657.integration.region-2.oraclecloud.com:443ur
n:opc:resource:consumer::all
```

4

Build and Test an Adapter

After completing the planning and prerequisite tasks, you can start building your adapter. Refer to the recommended workflow provided in this chapter for building adapters using the Rapid Adapter Builder.

Topics:

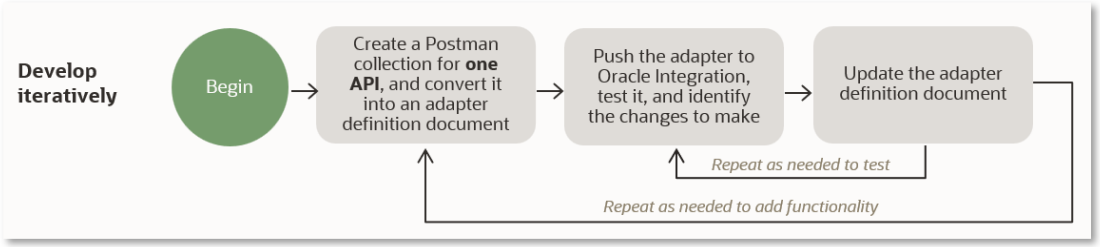
- Workflow for Building and Testing an Adapter
- Obtain the Details for an API Call
- Create a Postman Collection Using API Requests
- Generate an Adapter Definition Document
- Review and Update an Adapter Definition Document
- Validate an Adapter Definition Document
- Register an Adapter for Testing
- Test an Adapter
- Add Functionality to an Adapter Definition Document

Workflow for Building and Testing an Adapter

Use the following approach as a general guideline to build and test an adapter; however, you may choose to complete some of these steps in any order. Iteratively refine your adapter as you build it to suit your requirements.

> **Note:**
>
> The workflow and the associated steps illustrate building an adapter using a Postman collection of an application's APIs. However, you can also use this workflow to build an adapter using an OpenAPI document that describes a set of application APIs. If you'd like to use an OpenAPI document, import the document into VS Code and start from Step 3 in the Workflow Table.

Here is an illustration of the recommended iterative-development process:



Workflow Table

Order	Step	More information
1	Obtain the details for an API call	To get started, refer to the public APIs available for an application, and obtain the URL and parameters necessary to make a particular API request. See Obtain the Details for an API Call.
2	Create a Postman collection using the request	In the Postman application, create a collection using the API request, test the request, and export the collection to a JSON file. As a best practice, start with one API request in your collection. Create a Postman Collection Using API Requests.
3	Generate the adapter definition document	Using the VS Code extension for Rapid Adapter Builder, generate an adapter definition document from the Postman collection (or an OpenAPI document). Generate an Adapter Definition Document.
4	Review and update the adapter definition document	When you convert a Postman collection into an adapter definition document, several sections of the document are automatically populated for you, including connections and actions. However, you can review and update the document according to your requirements in the VS Code editor. See Review and Update an Adapter Definition Document
5	Validate the adapter definition document	After making updates to your adapter definition document, validate the document to ensure it doesn't contain any errors. See Validate an Adapter Definition Document.
6	Register the adapter for testing	Make you adapter available on an development instance of Oracle Integration for testing. See Register an Adapter for Testing.
7	Test the adapter	Test the adapter's functionality to ensure that everything works as expected. See Test an Adapter. Depending on your findings during testing, update your adapter definition document. Review, validate, and test until you achieve the desired results.
8	Repeat the previous steps to add functionality	After you've successfully added one functionality to your adapter, add support for another API request. However, you don't have to generate a new adapter definition document for each new request. The VS Code extension allows you to add new requests to the same adapter definition document. See Add Functionality to an Adapter Definition Document.

> **Note:**
>
> While building or testing your adapter, create documentation that helps integration developers work with the adapter. You can use any adapter user guide created by Oracle as a template. See the Adapters page on the Oracle Help Center.

After you have completed the iterative-development of your adapter, you can release it on a production instance of Oracle Integration. See Deploy an Adapter to Production.

Obtain the Details for an API Call

To make an API request for any application, you must obtain the required details, such as URL, parameters, and so on, using the API documentation available for the application.

Previous step: Complete the Prerequisites

1. Refer to the API documentation of the application for which you want to build an adapter.
2. Identify the API requests to use in your adapter.
3. Note the URL, parameters, and other details necessary to make those requests.

Next step: Create a Postman Collection Using API Requests

Create a Postman Collection Using API Requests

Create a collection in the Postman application, add and test the required API request, and export the collection to a JSON file.

Previous step: Obtain the Details for an API Call

While building an adapter, starting with a Postman collection saves you time as you can group several API requests into a collection and also shape the responses according to your requirements.

When you test the API requests in a Postman collection, you must save the responses received, so that the Rapid Adapter Builder can generate a schema in the adapter definition document.

> **Tip:**
>
> Start small. For at least your first adapter, create a Postman collection with just one API request. After you build and test your adapter, you can add support for another API if necessary. See Best Practices for Building an Adapter. For subsequent iterations, you can use an *existing* Postman collection and add new requests to it. See Adding a new request to a collection.

1. Open the Postman application and sign in with your account.
2. Create a collection.
   a. With **Collections** selected in the left pane, click **New**, and in the resulting dialog, click **Collection**.

      A new, blank collection is created and listed in the left pane.

b. Under the **Overview** tab on the right, click the default collection name (**New Collection**) to rename the collection. For example, enter `<ApplicationName>_Adapter`.

3. Add the API request for which you obtained the details earlier, test it, and save the response as an example.

   a. In the left pane, click **Add a request** under the newly-created collection.

      A new, blank request is created and displayed. Click the **New Request** field at the top of the workspace to rename the request.

   b. In the **URL** field of the request, select the right HTTP verb, and enter the request URL you obtained earlier. See Obtain the Details for an API Call.

   c. Enter all the other information you obtained for the request, such as parameters, authorization details, and headers.

   d. Click **Send** to test the request.

      The Response section displays the data received.

   e. In the Response section, click **Save as example** to save the response.

      A new example is created and displayed. Click the **New Request** field at the top of the workspace to rename the example.

      > **Tip:**
      >
      > Before you save the response as an example, shape the response by removing properties that aren't applicable to your adapter. You can also remove this information after converting the Postman collection to an adapter definition document.

   f. In the left pane, click the request item to switch back to the Request page from the Example page, and then click **Save** to save the entire request.

4. Export the Postman collection to your workspace.

   a. In the left pane, point to the collection, click **View more actions** ..., and then click **Export** from the resulting menu.

   b. In the Export collection dialog, leave the default options selected, and click **Export**.

   c. In the Select path to save file window, navigate to the folder that you selected and initialized as workspace previously. See Configure the VS Code Extension for Rapid Adapter Builder. In the workspace folder, open the **misc** directory, and click **Save** to export the file to this directory.

      The Postman collection is saved as a JSON file, named `<ApplicationName>_Adapter.postman_collection.json`.

      > **Note:**
      >
      > For the Rapid Adapter Builder to convert the collection into an adapter definition document, the file name of the Postman collection should always contain the suffix `.postman_collection`.

Next step: Generate an Adapter Definition Document

Generate an Adapter Definition Document

Using the VS Code extension, you can generate an adapter definition document from an exported Postman collection or from an OpenAPI document.

1. For Postman Collections, perform the following steps:
   a. In VS Code, click **Explorer** on the sidebar to bring up the Explorer view.

      The directory structure of your workspace folder is displayed.
   b. Click the **misc** directory listing to expand it.
   c. Right-click the exported collection file (`<ApplicationName>_Adapter.postman_collection.json`), and then click **RAB: Convert Postman Collection**.
   d. In the Select request(s) pane, select the request to convert, and click **Done**.
2. For OpenAPI documents, perform the following steps:
   a. Import the OpenAPI document (in a JSON format) into the **api** directory of your workspace folder.
   b. In VS Code, click **Explorer** on the sidebar to bring up the Explorer view.

      The directory structure of your workspace folder is displayed.
   c. Click the **api** directory listing to expand it.
   d. Right-click the OpenAPI document you imported, and click **Rename**.

      Rename the document as `openapi.resource.json`.
   e. Right-click the document again, and click **RAB: Convert OpenAPI document**.
   f. In the Select request(s) pane, select the request to convert, and click **Done**.

   > **Note:**
   >
   > You cannot convert an OpenAPI document with more than 200 requests.
3. In the dialog box that appears, you can choose either of the following options:
   - Click **Update main** to add the request to the default document provided with the VS Code extension (`main.add.json`).
   - Click **Save new** to generate a new adapter definition document.

   The VS Code extension converts the request into the document you select. If you choose to create a new document, the extension generates the document and saves it in the `definitions` directory of your workspace. In addition, the document opens in the VS Code editor.

> **Note:**
>
> If you create a new document, the VS Code extension doesn't recognize it as an adapter definition document. The extension only recognizes the default document, `main.add.json`. You can either copy the content from the new document to the default document or rename the new document to `main.add.json`.

4. Ensure that the adapter definition document contains an API response schema.
5. Press **Ctrl+S** on your keyboard to save the file.

Next step: Review and Update an Adapter Definition Document

Review and Update an Adapter Definition Document

After you convert a Postman collection (or an OpenAPI document) to an adapter definition document, you can review the generated document and make changes according to your requirements.

Previous step: Generate an Adapter Definition Document

To a large extent, the Rapid Adapter Builder automatically populates various components of the document from the data in the Postman collection (or OpenAPI document). However, if you publish a converted adapter definition document without much changes, you may have a functional adapter, but the user experience in terms of the endpoint configuration or the mapper elements may not be optimal. Therefore, you must review the document and make the changes required for your adapter.

Among other things, you can extend or override the interfaces in the document (for example, the adapter-endpoint configuration interface), or add new capabilities to the document.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Click the **definitions** directory listing to expand it.
3. Click the adapter definition document that you want to review or update.

   The document is displayed in the VS Code editor.
4. Update the document as needed. For guidelines on how to update different sections of the adapter definition document, see Work with the Adapter Definition Document.

> **Note:**
>
> After conversion, at a minimum, you must add the following details to an adapter definition document:
>
> - A unique ID. See Add Info Definitions.
> - An authentication scheme. See Use Authentication Scheme Templates.

Next step: Validate an Adapter Definition Document

Validate an Adapter Definition Document

Before you publish the adapter to Oracle Integration, you must validate the adapter definition document. Validating the document ensures that it does not contain any syntactic or semantic errors.

Previous step: Review and Update an Adapter Definition Document

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.

2. Within the **definitions** directory listing, right-click the adapter definition document, and then click **RAB: Validate**.

   Now, the extension validates the document. If your document isn't valid, errors are displayed in the VS Code editor's bottom panel, on the **Output** tab.

3. Review the validation errors, update the document, and validate again until the document has no errors. For information on understanding the errors, see Validation Rules for the Adapter Definition Document.

   If your document is valid, a success message is displayed.

Next step: Register an Adapter for Testing

Validation Rules for the Adapter Definition Document

When you push an adapter to Oracle Integration, the Rapid Adapter Builder validates the adapter definition document for syntactic and semantic errors. Additionally, you can validate the adapter definition document at any time from within the VS Code extension.

The following sections describe the warnings and errors that can occur for an adapter definition document.

Errors

You must resolve all errors. You can't push an adapter to Oracle Integration if the adapter definition document has one or more validation errors.

Rule ID	What happened, and how to fix it
E110	The `connection` section contains code for testing a connection, but the code references a flow that isn't in the `flows` section. If the flow is missing, add it. Otherwise, update the name of the flow in the `connection` section.
E111	In the `actions` section, The action flow is undefined.
E112	In the `actions` section, the `input` property is empty. Define an input schema using the `input` property. If flow is referenced by action schema, then flow must exist.

Rule ID	What happened, and how to fix it
E113	In the `actions` section, the `output` property is empty. Define an output schema using the `output` property. **Note:** Some actions may not need an output schema.
E114	In the `connection` section, Extended OAuth flow in a connection is undefined. If an extended OAuth flow is defined, the referred flow must exist.
E115	In the `flows` section, The configuration field flow is not defined. If a configuration field references a flow, then the flow must exist.
E118	In the `actions` section, If an action's input or output schema reference is not defined. If a referenced schema does not exist.
E119	In the `flows` section, The default testConnectionFlow URI from the code snippet on Visual Studio Code is not updated.
E125	In the `flows` section, The jq expression is invalid.
E128	In the `actions` section, The parent dependency field does not exist.
E131	In the `info` section, Allowed domain is an array and supports a valid host name/IP:Port combination in value. The only allowed wildcard special character is `*`.
E132	In the `flows` section, Number of outbound invocations using connectivity::rest must not exceed 4 for each flow (design time and runtime) in synchronous flow.
E133	In the `flows` section, The security property is invalid.
E134	In the `flows` section, Flow contains multiple actions with the same name.
E135	In the `flows` section, Flow contains an invalid usage of a function.

Warnings

Review all warnings and address them if you can. You can still push an adapter to Oracle Integration if the adapter definition document has one or more warnings.

Rule ID	What happened, and how to fix it
W101	No triggers are defined in the `triggers` section. Define one or more triggers.
W102	In the `info` section, Empty keys or empty strings within the application information section.
W103	In the `actions` section, two or more actions have the same `displayName` value. Update the actions so that they have unique `displayName` values.
W104	In the `connection` section, two or more security policies have the same `displayName` value. Update the security policies so that they have unique `displayName` values.
W105	In the `connection` section, two or more security properties have the same `displayName` value. Update the security properties so that they have unique `displayName` values.
W106	In the `connection` section, two or more connection properties have the same `displayName` value. Update the connection properties so that they have unique `displayName` values.
W107	In the `categories` section, two or more category groups have the same `displayName` value. Update the category groups so that they have unique `displayName` values.

W108	In the `flows` section,
	Absolute endpointURI must not be specified as URI within CNCF Actions.
	
	`actions":[`
	`{`
	`"functionRef": {`
	`"refName": "generalRestFunc",`
	`"arguments": {`
	`"uri": "https://www.googleapis.com/drive/v3/files",`
	`"method": "GET"`
	`}`
	`},`
	
	Note:
	The baseURL should be something set at the connection level and if this is static, to make the property hidden and offer a default value.

W110	In the `info` section, This warning appears when you use the following constructs to define the domain: • Wildcard as a value. • Only top level domain. • Wildcard in top level. • Wildcard in second level domain.

Rule ID	What happened, and how to fix it
W111	the `info` section doesn't define the allowed domains for the adapter. Define the allowed domains for the adapter.
W112	In the `info` section, below `publisherInfo`, one or more required properties isn't defined. Define all required properties in the `publisherInfo` section.

Unidentified Rules

Rule ID	What happened, and how to fix it
warning	The URLs that are specified in the `flows` and `connections` sections, including `securityPolicies` are not specified in the `info` section, in the `allowedDomains` section. Make sure that all URLs that are specified in the document are also specified in the `info` section. This security requirement prevents an adapter from reaching a third-party endpoint that hasn't been specified as an allowable domain.
warning	In the `schemas` section, one or more headers contains a capital letter. Update the headers so that they include only lower-case letters.
error	In the `connection` section, The connection information does not contain flows, and if a connection property is used in security policies.
error	In the `connection` section, jq expression is invalid in a default security property value.
error	The `connection` section has more than 10 connection properties. Reduce the number of connection properties to 10 or fewer.
error	In the `connection` section, Name of the managed security policy name is unknown.
error	In the `triggers` section, The number of static triggers exceed 100.
error	In the `triggers` section, The JSON schema contains unknown properties.
error	In the `triggers` section, In the schema, the trigger and action sections using JSON Schema, they should either choose reference ($ref), to the flattened schema.
warning	In the `actions` section, Number of static actions exceed 100.
error	In the `actions` section, Number of action configurations exceed 30.
error	In the `actions` section, The different ways to offer schemas for input/output should be mutual exclusion. For example, if resolved from urn, there should not have hard-coded schemaType and schema

Rule ID	What happened, and how to fix it
	In the `actions` section, The configuration field contains more than one dependency declaration.
warning	In the `categories` section, Unused categories are specified.
error	In the `categories` section, Category support exceeds the maximum limit of 20.
error	In the `flows` section, A testing flow must not have any references to .Input. There are no action inputs while calling a testing flow.
	In the `flows` section, Number of CNCF actions within a state exceed 10.
	In the `flows` section, Number of states with in a flow should not exceed 10

Register an Adapter for Testing

After you have validated the adapter definition document, you can make your adapter available for testing on a development instance of Oracle Integration.

Previous step: Validate an Adapter Definition Document

Ensure that you've specified a development instance of Oracle Integration as *active* in the publisher profile. See Configure the VS Code Extension for Rapid Adapter Builder.

> **Note:**
>
> - For the first time, you can register your adapter on Oracle Integration with the default version number in the adapter definition document (that is, 1.0.0). However, for subsequent pushes, you'll need to update the version number in the document's `info` section to make your adapter available on Oracle Integration. For testing purposes, you can use a versioning format of your choice. However, while deploying your adapter to production, use the semantic versioning rules. See Determine an Adapter's Version Number.
> - For each adapter bundle, an Oracle Integration instance can have exactly one live version of an adapter. If you update an adapter bundle and register the latest version, the latest version of the bundle always overwrites the previous version.

1. Press **Ctrl + Shift + P** to open the command palette. From the commands displayed, click the `RAB: Register RAB Bundle` command.

   Now, the VS Code extension registers the adapter on your Oracle Integration instance. The adapter is registered on the instance specified as *active* in the publisher profile.

   After the registration is complete, a success message is displayed in the VS Code editor's bottom panel, on the **Output** tab.

2. Log in to your Oracle Integration instance and check for the registered adapter.

a. On the Oracle Integration Home page, click **Show/Hide navigation menu** to view the navigation pane.

b. In the navigation pane, click **Design**, and then click **Adapters**.

c. On the Adapters page, click the **Search** icon, type the name of your adapter in the **Search** field, and hit **Enter**.

You'll see the adapter you registered in the search results. Point to the adapter's row and click **View** to view its details.

Next step: Test an Adapter

Test an Adapter

After pushing an adapter to an Oracle Integration instance, test the adapter to ensure it is functioning as expected.

Previous step: Register an Adapter for Testing

1. Gather the information that you need to test the adapter.

   For example:

   - Register a client application with the authorization server to generate a set of client credentials.
   - Get an access token and understand how to use it to access protected resources.

   For guidance, read the provider documentation.

2. Create and implement a test plan for your adapter.

   For guidance, see Checklist for Testing an Adapter.

Next step: Depending on your findings during the test, determine your next step.

- If you need to fix issues in the adapter: Review and Update an Adapter Definition Document
- If you need to add more functionality to the adapter: Add Functionality to an Adapter Definition Document
- If you want to publish the adapter to a production instance: Deploy an Adapter to Production

Checklist for Testing an Adapter

Use the guidelines provided here to create a test plan for your adapter built using the Rapid Adapter Builder.

At a minimum, you must perform the following tests:

- Review the Metadata
- Test Connections
- Test Integrations
- Identify Opportunities for Improvement
- Test the Documentation
- Complete Additional Testing as Needed

Review the Metadata

After pushing an adapter to Oracle Integration, check whether its metadata is accurate and easy to read.

1. In the navigation pane, click **Design**, then **Adapters**.
2. Find your adapter in the list.
3. Verify that the following information is correct:
   - Name of the adapter
   - Version
4. Point to the adapter, and select **Open Details** ⌄
5. Verify that the adapter details are correct and appear as you expect.

   For example:
   - Is all text accurate?
   - Are all links functional, and do they open the correct URLs?
   - Is all text fields a reasonable length?
   - Are there any typos?
6. Update the metadata as necessary, either now or after you finish testing.

   See Add Info Definitions.

Test Connections

Check whether you can create a connection, whether it meets your requirements, and whether the connection information that is accurate and easy to understand.

1. Create one or more connections that are based on the adapter.

   > **Tip:**
   >
   > To optimize the adapter and reduce the need for documentation and support, partner with a UX writer for this review.

   While creating the connection, verify the following:
   - Are you able to create a connection using your adapter?
   - Does the adapter address all the required scenarios?
   - Do the descriptions of each action make sense?
   - Review all the connection properties and security properties, as well as their corresponding tips.
     - Does the text appear as expected?
     - Does the text make sense?
     - Is the text a reasonable length?

       In general, shorter is better, but being understandable is most important.
     - Is the text consistent?

For instance, do some fields use verbs while others use nouns?

- Is terminology consistent and easy to understand?

  For instance, if some fields say **Change order** while others use **Order update**, users might get confused.

- Will the values for any connection or security properties be the same for all connections that developers create using the adapter? If so, set a default value and hide the property in the adapter definition document.

2. While creating mappings, verify the following:
   - Can you navigate the mapper easily, or is it overwhelming?
   - Are the mappings for the custom response easy to understanding? Are the properties easy to understand?
3. Create an integration that uses the connection(s).

   See Create an Integration in *Using Integrations in Oracle Integration 3*.
4. While testing, if you identified changes that you need to make in the adapter definition document, make the changes either now or after you finish testing.

   See Review a Connection Definition.

Test Integrations

Determine whether the integration runs as expected in runtime.

1. Test the integration in runtime.

   See Activate an Integration in *Using Integrations in Oracle Integration 3*.
2. Verify that the integration runs as expected as well as the following information:
   - Is the response easy to understand?

     For instance, an ideal response returns only the information that users need to know and nothing extra.
   - Did the integration do what it was supposed to do?

     If the integration creates a new record, ensure that the correct information is placed in each field. For example, make sure that the first and last name fields aren't swapped.
3. If the integration doesn't run as expected, determine why.

   For instance, the API call might have failed.
4. Update the adapter definition document as necessary, either now or after you finish testing.

   See

Identify Opportunities for Improvement

An adapter that is functional and consistent might still have room for improvement.

For example:

- Consider a situation where a user must enter the identifier for an object.

  When you test the adapter and enter an identifier, the connection and its integration function as expected. However, in the real world, users can't easily obtain this identifier. For instance, they might know only the display name. Or, you might be able to display only the relevant identifiers in a drop-down.
- Your adapter might not display custom properties.

For instance, international companies sometimes have country-specific HR software, each with different required fields for employee records. An adapter that works with only one country's HR software is of limited use. An adapter that works with all of the HR systems, including all of their custom fields, offers a more comprehensive solution.

Such capabilities aren't part of the adapter definition document that you convert from a Postman collection. Instead, you must update the adapter definition document for these requirements.

The opportunities for improvement can be immense and can lead to scope creep. Work with end users and product owners to identify and prioritize these opportunities for improvement.

Test the Documentation

Before you go live, test the documentation for the adapter against the user interface, and verify the following:

- Does documentation exist for the adapter?
- Is the documentation accurate?
- Are users able to use your documentation to create connections using the adapter?

See Publish the Adapter Documentation.

Complete Additional Testing as Needed

For instance, complete any of the following testing according to your organization's requirements:

- Functional testing
- Requirements testing
- User experience testing
- Performance testing

Add Functionality to an Adapter Definition Document

After you've successfully added and tested one functionality to your adapter, you can add support for another API request.

1. For Postman Collections, perform the following steps:

> **Note:**
>
> To update an adapter definition document with a new API request, you must first add the new request to your Postman collection, and export the collection to your workspace. See Create a Postman Collection Using API Requests.

   a. In VS Code, click **Explorer** on the sidebar.

      The directory structure of your workspace folder is displayed.

   b. Within the **misc** directory listing, right-click the updated collection file, and then click **RAB: Add Requests from Postman Collection**.

   c. In the Select request(s) pane, select the new request to add, and click **Done**.

2. For OpenAPI documents, perform the following steps:
   a. Import the updated OpenAPI document into the **api** directory of your workspace folder.
   b. In VS Code, click **Explorer** on the sidebar.

      The directory structure of your workspace folder is displayed.
   c. Within the **api** directory listing, right-click the updated OpenAPI document you imported, and click **Rename**.

      Rename the updated document as `openapi.resource.json`.

      > **Note:**
      >
      > You may not have to rename the document if you are using the same OpenAPI document imported earlier to select new requests.

   d. Right-click the document again, and then click **RAB: Update actions with OpenAPI document**.
   e. In the Select request(s) pane, select the new request to add, and click **Done**.

      Note that you can add new requests as well as remove requests that you had converted earlier by unchecking them. A summary of your selection is displayed at the top of the pane.
3. In the dialog box that appears, select either of the following options based on what you had chosen while converting the original Postman collection (or OpenAPI document) into an adapter definition document. See Generate an Adapter Definition Document.
   - If you had chosen to use the main document (`main.add.json`) earlier, click **Update main** to add the new request to the same document.
   - If you aren't using the main document, click **Save new**.

   > **Note:**
   >
   > If you choose **Save new**, the extension generates a new document and saves it in the `definitions` directory of your workspace. You can then rename this new document (containing updated requests) with the name of your earlier document to replace it.

   Now, the VS Code extension adds the new request to the document you select.
4. Press **Ctrl+S** on your keyboard to save the document.

Next step: Review and Update an Adapter Definition Document

5

Deploy an Adapter to Production

After you finish building and testing your adapter, you can push the adapter to a production instance of Oracle Integration, so that it's available to all integration developers. The following topics detail all the tasks that you must complete to release your adapter.

Topics:

- Determine an Adapter's Version Number
- Register an Adapter on Oracle Integration
- Publish the Adapter Documentation

Determine an Adapter's Version Number

Before you deploy an adapter to a production instance of Oracle Integration, determine and set a version number for the adapter.

- **For the first release:** If you are deploying an adapter for the first time on an Oracle Integration instance, set the version number to `1.0.0`. To edit the version number, see Set the Version Number of an Adapter.
- **For subsequent releases:** If you are releasing an update for an existing adapter on an Oracle Integration instance, perform the following tasks to determine the version number for the adapter:

  1. Using the VS Code extension, perform a version check. See Check the Version Number Recommendation.
     The extension makes a recommendation after analyzing the difference between the current and previous versions of the adapter definition document. The recommendation is in terms of the update type, for example: major, minor, or patch.

     > **Note:**
     >
     > Currently, the Rapid Adapter Builder doesn't allow a major version change.

  2. Translate the recommendation to a version number. See Types of Updates with Semantic Versioning.
  3. Update the version number in the adapter definition document. See Set the Version Number of an Adapter.

> **Note:**
>
> When you iteratively build your adapter, you may have updated the version number several times for test deployments of the adapter. Ensure that you change the adapter's version number to an appropriate value for the production deployment.

Check the Version Number Recommendation

The VS Code extension for Rapid Adapter Builder includes a version check that helps you determine how to update an adapter's version, according to the semantic versioning rules.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.

2. Within the **definitions** directory listing, right-click the adapter definition document, and then click **RAB: Version Check**.

   The extension compares the previous version of the adapter definition document with the current version.

   The results of the comparison are displayed in the VS Code editor's bottom panel, on the **Output** tab. For example, after the **Recommended version change** text, review the type of version update that you must make, such as `PATCH`, `MINOR`, or `MAJOR`. The `message` property includes the changes that you made.

   See Rules for the Semantic Version Check.

Rules for the Semantic Version Check

When you push an adapter to Oracle Integration, Oracle runs a semantic version check. This check determines whether your changes are maintaining backward compatibility and provides guidance for updating the version number of the adapter.

> **Note:**
>
> Currently, the Rapid Adapter Builder doesn't allow a major version change.

- About Semantic Versioning
- About the Semantic Version Check
- Types of Updates with Semantic Versioning
- Changes That Require a Major Version Update
- Changes That Require a Minor Version Update
- Changes That Require a Patch Version Update

About Semantic Versioning

Oracle enforces semantic versioning rules, also known as SemVer, for all adapters built using the Rapid Adapter Builder.

With semantic versioning, you increment your version numbers in a meaningful and controlled way. When users know that your adapter uses semantic versioning, they understand that the version number provides a quick explanation of the types of changes that are in the version.

To learn more, see the Semantic Versioning 2.0.0 website, or keep reading.

About the Semantic Version Check

When you publish an adapter, the semantic version check completes the following tasks.

Task	More information
Identify changes that break backward compatibility	To identify these changes, the semantic version check compares the previous and current versions of the adapter.
Mandates semantic versioning	The check ensures that yur version number increments appropriately for the type of changes that you've made. For example, if you make a minor release update to version 1.1.1, your next version must be 1.2 or higher.

Types of Updates with Semantic Versioning

Semantic versioning supports the following types of updates.

Type of update	Example	Guidance
Major versions	1.2.3 to 2.0.0	The change breaks backward compatibility or has the potential to break backward compatibility.
Minor versions	1.2.3 to 1.3.0	The change adds new features, such as new functions or parameters, while maintaining backward compatibility.
Patch versions	1.2.3 to 1.2.4	The change improves quality while maintaining backward compatibility.

Changes That Require a Major Version Update

> **Note:**
>
> Currently, the Rapid Adapter Builder doesn't support changes that break the backward compatibility.

The following changes conceptually break the backward compatibility. Whether a change actually breaks backward compatibility for a live integration depends on how integrations use your adapter. For example, if you remove an action from an adapter, you've broken backward compatibility. However, if no integrations are using the action, this change doesn't impact any integrations.

These changes require a major version update.

RuleID	Change
A01	Removing an action or trigger.
A02	Removing the input or output of a trigger or action.
A03	Changing the identifier of an action or trigger.
A04	Removing a property from the input or output schema of a trigger or action. **Note:** Semantic analysis is not possible when the flow determines the input and output schemas. As a result, when the flow determines the schemas, the change is considered backward compatible.
A05	Adding a required property or changing an existing property to be required for an input schema for a trigger or action.
A06	Changing the data type of an existing field, such as changing from string to number.

RuleID	Change
A07	Changing a connection property. Updates to some connection properties, including `displayName` and `description`, are not part of this check because these updates do not break backward compatibility.
A08	Changing a security policy.
A09	Changing a security property.

Changes That Require a Minor Version Update

RuleID	Change
B01	Adding an action or trigger
B02	Adding an optional field to the input or output schema of a trigger or action
B03	Changing an existing property from required to optional
B04	Adding or changing enumerated values You can restrict the input values for a property by providing a set of options that the integration developer chooses from. These options are called enumerated values.
B05	Adding a pattern matching regular expression You can use regular expression to specify the allowable pattern and text that an integration developer can enter for a property. If needed, you can add pattern enforcement to a field.
B06	Adding formatting or customizing the user interface appearance of a field You can specify the way that a field is rendered in the user face. For example, when a field has a format of date-time, the user interface displays a date picker.

Changes That Require a Patch Version Update

RuleID	Change
C01	Changing the logic of a flow in a backward-compatible manner without changing the input or output schema, such as an internal change to fix a defect
C02	Changing the properties that define the visual representation of the adapter, including title, description, tags, and icons
C03	Removing an enum from a field, thus allowing the field to accept a wider set of values
C04	Removing a format enforcing regular expression, thus allowing the field to accept a wider set of values
C05	Removing a pattern enforcing regular expression
C06	Changing a test reference when your adapter allows integration developers to test their connections.
C99	Making any other change that is considered backward compatible

Set the Version Number of an Adapter

After you have determined the version number for an adapter, update the number in the adapter definition document.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.

2. Within the **definitions** directory listing, click the required adapter definition document.

   The document is displayed in the VS Code editor.

3. In the editor, expand the `info` section, and update the `version` field with an appropriate number.

4. Press **Ctrl+S** on your keyboard to save the changes.

Register an Adapter on Oracle Integration

After you've set the appropriate release version in your adapter definition document, you're ready to register the adapter on a production instance.

For each adapter bundle, an Oracle Integration instance can have exactly one live version of an adapter. If you update an adapter bundle and register the latest version, the latest version of the bundle always overwrites the previous version.

> **Note:**
>
> Before you make the final push:
>
> - Specify the production instance of Oracle Integration as *active* in the publisher profile. See Configure the VS Code Extension for Rapid Adapter Builder.
> - Run a final validation check. See Validate an Adapter Definition Document.

1. Press **Ctrl + Shift + P** to open the command palette. From the commands displayed, click the `RAB: Register RAB Bundle` command.

   Now, the VS Code extension registers the adapter on your Oracle Integration instance.

   After the registration is complete, a success message is displayed in the VS Code editor's bottom panel, on the **Output** tab.

2. Log in to your Oracle Integration instance and check for the registered adapter.

   a. On the Oracle Integration Home page, click **Show/Hide navigation menu** to view the navigation pane.

   b. In the navigation pane, click **Design**, and then click **Adapters**.

   c. On the Adapters page, click the **Search** icon, type the name of your adapter in the **Search** field, and hit **Enter**.

   You'll see the adapter you registered in the search results. Point to the adapter's row and click **View** to view its details.

Publish the Adapter Documentation

When you publish an adapter, you must also publish the documentation for it to help integration developers work with the adapter.

You can use any adapter user guide created by Oracle as a template. See the Adapters page on the Oracle Help Center.

6

Download, Import, or Delete Adapters

Using the VS Code extension for Rapid Adapter Builder, you can download a registered adapter as a bundle file and import it into VS Code. You can also delete the adapters from an instance without leaving the VS Code environment.

Topics:

- Download or Delete Adapters
- Create or Import Adapter Bundles

Download or Delete Adapters

In VS Code, click **OIC Rapid Adapter Builder** on the sidebar to view all the registered adapters on an Oracle Integration instance.

The extension displays the adapters from the Oracle Integration instance set as *active* in the publisher profile. See Configure the VS Code Extension for Rapid Adapter Builder.

You can perform the following tasks in the Registered Adapters section.

1. **View all adapters:** View the list of registered adapters that is displayed. Click **Refresh** to refresh the list.
2. **Download adapter bundles:** Right-click a registered adapter, and then click **Download**.

   The corresponding adapter bundle is downloaded as an archive file with the `.rab` extension. The bundle contains all the adapter artifacts, such as the adapter definition document, the logo, and so on. Using the VS Code extension, you can seamlessly begin a new project by importing this bundle into a clean workspace. See Create or Import Adapter Bundles.
3. **Delete adapters:** Right-click a registered adapter, and then click **Delete** to delete an adapter from the instance.

Create or Import Adapter Bundles

Using the VS Code extension for Rapid Adapter Builder, you can create a bundle file for an adapter project and import the same into a new workspace to begin another project.

1. Create a bundle file for an adapter project.

   > **Note:**
   >
   > Before you begin, make sure your workspace is valid, with the correct folder and file structure. In addition, save all the files in your workspace.

   a. In VS Code, click **Explorer** on the sidebar.

b. Press **Ctrl + Shift + P** to open the command palette, and select the **RAB: Create RAB Bundle** command.

   An adapter bundle with the `.rab` extension is created under the root directory. The bundle contains all the adapter artifacts of your project, such as the adapter definition document, the logo, and so on.

2. Import an adapter bundle into a new workspace.

   You can import the following bundles into a workspace:

   - The adapter bundles that you create.
   - The registered-adapter bundles that you download from an Oracle Integration instance. See Download or Delete Adapters.

   a. In VS Code, click **Explorer** on the sidebar.

   b. Click **File** in the top ribbon, select **Open Folder...**, and choose a folder on your system as the new workspace.

   > **Note:**
   >
   > To import an adapter bundle, you must use an empty folder as your workspace.

   c. Press **Ctrl + Shift + P** to open the command palette, and select the **RAB: Import RAB Bundle** command.

   d. In the resulting window, select the `.rab` file to import, and click **Select**.

      The adapter bundle, containing unarchived files and folders, is imported into the workspace.

7

Guidelines for Oracle Partners

If your organization wants to distribute your adapters to all Oracle Integration users, you must complete additional onboarding and publishing tasks.

Topics:

- Introduction
- Build Adapters for the Integration Store
- Build Private Adapters

Introduction

If you are an Oracle Integration customer, partner, or independent software vendor, you can build application-specific adapters for your own use or make it available for others.

As an Oracle partner, you may want to build adapters to better support your own customers, or you may want to make your adapters available for any Oracle Integration customer and monetize the adapters, thereby creating new revenue streams. This document provides the guidelines for building adapters for Oracle Integration using the Rapid Adapter Builder.

You can build the following types of adapters:

- **Integration Store Adapters:** These adapters are available for use by any Oracle Integration customer. To develop these adapters, see Build Adapters for the Integration Store.
- **Private Adapters:** These adapters are available only on your or your customer's local Oracle Integration instance. There are no extensive guidelines to develop these adapters; however, you must follow some basic standards. See Build Private Adapters.

Build Adapters for the Integration Store

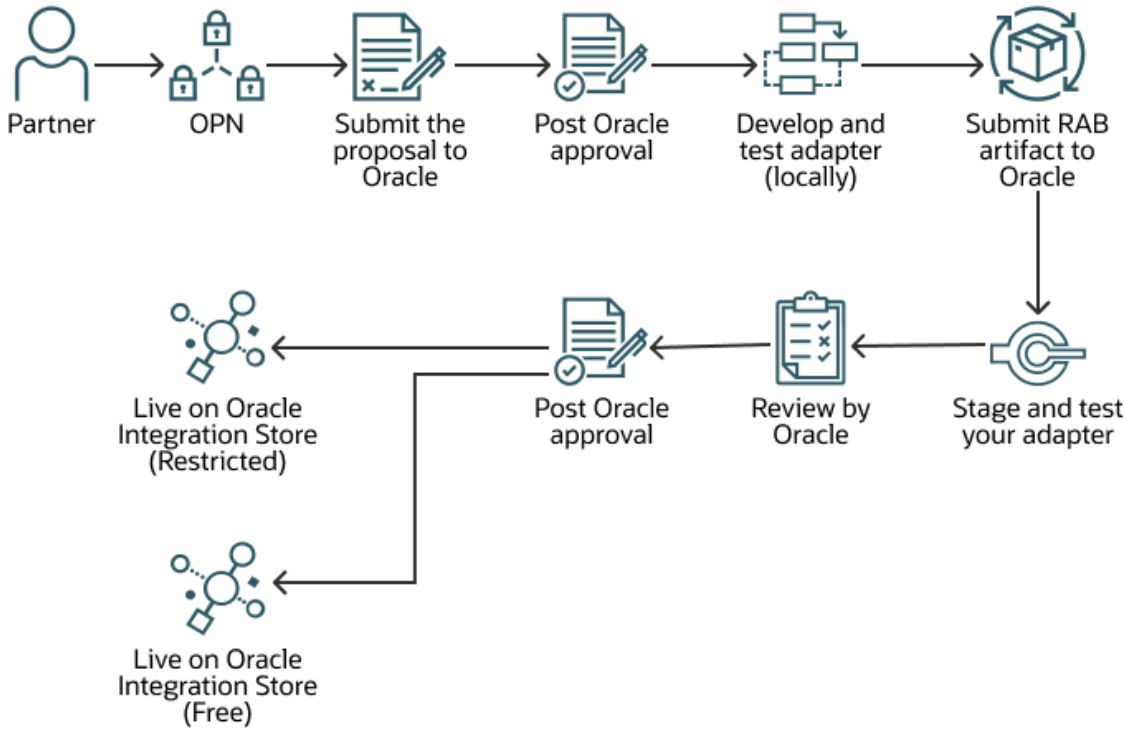
This section details the complete lifecycle of building adapters for the Integration Store of Oracle Integration. The lifecycle includes building, publishing, supporting, and maintaining your adapter.

Topics:

- Lifecycle Overview
- Get Started
- Build and Test Your Adapter
- Publish Your Adapter
- Support and Maintain Your Adapter

Lifecycle Overview

To make your adapter available on the Integration Store, you must join the Oracle Partner Network, submit your adapter proposal to Oracle, develop and test your adapter, and obtain the final approval from Oracle.



Get Started

Before you begin the adapter development process, you must perform the following tasks.

1. Register as a Partner on the Oracle Partner Network. Choose the build track.
2. Submit a proposal to Oracle with the details of the adapter you intend to develop. You should include the following details in your proposal:
   - The application for which you plan to build the adapter.
   - The business use cases for the adapter.
   - The support model for the adapter.
     Specify if you are adhering to the Oracle Support service level agreements (SLAs). Ensure that you provide detailed information in the terms and conditions.
   - The test plan for the adapter.
     Indicate if you plan to test the adapter for each new release of Oracle Integration.
   - The documentation plan for the adapter.
     You must adhere to the adapter documentation standards and templates shared by Oracle.
   - The monetization plan for the adapter.
3. After the proposal is approved by Oracle, you can begin the development process.

Build and Test Your Adapter

This section lists the guidelines to build an adapter for the Integration Store.

1. Build your adapter from an OpenAPI document or a Postman collection of an application's APIs. See Build and Test an Adapter.
2. Adhere to the adapter-development best practices. See Best Practices for Building an Adapter.
3. Design and execute automation tests for your adapter.

Publish Your Adapter

After you finish building and testing your adapter, follow the steps provided here to publish the adapter.

1. Send the adapter artifacts to Oracle for approval.

   You must send the `.rab` file to Oracle, adhering to the following guidelines:

   a. In the adapter definition document, enter the value for the `id` field (in the `info` section) as follows:

      - `<OPNID>:<adapterId>`
        Where `OPNID` is your Oracle Partner Network ID; this should not contain the hyphen (-) character.

   b. Use only the **RAB: Create RAB Bundle** command in VS Code to create the `.rab` file. See Create or Import Adapter Bundles.

   c. Ensure that the folder structure of your adapter bundle is according to the following illustration:

```
<OPNID>_<adapterId>-<xx.xx.xx>
|-->logo.svg
|-->definitions
   |-->main.add.json
|-->api
```

   d. Name the adapter bundle as: `<OPNID>_<adapterId>-<xx.xx.xx>.rab`.

      Where `OPNID` is your Oracle Partner Network ID; this should not contain the hyphen (-) character. Example name: 4712203150920_auraplayer-1.0.23.rab.

   e. Along with the adapter bundle, share the terms and conditions document, containing the following details:

      - The support model and the plan to address customer issues.
      - The amount of time you plan to provide support.
        For example: 24*7, or limited hours in a day, or limited days in a week, and so on.
      - The partnership plan with Oracle.
        For example, provide an estimate of how long you plan to continue partnering with Oracle and supporting the adapter.
      - The termination plan for the adapter.
        For example, if you terminate the partnership with Oracle, provide a plan to mitigate the issues for customers.

2. In addition to submitting the artifacts for a review, stage your adapter on the Integration Store, test it, and inform Oracle if the adapter can be made generally available (GA). Perform the following checks while staging your adapter:

   a. Verify if your instance has the following feature flag enabled: `oic.common.enable.staging-accelerators`. If not, please log an SR with the following information to enable it.

      - FF: `oic.common.enable.staging-accelerators`
      - region : <your instance's region>
      - URL: <your Oracle Integration URL>
      - OCID: <your Oracle Integration OCID>

   b. For the staged adapter to show up on your instance immediately, perform the invalidate cache operation on your instance. Otherwise, it may take around eight hours for the adapter to appear. An example URL to perform the invalidate cache operation is as follows:

      ```
      https://<OICDomainName>/ic/api/integration/v1/accelerators?
      integrationInstance=<InstanceName>&invalidateCache=true
      ```

   c. Ensure that you don't reuse an existing adapter ID. To set your adapter ID, see Step 1a above.

3. After the adapter artifacts are reviewed, Oracle makes your adapter GA on the Integration Store. Here are a few tasks related to the GA release:

   a. Export all the integrations (IAR or CAR files) that you created with the staged adapter, uninstall the adapter from your instance, and confirm the same to Oracle.

      This ensures that you don't lose any of the integrations you've built.

   b. Now, Oracle makes your adapter generally available and notifies you.

   c. To immediately view the adapter, perform the invalidate cache operation. Otherwise, the adapter appears on your instance after an auto refresh, which happens after eight hours.

   d. You can now import back the earlier integrations into your instance.

Support and Maintain Your Adapter

This section lists the support and maintenance guidelines for an adapter published to the Integration Store.

1. You must adhere to the support model you have mentioned in the terms and conditions document.
2. If the adapter is restricted to a customer or set of customers, you must create a separate contract with that particular customer or set of customers. Oracle will not be involved in such contracts.
3. For each new Oracle Integration release, patch the release to your development environment and run automation tests for all the use cases of your adapter.
4. Follow the best practices while fixing bugs or making enhancements to your adapter.
5. In case you identify an issue in Oracle Integration, create an SR with detailed logs and information.

Build Private Adapters

Here are the guidelines for building and maintaining adapters for your or your customer's private use.

1. Adhere to the adapter-development best practices. In addition, follow the best practices while fixing bugs or making enhancements to the adapter. See Best Practices for Building an Adapter.
2. For each new Oracle Integration release, execute automation tests for the adapter, so that it does not break due to changes in Oracle Integration.

Part III

Work with the Adapter Definition Document

An adapter definition document is a JSON-formatted document that describes the functional behavior and the implementation for an adapter. You build an adapter by authoring an adapter definition document.

Typically, you generate a draft version of the document using the VS Code extension for Rapid Adapter Builder. The extension can quickly create an adapter definition document from a Postman collection of an application's APIs. It automatically populates various components of the document from the data in the Postman collection as best it can. Thereafter, in the process of building your adapter, you can update this document to add more capabilities to it or to fix issues in it.

The following chapters provide guidelines on how to work with the different sections of an adapter definition document.

Topics:

- Use the VS Code Extension's Authoring Features
- Update Info Definitions
- Review Schemas
- Update Connection Definitions
- Update Action Definitions
- Update Trigger Definitions
- Update Flow Definitions
- Update Category Definitions

8

Use the VS Code Extension's Authoring Features

To make it easy to work with an adapter definition document, the VS Code extension offers several authoring features, such as JSON schema validation, code hinting, code snippets as templates, and so on.

You can use these features when you work with different sections of the document.

Topics:

- Validate Documents Against a Schema
- Use Code Templates
- View the VS Code Extension Logs

Validate Documents Against a Schema

The VS Code extension for Rapid Adapter Builder has a JSON schema packaged with it. The extension validates all adapter definition documents it generates against the schema.

If a file name ends with the suffix `.add.json`, the extension automatically validates the document against the JSON schema. You can review the validation errors, if any, in the VS Code editor.

Using the schema-validation feature, you can perform several tasks, such as viewing error highlights, viewing descriptions for valid keys, and viewing code hints.

To perform these tasks, open an adapter definition document in the VS Code editor.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.

   > **Note:**
   >
   > The file name of the document you select must have the suffix `.add.json`.
3. See the following topics for details on each schema-validation feature and how to use it:
   - View Error Highlights
   - View Descriptions for Valid Keys
   - View Code Hints

View Error Highlights

If a key or value in your document does not comply with the vocabulary described in the JSON schema, the VS Code extension highlights the violations.

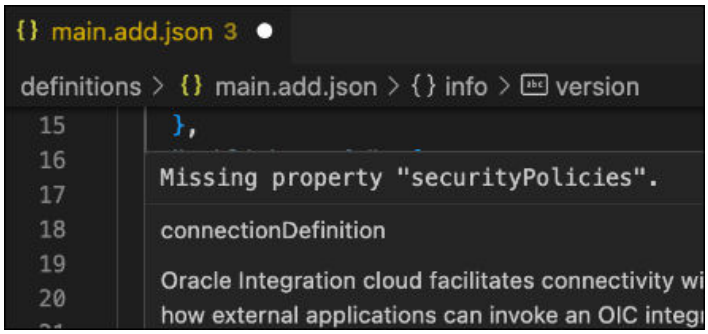
1. In the VS Code editor, check for error highlights in your document.
2. Fix the highlighted violations, if any.

Here are some examples of schema-validation errors:

- **Invalid Key**

- **Invalid Value**

- **Missing Required Property**

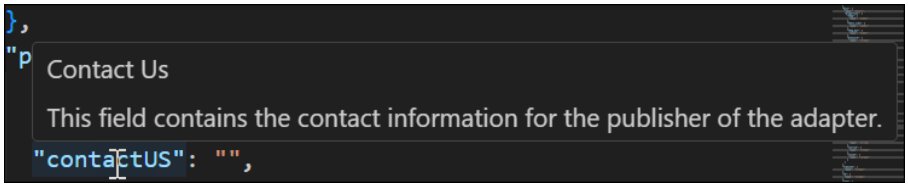


View Descriptions for Valid Keys

If a key in your document is valid (that is, complaint with the vocabulary in the JSON schema), the VS Code extension provides a description for the key. This description is displayed as a tooltip.

1. In your document, point to a valid key to view its description.
2. Enter a value for the key according to the information in the description.

Here's an example description:



View Code Hints

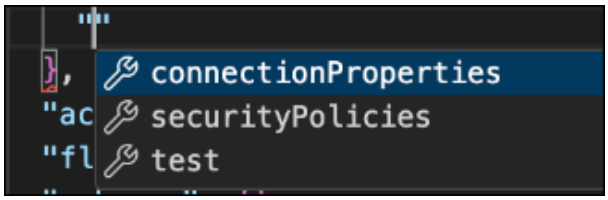
The VS Code extension can suggest relevant keys or values to add to your document at a particular location. It suggests keys or values that are driven by the packaged JSON schema.

1. To view code hints in your document, enter the double quotation mark (") at the location where you want to add a new key or value.
2. Select an appropriate key or value from the suggestions.

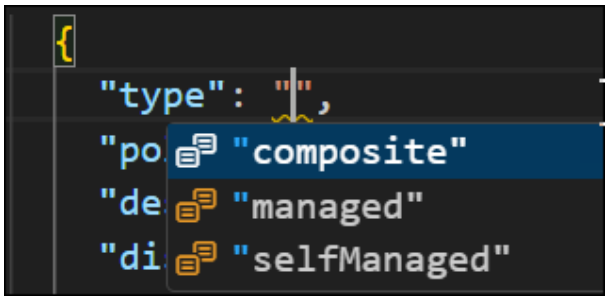
> **Note:**
>
> For the suggestions to appear for a value field, the value expected needs to be of an enumerated type.

Here are a few example suggestions:

- **Key Hints**



- **Value Hints**



Use Code Templates

The VS Code extension for Rapid Adapter Builder provides code snippets as templates to help you author an adapter definition document.

There are templates available for various sections of the adapter definition document, such as connections, triggers, actions, and so on. To author a section in a document, you can use the corresponding template and update it with your details, without having to write the code from scratch.

To use a code template, open an adapter definition document in the VS Code editor.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. See the following topics for details on each code template and how to use it in your document:
   - Use Authentication Scheme Templates
   - Use the Test Connection Template
   - Use the Action Code Template
   - Use the Trigger Code Template

Use Authentication Scheme Templates

The VS Code extension provides code templates for a number of authentication schemes. You can use these templates to quickly define a security policy in the `connection` section of an adapter definition document.

Select and use the required authentication template for your document.

1. Right-click anywhere in your document to view the context menu, and then click **RAB: Insert Authentication Scheme**.

   A list of authentication schemes is displayed.
2. Select an authentication scheme from the list to insert the corresponding template into your document.

   Under the `connection` object of your document, the `connectionProperties` and `securityPolicies` sections are populated as per the template.
3. Update the values in these sections according to your requirements.

> **Note:**
>
> Consider that you model an authentication scheme for your adapter and deploy the adapter to Oracle Integration. At a future time, if the target application for your adapter deprecates the authentication scheme you have used, append a new authentication scheme in your adapter definition document. Do not delete the earlier authentication scheme as it breaks the backward compatibility. Instead, you can rename the earlier authentication scheme to indicate that it has been deprecated.

Use the Test Connection Template

The VS Code extension provides a code template for testing a connection. You can use this template to quickly define a test-connection flow in the `flows` section of an adapter definition document.

1. Right-click anywhere in your document to view the context menu, and then click **RAB: Insert Test Connection Flow**.
2. In the dialog box that appears, click **Yes** to confirm your action.

   A test statement is inserted in the `connection` section, and a corresponding flow is inserted in the `flows` section of the document.
3. Update the values in the inserted flow according to your requirements.

Use the Action Code Template

The VS Code extension provides a code template for action definitions. You can use this template to quickly define an action in the `actions` section of an adapter definition document.

1. Right-click anywhere in your document to view the context menu, and then click **RAB: Insert New Action**.
2. In the **Enter an Action ID** dialog box, enter a name for the action and press **Enter** on your keyboard.

   A new action definition is inserted in the `actions` section, and a corresponding flow is inserted in the `flows` section of the document.
3. Update the values in these sections according to your requirements.

Use the Trigger Code Template

The VS Code extension provides a code template for trigger definitions. You can use this template to quickly define a trigger in the `triggers` section of an adapter definition document.

1. Right-click anywhere in your document to view the context menu, and then click **RAB: Insert New Trigger**.
2. In the **Enter a Trigger ID** dialog box, enter a name for the trigger and press **Enter** on your keyboard.

   A new trigger definition is inserted in the `triggers` section of the document.
3. Update the values in this section according to your requirements.

View the VS Code Extension Logs

While authoring and publishing your adapter definition document, you can view the logs of the VS Code extension in the editor's bottom panel, on the **Output** tab.

To view the extension logs:

1. In VS Code, click **View** in the top ribbon, and then click **Output** to bring up the panel area with the **Output** tab selected.
2. In the **Log Filter Selector** drop-down list of the panel, select **OIC Rapid Adapter Builder**.
3. To change the log level:
   a. Click **File** in the top ribbon, then select **Preferences**, and then **Settings**.
   b. In the **Search** field, type `rab`.
   c. In the **RAB: Log Level** field, select an option of your choice.

9

Update Info Definitions

The `info` section of an adapter definition document contains information about the adapter's identity, the publisher of the adapter, and the functional capabilities of the adapter. You can update this information as needed for your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Info Definitions Where Info Definitions Appear in Oracle Integration
Review the properties and see sample code	Info Properties and Sample Code
Define the adapter's information	Add Info Definitions Change the Logo of an Adapter

Learn About Info Definitions

Before updating the `info` section of an adapter definition document, familiarize yourself with this section and its properties.

Topics:

- Introduction to Info Definitions
- Where Info Definitions Appear in Oracle Integration
- Info Properties and Sample Code

Introduction to Info Definitions

While developing your adapter, you specify information about the branding and identity of the adapter, the publisher details, and the supported capabilities in the `info` section of an adapter definition document.

As a basic requirement, you must specify the following information for an adapter in the `info` section:

- A unique ID
- A display name and description, which appear in Oracle Integration.
- A version number according to the semantic versioning rules. To determine the version number for your adapter, see Determine an Adapter's Version Number.

> **Note:**
>
> If you have already published an adapter definition document to an Oracle Integration instance, the Rapid Adapter Builder doesn't allow you to publish an updated document without an update to the version number.

Additionally, you can provide your organization's information as the publisher details and make security-related settings. As a best practice, you should update the security settings, so that the data of the adapter's users is handled securely.

For the full list of properties in the `info` section and their descriptions, see Info Properties and Sample Code.

Where Info Definitions Appear in Oracle Integration

In Oracle Integration, you can view the identity and publisher information of an adapter on the Adapters page.

On the Adapters page, point to an adapter's row and click **View** to view the adapter's details, such as name, description, version, and publisher information.

Info Properties and Sample Code

The `info` section of an adapter definition document typically contains the properties listed here and has the following sample structure.

- Properties
- Sample Code

Properties

The `info` section includes the following properties.

Property	Description
`id`	Unique identifier of the adapter. **Note:** Ensure that the `id` value is prefixed with a namespace and is separated from the identifier with a colon. For example: `"id" : "oracle:zuora"`, where `oracle` is the namespace.

Property	Description
`type`	Type of application that the adapter connects to. Choose from the following values: • `Technology` • `Enterprise Messaging` • `Storage` • `Third Party` • `IaaS Services (Third Party & OCI)` • `Database` • `Block Chain`
`displayName`	Name of the adapter. This name appears in several places on the user interface of Oracle Integration, for example, the adapter registry page, and connection creation and selection related pages. You can localize this field, and also modify the name if the third party decides on an identity or a name change.
`description`	Description of the adapter. This value appears in Oracle Integration on the Adapters page, which is within the Design page.
`version`	Version of the adapter. While specifying the version, ensure to follow the semantic version rules where the first digit represents a major version, which is backward incompatible with the last major version. The next digit represents backward compatible functional additions and the third digit presents fixes that do not change the contract of the adapter actions and triggers. For more information on semantic versioning, see https://semver.org/. When you update the version number, the Rapid Adapter Builder platform validates it against the existing adapter version. for more information, see Rules for the Semantic Version Check.
`specVersion`	Version of the language that the adapter definition document uses. This field indicates to the Rapid Adapter Builder which language version to execute.
`categories`	Category for the application for which you're creating an adapter. The category determines how the adapter is organized within Oracle Integration. Choose from the following values: • `ERP` • `HCM & HRMS` • `CRM & CX` • `Industries` • `On-premises Enterprise` • `SOA Connectivity` • `Social and Productivity` • `Database` • `Supply Chain Management` • `E-Commerce` • `Enterprise Messaging` • `Technology` • `Health Care` • `IaaS Services & OCI Services`

Property	Description
`appInfo`	Information about the adapter. This information typically is relevant if you are part of the Oracle Partner Network and plan to distribute the adapter outside your organization. The following properties are available: • `name`: Organization, application, or technology that the adapter provides integration capabilities for. Ensure that the name complies with the third party branding requirements. This name appears on in the adapter registry metadata. For example, Asana or SurveyMonkey. • `description`: Description of the adapter's integration capabilities. • `contactUS`: Contact information of the organization that developed the adapter. • `supportURL`: URL that users can use to request for support or report issues related to the adapter. • `documentationURL`: Web address that points to the detailed documentation of the adapter.
`publisherInfo`	Information about the organization that developed the adapter (typically the organization that you work for). The following properties are available: • `name`: Name of the organization that owns and maintains the adapter. Ensure that the name complies with third party branding requirements. This name appears in the adapter registry metadata. For example, `Advanco` or `Auraplayer`. • `description`: Description of the organization that owns the adapter. This description appears in the adapter registry metadata. • `supportURL`: URL that users can use to request for support or report issues related to the adapter. • `documentationURL`: Web address that points to the detailed documentation of the adapter.

Property	Description
`settings`	Information about the adapter's security settings and functional capabilities. • `supportsConnectivityAgent`: Either `true` or `false`, indicating whether the adapter can access an on-premises environment by using the connectivity agent. • `supportsRegeneration`: Either `true` or `false`, indicating whether the adapter supports the regeneration of a configured endpoint. When you regenerate an endpoint, the schemas for a configured action are refreshed with new schema information for the request and response based on changes to the back-end metadata object model of the third party. See Refresh Endpoints for Integrations in *Using Integrations in Oracle Integration 3*. • `_debug`: Either `true` or `false`, indicating whether Oracle Integration captures the execution logs for background integration flows, which may be required while configuring fields or operations of the adapter. See Debug Integration Flows in Designtime. • `allowNonSSL`: Either `true` or `false`, indicating whether the adapter can make outbound requests to endpoints that are not protected by SSL. • `disableOutboundPayloadSanitization`: By default, Oracle Integration encodes the JSON data while sending a request or receiving a response to prevent injection attacks. To disable this behavior, set this property to `true`. For example, when this property is set to `true`, an integration developer can customize a connection to inject a script into an application. – When this property is set to `false`, the outbound payload that is sent to a target application, including requests and responses, is sanitized and doesn't include injected scripts. – When this property is set to `true`, the outbound payload that is sent to a target application, including requests and responses, is not sanitized and is allowed to include injected scripts. • `allowedDomains`: Array of the domains that are allowed for outbound requests. Every domain that is specified in the `flows` and `connection` sections must also be specified in this property. See Restrict Outbound Invocations to Specific Domains. • `followRedirects`: Reserved for future use.

Sample Code

```
{
  "info": {
    "id": "cloud:microsoft-task",
    "displayName": "Microsoft To-do Task",
    "description": "Microsoft To-do Task is a cloud-based task management
service.",
    "version": "1.0.0",
    "specVersion": "1.0",
    "categories": [],
    "appInfo": {
      "name": "Oracle Corporation",
      "description": "https://www.oracle.com/about",
      "contactUS": "http://www.oracle.com/us/corporate/contact/index.html",
      "supportURL": "http://otn.oracle.com",
      "documentationURL": "https://www.oracle.com/pls/topic/lookup?
ctx=appintICSGM-GUID-66A8F995-D32F-420B-9250-290B7D210FBB"
    },
```

```
    "publisherInfo": {
      "name": "Oracle Corporation",
      "description": "https://www.oracle.com/about",
      "contactUS": "http://www.oracle.com/us/corporate/contact/index.html",
      "supportURL": "http://otn.oracle.com",
      "documentationURL": "https://www.oracle.com/pls/topic/lookup?
ctx=appintICSGM-GUID-66A8F995-D32F-420B-9250-290B7D210FBB"
    },
    "settings": {

    }
  }
}
```

Work with Info Definitions

Edit an adapter definition document to add information about the adapter's identity, the publisher of the adapter, and its functional capabilities.

What Do You Want to Do?

Task	More information
Update the info definitions of an adapter	**Required:** Add Info Definitions
Review other common updates to adapter's branding	**Optional:** Change the Logo of an Adapter

Add Info Definitions

Add the details for your adapter in the `info` section.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `info` section in the document.
4. Add the basic information for your adapter, such as the ID, display name, description, version, and publisher information.
5. Specify the functional capabilities and security settings for your adapter in the `settings` object.

   For more information about each field in the `info` section, see Info Properties and Sample Code

The following topics provide the procedure to make some common updates to the `settings` properties:

- Restrict Outbound Invocations to Specific Domains
- Allow Non-HTTPS Traffic

- Debug Integration Flows in Designtime

Restrict Outbound Invocations to Specific Domains

An adapter developer can restrict outbound invocations only to allow-listed domains in order to establish trust with the integration developers who'll use the adapter.

After the developer specifies the allow-listed domains in the adapter definition document, the Rapid Adapter Builder enforces the list to all the outbound invocations, like design-time, runtime, OAuth policies, and so on.
While specifying the domains, ensure that the domains conform to any of the following patterns:

- .<secondLevelDomain>.<topLevelDomain>,
- .<topLevelDomain>,
- Any sub-resource of .<secondLevelDomain>.<topLevelDomain> (eg: *.a.b.c, a.b.c),
- Valid IP,
- Valid IP+Port

> **Note:**
>
> - The Rapid Adapter Builder supports only the wildcard (*) character in the domain name.
> - You can specify a maximum of ten domains.
> - A generic domain value like * , *.com lowers the security score of the adapter.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `info` section in the document, and scroll to the `settings` object.
4. Update the `allowedDomains` property by adding the list of allowed domains as a JSON array, and press **Ctrl+S** to save the document.

Here is an example scenario to understand allowed domains better.

If you are developing an adapter for an application, for example, myDemoApp, the adapter will access the following:

- The application's APIs hosted on the domain, `https://myDemoApp.com`
- The API URLs, like `https://auth.myDemoApp.com/v1/token`, or `https://identity.myDemoApp.com/v1/users`, and so on

Now, if you want to restrict access only to the myDemoApp APIs from the adapter, you can set the `allowedDomains` property.

Sample Code:

```
"info": {
  "settings": {
  ...
      "allowedDomains" : ["*.myDemoApp.com"]
}
```

Allow Non-HTTPS Traffic

By default, the adapters built using the Rapid Adapter Builder allow HTTPS-only outbound calls from the design-time or runtime environments, and according to the defined security policies.

To test an adapter in the development environment, the adapter must also handle HTTP (non-HTTPS) traffic.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `info` section in the document, and scroll to the `settings` object.
4. Set the `allowNonSSL` property to `true`.
5. Press **Ctrl+S** to save the document.

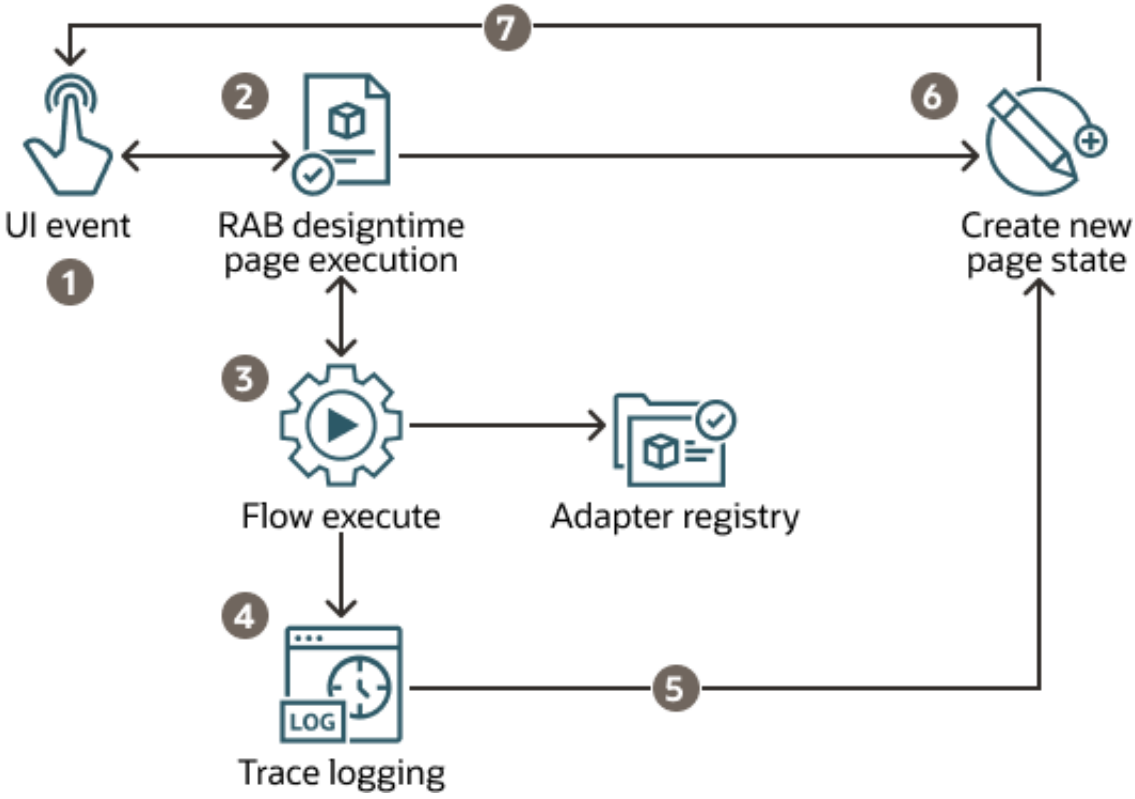
Debug Integration Flows in Designtime

While configuring an adapter (built using the Rapid Adapter Builder) in an integration flow, you may have a configuration field or operation that requires execution of an additional flow to fetch data from external applications.

In the Oracle Integration designtime user interface (integration canvas), you can monitor and debug the execution of such background flows, with the help of browser tools. You can view the milestone information during the execution of a background flow and use this information to debug errors, if any. The information you see in designtime is consistent with the information that is presented at run time.

> **Note:**
>
> You can view the execution logs of background flows in designtime only while configuring adapters that are built using the Rapid Adapter Builder.

While configuring your adapter, if a UI event occurs, an event call is issued to the backend. The Rapid Adapter Builder designtime framework executes the page logic. If the configuration field of the adapter requires the execution of a background flow, the flow execution engine reads the adapter registry to check if the `debug` flag is turned on for the adapter. If the flag is turned on, the tracing is executed for the background flow, and the milestone information is appended in the response to the web client. You can monitor this milestone information using browser tools.



Configure Designtime Debugging
To view the execution logs of background flows in Oracle Integration designtime, you must turn on the `_debug` flag in the adapter definition document of the respective adapter.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `info` section in the document, and scroll to the `settings` object.
4. Set the `_debug` flag to `true`.
5. Press **Ctrl+S** to save the document.

View the Execution Logs in Designtime
While configuring the adapter's fields or operations in the integration canvas of Oracle Integration, open the developer tools in your browser. For example, in Google Chrome, click the vertical ellipsis, then **More tools**, and then **Developer tools**.

In the developer tools window:

1. With the **Network** tab selected, click the required request on the left.
2. View and debug the milestone information on the **Response** tab.
   The milestone log information is shown as the last node of the JSON response.

Sample Response

Here's an example response resulting from the execution of a background flow, while configuring an adapter built using the Rapid Adapter Builder.


```
{
  "metaData": {
    "viewMode": false,
    "pages": [
      {
        "pageId": "welcome",
        "pageName": "Basic Info",
        "pageTitle": "Welcome to the Cloud Endpoint Configuration Wizard",
        "welcomeText": "This wizard helps you create a service using
Metadata. \nYou will be asked to specify configuration parameters and define
an action for the service.",
        "helpId": "HelpId",
        "disabled": false
      },
      {
        "pageId": "configuration",
        "pageName": "Configuration",
        "pageTitle": "Configuration",
        "welcomeText": "Configuration",
        "helpId": "GUID-139B596F-500D-4F59-9F1E-6B432AC0BE4D",
        "disabled": false
      },
      {
        "pageId": "summary",
        "pageName": "Summary",
        "pageTitle": "Cloud Endpoint Configuration Summary",
        "welcomeText": "Cloud Adapter configuration was successful.",
        "helpId": "summary",
        "disabled": false
      }
    ],
    "currentPageId": "configuration",
    "pluginId": "cloud:google-sheets"
  },
  "editFields": [
    {
      "labelFieldLayout": "twoRowFields",
      "helpText": null,
      "rowIdentifier": -1,
      "oneRowLabelFieldAlignment": "LEFT_LEFT",
      "triggerNext": false,
      "name": "spreadsheetId",
      "object": {
        "filterRequired": false,
        "filterPlaceHolderText": null,
        "groupFilterObject": null,
        "width": null,
        "filterHelpText": null,
        "itemDescriptionLabel": null,
        "filterText": null,
        "previousSelection": null,
        "size": 10,
        "type": "comboBoxObject",
```

```
      "groupFilterRequired": false,
      "displayItemDescription": false,
      "displayMode": 0,
      "options": [
        {
          "value": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
          "type": null,
          "formattedValue": "Customer Managment"
        },
        {
          "value": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
          "type": null,
          "formattedValue": "HL7 Message - FHIR R4: VXU_V04"
        },
        {
          "value": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
          "type": null,
          "formattedValue": "Copy of Restricted Stock Unit (RSU) Calculator"
        },
        {
          "value": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
          "type": null,
          "formattedValue": "Development Management"
        },
        {
          "value": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
          "type": null,
          "formattedValue": "Event Attendee List"
        },
        {
          "value": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
          "type": null,
          "formattedValue": "HR Management"
        },
        {
          "value": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
          "type": null,
          "formattedValue": "Financial Managment"
        },
        {
          "value": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
          "type": null,
          "formattedValue": "UAT-File"
        },
        {
          "value": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
          "type": null,
          "formattedValue": "SalesAccounts"
        }
      ],
      "value": null,
      "hasEvent": true
    },
    "required": true,
    "description": "",
    "disabled": false,
```

```
      "basic": true,
      "advanced": false,
      "label": "Spreadsheet Name"
    }
  ],
  "logs": [
    {
      "time": "2024-08-12T18:56:30.604071223",
      "flow": "spreadsheetIdFlow",
      "state": "startState",
      "summary": "State Execution Start",
      "input": {},
      "context": {
        "connectionProperties": {
          "applicationVersion": "1.0",
          "serviceInstanceOcid": "ocid5.exampleinstance.oc1.region.jjack",
          "pluginID": "cloud:google-sheets",
          "managedSecurity": "true",
          "integration_role": "TARGET",
          "oauth.scope": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
          "useProxy": "true",
          "securityPolicy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "proxyHost": "10.xx.xy.yy",
          "oauth.client.id": "813090821001-
uqpc1lsvauooc3lv8j3bh52cfs6kmfl4.apps.googleusercontent.com",
          "httpsProxyPort": "80",
          "oauth.client.secret": "gsYK8nYxxxxKx5p111188daJ",
          "oauth.auth.code.uri": "https://accounts.google.com/o/oauth2/auth",
          "oauth.access.token.uri": "https://www.googleapis.com/oauth2/v4/
token",
          "proxyPort": "80",
          "RABEngineVersion": "24.10",
          "csfMap": "example.wsm.security",
          "useCache": "true",
          "onPremise": "false",
          "clientAuthentication": "client_credentials_as_body",
          "httpsProxyHost": "10.xx.xy.yy",
          "csfkey":
"fac648317448763xxxxx2efd61a50c80243xxxxx2634ad87d396yyy7e5189038"
        },
        "input": {},
        "cloudAdapterUIBindingId": "04881642-6e72-8744-51f6-bea7270bf9bf",
        "configuration": {},
        "self": {
          "info": {
            "id": "cloud:google-sheets",
            "displayName": "Google Sheets",
            "description": "Google Sheets is a cloud-based sheet management
service.",
            "version": "1.0.0",
            "specVersion": "1.0",
            "categories": [],
            "appInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
```

```
                "contactUS": "http://www.example.com/corp/contact/index.html",
                "supportURL": "http://support.example.com",
                "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
              },
              "publisherInfo": {
                "name": "Example Corporation",
                "description": "https://www.example.com/about",
                "contactUS": "http://www.example.com/corp/contact/index.html",
                "supportURL": "http://support.example.com",
                "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
              },
              "settings": {
                "supportsConnectivityAgent": true,
                "_debug": true
              }
            },
            "connection": {
              "connectionProperties": [],
              "securityPolicies": [
                {
                  "type": "managed",
                  "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
                  "description": "Google Authorization Policy",
                  "displayName": "Google Authorization Policy",
                  "scope": "ACTION",
                  "securityProperties": [
                    {
                      "name": "oauth.client.id",
                      "displayName": "Google Client ID",
                      "description": "Google Client ID",
                      "shortDescription": "Google Client ID",
                      "hidden": false,
                      "required": true
                    },
                    {
                      "name": "oauth.client.secret",
                      "displayName": "Google Client Secret",
                      "description": "Google Client secret",
                      "shortDescription": "Google Client secret",
                      "hidden": false,
                      "required": true
                    },
                    {
                      "name": "oauth.auth.code.uri",
                      "default": "https://accounts.google.com/o/oauth2/auth",
                      "hidden": true,
                      "required": true
                    },
                    {
                      "name": "oauth.access.token.uri",
                      "default": "https://www.googleapis.com/oauth2/v4/token",
                      "hidden": true,
                      "required": true
                    },
```

```
                    {
                      "name": "oauth.scope",
                      "default": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
                      "hidden": true,
                      "required": true
                    },
                    {
                      "name": "clientAuthentication",
                      "default": "client_credentials_as_body",
                      "hidden": true,
                      "required": true
                    }
                  ],
                  "authExtension": {
                    "authorizationURL": {
                      "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
                      "params": {
                        "query": {
                          "response_type": "code",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "scope": "${.securityProperties.\"oauth.scope\"}",
                          "access_type": "offline",
                          "approval_prompt": "force"
                        }
                      }
                    },
                    "accessTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "query": {
                          "code": "${auth_code}",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                          "grant_type": "authorization_code"
                        }
                      },
                      "headers": {
                        "Content-Type": "application/x-www-form-urlencoded",
                        "Authorization": "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
                      },
                      "body": "false"
                    },
                    "refreshTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
```

```
                    "params": {
                      "template": {},
                      "query": {
                        "refresh_token": "${refresh_token}",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                        "grant_type": "refresh_token"
                      }
                    },
                    "headers": {
                      "Content-Type": "application/x-www-form-urlencoded"
                    },
                    "body": "false"
                  },
                  "fetchRules": {
                    "auth_code": "code",
                    "access_token": "access.[tT]oken",
                    "refresh_token": "refresh.[tT]oken",
                    "expiry": "expires.*",
                    "token_type": "token.?[tT]ype"
                  },
                  "accessTokenUsage": {
                    "headers": {
                      "Authorization": "Bearer ${access_token}"
                    }
                  }
                }
              }
            ],
            "test": "flow:testConnectionFlow"
          },
          "schemas": {
            "Sheet": {
              "type": "object",
              "properties": {
                "properties": {
                  "type": "object",
                  "properties": {
                    "sheetId": {
                      "type": "integer"
                    },
                    "title": {
                      "type": "string"
                    },
                    "index": {
                      "type": "integer"
                    },
                    "sheetType": {
                      "type": "string"
                    }
                  }
                }
              }
            },
```

```
"Spreadsheet": {
  "type": "object",
  "properties": {
    "spreadsheetId": {
      "type": "string"
    },
    "spreadsheetUrl": {
      "type": "string"
    },
    "properties": {
      "type": "object",
      "properties": {
        "title": {
          "type": "integer"
        },
        "locale": {
          "type": "string"
        },
        "autoRecalc": {
          "type": "integer"
        },
        "timeZone": {
          "type": "string"
        }
      }
    },
    "sheets": {
      "type": "array",
      "items": {
        "$ref": "#/schemas/Sheet"
      }
    }
  }
},
"getSpreadsheetInput": {
  "type": "object",
  "properties": {
    "spreadsheetId": {
      "type": "string"
    }
  }
},
"createSpreadsheetInput": {
  "type": "object",
  "properties": {
    "spreadsheetTitle": {
      "type": "string"
    },
    "sheets": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "sheetTitle": {
            "type": "string"
          }
```

```
            }
          }
        }
      }
    },
    "insertRowOutput": {
      "type": "object",
      "properties": {
        "spreadsheetId": {
          "type": "string"
        },
        "tableRange": {
          "type": "string"
        },
        "updates": {
          "type": "object",
          "properties": {
            "spreadsheetId": {
              "type": "string"
            },
            "updatedRange": {
              "type": "string"
            },
            "updatedRows": {
              "type": "integer"
            },
            "updatedColumns": {
              "type": "integer"
            },
            "updatedCells": {
              "type": "integer"
            }
          }
        }
      }
    },
    "getRowInput": {
      "type": "object",
      "properties": {
        "rowNumber": {
          "type": "integer"
        }
      }
    }
  },
  "actions": {
    "insertRowAction": {
      "displayName": "Insert Row Into Sheet",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
```

```
    },
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      }
    ]
  },
  "getRowAction": {
    "displayName": "Get Row From Sheet",
    "description": "This action get a row from sheet.",
    "execute": "flow:getRowFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/getRowInput"
      }
    },
    "output": "flow:sheetSchemaFlow",
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
```

```
        }
      },
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "currentRow",
            "displayName": "Get Specific Row"
          },
          {
            "keyName": "lastRow",
            "displayName": "Get Last Row"
          }
        ]
      }
    ]
  },
  "createSpreadsheetAction": {
    "displayName": "Create Spreadsheet",
    "description": "This action creates a new spreadsheet.",
    "execute": "flow:createSpreadsheetFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/createSpreadsheetInput"
      }
    },
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/Spreadsheet"
      }
    }
  },
  "insertRowActionErrorEngine": {
    "displayName": "Insert Row Into Sheet(Engine Error for Demo)",
    "description": "This action insert a new row into sheet.",
    "execute": "flow:insertRowFlow",
    "input": "flow:sheetSchemaFlow",
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/insertRowOutput"
      }
    },
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
```

```
                    "required": true
                  },
                  {
                    "name": "sheetId",
                    "displayName": "Sheet Name",
                    "description": "",
                    "type": "COMBO_BOX",
                    "options": "flow:sheetIdFlowErrorEngine",
                    "required": true,
                    "dependencies": {
                      "spreadsheetId": {
                        "values": []
                      }
                    }
                  }
                ]
              }
            },
            "flows": {
              "insertRowFlow": {
                "id": "insertRowFlow",
                "description": "insertRowFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "insertRowAction",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}:append",
                            "method": "POST",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                              "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                              "insertDataOption": "INSERT_ROWS",
                              "valueInputOption": "RAW"
                            },
                            "body": "${ {range: (.configuration.sheetId + \"!
A1:Z1\"), majorDimension:\"ROWS\", values: .input | to_entries |
[map(.value)] } }"
```

```
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "getRowFlow": {
              "id": "getRowFlow",
              "description": "getRowFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                },
                {
                  "name": "convertResult",
                  "type": "expression",
                  "operation": "(if .configuration.rowOperation ==
\"lastRow\" then (.lastLineNumber-1) else (.input.rowNumber-1) end)
as $rowNum | .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z40000\" }",
                            "majorDimension": "ROWS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values | length }",
                        "toStateData": "${ .lastLineNumber }"
```

```
                    }
                  },
                  {
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                        "method": "GET",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ if .configuration.rowOperation ==
\"lastRow\" then .configuration.sheetId + \"!A1:Z\" + (.lastLineNumber|
tostring) else .configuration.sheetId + \"!A1:Z\" + (.input.rowNumber|
tostring) end }",
                          "majorDimension": "COLUMNS"
                        }
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body.values }",
                      "toStateData": "${ .values }"
                    }
                  },
                  {
                    "functionRef": "convertResult",
                    "actionDataFilter": {
                      "toStateData": "${ .output }"
                    }
                  }
                ],
                "end": true
              }
            ]
          },
          "createSpreadsheetFlow": {
            "id": "createSpreadsheetFlow",
            "description": "createSpreadsheetFlow",
            "specVersion": "0.8",
            "version": "0.1",
            "start": "startState",
            "functions": [
              {
                "name": "generalRestFunc",
                "type": "custom",
                "operation": "connectivity::rest"
              }
            ],
            "states": [
              {
                "name": "startState",
                "type": "operation",
                "actions": [
                  {
                    "functionRef": {
```

```
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets",
                          "method": "POST",
                          "body": "${ {properties:
{title:.input.spreadsheetTitle},sheets:.input.sheets|map({properties:
{title:.sheetTitle}})} }"
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "testConnectionFlow": {
              "id": "testConnectionFlow",
              "description": "testConnectionFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://www.googleapis.com/drive/v3/files",
                          "method": "GET"
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ { success: (.status==200),
message: .body.error.message } }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
```

```
            },
            "spreadsheetIdFlow": {
              "id": "spreadsheetIdFlow",
              "description": "spreadsheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSpreadsheetIDs",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://www.googleapis.com/drive/v3/files",
                          "method": "GET",
                          "parameters": {
                            "q": "mimeType='application/vnd.google-
apps.spreadsheet' and trashed=false",
                            "pageSize": 1000
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlow": {
              "id": "sheetIdFlow",
              "description": "sheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
```

```
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSheetIDsBySpreadsheetId",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.sheets |
map({keyName:.properties.title, displayName:.properties.title}) }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "sheetSchemaFlow": {
                "id": "sheetSchemaFlow",
                "description": "sheetSchemaFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  },
                  {
                    "name": "constructReturnObject",
                    "type": "expression",
                    "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
```

```
                    "name": "generateSchemaBySheetHeader",
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                        "method": "GET",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                          "majorDimension": "ROWS"
                        }
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
                      "toStateData": "${ .schema }"
                    }
                  },
                  {
                    "name": "addSchemaType",
                    "functionRef": "constructReturnObject",
                    "actionDataFilter": {
                      "toStateData": "${ .output }"
                    }
                  }
                ],
                "end": true
              }
            ]
          },
          "sheetIdFlowErrorEngine": {
            "id": "sheetIdFlowErrorEngine",
            "description": "sheetIdFlowErrorEngine",
            "specVersion": "0.8",
            "version": "0.1",
            "start": "startState",
            "functions": [
              {
                "name": "generalRestFunc",
                "type": "custom",
                "operation": "connectivity::rest"
              }
            ],
            "states": [
              {
                "name": "startState",
                "type": "operation",
                "actions": [
                  {
                    "name": "getAllSheetIDsBySpreadsheetId",
                    "functionRef": {
                      "refName": "generalRestFunc",
```

```
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .sheets | map({keyName:.title,
displayName:.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            }
          }
        }
      }
    },
    {
      "time": "2024-08-12T18:56:30.605913431",
      "flow": "spreadsheetIdFlow",
      "state": "startState",
      "action": "getAllSpreadsheetIDs",
      "summary": "Action Execution Start",
      "input": {},
      "context": {
        "connectionProperties": {
          "applicationVersion": "1.0",
          "serviceInstanceOcid": "ocid5.exampleinstance.oc1.region.jjack",
          "pluginID": "cloud:google-sheets",
          "managedSecurity": "true",
          "integration_role": "TARGET",
          "oauth.scope": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
          "useProxy": "true",
          "securityPolicy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "proxyHost": "10.xx.xy.yy",
          "oauth.client.id": "813090821001-
uqpc1lsvauooc3lv8j3bh52cfs6kmfl4.apps.googleusercontent.com",
          "httpsProxyPort": "80",
          "oauth.client.secret": "gsYK8nYxxxxKx5p111188daJ",
          "oauth.auth.code.uri": "https://accounts.google.com/o/oauth2/auth",
          "oauth.access.token.uri": "https://www.googleapis.com/oauth2/v4/
token",
          "proxyPort": "80",
          "RABEngineVersion": "24.10",
          "csfMap": "example.wsm.security",
          "useCache": "true",
          "onPremise": "false",
```

```
          "clientAuthentication": "client_credentials_as_body",
          "httpsProxyHost": "10.xx.xy.yy",
          "csfkey":
"fac648317448763xxxxx2efd61a50c80243xxxxx2634ad87d396yyy7e5189038"
        },
        "input": {},
        "cloudAdapterUIBindingId": "04881642-6e72-8744-51f6-bea7270bf9bf",
        "configuration": {},
        "self": {
          "info": {
            "id": "cloud:google-sheets",
            "displayName": "Google Sheets",
            "description": "Google Sheets is a cloud-based sheet management
service.",
            "version": "1.0.0",
            "specVersion": "1.0",
            "categories": [],
            "appInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "publisherInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "settings": {
              "supportsConnectivityAgent": true,
              "_debug": true
            }
          },
          "connection": {
            "connectionProperties": [],
            "securityPolicies": [
              {
                "type": "managed",
                "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
                "description": "Google Authorization Policy",
                "displayName": "Google Authorization Policy",
                "scope": "ACTION",
                "securityProperties": [
                  {
                    "name": "oauth.client.id",
                    "displayName": "Google Client ID",
                    "description": "Google Client ID",
                    "shortDescription": "Google Client ID",
                    "hidden": false,
                    "required": true
                  },
```

```
                    {
                      "name": "oauth.client.secret",
                      "displayName": "Google Client Secret",
                      "description": "Google Client secret",
                      "shortDescription": "Google Client secret",
                      "hidden": false,
                      "required": true
                    },
                    {
                      "name": "oauth.auth.code.uri",
                      "default": "https://accounts.google.com/o/oauth2/auth",
                      "hidden": true,
                      "required": true
                    },
                    {
                      "name": "oauth.access.token.uri",
                      "default": "https://www.googleapis.com/oauth2/v4/token",
                      "hidden": true,
                      "required": true
                    },
                    {
                      "name": "oauth.scope",
                      "default": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
                      "hidden": true,
                      "required": true
                    },
                    {
                      "name": "clientAuthentication",
                      "default": "client_credentials_as_body",
                      "hidden": true,
                      "required": true
                    }
                  ],
                  "authExtension": {
                    "authorizationURL": {
                      "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
                      "params": {
                        "query": {
                          "response_type": "code",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "scope": "${.securityProperties.\"oauth.scope\"}",
                          "access_type": "offline",
                          "approval_prompt": "force"
                        }
                      }
                    },
                    "accessTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "query": {
                          "code": "${auth_code}",
```

```
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "redirect_uri": "${redirect_uri}",
                        "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                        "grant_type": "authorization_code"
                      }
                    },
                    "headers": {
                      "Content-Type": "application/x-www-form-urlencoded",
                      "Authorization": "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
                    },
                    "body": "false"
                  },
                  "refreshTokenRequest": {
                    "method": "POST",
                    "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                    "params": {
                      "template": {},
                      "query": {
                        "refresh_token": "${refresh_token}",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                        "grant_type": "refresh_token"
                      }
                    },
                    "headers": {
                      "Content-Type": "application/x-www-form-urlencoded"
                    },
                    "body": "false"
                  },
                  "fetchRules": {
                    "auth_code": "code",
                    "access_token": "access.[tT]oken",
                    "refresh_token": "refresh.[tT]oken",
                    "expiry": "expires.*",
                    "token_type": "token.?[tT]ype"
                  },
                  "accessTokenUsage": {
                    "headers": {
                      "Authorization": "Bearer ${access_token}"
                    }
                  }
                }
              }
            ],
            "test": "flow:testConnectionFlow"
          },
          "schemas": {
            "Sheet": {
              "type": "object",
```

```
    "properties": {
      "properties": {
        "type": "object",
        "properties": {
          "sheetId": {
            "type": "integer"
          },
          "title": {
            "type": "string"
          },
          "index": {
            "type": "integer"
          },
          "sheetType": {
            "type": "string"
          }
        }
      }
    }
  },
  "Spreadsheet": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "spreadsheetUrl": {
        "type": "string"
      },
      "properties": {
        "type": "object",
        "properties": {
          "title": {
            "type": "integer"
          },
          "locale": {
            "type": "string"
          },
          "autoRecalc": {
            "type": "integer"
          },
          "timeZone": {
            "type": "string"
          }
        }
      },
      "sheets": {
        "type": "array",
        "items": {
          "$ref": "#/schemas/Sheet"
        }
      }
    }
  },
  "getSpreadsheetInput": {
    "type": "object",
```

```
    "properties": {
      "spreadsheetId": {
        "type": "string"
      }
    }
  },
  "createSpreadsheetInput": {
    "type": "object",
    "properties": {
      "spreadsheetTitle": {
        "type": "string"
      },
      "sheets": {
        "type": "array",
        "items": {
          "type": "object",
          "properties": {
            "sheetTitle": {
              "type": "string"
            }
          }
        }
      }
    }
  },
  "insertRowOutput": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "tableRange": {
        "type": "string"
      },
      "updates": {
        "type": "object",
        "properties": {
          "spreadsheetId": {
            "type": "string"
          },
          "updatedRange": {
            "type": "string"
          },
          "updatedRows": {
            "type": "integer"
          },
          "updatedColumns": {
            "type": "integer"
          },
          "updatedCells": {
            "type": "integer"
          }
        }
      }
    }
  },
```

```
    "getRowInput": {
      "type": "object",
      "properties": {
        "rowNumber": {
          "type": "integer"
        }
      }
    }
  },
  "actions": {
    "insertRowAction": {
      "displayName": "Insert Row Into Sheet",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
      },
      "configuration": [
        {
          "name": "spreadsheetId",
          "displayName": "Spreadsheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:spreadsheetIdFlow",
          "required": true
        },
        {
          "name": "sheetId",
          "displayName": "Sheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:sheetIdFlow",
          "required": true,
          "dependencies": {
            "spreadsheetId": {
              "values": []
            }
          }
        }
      ]
    },
    "getRowAction": {
      "displayName": "Get Row From Sheet",
      "description": "This action get a row from sheet.",
      "execute": "flow:getRowFlow",
      "input": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/getRowInput"
        }
      },
      "output": "flow:sheetSchemaFlow",
```

```
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      },
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "currentRow",
            "displayName": "Get Specific Row"
          },
          {
            "keyName": "lastRow",
            "displayName": "Get Last Row"
          }
        ]
      }
    ]
  },
  "createSpreadsheetAction": {
    "displayName": "Create Spreadsheet",
    "description": "This action creates a new spreadsheet.",
    "execute": "flow:createSpreadsheetFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/createSpreadsheetInput"
      }
    },
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/Spreadsheet"
      }
```

```
      }
    },
    "insertRowActionErrorEngine": {
      "displayName": "Insert Row Into Sheet(Engine Error for Demo)",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
      },
      "configuration": [
        {
          "name": "spreadsheetId",
          "displayName": "Spreadsheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:spreadsheetIdFlow",
          "required": true
        },
        {
          "name": "sheetId",
          "displayName": "Sheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:sheetIdFlowErrorEngine",
          "required": true,
          "dependencies": {
            "spreadsheetId": {
              "values": []
            }
          }
        }
      ]
    }
  },
  "flows": {
    "insertRowFlow": {
      "id": "insertRowFlow",
      "description": "insertRowFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
```

```
                "actions": [
                  {
                    "name": "insertRowAction",
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}:append",
                        "method": "POST",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                          "insertDataOption": "INSERT_ROWS",
                          "valueInputOption": "RAW"
                        },
                        "body": "${ {range: (.configuration.sheetId + \"!
A1:Z1\"), majorDimension:\"ROWS\", values: .input | to_entries |
[map(.value)] } }"
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body }",
                      "toStateData": "${ .output }"
                    }
                  }
                ],
                "end": true
              }
            ]
          },
          "getRowFlow": {
            "id": "getRowFlow",
            "description": "getRowFlow",
            "specVersion": "0.8",
            "version": "0.1",
            "start": "startState",
            "functions": [
              {
                "name": "generalRestFunc",
                "type": "custom",
                "operation": "connectivity::rest"
              },
              {
                "name": "convertResult",
                "type": "expression",
                "operation": "(if .configuration.rowOperation ==
\"lastRow\" then (.lastLineNumber-1) else (.input.rowNumber-1) end)
as $rowNum | .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
              }
            ],
            "states": [
              {
                "name": "startState",
                "type": "operation",
```

```
                "actions": [
                  {
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                        "method": "GET",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ .configuration.sheetId + \"!
A1:Z40000\" }",
                          "majorDimension": "ROWS"
                        }
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body.values | length }",
                      "toStateData": "${ .lastLineNumber }"
                    }
                  },
                  {
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                        "method": "GET",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ if .configuration.rowOperation ==
\"lastRow\" then .configuration.sheetId + \"!A1:Z\" + (.lastLineNumber|
tostring) else .configuration.sheetId + \"!A1:Z\" + (.input.rowNumber|
tostring) end }",
                          "majorDimension": "COLUMNS"
                        }
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body.values }",
                      "toStateData": "${ .values }"
                    }
                  },
                  {
                    "functionRef": "convertResult",
                    "actionDataFilter": {
                      "toStateData": "${ .output }"
                    }
                  }
                ],
                "end": true
              }
            ]
          },
```

```
                "createSpreadsheetFlow": {
                  "id": "createSpreadsheetFlow",
                  "description": "createSpreadsheetFlow",
                  "specVersion": "0.8",
                  "version": "0.1",
                  "start": "startState",
                  "functions": [
                    {
                      "name": "generalRestFunc",
                      "type": "custom",
                      "operation": "connectivity::rest"
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://sheets.googleapis.com/v4/
spreadsheets",
                              "method": "POST",
                              "body": "${ {properties:
{title:.input.spreadsheetTitle},sheets:.input.sheets|map({properties:
{title:.sheetTitle}})} }"
                            }
                          },
                          "actionDataFilter": {
                            "results": "${ .body }",
                            "toStateData": "${ .output }"
                          }
                        }
                      ],
                      "end": true
                    }
                  ]
                },
                "testConnectionFlow": {
                  "id": "testConnectionFlow",
                  "description": "testConnectionFlow",
                  "specVersion": "0.8",
                  "version": "0.1",
                  "start": "startState",
                  "functions": [
                    {
                      "name": "generalRestFunc",
                      "type": "custom",
                      "operation": "connectivity::rest"
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
```

```
                    "type": "operation",
                    "actions": [
                      {
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://www.googleapis.com/drive/v3/files",
                            "method": "GET"
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ { success: (.status==200),
message: .body.error.message } }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "spreadsheetIdFlow": {
                "id": "spreadsheetIdFlow",
                "description": "spreadsheetIdFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSpreadsheetIDs",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://www.googleapis.com/drive/v3/files",
                            "method": "GET",
                            "parameters": {
                              "q": "mimeType='application/vnd.google-
apps.spreadsheet' and trashed=false",
                              "pageSize": 1000
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
```

```
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlow": {
              "id": "sheetIdFlow",
              "description": "sheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSheetIDsBySpreadsheetId",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.sheets |
map({keyName:.properties.title, displayName:.properties.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetSchemaFlow": {
              "id": "sheetSchemaFlow",
              "description": "sheetSchemaFlow",
              "specVersion": "0.8",
              "version": "0.1",
```

```
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  },
                  {
                    "name": "constructReturnObject",
                    "type": "expression",
                    "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "generateSchemaBySheetHeader",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                              "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                              "majorDimension": "ROWS"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
                          "toStateData": "${ .schema }"
                        }
                      },
                      {
                        "name": "addSchemaType",
                        "functionRef": "constructReturnObject",
                        "actionDataFilter": {
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "sheetIdFlowErrorEngine": {
                "id": "sheetIdFlowErrorEngine",
```

```
                "description": "sheetIdFlowErrorEngine",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSheetIDsBySpreadsheetId",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .sheets | map({keyName:.title,
displayName:.title}) }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              }
            }
          }
        }
      },
      {
        "time": "2024-08-12T18:56:30.628363538",
        "flow": "spreadsheetIdFlow",
        "state": "startState",
        "action": "getAllSpreadsheetIDs",
        "summary": "Connectivity Rest Start",
        "input": {
          "uri": "https://www.googleapis.com/drive/v3/files",
          "method": "GET",
          "parameters": {
            "q": "mimeType='application/vnd.google-apps.spreadsheet' and
```

```
trashed=false",
          "pageSize": 1000
        }
      }
    },
    {
      "time": "2024-08-12T18:56:31.124885162",
      "summary": "Connectivity Rest End",
      "output": {
        "body": {
          "kind": "drive#fileList",
          "incompleteSearch": false,
          "files": [
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
              "name": "Customer Managment"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
              "name": "HL7 Message - FHIR R4: VXU_V04"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
              "name": "Copy of Restricted Stock Unit (RSU) Calculator"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
              "name": "Development Management"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
              "name": "Event Attendee List"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
              "name": "HR Management"
            },
            {
              "kind": "drive#file",
              "mimeType": "application/vnd.google-apps.spreadsheet",
              "id": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
              "name": "Financial Managment"
            },
            {
```

```
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
            "name": "UAT-File"
          },
          {
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
            "name": "SalesAccounts"
          }
        ]
      },
      "status": 200,
      "headers": {
        "cache-control": "no-cache, no-store, max-age=0, must-revalidate",
        "vary": "Origin, X-Origin",
        "pragma": "no-cache",
        "expires": "Mon, 01 Jan 1990 00:00:00 GMT",
        "date": "Mon, 12 Aug 2024 18:56:31 GMT",
        "content-type": "application/json; charset=UTF-8",
        "server": "ESF",
        "x-xss-protection": "0",
        "x-frame-options": "SAMEORIGIN",
        "x-content-type-options": "nosniff",
        "alt-svc": "h3=\":443\"; ma=2592000,h3-29=\":443\"; ma=2592000",
        "transfer-encoding": "chunked",
        "x-span-id": "d3074029994608a9"
      }
    }
  },
  {
    "time": "2024-08-12T18:56:31.125949711",
    "summary": "Json Query Start",
    "input": " .body.files | map({keyName:.id, displayName:.name}) ",
    "context": {
      "body": {
        "kind": "drive#fileList",
        "incompleteSearch": false,
        "files": [
          {
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
            "name": "Customer Managment"
          },
          {
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
            "name": "HL7 Message - FHIR R4: VXU_V04"
          },
          {
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
```

```
        "name": "Copy of Restricted Stock Unit (RSU) Calculator"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1DjtOl1dvZPMT000LJZZts9foehDdRoicmfTP022Fkw4",
        "name": "Development Management"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
        "name": "Event Attendee List"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
        "name": "HR Management"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
        "name": "Financial Managment"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
        "name": "UAT-File"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
        "name": "SalesAccounts"
      }
    ]
  },
  "status": 200,
  "headers": {
    "cache-control": "no-cache, no-store, max-age=0, must-revalidate",
    "vary": "Origin, X-Origin",
    "pragma": "no-cache",
    "expires": "Mon, 01 Jan 1990 00:00:00 GMT",
    "date": "Mon, 12 Aug 2024 18:56:31 GMT",
    "content-type": "application/json; charset=UTF-8",
    "server": "ESF",
    "x-xss-protection": "0",
    "x-frame-options": "SAMEORIGIN",
    "x-content-type-options": "nosniff",
    "alt-svc": "h3=\":443\"; ma=2592000,h3-29=\":443\"; ma=2592000",
    "transfer-encoding": "chunked",
    "x-span-id": "d3074029994608a9"
  }
```

```
    }
  },
  {
    "time": "2024-08-12T18:56:31.127086899",
    "summary": "Json Query End",
    "input": " .body.files | map({keyName:.id, displayName:.name}) ",
    "output": [
      {
        "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
        "displayName": "Customer Managment"
      },
      {
        "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
        "displayName": "HL7 Message - FHIR R4: VXU_V04"
      },
      {
        "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
        "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
      },
      {
        "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
        "displayName": "Development Management"
      },
      {
        "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
        "displayName": "Event Attendee List"
      },
      {
        "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
        "displayName": "HR Management"
      },
      {
        "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
        "displayName": "Financial Managment"
      },
      {
        "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
        "displayName": "UAT-File"
      },
      {
        "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
        "displayName": "SalesAccounts"
      }
    ],
    "context": {
      "body": {
        "kind": "drive#fileList",
        "incompleteSearch": false,
        "files": [
          {
            "kind": "drive#file",
            "mimeType": "application/vnd.google-apps.spreadsheet",
            "id": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
            "name": "Customer Managment"
          },
          {
```

```
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
        "name": "HL7 Message - FHIR R4: VXU_V04"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
        "name": "Copy of Restricted Stock Unit (RSU) Calculator"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
        "name": "Development Management"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
        "name": "Event Attendee List"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
        "name": "HR Management"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
        "name": "Financial Managment"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
        "name": "UAT-File"
      },
      {
        "kind": "drive#file",
        "mimeType": "application/vnd.google-apps.spreadsheet",
        "id": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
        "name": "SalesAccounts"
      }
    ]
  },
  "status": 200,
  "headers": {
    "cache-control": "no-cache, no-store, max-age=0, must-revalidate",
    "vary": "Origin, X-Origin",
    "pragma": "no-cache",
    "expires": "Mon, 01 Jan 1990 00:00:00 GMT",
    "date": "Mon, 12 Aug 2024 18:56:31 GMT",
```

```
      "content-type": "application/json; charset=UTF-8",
      "server": "ESF",
      "x-xss-protection": "0",
      "x-frame-options": "SAMEORIGIN",
      "x-content-type-options": "nosniff",
      "alt-svc": "h3=\":443\"; ma=2592000,h3-29=\":443\"; ma=2592000",
      "transfer-encoding": "chunked",
      "x-span-id": "d3074029994608a9"
    }
  }
},
{
  "time": "2024-08-12T18:56:31.128194891",
  "flow": "spreadsheetIdFlow",
  "state": "startState",
  "action": "getAllSpreadsheetIDs",
  "summary": "Action Execution End",
  "input": {},
  "output": [
    {
      "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
      "displayName": "Customer Managment"
    },
    {
      "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
      "displayName": "HL7 Message - FHIR R4: VXU_V04"
    },
    {
      "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
      "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
    },
    {
      "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
      "displayName": "Development Management"
    },
    {
      "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
      "displayName": "Event Attendee List"
    },
    {
      "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
      "displayName": "HR Management"
    },
    {
      "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
      "displayName": "Financial Managment"
    },
    {
      "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
      "displayName": "UAT-File"
    },
    {
      "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
      "displayName": "SalesAccounts"
    }
  ],
```

```
      "context": {
        "connectionProperties": {
          "applicationVersion": "1.0",
          "serviceInstanceOcid": "ocid5.exampleinstance.oc1.region.jjack",
          "pluginID": "cloud:google-sheets",
          "managedSecurity": "true",
          "integration_role": "TARGET",
          "oauth.scope": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
          "useProxy": "true",
          "securityPolicy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "proxyHost": "10.xx.xy.yy",
          "oauth.client.id": "813090821001-
uqpc1lsvauooc3lv8j3bh52cfs6kmfl4.apps.googleusercontent.com",
          "httpsProxyPort": "80",
          "oauth.client.secret": "gsYK8nYxxxxKx5p111188daJ",
          "oauth.auth.code.uri": "https://accounts.google.com/o/oauth2/auth",
          "oauth.access.token.uri": "https://www.googleapis.com/oauth2/v4/
token",
          "proxyPort": "80",
          "RABEngineVersion": "24.10",
          "csfMap": "example.wsm.security",
          "useCache": "true",
          "onPremise": "false",
          "clientAuthentication": "client_credentials_as_body",
          "httpsProxyHost": "10.xx.xy.yy",
          "csfkey":
"fac648317448763xxxxx2efd61a50c80243xxxxx2634ad87d396yyy7e5189038"
        },
        "input": {},
        "cloudAdapterUIBindingId": "04881642-6e72-8744-51f6-bea7270bf9bf",
        "configuration": {},
        "self": {
          "info": {
            "id": "cloud:google-sheets",
            "displayName": "Google Sheets",
            "description": "Google Sheets is a cloud-based sheet management
service.",
            "version": "1.0.0",
            "specVersion": "1.0",
            "categories": [],
            "appInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "publisherInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
```

```
      },
      "settings": {
        "supportsConnectivityAgent": true,
        "_debug": true
      }
    },
    "connection": {
      "connectionProperties": [],
      "securityPolicies": [
        {
          "type": "managed",
          "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "description": "Google Authorization Policy",
          "displayName": "Google Authorization Policy",
          "scope": "ACTION",
          "securityProperties": [
            {
              "name": "oauth.client.id",
              "displayName": "Google Client ID",
              "description": "Google Client ID",
              "shortDescription": "Google Client ID",
              "hidden": false,
              "required": true
            },
            {
              "name": "oauth.client.secret",
              "displayName": "Google Client Secret",
              "description": "Google Client secret",
              "shortDescription": "Google Client secret",
              "hidden": false,
              "required": true
            },
            {
              "name": "oauth.auth.code.uri",
              "default": "https://accounts.google.com/o/oauth2/auth",
              "hidden": true,
              "required": true
            },
            {
              "name": "oauth.access.token.uri",
              "default": "https://www.googleapis.com/oauth2/v4/token",
              "hidden": true,
              "required": true
            },
            {
              "name": "oauth.scope",
              "default": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
              "hidden": true,
              "required": true
            },
            {
              "name": "clientAuthentication",
              "default": "client_credentials_as_body",
              "hidden": true,
              "required": true
```

```
                    }
                  ],
                  "authExtension": {
                    "authorizationURL": {
                      "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
                      "params": {
                        "query": {
                          "response_type": "code",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "scope": "${.securityProperties.\"oauth.scope\"}",
                          "access_type": "offline",
                          "approval_prompt": "force"
                        }
                      }
                    },
                    "accessTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "query": {
                          "code": "${auth_code}",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                          "grant_type": "authorization_code"
                        }
                      },
                      "headers": {
                        "Content-Type": "application/x-www-form-urlencoded",
                        "Authorization": "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
                      },
                      "body": "false"
                    },
                    "refreshTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "template": {},
                        "query": {
                          "refresh_token": "${refresh_token}",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                          "grant_type": "refresh_token"
                        }
                      },
                      "headers": {
```

```
            "Content-Type": "application/x-www-form-urlencoded"
          },
          "body": "false"
        },
        "fetchRules": {
          "auth_code": "code",
          "access_token": "access.[tT]oken",
          "refresh_token": "refresh.[tT]oken",
          "expiry": "expires.*",
          "token_type": "token.?[tT]ype"
        },
        "accessTokenUsage": {
          "headers": {
            "Authorization": "Bearer ${access_token}"
          }
        }
      }
    }
  ],
  "test": "flow:testConnectionFlow"
},
"schemas": {
  "Sheet": {
    "type": "object",
    "properties": {
      "properties": {
        "type": "object",
        "properties": {
          "sheetId": {
            "type": "integer"
          },
          "title": {
            "type": "string"
          },
          "index": {
            "type": "integer"
          },
          "sheetType": {
            "type": "string"
          }
        }
      }
    }
  },
  "Spreadsheet": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "spreadsheetUrl": {
        "type": "string"
      },
      "properties": {
        "type": "object",
        "properties": {
```

```
          "title": {
            "type": "integer"
          },
          "locale": {
            "type": "string"
          },
          "autoRecalc": {
            "type": "integer"
          },
          "timeZone": {
            "type": "string"
          }
        }
      },
      "sheets": {
        "type": "array",
        "items": {
          "$ref": "#/schemas/Sheet"
        }
      }
    }
  },
  "getSpreadsheetInput": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      }
    }
  },
  "createSpreadsheetInput": {
    "type": "object",
    "properties": {
      "spreadsheetTitle": {
        "type": "string"
      },
      "sheets": {
        "type": "array",
        "items": {
          "type": "object",
          "properties": {
            "sheetTitle": {
              "type": "string"
            }
          }
        }
      }
    }
  },
  "insertRowOutput": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "tableRange": {
```

```
            "type": "string"
          },
          "updates": {
            "type": "object",
            "properties": {
              "spreadsheetId": {
                "type": "string"
              },
              "updatedRange": {
                "type": "string"
              },
              "updatedRows": {
                "type": "integer"
              },
              "updatedColumns": {
                "type": "integer"
              },
              "updatedCells": {
                "type": "integer"
              }
            }
          }
        }
      },
      "getRowInput": {
        "type": "object",
        "properties": {
          "rowNumber": {
            "type": "integer"
          }
        }
      }
    },
    "actions": {
      "insertRowAction": {
        "displayName": "Insert Row Into Sheet",
        "description": "This action insert a new row into sheet.",
        "execute": "flow:insertRowFlow",
        "input": "flow:sheetSchemaFlow",
        "output": {
          "schemaType": "application/schema+json",
          "schema": {
            "$ref": "#/schemas/insertRowOutput"
          }
        },
        "configuration": [
          {
            "name": "spreadsheetId",
            "displayName": "Spreadsheet Name",
            "description": "",
            "type": "COMBO_BOX",
            "options": "flow:spreadsheetIdFlow",
            "required": true
          },
          {
            "name": "sheetId",
```

```
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      }
    ]
  },
  "getRowAction": {
    "displayName": "Get Row From Sheet",
    "description": "This action get a row from sheet.",
    "execute": "flow:getRowFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/getRowInput"
      }
    },
    "output": "flow:sheetSchemaFlow",
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      },
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "currentRow",
            "displayName": "Get Specific Row"
```

```
        },
        {
          "keyName": "lastRow",
          "displayName": "Get Last Row"
        }
      ]
    }
  ]
},
"createSpreadsheetAction": {
  "displayName": "Create Spreadsheet",
  "description": "This action creates a new spreadsheet.",
  "execute": "flow:createSpreadsheetFlow",
  "input": {
    "schemaType": "application/schema+json",
    "schema": {
      "$ref": "#/schemas/createSpreadsheetInput"
    }
  },
  "output": {
    "schemaType": "application/schema+json",
    "schema": {
      "$ref": "#/schemas/Spreadsheet"
    }
  }
},
"insertRowActionErrorEngine": {
  "displayName": "Insert Row Into Sheet(Engine Error for Demo)",
  "description": "This action insert a new row into sheet.",
  "execute": "flow:insertRowFlow",
  "input": "flow:sheetSchemaFlow",
  "output": {
    "schemaType": "application/schema+json",
    "schema": {
      "$ref": "#/schemas/insertRowOutput"
    }
  },
  "configuration": [
    {
      "name": "spreadsheetId",
      "displayName": "Spreadsheet Name",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:spreadsheetIdFlow",
      "required": true
    },
    {
      "name": "sheetId",
      "displayName": "Sheet Name",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:sheetIdFlowErrorEngine",
      "required": true,
      "dependencies": {
        "spreadsheetId": {
          "values": []
```

```
                    }
                  }
                }
              ]
            }
          },
          "flows": {
            "insertRowFlow": {
              "id": "insertRowFlow",
              "description": "insertRowFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "insertRowAction",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}:append",
                          "method": "POST",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                            "insertDataOption": "INSERT_ROWS",
                            "valueInputOption": "RAW"
                          },
                          "body": "${ {range: (.configuration.sheetId + \"!
A1:Z1\"), majorDimension:\"ROWS\", values: .input | to_entries |
[map(.value)] } }"
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
```

```
            "getRowFlow": {
              "id": "getRowFlow",
              "description": "getRowFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                },
                {
                  "name": "convertResult",
                  "type": "expression",
                  "operation": "(if .configuration.rowOperation ==
\"lastRow\" then (.lastLineNumber-1) else (.input.rowNumber-1) end)
as $rowNum | .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z40000\" }",
                            "majorDimension": "ROWS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values | length }",
                        "toStateData": "${ .lastLineNumber }"
                      }
                    },
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
```

```
                              "range": "${ if .configuration.rowOperation ==
\"lastRow\" then .configuration.sheetId + \"!A1:Z\" + (.lastLineNumber|
tostring) else .configuration.sheetId + \"!A1:Z\" + (.input.rowNumber|
tostring) end }",
                              "majorDimension": "COLUMNS"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.values }",
                          "toStateData": "${ .values }"
                        }
                      },
                      {
                        "functionRef": "convertResult",
                        "actionDataFilter": {
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "createSpreadsheetFlow": {
                "id": "createSpreadsheetFlow",
                "description": "createSpreadsheetFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets",
                            "method": "POST",
                            "body": "${ {properties:
{title:.input.spreadsheetTitle},sheets:.input.sheets|map({properties:
{title:.sheetTitle}})} }"
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body }",
```

```
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "testConnectionFlow": {
              "id": "testConnectionFlow",
              "description": "testConnectionFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://www.googleapis.com/drive/v3/files",
                          "method": "GET"
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ { success: (.status==200),
message: .body.error.message } }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "spreadsheetIdFlow": {
              "id": "spreadsheetIdFlow",
              "description": "spreadsheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
```

```
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "name": "getAllSpreadsheetIDs",
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://www.googleapis.com/drive/v3/files",
                              "method": "GET",
                              "parameters": {
                                "q": "mimeType='application/vnd.google-
apps.spreadsheet' and trashed=false",
                                "pageSize": 1000
                              }
                            }
                          },
                          "actionDataFilter": {
                            "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
                            "toStateData": "${ .output }"
                          }
                        }
                      ],
                      "end": true
                    }
                  ]
                },
                "sheetIdFlow": {
                  "id": "sheetIdFlow",
                  "description": "sheetIdFlow",
                  "specVersion": "0.8",
                  "version": "0.1",
                  "start": "startState",
                  "functions": [
                    {
                      "name": "generalRestFunc",
                      "type": "custom",
                      "operation": "connectivity::rest"
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "name": "getAllSheetIDsBySpreadsheetId",
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://sheets.googleapis.com/v4/
```

```
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.sheets |
map({keyName:.properties.title, displayName:.properties.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetSchemaFlow": {
              "id": "sheetSchemaFlow",
              "description": "sheetSchemaFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                },
                {
                  "name": "constructReturnObject",
                  "type": "expression",
                  "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "generateSchemaBySheetHeader",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
```

```
                            "majorDimension": "ROWS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
                        "toStateData": "${ .schema }"
                      }
                    },
                    {
                      "name": "addSchemaType",
                      "functionRef": "constructReturnObject",
                      "actionDataFilter": {
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlowErrorEngine": {
              "id": "sheetIdFlowErrorEngine",
              "description": "sheetIdFlowErrorEngine",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSheetIDsBySpreadsheetId",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .sheets | map({keyName:.title,
```

```
displayName:.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            }
          }
        },
        "output": [
          {
            "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
            "displayName": "Customer Managment"
          },
          {
            "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
            "displayName": "HL7 Message - FHIR R4: VXU_V04"
          },
          {
            "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
            "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
          },
          {
            "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
            "displayName": "Development Management"
          },
          {
            "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
            "displayName": "Event Attendee List"
          },
          {
            "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
            "displayName": "HR Management"
          },
          {
            "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
            "displayName": "Financial Managment"
          },
          {
            "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
            "displayName": "UAT-File"
          },
          {
            "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
            "displayName": "SalesAccounts"
          }
        ]
      }
    },
    {
      "time": "2024-08-12T18:56:31.130205607",
      "flow": "spreadsheetIdFlow",
      "state": "startState",
      "summary": "State Execution",
```

```
      "input": {},
      "output": [
        {
          "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
          "displayName": "Customer Managment"
        },
        {
          "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
          "displayName": "HL7 Message - FHIR R4: VXU_V04"
        },
        {
          "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
          "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
        },
        {
          "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
          "displayName": "Development Management"
        },
        {
          "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
          "displayName": "Event Attendee List"
        },
        {
          "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
          "displayName": "HR Management"
        },
        {
          "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
          "displayName": "Financial Managment"
        },
        {
          "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
          "displayName": "UAT-File"
        },
        {
          "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
          "displayName": "SalesAccounts"
        }
      ],
      "context": {
        "connectionProperties": {
          "applicationVersion": "1.0",
          "serviceInstanceOcid": "ocid5.exampleinstance.oc1.region.jjack",
          "pluginID": "cloud:google-sheets",
          "managedSecurity": "true",
          "integration_role": "TARGET",
          "oauth.scope": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
          "useProxy": "true",
          "securityPolicy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "proxyHost": "10.xx.xy.yy",
          "oauth.client.id": "813090821001-
uqpc1lsvauooc3lv8j3bh52cfs6kmfl4.apps.googleusercontent.com",
          "httpsProxyPort": "80",
          "oauth.client.secret": "gsYK8nYxxxxKx5p111188daJ",
          "oauth.auth.code.uri": "https://accounts.google.com/o/oauth2/auth",
```

```
          "oauth.access.token.uri": "https://www.googleapis.com/oauth2/v4/
token",
          "proxyPort": "80",
          "RABEngineVersion": "24.10",
          "csfMap": "example.wsm.security",
          "useCache": "true",
          "onPremise": "false",
          "clientAuthentication": "client_credentials_as_body",
          "httpsProxyHost": "10.xx.xy.yy",
          "csfkey":
"fac648317448763xxxxx2efd61a50c80243xxxxx2634ad87d396yyy7e5189038"
        },
        "input": {},
        "cloudAdapterUIBindingId": "04881642-6e72-8744-51f6-bea7270bf9bf",
        "configuration": {},
        "self": {
          "info": {
            "id": "cloud:google-sheets",
            "displayName": "Google Sheets",
            "description": "Google Sheets is a cloud-based sheet management
service.",
            "version": "1.0.0",
            "specVersion": "1.0",
            "categories": [],
            "appInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "publisherInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "settings": {
              "supportsConnectivityAgent": true,
              "_debug": true
            }
          },
          "connection": {
            "connectionProperties": [],
            "securityPolicies": [
              {
                "type": "managed",
                "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
                "description": "Google Authorization Policy",
                "displayName": "Google Authorization Policy",
                "scope": "ACTION",
                "securityProperties": [
                  {
```

```
                    "name": "oauth.client.id",
                    "displayName": "Google Client ID",
                    "description": "Google Client ID",
                    "shortDescription": "Google Client ID",
                    "hidden": false,
                    "required": true
                  },
                  {
                    "name": "oauth.client.secret",
                    "displayName": "Google Client Secret",
                    "description": "Google Client secret",
                    "shortDescription": "Google Client secret",
                    "hidden": false,
                    "required": true
                  },
                  {
                    "name": "oauth.auth.code.uri",
                    "default": "https://accounts.google.com/o/oauth2/auth",
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "oauth.access.token.uri",
                    "default": "https://www.googleapis.com/oauth2/v4/token",
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "oauth.scope",
                    "default": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "clientAuthentication",
                    "default": "client_credentials_as_body",
                    "hidden": true,
                    "required": true
                  }
                ],
                "authExtension": {
                  "authorizationURL": {
                    "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
                    "params": {
                      "query": {
                        "response_type": "code",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "redirect_uri": "${redirect_uri}",
                        "scope": "${.securityProperties.\"oauth.scope\"}",
                        "access_type": "offline",
                        "approval_prompt": "force"
                      }
                    }
                  },
```

```
                    "accessTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "query": {
                          "code": "${auth_code}",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "redirect_uri": "${redirect_uri}",
                          "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                          "grant_type": "authorization_code"
                        }
                      },
                      "headers": {
                        "Content-Type": "application/x-www-form-urlencoded",
                        "Authorization": "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
                      },
                      "body": "false"
                    },
                    "refreshTokenRequest": {
                      "method": "POST",
                      "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                      "params": {
                        "template": {},
                        "query": {
                          "refresh_token": "${refresh_token}",
                          "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                          "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                          "grant_type": "refresh_token"
                        }
                      },
                      "headers": {
                        "Content-Type": "application/x-www-form-urlencoded"
                      },
                      "body": "false"
                    },
                    "fetchRules": {
                      "auth_code": "code",
                      "access_token": "access.[tT]oken",
                      "refresh_token": "refresh.[tT]oken",
                      "expiry": "expires.*",
                      "token_type": "token.?[tT]ype"
                    },
                    "accessTokenUsage": {
                      "headers": {
                        "Authorization": "Bearer ${access_token}"
                      }
                    }
                  }
```

```
      }
    ],
    "test": "flow:testConnectionFlow"
  },
  "schemas": {
    "Sheet": {
      "type": "object",
      "properties": {
        "properties": {
          "type": "object",
          "properties": {
            "sheetId": {
              "type": "integer"
            },
            "title": {
              "type": "string"
            },
            "index": {
              "type": "integer"
            },
            "sheetType": {
              "type": "string"
            }
          }
        }
      }
    },
    "Spreadsheet": {
      "type": "object",
      "properties": {
        "spreadsheetId": {
          "type": "string"
        },
        "spreadsheetUrl": {
          "type": "string"
        },
        "properties": {
          "type": "object",
          "properties": {
            "title": {
              "type": "integer"
            },
            "locale": {
              "type": "string"
            },
            "autoRecalc": {
              "type": "integer"
            },
            "timeZone": {
              "type": "string"
            }
          }
        },
        "sheets": {
          "type": "array",
          "items": {
```

```
            "$ref": "#/schemas/Sheet"
          }
        }
      }
    },
    "getSpreadsheetInput": {
      "type": "object",
      "properties": {
        "spreadsheetId": {
          "type": "string"
        }
      }
    },
    "createSpreadsheetInput": {
      "type": "object",
      "properties": {
        "spreadsheetTitle": {
          "type": "string"
        },
        "sheets": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "sheetTitle": {
                "type": "string"
              }
            }
          }
        }
      }
    },
    "insertRowOutput": {
      "type": "object",
      "properties": {
        "spreadsheetId": {
          "type": "string"
        },
        "tableRange": {
          "type": "string"
        },
        "updates": {
          "type": "object",
          "properties": {
            "spreadsheetId": {
              "type": "string"
            },
            "updatedRange": {
              "type": "string"
            },
            "updatedRows": {
              "type": "integer"
            },
            "updatedColumns": {
              "type": "integer"
            },
```

```
            "updatedCells": {
              "type": "integer"
            }
          }
        }
      }
    },
    "getRowInput": {
      "type": "object",
      "properties": {
        "rowNumber": {
          "type": "integer"
        }
      }
    }
  },
  "actions": {
    "insertRowAction": {
      "displayName": "Insert Row Into Sheet",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
      },
      "configuration": [
        {
          "name": "spreadsheetId",
          "displayName": "Spreadsheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:spreadsheetIdFlow",
          "required": true
        },
        {
          "name": "sheetId",
          "displayName": "Sheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:sheetIdFlow",
          "required": true,
          "dependencies": {
            "spreadsheetId": {
              "values": []
            }
          }
        }
      ]
    },
    "getRowAction": {
      "displayName": "Get Row From Sheet",
      "description": "This action get a row from sheet.",
      "execute": "flow:getRowFlow",
```

```
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/getRowInput"
      }
    },
    "output": "flow:sheetSchemaFlow",
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      },
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "currentRow",
            "displayName": "Get Specific Row"
          },
          {
            "keyName": "lastRow",
            "displayName": "Get Last Row"
          }
        ]
      }
    ]
  },
  "createSpreadsheetAction": {
    "displayName": "Create Spreadsheet",
    "description": "This action creates a new spreadsheet.",
    "execute": "flow:createSpreadsheetFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/createSpreadsheetInput"
```

```
        }
      },
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/Spreadsheet"
        }
      }
    },
    "insertRowActionErrorEngine": {
      "displayName": "Insert Row Into Sheet(Engine Error for Demo)",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
      },
      "configuration": [
        {
          "name": "spreadsheetId",
          "displayName": "Spreadsheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:spreadsheetIdFlow",
          "required": true
        },
        {
          "name": "sheetId",
          "displayName": "Sheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:sheetIdFlowErrorEngine",
          "required": true,
          "dependencies": {
            "spreadsheetId": {
              "values": []
            }
          }
        }
      ]
    }
  },
  "flows": {
    "insertRowFlow": {
      "id": "insertRowFlow",
      "description": "insertRowFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
```

```
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "insertRowAction",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}:append",
                            "method": "POST",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                              "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                              "insertDataOption": "INSERT_ROWS",
                              "valueInputOption": "RAW"
                            },
                            "body": "${ {range: (.configuration.sheetId + \"!
A1:Z1\"), majorDimension:\"ROWS\", values: .input | to_entries |
[map(.value)] } }"
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "getRowFlow": {
                "id": "getRowFlow",
                "description": "getRowFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  },
                  {
                    "name": "convertResult",
                    "type": "expression",
                    "operation": "(if .configuration.rowOperation ==
\"lastRow\" then (.lastLineNumber-1) else (.input.rowNumber-1) end)
```

```
as $rowNum | .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z40000\" }",
                            "majorDimension": "ROWS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values | length }",
                        "toStateData": "${ .lastLineNumber }"
                      }
                    },
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ if .configuration.rowOperation ==
\"lastRow\" then .configuration.sheetId + \"!A1:Z\" + (.lastLineNumber|
tostring) else .configuration.sheetId + \"!A1:Z\" + (.input.rowNumber|
tostring) end }",
                            "majorDimension": "COLUMNS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values }",
                        "toStateData": "${ .values }"
                      }
                    },
                    {
                      "functionRef": "convertResult",
                      "actionDataFilter": {
                        "toStateData": "${ .output }"
```

```
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "createSpreadsheetFlow": {
                "id": "createSpreadsheetFlow",
                "description": "createSpreadsheetFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets",
                            "method": "POST",
                            "body": "${ {properties:
{title:.input.spreadsheetTitle},sheets:.input.sheets|map({properties:
{title:.sheetTitle}})} }"
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "testConnectionFlow": {
                "id": "testConnectionFlow",
                "description": "testConnectionFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
```

```
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://www.googleapis.com/drive/v3/files",
                            "method": "GET"
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ { success: (.status==200),
message: .body.error.message } }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "spreadsheetIdFlow": {
                "id": "spreadsheetIdFlow",
                "description": "spreadsheetIdFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSpreadsheetIDs",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://www.googleapis.com/drive/v3/files",
                            "method": "GET",
                            "parameters": {
                              "q": "mimeType='application/vnd.google-
apps.spreadsheet' and trashed=false",
```

```
                            "pageSize": 1000
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlow": {
              "id": "sheetIdFlow",
              "description": "sheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSheetIDsBySpreadsheetId",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.sheets |
map({keyName:.properties.title, displayName:.properties.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
```

```
            ]
          },
          "sheetSchemaFlow": {
            "id": "sheetSchemaFlow",
            "description": "sheetSchemaFlow",
            "specVersion": "0.8",
            "version": "0.1",
            "start": "startState",
            "functions": [
              {
                "name": "generalRestFunc",
                "type": "custom",
                "operation": "connectivity::rest"
              },
              {
                "name": "constructReturnObject",
                "type": "expression",
                "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
              }
            ],
            "states": [
              {
                "name": "startState",
                "type": "operation",
                "actions": [
                  {
                    "name": "generateSchemaBySheetHeader",
                    "functionRef": {
                      "refName": "generalRestFunc",
                      "arguments": {
                        "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                        "method": "GET",
                        "parameters": {
                          "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                          "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                          "majorDimension": "ROWS"
                        }
                      }
                    },
                    "actionDataFilter": {
                      "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
                      "toStateData": "${ .schema }"
                    }
                  },
                  {
                    "name": "addSchemaType",
                    "functionRef": "constructReturnObject",
                    "actionDataFilter": {
                      "toStateData": "${ .output }"
                    }
                  }
```

```
                    ],
                    "end": true
                  }
                ]
              },
              "sheetIdFlowErrorEngine": {
                "id": "sheetIdFlowErrorEngine",
                "description": "sheetIdFlowErrorEngine",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSheetIDsBySpreadsheetId",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .sheets | map({keyName:.title,
displayName:.title}) }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              }
            }
          },
          "output": [
            {
              "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
              "displayName": "Customer Managment"
            },
            {
```

```
            "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
            "displayName": "HL7 Message - FHIR R4: VXU_V04"
          },
          {
            "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
            "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
          },
          {
            "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
            "displayName": "Development Management"
          },
          {
            "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
            "displayName": "Event Attendee List"
          },
          {
            "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
            "displayName": "HR Management"
          },
          {
            "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
            "displayName": "Financial Managment"
          },
          {
            "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
            "displayName": "UAT-File"
          },
          {
            "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
            "displayName": "SalesAccounts"
          }
        ]
      }
    },
    {
      "time": "2024-08-12T18:56:31.132014793",
      "flow": "spreadsheetIdFlow",
      "summary": "Flow Execution Ended ...",
      "input": {},
      "output": [
        {
          "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
          "displayName": "Customer Managment"
        },
        {
          "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
          "displayName": "HL7 Message - FHIR R4: VXU_V04"
        },
        {
          "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
          "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
        },
        {
          "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
          "displayName": "Development Management"
        },
```

```
        {
          "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
          "displayName": "Event Attendee List"
        },
        {
          "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
          "displayName": "HR Management"
        },
        {
          "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
          "displayName": "Financial Managment"
        },
        {
          "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
          "displayName": "UAT-File"
        },
        {
          "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
          "displayName": "SalesAccounts"
        }
      ],
      "context": {
        "connectionProperties": {
          "applicationVersion": "1.0",
          "serviceInstanceOcid": "ocid5.exampleinstance.oc1.region.jjack",
          "pluginID": "cloud:google-sheets",
          "managedSecurity": "true",
          "integration_role": "TARGET",
          "oauth.scope": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
          "useProxy": "true",
          "securityPolicy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
          "proxyHost": "10.xx.xy.yy",
          "oauth.client.id": "813090821001-
uqpc1lsvauooc3lv8j3bh52cfs6kmfl4.apps.googleusercontent.com",
          "httpsProxyPort": "80",
          "oauth.client.secret": "gsYK8nYxxxxKx5p111188daJ",
          "oauth.auth.code.uri": "https://accounts.google.com/o/oauth2/auth",
          "oauth.access.token.uri": "https://www.googleapis.com/oauth2/v4/
token",
          "proxyPort": "80",
          "RABEngineVersion": "24.10",
          "csfMap": "example.wsm.security",
          "useCache": "true",
          "onPremise": "false",
          "clientAuthentication": "client_credentials_as_body",
          "httpsProxyHost": "10.xx.xy.yy",
          "csfkey":
"fac648317448763xxxxx2efd61a50c80243xxxxx2634ad87d396yyy7e5189038"
        },
        "input": {},
        "cloudAdapterUIBindingId": "04881642-6e72-8744-51f6-bea7270bf9bf",
        "configuration": {},
        "self": {
          "info": {
            "id": "cloud:google-sheets",
```

```
            "displayName": "Google Sheets",
            "description": "Google Sheets is a cloud-based sheet management
service.",
            "version": "1.0.0",
            "specVersion": "1.0",
            "categories": [],
            "appInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "publisherInfo": {
              "name": "Example Corporation",
              "description": "https://www.example.com/about",
              "contactUS": "http://www.example.com/corp/contact/index.html",
              "supportURL": "http://support.example.com",
              "documentationURL": "https://www.example.com/pls/topic/lookup?
ctx=DOCID"
            },
            "settings": {
              "supportsConnectivityAgent": true,
              "_debug": true
            }
          },
          "connection": {
            "connectionProperties": [],
            "securityPolicies": [
              {
                "type": "managed",
                "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
                "description": "Google Authorization Policy",
                "displayName": "Google Authorization Policy",
                "scope": "ACTION",
                "securityProperties": [
                  {
                    "name": "oauth.client.id",
                    "displayName": "Google Client ID",
                    "description": "Google Client ID",
                    "shortDescription": "Google Client ID",
                    "hidden": false,
                    "required": true
                  },
                  {
                    "name": "oauth.client.secret",
                    "displayName": "Google Client Secret",
                    "description": "Google Client secret",
                    "shortDescription": "Google Client secret",
                    "hidden": false,
                    "required": true
                  },
                  {
                    "name": "oauth.auth.code.uri",
                    "default": "https://accounts.google.com/o/oauth2/auth",
```

```
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "oauth.access.token.uri",
                    "default": "https://www.googleapis.com/oauth2/v4/token",
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "oauth.scope",
                    "default": "https://www.googleapis.com/auth/drive https://
www.googleapis.com/auth/spreadsheets",
                    "hidden": true,
                    "required": true
                  },
                  {
                    "name": "clientAuthentication",
                    "default": "client_credentials_as_body",
                    "hidden": true,
                    "required": true
                  }
                ],
                "authExtension": {
                  "authorizationURL": {
                    "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
                    "params": {
                      "query": {
                        "response_type": "code",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "redirect_uri": "${redirect_uri}",
                        "scope": "${.securityProperties.\"oauth.scope\"}",
                        "access_type": "offline",
                        "approval_prompt": "force"
                      }
                    }
                  },
                  "accessTokenRequest": {
                    "method": "POST",
                    "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                    "params": {
                      "query": {
                        "code": "${auth_code}",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "redirect_uri": "${redirect_uri}",
                        "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                        "grant_type": "authorization_code"
                      }
                    },
                    "headers": {
                      "Content-Type": "application/x-www-form-urlencoded",
                      "Authorization": "${\"Basic \" +
```

```
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
                    },
                    "body": "false"
                  },
                  "refreshTokenRequest": {
                    "method": "POST",
                    "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
                    "params": {
                      "template": {},
                      "query": {
                        "refresh_token": "${refresh_token}",
                        "client_id": "$
{.securityProperties.\"oauth.client.id\"}",
                        "client_secret": "$
{.securityProperties.\"oauth.client.secret\"}",
                        "grant_type": "refresh_token"
                      }
                    },
                    "headers": {
                      "Content-Type": "application/x-www-form-urlencoded"
                    },
                    "body": "false"
                  },
                  "fetchRules": {
                    "auth_code": "code",
                    "access_token": "access.[tT]oken",
                    "refresh_token": "refresh.[tT]oken",
                    "expiry": "expires.*",
                    "token_type": "token.?[tT]ype"
                  },
                  "accessTokenUsage": {
                    "headers": {
                      "Authorization": "Bearer ${access_token}"
                    }
                  }
                }
              }
            ],
            "test": "flow:testConnectionFlow"
          },
          "schemas": {
            "Sheet": {
              "type": "object",
              "properties": {
                "properties": {
                  "type": "object",
                  "properties": {
                    "sheetId": {
                      "type": "integer"
                    },
                    "title": {
                      "type": "string"
                    },
                    "index": {
```

```
            "type": "integer"
          },
          "sheetType": {
            "type": "string"
          }
        }
      }
    }
  },
  "Spreadsheet": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "spreadsheetUrl": {
        "type": "string"
      },
      "properties": {
        "type": "object",
        "properties": {
          "title": {
            "type": "integer"
          },
          "locale": {
            "type": "string"
          },
          "autoRecalc": {
            "type": "integer"
          },
          "timeZone": {
            "type": "string"
          }
        }
      },
      "sheets": {
        "type": "array",
        "items": {
          "$ref": "#/schemas/Sheet"
        }
      }
    }
  },
  "getSpreadsheetInput": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      }
    }
  },
  "createSpreadsheetInput": {
    "type": "object",
    "properties": {
      "spreadsheetTitle": {
        "type": "string"
```

```
        },
        "sheets": {
          "type": "array",
          "items": {
            "type": "object",
            "properties": {
              "sheetTitle": {
                "type": "string"
              }
            }
          }
        }
      }
    },
    "insertRowOutput": {
      "type": "object",
      "properties": {
        "spreadsheetId": {
          "type": "string"
        },
        "tableRange": {
          "type": "string"
        },
        "updates": {
          "type": "object",
          "properties": {
            "spreadsheetId": {
              "type": "string"
            },
            "updatedRange": {
              "type": "string"
            },
            "updatedRows": {
              "type": "integer"
            },
            "updatedColumns": {
              "type": "integer"
            },
            "updatedCells": {
              "type": "integer"
            }
          }
        }
      }
    },
    "getRowInput": {
      "type": "object",
      "properties": {
        "rowNumber": {
          "type": "integer"
        }
      }
    }
  },
  "actions": {
    "insertRowAction": {
```

```
    "displayName": "Insert Row Into Sheet",
    "description": "This action insert a new row into sheet.",
    "execute": "flow:insertRowFlow",
    "input": "flow:sheetSchemaFlow",
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/insertRowOutput"
      }
    },
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      }
    ]
  },
  "getRowAction": {
    "displayName": "Get Row From Sheet",
    "description": "This action get a row from sheet.",
    "execute": "flow:getRowFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/getRowInput"
      }
    },
    "output": "flow:sheetSchemaFlow",
    "configuration": [
      {
        "name": "spreadsheetId",
        "displayName": "Spreadsheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:spreadsheetIdFlow",
        "required": true
      },
      {
        "name": "sheetId",
```

```
        "displayName": "Sheet Name",
        "description": "",
        "type": "COMBO_BOX",
        "options": "flow:sheetIdFlow",
        "required": true,
        "dependencies": {
          "spreadsheetId": {
            "values": []
          }
        }
      },
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "currentRow",
            "displayName": "Get Specific Row"
          },
          {
            "keyName": "lastRow",
            "displayName": "Get Last Row"
          }
        ]
      }
    ]
  },
  "createSpreadsheetAction": {
    "displayName": "Create Spreadsheet",
    "description": "This action creates a new spreadsheet.",
    "execute": "flow:createSpreadsheetFlow",
    "input": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/createSpreadsheetInput"
      }
    },
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/Spreadsheet"
      }
    }
  },
  "insertRowActionErrorEngine": {
    "displayName": "Insert Row Into Sheet(Engine Error for Demo)",
    "description": "This action insert a new row into sheet.",
    "execute": "flow:insertRowFlow",
    "input": "flow:sheetSchemaFlow",
    "output": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/insertRowOutput"
```

```
                    }
                  },
                  "configuration": [
                    {
                      "name": "spreadsheetId",
                      "displayName": "Spreadsheet Name",
                      "description": "",
                      "type": "COMBO_BOX",
                      "options": "flow:spreadsheetIdFlow",
                      "required": true
                    },
                    {
                      "name": "sheetId",
                      "displayName": "Sheet Name",
                      "description": "",
                      "type": "COMBO_BOX",
                      "options": "flow:sheetIdFlowErrorEngine",
                      "required": true,
                      "dependencies": {
                        "spreadsheetId": {
                          "values": []
                        }
                      }
                    }
                  ]
                }
              },
              "flows": {
                "insertRowFlow": {
                  "id": "insertRowFlow",
                  "description": "insertRowFlow",
                  "specVersion": "0.8",
                  "version": "0.1",
                  "start": "startState",
                  "functions": [
                    {
                      "name": "generalRestFunc",
                      "type": "custom",
                      "operation": "connectivity::rest"
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "name": "insertRowAction",
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}:append",
                              "method": "POST",
                              "parameters": {
                                "spreadsheetId": "$
```

```
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                            "insertDataOption": "INSERT_ROWS",
                            "valueInputOption": "RAW"
                          },
                          "body": "${ {range: (.configuration.sheetId + \"!
A1:Z1\"), majorDimension:\"ROWS\", values: .input | to_entries |
[map(.value)] } }"
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "getRowFlow": {
              "id": "getRowFlow",
              "description": "getRowFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                },
                {
                  "name": "convertResult",
                  "type": "expression",
                  "operation": "(if .configuration.rowOperation ==
\"lastRow\" then (.lastLineNumber-1) else (.input.rowNumber-1) end)
as $rowNum | .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
```

```
                              "range": "${ .configuration.sheetId + \"!
A1:Z40000\" }",
                              "majorDimension": "ROWS"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.values | length }",
                          "toStateData": "${ .lastLineNumber }"
                        }
                      },
                      {
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                              "range": "${ if .configuration.rowOperation ==
\"lastRow\" then .configuration.sheetId + \"!A1:Z\" + (.lastLineNumber|
tostring) else .configuration.sheetId + \"!A1:Z\" + (.input.rowNumber|
tostring) end }",
                              "majorDimension": "COLUMNS"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.values }",
                          "toStateData": "${ .values }"
                        }
                      },
                      {
                        "functionRef": "convertResult",
                        "actionDataFilter": {
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "createSpreadsheetFlow": {
                "id": "createSpreadsheetFlow",
                "description": "createSpreadsheetFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
```

```
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://sheets.googleapis.com/v4/
spreadsheets",
                              "method": "POST",
                              "body": "${ {properties:
{title:.input.spreadsheetTitle},sheets:.input.sheets|map({properties:
{title:.sheetTitle}})} }"
                            }
                          },
                          "actionDataFilter": {
                            "results": "${ .body }",
                            "toStateData": "${ .output }"
                          }
                        }
                      ],
                      "end": true
                    }
                  ]
                },
                "testConnectionFlow": {
                  "id": "testConnectionFlow",
                  "description": "testConnectionFlow",
                  "specVersion": "0.8",
                  "version": "0.1",
                  "start": "startState",
                  "functions": [
                    {
                      "name": "generalRestFunc",
                      "type": "custom",
                      "operation": "connectivity::rest"
                    }
                  ],
                  "states": [
                    {
                      "name": "startState",
                      "type": "operation",
                      "actions": [
                        {
                          "functionRef": {
                            "refName": "generalRestFunc",
                            "arguments": {
                              "uri": "https://www.googleapis.com/drive/v3/files",
                              "method": "GET"
                            }
                          },
                          "actionDataFilter": {
```

```
                        "results": "${ { success: (.status==200),
message: .body.error.message } }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "spreadsheetIdFlow": {
              "id": "spreadsheetIdFlow",
              "description": "spreadsheetIdFlow",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSpreadsheetIDs",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://www.googleapis.com/drive/v3/files",
                          "method": "GET",
                          "parameters": {
                            "q": "mimeType='application/vnd.google-
apps.spreadsheet' and trashed=false",
                            "pageSize": 1000
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlow": {
              "id": "sheetIdFlow",
              "description": "sheetIdFlow",
```

```
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  }
                ],
                "states": [
                  {
                    "name": "startState",
                    "type": "operation",
                    "actions": [
                      {
                        "name": "getAllSheetIDsBySpreadsheetId",
                        "functionRef": {
                          "refName": "generalRestFunc",
                          "arguments": {
                            "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                            "method": "GET",
                            "parameters": {
                              "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                            }
                          }
                        },
                        "actionDataFilter": {
                          "results": "${ .body.sheets |
map({keyName:.properties.title, displayName:.properties.title}) }",
                          "toStateData": "${ .output }"
                        }
                      }
                    ],
                    "end": true
                  }
                ]
              },
              "sheetSchemaFlow": {
                "id": "sheetSchemaFlow",
                "description": "sheetSchemaFlow",
                "specVersion": "0.8",
                "version": "0.1",
                "start": "startState",
                "functions": [
                  {
                    "name": "generalRestFunc",
                    "type": "custom",
                    "operation": "connectivity::rest"
                  },
                  {
                    "name": "constructReturnObject",
                    "type": "expression",
                    "operation": "{\"schemaType\": \"application/schema+json\",
```

```
\"schema\": .schema}"
                }
              ],
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "generateSchemaBySheetHeader",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}/values/{range}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }",
                            "range": "${ .configuration.sheetId + \"!
A1:Z1\" }",
                            "majorDimension": "ROWS"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
                        "toStateData": "${ .schema }"
                      }
                    },
                    {
                      "name": "addSchemaType",
                      "functionRef": "constructReturnObject",
                      "actionDataFilter": {
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            },
            "sheetIdFlowErrorEngine": {
              "id": "sheetIdFlowErrorEngine",
              "description": "sheetIdFlowErrorEngine",
              "specVersion": "0.8",
              "version": "0.1",
              "start": "startState",
              "functions": [
                {
                  "name": "generalRestFunc",
                  "type": "custom",
                  "operation": "connectivity::rest"
                }
              ],
```

```
              "states": [
                {
                  "name": "startState",
                  "type": "operation",
                  "actions": [
                    {
                      "name": "getAllSheetIDsBySpreadsheetId",
                      "functionRef": {
                        "refName": "generalRestFunc",
                        "arguments": {
                          "uri": "https://sheets.googleapis.com/v4/
spreadsheets/{spreadsheetId}",
                          "method": "GET",
                          "parameters": {
                            "spreadsheetId": "$
{ .configuration.spreadsheetId }"
                          }
                        }
                      },
                      "actionDataFilter": {
                        "results": "${ .sheets | map({keyName:.title,
displayName:.title}) }",
                        "toStateData": "${ .output }"
                      }
                    }
                  ],
                  "end": true
                }
              ]
            }
          }
        },
        "output": [
          {
            "keyName": "1WxoiFLblMO6zeN7qkZkjnopI8_6vvfFB84dvbrZIdh0",
            "displayName": "Customer Managment"
          },
          {
            "keyName": "1_7maXDpc_qMMRzMm0x6d90-Hj6vvhGyO2kyFX0tpuZs",
            "displayName": "HL7 Message - FHIR R4: VXU_V04"
          },
          {
            "keyName": "1evBW_GzccQVcMM4OcBGXBiJOHhuywdahjQnAl5a9MjY",
            "displayName": "Copy of Restricted Stock Unit (RSU) Calculator"
          },
          {
            "keyName": "1DjtOl1dvZPMT0O0LJZZts9foehDdRoicmfTP022Fkw4",
            "displayName": "Development Management"
          },
          {
            "keyName": "1GT8i-J_PmgZwK0oer_VsHG1ev2JaR8jASK5Va7IUWEU",
            "displayName": "Event Attendee List"
          },
          {
            "keyName": "1Ol0b0Et3MccQsXfHI4tAqvabENSHE6c1HnU2kDfYU0E",
            "displayName": "HR Management"
```

```
            },
            {
              "keyName": "1MEv_Ifg87I0Qoe86RUp4cdOrVnQgcj8NLU1HKjHkKso",
              "displayName": "Financial Managment"
            },
            {
              "keyName": "1yOVioh6TuYayc78Sbgwo6yxsbOIZE7IQpKqXomJ-qdA",
              "displayName": "UAT-File"
            },
            {
              "keyName": "1ZveTrV1yDf92X_VPiE8FPDslUotfSiX7ztHfjCzWCQc",
              "displayName": "SalesAccounts"
            }
          ]
        }
      }
    ]
}
```

Change the Logo of an Adapter

You can change an adapter's logo that appears in Oracle Integration.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. To delete the existing logo, right-click the **logo.svg** file, and then click **Delete**.
3. Drag and drop the new logo image file (in the `.svg` format) under the root directory.

10

Review Schemas

The VS Code extension for Rapid Adapter Builder generates schemas for requests and responses when you convert a Postman collection to an adapter definition document. This information appears in the `schemas` section of the document. You can update this information as needed for your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Schemas Where Schemas Appear in Oracle Integration
Review the properties to define and see sample code	Schemas Properties and Sample Code

Learn About Schemas

Before reviewing or updating the `schema` section of an adapter definition document, familiarize yourself with schemas and their properties.

Topics:

- Introduction to Schemas
- Where Schemas Appear in Oracle Integration
- Schemas Properties and Sample Code

Introduction to Schemas

In the `schemas` section of an adapter definition document, you can review and update the schemas that the extension has generated from an application's APIs.

The JSON schemas define the structure of the input and output payloads. Schemas can reference other definitions in the adapter definition document. Generally, you don't need to update the schemas. However, if necessary, you can edit the schemas to:

- Remove certain fields that may not be necessary for your adapter.
- Remove nesting and have a flat list.
- Change the response format. For example, an API may return a time stamp as a UNIX epoch. You can transform the value into a human-readable format by editing the schema.

Where Schemas Appear in Oracle Integration

In Oracle Integration, you can view the schema information while configuring the mapping for an adapter in an integration flow.

The schema objects appear in the mapper as source or target data structures. See About Mapping Data Between Applications in *Using the Oracle Mapper with Oracle Integration 3*.

Schemas Properties and Sample Code

The `schemas` section of an adapter definition document typically contains the properties listed here and has the following sample structure.

- Properties
- Sample Code

Properties

Property	Description
`type`	Type of schema. The Rapid Adapter Builder platform supports the following schema types: • `array` • `object` • `string` • `boolean` • `number` • `integer`
`title`	Title of the schema. Use this property to display a name for a field on the user interface in Oracle Integration.
`description`	Description of the schema. Use this property to describe a field on the user interface in Oracle Integration.
`format`	Format of a schema. A supplement of `type` field and supports only binary format. Converts into binary type in Oracle Integration.
`required`	Indicates that the schema field is required. The asterisk symbol denotes all the required fields on the user interface.
`items`	Sub-schemas of a schema and is of type `array`.

Sample Code

```
"schemas": {
  "Task": {
    "title": "Task",
    "type": "object",
    "required": [
      "title"
    ],
    "properties": {
      "kind": {
        "type": "string",
        "title": "Kind"
      },
      "id": {
        "type": "string",
        "title": "Id"
      }
    }
  },
  "createTaskInput": {
    "type": "object",
```

```
    "properties": {
      "headers": {
        "type": "object",
        "properties": {
          "x-group-id": {
            "type": "string"
          }
        }
      },
      "parameters": {
        "type": "object",
        "properties": {
          "tasklist": {
            "type": "string"
          }
        }
      },
      "body": {
        "$ref": "#/schemas/Task"
      }
    }
  },
  "createTaskOutput": {
    "type": "object",
    "properties": {
      "headers": {
        "type": "object",
        "properties": {
          "x-transaction-id": {
            "type": "string"
          }
        }
      },
      "body": {
        "$ref": "#/schemas/Task"
      }
    }
  }
}
```

11

Update Connection Definitions

The VS Code extension for Rapid Adapter Builder collects connection and security information about the application's APIs when you convert a Postman collection or an OpenAPI document into an adapter definition document. This information appears in the `connection` section. You can update this information and add new connection definitions as needed, according to your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Connection Definitions Where Connection Definitions Appear in Oracle Integration
Review the properties to define and see sample code	Connection Properties and Sample Code
Modify the behavior of an existing connection or create a new connection	Work with Connection Definitions Learn About Security Policies for Invoke Connections Learn About Security Policies for Trigger Connections

Learn About Connection Definitions

Before updating the `connection` section of the adapter definition document, familiarize yourself with connections and their properties.

Topics:

- Introduction to Connection Definitions
- Where Connection Definitions Appear in Oracle Integration
- Connection Properties and Sample Code

Introduction to Connection Definitions

When designing your adapter, you specify the connection models that the adapter supports. Provide this information in the `connection` section of the adapter definition document. An integration developer must specify this information when creating a connection that is based upon the adapter.

Unfamiliar with Connections?

If you haven't worked with connections in Oracle Integration, here's a quick introduction.

Oracle Integration communicates with an application in a two-way conversation using the following types of connections:

- Trigger connections, which allow an adapter to authenticate a third-party application that calls an integration.

  After authentication, the connection securely exposes an integration to the application.

  You define the authentication, authorization, and verification schemes that the adapter uses for these inbound requests.

- Invoke connections, which allow an adapter to securely authenticate and connect to the third-party applications that the integration calls.

  After authentication, the connection securely exposes the application to the integration.

  You define the authentication and authorization schemes that the adapter uses for these outbound requests.

To learn about how integration developers define connections, see Configure Connection Properties and Security Properties in *Using Integrations in Oracle Integration 3*.

Information to Define for Connection Definitions

Define the following information in the `connection` section of the adapter definition document.

Information to define	Description
Connection properties	Your adapter allows integration developers to create connections. The connections are based on connection definitions, which you specify in the adapter definition document. Each connection definition includes connection properties. The connection properties provide the information that an integration needs to connect with a third party, including details that are specific to the application's API endpoints and other environment-specific information. Connection properties do not include information that is considered sensitive or related to security. Security-related information is part of the security policies for a connection definition. See: • Connection Properties and Sample Code • Review a Connection Definition
Security policies	An invoke connection that is based on your adapter must authenticate itself with the endpoint that it connects to. You specify the authentication and authorization schemas in the adapter definition document. Keep in mind the following points about these security policies: • The adapter definition document supports a set of managed security policies, including basic authentication, OAuth 2.0 Client Credentials, OAuth 2.0 Authorization Code, and more. The security policies in the adapter definition document are standards-based templates. You can update the values as needed and extend some policies, but you cannot modify the way that the security policies are represented. • The properties of security policies are sensitive and therefore are saved securely in a vault. After an integration developer saves the values for a security policy in Oracle Integration, the values become masked when the integration developer opens the page again. See: • Connection Properties and Sample Code • Review a Connection Definition

Information to define	Description
Testing requirements	You can model the testing of your connection in the adapter definition document. Integration developers who create connections that are based on your adapter can test their connections while designing an integration.

Where Connection Definitions Appear in Oracle Integration

You can view the connection information while creating a connection using an adapter in Oracle Integration.

On the Connection Configuration page in Oracle Integration:

- The data you enter for the `connectionProperties` object appears in the **Properties** section.
- The data you enter for the `securityPolicies` object appears in the **Security** section.
  - If you define one security policy, it appears as a default value.
  - If you define two or more security policies, they appear in a drop-down list.

Connection Properties and Sample Code

In the `connection` section of the adapter definition document, you specify how your adapter connects to applications.

- Properties in the connectionProperties Section
- Properties in the securityPolicies Section
- Sample Code: Define a Connection Definition
- Sample Code: Define a Connection Property
- Sample Code: Define Security Policies within a Connection Definition

Properties in the connectionProperties Section

In `connectionProperties`, you define the behavior of a single connection property. Keep in mind the following points:

- Each collection of properties in `connectionProperties` is related to a single connection property.

  The connection property appears as a field in Oracle Integration when an integration developer configures a connection. For instance, the `displayName` property defines the property name that appears in the user interface, the `shortDescription` property provides brief hint text, and so on.

  The collection of properties model the behavior of the connection property.
- You can specify whether a connection property in Oracle Integration appears in an invoke connection, a trigger connection, or both.
- Many properties below define metadata. However, pay particular attention to the functional properties: `required`, `hidden`, and `scope`.

`connectionProperties` includes the following properties.

Property	Description
`name`	Unique name for the connection property. The value must contain no spaces. The value does not appear in the Oracle Integration user interface. However, you use this value to reference the property from within the adapter definition document.
`displayName`	Name of the connection property. This value appears as the name of the property in the Oracle Integration user interface when an integration developer creates a connection using the adapter. See Update the Name or Description of a Connection Property.
`shortDescription`	Short hint text for the connection property. This text appears in the Oracle Integration user interface.
`description`	Description of the property. This description appears on the Oracle Integration user interface when you select a property to update or modify.
`type`	Data type of the property. The following data types are supported: • `STRING` • `NUMBER` • `URL` • `PASSWORD` • `TEXT` • `CHOICE` • `BOOLEAN`
`required`	Either `true` or `false`, indicating whether the adapter requires a value for the property. When `true`, the property has an asterisk in the Oracle Integration user interface, and integration developers must provide a value. Sometimes, an adapter requires a value for a property, and the value is always the same. In such cases, consider setting `required` and `hidden` to `true` and specifying the value using the `default` property. See Hide a Connection Property.
`hidden`	Either `true` or `false`, indicating whether the property is hidden to integration developers. If you specify `true`, indicating that the property is hidden, you must provide a default value using the `default` property. For example, you might set the `releaseVersion` parameter to hidden and specify its default value to `1A`. See Hide a Connection Property.
`tokenized`	Either `true` or `false`, indicating the location in which the value of the property is stored: • When `true`, the value is stored in a database, and a reference to the value is saved with the integration. If an integration developer exports the integration and imports it into another environment, the value is not included in the import. The developer must specify a new value for the property while configuring the connection in the target environment. • When `false`, the value is saved as part of the integration. If an integration developer exports the integration and imports it into another environment, the integration continues to work with the fixed value that the developer specified.
`default`	Default value for the property. This value appears in the property field in the Oracle Integration user interface.

Property	Description
`scope`	Type of connection that the connection property applies to: • `TRIGGER`: The connection property appears only when integration developers configure trigger connections, which securely expose integration flows to external applications. • `ACTION`: The connection property appears only when integration developers configure invoke connections, which securely connect to and invoke external applications. For example, you might configure ten connection properties, five for trigger connections and five for invoke connections.
`options`	Provide a predefined list of values in a drop-down list. An integration developer can select from the values when creating a connection in Oracle Integration. Below the options property, add one or more sets of the following properties: • `keyName`: Internal name of the value. • `displayName`: User-facing name of the value. This value appears to integration developers in a drop-down list for the connection property. See Create a Connection Property with a Drop-Down List.

Properties in the securityPolicies Section

In `securityPolicies`, define one or more security policies for a connection definition. Security policies define how a connection accesses protected endpoints and Oracle Integration.

When you convert a Postman collection or an OpenAPI document into an adapter definition document, the Rapid Adapter Builder collects security policy information about the APIs and enters it into the `securityPolicies` section. If needed, you can update some security policy details.

The Rapid Adapter Builder supports a number of managed security policies, including several extensible policies. You can't add and modify properties of managed security policies. However, you can make a number of updates, including hiding security policy properties and providing a default value instead, and changing the display name and description of a managed security policy. See Understand Security Policies in Connection Definitions.

You can define one or more security policies in each connection definition. Each security policy can contain one or more security properties. For each entry in the `securityProperties` section, you must define the following set of properties. These properties describe the behavior of the field that the integration developer interacts with while configuring a connection.

`securityPolicies` includes the following properties.

Property	Description
`type`	Usage of the security policy: • `managed`: The security policy is an authentication and authorization scheme that is natively implemented in the Rapid Adapter Builder. Additionally, the scheme is a managed security policy that governs one of the following types of connections: – Trigger connections, which securely expose integration flows to external applications. – Invoke connections, which securely connect to and invoke external applications. • `composite`: The security policy is an extension of a managed security policy and wraps two policies in one. A composite policy governs inbound (trigger) connections that also must make outbound (invoke) calls. • `selfManaged`: Reserved for future use.
`description`	Description of the security policy.
`displayName`	Name of the security policy. This value appears as the name of the security policy in the Oracle Integration user interface when an integration developer creates a connection using the adapter.
`scope`	Role for the security policy: • `TRIGGER`: The security policy is used for a trigger operation. • `ACTION`: The security policy is used for an invoke operation. **Note:** You specify a scope for both a connection property and a security policy. For example, you might define four connection properties for invoke connections and six connection properties for trigger connections. Similarly, you might define two security policies for invoke connections and one security policy for trigger connections.

Property	Description
`policy`	Name of the security policy that is used to access protected endpoints. You adapter can expose one or more of the following policies: • Security policies for invoke connections: – BASIC_AUTH – API_KEY_AUTHENTICATION – OCI_SIGNATURE_VERSION1 – OAUTH_ONE_TOKEN_BASED – OAUTH_RESOURCE_OWNER_PASSWORD_CREDENTIALS – OAUTH_CLIENT_CREDENTIALS – OAUTH_AUTHORIZATION_CODE_CREDENTIALS – OAUTH_JWT_USER_ASSERTION – OAUTH_JWT_CLIENT_ASSERTION • Security policies for trigger connections: – BASIC_AUTH – OAUTH2.0_TOKEN_VALIDATION – HMAC_SIGNATURE_VALIDATION – RSA_SIGNATURE_VALIDATION – JWT_VALIDATION
`policyInbound`	`policyInbound` is required when the `type` property is `composite`. `policyInbound` is a complex object and requires the following properties: • `type`: One of the following values: – `managed`: The inbound security policy is a predefined managed security policy. – `nonManaged`: Reserved for future use. • `policy`: See the `policy` row. • `securityProperties`: See the `securityProperties` row.
`policyOutbound`	`policyOutbound` is required when the `type` property is `composite`. `policyOutbound` is a complex object and requires the following properties: • `type`: One of the following values: – `managed`: The outbound security policy is a predefined managed security policy. – `nonManaged`: Reserved for future use. • `policy`: See the `policy` row. • `securityProperties`: See the `securityProperties` row.

Property	Description
`securityProperties`	Properties for each security policy. Most security policies require several sets of these properties. Each property that you define appears on the page for configuring connection security. Integration developers must specify the appropriate values for each property. • `name`: Name used to reference the security policy property. • `displayName`: Name of the security policy property that appears on the connection page. See Change the Display Name or Description of a Security Property • `shortDescription`: Short hint text for the security property. • `description`: Description of the security property. See Change the Display Name or Description of a Security Property • `type`: The data type of the property. The following data types are supported: – `STRING` – `NUMBER` – `URL` – `PASSWORD` – `TEXT` – `CHOICE` – `BOOLEAN` • `required`: Specifies whether it is a required property. If set to true, the property is marked as required on the connection page and the integration developer must provide a value. • `hidden`: Hides the property. If set to true, the property does not appear on the connection page. You must specify a default value of you are hiding a property. See Hide a Security Property. To see the predefined values for each of these properties security policy, see the appropriate link: • Outbound security policies: – HTTP Basic Authentication – API Key Based Authentication – OCI Signature Version 1 – OAuth 1.0 One-Legged Authentication – OAuth Resource Owner Password Credentials – OAuth Client Credentials – OAuth Authorization Code Policy – OAuth JWT User Assertion – OAuth JWT Client Assertion • Inbound security policies – Basic Authentication – OAuth 2.0 Access Tokens – HMAC Signature Validation – RSA Signature Validation – JWT Validation

Property	Description
`authExtension`	The default implementation of these policies is RFC 6749 compatible. However, in practice, many implementations different from the RFC guidelines. Using the `authExtension` properties, you can extend the steps of the OAuth flow in the security policy. You can extend the following OAuth policies if needed: • OAuth Resource Owner Password Credentials • OAuth Client Credentials • OAuth Authorization Code Policy

Sample Code: Define a Connection Definition

The following code sample defines a connection definition for the Square application using the OAuth Authorization Code Credentials authentication mechanism.

This sample includes one `connectionProperties` section, one `securityPolicies` section, and several `securityProperties` sections.

```
"connection": {
  "connectionProperties": [
    {
      "name": "baseURL",
      "type": "URL",
      "displayName": "Base URL",
      "description": "Base URL of the endpoints",
      "shortDescription": "Base URL of the endpoints",
      "default": "https://connect.squareupsandbox.com/v2",
      "required": true,
      "hidden": true,
      "scope": [
        "ACTION"
      ]
    }
  ],
  "securityPolicies": [
    {
      "type": "managed",
      "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
      "description": "Authorization Policy",
      "displayName": "Authorization Policy",
      "scope": "ACTION",
      "securityProperties": [
        {
          "name": "oauth.client.id",
          "displayName": "Square Application ID",
          "description": "Square Application ID",
          "shortDescription": "Square Application ID",
          "hidden": false,
          "required": true
        },
        {
          "name": "oauth.client.secret",
          "displayName": "Square Application Secret",
          "description": "Square Application Secret",
```

```
            "shortDescription": "Square Application Secret",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.auth.code.uri",
            "default": "https://connect.squareupsandbox.com/oauth2/authorize",
            "hidden": true,
            "required": true
          },
          {
            "name": "oauth.access.token.uri",
            "default": "https://connect.squareupsandbox.com/oauth2/token",
            "hidden": true,
            "required": true
          },
          {
            "name": "oauth.scope",
            "default": "CUSTOMERS_WRITE ORDERS_WRITE INVOICES_WRITE
CUSTOMERS_READ",
            "hidden": true,
            "required": true
          },
          {
            "name": "clientAuthentication",
            "default": "client_credentials_in_body",
            "hidden": true,
            "required": true
          }
        ],
        "authExtension": "flow:authExtension"
      }
    ],
    "test": "flow:testConnectionFlow"
}
```

Sample Code: Define a Connection Property

The following sample code shows how to define two connection properties. The first part of the sample demonstrates the definition of an editable property, and the second part demonstrates the definition of a hidden property.

```
"connectionProperties": [
  {
    "name": "accountid",
    "type": "string",
    "displayName": "Account ID",
    "shortDescription": "Enter the account id.",
    "required": false,
    "hidden": false,
      "scope": [
        "ACTION"
      ]
  },
   {
    "name": "baseURL",
```

```
    "type": "URL",
    "displayName": "Base URL",
    "shortDescription": "Ensure that this value corresponds to the type
selected.",
    "required": false,
    "hidden": true,
    "default": "https://connect.abcxyz.com/v2",
    "scope": "ACTION"
  }
]
```

Sample Code: Define Security Policies within a Connection Definition

The following sample code shows how to define security policies. In this case, the connection supports two managed security policies, each of which has a combination of displayable and hidden security properties.

```
"connection": {
    "connectionProperties": [
    ],
    "securityPolicies": [
      {
        "type": "managed",
        "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
        "description": "ABC Company Authorization Code Policy",
        "displayName": "ABC Company Authorization Code Policy",
        "scope": "ACTION",
        "securityProperties": [
          {
            "name": "oauth.client.id",
            "displayName": "ABC Company Client ID",
            "description": "ABC Company Client ID",
            "shortDescription": "ABC Company Client ID",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.client.secret",
            "displayName": "ABC Company Client Secret",
            "description": "ABC Company Client secret",
            "shortDescription": "ABC Company Client secret",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.auth.code.uri",
            "default": "https://login.ABC Companyonline.com/common/oauth2/
v2.0/authorize",
            "hidden": true,
            "required": true
          },
          {
            "name": "oauth.access.token.uri",
            "default": "https://login.ABC Companyonline.com/common/oauth2/
v2.0/token",
            "hidden": true,
```

```
            "required": true
          },
          {
            "name": "oauth.scope",
            "default": "Tasks.ReadWrite offline_access",
            "hidden": true,
            "required": true
          },
          {
            "name": "clientAuthentication",
            "default": "client_credentials_as_header",
            "hidden": true,
            "required": true
          }
        ]
      },
      {
        "type": "managed",
        "policy": "OAUTH_RESOURCE_OWNER_PASSWORD_CREDENTIALS",
        "description": "ABC Company Resource Owner Password Credentials
Policy",
        "displayName": "ABC Company Resource Owner Password Credentials
Policy",
        "scope": "ACTION",
        "securityProperties": [
          {
            "name": "oauth.access.token.uri",
            "default": "https://login.ABC Companyonline.com/
4a4a30f3-247a-47b6-acef-55ba707ea6df/oauth2/v2.0/token",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.client.id",
            "displayName": "ABC Company Client ID",
            "description": "ABC Company Client ID",
            "shortDescription": "ABC Company Client ID",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.client.secret",
            "displayName": "ABC Company Client Secret",
            "description": "ABC Company Client secret",
            "shortDescription": "ABC Company Client secret",
            "hidden": false,
            "required": true
          },
          {
            "name": "username",
            "displayName": "User Name",
            "description": "User Name",
            "shortDescription": "User Name",
            "hidden": false,
            "required": true
          },
```

```
            {
              "name": "password",
              "displayName": "Password",
              "description": "Password",
              "shortDescription": "Password",
              "hidden": false,
              "required": true
            },
            {
              "name": "oauth.scope",
              "default": "Tasks.ReadWrite offline_access",
              "hidden": true,
              "required": true
            },
            {
              "name": "oauth.request.content.type",
              "default": "application/x-www-form-urlencoded",
              "hidden": true,
              "required": true
            },
            {
              "name": "clientAuthentication",
              "default": "client_credentials_as_header",
              "hidden": true,
              "required": true
            }
          ]
        }
      ],
      "test": "flow:testConnectionFlow"
    },
```

Work with Connection Definitions

In the `connection` section of the adapter definition document, formulate the connection definitions for your adapter.

When you convert a Postman collection (or an OpenAPI document) into an adapter definition document, the VS Code extension automatically populates the connection and security information from the application's APIs. Review and update these connection definitions to meet your requirements, or create additional connection definitions if needed.

What Do You Want to Do?

Task	More information
Review the connection definitions automatically populated by the VS Code extension.	Review a Connection Definition
Implement a new connection definition in the adapter definition document	Implement a New Connection Definition

Task	More information
Learn about common updates to connection definitions	**Optional:** • Allow Integration Developers to Test Connections • Hide a Connection Property • Update the Name or Description of a Connection Property • Create a Connection Property with a Drop-Down List • Hide a Security Property • Change the Display Name or Description of a Security Property

Review a Connection Definition

Review and update the connection definitions that are automatically populated by the VS Code extension, while generating an adapter definition document from a Postman collection (or an OpenAPI document).

To update an existing connection definition, open the adapter definition document in the VS Code editor.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `connection` section.
4. Review the specifications for the existing connection definitions, and update them if needed. To understand the properties, see Connection Properties and Sample Code and Understand Security Policies in Connection Definitions.
5. Press **Ctrl+S** to save the document.

Implement a New Connection Definition

Use the authentication scheme templates to implement new connection definitions in the `connection` section of an adapter definition document.

To implement a new connection definition, open the adapter definition document in the VS Code editor.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Right-click anywhere in your document to view the context menu, and then click **RAB: Insert Authentication Scheme**.

   A list of authentication schemes is displayed.

4. Select an authentication scheme from the list to insert the corresponding template into your document.
   a. For invoke connections, you can insert the following authentication schemes:
      - HTTP Basic Authentication
      - API Key Based Authentication
      - OCI Signature Version 1
      - OAuth 1.0 One-Legged Authentication
      - OAuth Authorization Code Policy
      - OAuth Client Credentials
      - OAuth Resource Owner Password Credentials
      - OAuth JWT User Assertion
      - Oauth JWT Client Assertion
   b. For trigger connections, you can insert the following authentication schemes:
      - Basic Authentication
      - OAuth2.0
      - HMAC Signature Validation
      - RSA Signature Validation
      - JWT Validation

   Under the `connection` object of your document, the `connectionProperties` and `securityPolicies` sections are populated as per the template.
5. Update the values in these sections according to your requirements. To understand the properties of each section, see the following topics:
   - For connection properties, see Properties in the connectionProperties Section.
   - For properties related to different security policies, see Learn About Security Policies for Invoke Connections and Learn About Security Policies for Trigger Connections.
6. Press **Ctrl+S** to save the document.

Learn About Common Updates to Connection Definitions

This section provides details about some common updates to the `connection` section of an adapter definition document.

Topics:

- Allow Integration Developers to Test Connections
- Hide a Connection Property
- Update the Name or Description of a Connection Property
- Create a Connection Property with a Drop-Down List
- Hide a Security Property
- Change the Display Name or Description of a Security Property

Allow Integration Developers to Test Connections

The adapter developer can implement the test connection functionality in the adapter definition document so that the integration developers can test the connections.

To validate a connection configuration, call a working API on the target system to validate the connection and security configuration. Oracle recommends to choose an API that has no business impact so that the test action doesn't have any business and technical implications. Reasons for a test connection failure can include:

- Instance is not reachable.
- Security information is insufficient to authenticate or obtain tokens.
- The acquired tokens are insufficient to access a given resource or are forbidden.

Managed security policies automatically inherit the validation built into the managed security policies. For example, for OAuth security policy, the relevant tokens are acquired and refreshed. You must ensure that while calling idempotent APIs, you must provide additional information to call for further validation and assert the calls.

Related Topics

- Implement the Test-Connection Behavior Using Flows
  This procedure describes how you can implement the Test Connection functionality using a flow.

Hide a Connection Property

Oracle recommends obtaining as minimal information as possible from integration developers. Therefore, when a connection property has the same value across all connections, assign a static value to the property and hide the property. Integration developers who create a connection don't see the property in the Oracle Integration user interface.

For example, consider an application for which all invoke connections point to a specific URL. You can hard code the domain for every invoke connection and hide the URL field. That way, you simplify the experience for integration developers.

You can also use properties as global variables and hide the properties so that they can be substituted within an integration.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.
2. Expand the `connection` section of the adapter definition document.
3. Add the `hidden` and `default` properties below the `required` property for the connection definition and provide values for the properties.

   For example, the following code sample specifies a default value for the `baseURL` property and hides the value using the `hidden` property.

> **Note:**
>
> The following example does not include the `displayName` and `description` properties because the scenario doesn't require them.

```
"connectionProperties": [
  {
    "name": "accountid",
    "type": "string",
    "displayName": "Account ID",
    "shortDescription": "Enter the account id.",
    "required": true,
    "hidden": false,
    "scope": "ACTION"
  },
   {
    "name": "baseURL",
    "type": "URL",
    "required": true,
    "hidden": true,
    "default": "https://connect.squareupsandbox.com/v2",
    "scope": "ACTION"
  }
]
```

4. Save your changes.

Update the Name or Description of a Connection Property

You can update the user-facing name and description of a connection property at any time. Integration developers see these values in the Oracle Integration user interface when configuring a connection.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.
2. Expand the `connection` section of the adapter definition document.
3. Update the following properties for the connection definition as needed.
   - `displayName`: User-facing name of the connection property.
   - `description`: User-facing hint text for the connection property.
   - `shortDescription`: User-facing description of the connection property.

   This properties appear below in bold.

```
"connectionProperties": [
  {
    "name": "accountid",
    "type": "string",
```

```
    "displayName": "Customer Account ID",
    "description": "Account ID",
    "shortDescription": "Enter the account id.",
    "required": false,
    "hidden": false,
    "scope": ["ACTION"]
  }
]
```

4. Save your changes.

Create a Connection Property with a Drop-Down List

For any connection property, you can provide a predefined list of values in a drop-down list. An integration developer can select from the values when creating a connection in Oracle Integration.

You cannot configure a drop-down list for a security property.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.

2. Expand the `connection` section of the adapter definition document.

3. Update the `options` property for the connection.

   For example, the following code sample shows how to define a drop-down list for the `connectionType` connection property. The drop-down list has two choices, `baseURI` and `Open API(1.0/2.0/3.0) URL`. The properties for the drop-down list appear in bold.

   For more information about the properties, see Properties in the connectionProperties Section.

```
"connectionProperties": [
  {
    "name": "connectionType",
    "type": "CHOICE",
    "displayName": "Connection Type",
    "shortDescription": "Connection Type",
    "required": true,
    "hidden": false,
    "scope": ["ACTION"],
    "options": [
      {
        "keyName": "baseURI",
        "displayName": "REST API Base URI"
      },
      {
        "keyName": "openapi",
        "displayName": "Open API(1.0/2.0/3.0) URL"
      }
    ]
  }
]
```

4. Save your changes.

Hide a Security Property

Each managed security policy has security properties. Integration developers see the security properties and provide values for them while defining a connection. However, when a value is static or is programmatically derived across all connections for the adapter, you can hide the security property.

For example, consider an application that supports OAuth 2.0 Authorization Code as a security policy. Every invoke connection must use that security policy. In such cases, you can hard code the security policy for every invoke connection and hide the security policy. That way, you simplify the experience for integration developers.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.

2. Expand the `connection` section of the adapter definition document.

3. Add the `default` and `hidden` properties below the `name` property for the connection and provide values for the properties.

   For example, the following code sample shows how to provide a default value to the `name` property and hide it. Keep in mind that for this sample, the password field still appears when the integration developer configures the connection.

```
{
      "type": "managed",
      "policy": "BASIC_AUTH",
      "description": "Username and password credentials.",
      "displayName": "Username and password credentials.",
      "scope": "ACTION",
      "securityProperties": [
        {
          "name": "username",
          "default" : "FIXED_USER",
          "hidden": true,
          "required": true
        },
        {
          "name": "password",
          "displayName": "Account Password",
          "description": "<Overrides the description>",
          "shortDescription": "<Override the short description>",
          "hidden": false,
          "required": true
        }
      ]
    }
```

4. Save your changes.

Change the Display Name or Description of a Security Property

You set the user-facing name and description of a managed security policy using its `displayName` and `description` properties. Integration developers see these values in the Oracle Integration user interface when configuring a connection. You can update these value at any time.

> **Note:**
>
> Do not change the names of the connection property, security policy, and security property. Changing these values can cause unexpected failures.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.
2. Expand the `connection` section of the adapter definition document.
3. Update the `displayName` or `description` properties for the security policy as needed.

   This properties appear below in bold.

```
{
  "type": "managed",
  "policy": "BASIC_AUTH",
  "description": "<Overrides the description of the security policy>",
  "displayName": "<Overrides the display name of the security policy>",
  "scope": "ACTION",
  "securityProperties": [
    {
      "name": "username",
      "displayName": "accountID",
      "description": "<Overrides the description>",
      "shortDescription": "<Overrides the short description>",
      "hidden": false,
      "required": true
    },
    {
      "name": "password",
      "displayName": "accountPassword",
      "description": "<Overrides the description>",
      "shortDescription": "<Overrides the short description>",
      "hidden": false,
      "required": true
    }
  ]
}
```

4. Save your changes.

Understand Security Policies in Connection Definitions

Learn about the security policies and their properties that you can include your connection definitions.

What Do You Want to Do?

Task	More information
Understand the properties related to security policies for invoke connections	Learn About Security Policies for Invoke Connections
Understand the properties related to security policies for trigger connections	Learn About Security Policies for Trigger Connections

Learn About Security Policies for Invoke Connections

The adapter definition document supports a number of managed outbound security policies for invoke connection definitions. These security policies allow integrations to securely connect to and invoke external applications.

Some of the managed security policies do support extensions in order to accommodate the authentication enhancements warranted by the applications.

Managed Security Policies for Invoke Connections:

- HTTP Basic Authentication
- API Key Based Authentication
- OCI Signature Version 1
- OAuth 1.0 One-Legged Authentication
- OAuth Resource Owner Password Credentials (extendable)
- OAuth Client Credentials (extendable)
- OAuth Authorization Code Policy (extendable)
- OAuth JWT User Assertion
- OAuth JWT Client Assertion

HTTP Basic Authentication

To support invoking of REST APIs secured with Basic Authentication, use the `BASIC_AUTH` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: Basic Authentication

Overview

With HTTP basic authentication, the client sends HTTP requests with the Authorization header that contains the word **Basic** followed by a space and a Base64 encoded string that contains a user name and password in the following format: `username:password`.

To add the HTTP Basic Authentication policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> Some applications use different names for `username`, such as `accountId`. You can modify the managed policy to override the default name of a property.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
username	Username	Use UserName	A username credential	String	Yes
password	Password	Enter Password	A password credential	Password	Yes

Sample Code: Basic Authentication

The following code sample shows the default configuration for HTTP basic authentication within a connection in the adapter definition document.

```
"securityPolicies": [
  {
    "type": "managed",
    "policy": "BASIC_AUTH",
    "description": "HTTP Basic Authentication",
    "displayName": "HTTP Basic Authentication",
    "scope": "ACTION",
    "securityProperties": [
      {
        "name": "username",
        "displayName": "Username",
        "description": "Registered username.",
        "shortDescription": "Example: AC1234",
        "hidden": false,
        "required": true
      },
      {
        "name": "password",
        "displayName": "Password",
        "description": "Password for the registered user.",
        "shortDescription": "Example: <password for the user>",
        "hidden": false,
        "required": true
      }
```

```
    ]
  }
]
```

API Key Based Authentication

To support invoking of REST APIs secured with an API key, use the `API Key Based Authentication` managed security policy.

- Overview
- Security Properties
- Sample Code: API Key Based Authentication

Overview

To add the API Key Based Authentication policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

You can customize the security policy as needed for a connection definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
accessToken	API Key	Generated API key.	Generated API Key used to identify the client that is making the request.	Password	Yes
accessTokenUsage	API Key Usage	Default: -H Authorization: Bearer ${api-key}	URI syntax to show how the API key is passed to access a protected resource. To pass the API key as a query parameter, specify the following value as a query parameter: "?<query-param-name>=${api-key}" To pass the API key as a header, specify the following value: "-H Authorization: Bearer ${api-key}"	Text	No

Sample Code: API Key Based Authentication

The following code sample shows how to define the security policy in a connection using API key based authentication.

```
"securityPolicies": [
   {
     "type": "managed",
     "policy": "API_KEY_AUTHENTICATION",
     "description": "API Key Based Authentication",
     "displayName": "API Key Based Authentication",
     "scope": "ACTION",
     "securityProperties": [
       {
         "name": "accessToken",
         "displayName": "API Key",
         "description": "Provide the API Key",
         "shortDescription": "Example: <The generated api key>",
         "hidden": false,
         "required": true
       },
       {
         "name": "accessTokenUsage",
         "hidden": true,
         "required": true,
         "default": "-H Authorization: Bearer ${api-key}"
       }
     ]
   }
]
```

OCI Signature Version 1

To support invoking of OCI REST APIs secured using an API Key, use the OCI_SIGNATURE_VERSION1 managed security policy.

- Overview
- Security Properties
- Sample Code: OCI Signature Version 1

Overview

To add the OCI Signature Version 1 policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

You can customize the security policy as needed for a connection definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
TenancyOCID	Tenancy OCID	Tenancy OCID of OCI	Tenancy OCID of OCI	String	Yes
UserOCID	User OCID	User OCID of OCI	User OCID of OCI	String	Yes
PrivateKey	Private Key	OCI Private Key	OCI Private Key	URL_OR_FILE	Yes
FingerPrint	Finger Print	Finger Print	Finger Print	String	Yes
PassPhrase	Pass Phrase	Password entered at the time of generating the public/ private key.	Password entered at the time of generating the public/ private key.	Password	No

Sample Code: OCI Signature Version 1

The following sample code shows how to define the OCI Signature Versio 1 policy in a connection.

```
"securityPolicies": [
  {
    "type": "managed",
    "policy": "OCI_SIGNATURE_VERSION1",
    "description": "OCI Signature Version 1",
    "displayName": "OCI Signature Version 1",
    "scope": "ACTION",
    "securityProperties": [
      {
        "name": "TenancyOCID",
        "displayName": "Tenancy OCID",
        "description": "Tenancy OCID of OCI.",
        "shortDescription": "Example: ocid1.tenancy.oc1..mockId",
        "hidden": false,
        "required": true
      },
      {
        "name": "UserOCID",
        "displayName": "User OCID",
        "description": "User OCID of OCI.",
        "shortDescription": "Example: ocid1.user.oc1..mockUser",
        "hidden": false,
        "required": true
      },
      {
        "name": "PrivateKey",
        "displayName": "Private Key",
        "description": "OCI Private Key File.",
        "shortDescription": "Upload the private key file",
        "hidden": false,
        "required": true
```

```
      },
      {
        "name": "FingerPrint",
        "displayName": "Finger Print",
        "description": "Key Finger Print.",
        "shortDescription": "Example:
4c:8e:79:e3:a3:31:b3:8f:18:71:93:0c:c4:49:52:1c",
        "hidden": false,
        "required": true
      },
      {
        "name": "PassPhrase",
        "displayName": "Pass Phrase",
        "description": "Password entered at the time of generating the public/
private key.",
        "shortDescription": "Password entered at the time of generating the
public/private key.",
        "hidden": false,
        "required": true
      }
    ]
  }
]
```

OAuth 1.0 One-Legged Authentication

To support invoking of OAuth 1.0a-secured APIs, use the `OAUTH_ONE_TOKEN_BASED` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: OAuth 1.0a (One Legged)

Overview

OAuth 1.0a (One Legged) enables a client to make authenticated HTTP requests to gain access to protected resources by using their credentials.

This method includes two sets of credentials with each request. One set of credentials identifies the client, and the other set identifies the resource owner. Before a client makes authenticated requests on behalf of the resource owner, the client must obtain a token authorized by the resource owner.

To add the OAuth 1.0 One-Legged Authentication policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> By default, this security policy exposes all the fields to the integration developer in Oracle Integration. However, you can choose to hide fields that have fixed values.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
oauth_consumer_key	Consumer Key	Registered consumer key.	Used to identify the client that is making the request.	String	Yes
oauth_consumer_secret	Consumer Secret	Registered consumer Secret.	Used to authorize the client that is making the request.	Password	Yes
oauth_token	Token	Registered token	Used to access protected resource.	String	Yes
oauth_token_secret	Token Secret	Token secret provided by server.	Used for generating signature for the request.	Password	Yes
realm	Realm	Account identifier.	Used for identifying the account.	String	No

Sample Code: OAuth 1.0a (One Legged)

The following sample code shows the default configuration of the OAuth 1.0A One Legged authentication security policy:

```
  {
    "type": "managed",
    "policy": "OAUTH_ONE_TOKEN_BASED",
    "displayName": "OAuth 1.0 One Legged Authentication",
    "scope": "ACTION",
    "securityProperties": [
      {
        "name": "oauth_consumer_key",
        "displayName": "Consumer Key",
        "description": "Used to identify the client that is making the
request.",
        "shortDescription": "Registered consumer key.",
        "hidden": false,
```

```
        "required": true
      },
      {
        "name": "oauth_consumer_secret",
        "displayName": "Consumer Secret",
        "description": "Used to authorize the client that is making the
request.",
        "shortDescription": "Registered consumer Secret.",
        "hidden": false,
        "required": true
      },
      {
        "name": "oauth_token",
        "displayName": "Token",
        "description": "Used to access protected resource.",
        "shortDescription": "Registered token.",
        "hidden": false,
        "required": true
      },
      {
        "name": "oauth_token_secret",
        "displayName": "Token Secret",
        "description": "Used for generating signature for the request.",
        "shortDescription": "Token secret provided by server.",
        "hidden": false,
        "required": true
      },
      {
        "name": "realm",
        "displayName": "Realm",
        "description": "Used for identifying the account.",
        "shortDescription": "Account identifier.",
        "hidden": false,
        "required": false
      }
    ]
  }
]
```

OAuth Resource Owner Password Credentials

To support invoking of REST APIs secured using the OAuth ROPC grant, use the `OAUTH_RESOURCE_OWNER_PASSWORD_CREDENTIALS` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Extending the Security Policy
- Sample Code: OAuth Resource Owner Password Credentials
- Sample Code: OAuth Resource Owner Password Credentials, Extended

Overview

With the OAuth Resource Owner Password Credentials security policy, you make a REST call to get an access token, and you use the access token to call an API.

Since the resource owner shares its credentials with the client, this policy is used when there is a high degree of trust between the resource owner and the client. Oracle Integration ensures that credentials are stored securely in a vault.

The default implementation of this policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. Therefore, you extend the RFC 6749 compliant policy to override one or more steps. For example:

- To get the access token, you must provide the request to the authorization server in a specific format. RFC 6749 defines the format. However, different providers might require a different format. For instance, a server might require additional information beyond the authorization header; or a server might require the request to be part of a JSON body. Review the documentation for the authorization server to determine the required format.
- After you make a valid request for an access token, the authorization server returns a 200 response with some body. RFC 6749 specifies the format of the response. However, some authorization servers customize the response. For instance, the response might be XML instead of JSON. Often, the response is nested JSON that includes a lot of metadata and a nested token. You need to be able to find the access token and its expiry in the response. Review the documentation for the authorization server to understand the format of the response.

To add the OAuth Resource Owner Password Credentials policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
`oauth.client.id`	`Client Id`	`for example: 123-xxx.apps.googleusercontent.com`	`Used to identify the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.`	String	Yes

name	displayName	shortDescription	description	Data type	Required
oauth.client.secret	Client Secret	<unique random string matches your API console project>	Used to authorize the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.	Password	Yes
oauth.access.token.uri	Access Token URI	for example: https://accounts.google.com/o/oauth2/token	A request should be sent to this URI for obtaining an access token.	String	Yes
oauth.scope	Scope	for example: read,write.	Permissions your application is requesting on behalf of the user.	String	No
username	Username	Use UserName	A username credential	String	Yes
password	Password	Enter Password	A password credential	Password	Yes

Extending the Security Policy

If needed, extend the security policy.

A security policy defines the structure of an HTTP request. The default implementation of this security policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. If the application for which you're creating an adapter supports the standard security policy but requires additional information, you can extend the RFC 6749 compliant policy to override one or more steps. When you extend a security policy, you change the structure of the request.

1. Read your OAuth provider documentation and collect the following information.

Information to collect	Description

`AccessTokenRequest`	The authorization server expects the `AccessTokenRequest`. Determine the information that is needed and the method for sending the information to the authorization server.
	The following example is sample code for accessing a token request according to RFC 6749:
	
	`{`
	`"method": "POST",`
	`"uri": "<oauth.access.token.uri>",`
	`"params": {`
	`"template": {},`
	`"query": {`
	`}`
	`},`
	`"headers": {`
	`"Content-Type": "application/x-www-form-urlencoded",`
	`"Accept": "application/json",`
	`"Authorization" : "Basic " + ((.securityProperties.oauth.client.id)+\":\"+ (.securityProperties.oauth.client.secret") | @base64)`
	`},`
	`"body": "grant_type=password&client_id=&client_secret=&username=&password="`
	`}`

Response of the `AccessTokenRequest`	Determine how the authorization server returns a response to an access token request, as well as the information that is returned in the response. The type of response is different for each provider. Most providers return an access token response, as shown in the following example:
	
	`{`
	`"token_type": "Bearer",`
	`"access_token": "the access token value"`
	`}`

Information to collect	Description

`RefreshTokenRequest` that is expected by the authorization server	Determine the information that is needed and how the information must be sent to the authorization server.
	The following code snippet shows the access token usage according to RFC 6749:
	
	`{`
	`"method": "POST",`
	`"uri": "<oauth.access.token.uri>",`
	`"params": {`
	`"template": {},`
	`"query": {`
	`}`
	`},`
	`"headers": {`
	`"Content-Type": "application/x-www-form-urlencoded",`
	`"Authorization" : "Basic " + ((.securityProperties.oauth.client.id)+\":\"+ (.securityProperties.oauth.client.secret") | @base64)`
	`},`
	`"body": "grant_type=refresh_token&refresh_token=${refresh_token}`
	`}`

Response of the `RefreshTokenRequest`	Determine how the authorization server returns the response to a refresh token request and the information that is returned in the response.
	
	`{`
	`"token_type": "Bearer",`
	`"access_token": "the access token value"`
	`}`

The method by which the application expects the access token to be sent along with the request	Many applications expect additional information beyond the access token. `-H "Authorization: Bearer [access_token]"`

2. Extend the managed security policy as required by modifying one or more steps in the OAuth flow.

Sample Code: OAuth Resource Owner Password Credentials

In the example, the following security properties have default values and are hidden:

- `oauth.access.token.uri`: Default value is `https://login.eloqua.com/auth/oauth2/token`
- `oauth.scope`: Default value is `https://www.googleapis.com/auth/drive`
- `clientAuthentication`: Default value is `client_credentials_as_header`

- `oauth.request.content.type`: Default value is `application/x-www-form-urlencoded`

```
"securityPolicies": [
         {
             "type": "managed",
             "policy": "OAUTH_RESOURCE_OWNER_PASSWORD_CREDENTIALS",
             "description": "OAuth Resource Owner Password Credentials",
             "displayName": "OAuth Resource Owner Password Credentials",
             "scope": "ACTION",
             "securityProperties": [
                 {
                     "name": "oauth.client.id",
                     "displayName": "Client Id",
                     "description": "Client Id",
                     "shortDescription": "Client Id",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "oauth.client.secret",
                     "displayName": "Client Secret",
                     "description": "Client Secret",
                     "shortDescription": "Client Secret",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "oauth.access.token.uri",
                     "default": "https://login.eloqua.com/auth/oauth2/token",
                     "required": false,
                     "hidden": true
                 }, {
                     "name": "oauth.scope",
                     "default": "",
                     "required": false,
                     "hidden": false
                 }, {
                     "name": "username",
                     "displayName": "Username",
                     "description": "The resource owner user
name.",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "password",
                     "displayName": "Password",
                     "description": "The resource owner user
password.",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "oauth.request.content.type",
                     "default": "application/x-www-form-urlencoded",
                     "required": false,
                     "hidden": true
                 }, {
                     "name": "clientAuthentication",
                     "default": "client_credentials_as_header",
```

```
                    "required": false,
                    "hidden": true
                }
            ]
        }
    ],
    "test": "flow:TestConnectionFlow"
```

Sample Code: OAuth Resource Owner Password Credentials, Extended

The following code sample shows extended code authorization policy:

```
"securityPolicies": [
      {
        "type": "managed",
        "policy": "OAUTH_RESOURCE_OWNER_PASSWORD_CREDENTIALS",
        "description": "OAuth Resource Owner Password Credentials Policy",
        "displayName": "OAuth Resource Owner Password Credentials Policy",
        "scope": "ACTION",
        "securityProperties": [
                    {
                        "name": "oauth.client.id",
                        "displayName": "Client Id",
                        "description": "Client Id",
                        "shortDescription": "Client Id",
                        "required": true,
                        "hidden": false
                    }, {
                        "name": "oauth.client.secret",
                        "displayName": "Client Secret",
                        "description": "Client Secret",
                        "shortDescription": "Client Secret",
                        "required": true,
                        "hidden": false
                    }, {
                        "name": "oauth.access.token.uri",
                        "default": "https://login.ABCXYZ.com/auth/oauth2/
token",
                        "required": false,
                        "hidden": true
                    }, {
                        "name": "oauth.scope",
                        "default": "",
                        "required": false,
                        "hidden": false
                    }, {
                        "name": "username",
                        "displayName": "Username",
                        "description": "The resource owner user
name.",
                        "required": true,
                        "hidden": false
                    }, {
                        "name": "password",
                        "displayName": "Password",
                        "description": "The resource owner user
```

```
password.",
                        "required": true,
                        "hidden": false
                    }, {
                        "name": "oauth.request.content.type",
                        "default": "application/x-www-form-urlencoded",
                        "required": false,
                        "hidden": true
                    }, {
                        "name": "clientAuthentication",
                        "default": "client_credentials_as_header",
                        "required": false,
                        "hidden": true
                    }
            ],
            "authExtension": {
              "accessTokenRequest": {
                "method": "POST",
                "uri": "<authorization_uri>",
                "params": {
                  "template": {
                  },
                  "query": {
                    "client_id": "[your_client_id]",
                    "client_secret": "[your_client_secret]",
                    "grant_type": "password"
                  }
                },
                "headers": {
                  "Content-Type": "application/x-www-form-urlencoded",
                  "Authorization": "Basic " +
({.securityProperties.oauth.client.id}+\":\"+
{.securityProperties.oauth.client.secret"}) | @base64)
                },
                "body": "false"
              },
              "refreshTokenRequest": {
                "method": "POST",
                "uri": "<access_token_uri>",
                "params": {
                  "template": {
                  },
                  "query": {
                    "refresh_token": "${refresh_token}",
                    "client_id": "[your_client_id]",
                    "client_secret": "[your_client_secret]",
                    "grant_type": "refresh_token"
                  }
                },
                "headers": {
                  "Content-Type": "application/x-www-form-urlencoded"
                },
                "body": "false"
              },
              "fetchRules": {
                "access_token": "access.[tT]oken",
```

```
            "refresh_token": "refresh.[tT]oken",
            "expiry": "expires.*",
            "token_type": "token.?[tT]ype"
          },
          "accessTokenUsage": {
            "headers": {
              "Authorization": "Bearer : ${access_token}"
            }
          }
        }
      }
    ],
    "test": "flow:testConnectionFlow"
  }
```

OAuth Client Credentials

To support invoking of REST APIs secured with the OAuth-client-credentials grant, use the `OAUTH_CLIENT_CREDENTIALS` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Extending the Security Policy
- Sample Code: OAuth Client Credentials
- Sample Code: OAuth Client Credentials, Extended

Overview

The client application accesses resources from a third party application without using resource owner intervention.

The default implementation of this policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. Therefore, you extend the RFC 6749 compliant policy to override one or more steps.

To add the OAuth Client Credentials policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
oauth.client.id	Client Id	for example: 123-xxx.apps.googleusercontent.com	Used to identify the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.	String	Yes
oauth.client.secret	Client Secret	<unique random string matches your API console project>	Used to authorize the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.	Password	Yes
oauth.access.token.uri	Access Token URI	for example: https://accounts.google.com/o/oauth2/token	A request should be sent to this URI for obtaining an access token.	String	Yes
oauth.scope	Scope	for example: read,write.	Permissions your application is requesting on behalf of the user.	String	No

Extending the Security Policy

If needed, extend the security policy.

A security policy defines the structure of an HTTP request. The default implementation of this security policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. If the application for which you're creating an adapter supports the standard security policy but requires additional information, you can extend the RFC 6749 compliant policy to override one or more steps. When you extend a security policy, you change the structure of the request.

1. Read your OAuth provider documentation and collect the following information.

Information to collect	Description

`AccessTokenRequest`	The authorization server expects the AccessTokenRequest. Determine the information that is needed and the method for sending the information to the authorization server.
	The following example is sample code for accessing a token request according to RFC 6749:
	
	`{`
	`"method": "POST",`
	`"uri": "$`
	`{.securityProperties.oauth.access.token.uri}",`
	`"params": {`
	`"template": {},`
	`"query": {`
	`"client_id": "$`
	`{.securityProperties.oauth.client.id}",`
	`"redirect_uri": "${redirect_uri}",`
	`"client_secret": "$`
	`{.securityProperties.oauth.client.secret}",`
	`"grant_type": "authorization_code"`
	`}`
	`},`
	`"headers": {`
	`"Content-Type": "application/x-www-form-`
	`urlencoded",`
	`"Authorization" : "Basic " +`
	`((.securityProperties.oauth.client.id)+\":\"+`
	`(.securityProperties.oauth.client.secret") | @base64)`
	
	`},`
	`"body": "grant_type=client_credentials&scope=$`
	`{.securityProperties.oauth.scope}"`
	`}`

Response of the `AccessTokenRequest`	Determine how the authorization server returns the response to an access token request and the information that is returned in the response. The type of response is different for each provider. Most providers return an access token response, as shown in the following example:
	
	`{`
	`"token_type": "Bearer",`
	`"access_token": "the access token value"`
	`}`

The method by which the application expects the access token to be sent along with the request while calling the API	Many applications expect additional information beyond the access token. For example, most applications expect a valid access token to be sent as a header, as the following example illustrates with the request. The following code snippet shows the access token usage according to RFC 6749: `-H "Authorization: Bearer [access_token]"`

2. Extend the managed security policy as required by modifying one or more steps in the OAuth flow.

Sample Code: OAuth Client Credentials

The following sample code shows the implementation for the Client Credentials security policy.

```
"securityPolicies": [
         {
             "type": "managed",
             "policy": "OAUTH_CLIENT_CREDENTIALS",
             "description": "OAUTH2 CLIENT CREDENTIALS Policy",
             "displayName": "OAUTH2 CLIENT CREDENTIALS Policy",
             "scope": "ACTION",
             "securityProperties": [
                 {
                     "name": "oauth.client.id",
                     "displayName": "Client Id",
                     "description": "Client Id",
                     "shortDescription": "Client Id",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "oauth.client.secret",
                     "displayName": "Client Secret",
                     "description": "Client Secret",
                     "shortDescription": "Client Secret",
                     "required": true,
                     "hidden": false
                 }, {
                     "name": "oauth.access.token.uri",
                     "default": "https://login.abcxyz.com/01131-a123-4321-
a999-347dh4/oauth2/token",
                     "required": false,
                     "hidden": true
                 }, {
                     "name": "oauth.scope",
                     "default": "",
                     "required": false,
                     "hidden": true
                 }, {
                     "name": "oauth.request.content.type",
                     "default": "application/x-www-form-urlencoded",
                     "required": false,
                     "hidden": true
                 }, {
                     "name": "clientAuthentication",
                     "default": "client_credentials_as_header",
                     "required": false,
                     "hidden": true
                 }
             ]
         }
     ],
     "test": "flow:TestConnectionFlow"
 }
```

Sample Code: OAuth Client Credentials, Extended

The following code sample shows extended code authorization policy.

```
"securityPolicies": [
      {
        "type": "managed",
        "policy": "OAUTH_CLIENT_CREDENTIALS",
        "description": "OAuth Client Credentials Policy",
        "displayName": "OAuth Client Credentials Policy",
        "scope": "ACTION",
        "securityProperties": [
          {
            "name": "oauth.client.id",
            "displayName": "Client Id",
            "description": "Client Id",
            "shortDescription": "Client Id",
            "required": true,
            "hidden": false
          },
          {
            "name": "oauth.client.secret",
            "displayName": "Client Secret",
            "description": "Client Secret",
            "shortDescription": "Client Secret",
            "required": true,
            "hidden": false
          },
          {
            "name": "oauth.access.token.uri",
            "default": "https://accounts.abacxyz.com/o/oauth2/token",
            "required": false,
            "hidden": true
          },
          {
            "name": "oauth.scope",
            "default": "https://www.abcxyz.com/auth/videos",
            "required": false,
            "hidden": true
          },
          {
            "name": "clientAuthentication",
            "default": "client_credentials_in_header",
            "required": false,
            "hidden": true
          }
        ],
        "authExtension": {
          "accessTokenRequest": {
            "method": "POST",
            "uri": "https://www.googleapis.com/oauth2/v4/token",
            "params": {
              "template": {
              },
              "query": {
```

```
                "client_id": "[your_client_id]",
                "client_secret": "[your_client_secret]",
                "grant_type": "client_credentials"
              }
            },
            "headers": {
              "Content-Type": "application/x-www-form-urlencoded"
            },
            "body": "false"
          },
          "fetchRules": {
            "access_token": "access.[tT]oken",
            "expiry": "expires.*",
            "token_type": "token.?[tT]ype"
          },
          "accessTokenUsage": {
            "headers": {
              "Authorization": "Bearer : ${access_token}"
            }
          }
        }
      }
    ],
    "test": "flow:testConnectionFlow"
  }
```

OAuth Authorization Code Policy

To support invoking of REST APIs secured with the OAuth-code-authorization grant, use the `OAUTH_AUTHORIZATION_CODE_CREDENTIALS` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Extending the Security Policy
- Sample Code: OAuth Authorization Code
- Sample Code: OAuth Authorization Code, Extended

Overview

The Authorization Code security policy is an OAuth flow where the resource owner is required to provide consent before an access token can be granted to the client application. This policy has two sections: Basic properties and extensibility properties.

The default implementation of this policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. Therefore, you extend the RFC 6749 compliant policy to override one or more steps.

To add the OAuth Authorization Code policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required
oauth.client.id	Client Id	for example: 123-xxx.apps.myappusercontent.com	Used to identify the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.	String	Yes
oauth.client.secret	Client Secret	<unique random string matches your API console project>	Used to authorize the client(the software requesting an access token) that is making the request. The value passed in this parameter must exactly match the value shown in your API console project.	Password	Yes
oauth.auth.code.uri	Authorization Code URI	for example: https://accounts.myapp.com/o/oauth2/auth	This endpoint is the target of the initial request. It handles active session lookup, authenticating the user, and user consent.	String	Yes
oauth.access.token.uri	Access Token URI	for example: https://accounts.myapp.com/o/oauth2/token	A request should be sent to this URI for obtaining an access token.	String	Yes
oauth.scope	Scope	for example: read,write.	Permissions your application is requesting on behalf of the user.	String	No

Extending the Security Policy

If needed, extend the security policy.

A security policy defines the structure of an HTTP request. The default implementation of this security policy is RFC 6749. However, an implementation may vary from how the RFC illustrates. If the application for which you're creating an adapter supports the standard security policy but requires additional information, you can extend the RFC 6749 compliant policy to override one or more steps. When you extend a security policy, you change the structure of the request.

1. Read your OAuth provider documentation and collect the following information.

Information to collect	Description
AuthorizationURL	The authorization server expects the authorization URL. The following example is sample code for accessing a token request according to RFC 6749: <oauth.auth.code.uri>? client_id=[YOUR_CLIENT_ID]&response_type=code&redirect_ uri=[client's call back uri]&scope=[scope]

Information to collect	Description
The `AccessTokenRequest` that is expected by the authorization server.	Determine the information that is needed and how the information must be sent to the authorization server to obtain an access token. The content sent in the HTTP body depends on the content type defined in the HTTP header. Rapid Adapter Builder supports two content types: `"application/x-www-form-urlencoded"` and `"application/json"`. **Example 1:** With the content type as `"application/x-www-form-urlencoded"`. `{ "method": "POST", "uri": "<oauth.access.token.uri>", "headers": { "Content-Type": "application/x-www-form-urlencoded" }, "body": { "code": "${auth_code}", "client_id": "[your_client_id]", "redirect_uri": "${redirect_uri}", "client_secret": "[your_client_secret]", "grant_type": "authorization_code" } }` In this case, the body is sent as: `grant_type=authorization_code&code=tGzv3JOkF0XG5Qx2TlKWIA&client_id=s6BhdRkqt3&client_secret=7Fjfp0ZBr1KtDRbnfVdmIw&redirect_uri=https://your-app.com/oauth2/callback` **Example 2:** With the content type as `"application/json"`. `{ "method": "POST", "uri": "<oauth.access.token.uri>", "headers": { "Content-Type": "application/json" }, "body": { "code": "${auth_code}", "client_id": "[your_client_id]", "redirect_uri": "${redirect_uri}", "client_secret": "[your_client_secret]", "grant_type": "authorization_code" } }` In this case, the body is sent as: `{ "code": "tGzv3JOkF0XG5Qx2TlKWIA", "client_id": "s6BhdRkqt3", "redirect_uri": "https://your-app.com/oauth2/callback",`

Information to collect	Description

```
                "client_secret":
"7Fjfp0ZBr1KtDRbnfVdmIw",
                "grant_type": "authorization_code"
              }
```

Example 3: With jq expressions for the uri field and an additional header.

```
{
            "method": "POST",
            "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
            "headers": {
              "Content-Type": "application/x-www-form-
urlencoded",
              "Authorization" : "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") |
@base64)}"
            },
            "body": {
              "grant_type": "authorization_code",
              "code": "${auth_code}",
              "redirect_uri": "${redirect_uri}"
            }
          }
```

Example 4: Customized with a non-standard query parameter.

```
{
    "method": "POST",
    "uri": "<oauth.access.token.uri>",
    "params": {
        "template": {},
        "query": {
            "api-key": "${api-key}",
        }
    },
    "headers": {
        "Content-Type": "application/x-www-form-
urlencoded"
    },
    "body": {
            "code": "${auth_code}",
            "client_id": "[your_client_id]",
            "redirect_uri": "${redirect_uri}",
            "client_secret": "[your_client_secret]",
            "grant_type": "authorization_code"
        }
}
```

Information to collect	Description

Response of the `AccessTokenRequest`	The following code sample shows the response to the access token request:
	
	`{`
	`  "token_type": "Bearer",`
	`  "access_token": "the access token value",`
	`  "refresh_token": "the refresh token value"`
	`  "expires_in": "3600"`
	`}`

Information to collect	Description
`RefreshTokenRequest` that is expected by the authorization server	Determine the information that is needed and how the information must be sent to the authorization server to refresh an expired access token. The content sent in the HTTP body depends on the content type defined in the HTTP header. Rapid Adapter Builder supports two content types: `"application/x-www-form-urlencoded"` and `"application/json"`. **Example 1:** With the content type as `"application/x-www-form-urlencoded"`. `"refreshTokenRequest": { "method": "POST", "uri": "https://www.sampleapis.com/oauth2/v4/token", "headers": { "Content-Type": "application/x-www-form-urlencoded" }, "body": { "refresh_token": "${refresh_token}", "client_id": "${.securityProperties.\"oauth.client.id\"}", "client_secret": "${.securityProperties.\"oauth.client.secret\"}", "grant_type": "refresh_token" } }` In this case, the body is sent as: `grant_type=refresh_token&refresh_token=tGzv3JOkF0XG5Qx2TlKWIA&client_id=s6BhdRkqt3&client_secret=7Fjfp0ZBr1KtDRbnfVdmIw` **Example 2:** With the content type as `"application/json"`. `"refreshTokenRequest": { "method": "POST", "uri": "https://www.sampleapis.com/oauth2/v4/token", "headers": { "Content-Type": "application/json" }, "body": { "refresh_token": "${refresh_token}", "client_id": "${.securityProperties.\"oauth.client.id\"}", "client_secret": "${.securityProperties.\"oauth.client.secret\"}", "grant_type": "refresh_token" } }` In this case, the body is sent as: `{ "refresh_token": "tGzv3JOkF0XG5Qx2TlKWIA", "client_id": "s6BhdRkqt3",`

Information to collect	Description

```
                "client_secret":
"7Fjfp0ZBr1KtDRbnfVdmIw",
                "grant_type": "refresh_token"
              }
```

Example 3: With jq expressions for the uri field and an additional header.

```
{
            "method": "POST",
            "uri": "$
{.securityProperties.\"oauth.access.token.uri\"}",
            "headers": {
              "Content-Type": "application/x-www-form-
urlencoded",
              "Authorization" : "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") |
@base64)}"
            },
            "body": {
              "grant_type": "refresh_token",
              "refresh_token": "${refresh_token}"
            }
```

Example 4: Customized with non-standard query and body parameters.

```
{
    "method": "POST",
    "uri": "<oauth.access.token.uri>",
    "params": {
        "template": {},
        "query": {
            "api-key": "${api-key}",
        }
    },
    "headers": {
        "Content-Type": "application/x-www-form-
urlencoded"
    },
    "body":
   {
            "refresh_token": "${refresh_token}",
            "client_id": "[your_client_id]",
            "client_secret": "[your_client_secret]",
            "grant_type": "refresh_token",
            "account_id": "${account_id}"
        }
}
```

Information to collect	Description

Response of the `RefreshTokenRequest`	The following code sample shows the response to the refresh token request:
	
	`{`
	`"token_type": "Bearer",`
	`"access_token": "the access token value",`
	`"refresh_token": "the refresh token value"`
	`"expires_in": "3600"`
	`}`

Method by which the application expects the access token to be sent along with the request	Many applications expect additional information besides just the access token. `-H "Authorization": "Bearer ${access_token}"`

2. Extend the managed security policy as required by modifying one or more steps in the OAuth flow.

Sample Code: OAuth Authorization Code

In the example, the following security properties have default values and are hidden:

- `oauth.auth.code.uri`: Default value is `https://accounts.sample.com/o/oauth2/auth`
- `oauth.access.token.uri`: Default value is `https://www.sampleapis.com/oauth2/v4/token`
- `oauth.scope`: Default value is `https://www.sampleapis.com/auth/drive`
- `clientAuthentication`: Default value is `client_credentials_as_body`

```
"securityPolicies": [
      {
        "type": "managed",
        "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
        "description": "Sample Authorization Policy",
        "displayName": "Sample Authorization Policy",
        "scope": "ACTION",
        "securityProperties": [
          {
            "name": "oauth.client.id",
            "displayName": "Sample Client ID",
            "description": "Sample Client ID",
            "shortDescription": "Sample Client ID",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.client.secret",
            "displayName": "Sample Client Secret",
            "description": "Sample Client secret",
            "shortDescription": "Sample Client secret",
            "hidden": false,
            "required": true
```

```
    },
    {
      "name": "oauth.auth.code.uri",
      "default": "https://accounts.sample.com/o/oauth2/auth",
      "hidden": true,
      "required": true
    },
    {
      "name": "oauth.access.token.uri",
      "default": "https://www.sampleapis.com/oauth2/v4/token",
      "hidden": true,
      "required": true
    },
    {
      "name": "oauth.scope",
      "default": "https://www.sampleapis.com/auth/drive",
      "hidden": true,
      "required": true
    },
    {
      "name": "clientAuthentication",
      "default": "client_credentials_as_body",
      "hidden": true,
      "required": true
    }
  ]
```

Sample Code: OAuth Authorization Code, Extended

The following code sample shows extended code authorization policy.

```
"securityPolicies": [
      {
        "type": "managed",
        "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
        "description": "Sample Authorization Policy",
        "displayName": "Sample Authorization Policy",
        "scope": "ACTION",
        "securityProperties": [
          {
            "name": "oauth.client.id",
            "displayName": "Sample Client ID",
            "description": "Sample Client ID",
            "shortDescription": "Sample Client ID",
            "hidden": false,
            "required": true
          },
          {
            "name": "oauth.client.secret",
            "displayName": "Sample Client Secret",
            "description": "Sample Client secret",
            "shortDescription": "Sample Client secret",
            "hidden": false,
            "required": true
          },
          {
```

```
            "name": "oauth.auth.code.uri",
            "default": "https://accounts.sample.com/o/oauth2/auth",
            "hidden": true,
            "required": true
          },
          {
            "name": "oauth.access.token.uri",
            "default": "https://www.sampleapis.com/oauth2/v4/token",
            "hidden": true,
            "required": true
          },
          {
            "name": "oauth.scope",
            "default": "https://www.sampleapis.com/auth/drive",
            "hidden": true,
            "required": true
          },
          {
            "name": "clientAuthentication",
            "default": "client_credentials_as_body",
            "hidden": true,
            "required": true
          }
        ],
        "authExtension": {
          "authorizationURL": {
            "uri": "${.securityProperties.\"oauth.auth.code.uri\"}",
            "params": {
              "query": {
                "response_type": "code",
                "client_id": "${.securityProperties.\"oauth.client.id\"}",
                "redirect_uri": "${redirect_uri}",
                "scope": "${.securityProperties.\"oauth.scope\"}",
                "access_type": "offline"
              }
            }
          },
          "accessTokenRequest": {
            "method": "POST",
            "uri": "${.securityProperties.\"oauth.access.token.uri\"}",
            "headers": {
              "Content-Type": "application/x-www-form-urlencoded",
              "Authorization" : "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
            },
            "body": {
              "grant_type": "authorization_code",
              "code": "${auth_code}",
              "redirect_uri": "${redirect_uri}"
            }
          },
          "refreshTokenRequest": {
            "method": "POST",
            "uri": "${.securityProperties.\"oauth.access.token.uri\"}",
            "headers": {
```

```
              "Content-Type": "application/x-www-form-urlencoded",
              "Authorization" : "${\"Basic \" +
((.securityProperties.\"oauth.client.id\")+\":\"+
(.securityProperties.\"oauth.client.secret\") | @base64)}"
            },
            "body": {
              "grant_type": "refresh_token",
              "refresh_token": "${refresh_token}"
            }
          },
          "fetchRules": {
            "auth_code": "code",
            "access_token": "access.[tT]oken",
            "refresh_token": "refresh.[tT]oken",
            "expiry": "expires.*",
            "token_type": "token.?[tT]ype"
          },
          "accessTokenUsage": {
            "headers": {
              "Authorization": "Bearer ${access_token}"
            }
          }
        }
      }
    ]
```

OAuth JWT User Assertion

To support invoking of REST APIs that require JWT assertions of a user (on a client application), use the `JWT_USER_ASSERTION_FOR_OAUTH` managed security policy.

- Overview
- Security Properties
- Sample Code: OAuth JWT User Assertion (Google PubSub API)

Overview

An application may require users to authenticate using JWT user assertions, before it returns access tokens that allows access to its data/APIs. The OAuth JWT User Assertion security policy obtains an authorization to an application's API through a trusted relationship between the client (Oracle Integration) and authentication server, which involves an exchange of certificates.

When you add this security policy to your adapter, your adapter's user must create a client application on the corresponding authentication server and obtain a set of client credentials enabled with the correct grant type, that is, JWT assertion. During this process, the user may also need to exchange certificates to establish trust, both in the authentication server and Oracle Integration.

Subsequently, the adapter user can leverage the adapter interface to formulate and send a JWT user assertion from Oracle Integration to the authentication server, specifying the required privileges for a particular user (without the user having to explicitly provide consent, which is typically the case in the OAuth Authorization Code security policy). In exchange, the authentication server provides an access token that models the privileges for the specified user.

To add the OAuth JWT User Assertion policy to your adapter definition document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> If you choose to use this security policy in your adapter, ensure that you create a client application on the required authentication server, formulate the JWT assertion for the server, and thoroughly test the assertion on tools like jwt.io, Postman, or other browser plugins.

Security Properties

This section lists the security properties available in the OAuth JWT User Assertion authentication scheme template.

The user assertion is formulated using the values that are entered against these properties in the adapter interface by the adapter user. As an adapter developer, you can make the formulation of the assertion payload easier for your adapter's users by entering default values for the authentication server (corresponding to your adapter). This way, the adapter's users do not have to enter values for some properties, such as `audience`, `kid`, `jti`, `algorithm`, `expiration duration`, `scope`, and `access token URL`.

Typically, users only need to create a client application on the required authentication server and obtain values for `client_id`, `client_secret`, and the alias of the certificate uploaded.

If you need to include one or more custom properties to formulate the assertion, you can include them in the `connectionProperties` section. See Connection Properties and Sample Code.

> **Note:**
>
> To represent a particular user in an assertion, you require a property by the authentication server that represents the user. Typically, this is an Email ID or User ID.

Name	Display Name	Short Description	Description	Data Type	Required
`oauth.access.token.uri`	`Access Token URI`	`for example: https://www.google.com/oauth/token`	The URI to which a request should be sent for obtaining an access token.	String	Yes
`jwt_header`	`JWT Header`	`for example: {\"alg\":\"RSA256\"}`	The JWT header contains metadata about a JWT, including the key identifier, the algorithm used to sign in, and other information.	String (JSON String)	Yes
`jwt_payload`	`JWT Payload`	`for example: {"iss":"as12123", "sub":"nsjsas12"}`	The JWT payload contains statements about the entity (for example, the user) and additional entity attributes, which are called claims.	String (JSON String)	Yes

Name	Display Name	Short Description	Description	Data Type	Required
jwt_signing_key_alias_name	JWT Private Key Alias	for example: MSSignkey	The signing-key identifier received after uploading the private key in the Oracle Integration certificates.	String	Yes
scope	Scope	for example: read,write.	The permissions your application is requesting on behalf of the user.	String	No
jwtAccessTokenRequest	Customized Access Token Request	for example: -X POST -H 'Content-Type: application/x-www-form-urlencoded' -d 'grant_type=urn:ietf:params:oauth:grant-type:jwt-bearer&assertion=sjkdndaiwd&client_assertion_type=urn:ietf:params:oauth:client-assertion-type:jwt-bearer&client_assertion=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIs&scope=read' https://idcs-fb8d65590fb3413.identity.oraclecloud.com/oauth2/v1/token	The Access Token Request whose value is formed using a URI syntax of the HTTP request used to fetch the access token. The URI syntax resembles cURL.	String	No

Sample Code: OAuth JWT User Assertion (Google PubSub API)

In this example, Google's assertion requires custom properties (Service Account). Note the formulation of the header and the body using the JQ expression in the security policy definition. Some of the properties have been defaulted by the policy in the JQ expression itself.

```
"connectionProperties": [
  {
    "name": "projectID",
    "type": "STRING",
    "displayName": "Project ID",
    "description": "Google Cloud Project ID",
    "shortDescription": "Example: xxx-xxx-657890",
    "required": true,
    "hidden": false,
    "tokenized": true,
    "scope": [
      "ACTION",
```

```
        "TRIGGER"
      ]
    },
    {
      "name": "serviceAccount",
      "type": "STRING",
      "displayName": "Service Account",
      "description": "Google Cloud Service Account",
      "shortDescription": "Example: name@<projectId>.iam.gserviceaccount.com",
      "required": true,
      "hidden": false,
      "tokenized": true,
      "scope": [
        "TRIGGER",
        "ACTION"
      ]
    },
    {
      "name": "hostName",
      "type": "STRING",
      "displayName": "HostName",
      "description": "Environment hostname with which user needs to connect",
      "shortDescription": "Example: pubsub.googleapis.com",
      "required": true,
      "hidden": true,
      "scope": [
        "ACTION",
        "TRIGGER"
      ],
      "default": "pubsub.googleapis.com"
    }
  ],
  "securityPolicies": [
    {
      "type": "managed",
      "policy": "JWT_USER_ASSERTION_FOR_OAUTH",
      "description": "JWT User Assertion for OAuth Policy",
      "displayName": "JWT User Assertion for OAuth",
      "scope": "ACTION",
      "securityProperties": [
        {
          "name": "oauth.access.token.uri",
          "displayName": "Access token uri",
          "description": "Access token uri",
          "shortDescription": "https://oauth2.googleapis.com/token",
          "hidden": true,
          "required": true,
          "default": "https://oauth2.googleapis.com/token"
        },
        {
          "name": "jwt_header",
          "displayName": "JWT header",
          "description": "JWT header",
          "shortDescription": "{\"alg\":\"RSA256\"}",
          "hidden": true,
          "required": true,
```

```
        "default": "${{\"alg\" : \"RS256\",\"typ\": \"JWT\"}}"
      },
      {
        "name": "jwt_payload",
        "displayName": "JWT Payload",
        "description": "JWT Payload",
        "shortDescription": "{\"iss\":\"123as12\", \"sub\":\"as12123\"}",
        "hidden": true,
        "required": true,
        "default": "$
{{\"iss\": .connectionProperties.serviceAccount,\"sub\": .connectionProperties
.serviceAccount,\"aud\": \"https://oauth2.googleapis.com/token\",\"exp\":
\"\",\"iat\": \"\",\"scope\": \"https://www.googleapis.com/auth/pubsub\"}}"
      },
      {
        "name": "jwt_signing_key_alias_name",
        "displayName": "JWT Private Key Alias",
        "description": "Jwt Signing key. This is the key name of certificate
uploaded in the OIC certificates page.",
        "shortDescription": "Example: MSSignkey",
        "hidden": false,
        "required": true,
        "default": ""
      },
      {
        "name": "scope",
        "displayName": "Scope",
        "description": "Scope",
        "shortDescription": "Scope",
        "hidden": true,
        "required": false,
        "default": ""
      },
      {
        "name": "jwtAccessTokenRequest",
        "displayName": "Access Token Request",
        "description": "Access Token Request",
        "shortDescription": "Example: AC1234",
        "hidden": true,
        "required": false,
        "default": ""
      }
    ]
  }
]
```

OAuth JWT Client Assertion

To support invoking of REST APIs that require JWT assertions from client applications, use the `JWT_CLIENT_CREDENTIALS_ASSERTION_FOR_OAUTH` managed security policy.

- Overview
- Security Properties
- Sample Code: OAuth JWT Client Assertion (UK NHS MESH API)

Overview

An application may require clients to authenticate using JWT client assertions, before it returns access tokens that allows access to its data/APIs. The OAuth JWT Client Assertion security policy obtains an authorization to an application's API through a trusted relationship between the client (Oracle Integration) and authentication server, which involves an exchange of certificates.

When you add this security policy to your adapter, your adapter's user must create a client application on the corresponding authentication server and obtain a set of client credentials enabled with the correct grant type, that is, JWT assertion. During this process, the user may also need to exchange certificates to establish trust, both in the authentication server and Oracle Integration.

Subsequently, the adapter user can leverage the adapter interface to formulate and send a JWT assertion with client credentials from Oracle Integration to the authentication server, specifying the required privileges. In exchange, the authentication server provides an access token for the requested API. Similar to the OAuth Client Credentials security policy, this policy also does not require resource owner's intervention.

To add the OAuth JWT Client Assertion policy to your adapter definition document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> If you choose to use this security policy in your adapter, ensure that you create a client application on the required authentication server, formulate the JWT assertion for the server, and thoroughly test the assertion on tools like jwt.io, Postman, or other browser plugins.

Security Properties

This section lists the security properties available in the OAuth JWT Client Assertion authentication scheme template.

The client assertion is formulated using the values that are entered against these properties in the adapter interface by the adapter user. As an adapter developer, you can make the formulation of the assertion payload easier for your adapter's users by entering default values for the authentication server (corresponding to your adapter). This way, the adapter's users do not have to enter values for some properties, such as `audience`, `kid`, `jti`, `algorithm`, `expiration duration`, `scope`, and `access token URL`.

Typically, users only need to create a client application on the required authentication server and obtain values for `client_id`, `client_secret`, and the alias of the certificate uploaded.

If you need to include one or more custom properties to formulate the assertion, you can include them in the `connectionProperties` section. See Connection Properties and Sample Code.

Name	Display Name	Short Description	Description	Data Type	Required
oauth.access.token.uri	Access Token URI	for example: https://www.google.com/oauth/token	The URI to which a request should be sent for obtaining an access token.	String	Yes
jwt_header	JWT Header	for example: {\"alg\":\"RSA256\"}	The JWT header contains metadata about a JWT, including the key identifier, the algorithm used to sign in, and other information.	String (JSON String)	Yes
jwt_payload	JWT Payload	for example: {"iss":"as12123", "sub":"nsjsas12"}	The JWT payload contains statements about the entity (for example, the user) and additional entity attributes, which are called claims.	String (JSON String)	Yes
jwt_signing_key_alias_name	JWT Private Key Alias	for example: MSSignkey	The signing-key identifier received after uploading the private key in the Oracle Integration certificates.	String	Yes
scope	Scope	for example: read,write.	The permissions your application is requesting on behalf of the user.	String	No
jwtAccessTokenRequest	Customized Access Token Request	for example: -X POST -H 'Content-Type: application/x-www-form-urlencoded' -d 'grant_type=urn:ietf:params:oauth:grant-type:jwt-bearer&assertion=sjkdndaiwd&client_assertion_type=urn:ietf:params:oauth:client-assertion-type:jwt-bearer&client_assertion=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIs&scope=read' https://idcs-fb8d65590fb3413.identity.oraclecloud.com/oauth2/v1/token	The Access Token Request whose value is formed using a URI syntax of the HTTP request used to fetch the access token. The URI syntax resembles cURL.	String	No

Sample Code: OAuth JWT Client Assertion (UK NHS MESH API)

In this example, UK NHS's assertion requires custom properties (issuer). Note the formulation of the header and the body using the JQ expression in the security policy definition. Some of the properties have been defaulted by the policy in the JQ expression itself.

```
"connectionProperties": [
       {
           "name": "iss",
           "type": "STRING",
           "displayName": "ISSUER",
           "shortDescription": "The issuer claim that identifies the
principal that issued the JWT.",
           "required": true,
           "hidden": false,
           "scope": [
             "ACTION"
           ]UK
         },
         {
           "name": "kid",
           "type": "STRING",
           "displayName": "KID",
           "shortDescription": "Key Identifier that indicates what key was
used to sign the JWT.",
           "required": true,
           "hidden": false,
           "scope": [
             "ACTION"
           ]
         }
     ],
     "securityPolicies": [
       {
           "type": "managed",
           "policy": "JWT_CLIENT_CREDENTIALS_ASSERTION_FOR_OAUTH",
           "description": "Jwt Client Credentials Assertion For OAuth Policy",
           "displayName": "Jwt Client Credentials Assertion For OAuth",
           "scope": "ACTION",
           "securityProperties": [
             {
               "name": "oauth.access.token.uri",
               "displayName": "Access token uri",
               "description": "Access token uri",
               "shortDescription": "https://www.google.com/oauth/token",
               "hidden": true,
               "required": true,
               "default": "https://dev.api.service.nhs.uk/oauth2/token"
             },
             {
               "name": "jwt_header",
               "displayName": "JWT header",
               "description": "JWT header",
               "shortDescription": "{\"alg\":\"RSA256\"}",
               "hidden": true,
```

```
                "required": true,
                "default": "${{\"alg\" :
\"RS512\",\"kid\": .connectionProperties.kid}}"
              },
              {
                "name": "jwt_payload",
                "displayName": "JWT Payload",
                "description": "JWT Payload",
                "shortDescription": "{\"iss\":\"123as12\",
\"sub\":\"as12123\"}",
                "hidden": true,
                "required": true,
                "default": "$
{{\"iss\": .connectionProperties.iss,\"sub\": .connectionProperties.iss,\"aud\
": \"https://dev.api.service.nhs.uk/oauth2/token\",\"exp\":
1669859997,\"jti\": \"2c59e3b4-982a-44ea-94a2-3785e926df01\"}}"
              },
              {
                "name": "jwt_signing_key_alias_name",
                "displayName": "JWT Private Key Alias",
                "description": "Jwt Signing key. This is the key name of
certificate uploaded in the OIC certificates page.",
                "shortDescription": "Example: MSSignkey",
                "hidden": false,
                "required": true,
                "default": ""
              },
              {
                "name": "scope",
                "displayName": "Scope",
                "description": "Scope",
                "shortDescription": "Scope",
                "hidden": true,
                "required": false,
                "default": ""
              },
              {
                "name": "jwtAccessTokenRequest",
                "displayName": "Access Token Request",
                "description": "Access Token Request",
                "shortDescription": "Example: AC1234",
                "hidden": true,
                "required": false,
                "default": ""
              }
            ]
          }
      ]
```

Learn About Security Policies for Trigger Connections

The adapter definition document supports a number of managed inbound security policies for trigger connection definitions. These security policies protect your endpoints by validating incoming requests from external applications.

Managed Security Policies for Trigger Connections:

- Basic Authentication
- OAuth 2.0 Access Tokens
- HMAC Signature Validation
- RSA Signature Validation
- JWT Validation

Basic Authentication

To support invoking of Oracle Integration flows secured with Basic Authentication, use the `BASIC_AUTH` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: Basic Authentication

Overview

With HTTP basic authentication, the client sends HTTP requests with the Authorization header that contains the word **Basic** followed by a space and a Base64 encoded string that contains a user name and password in the following format: `username:password`.

To add the Basic Authentication policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Oracle uses OCI Identity and Access Management to authenticate requests for invoking integrations.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required	Recommendation
username	Username	Use UserName	A username credential	String	No	"hidden": true "fixed": "NA"
password	Password	Enter Password	A password credential	Password	No	"hidden": true "fixed": "NA"

Sample Code: Basic Authentication

```
"securityPolicies": [
    {
       "type": "managed",
       "policy": "BASIC_AUTH",
       "description": "Validate Basic Authentication token",
       "displayName": "Basic Authentication",
       "scope": "TRIGGER",
       "securityProperties": [
          {
            "name": "username",
            "hidden": true,
            "required": false,
            "default": "NA"
          },
          {
            "name": "password",
            "hidden": true,
            "required": false,
            "default": "NA"
          }
       ]
    }
 ]
```

OAuth 2.0 Access Tokens

To support invoking of Oracle Integration flows secured with OAuth 2.0 grant types, use the `OAUTH2.0_TOKEN_VALIDATION` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: OAuth 2.0 Access Tokens

Overview

With the OAuth 2.0 Access Tokens security policy, the client sends HTTP requests with the Authorization header that contains the word **Bearer**, followed by a OAuth 2.0 JWT token.

This policy validates the OAuth 2.0 JWT token from the Authorization header, validates its claims and signature, and asserts the user against Oracle Identity Cloud Service.

To add the OAuth 2.0 Access Tokens policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Oracle uses OCI Identity and Access Management to authenticate requests for invoking integrations.

Security Properties

The displayName for this security policy is `OAuth 2.0`. This security policy does not require properties.

For information about obtaining Oauth 2.0 tokens to access Oracle Integration, see Authenticate requests for invoking OIC Integration Flows.

Sample Code: OAuth 2.0 Access Tokens

```
"securityPolicies": [
     {
        "type": "managed",
        "policy": "OAUTH2.0_TOKEN_VALIDATION",
        "description": "Validates OAuth2.0 token",
        "displayName": "OAuth2.0",
        "scope": "TRIGGER"
      }
   ]
```

HMAC Signature Validation

To support validation of HMAC-based digital signatures of incoming requests for invoking Oracle Integration flows, use the `HMAC_SIGNATURE_VALIDATION` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: Digital Signature Validation (HMAC)

Overview

The HMAC policy validates the HMAC signature(s) part of incoming requests. The security policy validates signatures sent by an HTTP Client based on HMAC validation function (JQ) defined in the policy.

This security policy asserts the client ID value that is provided in `clientIdentifier` and checks that the clientID is associated with a ServiceInvoker role.

To add the HMAC Signature Validation policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> You can use the policy to verify multiple signatures in a single request.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required	Recommendation
signatureString	Signing String(s)/ Content(s) that are signed	Example: ${ .request .body}	JQ Expression or Flow indicating how build signatureString. In case of more than 1 signature output should be ArrayNode for all signing strings.	String	Yes	"hidden":true
signature	Request Signature Location(s)	Example: ${.request. headers.\" x-signature\"}	JQ Expression or Flow to extract signature(s) from request In case of more than 1 signature output should be ArrayNode for all signatures.	String	Yes	"hidden":true
secret	Shared Secret	secrets	Shared Secret (Password) If multiple signatures, This can either be comma separated or only one same password for all signatures.	Password	Yes	"hidden":false

name	displayName	shortDescription	description	Data type	Required	Recommendation
signatureAlgorithm	signature Algorithm	Signing Algorithm Example: HMACSHA256	One of the following: HMACSHA256 : HMAC Signature with SHA-256 HMACSHA384 : HMAC Signature with SHA-384 HMACSHA512 : HMAC Signature with SHA-512	CHOICE	Yes	"hidden":true
timestampValidator	Timestamp Validation Strategy	Example: ${.request.headers.ts < now()}	Optional, JQ Expression or Flow to validate message timestamp/ expiry	String	No	"hidden":true
clientIdentifier	This ID will be used to authorize after signature validation. ID must belong to valid client application in OIC IAM domain. It Must have ServiceInvoker Application Role assigned	Example: 741abdd2ca2ddddd055670cfa856bfv	This ID will be used to authorize after signature validation. ID must belong to valid client application in OIC IAM domain. It Must have ServiceInvoker Application Role assigned	String	Yes	"hidden":false

Sample Code: Digital Signature Validation (HMAC)

```
 "securityPolicies": [
     {
        "type": "managed",
        "policy": "HMAC_SIGNATURE_VALIDATION",
        "description": "Validates HMAC Signature",
        "displayName": "HMAC SIGNATURE VALIDATION",
        "scope": "TRIGGER",
        "securityProperties": [
           {
               "name": "signature",
               "hidden": true,
               "required": true,
               "default": "$
{connectivity::hexDecode(.request.headers.digest)}"
           },
           {
               "name": "signatureString",
               "displayName": "Request Signature Location",
               "hidden": true,
               "required": true,
               "default": "${.request.body}"
           },
           {
               "name": "signatureAlgorithm",
               "displayName": "Request Signature Location",
               "hidden": true,
               "required": true,
               "default": "HMACSHA256"
           },
           {
               "name": "secret",
               "displayName": "Shared Secret",
               "hidden": false,
               "required": true
           },
           {
               "name": "timestampValidator",
               "displayName": "Timestamp Validation",
               "hidden": true,
               "required": true,
               "default": ""
           },

        ]
     }
  ]
```

RSA Signature Validation

To support validation of RSA-based digital signatures of incoming requests for invoking Oracle Integration flows, use the `RSA_SIGNATURE_VALIDATION` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: Digital Signature Validation (RSA)

Overview

RSA signature verification validates the RSA signatures part of incoming requests. This policy validates signatures that are sent by an HTTP Client based on the HMAC validation function (JQ) that is defined in the policy.

This security policy asserts the client ID value that is provided in `clientIdentifier` and checks that the clientID is associated with a ServiceInvoker role.

To add the RSA Signature Validation policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

> **Note:**
>
> You can use this policy to verify multiple signatures in a single request.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	Value Description	Data type	Required	Recommendation
`signature`	`Actual Signature`	`Example: ${connectivity::base64Decode(.request.token)}`	JQ Expression or Flow to extract Signing String. In case of more than 1 signature output should be ArrayNode for all signatures.	String	Yes	`"hidden":true`
`signatureString`	`Signing String`	`Example: ${.request.body}`	JQ Expression or Flow to extract Signing String. In case of more than 1 signature output should be ArrayNode for all signing strings.	String	Yes	`"hidden":true`

name	displayName	shortDescription	Value Description	Data type	Required	Recommendation
signatureAlgorithm	Signing Algorithm	Example: SHA256withRSA	Text. Enter one of the following values: • SHA256withRSA: RSA Signature with SHA-256 • SHA384withRSA: RSA Signature with SHA-384 • SHA512withRSA: RSA Signature with SHA-512 • SHA256withRSASSA_PSS: RSASSA-PSS Signature with SHA-256 • SHA256withRSAandMGF1: RSASSA-PSS Signature with MGF1-SHA256 • SHA384withRSASSA_PSS: RSASSA-PSS Signature with SHA-384 • SHA384withRSAandMGF1: RSASSA-PSS Signature with MGF1-SHA384 • SHA384withRSASSA_PSS: RSASSA-PSS Signature with SHA-512 • SHA256withRSAandMGF1: RSASSA-PSS Signature with MGF1-SHA512	Choice	Yes	"hidden":true
signatureKey	Signature key alias.	Example: orakey	Enter one of the following values: **URL:** The HTTPS URL that returns the X.509 certificate. **Alias:** The alias of the Digital Signature certificate uploaded in Oracle Integration. For information on how to upload a Digital Signature certificate in Oracle Integration, see Upload a Certificate to Connect with External Services. **JQ/Flow:** JQ or flow that returns a JSON array of RSA certificate contents. If multiple signatures exist, you can either separate them by commas, or use one algorithm for all signatures.	String	Yes	N/A
timestampValidator	Timestamp Validation Strategy	Example: ${.request.headers.ts < now()}	Optional, JQ Expression or Flow to validate message timestamp or expiry.	String	No	"hidden":false

name	displayName	shortDescription	Value Description	Data type	Required	Recommendation
clientIdentifier	client Identifier	Example: 741abdd2ca2ddddd055670cfa856bf	This ID is used to authorize after signature validation. The ID must belong to a valid client application in Oracle Integration IAM domain. It must have the `ServiceInvoker` Application Role assigned.	String	Yes	`"hidden":false`

Sample Code: Digital Signature Validation (RSA)

```
"securityPolicies": [
     {
        "type": "managed",
        "policy": "RSA_SIGNATURE_VALIDATION",
        "description": "Validates RSA Signature",
        "displayName": "RSA SIGNATURE VALIDATION",
        "scope": "TRIGGER",
        "securityProperties": [
           {
               "name": "signatureString",
               "displayName": "Signature Statement",
               "hidden": true,
               "required": true,
               "default": "${.request.body)}"
           },
           {
               "name": "signature",
               "displayName": "Signature Statement",
               "hidden": true,
               "required": true,
               "default": "$
{connectivity::base64URLDecode(.request.query.signature)}"
           },
           {
               "name": "signatureAlgorithm",
               "displayName": "Request Signature Location",
               "hidden": true,
               "required": true,
               "default": "SHA256withRSA"
           },
           {
               "name": "signatureKey",
               "displayName": "Certificate Alias",
               "hidden": false,
               "required": true
           },
           {
               "name": "timestampValidator",
               "displayName": "Request Signature Location",
               "hidden": true,
               "required": true,
```

```
                "default": ""
            },

        ]
      }
  ]
```

JWT Validation

To support validation of the JWT present in the incoming requests for invoking Oracle Integration flows, use the `JWT_VALIDATION` managed security policy. You can customize the security policy as needed for a connection definition.

- Overview
- Security Properties
- Sample Code: JWT Validation

Overview

The policy validates JWT claims and signature, and asserts the user against IDCS. Many webhook publisher applications send a JWT token with signatures. Third-party providers issue the token.

Oracle uses OCI Identity and Access Management to authenticate requests for invoking integrations. The policy opens the incoming JWT token, extracts the JWT claim from the token, extracts the user from the JWT claim, and determines whether the user exists in OCI Identity and Access Management and has the ServiceInvoker role. If the user is valid and authorized, access is allowed.

To add the JWT Validation policy to your document, use the available authentication scheme template. See Implement a New Connection Definition.

Security Properties

A connection definition that uses this security policy defines the following properties in the `securityProperties` section. See Connection Properties and Sample Code.

The values in the `name`, `displayName`, `shortDescription`, and `description` columns list the default values that appear when you insert a security policy into an adapter definition document. You can update these values if needed.

name	displayName	shortDescription	description	Data type	Required	Recommendation
jwtToken	JWT Token	Example: ${.request.headers.authorization}	JQ Expression or Flow to extract Signing String.	String	Yes	"hidden":true

name	displayName	shortDescription	description	Data type	Required	Recommendation
signatureKey	JWK URL	Example: http::// host_port/ context/jwk	Signature key alias or shared secret or JWK location or JQ/flow that returns JSON array with first entry as RSA certificate contents. Text/JQ/ Flow	String	Yes	N/A
subjectClaim	Subject claim Override.	Example: user	Optional, Tex or JQ. Output is claim name. Default "sub".	String	No	"hidden":true
customClaimsValidation	Custom Claims Validation	Example: ${{\"scope\":\"write\"}}	Optional. JQ or flow to validate additional claims. Output is JsonNode with claim name and expected value.	String	No	"hidden":true

Sample Code: JWT Validation

The following sample code shows the configuration of JWT validation. Keep in mind the following points about this code sample:

- The security policy extracts the JWT token from the authorization header: `.request.headers.authorization|split(\" \")|.[1]`
- The security policy obtains the signature key from the the `signatureKey` property.

  This key resolve to the alias JWK URL of the JWT issuer, such as `"https://www.demosvc.com/oauth2/v3/certs"`
- The policy validates only the standard JWT claims and doesn't validate any custom claims.
- This policy uses the default subject claim, without any overrides.

```
{
 "connection": {
```

```
    "securityPolicies": [
       {
            "type": "managed",
            "policy": "JWT_VALIDATION",
            "scope": "TRIGGER",
            "securityProperties": [
                {
                    "name": "jwtToken",
                    "displayName": "JWT Token",
                    "hidden": true,
                    "required": true,
                    "default": "${.request.headers.authorization|split(\"
\")|.[1]}"
                },
                {
                    "name": "signatureKey",
                    "displayName": "JWK URL",
                    "hidden": true,
                    "required": true,
                    "default": "https://www.demosvc.com/oauth2/v3/certs"
                },
                 {
                    "name": "subjectClaim",
                    "displayName": "Subject claim Override",
                    "hidden": true,
                    "required": false,
                    "default": ""
                },
                {
                    "name": "customClaimsValidation",
                    "displayName": "Custom Claims Validation",
                    "hidden": true,
                    "required": false,
                    "default": ""
                }
            ]
        }
    ]
  }
}
```

12

Update Action Definitions

The VS Code extension for Rapid Adapter Builder pulls actions related information from the application's APIs when you convert the Postman collection into an adapter definition document. This information appears in the `actions` section of the adapter definition document. You can update this information as needed for your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Action Definitions Design Considerations to Implement an Action Identity of an Action Definition Input and Output Schemas of an Action Definition Configuring an Action Definition Runtime Implementation of an Action Definition
Review the properties to define and see sample code	Action Definition Properties and Sample Code
Implement an action definition	Implement an Action Definition Design Considerations to Implement an Action Design User Interface Components of an Adapter Design of Configuration Fields Dynamic Configuration of Options Combine Field and Value Dependencies

Learn about Action Definitions

Before updating the `actions` section of the adapter definition document, familiarize yourself with actions and their properties.

Topics:

- Introduction to Action Definitions
- Design Considerations to Implement an Action
- Identity of an Action Definition
- Input and Output Schemas of an Action Definition
- Configuring an Action Definition
- Runtime Implementation of an Action Definition

Introduction to Action Definitions

For business purposes, a company can chose to expose an application's APIs with some operations, such as creating or updating a record. When designing an integration, an

integration developer can program the integration to perform these operations by configuring one or more actions. Each action calls one or more application-specific APIs. Actions provide a user-centric encapsulation of the target application's operations.

The adapter definition document specifies all the actions that are available to integration developers. Each action has a reference. An action can also be a flow that implements this action. It can also contain the input schema and output schema definitions if the action has input and output data.

Many business scenarios have a requirement to initiate operations from the adapter to the external applications. For example, the adapter for a typical Customer Relationship Management (CRM) tool may be required to perform actions such as:

- Create, update, and delete customer data.
- Create an account.
- Create an opportunity.
- Log a ticket.

The action objects in an adapter definition document model the higher-order encapsulation of an adapters' operations on the external applications. The Rapid Adapter Builder platform supports:

- Invoking one or more APIs of the external application.
- Chaining API calls.

Additionally, the Rapid Adapter Builder platform provides the following capabilities that help integration developers design action definitions:

- Define the user facing identity and description of the action presented on the user interface.
- Define the design time experience for configuring an action definition with selection fields and dependencies.
- Define the request contract to the action definition exposed in the Oracle Integration mapper that may be different than the API contract.
- Process logic that maps the request contract exposed in the Oracle Integration mapper to one or more API invocations to the external applications.

To understand more about the properties of an action definition in the adapter definition document, see Action Definition Properties and Sample Code.

Design Considerations to Implement an Action

Since an action definition models the higher-order encapsulation of an adapter operation on an external application, the adapter developer must consider the following design requirements:

- Identity
  Based on the properties that describe the identity of the adapter.
- Input and output schemas
  Contract of the action
- Configuration
  The multiple options that must be made available to the user to configure an action.
- Runtime implementation
  The implementation logic for the action definition.

The next sections explore each of the design requirements in detail.

Identity of an Action Definition

The identity of an action definition is modeled by a number of properties that describe the identity of the adapter.

Property	Description
`id`	Unique identifier for the action definition within the adapter definition document.
`displayName`	Name that appears on the user interface of Oracle Integration.
`description`	Description of the action definition that appears on the user interface of Oracle Integration.
`group`	Categorization of the action definition that allows the adapter developer to easily organize and display it on the user interface of Oracle Integration.

Sample code for an action definition:

```
{
  "actions": {
    "insertRowAction": {
      "displayName": "Insert Row Into Sheet",
      "description": "This action insert a new row into sheet.",
      "execute": "flow:insertRowFlow",
      "input": "flow:sheetSchemaFlow",
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
      },
      "configuration": [
        {
          "name": "spreadsheetId",
          "displayName": "Spreadsheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:spreadsheetIdFlow",
          "required": true
        },
        {
          "name": "sheetId",
          "displayName": "Sheet Name",
          "description": "",
          "type": "COMBO_BOX",
          "options": "flow:sheetIdFlow",
          "required": true,
          "dependencies": {
            "spreadsheetId": {
              "values": []
            }
          }
        }
```

```
        ]
      }
    }
}
```

Input and Output Schemas of an Action Definition

The `input` and `output` properties model the contracts of an action definition. The property values can accept different type of values that determine the formulation of the schemas. The action definition's input and output properties can refer to a schema defined in the schema section of the adapter definition document. The action definition can also accept an inline schema expressed in JSON schema format. In the following code snippet, the input refers to a schema and the output references a flow. The `schemaType` property defines the type of schema.

```
"output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRowOutput"
        }
```

The Rapid Adapter Builder platform supports the following types of schemas:

- `application/schema+json`
  For JSON schemas, set the `schemaType` to `application/schema+json`.
- `avro/binary`
  For Avro schema, set the `schemaType` to `avro/binary`.

The following sections describe how to define a static schema, dynamic schema, and how to blend a static schema with a dynamic schema

Static Schema
You can build an adapter explicitly to support a particular customer's instance. In this business scenario, the adapter is used for internal consumption, and hence the adapter need not support dynamic behavior that is more suited for customization. For this requirement, the adapter developer can choose to define a JSON schema using name, data type, and other properties. The JSON schema is a static schema and is easy to define.

Some external applications and services may not provide the ability to define custom resources or objects, and the ability to extend existing resources or objects with user defined attributes. In this business scenario, the adapter developer can use the API schema that is defined and documented by the external application.

Sample code that describes the schema for the spreadsheet and sheet objects for Google Sheets service:

```
"schemas": {
  "Sheet": {
    "type": "object",
    "properties": {
      "properties": {
        "type": "object",
        "properties": {
          "sheetId": {
```

```
            "type": "integer"
          },
          "title": {
            "type": "string"
          },
          "index": {
            "type": "integer"
          },
          "sheetType": {
            "type": "string"
          }
        }
      }
    }
  },
  "Spreadsheet": {
    "type": "object",
    "properties": {
      "spreadsheetId": {
        "type": "string"
      },
      "spreadsheetUrl": {
        "type": "string"
      },
      "properties": {
        "type": "object",
        "properties": {
          "title": {
            "type": "integer"
          },
          "locale": {
            "type": "string"
          },
          "autoRecalc": {
            "type": "integer"
          },
          "timeZone": {
            "type": "string"
          }
        }
      },
      "sheets": {
        "type": "array",
        "items": {
          "$ref": "#/schemas/Sheet"
        }
      }
    }
  }
}
```

Dynamic Schema Generation
In this code sample, the `input` property is set to `flow:sheetSchemaFlow` value. This denotes that the schema is dynamically formulated by a flow.

The flow calls a metadata API provided by the external application. The flow then uses a jq expression and a JSON schema to give a response. The flow logic and jq expression differ based on the response of the metadata API. Some APIs return metadata that is close to JSON schema, and the jq expression required to modify the response to JSON schema is minimal.

For the above example, Google Sheet does not provide a metadata API. Instead, the adapter interprets the metadata as the value of the first row. The flow assumes that all columns are string types and reads the first row. Then the flow creates a JSON schema.

> **Note:**
>
> The Rapid Adapter Builder extension does not support the automatic creation of a trigger. However, the extension can import the schemas and flows that drive the trigger logic.

The following sample code shows the dynamic generation of the JSON schema based on the first row:

```
"sheetSchemaFlow": {
  "id": "sheetSchemaFlow",
  "description": "sheetSchemaFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "generalRestFunc",
      "type": "custom",
      "operation": "connectivity::rest"
    },
    {
      "name": "constructReturnObject",
      "type": "expression",
      "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
    }
  ],
  "states": [
    {
      "name": "startState",
      "type": "operation",
      "actions": [
        {
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://sheets.googleapis.com/v4/spreadsheets/
{spreadsheetId}/values/{range}",
              "method": "GET",
              "parameters": {
                "spreadsheetId": "${ .configuration.spreadsheetId }",
                "range": "${ .configuration.sheetId + \"!A1:Z1\" }",
                "majorDimension": "ROWS"
              }
```

```
            }
          },
          "actionDataFilter": {
            "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
            "toStateData": "${ .schema }"
          }
        },
        {
          "functionRef": "constructReturnObject",
          "actionDataFilter": {
            "toStateData": "${ .output }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

Blending Static Schema with Dynamic Schema

In some business scenarios where the external application supports extension fields for existing business objects, the adapter developer can combine a static schema with a dynamic schema. To accomplish this, do the following:

- Execute a flow that provides the JSON schema to the input or output.
- Design the static part of the JSON schema in the schema section.

In the flow, the adapter developer can use the following syntax to reference the schema:

```
"functions": [
  {
    "name": "getStaticSchema",
    "type": "expression",
    "operation": ".self.schemas.OrderEventSchema"
  },
```

The various steps involved in blending a static schema with a dynamic schema are:

- Declare a flow with a function that maps to the schema using the `.self.schemas` object.
- When this function is called within the flow, the state logic returns the schema in the form of a string. The schema is defined in the `schemas` section of the adapter definition document.
- To the static schema, append the schema information extended properties for an object. Use jq to perform this operation.

> **Note:**
>
> The adapter developer must design this considering the external application's API response.

Sample code that shows how to design a JSON schema for a Zuora order object with an extended field:

```
"OrderOutputSchemaFlow": {
    "id": "OrderOutputSchemaFlow",
    "version": "0.1",
    "start": "startState",
    "specVersion": "0.8",
    "functions": [
      {
        "name": "getStaticSchema",
        "type": "expression",
        "operation": ".self.schemas.OrderEventSchema"
      },
      {
        "name": "dynamicFlow1",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "dynamicFlow2",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "dynamicFlow3",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "dynamicFlow4",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "constructResult",
        "type": "expression",
        "operation": ".staticOutput"
      },
      {
        "name": "constructReturnObject",
        "type": "expression",
        "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
      }
    ],
    "states": [
      {
        "name": "startState",
        "type": "operation",
        "actions": [
          {
            "functionRef": "getStaticSchema",
            "actionDataFilter": {
              "toStateData": "${ .staticOutput }"
            }
```

```
          },
          {
            "functionRef": {
                "refName": "dynamicFlow1",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
Order\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow2",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
Account\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow3",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
PaymentMethod\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow4",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
Contact\"}",
```

```
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
            },
          {
            "functionRef": "constructResult",
            "actionDataFilter": {
              "toStateData": "${ .schema }"
            }
          },
          {
            "functionRef": "constructReturnObject",
            "actionDataFilter": {
              "toStateData": "${ .output }"
            }
          }
        ],
        "end": true
      }
    ]
}
```

Sample code:

```
{
  "flows": {
    "insertRecordInputFlow": {
      "id": "insertRecordInputFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generateSchema",
          "type": "expression",
          "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": {type:\"object\", properties:{firstName:
{type:\"string\"},lastName:{type:\"string\"},address:{type:\"string\"}}}}"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "generateSchema",
              "actionDataFilter": {
                "toStateData": "${ .output }"
              }
```

```
            }
          ],
          "end": true
        }
      ]
    }
  }
}
```

Configuring an Action Definition

In some business scenarios, the user may want to specify some values while configuring an action. For example, when inserting data into a spreadsheet, the user needs to specify which file is used.

To provide configuration options for an action definition, the Rapid Adapter Builder platform provides a configuration page where the user can perform configuration activities like:

- Selecting an input value.
- Typing an input value in a text box.
- Entering multiple values like entering values into a table.

The adapter can use these user input values for schema generation by the mapper in Oracle Integration or for runtime execution.

For more information on how to design a configuration for an action definition, see Design of Configuration Fields

The configuration page supports configuration activities like:

- Multiple user interface wizards.
- Dropdown selections.
- Text inputs.
- Dependency relationship on the user interface wizards.
  For example, the user can select a file only after a user selects a folder. For more details about field dependency, see Combine Field and Value Dependencies.

For each of the user interface widgets, there is a separate design construct. Some widgets require a configuration flow to return:

- A single value for a text box.
- An array of key/value pairs for a combo or list box.
- Rows of data for a table.

For more details about the user interface wizard, see Design User Interface Components of an Adapter.

The following sample code shows an action definition that has a configuration page:

```
{
  "actions": {
    "insertRecordConfigurationAction": {
      "displayName": "Insert Record Into Table(Configuration)",
      "execute": "flow:insertRecordActionFlow",
      "input": {
```

```
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRecordInputSchema"
        }
      },
      "output": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/insertRecordOutputSchema"
        }
      },
      "configuration": [
        {
          "name": "folderId",
          "displayName": "Folder Name",
          "description": "Folder Name",
          "type": "COMBO_BOX",
          "required": true,
          "options": [
            {
              "keyName": "1K5Qc444w-usPDtXUwP",
              "displayName": "Customers"
            },
            {
              "keyName": "1K5Qc444w-usPDwedesw",
              "displayName": "Orders"
            },
            {
              "keyName": "1K5Qc444w-usPsdewest",
              "displayName": "Contacts"
            }
          ]
        },
        {
          "name": "fileId",
          "displayName": "File Name",
          "description": "File Name",
          "type": "COMBO_BOX",
          "required": true,
          "options": "flow:dynamicOperationFlow",
          "dependencies": {
            "folderId": {
              "values": []
            }
          }
        }
      ]
    }
  }
}
```

Runtime Implementation of an Action Definition

The design of an action definition for design-time configuration can include all or some of the following:

- The identity that shows the action on the user interface so that the user can choose.
- The configuration fields that allow the scope of the action to be modified by the user on the user interface
- The input and output schemas.
  The input and output schemas may be dynamically decided.

The design-time configuration drives the runtime execution that the adapter developer can design and drive using the `execute` property and by referencing a flow definition. The flow definition accesses:

- The values selected in the configuration fields.
- Connectivity properties.
- The input data sent to the action.
  The input data must comply with the input schema.

For more information on how to author a flow, refer Introduction to Flow Definitions.

Sample code that shows how to invoke an external application Google Sheets API endpoint and execute using a flow definition:

```
"insertRowFlow": {
  "id": "insertRowFlow",
  "description": "insertRowFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "generalRestFunc",
      "type": "custom",
      "operation": "connectivity::rest"
    }
  ],
  "states": [
    {
      "name": "startState",
      "type": "operation",
      "actions": [
        {
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://sheets.googleapis.com/v4/spreadsheets/
{spreadsheetId}/values/{range}:append",
              "method": "POST",
              "parameters": {
                "spreadsheetId": "${ .configuration.spreadsheetId }",
                "range": "${ .configuration.sheetId + \"!A1:Z1\" }",
                "insertDataOption": "INSERT_ROWS",
```

```
                "valueInputOption": "RAW"
              },
              "body": "${ {range: (.configuration.sheetId + \"!A1:Z1\"),
majorDimension:\"ROWS\", values: .input | to_entries | [map(.value)] } }"
            }
          },
          "actionDataFilter": {
            "results": "${ .body }",
            "toStateData": "${ .output }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

Action Definition Properties and Sample Code

In the `actions` section of the adapter definition document, you define the actions that are available in your adapter.

- Properties
- Sample Code

Properties

Property	Description
`displayName`	Name of the action that appears as a selectable item in the Oracle Integration user interface.
`description`	Tooltip for the action in the Oracle Integration user interface.
`group`	Name of the group defined in the categories section of the adapter definition document.
`execute`	Refers to an action or a flow. The flow is the implementation for the action. Flows are defined in the flows section of the adapter definition document.

Property	Description

`input`	Schema used for the output message. It contains the `schemaType` and `schema` properties.
	The `schemaType` defines the type of the schema. Valid values are `application/json` and `avro/binary`. The `schema` parameter defines the structure of the input message. It can be an inline schema or a reference to a schema definition in the schemas section of the adapter definition document. The adapter developer can also reference the input to a flow in the flows section of the adapter definition document.
	Sample code 1:
	`"input": { "schemaType": "application/json", "schema": { "$ref": "#/schemas/staticInput" } },`
	Sample code 2:
	`"input": "flow:inputSchemaFlow"`

`output`	Schema used for the output message. It contains the `schemaType` and `schema` properties.
	The `schemaType` defines the type of the schema. Valid values are `application/json` and `avro/binary`. The `schema` parameter defines the structure of the output message. It can be an inline schema or a reference to a schema definition in the schemas section of the adapter definition document. The adapter developer can also reference the output to a flow in the flows section of the adapter definition document.
	Sample code 1:
	`"output": { "schemaType": "application/json", "schema": { "$ref": "#/schemas/staticOutput" } },`
	Sample code 2:
	`"output": "flow:outputSchemaFlow"`

Property	Description
`configuration`	Models the configuration option to fully configure the action. Certain actions may need additional information that the user needs to provide to allow runtime to function. The configuration options are modeled as user interface widgets, with data either statically defined or fed by flows. Selections of a configuration field can drive one or more configuration field selections, both in the values that are available or the set of configurations fields that are available. The configuration defines an extra page for configuring this action. The configuration consists of a set configuration fields (UI components), which will be rendered sequentially in the configuration page. Define a configuration field using the following required properties: • `name` The internal name of this configuration field, unique to the configuration section. • `displayName` The display name that appears in the Oracle Integration user interface. • `type`Type of user interface component. You can use the following valid user interfaces: – `COMBO_BOX` – `RADIO` – `TEXT_BOX` – `LIST_BOX` – `TEXT_AREA` – `BUTTON` – `CHECK_BOX` – `SHUTTLE_BOX` – `TABLE` – `LABEL` – `FILE` For more information, see Design User Interface Components of an Adapter You can also use the following optional configuration fields: • `description` Describes the field. This text appears in the Oracle Integration user interface as a tooltip. • `validation` References a flow that provides the validation result of the field. • `default` Defines the default value of the configuration field. • `required` Specifies whether this configuration selection is required. • `options` This property is only valid for `COMBO_BOX`, `RADIO`, and `LIST_BOX` user interface components. It consists of a set of key/value (keyName/displayName) pairs. • `columns` This property is only valid for the `TABLE` user interface component. It consists of a set of columns that are defined by the properties `name`, `displayName`, `type`, and `options`. • `dependencies` Defines the relationship of configuration fields.

Sample Code

```
"actions": {
   "staticInputOutputAction": {
     "displayName": "Simple Action With Static Input/Output",
     "execute": "flow:generalActionFlow",
     "input": {
       "schemaType": "application/json",
       "schema": {
         "$ref": "#/schemas/staticInput"
       }
     },
     "output": {
       "schemaType": "application/json",
       "schema": {
         "$ref": "#/schemas/staticOutput"
       }
     }
   }
}
```

Work with Action Definitions

Specify details about each action definition in the `actions` section of the adapter definition document.

What Do You Want to Do?

Task	More information
Implement an action definition in the adapter definition document	**Required:** Implement an Action Definition Design Considerations to Implement an Action See also: Action Definition Properties and Sample Code
Review other common modeling options for action definitions	**Optional:** • Design User Interface Components of an Adapter • Design of Configuration Fields • Combine Field and Value Dependencies • Dynamic Configuration of Options

Implement an Action Definition

The VS Code extension for Rapid Adapter Builder pulls action information from the Postman collection. Each API in the collection becomes an action in the adapter definition document. An action is an outbound call from Oracle Integration to the application that you're creating an adapter for.

The adapter developer can author the code for an action definition in the following ways:

- Use the Rapid Adapter Builder extension in Visual Studio Code to convert a Postman collection to the adapter definition document, and subsequently modify the actions section as per requirements.
- Manually author code from scratch for the action definition object in the adapter definition document based on the higher-order encapsulation of an adapter operation on the external application.

1. In Visual Studio Code, in the Explorer pane, expand **definitions**, and select the adapter definition document.

   The adapter definition document has an extension of `.add.json`, such as `adapter.add.json`.

   The adapter definition document appears in the workspace.
2. Expand the `actions` section.
3. Review and update as needed.
   - Action Definition Properties and Sample Code
   - Update Action Definitions

Design User Interface Components of an Adapter

This topic describes the configurable user interface components that the adapter developer can build on the adapter configuration page.

The Rapid Adapter Builder platform supports the following user interface components on the adapter configuration page:

- COMBO_BOX
- RADIO
- LIST_BOX
- TEXT_BOX
- TEXT_AREA
- BUTTON
- CHECK_BOX
- SHUTTLE_BOX
- TABLE
- FILE

The following sections show how to define the user interface components in flows and how to use them in the actions.

COMBO_BOX

A dropdown combo box user interface component that the adapter developer defines in a flow.

Sample code that shows how to implement the component in actions:

```
{
  "name": "comboBoxField",
  "displayName": "Combo Box",
  "description": "Combo Box Description",
```

```
  "type": "COMBO_BOX",
  "required": true,
  "options": [
    {
      "keyName": "ComboBoxOptionOne",
      "displayName": "Combo Box Option One"
    },
    {
      "keyName": "ComboBoxOptionTwo",
      "displayName": "Combo Box Option Two"
    },
    {
      "keyName": "ComboBoxOptionThree",
      "displayName": "Combo Box Option Three"
    }
  ]
}
```

Sample code that defines the component in flows:

```
{
  "name": "dynamicOperationField",
  "displayName": "Dynamic Operation",
  "description": "Dynamic Operation Description",
  "type": "COMBO_BOX",
  "options": "flow:dynamicOperationFlow",
  "required": true
}
```

Sample code for flow:

```
{
  "id": "dynamicOperationFlow",
  "description": "dynamicOperationFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "[{keyName:\"operationOne\", displayName:\"Dynamic
Operation One\"},{keyName:\"operationTwo\", displayName:\"Dynamic Operation
Two\"},{keyName:\"operationThree\", displayName:\"Dynamice Operation
Three\"}]"
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
```

```
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

RADIO

A selectable radio selection user interface component that the adapter developer defines in a flow.

Sample code that shows how to implement the component in actions:

```
{
  "name": "radioField",
  "displayName": "Radio Option",
  "description": "Radio Option Description",
  "type": "RADIO",
  "required": true,
  "options": [
    {
      "keyName": "RadioOptionOne",
      "displayName": "Radio Option One"
    },
    {
      "keyName": "RadioOptionTwo",
      "displayName": "Radio Option Two"
    },
    {
      "keyName": "RadioOptionThree",
      "displayName": "Radio Option Three"
    }
  ]
}
```

Sample code that defines the component in flows:

```
{
  "name": "radioField",
  "displayName": "Dynamic Operation",
  "description": "Dynamic Operation Description",
  "type": "RADIO",
  "options": "flow:dynamicOperationFlow",
  "required": true
}
```

Sample code for flow:

```
{
  "id": "dynamicOperationFlow",
  "description": "dynamicOperationFlow",
  "specVersion": "0.8",
```

```
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "[{keyName:\"operationOne\", displayName:\"Dynamic
Operation One\"},{keyName:\"operationTwo\", displayName:\"Dynamic Operation
Two\"},{keyName:\"operationThree\", displayName:\"Dynamice Operation
Three\"}]"
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

LIST_BOX

A selectable dropdown list user interface component that the adapter developer defines in a flow.

Sample code that shows how to implement the component in actions:

```
{
  "name": "listBoxField",
  "displayName": "List Box",
  "description": "List Box Description",
  "type": "LIST_BOX",
  "required": true,
  "defaultValue": "ListBoxOptionThree",
  "options": [
    {
      "keyName": "ListBoxOptionOne",
      "displayName": "List Box Option One"
    },
    {
      "keyName": "ListBoxOptionTwo",
      "displayName": "List Box Option Two"
    },
    {
      "keyName": "ListBoxOptionThree",
      "displayName": "List Box Option Three"
    }
```

```
  ]
}
```

Sample code that defines the component in flows:

```
{
  "name": "listBoxField",
  "displayName": "Dynamic Operation",
  "description": "Dynamic Operation Description",
  "type": "LIST_BOX",
  "options": "flow:dynamicOperationFlow",
  "required": true
}
```

Sample code for flow:

```
{
  "id": "dynamicOperationFlow",
  "description": "dynamicOperationFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "[{keyName:\"operationOne\", displayName:\"Dynamic
Operation One\"},{keyName:\"operationTwo\", displayName:\"Dynamic Operation
Two\"},{keyName:\"operationThree\", displayName:\"Dynamice Operation
Three\"}]"
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

TEXT_BOX

A text box user interface component that the adapter developer can refer to in a flow.

Sample code that shows how to implement the component in actions:

```
{
  "name": "textBoxField",
  "displayName": "Text Box",
  "description": "Text Box Description",
  "type": "TEXT_BOX",
  "default": "This Text Box Default Value",
  "required": true
}
```

Sample code that defines the component in flows:

```
{
  "name": "textBoxField",
  "displayName": "Text Box",
  "description": "Text Box Description",
  "type": "TEXT_BOX",
  "default": "flow:dynamicValueFlow",
  "required": true
}
```

Sample code for flow:

```
{
  "id": "dynamicValueFlow",
  "description": "dynamicValueFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "\"This value is returned from CNCF flow.\""
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

TEXT_AREA

A text area box user interface component that the adapter developer defines in a flow.

Sample code that shows how to implement the component in actions:

```
{
  "name": "textAreaField",
  "displayName": "Text Area",
  "description": "Text Area Description",
  "type": "TEXT_AREA",
  "default": "This Text Area Default Value",
  "required": true
}
```

Sample code that defines the component in flows:

```
{
  "name": "textBoxField",
  "displayName": "Text Box",
  "description": "Text Box Description",
  "type": "TEXT_BOX",
  "default": "flow:dynamicValueFlow",
  "required": true
}
```

Sample code for flows:

```
{
  "id": "dynamicValueFlow",
  "description": "dynamicValueFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "\"This value is returned from CNCF flow.\""
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
```

```
    }
  ]
}
```

BUTTON

A button user interface component that the adapter developer can define in actions.

Sample code that shows how to implement the component in actions:

```
{
  "name": "buttonField",
  "displayName": "Button",
  "description": "Button Description",
  "default": "Click Me",
  "type": "BUTTON",
  "required": true
}
```

CHECK_BOX

A checkbox user interface component that the adapter developer can define in actions.

Sample code that shows how to implement the component in actions:

```
{
  "name": "checkBoxField",
  "displayName": "Check Box",
  "description": "Check Box Description",
  "default": "true",
  "type": "CHECK_BOX",
  "required": false
}
```

SHUTTLE_BOX

A shuttle box user interface component that the adapter developer can refer to in a flow.

Sample code that shows how to implement the component in actions:

```
{
          "name": "shuttleBoxField",
          "displayName": "Shuttle Box",
          "description": "Shuttle Box Description",
          "type": "SHUTTLE_BOX",
          "required": true,
          "default": "[\"shuttleBoxOptionOne\", \"shuttleBoxOptionThree\"]",
          "options": [
            {
              "keyName": "shuttleBoxOptionOne",
              "displayName": "Shuttle Box Option One"
            },
            {
              "keyName": "shuttleBoxOptionTwo",
              "displayName": "Shuttle Box Option Two"
            },
```

```
            {
              "keyName": "shuttleBoxOptionThree",
              "displayName": "Shuttle Box Option Three"
            },
            {
              "keyName": "shuttleBoxOptionFour",
              "displayName": "Shuttle Box Option Four"
            }
          ]
        }
```

Sample code that defines the component in flows:

```
{
  "name": "radioField",
  "displayName": "Dynamic Operation",
  "description": "Dynamic Operation Description",
  "type": "RADIO",
  "options": "flow:dynamicOperationFlow",
  "required": true
}
```

Sample flow code:

```
{
  "id": "dynamicOperationFlow",
  "description": "dynamicOperationFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "constructResult",
      "type": "expression",
      "operation": "[{keyName:\"operationOne\", displayName:\"Dynamic
Operation One\"},{keyName:\"operationTwo\", displayName:\"Dynamic Operation
Two\"},{keyName:\"operationThree\", displayName:\"Dynamice Operation
Three\"}]"
    }
  ],
  "states":[
    {
      "name":"startState",
      "type":"operation",
      "actions":[
        {
          "functionRef": "constructResult",
          "actionDataFilter": {
            "toStateData": "${ .actionOutput }"
          }
        }
      ],
      "end": true
    }
```

```
  ]
}
```

TABLE

A table user interface component that the adapter developer defines in actions.

Sample code that shows how to implement the component in actions:

```
{
  "name": "tableField",
  "displayName": "Table",
  "description": "Table",
  "type": "TABLE",
  "required": true,
  "default": "[[\"param3\", \"string\"], [\"param4\", \"boolean\"]]",
  "columns": [
    {
      "name": "queryParamField",
      "displayName": "Name",
      "type": "TEXT_BOX"
    },
    {
      "name": "queryParamField",
      "displayName": "Data Type",
      "type": "COMBO_BOX",
      "options": [
        {
          "keyName": "string",
          "displayName": "string"
        },
        {
          "keyName": "boolean",
          "displayName": "boolean"
        },
        {
          "keyName": "integer",
          "displayName": "integer"
        },
        {
          "keyName": "number",
          "displayName": "number"
        }
      ]
    }
  ]
}
```

FILE

A drag and drop user interface component to attach a file. The adapter developer can define this component in actions.

Sample code that shows how to implement the component in actions:

```
{
          "name": "fileField",
          "displayName": "File Field",
          "description": "File Field Description",
          "type": "FILE",
          "required": false
        }
```

Design of Configuration Fields

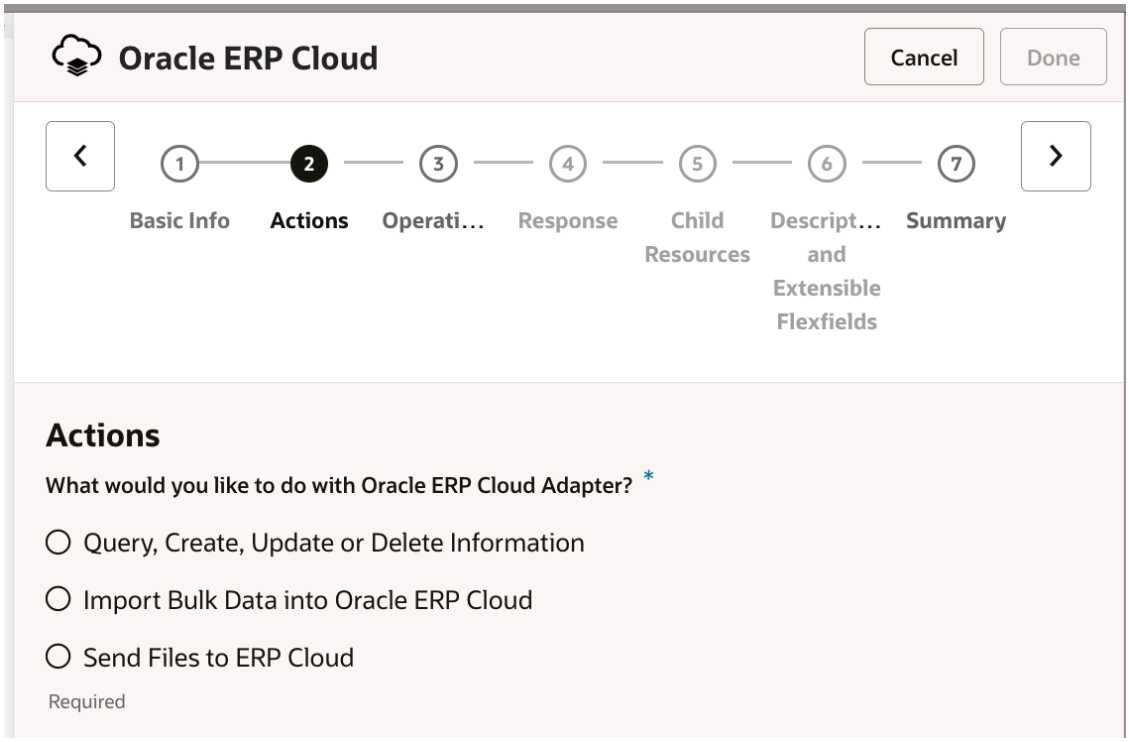
This section describes how to design configuration fields to configure an endpoint.

The configuration of an action may require the user to select multiple configuration fields. The Rapid Adapter Builder platform provides the flexibility to configure actions so that users:

- Can select configuration fields through selective options.
- Are presented with a set of valid choices to make.
- Do not make mistakes while configuring actions.
- Are presented with a set of values that are retrieved based on the user's input.

Once the user makes the required choices, the set configuration fields drive the runtime execution of the action.

For example, the adapters in Oracle Integration allow integration designers to configure how APIs call external applications. The following screenshot shows how the Oracle ERP Adapter in Oracle Integration presents the user with three choices representing various areas of fusion applications.



These configuration fields appear as user interface widgets in the Oracle Integration user interface. A configuration field has the following properties:

- `name`
- `displayName`
- `description`
- `type`
- `required`
- `options`
- `dependencies`

For more information about the properties, see Actions Properties and Sample Code.

The `type` property defines the type of user interface component to use for user selection. It includes user interface components like list box, text box, button, and so on. To learn more about how to use each user interface component, see Design User Interface Components of an Adapter.

Types of Configuration Dependencies

The types of adapter configurations are:

- No dependency
  A set of configuration fields that are independent of each other. Selecting a value from a configuration field doesn't impact or drive other configurations.

  > **Note:**
  >
  > The adapter developer need not specify the `dependencies` property if a configuration field has no dependency.

- Field dependency
  Selecting a value from a configuration field may impact and initiate other configurations.

  Example 1: In the Oracle ERP adapter, selecting the business objects displays a list of business objects in the list box.

  Example 2: A Google Sheet adapter provides an operation to insert a row into a spreadsheet. As the first configurable step, the user selects the spreadsheet. Since a spreadsheet can contain multiple worksheets, as the next configurable field, the adapter initiates the selection of a worksheet.

- Value dependency
  A configuration field drives the visibility and field values of one or more subsequent fields. For example, in the Oracle ERP adapter, when the user selects **Business (REST) Resources** from the **Browse By** dropdown, the field initiates/displays an additional dropdown option for **Service Application** where the user can provide the values like `crmRestApp` and `hcmRestApp`. However, if the user selects **Business Objects** from the **Browse By** field, the adapter doesn't display the **Service Application** field. In the example, only the value of **Business (REST) Resources** determine the display of the **Service Application** field.

Designing a Dependency

Declaring the dependency clause in the child or dependent field provides an indication to the child field about the parent field that it depends on. A child field can have only one parent that drives its functionality. Specifying an empty array in the `values` key signifies that any value of the parent can drive the functionality of the child.

Consider a business use case of a user selecting a worksheet within a spreadsheet. Displaying worksheets is not driven by specific values of the spreadsheet. Any value selected drives the worksheet field's visibility and field values. To configure an action, a spreadsheet selection is not dependent on the selection of a worksheet.

Sample code:

```
"dependencies": {
  "parentOption": {
    "values": []
  }
}
```

No Dependency

User selection of options may result in the action to call specific APIs. Consider a business scenario where a user has to select two independent static query parameters so that the API responds correctly and retrieves detailed object information. In this scenario, since the two static query parameters are independent of each other, the fields don't depend on each other. When there are no dependencies, user selection of configuration fields can happen in any order, and hence all the user interface widgets appear.

The following sample code shows two configuration fields that don't have any dependencies.

- `browse_by`
- `restServiceApplication`

The `browse_by` configuration field declares a static set of options for user selections and is designed as an array of objects. Each object has a `keyName` and `displayName`. The `keyName` is the internal identifier for the option, and the `displayName` is the name that appears on the user interface.

The `restServiceApplication` configuration field references a flow. The `COMBO_BOX` configuration field that appears on the user interface is derived from the execution of the flow.

Sample code:

```
[
      {
        "name": "browse_by",
        "displayName": "Browse By",
        "description": "Select the style of objects to configure.",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "businessObjects",
            "displayName": "Business Objects"
          },
```

```
        {
          "keyName": "businessServices",
          "displayName": "Business Services"
        },
        {
          "keyName": "restResources",
          "displayName": "Business (REST) Resources"
        }
      ]
    },
    {
      "name": "restServiceApplication",
      "displayName": "Service application",
      "description": "Please select the REST service application",
      "type": "COMBO_BOX",
      "required": true,
      "options": "flow:getListOfRESTServiceApplication"
    }
  ]
```

Field Dependency

Selecting a parent field that results in deriving the values of a dependent child field is known as field dependency.

Consider a business use case where a user needs to insert a row into a Google spreadsheet. As a first step, the user selects the required spreadsheet from a dropdown list. The Google Sheet API is invoked to get the list of spreadsheets and populate the dropdown list. As the next step, the user selects the required worksheet since a spreadsheet can contain multiple worksheets.

This business use case shows how the selection of a spreadsheet results in the next step of selecting the worksheet. Selecting the worksheet configuration field is dependent on the selection of the spreadsheet configuration field.

The adapter developer can define the configuration fields for an action in the adapter definition document and ensure that both the following objectives are achieved:

- Define the configuration field properties to display user interface widgets to enable the user to select.
- Retrieve the keys and their values from external applications.

Value Dependency

In value dependency, the dependency clause is declared in the child or dependent field. The dependency clause specifies details of the parent to the child. The `"values"` array in the dependency declaration contains a series of values. These values can be either statically defined or driven by a function. The child field of the parent is relevant to the configuration only when the selected parent matches any of the values listed in the dependency declaration. For example, in the Oracle ERP adapter, the **Service Application** option is relevant when the parent option selected is **Business (REST) Resources**. If the selected parent option is **Business Services** or **Business Objects**, then the **Service Application** option does not appear to the user for selection because it is not relevant.

Sample code: Static keys for values for demonstration

```
"dependencies": {
  "parentField": {
```

```
    "values": ["bar1", "bar2"]
  }
}
```

Business use case

The top level option is the `"Browse By"`. If the `"Browse By"` field selection matches `"restResources"`, then the `"Service application"` field appears to the user. Selecting the `"Service application"` field results in displaying the `"Rest Resources"` field. However, if the `"Browse By"` field selection matches `"businessServices"`, then the `"Service application"` field does not apply. Instead the backend logic for the `"Business Services"` field is executed and the resulting dropdown with the appropriate values is displayed to the user.

```
[
      {
        "name": "browse_by",
        "displayName": "Browse By",
        "description": "Select the style of objects to configure.",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "businessObjects",
            "displayName": "Business Objects"
          },
          {
            "keyName": "businessServices",
            "displayName": "Business Services"
          },
          {
            "keyName": "restResources",
            "displayName": "Business (REST) Resources"
          }
        ]
      },
      {
        "name": "restServiceApplication",
        "displayName": "Service application",
        "description": "Please select the REST service application",
        "type": "COMBO_BOX",
        "required": true,
        "options": "flow:getListOfRESTServiceApplication",
        "dependencies": {
          "browser_by": {
            "values": ["restResources"]
          }
        }
      },
      {
        "name": "restResources",
        "displayName": "REST resource",
        "description": "Please select the REST resource",
        "type": "COMBO_BOX",
        "required": true,
        "options": "flow:getListOfRESTResources",
```

```
      "dependencies": {
        "restServiceApplication": {
          "values": []
        }
      }
    },
    {
      "name": "businessServices",
      "displayName": "Business Services",
      "description": "Please select the Business Services",
      "type": "COMBO_BOX",
      "required": true,
      "options": "flow:getListOfRESTResources",
      "dependencies": {
        "restServiceApplication": {
          "values": ["businessServices"]
        }
      }
    }
  ]
```

Combine Field and Value Dependencies

To combine the field and value dependencies, the adapter developer can statically set the keys for readability.

However, based on prior selections, the adapter developer can also use a function to dynamically query and set the keys.

Sample code:

```
"configurationWithComplexUIAction": {
    "displayName": "Configuration with Complex UI",
    "description": "Configuration with Complex UI",
    "group": "advanced",
    "execute": "flow:generalActionFlow",
    "input": {
      "$ref": "#/schemas/staticInput"
    },
    "output": {
      "$ref": "#/schemas/staticOutput"
    },
    "configuration": [
      {
        "name": "rowOperation",
        "displayName": "Operation",
        "description": "",
        "type": "RADIO",
        "required": true,
        "options": [
          {
            "keyName": "CRUD",
            "displayName": "Query, Create, Update or Delete Information"
          },
          {
            "keyName": "importBulkData",
```

```
            "displayName": "Import Bulk Data into Oracle ERP Cloud"
          },
          {
            "keyName": "fileUploadWebCenterUCM",
            "displayName": "Send Files to ERP Cloud"
          }
        ]
      },
      {
        "name": "selectServiceBy",
        "displayName": "Browse by",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "browseByBusinessObject",
            "displayName": "Business Objects"
          },
          {
            "keyName": "browseByService",
            "displayName": "Services"
          },
          {
            "keyName": "browseByServiceRest",
            "displayName": "Business (REST) Resources"
          }
        ],
        "dependencies": {
          "rowOperation": {
            "values": ["CRUD"]
          }
        }
      },
      {
        "name": "bizObjList",
        "displayName": "Select a Business Object",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "{http://xmlns.oracle.com/apps/crmCommon/salesParties/
accountService/OSCAdapter}Account : AccountService",
            "displayName": "Account : AccountService"
          },
          {
            "keyName": "{http://xmlns.oracle.com/apps/crmCommon/salesParties/
addressService/OSCAdapter}Account : AddressService",
            "displayName": "Account : AddressService"
          },
          {
            "keyName": "{http://xmlns.oracle.com/apps/crmCommon/salesParties/
customerClassificationService/OSCAdapter}Account :
CustomerClassificationService",
            "displayName": "Account : CustomerClassificationService"
```

```
          },
          {
            "keyName": "{http://xmlns.oracle.com/apps/crmCommon/salesParties/
relationshipService/OSCAdapter}Account : RelationshipService",
            "displayName": "Account : RelationshipService"
          },
          {
            "keyName": "{http://xmlns.oracle.com/apps/crmCommon/activities/
activityManagementService/OSCAdapter}Activity",
            "displayName": "Activity"
          }
        ],
        "dependencies": {
          "selectServiceBy": {
            "values": ["browseByBusinessObject"]
          }
        }
      },
      {
        "name": "OPR_LIST_LABEL_FIELD_PARENT_PAGE",
        "displayName": "Select the Operation to Perform on the Business
Object",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "create",
            "displayName": "Create"
          },
          {
            "keyName": "delete",
            "displayName": "Delete"
          },
          {
            "keyName": "find",
            "displayName": "Find"
          },
          {
            "keyName": "get",
            "displayName": "Get"
          },
          {
            "keyName": "update",
            "displayName": "Update"
          }
        ],
        "dependencies": {
          "bizObjList": {
            "values": []
          }
        }
      },
      {
        "name": "bizServiceList",
        "displayName": "Select a Service",
```

```
          "description": "",
          "type": "COMBO_BOX",
          "required": true,
          "options": [
            {
              "keyName": "AccountService",
              "displayName": "AccountService"
            },
            {
              "keyName": "ActionPlanService",
              "displayName": "ActionPlanService"
            },
            {
              "keyName": "ActivityAppointmentService",
              "displayName": "ActivityAppointmentService"
            },
            {
              "keyName": "ActivityService",
              "displayName": "ActivityService"
            }
          ],
          "dependencies": {
            "selectServiceBy": {
              "values": ["browseByService"]
            }
          }
        },
        {
          "name": "OPR_LIST_LABEL_FIELD_PARENT_PAGE2",
          "displayName": "Select the Operation to Perform on the Business
Service",
          "description": "",
          "type": "COMBO_BOX",
          "required": true,
          "options": [
            {
              "keyName": "create",
              "displayName": "Create"
            },
            {
              "keyName": "delete",
              "displayName": "Delete"
            },
            {
              "keyName": "find",
              "displayName": "Find"
            },
            {
              "keyName": "get",
              "displayName": "Get"
            },
            {
              "keyName": "update",
              "displayName": "Update"
            }
          ],
```

```
    "dependencies": {
      "bizServiceList": {
        "values": []
      }
    }
  },

  {
    "name": "selectServiceApp",
    "displayName": "Select a Service Application",
    "description": "",
    "type": "COMBO_BOX",
    "required": true,
    "options": [
      {
        "keyName": "crmRestApp",
        "displayName": "crmRestApp"
      },
      {
        "keyName": "fscmRestApp",
        "displayName": "fscmRestApp"
      },
      {
        "keyName": "hcmRestApp",
        "displayName": "hcmRestApp"
      }
    ],
    "dependencies": {
      "selectServiceBy": {
        "values": ["browseByServiceRest"]
      }
    }
  },
  {
    "name": "restBizObjList",
    "displayName": "Select a Business Resource",
    "description": "",
    "type": "COMBO_BOX",
    "required": true,
    "options": [
      {
        "keyName": "Accounts",
        "displayName": "Accounts"
      },
      {
        "keyName": "ActionEvents",
        "displayName": "ActionEvents"
      },
      {
        "keyName": "Actionplans",
        "displayName": "Actionplans"
      },
      {
        "keyName": "Actions",
        "displayName": "Actions"
      }
```

```
        ],
        "dependencies": {
          "selectServiceApp": {
            "values": []
          }
        }
      },
      {
        "name": "OPR_LIST_LABEL_FIELD_PARENT_PAGE3",
        "displayName": "Select the operation to perform on the selected
resource",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "create",
            "displayName": "Create"
          },
          {
            "keyName": "delete",
            "displayName": "Delete"
          },
          {
            "keyName": "find",
            "displayName": "Find"
          },
          {
            "keyName": "get",
            "displayName": "Get"
          },
          {
            "keyName": "update",
            "displayName": "Update"
          }
        ],
        "dependencies": {
          "restBizObjList": {
            "values": []
          }
        }
      },
      {
        "name": "jobImportInterfaceId",
        "displayName": "Select Bulk Data Import Process",
        "description": "",
        "type": "COMBO_BOX",
        "required": true,
        "options": [
          {
            "keyName": "Post Mass Transfers",
            "displayName": "Post Mass Transfers"
          },
          {
            "keyName": "Post Mass Retirements",
            "displayName": "Post Mass Retirements"
```

```
            },
            {
              "keyName": "Post Mass Additions",
              "displayName": "Post Mass Additions"
            },
            {
              "keyName": "Asset Physical Inventory Comparison",
              "displayName": "Asset Physical Inventory Comparison"
            },
            {
              "keyName": "Import Asset Leases",
              "displayName": "Import Asset Leases"
            },
            {
              "keyName": "Upload Units of Production",
              "displayName": "Upload Units of Production"
            }
          ],
          "dependencies": {
            "rowOperation": {
              "values": ["importBulkData"]
            }
          }
        },
        {
          "name": "jobCheckBox",
          "displayName": "Reuse job property file uploaded separately in
respective UCM Account",
          "description": "",
          "type": "CHECK_BOX",
          "required": false,
          "dependencies": {
            "rowOperation": {
              "values": ["importBulkData"]
            }
          }
        },
        {
          "name": "ExtractFileType",
          "displayName": "Extract File",
          "description": "",
          "type": "COMBO_BOX",
          "required": true,
          "options": [
            {
              "keyName": "All",
              "displayName": "All"
            },
            {
              "keyName": "Error",
              "displayName": "Error"
            },
            {
              "keyName": "Log",
              "displayName": "Log"
            },
```

```
      {
        "keyName": "Out",
        "displayName": "Out"
      },
      {
        "keyName": "None",
        "displayName": "None"
      }
    ],
    "dependencies": {
      "rowOperation": {
        "values": ["importBulkData"]
      }
    }
  },
  {
    "name": "importJobOption",
    "displayName": "Additional Import Options",
    "description": "",
    "type": "TEXT_BOX",
    "required": true,
    "dependencies": {
      "rowOperation": {
        "values": ["importBulkData"]
      }
    }
  },
  {
    "name": "securityGroup",
    "displayName": "Security Group",
    "description": "",
    "type": "LIST_BOX",
    "required": true,
    "options": [
      {
        "keyName": "CRM",
        "displayName": "CRM"
      },
      {
        "keyName": "CRMStage",
        "displayName": "CRMStage"
      },
      {
        "keyName": "CSMImportExport",
        "displayName": "CSMImportExport"
      },
      {
        "keyName": "FAAuthPubContent",
        "displayName": "FAAuthPubContent"
      },
      {
        "keyName": "FAFusionImportExport",
        "displayName": "FAFusionImportExport"
      },
      {
        "keyName": "FolderAccess",
```

```
        "displayName": "FolderAccess"
      }
    ],
    "dependencies": {
      "rowOperation": {
        "values": ["fileUploadWebCenterUCM"]
      }
    }
  },
  {
    "name": "docAccount",
    "displayName": "Doc Account",
    "description": "",
    "type": "LIST_BOX",
    "required": true,
    "options": [
      {
        "keyName": "AUTHEN",
        "displayName": "AUTHEN"
      },
      {
        "keyName": "PEWebCenter/PU",
        "displayName": "PEWebCenter/PU"
      },
      {
        "keyName": "PUBLIC",
        "displayName": "PUBLIC"
      },
      {
        "keyName": "UCM_Spaces/PU",
        "displayName": "UCM_Spaces/PU"
      },
      {
        "keyName": "WCILS",
        "displayName": "WCILS"
      },
      {
        "keyName": "crm$/accessGroups$/import$",
        "displayName": "crm$/accessGroups$/import$"
      }
    ],
    "dependencies": {
      "rowOperation": {
        "values": ["fileUploadWebCenterUCM"]
      }
    }
  },
  {
    "name": "encryptFile",
    "displayName": "Encrypt the File",
    "description": "",
    "type": "CHECK_BOX",
    "required": false,
    "dependencies": {
      "rowOperation": {
        "values": ["fileUploadWebCenterUCM"]
```

```
            }
          }
        }
      ]
    }
}
```

Dynamic Configuration of Options

The configuration field can derive values from:

- Static values from an array of key pairs.
- A function defined by the `"execute"` field.
  The function derives the field values.

The following code example shows how the flow `"execute": "flow:spreadsheetIdFlow"` populates the values of the `"spreadsheetId"` configuration field.

Sample code:

```
"insertRowAction": {
  "description": "Insert Row Into Sheet",
  "summary": "This action inserts a new row into sheet.",
  "execute": "flow:insertRowFlow",
  "input": {
    "execute": "flow:sheetSchemaFlow"
  },
  "output": {
    "$ref": "#/schemas/insertRowOutput"
  },
  "configuration": [
    {
      "name": "spreadsheetId",
      "displayName": "Spreadsheet Name",
      "description": "Please select a spreadsheet to insert a row",
      "type": "COMBO_BOX",
      "options": "flow:spreadsheetIdFlow",
      "required": true
    },
    {
      "name": "workSheetId",
      "displayName": "Worksheet Name",
      "description": "Please select a worksheet to insert a row",
      "type": "COMBO_BOX",
      "options": "flow:worksheetIdFlow",
      "required": true,
      "dependencies": {
        "spreadsheetId": {
          "values": []
        }
      }
    }
  ]
}
```

Configuration field flows can dynamically derive the values for the options `keyName` and `displayName` by reading the data from external applications by invoking APIs. The following code example shows how a dynamic call to Google Sheets retrieves a set of spreadsheets that appear for user selection.

Sample code:

```
"flows":
{
   "spreadsheetIdFlow": {
     "id": "spreadsheetIdFlow",
     "description": "spreadsheetIdFlow",
     "specVersion": "0.8",
     "version": "0.1",
     "start": "startState",
     "functions": [
       {
         "name": "generalRestFunc",
         "type": "custom",
         "operation": "connectivity::rest"
       }
     ],
     "states": [
       {
         "name": "startState",
         "type": "operation",
         "actions": [
           {
             "functionRef": {
               "refName": "generalRestFunc",
               "arguments": {
                 "uri": "https://www.googleapis.com/drive/v3/files",
                 "method": "GET",
                 "parameters": {
                   "q": "mimeType='application/vnd.google-apps.spreadsheet'
and trashed=false",
                   "pageSize": 1000
                 }
               }
             },
             "actionDataFilter": {
               "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
               "toStateData": "${ .output }"
             }
           }
         ],
         "end": true
       }
     ]
   },
   "worksheetIdFlow": {
     "id": "sheetIdFlow",
     "description": "sheetIdFlow",
     "specVersion": "0.8",
     "version": "0.1",
```

```
    "start": "startState",
    "functions": [
      {
        "name": "generalRestFunc",
        "type": "custom",
        "operation": "connectivity::rest"
      }
    ],
    "states": [
      {
        "name": "startState",
        "type": "operation",
        "actions": [
          {
            "functionRef": {
              "refName": "generalRestFunc",
              "arguments": {
                "uri": "https://sheets.googleapis.com/v4/spreadsheets/
{spreadsheetId}",
                "method": "GET",
                "parameters": {
                  "spreadsheetId": "${ .configuration.spreadsheetId }"
                }
              }
            },
            "actionDataFilter": {
              "results": "${ .body.sheets | map({keyName:.properties.title,
displayName:.properties.title}) }",
              "toStateData": "${ .output }"
            }
          }
        ],
        "end": true
      }
    ]
  }
}
```

The `".configuration"` property structure allows selection of values from the configuration fields. For example, the `worksheetflowID` flow accesses the user selected `"spreadsheetId"` value from the `".configuration.spreadsheetId"` list of values.

The `"actionDataFilter"` property extracts data from the `body.sheets` JSON object of the API response, and then formats the data into key value map of `keyName` and `displayName`.

Create a Configuration Field Flow

You can author this type of flow in the following ways:

- Manually author the relevant JSON code in the adapter definition document based on an understanding of the Rapid Adapter Builder semantics and syntax.
- Use the Rapid Adapter Builder extension in Visual Studio Code to author the flow.
- Model a flow in a Postman collection and then use the Rapid Adapter Builder extension in Visual Studio Code to import the collection and then convert it into adapter definition document.

Use Postman Conversion of Flows to Design Configuration Fields

Configuration fields often derive their values by querying data from APIs of external applications and services. For example, an action that can create a resource may need to query the API of an external service and retrieve a set of available resources. The query results may also include custom objects.

In this scenario, the adapter developer can do the following:

- Test the API to get the list of available resources and save the example in the Postman collection.
- Filter the response to contain only the required fields.

  Typically, for a specific configuration field, the retrieved data is a set of names and values corresponding to the set of the resources as defined external application or service.
- Populate the configuration field.
- The adapter developer can use the Rapid Adapter Builder extension for Visual Studio Code to import only the flow corresponding to the request.

Configuration fields use flows to substitute values that are needed with the variables. The input data is imported as hardcoded values and can derive their values from other configuration field selection. If the format of the data from the API response does not comply with the requirement of the configuration field, the adapter developer can use jq expressions to restructure the data into the required format.

> **Note:**
>
> The format of the required data differs based on the widget used. For text boxes, the required data is a single object with a name and a value property. For a table widget, the required data to initialize the configuration field requires an array of rows of data. For more information, see Design User Interface Components of an Adapter.

The following sample code shows a flow that:

- Retrieves the collection of spreadsheets for a particular Google Sheet user, formats the results into an array of name and values, and displays in a combo dropdown.
- Retrieves the collection of sheets within a spreadsheet, formats the results into an array of name and values, and displays in a combo dropdown.

```
"spreadsheetIdFlow": {
  "id": "spreadsheetIdFlow",
  "description": "spreadsheetIdFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "generalRestFunc",
      "type": "custom",
      "operation": "connectivity::rest"
    }
  ],
```

```
  "states": [
    {
      "name": "startState",
      "type": "operation",
      "actions": [
        {
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://www.googleapis.com/drive/v3/files",
              "method": "GET",
              "parameters": {
                "q": "mimeType='application/vnd.google-apps.spreadsheet' and
trashed=false",
                "pageSize": 1000
              }
            }
          },
          "actionDataFilter": {
            "results": "${ .body.files | map({keyName:.id,
displayName:.name}) }",
            "toStateData": "${ .output }"
          }
        }
      ],
      "end": true
    }
  ]
},
"sheetIdFlow": {
  "id": "sheetIdFlow",
  "description": "sheetIdFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "generalRestFunc",
      "type": "custom",
      "operation": "connectivity::rest"
    }
  ],
  "states": [
    {
      "name": "startState",
      "type": "operation",
      "actions": [
        {
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://sheets.googleapis.com/v4/spreadsheets/
{spreadsheetId}",
              "method": "GET",
              "parameters": {
                "spreadsheetId": "${ .configuration.spreadsheetId }"
```

```
              }
            }
          },
          "actionDataFilter": {
            "results": "${ .body.sheets | map({keyName:.properties.title,
displayName:.properties.title}) }",
            "toStateData": "${ .output }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

The following sample code retrieves the list of topics, formats the results into an array of name and values, and displays in a combo dropdown.

```
"ListTopicsUIFlow": {
  "id": "ListTopicsUIFlow",
  "version": "0.1",
  "start": "startState",
  "specVersion": "0.8",
  "functions": [
    {
      "name": "generalRestFunc",
      "operation": "connectivity::rest",
      "type": "custom"
    }
  ],
  "states": [
    {
      "actions": [
        {
          "name": "listTopics",
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "${\"https://\"+.connectionProperties.hostName+\"/v1/
projects/{projectID}/topics\"}",
              "method": "GET",
              "parameters": {
                "projectID": "${.connectionProperties.projectID}",
                "pageSize": 1000
              }
            }
          },
          "actionDataFilter": {
            "results": "${ .body|if .topics!=null then(.topics | map({keyName:
(.name|split(\"/\")|.[3]), displayName:(.name|split(\"/\")|.[3])})) else
map({keyName:\"\", displayName:\"\"}) end}",
            "toStateData": "${ .output }"
          }
        }
      ],
```

```
      "name": "startState",
      "type": "operation",
      "end": true
    }
  ]
}
```

Use Postman Conversion of Flows to Model Dynamic Schema

Typically, schemas are statically defined for the input and output of an action. However, an enterprise adapter may need to support the ability to call the external application or service to query a particular metadata model of an object or a resource. The API response contains:

- A list of properties.
- The property name and datatype of each property.

> **Note:**
>
> This format may vary from service to service.

However, the configuration field requires an array of objects in name and value format. The response for the flow that can support dynamic schema requires the response of the flow to produce metadata corresponding to the supported schema type that defines the structure.

The Rapid Adapter Builder platform supports the following formats:

- `application/schema+json`

  For JSON schema, set the `schemaType` to `application/schema+json` value.

- `avro/binary`

  For arvo schema, set the `schemaType` to `avro/binary` value.

If the API response returns a format of the object definiton that is neither of the above formats, the adapter developer can use jq expressions to transform and format the data into a JSON schema form.

The following sample code shows how to query the first row of a sheet in Google Sheet by using the heading as metadata.

```
"sheetSchemaFlow": {
  "id": "sheetSchemaFlow",
  "description": "sheetSchemaFlow",
  "specVersion": "0.8",
  "version": "0.1",
  "start": "startState",
  "functions": [
    {
      "name": "generalRestFunc",
      "type": "custom",
      "operation": "connectivity::rest"
    },
    {
      "name": "constructReturnObject",
```

```
      "type": "expression",
      "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
    }
  ],
  "states": [
    {
      "name": "startState",
      "type": "operation",
      "actions": [
        {
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://sheets.googleapis.com/v4/spreadsheets/
{spreadsheetId}/values/{range}",
              "method": "GET",
              "parameters": {
                "spreadsheetId": "${ .configuration.spreadsheetId }",
                "range": "${ .configuration.sheetId + \"!A1:Z1\" }",
                "majorDimension": "ROWS"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ .body.values[0] | map({key:., value:
{type:\"string\"}}) | from_entries | {type: \"object\", properties: .} }",
            "toStateData": "${ .schema }"
          }
        },
        {
          "functionRef": "constructReturnObject",
          "actionDataFilter": {
            "toStateData": "${ .output }"
          }
        }
      ],
      "end": true
    }
  ]
}
```

The following sample code shows a flow that combines a static schema with a custom metadata to create a unified JSON schema.

```
"RetrieveAPaymentMethodCustomFieldFlow": {
    "id": "RetrieveAPaymentMethodCustomFieldFlow",
    "version": "0.1",
    "start": "startState",
    "specVersion": "0.8",
    "functions": [
        {
            "name": "staticFlow",
            "type": "expression",
            "operation": ".self.schemas.RetrieveAPaymentMethod0ResponseSchema"
```

```
        },
        {
            "name": "dynamicFlow",
            "operation": "connectivity::rest",
            "type": "custom"
        },
        {
            "name": "constructResult",
            "type": "expression",
            "operation": ".staticOutput"
        },
        {
            "name": "constructReturnObject",
            "type": "expression",
            "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
        }
    ],
    "states": [
        {
            "name": "startState",
            "type": "operation",
            "actions": [
                {
                    "functionRef": "staticFlow",
                    "actionDataFilter": {
                        "toStateData": "${ .staticOutput }"
                    }
                },
                {
                    "functionRef": {
                        "refName": "dynamicFlow",
                        "arguments": {
                            "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
PaymentMethod\"}",
                            "method": "GET"
                        }
                    },
                    "actionDataFilter": {
                        "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                        "toStateData": "$
{ .staticOutput.properties.body.properties }"
                    }
                },
                {
                    "functionRef": "constructResult",
                    "actionDataFilter": {
                        "toStateData": "${ .schema }"
                    }
                },
                {
                    "functionRef": "constructReturnObject",
                    "actionDataFilter": {
                        "toStateData": "${ .output }"
```

```
                        }
                    }
                ],
                "end": true
            }
        ]
    }
```

13

Update Trigger Definitions

The VS Code extension for Rapid Adapter Builder pulls triggers related information from the application's APIs when you convert the Postman collection into an adapter definition document. This information appears in the `triggers` section of the adapter definition document. You can update this information as needed for your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Learn About Trigger Definitions
Implement a trigger	Work with Trigger Definitions
Implement a security policy in a trigger	Work with Security Policies in Triggers

Learn About Trigger Definitions

Familiarize yourself with triggers and their properties.

What Do You Want to Do?

Task	More information
Understand various aspects involved with designing a trigger	**Everyone completes this task:** • Introduction to Trigger Definitions • Understand Event Subscription • Understand Trigger Security • Understand Configuration Scenarios • Runtime Implementation of Triggers
Review the recommended workflow to define a trigger	**Everyone completes this task:** • Workflow to Implement a Trigger
Review the properties to define and see sample code	**Everyone completes this task:** • Trigger Properties and Sample Code

Introduction to Trigger Definitions

While building your adapter, you can specify the trigger model that the adapter supports and define other trigger-related information in the `triggers` section of the adapter definition document.

Types of Triggers

A trigger is a mechanism that initiates an integration flow when a specific event occurs in an external system. While designing an adapter for Oracle Integration, you can model two types of triggers:

- **Webhook trigger:** In this trigger type, external applications automatically send new data to Oracle Integration when a specific event occurs. See What Is a Webhook Trigger.

- **Polling trigger:** In this trigger type, Oracle Integration actively checks for changes in an external application. See What Is a Polling Trigger.

Overview of Designing a Trigger

Using the adapter definition document, you can design a trigger object that can receive or retrieve events occurring in external applications and services. A trigger definition contains several components beyond just event handling. Here are all the components of a trigger definition:

Component	Description
Identity	The identity of the adapter. It contains properties such as display name, description, and group. See Trigger Properties and Sample Code.
Trigger contract	The message that is received (or retrieved) from an external application as a trigger. See What Is a Webhook Trigger and What Is a Polling Trigger.
Subscription registration	The component that allows the modeling of the webhook registration. See Understand Event Subscription. **Note:** This applies only to webhook triggers.
Security	The security schemes that validate if the events are received (or retrieved) from the right application and according to the security policy defined in the connection. See Understand Trigger Security.
Configuration	The component that allows the modeling of trigger configuration, whenever necessary. In design time, the integration developer can configure the trigger using the options available or mechanism defined. See Understand Configuration Scenarios.
Runtime implementation	The component that transforms the incoming data to a human-readable format (mapper-friendly format). See Runtime Implementation of Triggers. **Note:** This applies only to webhook triggers.

What Is a Webhook Trigger

With webhook triggers, external applications automatically send new data to Oracle Integration using webhooks, whenever a specific event occurs in the application. When the new data is received, the integration flow runs and processes the data.

A webhook trigger consists of the following two contracts. The Rapid Adapter Builder supports designing of both these contracts and mapping of one form into the other.

- Webhook Contract
- Mapper Contract

Webhook Contract

A webhook contract is the raw inbound message sent by an external application (likely through an HTTP post) that has headers, query parameters, and body components.

To handle a webhook:

- Set the `type` property to `webhook`.
- Set the `httpMethod` property to `POST` or as per your requirement.
- In the `webhook` property, model the headers, query parameters, and the body of the webhook request.
- In the `request` property, reference the JSON schema that models the structure of the message, which appears in the integration mapper.

Note that, for a webhook, it might not be necessary to model the headers or query parameters. Additionally, in most cases, you can present the entire webhook body to the integration developer, and need not transform the raw, inbound webhook request into a different form. Therefore, the request schema and the body schema can generally be the same as shown in the following example.

```
"OrderProcessedTrigger": {
    "displayName": "Order Processed Event",
    "description": "This event gets triggered when order has been processed.",
    "group": "Orders",
    "type": "webhook",
    "httpMethod": "POST",
    "webhook": {
        "header" : {
            "type": "object",
            "properties": {
                "name" : "x-custom-header",
                "type": "string"
            }
        },
        "body": {
            "schemaType": "application/schema+json",
            "schema": {
                "$ref": "#/schemas/OrderEventSchema"
            }
        }
    }
    "request" : {
        "schemaType": "application/schema+json",
        "schema": {
            "$ref": "#/schemas/OrderEventSchema"
        }
    }
}
```

Mapper Contract

A mapper contract is the trigger request (modeled in the `request` property) that appears in the Oracle Integration mapper.

If you want to transform the raw, inbound webhook request into a different form, you can use the `execute` property to process the webhook and transform the raw request into the desired schema (specified in the `request` property).

Additionally, both the webhook body and request properties can be driven by flows and derive schemas dynamically.

In the following example, the `request` property references a flow, which creates a dynamic schema based on a call to the external application's metadata API. This schema is displayed in the Oracle Integration mapper. The `execute` property also references a flow to transform the webhook message to the final message that is presented to the mapper.

```
 "OrderProcessedTrigger": {
            "displayName": "Order Processed Event",
            "description": "This event gets triggered when order has been
processed.",
            "group": "Orders",
            "type": "webhook",
            "httpMethod": "POST",
            "contentType": "application/cloudevents+json",
            "request": "flow:createRefinedrderEventSchema",
            "webhook": {
                "body": {
                    "schemaType": "application/schema+json",
                    "schema": {
                        "$ref": "#/schemas/OrderEventSchema"
                    }
                }
            }
            "execute" : "flow:processOrderWebhookMsgToRetrieveOrderObjectFlow"
        }
```

Here is an example of a `request` flow that demonstrates the creation of the `request` schema based on a static schema that's enriched with the custom field extension from a metadata API call to an external application.

```
"createRefinedrderEventSchema": {
    "id": "createRefinedrderEventSchema",
    "version": "0.1",
    "start": "startState",
    "specVersion": "0.8",
    "functions": [
      {
        "name": "getStaticSchema",
        "type": "expression",
        "operation": ".self.schemas.OrderEventSchema"
      },
      {
        "name": "dynamicFlow1",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "dynamicFlow2",
        "operation": "connectivity::rest",
        "type": "custom"
```

```
      },
      {
        "name": "dynamicFlow3",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "dynamicFlow4",
        "operation": "connectivity::rest",
        "type": "custom"
      },
      {
        "name": "constructResult",
        "type": "expression",
        "operation": ".staticOutput"
      },
      {
        "name": "constructReturnObject",
        "type": "expression",
        "operation": "{\"schemaType\": \"application/schema+json\",
\"schema\": .schema}"
      }
    ],
    "states": [
      {
        "name": "startState",
        "type": "operation",
        "actions": [
          {
            "functionRef": "getStaticSchema",
            "actionDataFilter": {
              "toStateData": "${ .staticOutput }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow1",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
Order\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow2",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
```

```
Account\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow3",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
PaymentMethod\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
          },
          {
            "functionRef": {
                "refName": "dynamicFlow4",
                "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName+\"/settings/custom-fields/zuora/
Contact\"}",
                    "method": "GET"
                }
            },
            "actionDataFilter": {
                "results": "${ .body | .schema.properties |
with_entries(select(.value.type != null)) | map_values({type: .type})}",
                "toStateData": "${ .staticOutput.properties }"
            }
            },
          {
            "functionRef": "constructResult",
            "actionDataFilter": {
              "toStateData": "${ .schema }"
            }
          },
          {
            "functionRef": "constructReturnObject",
            "actionDataFilter": {
              "toStateData": "${ .output }"
            }
          }
        ],
        "end": true
```

```
        }
      ]
}
```

What Is a Polling Trigger

You can use polling triggers if an external application doesn't have an eventing system. In such cases, the external application provides an API call that allows you to retrieve data.

With polling triggers, Oracle Integration periodically queries an external application for new data using an API call. If there is new data, it is retrieved and processed by the integration flow.

You can employ different schemes to poll for new data. Some examples are listed here:

- Creation date: Query for records created on a certain date.
- Creation time: Query for records created within a certain time interval.
- Sequence number: Query for records using their unique ID in the system.

> **Note:**
>
> In this trigger type, Oracle Integration actively queries for data, compares the new data to avoid duplication, and then processes it. For the purpose of comparison, Oracle Integration stores data of the previous query. The stored data is typically known as *bag of properties*. For details on how to use these properties, see the functions `connectivity::getPersistentState` and `connectivity::setPersistentState` in Supported Custom CNCF Serverless Workflow.

What's Supported

Oracle Integration supports the following polling features in a polling trigger:

- Singleton polling on cloud (single poller thread per activation)
- Acknowledgment or success polling handling
- Agent polling
- State persistence across polling events
- Non-high-availability state persistence across polling events on agent

Definition for Polling Triggers

The following properties are available in the `triggers` section of an adapter definition document for a polling trigger:

- The `type` property: Specify the trigger type. Set it to `polling`.
- The `execute` property: Specify the polling method and target. It is executed on each poll.
- The `polling` object with the following properties; this is specific to polling triggers:
  - `onInit`: Specify the initialization flow. It's executed once on activation.
  - `onSuccess`: Specify the flow to be executed on successful publish of polling data on mcube. It's executed once per poll.

– `pollFrequencyInSeconds`: Specify the frequency of polling. The default value is ten seconds.

> **Note:**
>
> - No changes are required in the `connection` section. The details you specify to connect to the target application are used for the polling trigger as well.
> - For security configuration, polices that you define in the trigger role are applied to polling calls. The non-http protocol calls use the non-manged policy schema. You must implement a non-managed policy in the protocol handler.

Sample Code

```
"pollingTypeTrigger": {
    "displayName": "Polling Type Trigger",
    "description": "This trigger Polls.",
    "type": "polling",
    "execute":"flow:execute",
    "request": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/staticInput"
      }
    },
    "polling": {
      "onInit":"flow:init",
      "onSuccess":"flow:ack",
      "pollFrequencyInSeconds": 3
    }
  },
  "pollingTypeTriggerParallel": {
    "displayName": "Polling Type Trigger parallel",
    "description": "This trigger Polls.",
    "type": "polling",
    "execute":"flow:execute",
    "request": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/staticInput"
      }
    },
    "polling": {
      "onInit":"flow:init",
      "onSuccess":"flow:ack",
      "pollFrequencyInSeconds": 3
    }
  },
  "pollingTypeTriggerCloudParallel": {
    "displayName": "Polling Type Trigger parallel",
    "description": "This trigger Polls.",
    "type": "polling",
    "execute":"flow:execute",
```

```
    "request": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/staticInput"
      }
    },
    "polling": {
      "onInit":"flow:init",
      "onSuccess":"flow:ack",
      "pollFrequencyInSeconds": 3
    }
  }
```

Understand Event Subscription

In case of webhook triggers, to receive events from an external application, you must subscribe to those events.

The webhook configuration process involves subscribing to events and specifying the destination URI where the external application should send the event data. You can create a subscription for events in either of the following ways:

- **Manual:** Create a webhook subscription through the administration page of the external application.
- **Programmatic:** Execute an API call to the external application.

As part of the subscription set up, the external application may issue a ping to ensure that the endpoint can be reached. It may also require an explicit acknowledgment of the ping. See Validation During Webhook Registration.

In a similar way, you can unsubscribe to events when necessary.

In Oracle Integration, the registration of a webhook corresponds to the activation of an integration flow, which indicates that Oracle Integration is ready to receive events. A deregistration corresponds to the deactivation of an integration flow, which indicates that the integration flow is not receiving events.

The adapter definition document of the Rapid Adapter Builder supports both the `Register` and `Deregister` constructs. As shown in the following example, both these constructs take a flow. The purpose of each flow is to execute one or more API calls to programmatically subscribe to the webhook on the external service.

```
"subscription": {
  "register": "flow:CreateSubscriptionFlow",
  "deregister": "flow:DeleteSubscriptionFlow"
},
```

The Flow to Register and Deregister a Subscription

Here is an example flow for the `Register` and `Deregister` constructs.

The `Register` logic in the flow performs the following actions:

- Lists the subscriptions for the logged-in account by executing an API call to the external application.
- Determines if a subscription already exists by parsing the results and executing a jq expression. This is modeled as a CNCF function.

- If a subscription exists, it updates the subscription through an API call.
- If a subscription does not exist, it creates a new subscription through an API call.

The `Deregister` logic in the flow just calls the API to delete the subscription, but it does not check if a subscription exists (to avoid additional API calls).

> **Note:**
>
> In the following example code, the Oracle Integration endpoint (represented as `.integrationProperties.INTEGRATION_FLOW_URL`) helps find the details of the subscription registration.

```
"CreateSubscriptionFlow": {
  "id": "CreateSubscriptionFlow",
  "description": "CreateSubscriptionFlow",
  "version": "0.1",
  "start": "startState",
  "specVersion": "0.8",
  "functions": [
    {
      "name": "ResolveSubscriptionName",
      "operation": "(.integrationProperties.INTEGRATION_FLOW_URL|
split(\"/\") | .[-3]) as $intName |
(.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\") | .[-2]) as $flow
| (.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\") | .[2]) as$host
| (if (.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\") | .[-5])|
test(\"project\") then $intName+\"_\"+\"project_\"+$flow+\"_\"+$host
else $intName+\"_\"+$flow+\"_\"+$host end )",
      "type": "expression"
    },
    {
      "name": "generalRestFunc",
      "operation": "connectivity::rest",
      "type": "custom"
    },
    {
      "name": "TransformSubName",
      "operation":
"[.integrationProperties.INTEGRATION_FLOW_URL==.subscriptionList.body.subscrip
tions[].pushConfig.pushEndpoint]|any",
      "type": "expression"
    }
  ],
  "states": [
    {
      "actions": [
        {
          "functionRef": "ResolveSubscriptionName",
          "actionDataFilter": {
            "toStateData": "${ .subscriptionName }"
          }
        },
        {
```

```
          "name": "ListSubscriptionFunction",
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "${\"https://\"+.connectionProperties.hostName+\"/v1/
projects/\"+.connectionProperties.projectID+\"/subscriptions\"}",
              "method": "GET",
              "parameters": {
                "pageSize": 1000
              }
            }
          },
          "actionDataFilter": {
            "results": "${{ body: .body, headers: .headers }}",
            "toStateData": "${ .subscriptionList }"
          }
        },
        {
          "functionRef": "TransformSubName",
          "actionDataFilter": {
            "toStateData": "${ .subscriptionExists }"
          }
        },
        {
          "name": "UpdateSubscriptionFunction",
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "${\"https://\"+.connectionProperties.hostName+\"/v1/
projects/\"+.connectionProperties.projectID+\"/subscriptions/
\"+.subscriptionName}",
              "method": "PATCH",
              "body": "${{subscription:
{ackDeadlineSeconds: .configuration.ackDeadline,pushConfig:{oidcToken:
{serviceAccountEmail: .connectionProperties.serviceAccount}},deadLetterPolicy:
 (if .configuration.enableDeadLettering == \"true\" then {deadLetterTopic:
(\"projects/\"+.connectionProperties.projectID+\"/topics/
\"+.configuration.deadLetterTopic),
maxDeliveryAttempts:.configuration.maxDeliveryAttempts} else null
end),retainAckedMessages: (if .configuration.retainAckedMessages == \"true\"
then .configuration.retainAckedMessages else null
end),messageRetentionDuration: (if .configuration.messageRetentionDuration
==\"true\" then (.configuration |if .days!=null and .days!=\"7\" then((.days|
tonumber * 24*60*60 )+  (.hours|tonumber * 60*60) +  (.minutes|tonumber * 60)
| tostring +\"s\")  else ((.days|tonumber * 24*60*60 )| tostring +\"s\") end)
else null end) }| with_entries(select(.value != null)),updateMask:
(.configuration |
\"retainAckedMessages,ackDeadlineSeconds,pushConfig.oidcToken.serviceAccountEm
ail\" + (if .enableDeadLettering == \"true\" then
\",deadLetterPolicy.deadLetterTopic,deadLetterPolicy.maxDeliveryAttempts\"
else \"\" end) + (if .messageRetentionDuration == \"true\" then
\",messageRetentionDuration\" else \"\" end))}}"
            }
          },
          "actionDataFilter": {
            "results": "${ { body: .body, headers: .headers } }",
```

```
            "toStateData": "${ .output }"
          },
          "condition": "${.subscriptionExists==true}"
        },
        {
          "name": "CreateSubscriptionFunction",
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "${\"https://\"+.connectionProperties.hostName+\"/v1/
projects/\"+.connectionProperties.projectID+\"/subscriptions/
\"+.subscriptionName}",
              "method": "PUT",
              "body": "${{topic: (\"projects/
\"+.connectionProperties.projectID+\"/topics/
\"+.configuration.TopicName),retryPolicy:{minimumBackoff:
\"30s\",maximumBackoff:
\"300s\"},ackDeadlineSeconds: .configuration.ackDeadline,expirationPolicy:{},
pushConfig:
{pushEndpoint: .integrationProperties.INTEGRATION_FLOW_URL,oidcToken:
{serviceAccountEmail: .connectionProperties.serviceAccount}},
deadLetterPolicy: (if .configuration.enableDeadLettering == \"true\" then
{deadLetterTopic: (\"projects/\"+.connectionProperties.projectID+\"/topics/
\"+.configuration.deadLetterTopic),
maxDeliveryAttempts:.configuration.maxDeliveryAttempts} else null end),
filter: (if .configuration.filter then .configuration.filter else null end),
enableMessageOrdering: (if .configuration.enableMessageOrdering == \"true\"
then .configuration.enableMessageOrdering else null end),retainAckedMessages:
(if .configuration.retainAckedMessages == \"true\"
then .configuration.retainAckedMessages else null
end),messageRetentionDuration: (if .configuration.messageRetentionDuration
==\"true\" then (.configuration |if .days!=null and .days!=\"7\" then((.days|
tonumber * 24*60*60 )+  (.hours|tonumber * 60*60) +  (.minutes|tonumber * 60)
| tostring +\"s\")  else ((.days|tonumber * 24*60*60 )| tostring +\"s\") end)
else null end) } | with_entries(select(.value != null))}"
            }
          },
          "actionDataFilter": {
            "results": "${ { body: .body, headers: .headers } }",
            "toStateData": "${ .output }"
          },
          "condition": "${.subscriptionExists==false}"
        }
      ],
      "name": "startState",
      "type": "operation",
      "end": true
    }
  ]
},
"DeleteSubscriptionFlow": {
  "id": "DeleteSubscriptionFlow",
  "version": "0.1",
  "start": "startState",
  "specVersion": "0.8",
  "functions": [
```

```
    {
      "name": "generalRestFunc",
      "operation": "connectivity::rest",
      "type": "custom"
    }
  ],
  "states": [
    {
      "actions": [
        {
          "name": "DeleteSubscriptionFunction",
          "functionRef": {
            "refName": "generalRestFunc",
            "arguments": {
              "uri": "https://{hostName}/v1/projects/{projectID}/
subscriptions/{name}",
              "method": "DELETE",
              "parameters": {
                "hostName": "${.connectionProperties.hostName}",
                "name": "${(.integrationProperties.INTEGRATION_FLOW_URL|
split(\"/\") | .[-3]) as $intName |
(.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\") | .[-2]) as $flow
| (.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\") | .[2]) as$host
| (if (.integrationProperties.INTEGRATION_FLOW_URL| split(\"/\")| .[-5])|
test(\"project\") then $intName+\"_\"+\"project_\"+$flow+\"_\"+$host
else $intName+\"_\"+$flow+\"_\"+$host end )}",
                "projectID": "${.connectionProperties.projectID}"
              }
            }
          },
          "actionDataFilter": {
            "results": "${ { body: .body, headers: .headers } }",
            "toStateData": "${ .output }"
          }
        }
      ],
      "name": "startState",
      "type": "operation",
      "end": true
    }
  ]
}
```

Understand Trigger Security

You must specify a security policy for your adapter according to the trigger type you choose.

With webhook triggers, an external application sends the event information to the adapter. You must configure security policies to authenticate or validate the incoming messages. You may use any of the trigger-specific security policies. See What Is Webhook Security.

With polling triggers, a polling flow in the adapter logic invokes the external application. Therefore, you must configure an invoke-specific security policy.

What Is Webhook Security

Webhooks require the external application or service to send the event information to the adapter using the HTTP protocol.

Before processing these messages, you must use a security logic to authenticate or validate that the messages are from a valid source or sender.

Additionally, external applications or services may transmit validation information during the exchange of messages. For example, the message may include an event-specific key, which you can compare with the key sent with the event. This allows you to verify if the message is sent from the correct source.

Webhooks can implement security at the following stages:

- Registration of the webhook during activation. See Validation During Webhook Registration.
- Reception of webhook messages during runtime. See Authentication and Validation of Webhook Messages.

> **Note:**
>
> Some scenarios may require a webhook trigger to invoke APIs of the external application, thereby requiring authentication mechanisms before invoking. To handle this scenario, you can design a composite security policy in the connection section of the adapter definition document. For more information on how to configure a trigger connection that can support inbound message authentication, see Create a Trigger Connection Definition to Invoke Protected Endpoints .

Validation During Webhook Registration

An external application or service may enforce validation during the webhook registration process.

For complete details about the webhook registration process, see Understand Event Subscription.

During this registering process, the external application may enforce the following requirements:

- Respond to a ping sent to the endpoint, which the adapter receives.

  The response may either be a simple acknowledgment or a more specific or custom response message.
- Store security information regarding the validation of a runtime message from the external application.
- Initialize the external application with the information that allows it to obtain authorization or authentication to send the message successfully to Oracle Integration.

You can use the Rapid Adapter Builder to model all of these scenarios.

The most common requirement of registration are ping requests. If the registration of the endpoint is manual, the integration that processes the event must be activated before a ping can be acknowledged. Additionally, the trigger must model a `validationRequests` logic that returns the acknowledgment for the ping request.

You can model this logic as shown in the following example, where a condition is applied to discern the source of the ping. In this scenario, the acknowledgment is issued if the request has a header that contains a key of `x-custom-event` and a value of `ping`. The response is created with a HTTP status code of `204`. The response can be of any content that the event producer requires.

```
"validationRequests": [
  {
    "condition": "${.request.headers.\"x-custom-event\"==\"ping\"}",
    "response": {
      "status": 204
    }
  }
],
```

Additionally, in some scenarios, the registration might require processing of the ping or incoming request. In such scenarios, you can use a flow to model the additional logic. The set of objects in the `validationRequests` are executed to ensure that the message is valid. If the conditions are not fulfilled, the incoming message is not executed during runtime.

Authentication and Validation of Webhook Messages

You can implement authentication and validation mechanisms for webhook messages during the runtime execution.

In the `connection` section of the adapter definition document, you can define the runtime security enforcement for inbound messages as a *composite* security policy. The trigger security policy supports both inbound and outbound sections.

Additionally, some scenarios may require the webhook trigger to invoke APIs of the external application, thus requiring authentication mechanisms for invoking. These scenarios include:

- Configuration of a trigger that reads data from the external application to allow for selection of options.
- Dynamic schema creation that requires reading of metadata information from the external application.
- Subscription registration and deregistration that requires calling the external application's API to create these entities.

The composite security policy consists of the following important properties:

- `policyOutbound`: Models the external application's authentication scheme for invoking its API. Most applications use the standard OAuth policies to protect their APIs and permit outbound calls.
- `policyInbound`: Models the security schemes that allow Oracle Integration to authenticate or validate runtime messages sent by external applications.

> **Note:**
>
> External applications and services mostly influence the design of webhook security. Most services use digital signatures to validate webhook messages.

The `policyInbound` section supports digital signatures. The valid values are:

- `DIGITAL_SIGNATURE`
- `HMAC_SIGNATURE_VALIDATION`
- `RSA_SIGNATURE_VALIDATION`
- `JWT_VALIDATION`

Here's a sample code that shows a security policy definition with Github triggers and the `HMAC_SIGNATURE_VALIDATION` managed policy:

```
"securityPolicies": [
   {
     "type": "composite",
     "description": "This policy is used by OIC for validating incoming
requests as well as for invoking GitHub APIs",
     "displayName": "GitHub security policy",
     "scope": "TRIGGER",
     "policyOutbound": {
       "type": "managed",
       "policy": "OAUTH2.0_AUTHORIZATION_CODE_CREDENTIALS",
       "securityProperties": [
         {
           "name": "oauth.client.id",
           "displayName": "Client Id",
           "description": "Client Id",
           "shortDescription": "Client Id",
           "required": true,
           "hidden": false
         },
         {
           "name": "oauth.client.secret",
           "displayName": "Client Secret",
           "description": "Client Secret",
           "shortDescription": "Client Secret",
           "required": true,
           "hidden": false
         },
         {
           "name": "oauth.access.token.uri",
           "default": "https://github.com/login/oauth/access_token",
           "required": false,
           "hidden": true
         },
         {
           "name": "oauth.scope",
           "displayName": "Scope",
           "description": "The scope of the access request",
           "shortDescription": "scope",
           "required": false,
           "hidden": false
         },
         {
           "name": "oauth.auth.code.uri",
           "default": "https://github.com/login/oauth/authorize",
           "required": false,
           "hidden": true
```

```
        },
        {
          "name": "authRequest",
          "default": "${(.securityProperties.\"oauth.auth.code.uri\") + \"?
response_type=code&client_id=\" + (.securityProperties.\"oauth.client.id\") +
\"&redirect_uri=${redirect_uri}&scope=\" +
(.securityProperties.\"oauth.scope\")}",
          "required": true,
          "hidden": true
        },
        {
          "name": "accessTokenRequest",
          "description": "Access Token Request that should be used to fetch
the access token",
          "shortDescription": "Example: -X <Method> -H <headers> -d <string-
data> <access-token-uri>?<query params>",
          "default": "${\"-X POST -H \" + \"\\\"Accept: application/json\\\"
-H \\\"Content-Type: application/x-www-form-urlencoded\\\" -d \\\"false\\\" \\
\"\" +  (.securityProperties.\"oauth.access.token.uri\") + \"?code=$
{auth_code}&client_id=\" + (.securityProperties.\"oauth.client.id\") +
\"&redirect_uri=${redirect_uri}&client_secret=\" +
(.securityProperties.\"oauth.client.secret\") +
\"&grant_type=authorization_code\\\"\"}",
          "required": true,
          "hidden": true
        },
        {
          "name": "refreshTokenRequest",
          "description": "Refresh Token Request that should be used to fetch
the access token",
          "shortDescription": "Example: -X <Method> -H <headers> -d <string-
data> <refresh-token-uri>?<query params>",
          "required": true,
          "default": "${\"-X POST -H \" + \"\\\"Accept: application/json\\\"
-H \\\"Content-Type: application/x-www-form-urlencoded\\\" -d \\\"false\\\" \\
\"\" +  (.securityProperties.\"oauth.access.token.uri\") + \"?refresh_token=$
{refresh_token}&client_id=\" + (.securityProperties.\"oauth.client.id\") +
\"&redirect_uri=${redirect_uri}&client_secret=\" +
(.securityProperties.\"oauth.client.secret\") + \"&grant_type=refresh_token\\
\"\"}",
          "hidden": true
        },
        {
          "name": "$auth_code",
          "description": "Regex that identifies the auth code",
          "shortDescription": "Auth Code Regex",
          "required": false,
          "default": "${auth_code}",
          "hidden": true
        },
        {
          "name": "$access_token",
          "description": "Regex that identifies the access token",
          "shortDescription": "Access Token Regex",
          "required": false,
          "default": "access.[tT]oken",
```

```
            "hidden": true
          },
          {
            "name": "$refresh_token",
            "description": "Regex that identifies the refresh token",
            "shortDescription": "Default: refresh.[tT]oken",
            "required": false,
            "default": "refresh.[tT]oken",
            "hidden": true
          },
          {
            "name": "$expiry",
            "description": "Numeric value (in seconds)that specifies the
expiry interval or a regex that identifies when the access token expires.",
            "shortDescription": "Default: expires_in",
            "required": false,
            "default": "expires.*",
            "hidden": true
          },
          {
            "name": "$token_type",
            "description": "Regex that identifies the access token type.",
            "shortDescription": "Default: token.?[tT]ype",
            "required": false,
            "default": "token.?[tT]ype",
            "hidden": true
          },
          {
            "name": "accessTokenUsage",
            "description": "Access token usage. A curl type syntax to
illustrate how access token should be passed to access a protected resource.",
            "shortDescription": "Default: -H Authorization ${token_type} $
{access_token}",
            "required": false,
            "default": "-H Authorization: Bearer ${access_token}",
            "hidden": true
          }
        ]
      },
      "policyInbound": {
        "type": "managed",
        "policy": "HMAC_SIGNATURE_VALIDATION",
        "securityProperties": [
          {
            "name": "signature",
            "hidden": true,
            "required": true,
            "default": "${connectivity::hexDecode(.request.headers.\"x-hub-
signature-256\" | split(\"sha256=\")[1])}"
          },
          {
            "name": "signatureString",
            "displayName": "Request Signature Location",
            "hidden": true,
            "required": true,
            "default": "${.request.body}"
```

```
        },
        {
          "name": "signatureAlgorithm",
          "displayName": "Request Signature Algorithm",
          "hidden": true,
          "required": true,
          "default": "HMACSHA256"
        },
        {
          "name": "secret",
          "displayName": "Shared Secret",
          "hidden": false,
          "required": true
        },
        {
          "name": "timestampValidator",
          "displayName": "Timestamp Validation",
          "hidden": true,
          "required": false,
          "default": ""
        }
      ]
    }
  }
```

Here's a sample code that shows the modeling for the `JWT_VALIDATION` security policy, which Google Cloud Publication Subscription dictates for message validation.

```
"securityPolicies": [
  {
    "type": "managed",
    "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
    "description": "This policy is used by OIC for invoking GCP Pub/Sub APIs",
    "displayName": "Google Authentication/Authorization Policy",
    "scope": "ACTION",
    "securityProperties": [
      {
        "name": "oauth.client.id",
        "displayName": "Google Client ID",
        "description": "Google Client ID",
        "shortDescription": "Example: 6-
jdek24mqqhdleori19r.apps.googleusercontent.com",
        "required": true,
        "hidden": false
      },
      {
        "name": "oauth.client.secret",
        "displayName": "Google Client Secret",
        "description": "Google Client Secret",
        "shortDescription": "Example: GOCDPX-gBQdjksUPXWer4",
        "required": true,
        "hidden": false
      },
      {
        "name": "oauth.access.token.uri",
```

```
        "default": "https://oauth2.googleapis.com/token",
        "required": false,
        "hidden": true
      },
      {
        "name": "oauth.scope",
        "default": "https://www.googleapis.com/auth/pubsub",
        "required": false,
        "hidden": true
      },
      {
        "name": "oauth.auth.code.uri",
        "default": "https://accounts.google.com/o/oauth2/auth",
        "required": false,
        "hidden": true
      },
      {
        "name": "clientAuthentication",
        "default": "client_credentials_in_body",
        "required": false,
        "hidden": true
      }
    ],
    "authflow": "flow:extended-oauth"
  },
  {
    "type": "composite",
    "description": "This policy is used by OIC for validating incoming
requests as well as for invoking GCP Pub/Sub APIs",
    "displayName": "GCP Pub/Sub security policy",
    "scope": "TRIGGER",
    "policyOutbound": {
      "type": "managed",
      "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
      "securityProperties": [
        {
          "name": "oauth.client.id",
          "displayName": "Google Client ID",
          "description": "Google Client ID",
          "shortDescription": "Example: 35532456156-
jdek24mdmlqutog3gnc3rfqqhdleori19r.apps.googleusercontent.com",
          "required": true,
          "hidden": false
        },
        {
          "name": "oauth.client.secret",
          "displayName": "Google Client Secret",
          "description": "Google Client Secret",
          "shortDescription": "Example: GOCDPX-gBQdjnPG4Hdi940zJCuksUPXWer4",
          "required": true,
          "hidden": false
        },
        {
          "name": "oauth.access.token.uri",
          "default": "https://oauth2.googleapis.com/token",
          "required": false,
```

```
            "hidden": true
          },
          {
            "name": "oauth.scope",
            "default": "https://www.googleapis.com/auth/pubsub",
            "required": false,
            "hidden": true
          },
          {
            "name": "oauth.auth.code.uri",
            "default": "https://accounts.google.com/o/oauth2/auth",
            "required": false,
            "hidden": true
          },
          {
            "name": "clientAuthentication",
            "default": "client_credentials_in_body",
            "required": false,
            "hidden": true
          }
        ]
      },
      "policyInbound": {
          "type": "managed",
          "policy": "JWT_VALIDATION",
          "securityProperties": [
              {
                  "name": "subjectClaim",
                  "displayName": "Subject claim Override",
                  "hidden": true,
                  "required": false,
                  "default": ""
              },
              {
                  "name": "jwtToken",
                  "displayName": "JWT Token",
                  "hidden": true,
                  "required": true,
                  "default": "${.request.headers.authorization|split(\" \")|.
[1]}"
              },
              {
                  "name": "signatureKey",
                  "displayName": "JWK URL",
                  "hidden": true,
                  "required": true,
                  "default": "https://www.googleapis.com/oauth2/v3/certs"
              },
              {
                  "name": "customClaimsValidation",
                  "displayName": "Custom Claims Validation",
                  "hidden": true,
                  "required": false,
                  "default": ""
              }
```

```
        ]
    }
```

Here's a sample code that shows a security policy where the `policyInbound` section uses the OAuth managed policy, `OAUTH_INBOUND`. The webhook message authenticates with Oracle Integration similar to how all applications invoke APIs of Oracle Integration.

However, this policy requires a manual setup to initialize the external application with proper credentials, and the external application must obtain the right authentication artifacts.

A bearer token in the header of the webhook message allows Oracle Integration to authenticate the inbound message.

```
{
    "type": "composite",
    "description": "This policy is used by Oracle Integration to validate
incoming requests for Zuora APIs",
    "displayName": "Zuora Composite Security Policy",
    "scope": "TRIGGER",
    "policyOutbound": {
        "type": "managed",
        "policy": "OAUTH2.0_CLIENT_CREDENTIALS",
        "securityProperties": [
            {
                "name": "oauth.client.id",
                "displayName": "Client Id",
                "description": "Used to identify the client(the software
requesting an access token) that is making the request. The value passed in
this parameter must exactly match the value shown in your API console
project.",
                "shortDescription": "Used to identify the client.",
                "required": true,
                "hidden": false
            },
            {
                "name": "oauth.client.secret",
                "displayName": "Client Secret",
                "description": "Used to authorize the client(the software
requesting an access token) that is making the request.  The value passed in
this parameter must exactly match the value shown in your API console
project.",
                "shortDescription": "<unique random string matches your API
console project>",
                "required": true,
                "hidden": false
            },
            {
                "name": "oauth.access.token.uri",
                "description": "A request should be sent to this URI for
obtaining an access token.",
                "default": "${\"https:/\"+\"/
\"+.connectionProperties.invokeHostName + \"/oauth/token\"}",
                "required": true,
                "hidden": true
            },
            {
```

```
                "name": "oauth.scope",
                "description": "Permissions your application is requesting on
behalf of the user.",
                "shortDescription": "For example: read,write.",
                "default": "",
                "required": false,
                "hidden": true
            },
            {
                "name": "accessTokenRequest",
                "description": "Access Token Request that should be used to
fetch the access token",
                "shortDescription": "Example: -X <Method> -H <headers> -d
<string-data> <access-token-uri>?<query params>",
                "default": "${\"-X POST -H \" + \"\\\"Content-Type:
application/x-www-form-urlencoded\\\" -d \\\"client_id=\" +
(.securityProperties.\"oauth.client.id\") + \"&client_secret=\" +
((.securityProperties.\"oauth.client.secret\"|
=@uri).securityProperties.\"oauth.client.secret\") +
\"&grant_type=client_credentials\\\" \" + \"\\\"https://\"
+ .connectionProperties.invokeHostName + \"/oauth/token\\\"\"}",
                "required": true,
                "hidden": true
            },
            {
                "name": "refreshTokenRequest",
                "description": "Refresh Token Request that should be used to
fetch the access token",
                "shortDescription": "Example: -X <Method> -H <headers> -d
<string-data> <refresh-token-uri>?<query params>",
                "required": false,
                "default": "",
                "hidden": true
            },
            {
                "name": "$access_token",
                "description": "Regex that identifies the access token",
                "shortDescription": "Access Token Regex",
                "required": false,
                "default": "access.[tT]oken",
                "hidden": true
            },
            {
                "name": "$refresh_token",
                "description": "Regex that identifies the refresh token",
                "shortDescription": "Default: refresh.[tT]oken",
                "required": false,
                "default": "refresh.[tT]oken",
                "hidden": true
            },
            {
                "name": "$expiry",
                "description": "Numeric value (in seconds)that specifies the
expiry interval or a regex that identifies when the access token expires.",
                "shortDescription": "Default: expires_in",
                "required": false,
```

```
                "default": "expires.*",
                "hidden": true
            },
            {
                "name": "$token_type",
                "description": "Regex that identifies the access token type.",
                "shortDescription": "Default: token.?[tT]ype",
                "required": false,
                "default": "token.?[tT]ype",
                "hidden": true
            },
            {
                "name": "accessTokenUsage",
                "description": "Access token usage. A curl type syntax to
illustrate how access token should be passed to access a protected resource.",
                "shortDescription": "Default: -H Authorization ${token_type} $
{access_token}",
                "required": false,
                "default": "-H Authorization: Bearer ${access_token}",
                "hidden": true
            }
        ]
    },
    "policyInbound": {
        "type": "managed",
        "policy": "OAUTH_INBOUND"
    }
}
```

Understand Configuration Scenarios

In some scenarios, the integration developer may have to configure the trigger in Oracle Integration. The configuration may involve either selecting values from a list of options or entering information into a field.

For example, the external application may provide a generic mechanism to receive an event from any object (like a creation event). However, the trigger configuration requires the integration developer to select the type of object to monitor. In this scenario, the integration developer must select the required object.

When an action requires configuration, the Rapid Adapter Builder provides a separate adapter page, where the integration developer can perform the required configuration by selecting values, entering text, or entering data in complex multi-values table. These values are used for schema generation for the integration mapper or in the runtime execution.

On the configuration page, the Rapid Adapter Builder supports design of multiple user-interface components, like the dropdown field, text field, and so on. For more information, see Design User Interface Components of an Adapter and Design of Configuration Fields.

Runtime Implementation of Triggers

For a particular scenario, if you need to transform the inbound webhook message into a more mapper-friendly form as modeled by the `request` property, you can define the runtime implementation for a trigger.

The following code sample shows a trigger definition that defines a `request` structure that's different than the inbound webhook. The `execute` property processes the webhook and transforms the raw request into the desired schema (specified in the `request` property). .

```
"GCPPubSubNotificationTrigger": {
  "displayName": "Subscribe to a topic",
  "description": "This trigger detects when a message is publised to a
subscribed topic.",
  "type": "webhook",
  "execute": "flow:mapNotificationFlow",
  "httpMethod": "POST",
  "request": "flow:DynamicInputFlowTrigger",
  "webhook": {
    "headers": [
      {
        "name": "from",
        "type": "string"
      },
      {
        "name": "content-type",
        "type": "string"
      },
      {
        "name": "host",
        "type": "string"
      },
      {
        "name": "content-length",
        "type": "string"
      },
      {
        "name": "user-agent",
        "type": "string"
      },
      {
        "name": "authorization",
        "type": "string"
      },
      {
        "name": "x-span-id",
        "type": "string"
      },
      {
        "name": "transfer-encoding",
        "type": "string"
      },
      {
        "name": "connection",
        "type": "string"
```

```
      },
      {
        "name": "accept-encoding",
        "type": "string"
      },
      {
        "name": "accept",
        "type": "string"
      }
    ],
    "body": {
      "schemaType": "application/schema+json",
      "schema": {
        "$ref": "#/schemas/NotificationSchemaGCP"
      }
    }
  },
  "subscription": {
    "register": "flow:CreateSubscriptionFlow",
    "deregister": "flow:DeleteSubscriptionFlow"
  },
  "configuration": [
    {
      "name": "TopicName",
      "displayName": "Select topic",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:ListTopicsUIFlow",
      "required": true
    },
    {
      "name": "SchemaName",
      "displayName": "Schema type associated with topic",
      "description": "Schema type associated with topic",
      "type": "TEXT_BOX",
      "default": "flow:GetSchemaNameFlow",
      "required": true,
      "dependencies": {
        "TopicName": {
          "values": []
        }
      }
    },
    {
      "name": "SchemaSupport",
      "displayName": "Schema not supported",
      "description": "Schema not supported",
      "type": "TEXT_BOX",
      "default": "flow:SchemaNotSuportedFlow",
      "required": true,
      "dependencies": {
        "SchemaName": {
          "values": [
            "PROTOCOL_BUFFER",
            "AVRO"
          ]
```

```
        }
      }
    },
    {
      "name": "InputField",
      "displayName": "Provide JSON Sample",
      "description": "Provide JSON sample for data",
      "type": "TEXT_AREA",
      "required": true,
      "validation": "flow:ValidateJsonFlow",
      "dependencies": {
        "SchemaName": {
          "values": [
            "No schema"
          ]
        }
      }
    },
    {
      "name": "filter",
      "displayName": "Subscription filter",
      "description": "Max 256 characters. Example: attributes:k OR
(attributes.k1=\"v\" AND NOT hasPrefix (attributes.k2,\"v\"))",
      "type": "TEXT_AREA",
      "required": false,
      "validation": "flow:ValidateFilterLengthFlow"
    },
    {
      "name": "ackDeadline",
      "displayName": "Acknowledgement deadline",
      "description": "Deadline time is from 10 seconds to 600 seconds.",
      "type": "TEXT_BOX",
      "required": true,
      "default": "10",
      "validation": "flow:ValidateAckDeadlineFlow"
    },
    {
      "name": "enableDeadLettering",
      "displayName": "Enable dead lettering",
      "description": "Subscriptions may configure a maximum number of
delivery attempts. When a message cannot be delivered, it is republished to
the specified dead-letter topic.",
      "default": "false",
      "type": "CHECK_BOX",
      "required": false
    },
    {
      "name": "deadLetterTopic",
      "displayName": "Select a dead-letter topic",
      "description": "Topic should have a subscription associated to it.",
      "type": "COMBO_BOX",
      "options": "flow:ListTopicsUIFlow",
      "required": true,
      "dependencies": {
        "enableDeadLettering": {
          "values": [
```

```
            "true"
          ]
        }
      }
    },
    {
      "name": "maxDeliveryAttempts",
      "displayName": "Maximun delivery attempts",
      "description": "Maximum delivery attempts is from 5 to 100.",
      "type": "TEXT_BOX",
      "required": true,
      "default": "5",
      "validation": "flow:ValidateMaxDeliveryAttemptsFlow",
      "dependencies": {
        "enableDeadLettering": {
          "values": [
            "true"
          ]
        }
      }
    },
    {
      "name": "enableMessageOrdering",
      "displayName": "Order messages with an ordering key",
      "description": "When enabled, messages tagged with the same ordering
key will be received in the order that they are published. This option cannot
be changed later.",
      "default": "false",
      "type": "CHECK_BOX",
      "required": false
    },
    {
      "name": "retainAckedMessages",
      "displayName": "Retain acknowledged messages",
      "description": "When enabled, acknowledged messages are retained for
the message retention duration specified above. This increases message
storage fees.",
      "default": "false",
      "type": "CHECK_BOX",
      "required": false
    },
    {
      "name": "messageRetentionDuration",
      "displayName": "Specify message retention duration",
      "description": "Duration is from 10 minutes to 7 days",
      "default": "false",
      "type": "CHECK_BOX",
      "required": false
    },
    {
      "name": "days",
      "displayName": "Days",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:DaysFlow",
      "required": true,
```

```
      "dependencies": {
        "messageRetentionDuration": {
          "values": [
            "true"
          ]
        }
      }
    },
    {
      "name": "hours",
      "displayName": "Hours",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:HoursFlow",
      "required": true,
      "dependencies": {
        "days": {
          "values": [
            "0",
            "1",
            "2",
            "3",
            "4",
            "5",
            "6"
          ]
        }
      }
    },
    {
      "name": "minutes",
      "displayName": "Minutes",
      "description": "",
      "type": "COMBO_BOX",
      "options": "flow:MinutesFlow",
      "required": true,
      "dependencies": {
        "days": {
          "values": [
            "0",
            "1",
            "2",
            "3",
            "4",
            "5",
            "6"
          ]
        }
      }
    }
  ]
}
```

The following sample code shows a typical scenario, where the data transformation does not need to call an external API. Instead, the flow logic uses jq expressions and out-of-box

functions provided by the Rapid Adapter Builder, which can be called from jq to execute this transformation.

In this example, a part of the webhook inbound message sent by the external application is base64 encoded. The flow logic extracts this part of the message and transforms it into the form that is presented in the integration mapper.

```
"mapNotificationFlow": {
  "id": "mapNotificationFlow",
  "version": "0.1",
  "start": "startState",
  "specVersion": "0.8",
  "functions": [
    {
      "name": "mapNotification",
      "operation": ".input|(if .message.attributes != null then
(.message.attributes | to_entries | map({key: .key, value: .value}))
as $attributes | .message.attributes = $attributes else . end | .message.data
|= (if . then @base64d | fromjson else empty end))",
      "type": "expression"
    }
  ],
  "states": [
    {
      "actions": [
        {
          "functionRef": "mapNotification",
          "actionDataFilter": {
            "toStateData": "${ .output }"
          }
        }
      ],
      "name": "startState",
      "type": "operation",
      "end": true
    }
  ]
}
```

Workflow to Implement a Trigger

Use the following approach as a general guideline to build a trigger.

- Complete the Prerequisite Tasks
- Design the Trigger
- Test an Event with a Request Bucket
- Create a Basic Trigger
- Test the Trigger
- Enhance the Functionality of a Trigger

Complete the Prerequisite Tasks

Before you start authoring a trigger definition, complete the following prerequisite tasks.

- Understand the requirements of the adapter you plan to build.
- Understand the type of integration tasks that the adapter should support.

  That is, understand the actions that the adapter must support, including the contract that needs to be exposed to the integration developer in the mapper.
- Model and test the authentication through the connection definition.
- Understand and be able to test the external application's API.

  You may require a test-development account with the external application or service.

Design the Trigger

As a first step, you must:

- Determine the behavior of the trigger.
- Determine the event that triggers the runtime execution.

For example, an external application may send an order-create event as a webhook. Determine if this event should be a trigger in your adapter.

Test an Event with a Request Bucket

After you identify the required event, test it using various request buckets, such as mockable.io, request.bin, or webhook.site. These sites expose an endpoint that the external application can send an event for testing. The request buckets show the event body and the various headers that are part of the event.

Create a Basic Trigger

Using the VS Code extension for Rapid Adapter Builder, create the skeletal trigger code from a template and modify it to map to the event you have tested.

The creation of the input schema is important. Using the payload of the event (which can be obtained from the request bucket), you can convert the JSON content into a JSON schema for the purpose of designing the webhook.

> **Note:**
>
> To quickly test the delivery of an event and its usage in an integration, send the event without any security policy.

Test the Trigger

To test the trigger using Oracle Integration in runtime. Design an integration flow to use the trigger and verify if it is working as expected. Using the same steps that you used to trigger the event for the request bin, kick off an event and ensure that the integration is working.

This basic trigger implementation may have the following limitations:

- Manual registration of subscription on the external application's administrative user interface.
- No message validation or authentication.
- A static-event message structure according to the modeled event, received in the request-bin test.

Enhance the Functionality of a Trigger

The basic implementation of a trigger supports only a specific event and the event schema is defined by the request-bin test. It also requires the user to manually register and deregister the trigger.

However, for real-world business scenarios, you may need to enhance the functionality of the trigger. For example, an action that can create a customer object in Oracle ERP may require the integration developer to enter or select custom attributes that have been modeled by the ERP administrator. These custom attributes vary based on the ERP system instance.

To handle such complicated functionalities, refer the following topics:

- Use Postman Conversion of Flows to Design Configuration Fields
- Use Postman Conversion of Flows to Model Dynamic Schema

Trigger Properties and Sample Code

In the `triggers` sections of the adapter definition document, you define the actions or events that can start an integration.

- Properties
- Sample Code

Properties

For each trigger, you must specify the required information, such as the element to use for the request and response and how to process the data. The element to use for the request could be a business object, operation, or another element. The following information is required:

Property	Description
`displayName`	The name of the trigger that shows for an integration developer in Oracle Integration.
`description`	The description for the trigger.
`type`	The type of trigger. Notification signifies that the external source notifies Oracle Integration of the message but does not require a response.
`group`	The name of the group that is defined in the groups section of the adapter definition document.
`execute`	The uniform resource name (URN) that references the flow that implements the trigger. The flows are defined in the flows section of the adapter definition document.
`httpMethod`	The methodology to receive a message from the external source. For example, webhooks use a POST method with a body.

Property	Description
request	The schema for an incoming message of this trigger. The request contains schemaType and schema information. The schemaType defines the type of the schema, and supports only the type application/json. The schema defines the structure of the incoming message, and the structure is exposed to the integration for downstream mapping. It can be an inline schema or a reference to a schema definition in the schemas section of the adapter definition document. The request can refer a flow in the flows section. The flow implements return of the whole structure that includes both schemaType and schema. Example code: `"request": { "schemaType": "application/json", "schema": { "$ref": "#/schemas/staticInput" } }, "request": "flow:inputSchemaFlow"`
webhook	The third-party notification message structure, which includes headers, parameters,and body.
headers	A set of HTTP-header parameters that are defined by name and type.
parameters	A set of query parameters defined by name and type, and models the parameters from the webhook request to trigger.
body	The HTTP body including schema type and schema. This is similar to the request section.
subscription	This object models the ability of the trigger to programmatically subscribe to an event if the external source supports an API that supports subscription. The activation section models the subscription during activation and the deletion of a subscription during the deactivation of an integration. This section consists of filter, register and deregister properties.
filter	Enables event filtering at server through registration, or at the client through condition. This property can also filter ping events, and specify response status for client and response filters.
register	References a flow that executes an API call on the external service to programmatically register to an event.
deregister	References a flow that executes an API call on the external service to programmatically deregister to an event.
validationRequests	Identifies validation requests and allows override response behavior for such requests. Validation requests cannot execute integration flows. This property consists of condition, skipAuthentication and response.condition, jq expression on an incoming request object. It returns either true or false. If true, the validation request is ignored. The output must be a JSON object that represents response status, headers, and body.
configuration	Models the configuration option required to configure the trigger. Some triggers may require additional information that users must provide to allow runtime to function. The configuration properties are modeled as user interface widgets with data that is statically defined and fed by flows.

Sample Code

> **Note:**
>
> The sample code provided here is for a webhook trigger.

```
"triggers": {
   "notificationTypeTrigger": {
     "displayName": "Notification Type Trigger",
     "description": "This trigger detects when an event has been raised.",
     "group": "simpleTrigger",
     "type": "webhook",
     "httpMethod": "POST",
     "request": {
       "schemaType": "application/json",
       "schema": {
         "$ref": "#/schemas/staticInput"
       }
     },
     "webhook": {
       "headers": [
         {
           "name": "http-header-1",
           "type": "string"
         }
       ],
       "parameters": [
         {
           "name": "query-parameter-1",
           "type": "string"
         }
       ],
       "body": {
         "schemaType": "application/json",
         "schema": {
           "$ref": "#/schemas/staticInput"
         }
       }
     }
}
```

Work with Trigger Definitions

In the `triggers` section of the adapter definition document, design the trigger for your adapter. This section provides examples of how to design trigger definitions for various scenarios.

What Do You Want to Do?

Task	More information
Implement a trigger definition for webhook triggers	**Complete these tasks as needed:** • Define a Pass-Through Trigger • Define a Trigger with Webhook Transformation • Define a Trigger That Registers Itself with the Producer Application • Define Validation-Request Handling for a Trigger

Define a Pass-Through Trigger

Learn how to define a simple trigger that passes the incoming event payload directly to the integration flow, without performing any transformation or modification.

After creating this trigger, you have to manually register it with the target application.

Prerequisite:

You must define a schema representing the trigger object, to be presented to the Oracle Integration orchestration or mapper.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. In the `triggers` section of the document, create a skeleton trigger code.
4. Set the `type` property to `webhook`.
5. Specify the request method.
6. Specify the request schema.

   Sample code:

```
"triggers":{
   "pageChangeEvent": {
      "displayName": "Wiki Page Updated Event",
      "description": "This trigger detects when a Wiki Page Updated Event
is raised.",
      "type": "webhook",
      "httpMethod": "POST",
      "request": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/eventSchema"
```

```
        }
      }
    }
}
```

Define a Trigger with Webhook Transformation

Learn how to define a trigger that transforms a webhook payload into a simplified format, which is displayed in the Oracle Integration mapper.

Prerequisites:

- Define a schema representing the trigger object, to be presented to the Oracle Integrationorchestration or mapper.
- Define a schema representing the actual webhook received by Oracle Integration.
- Ensure a flow for transforming data exists in the adapter definition document. For information on how to define a flow, see Invoke a POST API.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.

2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.

3. In the `triggers` section of the document, define a pass-through trigger. See Define a Pass-Through Trigger.

4. Add a webhook with the body referencing its schema.

5. In `execute` object, refer to the data-transformation flow.

   The following sample code shows how to refer to a flow in the `execute` property:

```
"triggers":{
   "pageChangeEvent": {
      "displayName": "Wiki Page Updated Event",
      "description": "This trigger detects when a Wiki Page Updated Event
has been raised.",
      "type": "webhook",
      "httpMethod": "POST",
      "request": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/eventSchema"
        }
      },
      "execute": "flow:handleNotificationRequestFlow",
      "webhook": {
          "body": {
          "schemaType": "application/schema+json",
          "schema": {
            "$ref": "#/schemas/webhookSchema"
          }
        }
     }
```

```
    }
}
```

Define a Trigger That Registers Itself with the Producer Application

Learn how to define a trigger that registers programmatically with the external application.

Ensure that a flow to register and deregister with the external application exists in the adapter definition document. For information on how to define a flow, see Invoke a POST API.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.

2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.

3. In the `triggers` section of the document, define a pass-through trigger. See Define a Pass-Through Trigger.

4. Add a `subscription` section that references the flows to register and deregister.

   Sample code:

```
"triggers":{
   "pageChangeEvent": {
      "displayName": "Wiki Page Updated Event",
      "description": "This trigger detects when a Wiki Page Updated Event
is raised.",
      "type": "webhook",
      "httpMethod": "POST",
      "request": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/eventSchema"
        }
      },
      "subscription": {
        "register": "flow:eventRegisterFlow",
        "deregister": "flow:eventDeregisterFlow"
      }
    }
}
```

Define Validation-Request Handling for a Trigger

Learn how to handle the validation requests from external applications during the webhook registration process.

For complete information on validation requests during the registration process, see Validation During Webhook Registration.

As a first step, you must identify the validation-request contract from the external application. Additionally, for this example, we'll assume the following:

- The validation request includes the header event-type with the value `SubscriptionValidation`.
- The JSON payload includes the validation code.
- Oracle Integration returns a `200 OK` response with the validation code.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. In the `triggers` section of the document, define a pass-through trigger. See Define a Pass-Through Trigger.
4. Add a `validationRequests` block to the trigger code.

   Sample code for `validationRequests`:

```
"validationRequests": [{
            "condition": "${.request.headers.\"event-
type\"==\"SubscriptionValidation\"}",
             "response": {
                 "status": 200,
                "body": {
                    "validationResponse": "${.request.body.validationCode}"
                }
      }
],
```

   Sample code for a trigger with validation-request handling:

```
"triggers":{
   "pageChangeEvent": {
      "displayName": "Wiki Page Updated Event",
      "description": "This trigger detects when a Wiki Page Updated Event
is raised.",
      "type": "webhook",
      "httpMethod": "POST",
      "request": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/eventSchema"
        }
      },
     "validationRequests": [{
            "condition": "${.request.headers.\"event-
type\"==\"SubscriptionValidation\"}",
             "response": {
                 "status": 200,
                "body": {
                    "validationResponse": "${.request.body.validationCode}"
                }
      }
     ]
```

```
        }
    }
```

Work with Security Policies in Triggers

Specify details about the security policies for your trigger definitions.

What Do You Want to Do?

Task	More information
Implement a security policy for a trigger definition in the adapter definition document	**Everyone completes this task:** Implement security policies for each of your triggers. The following trigger related security policies are available: • Create a Trigger Connection Definition Using RSA Signatures • Create a Trigger Connection Definition to Invoke Protected Endpoints • Create a Trigger Connection Definition Using Basic Authentication • Create a Trigger Connection Definition Using HMAC Signatures • Implement a Trigger Connection Definition With JWT Signatures Security Policy • Create a Trigger Connection Definition Using OAuth2.0 Security Policy

Create a Trigger Connection Definition to Invoke Protected Endpoints

This topic describes how to implement a trigger connection that can invoke protected endpoints.

Before you start implementing a trigger connection to invoke protected endpoints, collect and note the following information:

- Check the webhook producer documentation for information on webhook protection. Identify the relevant trigger policy and its configuration.
- Check how the endpoints are protected. Identify the required invoke or action policies and related configuration.

1. In VS Code, click **Explorer** on the sidebar.

   The directory structure of your workspace folder is displayed.
2. Within the **definitions** directory listing, click the adapter definition document that you want to work on.

   The document is displayed in the VS Code editor.
3. Expand the `connection` section in the document.
4. In the `securityPolicies` object, set the following values:
   - Set the scope to `TRIGGER`.
   - Set the type as `composite`.
   - Configure `policyInbound` and `policyOutbbound` with the actual policy information.

Sample Code

In this example:

- Outbound uses the authorization code policy.
- Inbound uses the JWT validation policy.
- Outbound extension is defined at composite policy level.

```
{
       "type": "composite",
       "description": "This policy is used by OIC for validating incoming
requests as well as for invoking Pub/Sub APIs",
       "displayName": "Pub/Sub security policy",
       "scope": "TRIGGER",
       "policyOutbound": {
         "type": "managed",
         "policy": "OAUTH_AUTHORIZATION_CODE_CREDENTIALS",
         "securityProperties": [
           {
             "name": "oauth.client.id",
             "displayName": "PubSub Client ID",
             "description": "PubSub Client ID",
             "shortDescription": "Example: 35532456156-
jdek24mdmlqutog3gnc3rfqqhdleori19r",
             "required": true,
             "hidden": false
           },
           {
             "name": "oauth.client.secret",
             "displayName": "PubSub Client Secret",
             "description": "PubSub Client Secret",
             "shortDescription": "Example: GOCDPX-
gBQdjnPG4Hdi940zJCuksUPXWer4",
             "required": true,
             "hidden": false
           },
           {
             "name": "oauth.access.token.uri",
             "default": "https://oauth2.PubSub.com/token",
             "required": false,
             "hidden": true
           },
           {
             "name": "oauth.scope",
             "default": "https://www.PubSub.com/auth/pubsub",
             "required": false,
             "hidden": true
           },
           {
             "name": "oauth.auth.code.uri",
             "default": "https://accounts.PubSub.com/o/oauth2/auth",
             "required": false,
             "hidden": true
           },
           {
             "name": "clientAuthentication",
```

```
                "default": "client_credentials_in_body",
                "required": false,
                "hidden": true
            }
          ]
        },
        "policyInbound": {
            "type": "managed",
            "policy": "JWT_VALIDATION",
            "securityProperties": [
                {
                    "name": "subjectClaim",
                    "displayName": "Subject claim Override",
                    "hidden": true,
                    "required": false,
                    "default": ""
                },
                {
                    "name": "jwtToken",
                    "displayName": "JWT Token",
                    "hidden": true,
                    "required": true,
                    "default": "${.request.headers.authorization|split(\" \")|.
[1]}"
                },
                {
                    "name": "signatureKey",
                    "displayName": "JWK URL",
                    "hidden": true,
                    "required": true,
                    "default": "https://www.PubSub.com/oauth2/v3/certs"
                },
                {
                    "name": "customClaimsValidation",
                    "displayName": "Custom Claims Validation",
                    "hidden": true,
                    "required": false,
                    "default": ""
                }
            ]
        }
    }
```

Implement a Trigger Connection Definition With JWT Signatures Security Policy

This procedure gives an overview of how to implement a trigger connection definition that uses JWT signature security policy.

Prerequisite:

Check the webhook producer documentation on how the webhook is signed. Collect information on the following:

- Where the JWT token is sent as part of the request

- If there are any custom claims to validate
- Whether to use HMAC or RSA signatures
- Signature keys
- Override subject claim

1. Open the adapter definition document in Visual Studio Code Editor.
2. Navigate to the `connection` code section of the document and set the scope to `TRIGGER`.
3. Set the value of the security properties according to the information you collected in the prerequisites.

   Here is code sample 1 where:

   - JWT Token is extracted from authorization header (.request.headers.authorization|split(\" \")|.[1]).
   - Signature key is added in securityProperty signKey part of same policy and referred in jwtVerify function (.securityProperties.signKey). This key resolved to alias JWK URL of JWT issuer for example, "https://www.demosvc.com/oauth2/v3/certs".
   - No custom claims to validate.
   - Default subject claim (no overrides).

```
{
 "connection": {

    "securityPolicies": [
       {
            "type": "managed",
            "policy": "JWT_VALIDATION",
            "securityProperties": [
                {
                    "name": "jwtToken",
                    "displayName": "JWT Token",
                    "hidden": true,
                    "required": true,
                    "default": "${.request.headers.authorization|split(\"
\")|.[1]}"
                },
                {
                    "name": "signatureKey",
                    "displayName": "JWK URL",
                    "hidden": true,
                    "required": true,
                    "default": "https://www.demosvc.com/oauth2/v3/certs"
                },
                 {
                    "name": "subjectClaim",
                    "displayName": "Subject claim Override",
                    "hidden": true,
                    "required": false,
                    "default": ""
                },
                {
                    "name": "customClaimsValidation",
                    "displayName": "Custom Claims Validation",
```

```
                    "hidden": true,
                    "required": false,
                    "default": ""
                }
            ]
        }
    ]
  }
}
```

Here is code sample 2 where:

- JWT Token is extracted from authorization header (.request.headers.authorization|split(\" \")|.[1])
- Signature key is added in securityProperty signKey part of same policy and referred in jwtVerify function (.securityProperties.signKey). This key resolved to alias JWK URL of JWT issuer for example, "https://www.dummysvc.com/oauth2/v3/certs"
- Email is the subject claim
- Custom claims issuer validation

```
{
 "connection": {

    "securityPolicies": [
       {
            "type": "managed",
            "policy": "JWT_VALIDATION",
            "securityProperties": [
                {
                    "name": "jwtToken",
                    "displayName": "JWT Token",
                    "hidden": true,
                    "required": true,
                    "default": "${.request.headers.authorization|split(\"
\")|.[1]}"
                },
                {
                    "name": "signatureKey",
                    "displayName": "JWK URL",
                    "hidden": true,
                    "required": true,
                    "default": "https://www.dummysvc.com/oauth2/v3/certs"
                },
                 {
                    "name": "subjectClaim",
                    "displayName": "Subject claim Override",
                    "hidden": true,
                    "required": false,
                    "default": "email"
                },
                {
                    "name": "customClaimsValidation",
                    "displayName": "Custom Claims Validation",
                    "hidden": true,
```

```
                    "required": false,
                    "default": "${{\"iss\": \"https://dummyIssuer\"}}"
                }
            ]
        }
    ]
  }
}
```

Here is code sample 3 where the flow:

- Contains token, signkey, custom claims and subject claim.
- Fetches signature key.
- Uses a jq function.

```
{
 "connection": {

    "securityPolicies": [
       {
            "type": "managed",
            "policy": "JWT_VALIDATION",
            "securityProperties": [
                {
                    "name": "jwtToken",
                    "displayName": "JWT Token",
                    "hidden": true,
                    "required": true,
                    "default": "${.request.headers.authorization|split(\"
\")|.[1]}"
                },
                {
                    "name": "signatureKey",
                    "displayName": "JWK URL",
                    "hidden": true,
                    "required": true,
                    "default": "flow:getCerts"
                },
                 {
                    "name": "subjectClaim",
                    "displayName": "Subject claim Override",
                    "hidden": true,
                    "required": false,
                    "default": "email"
                },
                {
                    "name": "customClaimsValidation",
                    "displayName": "Custom Claims Validation",
                    "hidden": true,
                    "required": false,
                    "default": ""
                }
            ]
        }
    ]
```

```
  }
}

   "flows": {
    "getCerts": {
      "id": "getCerts",
      "description": "getCerts",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "getCertificate",
          "type": "expression",
          "operation": "{\"certificates\":[\"-----BEGIN CERTIFICATE-----\
\nMIIbTAe\\nFw0GVy\\nYX\/wQCMAAw\nIYX\\n-----END CERTIFICATE-----\"]}"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "getCertificate",
              "actionDataFilter": {
                "results": "${.certificates}"
                "toStateData": "${ .output }"
              }
            }
          ]
          "end": true
        }
      ]
    }
  } }
```

Create a Trigger Connection Definition Using OAuth2.0 Security Policy

This procedure gives an overview of how to implement a trigger connection definition that uses OAuth2.0 security policy.

1. Open the adapter definition document in Visual Studio Code Editor.
2. Navigate to the `connection` code section of the document and in the security policy, set the scope to `TRIGGER`.
3. Set the `refName` parameter to `OAUTH2.0_TOKEN_VALIDATION`.

   Sample code:

```
 "securityPolicies": [
     {
        "type": "managed",
        "refName": "OAUTH2.0_TOKEN_VALIDATION",
        "description": "Validates OAuth2.0 token",
```

```
      "displayName": "OAuth2.0",
      "scope": "TRIGGER"
    }
  ]
```

Create a Trigger Connection Definition Using Basic Authentication

HTTP Basic Authentication is a simple authentication scheme built into the HTTP protocol.

The client sends HTTP requests with the Authorization header that contains the word Basic followed by a space and a base64-encoded string containing the username and password. For example, `Basic username:password`.

1. Open the adapter definition document in Visual Studio Code Editor.
2. Navigate to the `connection` code section of the document.
3. Define the type of security policy as `managed` in the connection definition.
4. Define the value of the scope property as `TRIGGER`.
5. Set the hidden security properties to `true` value to hide the username and password.
6. Set any default parameter to `NA`.

> **Note:**
>
> The default parameter does not impact this security policy.

Sample code:

```
"securityPolicies": [
    {
       "type": "managed",
       "refName": "BASIC_AUTH",
       "description": "Validate Basic Authentication token",
       "displayName": "Basic Authentication",
       "scope": "TRIGGER",
       "securityProperties": [
          {
            "name": "username",
            "hidden": true,
            "required": false,
            "default": "NA"
          },
          {
            "name": "password",
            "hidden": true,
            "required": false,
            "default": "NA"
          }
       ]
    }
]
```

Create a Trigger Connection Definition Using HMAC Signatures

This procedure gives an overview of how to create a trigger connection definition with a security policy that uses hash-based message authentication code (HAMC) signature.

Prerequisites:

Check the webhook producer documentation on how the webhook is signed. Collect information on the following:

- What is signed?
- Which algorithm is used.
- How the signature is sent in the request? For example, find out which header and the format of the header value.
- Does it contain any information on timestamp to be validated. If yes where and what format.
- How to get the signing key?

1. Open the adapter definition document in Visual Studio Code Editor.
2. Navigate to the `connection` code section of the document and set the scope to `TRIGGER`.
3. Set the value of the security properties according to the information you collected in the prerequisites.

   Here is a code sample where:

   - Signature is computed by signing request body (.request.body).
   - Signature key is added in securityProperty signKey part of same policy and referred in HMAC function (.securityProperties.signKey).
   - Signature to match is present in HTTP header x-hub-signature-256.
   - Timestamp validation is not required and hence empty string.

```
"securityPolicies": [
     {
        "type": "managed",
        "refName": "HMAC_SIGNATURE_VALIDATION",
        "description": "Validates HMAC Signature",
        "displayName": "HMAC SIGNATURE VALIDATION",
        "scope": "TRIGGER",
        "securityProperties": [
           {
               "name": "signature",
               "hidden": true,
               "required": true,
               "default": "$
{connectivity::hexDecode(.request.headers.digest)}"
           },
           {
               "name": "signatureString",
               "displayName": "Request Signature Location",
               "hidden": true,
               "required": true,
               "default": "${.request.body}"
```

```
            },
            {
                "name": "signatureAlgorithm",
                "displayName": "Request Signature Location",
                "hidden": true,
                "required": true,
                "default": "HMACSHA256"
            },
            {
                "name": "secret",
                "displayName": "Shared Secret",
                "hidden": false,
                "default": true
            },
            {
                "name": "timestampValidator",
                "displayName": "Timestamp Validation",
                "hidden": true,
                "required": true,
                "default": ""
            },

        ]
       }
   ]
```

Here is another code sample where:

- Signature is computed by signing request body (.request.body).
- Signature key is added in securityProperty signKey part of same policy and referred in HMAC function (.securityProperties.signKey).
- Signature to match is present in HTTP header x-hub-signature-256.
- Timestamp validation is done against header x-timestamp. In this sample code, the timestamp is after current time and valid for only five minutes.

```
{
 "connection": {

    "securityPolicies": [
     {
        "type": "managed",
        "refName": "HMAC_SIGNATURE_VALIDATION",
        "description": "Validates HMAC Signature",
        "displayName": "HMAC SIGNATURE VALIDATION",
        "scope": "TRIGGER",
        "securityProperties": [
            {
                "name": "signature",
                "hidden": true,
                "required": true,
                "default": "$
{connectivity::base64Decode(.request.headers.digest)}"
            },
            {
```

```
                "name": "signatureString",
                "hidden": true,
                "required": true,
                "default": "${.request.body}"
            },
            {
                "name": "signatureAlgorithm",
                "displayName": "Request Signature Location",
                "hidden": true,
                "required": true,
                "default": "HMACSHA256"
            },
            {
                "name": "secret",
                "displayName": "Shared Secret",
                "hidden": false,
                "required": true
            },
            {
             "name": "timestampValidator",
             "displayName": "Shared Secret",
             "hidden": true,
             "required": true,
             "default": "(.request.headers."x-timestamp"|tonumber <=
(now*1000)) and  ((now*1000)-(.request.headers."x-timestamp"|tonumber)
<=300000 ) "
           }

        ]
       }
    ]
   }
}
```

Create a Trigger Connection Definition Using RSA Signatures

This procedure gives an overview of how to implement a trigger connection definition that uses RSA signatures.

Prerequisites

Check the webhook producer documentation on how the webhook is signed. Collect information on the following:

- What is signed?
- Which algorithm is used.
- How the signature is sent in the request? For example, find out which header and the format of the header value.
- Does it contain any information on timestamp to be validated. If yes where and what format.
- How to get the signing key?

> **Note:**
>
> Customer must upload certificate in Oracle Integration without which this security policy does not work.

1. Open the adapter definition document in Visual Studio Code Editor.
2. Navigate to the `connections` code section of the document and set the scope to `TRIGGER`.
3. Set the value of the security properties according to the information you collected in the prerequisites.

   Here is a code sample where:

   - Signed content is extracted from authorization header (.request.headers.authorization|split(\" \")|.[1]|split(\".\")|(.[0]+\".\"+.[1])).
   - Signature is base64URLencodeString and part of authorization header (connectivity::base64URLDecode(.request.headers.authorization|split(" ")|.[1]|split(".")|.[2])).
   - Signature key is added in securityProperty signKey part of same policy and referred in RSA function (.securityProperties.signKey). This key resolved to alias of certificate uploaded in OIC eg:- "orakey".
   - Timestamp validation is not required and hence empty string.

```
"securityPolicies": [
      {
         "type": "managed",
         "refName": "RSA_SIGNATURE_VALIDATION",
         "description": "Validates RSA Signature",
         "displayName": "RSA SIGNATURE VALIDATION",
         "scope": "TRIGGER",
         "securityProperties": [
            {
                "name": "signatureString",
                "displayName": "Signature Statement",
                "hidden": true,
                "required": true,
                "default": "${.request.body)}"
            },
            {
                "name": "signature",
                "displayName": "Signature Statement",
                "hidden": true,
                "required": true,
                "default": "$
{connectivity::base64URLDecode(.request.query.signature)}"
            },
{
                "name": "signatureAlgorithm",
                "displayName": "Request Signature Location",
                "hidden": true,
                "required": true,
                "default": "SHA256withRSA"
            },
            {
```

```
                "name": "signKey",
                "displayName": "Certificate Alias",
                "hidden": false,
                "required": true
            },
            {
                "name": "timestampValidator",
                "displayName": "Request Signature Location",
                "hidden": true,
                "required": true,
                "default": ""
            },

        ]
    }
]
```

14

Update Flow Definitions

When you convert a Postman collection into an adapter definition document, the VS Code extension for Rapid Adapter Builder generates flow definitions using the information in the collection. The flow definitions are in the `flows` section and are used to invoke the external APIs at runtime. You can update these definitions as needed, according to your requirements.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Flow Definitions Supported CNCF and jq Functions
Review the properties and syntax to define and see sample code	Flows Properties and Syntax for Dot (.) Notation
Define the requirements for a flow definition	Implement the Test-Connection Behavior Using Flows Switch States Conditionally in a Flow Invoke a GET API Invoke a POST API Response Transformation in Post-processing Send or Receive Form URL Encoded Content Receive Binary Content Send Multi-part Content Send Binary Content Override Default Content Handling in connectivity::rest Pre and Post Processing Expressions and Multiple Outbound Requests Fetch Design Time Configuration Values Using a Flow

Learn About Flow Definitions

Before updating the `flows` section of the adapter definition document, familiarize yourself with flows and their properties.

Topics:

- Introduction to Flow Definitions
- Supported CNCF and jq Functions

Introduction to Flow Definitions

The `flows` section of the adapter definition document defines the runtime and design-time activities that an adapter is capable of.

The following examples show how your adapter might use flows:

Design-time examples

- When an integration developer is configuring a connection based on your adapter, flows can pull data from an external application and display the data in the user interface.
- When an integration developer selects a custom object in the Integration Designer, flows can drive the dynamic schema.

Runtime examples

- Invoke APIs.
- Start preprocessing activities, like transforming a JSON payload to an avro string.
- Perform post processing, like transforming the avro-encoded data to a JSON key value.

Capabilities of flow definitions

Flow definitions support the following capabilities:

- Cloud Native Computing Foundation (CNCF) Serverless Workflow specification as DSL CNCF Serverless Workflows provide open-source, vendor-neutral DSL to define workflows.
- Expressions and additional custom functions, per Serverless Workflows guidelines
- Logging and outbound calls through custom Serverless Workflow functions
- jq expression language
- Custom jq functions to simplify preprocessing and postprocessing
- Accessing input and output context using dot (.) notation
- Multiple states, state transitions, and filtering

> **Note:**
>
> The adapter definition document supports only a subset of the available serverless workflow constructs. The Rapid Adapter Builder platform ensures that the flow definitions are compliant with the adapter definition document specification.

In the adapter definition document, you can use flow definitions in:

- Test connection

  Use flows to express test behavior for connections. The flow must return true or false in the output. For example, the flow can return true if the test connection is a success.
- Actions
  - Execute: Process actions during integration runtime execution.
  - Input: Build input schema for actions in design time.
  - Output: Build output schema for actions in design time.
  - Configuration: Fetch dynamic values for configuration properties in design time.
- Triggers
  - Request schema: Define the schema of a request for Oracle Integration in design time.
  - Webhook schema: Define the schema of a webhook request in design time.
  - Configuration: Fetch dynamic values for configuration properties in design time.

- Subscription
  - Register a webhook with a target application.
  - Unregister a webhook from a target application.
- Validation requests
  - Identify a validation request.
  - Build a response/out-of-band acknowledgment for a validation request.
- Inbound authentication signature policies, such as HMAC_SIGNATURE_VALIDATION, RSA_SIGNATURE_VALIDATION, and JWT_VALIDATION

Flow definitions for execution scenario

As the adapter developer, you can access requests, orchestration inputs, action/trigger configurations, and connection properties as context items using the dot (.) notation in the flow definitions. Additionally, you can add *temporary* context variables. After you define the output in the flow definition, Oracle Integration can consume and interpret the output.

Supported CNCF and jq Functions

Topics:

- CNCF Functions Support
- Supported jq Features
- Supported Custom CNCF Serverless Workflow
- Supported Custom jq Functions
- Supported Custom jq Functions for Encoding and Decoding

CNCF Functions Support

The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.

Adapter developers can use jq expression language while building flows using Cloud Native Computing Foundation (CNCF) functions.

Sample code to define a CNCF function:

```
"functions": [
        {
          "name": "extractWebhookIdFunc",
          "operation": "[{fileId: .configuration.fileId,
webhook: .entries[]}] | map(select(.webhook.target.id==.fileId)) | .
[0].webhook.id",
          "type": "expression"
        }
      ],
```

Sample code to implement the CNCF function:

```
"states": [
        {
```

```
            "actions": [
               {
                "functionRef": {
                  "refName": "extractWebhookIdFunc"
                },
                "actionDataFilter": {
                  "toStateData": "${ .webhookId }"
                }
              }
            ]
      }
]
```

The Rapid Adapter Builder platform supports the following constructs of the Serverless Workflow specification:

- Workflow Constructs
- Function Definitions
- State Definitions
- Operation State
- Switch State
- State Actions

Workflow Constructs

This table shows the supported operations in the CNCF workflow function.

Operation	Details
specVersion	Indicates the supported version of the Serverless Workflow specification. The Rapid Adapter Builder platform supports 0.8 version.
functions	A unique function name that follows the serverless workflow naming convention.
states	States dictate the control flow logic instructions of a workflow.

Function Definitions

This table shows the supported operations in the CNCF function definition.

Operation	Details
expression	The expression operation.
custom	A property that defines a list of function types that are set by the specification.

State Definitions

This table shows the supported operations in the state definition.

Operation	Details
functionRef	Defines the name of the referenced function. This can be either a string or an object.
actionDataFilter	A filter that you can use within an action.
condition	The condition operation.

Operation State

This table shows the supported operations in the CNCF operation state.

Operation	Details
Multiple actions	Supports multiple actions.
transition	One or more incoming or outgoing transitions of a serverless workflow state.
end	A boolean or object type.

Switch State

This table shows the supported operations in the CNCF switch state.

Operation	Details
dataConditions	A data-based condition statement. When evaluated to true causes a transition to a predefined workflow state.
defaultCondition	A switch state that enables the default transition of a workflow when there are no matching data conditions or if an event timeout occurs.

State Actions

This table shows the supported operations in the CNCF state actions.

Operation	Details
Operation State	A set of actions that are performed in sequence or in parallel. State transition occurs once all the actions are completed.
Switch State	A workflow gateway that directs transition of workflows based on defined conditions.

Supported jq Features

The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.

See the jq Manual for the complete list of jq functions. Read the limitations applicable for jq 1.6.

The Rapid Adapter Builder platform supports the following jq features:

Basic Filters

- .

- .foo, .foo.bar
- .[<string>], .[2], .[10:15]
- .[]
- .[]?
- ,
- ¿

Types and values

- Array construction - []
- Objects - {}

Built-in operators and functions

- Addition - +
- Subtraction - -
- Multiplication, division, modulo - *, /, and %
- length
- keys, keys_unsorted
- has(key)
- in
- path(path_expression)
- del(path_expression)
- to_entries, from_entries, with_entries
- select(boolean_expression)
- arrays, objects, iterables, booleans, numbers, normals, finites, strings, nulls, values, scalars
- empty
- error(message)
- map(x), map_values(x)
- paths, paths(node_filter), leaf_paths
- add
- any, any(condition), any(generator; condition)
- all, all(condition), all(generator; condition)
- flatten, flatten(depth)
- range(upto), range(from;upto) range(from;upto;by)
- floor
- sqrt
- tonumber
- tostring
- type

- infinite, nan, isinfinite, isnan, isfinite, isnormal
- sort, sort_by(path_expression)
- group_by(path_expression)
- min, max, min_by(path_exp), max_by(path_exp)
- unique, unique_by(path_exp)
- reverse
- contains(element)
- indices(s)
- index(s), rindex(s)
- inside
- startswith(str)
- endswith(str)
- combinations, combinations(n)
- ltrimstr(str)
- rtrimstr(str)
- explode
- implode
- split
- join(str)
- ascii_downcase, ascii_upcase
- recurse(f), recurse, recurse(f; condition), recurse_down
- walk
- ..
- setpath, delpaths, getpath
- utf8bytelength
- transpose
- String interpolation - \(foo)
- Convert to/from JSON
- Format strings and escaping
- Dates

Conditionals and comparisons

- ==, !=
- if-then-else
- >, >=, <=, <
- and/or/not
- Alternative operator - //
- try-catch

Regular expressions (PCRE)

- test(val), test(regex; flags)
- match(val), match(regex; flags)
- capture(val), capture(regex; flags)
- scan(regex), scan(regex; flags)
- split(regex; flags)
- splits(regex), splits(regex; flags)
- sub(regex; tostring) sub(regex; string; flags)

Advanced features

- Variables
  Use brackets to ensure operator precedence.
- Defining Functions
  Avoid duplicate function names.
- Reduce
- limit(n; exp)
- first(expr), last(expr), nth(n; expr)
- first, last, nth(n)
- foreach
- Recursion
  Up to five levels.
- Generators and iterators

Math

Assignment

- =
- |=
- +=, -=, *=, /=, %=, //=
- Complex assignments

Sample code showing how to use jq features

- to_entries
- from_entries
- @csv
- @tsv
- @base64
- @base64d

to_entries

Examples:

Input json	Expression in adapter definition document	Output json

{
"potato": true,
"broccoli": false
}
	{
	"uri": "http://
	surveyUrl.com",
	"method": "POST",
	"body": "$
	{ {vegetableFeedback : .
	input|to_entries } }"
	}
		{
		...
		"body" : {
		
		"vegetableFeedback" :
		[{
		"key" : "potato",
		"value" : true
		}, {
		"key" :
		"broccoli",
		"value" : false
		}]
		}
		}

from_entries

Examples:

Input json	Expression in adapter definition document	Output json

{
"vegetables": [
{
"key": "potato",
"value": true
},
{
"name": "broccoli",
"value": false
}
]
}
	"standardFromEntriesArgu
	ments": {
	"uri": "http://
	surveyUrl.com",
	"method": "POST",
	"body": "$
	{ {vegetableFeedback : .
	input.vegetables|
	from_entries } }"
	}
		{
		...
		"body" : {
		
		"vegetableFeedback" : {
		"potato" : true,
		"broccoli" : false
		}
		}

@csv

This function converts an array to CSV format.

Examples:

Input json	Expression in adapter definition document	Output json

{
"vegetables": [
{
"veggieName": "potato",
"like": true
},
{
"veggieName":
"broccoli",
"like": false
}
]
}
	{
	"arguments": {
	
	"body": "$
	{ .input.vegetables[]|
	[.veggieName, .like]|
	@csv }"
	
	}
	}
		{
		
		...
		
		"body" :
		
		"
		
		\"potato\",true
		\"broccoli\",false
		
		"
		
		}
		
		}

@tsv

This function converts an array to tab separated values.

Examples:

Input json	Expression in adapter definition document	Output json

{
"vegetables": [
{
"veggieName": "potato",
"like": true
},
{
"veggieName":
"broccoli",
"like": false
}
]
}
	{
	"arguments": {
	"body": "$
	{ .input.vegetables[]|
	[.veggieName, .like]|
	@tsv }"
	}
	}
		{
		...
		"body" :
		"
		potato true
		broccoli false
		"
		}
		}

@base64

Examples:

Input json	Expression in adapter definition document	Output json

{
"username":"a","password
":"b"
}
	{
	"arguments": {
	"body": "${ .username +
	":"+ .password|
	@base64 }"
	}
	}
		{
		
		...
		
		"body" : "YTpi" //
		(base64 encoded a:b)
		
		}

@base64d

Examples:

Input json	Expression in adapter definition document	Output json
{ "username":"a","password ":"b" }	{ "arguments": { "body": "${ .username + ":"+ .password|@base64| @base64d }" } }	{ ... "body" : "a:b" // (base64 encoded a:b) }

Supported Custom CNCF Serverless Workflow

The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.

Each input argument to a function can be a JSON node, string, or jq expression, and follows the CNCF syntax.

- connectivity::rest
- connectivity::log
- connectivity::avroSchemaToJsonSchema
- connectivity::avroEncode
- connectivity::avroDecode
- connectivity::jsonPayloadToXMLPayload
- connectivity::xmlPayloadToJsonSchema
- connectivity::sampleXmlToJson
- connectivity::updateSecurityProperty
- connectivity::getSecurityProperty
- connectivity::getPersistentState
- connectivity::setPersistentState

`connectivity::rest`

This function is an outbound HTTP call. By default, it uses the configured security policy. Values for arguments can be valid jq expressions.

This table describes JSON node input format for this function.

Input object	Description
`uri`	This is the Outbound URL.
`method`	This is the HTTP method.
`headers`	(Optional) Set the headers where the JSON node is the key header name, and the value is a string or array of header value(s).
`parameters`	(Optional) This is a collection of path or query parameters, or both.
`body`	(Optional) This is the outbound request payload.
`skipPolicy`	(Optional) This is a boolean value, set by default to `FALSE`. Set it to `TRUE` if you want to skip the security policy.
`requestMediaType`	(Optional) Use this to override the default connectivity::rest behavior to process request media like structured (JSON), binary, multipart, and form-urlencoded.
`responseMediaType`	(Optional) Use this to override the default connectivity::rest behavior to process response media like structured (JSON), binary, multipart, and form-urlencoded.

This table describes JSON node output format for this function.

Response/output object for connectivity::rest	Description
`headers`	This is the response header, where the JSON node is the key header name, and the value is a string or array of header values.
`status`	This is the response status. It is an integer.
`body`	This is the outbound response payload.

Sample code:

```
"functions": [
  {
   "name": "createTaskListFunction",
    "type": "custom",
    "operation": "connectivity::rest"
  }
],
"states":[
 {
  ...
  "actions":[
  {
```

```
  "functionRef": {
  "refName": "createTaskListFunction",
   "arguments": {
     "uri": <String/JQ expression>,
     "method": <String/JQ expression>,
     "headers": <Object/JQ expression>",
     "parameters": "<Object/JQ expression>,
     "body": <Object/String/JQ>",
     "skipPolicy": boolean

          }
        },
 "actionDataFilter": {
  "results": "${ {body: .body, headers: .headers, status: .status} }",
     ..
   }
  }
  ],
   ....
  }
```

`connectivity::log`

This function logs a message at the specified level.

This table describes the input arguments for this function.

Input argument	Description
`message`	This is the message to log.
`level`	This is the level at which to log the message. The value can be `SEVERE`, `WARN`, `INFO`, `DEBUG`, or `TRACE`.

> **Note:**
>
> DEBUG mode publishes the message to the activity stream.

Sample code:

```
"functions": [
  {
   "name": "logFunction",
    "type": "custom",
    "operation": "connectivity::log"
  }
],
"states":[
 {
  ...
  "actions":[
  {
  "functionRef": {
```

```
  "refName": "logFunction",
   "arguments": {
     "message": <String/JQ expression>
      }
   }
  }
  ],
   ....
  }
```

`connectivity::avroSchemaToJsonSchema`

This function converts the AVRO schema to JSON schema.

This table describes the input arguments for this function.

Input	Example Value
avroSchema	"{\"type\":\"record\",\"name\":\"Person\",\"fields\": [{\"name\":\"name\",\"type\":\"string\"},{\"name\":\"age\",\"type\":\"int\"},{\"name\":\"email\",\"type\":\"string\"}]}"

Sample output:

```
{
  "type" : "object",
  "required" : [ "name", "age", "email" ],
  "properties" : {
    "name" : {
      "type" : "string"
    },
    "age" : {
      "type" : "integer"
    },
    "email" : {
      "type" : "string"
    }
  }
}
```

Sample code:

```
"functions": [

{
          "name": "dynamicAvroSchema",
          "type": "custom",
          "operation": "connectivity::avroSchemaToJsonSchema"
        }

],
```

```
"states": [

 {

...

"actions": [

{
               "functionRef": {
               "refName":"dynamicAvroSchema",
               "arguments": {
                 "avroSchema": <String/JQ expression>              }
               },
               "actionDataFilter": {
                 "toStateData":

<JQ expression>

               }
             }

]

}
```

connectivity::avroEncode

This function converts the JSON data corresponding to the AVRO schema to AVRO binary encoded data.

This table describes the input arguments.

Key	Description	Example Value
avroSchema	AVRO schema (string format).	"{\"name\":\"Jane Smith\",\"age\":25,\"email\":\"janesmith@example.com\"}"
avroJsonData	JSON data (string format) corresponding to AVRO schema.	"{\"type\":\"record\",\"name\":\"Person\",\"fields\": [{\"name\":\"name\",\"type\":\"string\"}, {\"name\":\"age\",\"type\":\"int\"}, {\"name\":\"email\",\"type\":\"string\"}]}"

Here is the sample output:

```
Jane Smith2*janesmith@example.com
```

Sample code:

```
"functions": [
        {
          "name": "avroEncodeFunc",
          "operation": "connectivity::avroEncode",
          "type": "custom"
        }
      ],

"states": [
        {

"actions": [
            {
              "functionRef": {
                "refName":"avroEncodeFunc",
                "arguments": {
                  "avroSchema": "<String/JQ expression>
                  "avroJsonData": <String/JQ expression>
                }
                },
                "actionDataFilter": {
                  "toStateData":

<JQ expression>

                }
            }

}
      ]
```

`connectivity::avroDecode`

This function decodes the AVRO binary encoded data.

This table describes the input argument.

Key	Description	Example Value
`avroSchema`	AVRO schema (string format), that was converted to JSON schema.	`"{\"type\":\"record\",\"name\":\"Person\",\"fields\":[{\"name\":\"name\",\"type\":\"string\"},{\"name\":\"age\",\"type\":\"int\"},{\"name\":\"email\",\"type\":\"string\"}]}"`
`avroEncodedData`	AVRO binary encoded data (text).	`Jane Smith2*janesmith@example.com`

Sample output:

```
{"name":"Jane Smith","age":25,"email":"janesmith@example.com"}
```

Sample code:

```
"functions": [
        {
          "name": "avroDecodeFunc",
          "operation": "connectivity::avroDecode",
          "type": "custom"
        }
      ],

"states": [
        {

"actions": [
            {
              "functionRef": {
                "refName":"avroDecodeFunc",
                "arguments": {
                  "avroSchema": "<String/JQ expression>"
                  "avroEncodedData": "<String/JQ expression>"
                }
                },
                "actionDataFilter": {
                  "toStateData":

<JQ expression>

                }
            }

}
      ]
```

`connectivity::jsonPayloadToXMLPayload`

This function converts a JSON payload to an XML payload. You can use this function in two ways, that is, with or without the `targetNamespace` attribute. Using this attribute, you can add a target namespace to the output xml payload.

The input arguments for both the scenarios are described below.

Table 14-1 Without the targetNamespace argument

Key	Description	Example Input Value
jsonPayload	The JSON payload (in String format) to be converted to the XML payload.	"{ \"StorageServiceProperties\" : { \"Logging\" : { \"Version\" : \"1.0\", \"Read\" : \"false\", \"Write\" : \"true\", \"Delete\" : \"false\", \"RetentionPolicy\" : { \"Enabled\" : \"false\" } }, \"HourMetrics\" : { \"Version\" : \"1.0\", \"Enabled\" : \"true\", \"IncludeAPIs\" : \"true\", \"RetentionPolicy\" : { \"Enabled\" : \"true\", \"Days\" : \"7\" } }, \"MinuteMetrics\" : { \"Version\" : \"1.0\", \"Enabled\" : \"false\", \"RetentionPolicy\" : { \"Enabled\" : \"false\" } }, \"Cors\" : \"\", \"DeleteRetentionPolicy\" : { \"Enabled\" : \"true\", \"Days\" : \"7\", \"AllowPermanentDelete\" : \"false\" }, \"StaticWebsite\" : { \"Enabled\" : \"false\" } }}"

Table 14-2 With the targetNamespace argument

Key	Description	Example Value
jsonPayload	The JSON payload (in String format) to be converted to the XML payload.	{ "AccelerateConfiguration": { "Status": "Suspended" } }
targetNamespace	The target namespace to add to the output xml payload.	"http://s3.amazonaws.com/doc/2006-03-01/"

Sample output without the targetNamespace property:

```
<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"no\"?
><StorageServiceProperties><Logging><Version>1.0</Version><Read>false</
Read><Write>true</Write><Delete>false</
Delete><RetentionPolicy><Enabled>false</Enabled></RetentionPolicy></
Logging><HourMetrics><Version>1.0</Version><Enabled>true</
Enabled><IncludeAPIs>true</IncludeAPIs><RetentionPolicy><Enabled>true</
Enabled><Days>7</Days></RetentionPolicy></
HourMetrics><MinuteMetrics><Version>1.0</Version><Enabled>false</
Enabled><RetentionPolicy><Enabled>false</Enabled></RetentionPolicy></
MinuteMetrics><Cors/><DeleteRetentionPolicy><Enabled>true</Enabled><Days>7</
Days><AllowPermanentDelete>false</AllowPermanentDelete></
DeleteRetentionPolicy><StaticWebsite><Enabled>false</Enabled></
StaticWebsite></StorageServiceProperties>
```

Sample output with the targetNamespace property:

```
<?xml version="1.0" encoding="UTF-8"?>
<AccelerateConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Status>Suspended</Status>
</AccelerateConfiguration>
```

Sample code without the targetNamespace property:

```
"functions": [

{
          "name": "dynamicXMLPayload",
          "type": "custom",
          "operation": "connectivity::jsonPayloadToXMLPayload"
        }

],

"states": [

 {

...

"actions": [

{
              "functionRef": {
              "refName":"dynamicXMLPayload",
              "arguments": {
                "jsonPayload": <String/JQ expression>

}
              },
              "actionDataFilter": {
                "toStateData":

<JQ expression>

              }
            }

]

}
```

Sample code with the targetNamespace property:

```
"functions": [

{
          "name": "dynamicXMLPayload",
```

```
          "type": "custom",
          "operation": "connectivity::jsonPayloadToXMLPayload"
        }

],

"states": [

 {

...

"actions": [

{
              "functionRef": {
              "refName":"dynamicXMLPayload",
              "arguments": {
                "jsonPayload": <String/JQ expression>,

  "targetNamespace": "${.configuration.TargetNamespace}"

}
              },
              "actionDataFilter": {
                "toStateData":

<JQ expression>

              }
            }

]

}
```

connectivity::xmlPayloadToJsonSchema

This function converts an XML payload to a JSON schema. You can provide the XML payload with attributes on the Adapter Endpoint Configuration Wizard, which will be converted to a JSON schema. You can also use this to create a JSON trigger at runtime.

The input arguments for the function are described below.

Key	Description	Example Value
XMLPayload	The XML payload to be converted to the JSON schema.	<?xml version = '1.0' encoding = 'UTF-8'?> <ListBucketResult> <Name firstname="chai">chaitanya</Name> <Prefix/> <Marker/> <MaxKeys>0</ MaxKeys> <IsTruncated>false</ IsTruncated> </ListBucketResult>

Sample output:


```
{
   "type": "object",
   "properties": {
       "ListBucketResult": {
           "type": "object",
           "properties": {
                "Name": {
                   "type": "object",
                   "properties": {
                       "firstname": {
                           "type": "string",
                           "xml": {
                               "attribute": true
                           }
                       },
                       "xml": {
                           "type": "string"
                       }
                   }
               },
               "Prefix": {
                   "type": "string"
               },
               "Marker": {
                   "type": "string"
               },
               "MaxKeys": {
```

```
                    "type": "number"

                },

                "IsTruncated": {

                    "type": "boolean"

                }

            }

        }

    }

}
```

Sample code:

```
"functions": [

{
          "name": "dynamicJSONSchema",
          "type": "custom",
          "operation": "connectivity::xmlPayloadToJsonSchema"
        }

],

"states": [

 {

...

"actions": [

{
              "functionRef": {
              "refName":"dynamicJSONSchema",
              "arguments": {
                "XMLPayload": <String/JQ expression>              }
              },
              "actionDataFilter": {
                "toStateData":

<JQ expression>

              }
            }

]
```

`connectivity::sampleXmlToJson`

This function converts a sample XML payload to a JSON schema in designtime.

The input arguments for the function are described below.

Key	Description	Example Value
XMLPayload	The XML payload (in String format) to be converted to the JSON payload in designtime.	<?xml version="1.0" encoding="UTF-8"?> <ListAllMyBucketsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/"> <Owner> <ID>77fd6d45133de29e6f96bf93a519ffcff023b726995f38e8100e7d95ff8fe501</ID> </Owner> <Buckets> <Bucket> <Name>ap-bucket-bcone</Name> <CreationDate>2024-01-25T05:44:12.000Z</CreationDate> </Bucket> <Bucket> <Name>ap-bucket-bconevi</Name> <CreationDate>2024-01-18T08:40:28.000Z</CreationDate> </Bucket> <Bucket> <Name>ap-bucket-bconevj</Name> <CreationDate>2024-01-18T08:28:40.000Z</CreationDate> </Bucket> <Bucket> <Name>ap-bucket-bconevo</Name> <CreationDate>2024-01-18T08:40:46.000Z</CreationDate> </Bucket> <Bucket> <Name>bconeb</Name> <CreationDate>2024-01-08T14:48:21.000Z</CreationDate> </Bucket> <Bucket> <Name>bconebkt1</Name> <CreationDate>2024-01-02T11:54:28.000Z</CreationDate> </Bucket> <Bucket> <Name>bconebuck16thjan</Name> <CreationDate>2024-01-16T09:15:53.000Z</CreationDate> </Bucket> <Bucket> <Name>bconebuck18thjan</Name> <CreationDate>2024-01-18T07:46:16.000Z</CreationDate> </Bucket> <Bucket> <Name>bconebucket</Name> <CreationDate>2024-02-14T06:57:49.000Z</CreationDate> </Bucket> <Bucket> <Name>bconegbbucketab</Name> <CreationDate>2024-02-21T11:40:35.000Z</CreationDate> </Bucket> <Bucket> <Name>bucket12feb</Name> <CreationDate>2024-02-12T10:21:33.000Z</CreationDate> </Bucket> <Bucket> <Name>bucket23rdjan</Name> <CreationDate>2024-01-23T05:10:29.000Z</CreationDate> </Bucket> <Bucket> <Name>bucket30thjan</Name> <CreationDate>2024-01-30T05:39:54.000Z</CreationDate> </Bucket> <Bucket> <Name>bucket31stjan</Name> <CreationDate>2024-01-31T05:37:53.0

Key	Description	Example Value
		00Z</CreationDate> </Bucket> <Bucket> <Name>demobucket2vj</Name> <CreationDate>2024-02-13T08:21:30.000Z</CreationDate> </Bucket> <Bucket> <Name>demotest88</Name> <CreationDate>2024-02-13T12:05:51.000Z</CreationDate> </Bucket> <Bucket> <Name>demotest89</Name> <CreationDate>2024-02-23T07:49:19.000Z</CreationDate> </Bucket> <Bucket> <Name>east-2-11</Name> <CreationDate>2024-01-24T06:43:23.000Z</CreationDate> </Bucket> <Bucket> <Name>mulebucket1abc</Name> <CreationDate>2024-02-12T08:08:27.000Z</CreationDate> </Bucket> <Bucket> <Name>oicawsbucket</Name> <CreationDate>2024-02-15T12:19:06.000Z</CreationDate> </Bucket> <Bucket> <Name>oicbucketaws6thfeb</Name> <CreationDate>2024-02-09T11:32:52.000Z</CreationDate> </Bucket> <Bucket> <Name>sstestbucket10</Name> <CreationDate>2024-01-30T04:58:40.000Z</CreationDate> </Bucket> <Bucket> <Name>sstestbucket11</Name> <CreationDate>2024-01-30T05:16:19.000Z</CreationDate> </Bucket> <Bucket> <Name>sstestbucket13</Name> <CreationDate>2024-01-30T05:27:11.000Z</CreationDate> </Bucket> <Bucket> <Name>test2vijaya</Name> <CreationDate>2024-02-16T07:23:21.000Z</CreationDate> </Bucket> <Bucket> <Name>vijaya2000</Name> <CreationDate>2024-02-19T10:42:51.000Z</CreationDate> </Bucket> </Buckets> </ListAllMyBucketsResult>

Sample output:

```
 {
    "ListAllMyBucketsResult": {
        "Owner": {
            "ID":
"77fd6d45133de29e6f96bf93a519ffcff023b726995f38e8100e7d95ff8fe501"
        },
        "Buckets": {
            "Bucket": [
```

```
{
    "Name": "ap-bucket-bcone",
    "CreationDate": "2024-01-25T05:44:12.000Z"
},
{
    "Name": "ap-bucket-bconevi",
    "CreationDate": "2024-01-18T08:40:28.000Z"
},
{
    "Name": "ap-bucket-bconevj",
    "CreationDate": "2024-01-18T08:28:40.000Z"
},
{
    "Name": "ap-bucket-bconevo",
    "CreationDate": "2024-01-18T08:40:46.000Z"
},
{
    "Name": "bconeb",
    "CreationDate": "2024-01-08T14:48:21.000Z"
},
{
    "Name": "bconebkt1",
    "CreationDate": "2024-01-02T11:54:28.000Z"
},
{
    "Name": "bconebuck16thjan",
    "CreationDate": "2024-01-16T09:15:53.000Z"
},
{
    "Name": "bconebuck18thjan",
    "CreationDate": "2024-01-18T07:46:16.000Z"
},
{
    "Name": "bconebucket",
    "CreationDate": "2024-02-14T06:57:49.000Z"
},
{
    "Name": "bconegbbucketab",
    "CreationDate": "2024-02-21T11:40:35.000Z"
},
{
    "Name": "bucket12feb",
    "CreationDate": "2024-02-12T10:21:33.000Z"
},
{
    "Name": "bucket23rdjan",
    "CreationDate": "2024-01-23T05:10:29.000Z"
},
{
    "Name": "bucket30thjan",
    "CreationDate": "2024-01-30T05:39:54.000Z"
},
{
    "Name": "bucket31stjan",
    "CreationDate": "2024-01-31T05:37:53.000Z"
},
```

```
                    {
                        "Name": "demobucket2vj",
                        "CreationDate": "2024-02-13T08:21:30.000Z"
                    },
                    {
                        "Name": "demotest88",
                        "CreationDate": "2024-02-13T12:05:51.000Z"
                    },
                    {
                        "Name": "demotest89",
                        "CreationDate": "2024-02-21T11:43:54.000Z"
                    },
                    {
                        "Name": "east-2-11",
                        "CreationDate": "2024-01-24T06:43:23.000Z"
                    },
                    {
                        "Name": "mulebucket1abc",
                        "CreationDate": "2024-02-12T08:08:27.000Z"
                    },
                    {
                        "Name": "oicawsbucket",
                        "CreationDate": "2024-02-15T12:19:06.000Z"
                    },
                    {
                        "Name": "oicbucketaws6thfeb",
                        "CreationDate": "2024-02-09T11:32:52.000Z"
                    },
                    {
                        "Name": "sstestbucket10",
                        "CreationDate": "2024-01-30T04:58:40.000Z"
                    },
                    {
                        "Name": "sstestbucket11",
                        "CreationDate": "2024-01-30T05:16:19.000Z"
                    },
                    {
                        "Name": "sstestbucket13",
                        "CreationDate": "2024-01-30T05:27:11.000Z"
                    },
                    {
                        "Name": "test2vijaya",
                        "CreationDate": "2024-02-16T07:23:21.000Z"
                    },
                    {
                        "Name": "vijaya2000",
                        "CreationDate": "2024-02-19T10:42:51.000Z"
                    }
                ]
            }
        }
    }
```

Sample code:

```
"functions": [

{
            "name": "samplexmlToJsonFunc",
            "operation": "connectivity::sampleXmlToJson",
            "type": "custom"
          }

],

"states": [

 {

...

"actions": [

{
                "name": "SampleXMLToJSONConverstion",
                "functionRef": {
                  "refName": "samplexmlToJsonFunc",
                  "arguments": {
                    "XMLPayload": <String/JQ expression>
                  }
                },
                "actionDataFilter": {
                    "toStateData": <String/JQ expression>
                }
              }

]
```

`connectivity::updateSecurityProperty`

This function updates or adds a security property in the secret store.

> **Note:**
>
> This function is available only in the subscription register or unregister flow.

The input arguments for the function are described below. The output will be a Boolean true node if successful.

Key	Type	Description
`key`	Text node	The property key.
`value`	Text node	The property value.

Key	Type	Description
`scope`	Text node (Optional)	The scope for the composite policy. The valid values are: • inbound • outbound

> **Note:**
>
> If the property is not present, check if it is present in the inbound scope. If present, it adds with inbound scope else adds as outbound scope.

Sample code: Unscoped property update

```
"functions": [
    {
        "name": "updateSecurityPropertyFn",
        "type": "custom",
        "operation": "connectivity::updateSecurityProperty"
    }
...

{
    "functionRef": {
        "refName": "updateSecurityPropertyFn",
        "arguments": {
            "name": "secret",
            "value": "${.configuration.value}"
        }
    },
    "actionDataFilter": {
        "toStateData": "${ .output }"
    }
}
//.output is set to Boolean(True or false)
```

Sample code: Scoped property update

```
"functions": [
    {
        "name": "updateSecurityPropertyFn",
```

```
        "type": "custom",
        "operation": "connectivity::updateSecurityProperty"
    }
]
....

{
    "functionRef": {
        "refName": "updateSecurityPropertyFn",
        "arguments": {
            "name": "signKey",
            "value": "${.configuration.value}",
            "scope": "inbound"
        }
    },
    "actionDataFilter": {
        "toStateData": "${ .output }"
    }
}
//.output is set to Boolean(True or false)
```

`connectivity::getSecurityProperty`

This function gets the security property value.

> **Note:**
>
> This function is available only in the subscription register or unregister flow.

The input arguments for the function are described below. The output will be a text value.

Key	Type	Description
`key`	Text node	The property key.
`scope`	Text node (Optional)	The scope for the composite policy. The valid values are: • inbound • outbound **Note:** If the property is not present, check if it is present in the outbound scope first, and then in the inbound scope.

Sample code: Fn declaration

```
{
  "name": "getSecurityPropertyFn",
  "type": "custom",
  "operation": "connectivity::getSecurityProperty"
}
```

Sample code: Unscoped Get

```
{
  "functionRef": {
    "refName": "getSecurityPropertyFn",
    "arguments": {
      "name": "secret"
    }
  },
  "actionDataFilter": {
    "toStateData": "${ .output }"
  }
}
...

{
  "functionRef": {
    "refName": "getSecurityPropertyFn",
    "arguments": {
      "name": "secret"
    }
  },
  "actionDataFilter": {
    "toStateData": "${ .output }"
  }
}
//.output is textNode with property value
```

Sample code: Scoped Get

```
{
  "functionRef": {
    "refName": "getSecurityPropertyFn",
    "arguments": {
      "name": "secret"
    }
  },
  "actionDataFilter": {
    "toStateData": "${ .output }"
  }
}
...
```

```
{
  "functionRef": {
    "refName": "getSecurityPropertyFn",
    "arguments": {
      "name": "secret",
      "scope": "inbound"
    }
  },
  "actionDataFilter": {
    "toStateData": "${ .output }"
  }
}
//.output is textNode with property value
```

connectivity::getPersistentState

This function gets a bag of persistent properties.

> **Note:**
>
> This function is allowed in polling init, polling, success, and error flows. To prevent multiple storage calls, it returns a bag of properties.

The function automatically derives the key from the flow context. The output is a JSON node, which is a bag of properties that you set.

Sample code:

```
"functions": [
  {
    "name": "getRTSessionPropsFn",
    "type": "custom",
    "operation": "connectivity::getPersistentState"
  }
],
"states":[
  {
    "name": "startState",
    "type": "operation",
    "actions": [
      {
        "functionRef": {
          "refName": "getRTSessionPropsFn"
        },
        "actionDataFilter": {
          "toStateData": "${ .output }"
        }
      }
    ],
```

connectivity::setPersistentState

This function sets a bag of persistent properties.

> **Note:**
>
> This function is restricted to polling init, success, and error flows. Only one usage of the function is allowed.

The input arguments for the function are described below. It automatically derives the key from the flow context.

Input Argument	Type	Description
`properties`	JSON node	Represents a bag of properties.

The output is a Boolean node indicating success or failure.

Sample code:

```
"functions": [
  {
    "name": "setRTSessionPropsFn",
    "type": "custom",
    "operation": "connectivity::setPersistentState"
  }
],
"states":[
  {
    "name": "startState",
    "type": "operation",
    "actions": [
      {
        "functionRef": {
          "refName": "setRTSessionPropsFn",
          "arguments": {
           "properties": "${.input}"
          }
        },
        "actionDataFilter": {
          "toStateData": "${ .output }"
        }
      }
```

Supported Custom jq Functions

The Rapid Adapter Builder platform supports the following custom jq functions:

- connectivity::to_entries
- connectivity::from_entries
- connectivity::json_data_to_json_schema
- connectivity::swagger_schema

`connectivity::to_entries`

This custom `to_entries` function allows custom keys and value labels.

This table describes input and output arguments for this function.

Input	Output	Examples

• JSON object converted to array.
• KeyName for each key in array.
• KeyName for each value in array.
	JSON array of entries.	Input JSON:
		`{ "potato": true, "broccoli": false }`
		Expression in adapter definition document:
		`{ "arguments": { "body": "${ {vegetableFeedback : connectivity::to_entries(.input; .configuration.key; \"like\") } }" } }`
		Output in JSON format where `.configuration.key` resolves to `veggieName`:
		`{ ... "body" : { "vegetableFeedback" : [ { "veggieName" : "potato", "like" : true }, { "veggieName" : "broccoli", "like" : false } ] } }`

`connectivity::from_entries`

This custom `from_entries` function allows custom keys and value labels.

This table describes input and output arguments for this function.

Input	Output	Examples

• JSON object converted to object.
• KeyName for each key in array.
• KeyName for each value in array.
	JSON object	Input JSON:
		`{ "vegetables": [ { "veggieName": "potato", "like": true }, { "veggieName": "broccoli", "like": false } ] }`
		Expression in adapter definition document:
		`{ "arguments": { "body": "${ {vegetableFeedback : connectivity::from_entries(.input.vegetables; .configuration.key; \"like\") } }" } }`
		Output in JSON format where `.configuration.key` resolves to `veggieName`:
		`{ ... "body" : { "vegetableFeedback" : { "potato" : true, "broccoli" : false } } }`

`connectivity::json_data_to_json_schema`

This function converts JSON data into JSON schema.

This table describes the input and output arguments.

Input	Output	Examples

Any JSON object or array.	JSON schema as JSON Object type.	Input JSON
		`{ "vegetables": [ { "veggieName": "potato", "like": true }, { "veggieName": "broccoli", "like": false } ] }`
		Expression in adapter definition document:
		`{ "arguments": { "body": "${ {vegetableFeedback : connectivity::json_data_to_json_schema(.input.vegetables) } }" } }`
		Output JSON
		`{ "body": { "vegetableFeedback": { "type": "array", "items": { "type": "object", "properties": { "veggieName": { "type": "string" }, "like": { "type": "boolean" } } } } } }`

`connectivity::swagger_schema`

This function parses the swagger document and extracts the request and response json schema.

This table describes the input and output arguments.

Input	Output	Examples

• The Swagger document in JSON format.
• The path Id in the Swagger document.
• The HTTP method under the path of the second parameter.
• The response or request in the form of a enum value.
	JSON schema	Input JSON
		`{ "swagger": "2.0", "info": { "description": "This is a sample server Petstore server. You can find out more about Swagger at [http://swagger.io](http://swagger.io) or on [irc.freenode.net, #swagger](http://swagger.io/irc/). For this sample, you can use the api key `special-key` to test the authorization filters.", "version": "1.0.6", "title": "Swagger Petstore", ... }, "host": "petstore.swagger.io", "basePath": "/v2", ... }`
		Expression in adapter definition document:
		`{ "name": "parseSchema", "type": "expression", "operation": "connectivity::swagger_schema(.swagger; .configuration.path; .configuration.method; \"request\")" }`
		Output JSON
		`{ "type": "object", "properties": { "body": { "type": "object", "properties": { "id": { "type": "integer", "format": "int64" }, ... } } } }`

Supported Custom jq Functions for Encoding and Decoding

The Rapid Adapter Builder platform supports custom jq functions.

The following list of supported custom jq functions enable the developer to use encoding and decoding mechanisms for inbound security policies:

- connectivity::base64
- connectivity::hex
- connectivity::base64URLDecode
- connectivity::stringToBinary
- connectivity::binaryToString

- connectivity::hexDecode
- connectivity::base64Decode

connectivity::base64

Base64 Encode Binary/Text Content

Input	Output	Expression in Adapter Definition Document
A JSON node that can be either binary or text.	Hex encoded string as a text node.	To compute base64 encoded HMAC digest: `{ "arguments": { "body": "$ {connectivity::hex(connectivity::hmac(\"HMACSHA256\"; .securityProperties.signKey; .request.url + (.request.body| tostring))) }" } }`

connectivity::hex

Hex Encode Binary/Text Content

Input	Output	Expression in Adapter Definition Document
A JSON node that can be either binary or text.	Base64 encoded string as text node.	To compute Hex encoded HMAC digest: `{ "arguments": { "body": "$ {connectivity::base64(connectivity::hmac(\"HMACSHA256\"; .securityProperties.signKey; .request.url + (.request.body| tostring))) }" } }`

connectivity::base64URLDecode

base64URL decode the value

Input	Output	Expression in Adapter Definition Document
JSON text node	Binary node with decode byte array.	`{ "arguments": { "body": "$ {connectivity::rsa(\"SHA256withRSA\"; .securityProperties.signKey; .request.body; connectivity::base64URLDecode(.request.headers.signature))}" } }`

`connectivity::stringToBinary`

Converts string to binary (byte string) node.

Input	Output	Expression in Adapter Definition Document
Consists of: • Key • Text node • String to be converted	Binary node	`{ "arguments": { "body": "$ {connectivity::stringToBinary(.input.message)}" } }`

`connectivity::binaryToString`

Converts binary to string

Input	Output	Expression in Adapter Definition Document
Consists of: • Key • Binary node • Binary to be converted	Text node	`{ "arguments": { "body": "$ { connectivity::binaryToString(connectivity::stringToBinary(.input.message))}" } }`

`connectivity::hexDecode`

Hex decode string or byte array

Input	Output	Expression in Adapter Definition Document
Consists of: • Key • Binary node • Text or binary to be converted	Binary node	`{ "name": "decodedSignature", "type": "expression", "operation": "connectivity::hexDecode(.request.headers.digest)" } ,`

`connectivity::base64Decode`

Hex decode string or byte array

Input	Output	Expression in Adapter Definition Document
Consists of: • Key • Binary node • Text or binary to be converted	Binary node	`{ "name": "decodedSignature", "type": "expression", "operation": "connectivity::base64Decode(.request.headers.digest)" },`

Flows Properties and Syntax for Dot (.) Notation

In the `flows` section of the adapter definition document, you can access context using the dot(.) notation. The adapter definition document exposes flows for driving both design time and runtime.

The following sections describe the dot (.) notations that the adapter developer can use:

- Dot (.) Notation Syntax for Validation Request Condition in Trigger
- Dot (.) Notation Syntax for Validation Request-Response Generation Flow
- Dot (.) Notation Syntax for Trigger Subscription Register and Deregister Flows
- Dot (.) Notation Syntax for Action Runtime
- Dot (.) Notation Syntax to Authenticate Inbound Signature Policies
- Dot (.) Notation Syntax for Request Transformation in Trigger Runtime
- Dot (.) Notation Syntax for Generate Schema and Configuration Flows
- Dot (.) Notation Syntax for the Init Flow
- Dot (.) Notation Syntax for the Poll Flow
- Dot (.) Notation Syntax for the Success Flow

Dot (.) Notation Syntax for Validation Request Condition in Trigger

This topic describes how to validate trigger requests with a condition using the Dot (.) notation syntax.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`

Context object	Description	Dot (.) notation	Example

Request	The contents of the inbound request are:
	• method
	Uppercase HTTP method.
	• uri
	Request URL
	• headers
	Map of headers. Ensure that the header key is in lower case. Use text for a single value and use an array if there are multiple values.
	• query
	Map of query parameters. Use text for a single value and use an array if there are multiple values.
	• body
	Request body as String
		.request	`.request.body| fromjson`
			For a single value:
			`.request.headers.token .request.query.\"x-custom-id\"`
			For an array:
			`.request.headers.\"x-custom-token\"[0]`
			`.request.query.\"x-custom-id\"[0]`

Output	Output of a flow must be a boolean node.	.output	`.output`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for Validation Request-Response Generation Flow

This topic describes how to validate requests in response flow using the Dot (.) notation syntax.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection Properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`

Request	The contents of the inbound request are:
	• method
	Uppercase HTTP method.
	• uri
	Request URL
	• headers
	Map of headers. Ensure that the header key is in lower case. Use text for a single value and use an array if there are multiple values.
	• query
	Map of query parameters. Use text for a single value and use an array if there are multiple values.
	• body
	Request body as string.
		.request	`.request.body|fromjson`
			For a single value:
			`.request.headers.token .request.query.\"x-custom-id\"`
			For an array:
			`.request.headers.\"x-custom-token\"[0]`
			`.request.query.\"x-custom-id\"[0]`

Output	The output represents the response. Set the flow output in this context object. • status HTTP status code(int) • body Response body. If it is not a JSON data, then set it to single text node. • headers Map of headers. Use text for a single value and use an array if there are multiple values.	.output	`.output.status`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.

- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for Trigger Subscription Register and Deregister Flows

This topic covers the Trigger Subscription Register and Deregister flows.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Integration properties	The `INTEGRATION_FLOW_URL` property that identifies the integration.	.integrationProperties	`.integrationProperties.INTEGRATION_FLOW_URL`
Self	Reference to the adapter definition document.	.self	`.self.triggers`
Configuration	Additional configuration defined as part of trigger, and populated or selected in the design time wizard.	.configuration	`.configuration.compartmentId`
Flow output	Context propagation does not expect any output. You can set the flow output to any empty output object.	.output	

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for Action Runtime

This topic describes how to use the Dot (.) notation syntax for runtime in a flow definition.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Configuration	Additional configuration defined as part of action and populated or selected in design time wizard.	.configuration	`.configuration.compartmentId`
Connection properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Flow input	Input data for the flows. Represents the object complying to effective input schema of action. The adapter developer can use this notation to access the transformed JSON payload that is received from orchestration.	.input	`.input.body`
Flow output	Output of flow. The adapter developer must set the JSON and ensure that it complies to the output schema of action. Stores the response that is passed to Oracle Integration orchestration.	.output	`.output`

Dot (.) Notation Syntax to Authenticate Inbound Signature Policies

This topic describes how to authenticate inbound signature policies with a condition using the Dot (.) notation syntax.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection Properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`

Context object	Description	Dot (.) notation	Example

Request	The contents of the input request are:
	• method
	Uppercase HTTP method.
	• url
	Request URL
	• headers
	Map of headers. Ensure that the header key is in lower case. Use text for a single value and use an array if there are multiple values.
	• query
	Map of query parameters. Use text for a single value and use an array if there are multiple values.
	• body
	Request body as string.
		.request	`.request.body| fromjson`
			For a single value:
			`.request.headers.token .request.query.\"x-custom-id\"`
			For an array:
			`.request.headers.\"x-custom-token\"[0]`
			`.request.query.\"x-custom-id\"[0]`

Security properties	Security properties for trigger as part of the configuration.	.securityProperties	`.securityProperties .signKey`
Integration properties	Property that identifies the integration. instanceUrl: Oracle Integration instance URL	.integrationProperties	`.integrationProperties. instanceUrl`
Output	Set the output in this context object. Set the output based on the contract of the property.	.output	`.output`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.

- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for Request Transformation in Trigger Runtime

This topic describes how to use the Dot (.) notation syntax to transform a request where a webhook payload is converted to Oracle Integration payload.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection Properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Flow input	Input data for the flow. The input data is a trigger request payload in the form a string.	.input	`.input| fromJson|.field1`
Configuration	The adapter developer can define additional configuration as part of trigger. This value is populated or selected in the design time wizard of Oracle Integration.	.configuration	`.configuration.compartmentId`
Temporary context	The adapter developer can add temporary context variables. These variables are valid for that particular instance of flow execution.	.<temporary context variable>	`intermediate`
Flow output	Set the output of flow in this context object. The output is a transformed message.	.output	`.output`

Dot (.) Notation Syntax for Generate Schema and Configuration Flows

This topic covers the Generate Schema and Configuration flows in triggers and actions.

This table describes the context objects and the associated dot (.) notation with examples.

Context object	Description	Dot (.) notation	Example
Connection Properties	Properties for connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Self	Reference to adapter definition document.	.self	`.self.actions`
Configuration	The adapter developer can define additional configuration as part of trigger. This value is populated or selected in the design time wizard of Oracle Integration.	.configuration	`.configuration.compartmentId`

Context object	Description	Dot (.) notation	Example
Temporary context	The adapter developer can add temporary context variables. These variables are valid for that particular instance of flow execution.	.<temporary context variable>	`.intermediate`
Flow output	Output of flow must bet set in this context object • Output schema • Output configuration values	.output	`.output`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for the Init Flow

This topic describes the Dot (.) Notation syntax for the Init flow in a polling trigger.

Context object	Description	Dot (.) notation	Example
Connection Properties	The properties for the connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Configuration	The additional configuration as part of the action, which is populated or selected in the design-time wizard of Oracle Integration.	.configuration	`.configuration.compartmentId`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.

- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for the Poll Flow

This topic describes the Dot (.) Notation syntax for the Poll flow in a polling trigger.

Context object	Description	Dot (.) notation	Example
Connection Properties	The properties for the connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Configuration	The additional configuration as part of the action, which is populated or selected in the design-time wizard of Oracle Integration.	.configuration	`.configuration.compartmentId`
Flow Output	The output of the flow. You must set the JSON to comply with the effective output schema of the polling trigger. This object stores the response that is passed to the Oracle Integration orchestration.	.output	`.output`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Dot (.) Notation Syntax for the Success Flow

This topic describes the Dot (.) Notation syntax for the Success flow in a polling trigger.

Context object	Description	Dot (.) notation	Example
Connection Properties	The properties for the connection configuration.	.connectionProperties	`.connectionProperties.baseUrl`
Configuration	The additional configuration as part of the action, which is populated or selected in the design-time wizard of Oracle Integration.	.configuration	`.configuration.compartmentId`
Flow Input	Set the output of the poll event here if available, else it's an empty JSON.	.input	`.input.keyid`

Related Topics

- CNCF Functions Support
  The Rapid Adapter Builder platform supports expressions from the Cloud Native Computing Foundation (CNCF) functions.
- Supported Custom CNCF Serverless Workflow
  The Rapid Adapter Builder supports a subset of the constructs that are supported in the Serverless Workflow specification of the Cloud Native Computing Foundation (CNCF) project.
- Supported Custom jq Functions for Encoding and Decoding
  The Rapid Adapter Builder platform supports custom jq functions.
- Supported jq Features
  The Rapid Adapter Builder platform supports a subset of jq features, as specified in this topic.
- Supported Custom jq Functions
  The Rapid Adapter Builder platform supports the following custom jq functions:

Work with Flow Definitions

Specify details about each flow definition in the `flows` section of the adapter definition document.

What Do You Want to Do?

Task	More information
Implement a flow to test a connection definition in the adapter definition document.	**Everyone completes this task:** Implement the Test-Connection Behavior Using Flows

Task	More information
Review other common modeling options for flow definitions.	**Complete these tasks as needed for your requirements:** • Switch States Conditionally in a Flow • Invoke a GET API • Invoke a POST API • Response Transformation in Post-processing • Send or Receive Form URL Encoded Content • Receive Binary Content • Send Multi-part Content • Send Binary Content • Override Default Content Handling in connectivity::rest • Pre and Post Processing Expressions and Multiple Outbound Requests • Fetch Design Time Configuration Values Using a Flow

Implement the Test-Connection Behavior Using Flows

This procedure describes how you can implement the Test Connection functionality using a flow.

1. Define a simple flow with a GET call to an idempotent test API.

   Here is a sample code:

```
"flows": {
   "testConnectionFlow" :{
      "id": "testConnectionFlow",
      "description": "testConnectionFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/
customers\"}",
                  "method": "GET"
                }
```

```
              },

            }
          ],
          "end": true
        }
      ]
    }
}
```

2. Add a `actionDataFilter` action that asserts a condition and set the `.output.success` to true to assert the status code.

   Here is a sample code:

```
"actionDataFilter": {
                "results": "${ { success: (.status==200),
message: .body.origin } }",
                "toStateData": "${ .output }"
              }
```

   Here is the full sample code that makes an outbound GET call to the test endpoint and uses state's boolean output value to indicate the result:

```
"flows": {
   "testConnectionFlow" :{
      "id": "testConnectionFlow",
      "description": "testConnectionFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/
customers\"}",
                  "method": "GET"
                }
              },
              "actionDataFilter": {
                "results": "${ { success: (.status==200),
message: .body.origin } }",
                "toStateData": "${ .output }"
```

```
                }
              }
            ],
            "end": true
          }
        ]
      }
}
```

3. Call the flow in the connection definition.

   Here is the sample code:

```
"connectionDefinition": {
  "connectionProperties": [
  ],
  "securityPolicies": [
    {
    "type": "managed",
    "policy": "BASIC_AUTH",
    "description": "Account credentials",
    "displayName": "Account credentials",
    "scope": "ACTION",
    "securityProperties": [
      {
        "name": "username",
        "hidden" : true,
        "default": "oracle"
      }
    ]
    }
  ],
  "test": "flow:testConnectionFlow"
```

Switch States Conditionally in a Flow

This procedure demonstrates how to implement a multi-state flow.

This sample code shows how to design a flow that checks for a resource and updates it. If the resource doesn't exist, the flow creates a new resource.

1. Open the adapter definition document in Visual Studio Code.
2. In the `flows` code section of the document, add code for a flow with GET call to a specific resource.

   In the following code sample, the `httpInvoke` function implements the GET call.

```
"flows": {
   "EventRegisterFlow": {
      "id": "EventRegisterFlow",
      "description": "EventRegisterFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
```

```
            "name": "httpInvoke",
            "type": "custom",
            "operation": "connectivity::rest"
          }
        ],
        "states": [
          {
            "name": "startState",
            "type": "operation",
            "actions": [
              {
                "name": "httpInvoke",
                "functionRef": {
                  "refName": "retriveListOfSubscriptionFunc",
                  "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks\"}",
                    "method": "GET",
                    "parameters": {
                      "owner": "${ .configuration.orgId }",
                      "repo": "${ .configuration.repositoryId }"
                    }
                  }
                },
                "actionDataFilter": {
                  "results": "${ .body}",
                  "toStateData": "${ .subscriptions }"
                }
              }
            ],
            "end": true
          }

      }
```

3. Add actions to the flow to evaluate if a specific resource exists.

   Sample code:

```
"functions": [
  ...
  {
    "name": "SubscriptionExistsFunction",
    "operation":
"[.integrationProperties.INTEGRATION_FLOW_URL==.subscriptions[].config.url]
|any",
    "type": "expression"
  },
  {
    "name": "getSubscriptionId",
    "type": "expression",
    "operation": "[{subscription:.subscriptions[],
INTEGRATION_FLOW_URL:.integrationProperties.INTEGRATION_FLOW_URL}] |
map(select(.subscription.config.url==.INTEGRATION_FLOW_URL))
[0].subscription.id | tostring"
  }
```

```
],
"states": [
  {
    "name": "startState",
    "type": "operation",
    "actions": [
      ...
      {
        "functionRef": "SubscriptionExistsFunction",
        "actionDataFilter": {
          "toStateData": "${ .subscriptionExists }"
        }
      },
      {
        "functionRef": "getSubscriptionId",
        "actionDataFilter": {
          "toStateData": "${ .subscriptionId }"
        }
      }
    ],
    end: true
  }
```

4. Add a switch state to transition to another state that can create or register a resource.

   Sample code:

```
"flows": {
   "EventRegisterFlow": {
      "id": "EventRegisterFlow",
      "description": "EventRegisterFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        ...
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            ...
          ],

          "transition": "SwitchBasedOnSubscription"

        },
        {
          "name": "SwitchBasedOnSubscription",
          "type": "switch",
          "dataConditions": [
            {
              "condition": "${ .subscriptionExists == true }",
              "transition": "UpdateRepositoryEndpoint"
            },
```

```
            {
              "condition": "${ .subscriptionExists == false }",
              "transition": "RegisterRepositoryEvent"
            }
          ],
          "defaultCondition": {
            "transition": "RegisterRepositoryEvent"
          }
        },

}
```

5. Add termination states that branch from a switch.

   The termination states contain the required actions.

   Sample code:

```
"flows": {
   "EventRegisterFlow": {
      "id": "EventRegisterFlow",
      "description": "EventRegisterFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
       ...
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            ...
          ],
          "transition": "SwitchBasedOnSubscription"
        },
        {
          "name": "SwitchBasedOnSubscription",
          "type": "switch",
          "dataConditions": [
            {
              "condition": "${ .subscriptionExists == true }",
              "transition": "UpdateRepositoryEndpoint"
            },
            {
              "condition": "${ .subscriptionExists == false }",
              "transition": "RegisterRepositoryEvent"
            }
          ],
          "defaultCondition": {
            "transition": "RegisterRepositoryEvent"
          }
        },
        {
          "name": "UpdateRepositoryEndpoint",
          "type": "operation",
```

```
          "actions": [
            {
              "name": "UpdateRepositoryEnpointAction",
              "functionRef": {
                "refName": "httpInvoke",
                "arguments": {
                  "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks/
{subscriptionId}\"}",
                  "parameters": {
                    "owner": "${ .configuration.orgId }",
                    "repo": "${ .configuration.repositoryId }"
                    "subscriptionId": "${ .subscriptionId }"
                  },
                  "method": "POST",
                  "body": "${ { \"name\": \"web\", \"active\": true,
\"events\": [.configuration.eventName] } }"
                }
              },
              "actionDataFilter": {
                "results": "${.body}",
                "toStateData": "${ .output }"
              },
              "condition": "${ .subscriptionExists==true }"
            }
          ],
          "end": true
        },
        {
          "name": "RegisterRepositoryEvent",
          "type": "operation",
          "actions": [
            {
               "name": "RepositoryEventRegisterAction",
               "functionRef": {
                "refName": "httpInvoke",
                "arguments": {
                  "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks\"}",
                  "parameters": {
                    "owner": "${ .configuration.orgId }",
                    "repo": "${ .configuration.repositoryId }"
                    "subscriptionId": "${ .subscriptionId }"
                  },
                  "method": "POST",
                  "body": "${ { \"name\": \"web\", \"active\": true,
\"events\": [.configuration.eventName], \"config\": { \"content_type\":
\"json\", \"secret\": .connectionProperties.\"trigger.secret\",
\"url\": .integrationProperties.INTEGRATION_FLOW_URL } } }"
                }
              },
              "actionDataFilter": {
                "results": "${.body}",
                "toStateData": "${ .output }"
              },
              "condition": "${ .subscriptionExists==false }"
```

```
            }
          ],
          "end": true
        }
      ]
    },
}
```

Complete sample code:

```
"flows": {
   "EventRegisterFlow": {
      "id": "EventRegisterFlow",
      "description": "EventRegisterFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "httpInvoke",
          "type": "custom",
          "operation": "connectivity::rest"
        },
        {
          "name": "SubscriptionExistsFunction",
          "operation":
"[.integrationProperties.INTEGRATION_FLOW_URL==.subscriptions[].config.url]
|any",
          "type": "expression"
        },
        {
          "name": "getSubscriptionId",
          "type": "expression",
          "operation": "[{subscription:.subscriptions[],
INTEGRATION_FLOW_URL:.integrationProperties.INTEGRATION_FLOW_URL}] |
map(select(.subscription.config.url==.INTEGRATION_FLOW_URL))
[0].subscription.id | tostring"
        }
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            {
              "name": "httpInvoke",
              "functionRef": {
                "refName": "retriveListOfSubscriptionFunc",
                "arguments": {
                  "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks\"}",
                  "method": "GET",
                  "parameters": {
                    "owner": "${ .configuration.orgId }",
                    "repo": "${ .configuration.repositoryId }"
```

```
                    }
                  }
                },
                "actionDataFilter": {
                  "results": "${ .body}",
                  "toStateData": "${ .subscriptions }"
                }
              },
              {
                "functionRef": "SubscriptionExistsFunction",
                "actionDataFilter": {
                  "toStateData": "${ .subscriptionExists }"
                }
              },
              {
                "functionRef": "getSubscriptionId",
                "actionDataFilter": {
                  "toStateData": "${ .subscriptionId }"
                }
              }
            ],
            "transition": "SwitchBasedOnSubscription"
          },
          {
            "name": "SwitchBasedOnSubscription",
            "type": "switch",
            "dataConditions": [
              {
                "condition": "${ .subscriptionExists == true }",
                "transition": "UpdateRepositoryEndpoint"
              },
              {
                "condition": "${ .subscriptionExists == false }",
                "transition": "RegisterRepositoryEvent"
              }
            ],
            "defaultCondition": {
              "transition": "RegisterRepositoryEvent"
            }
          },
          {
            "name": "UpdateRepositoryEndpoint",
            "type": "operation",
            "actions": [
              {
                "name": "UpdateRepositoryEnpointAction",
                "functionRef": {
                  "refName": "httpInvoke",
                  "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks/
{subscriptionId}\"}",
                    "parameters": {
                      "owner": "${ .configuration.orgId }",
                      "repo": "${ .configuration.repositoryId }"
                      "subscriptionId": "${ .subscriptionId }"
```

```
                    },
                    "method": "POST",
                    "body": "${ { \"name\": \"web\", \"active\": true,
\"events\": [.configuration.eventName] } }"
                  }
                },
                "actionDataFilter": {
                  "results": "${.body}",
                  "toStateData": "${ .output }"
                },
                "condition": "${ .subscriptionExists==true }"
              }
            ],
            "end": true
          },
          {
            "name": "RegisterRepositoryEvent",
            "type": "operation",
            "actions": [
              {
                 "name": "RepositoryEventRegisterAction",
                 "functionRef": {
                  "refName": "httpInvoke",
                  "arguments": {
                    "uri": "${\"https:/\"+\"/
\"+.connectionProperties.hostNameTrigger+\"/repos/{owner}/{repo}/hooks\"}",
                    "parameters": {
                      "owner": "${ .configuration.orgId }",
                      "repo": "${ .configuration.repositoryId }"
                      "subscriptionId": "${ .subscriptionId }"
                    },
                    "method": "POST",
                    "body": "${ { \"name\": \"web\", \"active\": true,
\"events\": [.configuration.eventName], \"config\": { \"content_type\":
\"json\", \"secret\": .connectionProperties.\"trigger.secret\",
\"url\": .integrationProperties.INTEGRATION_FLOW_URL } } }"
                  }
                },
                "actionDataFilter": {
                  "results": "${.body}",
                  "toStateData": "${ .output }"
                },
                "condition": "${ .subscriptionExists==false }"
              }
            ],
            "end": true
          }
        ]
      },
}
```

Invoke a GET API

This procedure shows how to create a flow to invoke a GET API.

The sample code implements a procedure that uses connection properties to retrieve API host information.

Prerequisites:

The action is defined in the schema. The header and parameters (query/path) are input to the action.

1. Open the adapter definition document in Visual Studio Code.
2. In the `flows` code section of the document, define the flow with a single state of `Operation` type.

   Sample code:

```
"flows": {
   "postOrdersFlow" : {
      "id": "postOrdersFlow",
      "description": "postOrdersFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [

      ],
      "states": [
        {
          "actions": [

          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

3. Define a function with a unique name, of type `custom`, and operation `connectivity::rest`.

   Sample code:

```
"functions": [
        {
          "name": "postOrdersFunction",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ]
```

4. Add an action to the state where:
   - The action refers to the function created in step 2.

- Arguments to function define the HTTP method, URI, parameters (template and query), and body.
- Response passes through, and the result is set to output.

Sample code:

```
{
              "functionRef": {
                "refName": "postOrdersFunction",
                "arguments": {
                 "uri": "${.connectionProperties.baseURL + \"/customers/
{customer_id}\"}",
                  "method": "GET",
                  "headers": "${ .input.headers }",
                  "parameters": "${ .input.parameters }"
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
                "toStateData": "${ .output }"
              }
            }


"flows": {
   "getCustomersCustomerIdFlow" : {
      "id": "getCustomersCustomerIdFlow",
      "description": "getCustomersCustomerIdFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [
        {
          "name": "getCustomersCustomerIdFunction",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ],
      "states": [
        {
          "actions": [
            {
              "functionRef": {
                "refName": "getCustomersCustomerIdFunction",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/customers/
{customer_id}\"}",
                  "method": "GET",
                  "headers": "${ .input.headers }",
                  "parameters": "${ .input.parameters }"
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
                "toStateData": "${ .output }"
              }
```

```
            }
          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

Invoke a POST API

This procedure shows how to create a flow to invoke a POST API and uses connection properties to update API host information.

Prerequisites:

The body is one of the action input schema properties and is passed to the target schema.

1. Open the adapter definition document in Visual Studio Code.
2. In the `flows` section of the document, define the flow with a single state of `Operation` type.

   Sample code:

```
"flows": {
   "postOrdersFlow" : {
      "id": "postOrdersFlow",
      "description": "postOrdersFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [

      ],
      "states": [
        {
          "actions": [

          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

3. Define a function with a unique name, of type `custom`, and operation `connectivity::rest`.

   Sample code:

```
"functions": [
        {
          "name": "postOrdersFunction",
          "operation": "connectivity::rest",
          "type": "custom"
```

```
        }
      ]
```

4. Add an action to the state where:
   - The action refers to the function created in step 2.
   - Arguments to function define the HTTP method, URI, parameters (template and query), and body.
   - Response is passed through, and the result is set to output.

   Sample code:

```
{
            "functionRef": {
              "refName": "postOrdersFunction",
              "arguments": {
                "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                "method": "POST",
                "body": "${ .input.body }"
              }
            },
            "actionDataFilter": {
              "results": "${ { body: .body, headers: .headers } }",
              "toStateData": "${ .output }"
            }
          }
```

```
"flows": {
   "postOrdersFlow" : {
      "id": "postOrdersFlow",
      "description": "postOrdersFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [
        {
          "name": "postOrdersFunction",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ],
      "states": [
        {
          "actions": [
            {
              "functionRef": {
                "refName": "postOrdersFunction",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                  "method": "POST",
                  "body": "${ .input.body }"
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
```

```
                "toStateData": "${ .output }"
              }
            }
          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

Response Transformation in Post-processing

This procedure shows how to create a flow that transforms the response in a post-processing operation.

The example code implements a flow with a GET API call for a range of values from Google Sheets. It uses the `convertResult` expression to convert the output to an array of values.

1. Add a flow with a Get call to the Google Sheets API.

```
"flows": {
   "getRowFlow" :{
      "id": "getRowFlow",
      "description": "getRowFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "https://shhets-svc.api.com/v4/spreadsheets/
{spreadsheetId}/values/{range}",
                  "method": "GET",
                  "parameters": {
                    "spreadsheetId": "${ .configuration.spreadsheetId }",
                    "range": "${ .configuration.sheetId + \"!A1:Z\" +
(.input.rowNumber|tostring) }",
                    "majorDimension": "COLUMNS"
                  }
                }
              },
```

```
                "actionDataFilter": {
                  "results": "${ .body.values }",
                  "toStateData": "${ .values }"
                }
              }
            ],
            "end": true
          }
        ]
      }
}
```

2. Configure the post-processing transformation.
   a. Add a function definition in the flow.
   b. Refer the function defined from the new action in state.

```
"functions": [
       ...
       {
          "name": "convertResult",
          "type": "expression",
          "operation": "(if .configuration.rowOperation == \"lastRow\"
then (.lastLineNumber-1) else (.input.rowNumber-1) end) as $rowNum
| .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
       }
]
...
 "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            ...
           {
              "functionRef": "convertResult",
              "actionDataFilter": {
                "toStateData": "${ .output }"
              }
            }
        ]
```

3. Run the effective flow.
   a. Make a Get call to the Google Sheets API for the range till the row number for which action has been configured.
   b. Use the `convertResult` expression to convert the output to an array of values.

```
"flows": {
   "getRowFlow" :{
      "id": "getRowFlow",
      "description": "getRowFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
```

```
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        },
        {
          "name": "convertResult",
          "type": "expression",
          "operation": "(if .configuration.rowOperation == \"lastRow\"
then (.lastLineNumber-1) else (.input.rowNumber-1) end) as $rowNum
| .values | map({key:.[0], value:.[$rowNum]}) | from_entries"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "https://shhets-svc.api.com/v4/spreadsheets/
{spreadsheetId}/values/{range}",
                  "method": "GET",
                  "parameters": {
                    "spreadsheetId": "${ .configuration.spreadsheetId }",
                    "range": "${ .configuration.sheetId + \"!A1:Z\" +
(.input.rowNumber|tostring) }",
                    "majorDimension": "COLUMNS"
                  }
                }
              },
              "actionDataFilter": {
                "results": "${ .body.values }",
                "toStateData": "${ .values }"
              }
            },
            {
              "functionRef": "convertResult",
              "actionDataFilter": {
                "toStateData": "${ .output }"
              }
            }
          ],
          "end": true
        }
      ]
    }
}
```

Send or Receive Form URL Encoded Content

This procedure shows how to implement a flow to send or receive form URL encoded content.

The `connectivity::rest` function implicitly determines how the content is handled based on the payload. This parameter treats the payload as form data if the content type is application/x-www-form-urlencoded, and automatically converts the JSON key-value pair into form-data.

1. Open the adapter definition document in Visual Studio Code.
2. Define the action's input and output schema with key values.

   Sample code:

```
"formData": {
      "type": "object",
      "properties": {
        "key1": {
          "type": "string"
        },
        "key2": {
          "type": "string"
        }
      }
    },
```

3. Refer to the schema in the action as input and output schema.

   Sample code:

```
"sendRecieveFormData": {
      "description": "",
      "displayName": "upload file",
      "execute": "flow:formDataFlow",
      "input": {
        "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/formData"
        }
      },
      "output": {
          "schemaType": "application/schema+json",
        "schema": {
          "$ref": "#/schemas/formData"
        }
      }
    },
```

4. Create a flow with `connectivity::rest` parameter and pass the body as input.

   Sample code:

```
{
              "functionRef": {
                "refName": "httpOutbound",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
```

```
                    "method": "POST",
                    "headers":{
                      "content-type":"application/x-www-form-urlencoded"
                    }
                    "body": "${ .input}"
                  }
                },
                "actionDataFilter": {
                  "results": "${ .body } }",
                  "toStateData": "${ .output }"
                }
              }
```

> **Note:**
>
> - In request phase, the `connectivity::rest` function automatically converts JSON input. For example, {"key1":"value1","key2":"value2"} to key1=value1&key2=value2.
> - If the response media type is `application/www-form-urlencoded`, in the response phase, the `connectivity::rest` function automatically converts the response payload. For example, key1=value11&key2=value21 to JSON {"key1":"value11","key2":"value21"}.

```
"flows": {
   "UploadFileFlow" : {
      "id": "UploadFileFlow",
      "description": "UploadFileFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [
        {
          "name": "httpOutbound",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ],
      "states": [
        {
          "actions": [
            {
              "functionRef": {
                "refName": "httpOutbound",
                 "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                  "method": "POST",
                  "headers":{
                    "content-type":"application/x-www-form-urlencoded"
                  }
                  "body": "${ .input}"
                }
              },
```

```
            "actionDataFilter": {
              "results": "${ .body } }",
              "toStateData": "${ .output }"
            }
          }
        ],
        "name": "startState",
        "type": "operation",
        "end": true
      }
    ]
  }
}
```

Receive Binary Content

This procedure shows how to create a flow to receive binary content.

The sample code shows how to use connection properties to receive API host information. The `connectivity::rest` function implicitly determines the response content handling based on payload. If response is attachment and not multipart, the function treats the response as binary.

1. Open the adapter definition document in Visual Studio Code.
2. Define the action's input schema as `binary`.

   Sample code:

   ```
   "DownloadFileResponseSchema": {
        "type": "object",
        "properties": {
          "File": {
            "type": "string",
            "format": "binary"
          }
        }
      }
   ```

3. Refer to the schema in the action as input schema.

   Sample code:

   ```
   "DownloadFileAction": {
        "description": "",
        "displayName": "download file",
        "execute": "flow:DownloadFileFlow",
        "input": {
         ...
        },
        "output": {
          "schemaType": "application/schema+json",
          "schema": {
            "$ref": "#/schemas/DownloadFileResponseSchema"
          }
        }
      },
   ```

4. Create a flow with `connectivity::rest` operation and pass the `.input.File` as body.

   Sample code:

```
{
         "functionRef": {
           "refName": "httpOutbound",
           "arguments": {
             "uri": "${.connectionProperties.baseURL + \"/orders\"}",
             "method": "GET"
           }
         },
         "actionDataFilter": {
           "results": "${ { File: .body} }",
           "toStateData": "${ .output }"
         }
       }


"flows": {
   "DownloadFileFlow" : {
      "id": "DownloadFileFlow",
      "description": "DownloadFileFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [
        {
          "name": "httpOutbound",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ],
      "states": [
        {
          "actions": [
            {
              "functionRef": {
                "refName": "httpOutbound",
               "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                  "method": "GET"
                }
              },
              "actionDataFilter": {
                "results": "${ { File: .body} }",
                "toStateData": "${ .output }"
              }
            }
          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
```

```
    }
}
```

Send Multi-part Content

This procedure shows how to create a flow that sends multi-part content.

The `connectivity::rest` parameter implicitly treats the request as multi-part where the content type is multipart/*, and the body is an array of parts.

1. Open the adapter definition document in Visual Studio Code.
2. Define an action input and output schema with key values.

   Sample code:

```
"UploadFileRequestSchema": {
     "type": "object",
     "properties": {
       "File": {
         "type": "string",
         "format": "binary"
       },
       "FileMetadata": {
         "type": "object",
         "properties": {
           "title": {
             "type": "string"
           },
           "mimeType": {
             "type": "string"
           },
           "description": {
             "type": "string"
           }
         }
       }
     }
   }
```

3. Refer to the schema in the action as input and output schema.

   Sample code:

```
 "UploadFileAction": {
     "description": "",
     "displayName": "upload file",
     "execute": "flow:UploadFileFlow",
     "input": {
       "schemaType": "application/schema+json",
       "schema": {
         "$ref": "#/schemas/UploadFileRequestSchema"
       }
     },
     "output": {
       ...
       }
```

```
        }
      }
```

4. Create a flow with `connectivity::rest` parameter and use an array of PART objects to define the body.

   Sample code:

```
{
              "functionRef": {
                "refName": "httpOutbound",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/upload/
drive/v2/files\"}",
                  "method": "POST",
                  "body": [
                    {
                      "Content-Type": "text/plain",
                      "Content-Disposition": "form-data;
name=\"metadata\"",
                      "Content": "${ .input.FileMetadata }"
                    },
                    {
                      "Content-Type": "${ .input.FileMetadata.mimeType }",
                      "Content-Disposition": "${ \"form-data; name=\\
\"file\\\"\"}",
                      "Content": "${ .input.File }"
                    }
                  ]
                }
              },
              "actionDataFilter": {
                "results": "${ .body } }",
                "toStateData": "${ .output }"
              }
            }


"flows": {
      "UploadFileFlow": {
      "id": "UploadFileFlow",
      "version": "0.1",
      "start": "startState",
      "specVersion": "0.8",
      "functions": [
        {
          "name": "UploadFileFunction",
          "operation": "connectivity::rest",
          "type": "custom"
        }
      ],
      "states": [
        {
          "name": "startState",
          "type": "operation",
          "actions": [
            {
```

```
              "functionRef": {
                "refName": "httpOutbound",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/upload/
drive/v2/files\"}",
                  "method": "POST",
                  "body": [
                    {
                      "Content-Type": "text/plain",
                      "Content-Disposition": "form-data; name=\"metadata\"",
                      "Content": "${ .input.FileMetadata }"
                    },
                    {
                      "Content-Type": "${ .input.FileMetadata.mimeType }",
                      "Content-Disposition": "${ \"form-data; name=\\\"file\\
\"\"}",
                      "Content": "${ .input.File }"
                    }
                  ]
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
                "toStateData": "${ .output }"
              }
            }
          ],
          "end": true
        }
      ]
    }
}
```

Send Binary Content

This procedure shows how to create a flow to send binary content.

The `connectivity::rest` parameter implicitly determines the response content handling based on payload. If the action's input schema defines input type as binary and if the binary object is body for `connectivity::rest`, the `connectivity::rest` parameter treats as a binary stream.

1. Open the adapter definition document in Visual Studio Code.
2. Define the action's input schema as `binary`.

   Sample code:

```
"UploadFileRequestSchema": {
   "type": "object",
   "properties": {
     "File": {
       "type": "string",
       "format": "binary"
     }
```

```
    }
  },
```

3. Refer to the schema in the action as input schema.

   Sample code:

```
"UploadFileAction": {
     "description": "",
     "displayName": "upload file",
     "execute": "flow:UploadFileFlow",
     "input": {
       "schemaType": "application/schema+json",
       "schema": {
         "$ref": "#/schemas/UploadFileRequestSchema"
       }
     },
     "output": {
      ...
     }
   },
```

4. Create a flow with `connectivity::rest` operation and pass the `.input.File` as body.

   Sample code:

```
{
              "functionRef": {
                "refName": "httpOutbound",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                  "method": "POST",
                  "body": "${ .input.File}"
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
                "toStateData": "${ .output }"
              }
            }

"flows": {
   "UploadFileFlow" : {
     "id": "UploadFileFlow",
     "description": "UploadFileFlow",
     "version": "0.1",
     "start": "startState",
     "specVersion": "0.8",
     "functions": [
       {
         "name": "httpOutbound",
         "operation": "connectivity::rest",
         "type": "custom"
       }
     ],
     "states": [
       {
         "actions": [
```

```
            {
              "functionRef": {
                "refName": "httpOutbound",
                "arguments": {
                  "uri": "${.connectionProperties.baseURL + \"/orders\"}",
                  "method": "POST",
                  "body": "${ .input.File}"
                }
              },
              "actionDataFilter": {
                "results": "${ { body: .body, headers: .headers } }",
                "toStateData": "${ .output }"
              }
            }
          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

Override Default Content Handling in connectivity::rest

By default, the `connectivityt::rest` function implements a default behavior that handles the request and response content based on the content headers and actual content type like binary and JSON.

The Rapid Adapter Builder platform allows you to override the default implementation behavior.

To override the default behavior in request phase, add the following argument to `connectivity::rest` function.

> **Note:**
>
> Valid values for `requestMediaType` are `structured`, `form-urlencoded`, `multipart`, and `binary`.

Sample code:

```
{
          "functionRef": {
            "refName": "httpOutbound",
            "arguments": {
              "uri": "${.connectionProperties.baseURL + \"/orders\"}",
              "method": "POST",
              "requestMediaType" : "binary"
            }
          },
          "actionDataFilter": {
            "results": "${ { File: .body} }",
            "toStateData": "${ .output }"
          }
        }
```

To override the default behavior in response phase, add the following argument to `connectivity::rest` function.

> **Note:**
>
> Valid values for `responseMediaType` are `structured`, `form-urlencoded`, `multipart`, and `binary`.

Sample code:

```
{
         "functionRef": {
           "refName": "httpOutbound",
           "arguments": {
             "uri": "${.connectionProperties.baseURL + \"/orders\"}",
             "method": "GET",
             "responseMediaType" : "binary"
           }
         },
         "actionDataFilter": {
           "results": "${ { File: .body} }",
           "toStateData": "${ .output }"
         }
       }
```

Pre and Post Processing Expressions and Multiple Outbound Requests

This procedure shows how to implement a flow with pre or post processing expressions, and multiple outbound requests.

The sample code shows how to:

- Pre-processes a task and build tenant information.
- Makes multiple outbound requests.
- Pre-processes the output to make a final request.
- Merges the output of two requests as a post-processing operation.

1. Open the adapter definition document in Visual Studio Code.
2. Locate the `flows` code section of the document.
3. Define a flow with a single state of type operation. For outbound calls, add function of type custom, operation `connectivity::rest`, and a unique name.

   Sample code:

   ```
   "flows": {
       "compartmentIdFlow" :      {
         "id": "compartmentIdFlow",
         "description": "compartmentIdFlow",
         "specVersion": "0.8",
         "version": "0.1",
         "start": "startState",
         "functions": [
            {
   ```

```
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }

      ],
      "states": [
        {
          "actions": [

          ],
          "name": "startState",
          "type": "operation",
          "end": true
        }
      ]
    }
}
```

4. Add a function definition in the flow and refer the function defined from first action in state.

   The pre-processing operation involves extraction of the tenantId from the configuration or connection properties based on a condition.

   Sample code:

```
"functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        },
        {
          "name": "getTenantId",
          "type": "expression",
          "operation": "if .configuration.tenantId
then .configuration.tenantId else .connectionProperties.TenancyOCID end"
        }
]
...
 "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "getTenantId",
              "actionDataFilter": {
                "toStateData": "${ .configuration.tenantId }"
              }
            }
        ]
```

5. For outbound invokes, add two more actions to the outbound invocation. The first invoke retrieves the resourceId for the identified tenant. The second invoke retrieves the sub-reources id from the identified tenant.

Sample code:

```
"actions":[
            {
              "functionRef": "getTenantId",
              "actionDataFilter": {
                "toStateData": "${ .configuration.tenantId }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments/\" + .configuration.tenantId }",
                  "method": "GET"
                }
              },
              "actionDataFilter": {
                "results": "${ {keyName:.body.id,
displayName:.body.name} }",
                "toStateData": "${ .rootCompartment }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments\" }",
                  "method": "GET",
                  "parameters": {
                    "compartmentId": "${ .configuration.tenantId }",
                    "lifecycleState": "ACTIVE",
                    "compartmentIdInSubtree": true
                  }
                }
              },
              "actionDataFilter": {
                "results": "${ .body | map({keyName:.id,
displayName:.name}) }",
                "toStateData": "${ .childCompartments }"
              }
            }
  ]
```

6. As part of post- processing operation, merge the output of the two invokes.

   Sample code:

```
"functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
```

```
          "operation": "connectivity::rest"
        },
        {
          "name": "getTenantId",
          "type": "expression",
          "operation": "if .configuration.tenantId
then .configuration.tenantId else .connectionProperties.TenancyOCID end"
        },
        {
          "name": "mergeTenantsList",
          "type": "expression",
          "operation": "[.rootCompartment] + .childCompartments"
        }
      ]
...
 "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "getTenantId",
              "actionDataFilter": {
                "toStateData": "${ .configuration.tenantId }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments/\" + .configuration.tenantId }",
                  "method": "GET"
                }
              },
              "actionDataFilter": {
                "results": "${ {keyName:.body.id,
displayName:.body.name} }",
                "toStateData": "${ .rootCompartment }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments\" }",
                  "method": "GET",
                  "parameters": {
                    "compartmentId": "${ .configuration.tenantId }",
                    "lifecycleState": "ACTIVE",
                    "compartmentIdInSubtree": true
                  }
                }
```

```
                },
                "actionDataFilter": {
                  "results": "${ .body | map({keyName:.id,
    displayName:.name}) }",
                  "toStateData": "${ .childCompartments }"
                }
              },
              {
                "functionRef": "mergeTenantsList",
                "actionDataFilter": {
                  "toStateData": "${ .output }"
                }
              }
            ],
            "end": true
          }
        ]
      }
    }
```

Complete sample code:

```
"flows": {
   "compartmentIdFlow" :      {
      "id": "compartmentIdFlow",
      "description": "compartmentIdFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        },
        {
          "name": "getTenantId",
          "type": "expression",
          "operation": "if .configuration.tenantId
then .configuration.tenantId else .connectionProperties.TenancyOCID end"
        },
        {
          "name": "mergeTenantsList",
          "type": "expression",
          "operation": "[.rootCompartment] + .childCompartments"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "getTenantId",
              "actionDataFilter": {
                "toStateData": "${ .configuration.tenantId }"
```

```
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/compartments/\"
+ .configuration.tenantId }",
                  "method": "GET"
                }
              },
              "actionDataFilter": {
                "results": "${ {keyName:.body.id, displayName:.body.name} }",
                "toStateData": "${ .rootCompartment }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments\" }",
                  "method": "GET",
                  "parameters": {
                    "compartmentId": "${ .configuration.tenantId }",
                    "lifecycleState": "ACTIVE",
                    "compartmentIdInSubtree": true
                  }
                }
              },
              "actionDataFilter": {
                "results": "${ .body | map({keyName:.id,
displayName:.name}) }",
                "toStateData": "${ .childCompartments }"
              }
            },
            {
              "functionRef": "mergeTenantsList",
              "actionDataFilter": {
                "toStateData": "${ .output }"
              }
            }
          ],
          "end": true
        }
      ]
    }
}
```

Fetch Design Time Configuration Values Using a Flow

This procedure provides an overview of how to use a flow to retrieve design time configuration values.

1. Open the adapter definition document in Visual Studio Code Editor.
2. In the `flows` code section of the document, write the code for the required flows. For example, the following code sample contains `compartmentIdFlow`, `applicationIdFlow`, and `functionIdFlow` flows.

> **Note:**
>
> In this code sample, the flows return array outputs that work as input for another flow.

```
"flows":     {
   "compartmentIdFlow":{
      "id": "compartmentIdFlow",
      "description": "compartmentIdFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        },
        {
          "name": "getTenantId",
          "type": "expression",
          "operation": "if .configuration.tenantId
then .configuration.tenantId else .connectionProperties.TenancyOCID end"
        },
        {
          "name": "mergeTenantsList",
          "type": "expression",
          "operation": "[.rootCompartment] + .childCompartments"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": "getTenantId",
              "actionDataFilter": {
                "toStateData": "${ .configuration.tenantId }"
              }
            },
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments/\" + .configuration.tenantId }",
```

```
                    "method": "GET"
                  }
                },
                "actionDataFilter": {
                  "results": "${ {keyName:.body.id,
displayName:.body.name} }",
                  "toStateData": "${ .rootCompartment }"
                }
              },
              {
                "functionRef": {
                  "refName": "generalRestFunc",
                  "arguments": {
                    "uri": "${ \"https://identity.\"
+ .connectionProperties.region + \".oraclecloud.com/20160918/
compartments\" }",
                    "method": "GET",
                    "parameters": {
                      "compartmentId": "${ .configuration.tenantId }",
                      "lifecycleState": "ACTIVE",
                      "compartmentIdInSubtree": true
                    }
                  }
                },
                "actionDataFilter": {
                  "results": "${ .body | map({keyName:.id,
displayName:.name}) }",
                  "toStateData": "${ .childCompartments }"
                }
              },
              {
                "functionRef": "mergeTenantsList",
                "actionDataFilter": {
                  "toStateData": "${ .output }"
                }
              }
            ],
            "end": true
          }
        ]
      },
      "applicationIdFlow": {
        "id": "applicationIdFlow",
        "description": "applicationIdFlow",
        "specVersion": "0.8",
        "version": "0.1",
        "start": "startState",
        "functions": [
          {
            "name": "generalRestFunc",
            "type": "custom",
            "operation": "connectivity::rest"
          }
        ],
        "states":[
          {
```

```
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://functions.\"
+ .connectionProperties.region + \".oraclecloud.com/20181201/
applications\" }",
                  "method": "GET",
                  "parameters": {
                    "compartmentId": "${ .configuration.compartmentId }"
                  }
                }
              },
              "actionDataFilter": {
                "results": "${ .body | map({keyName:.id,
displayName:.displayName}) }",
                "toStateData": "${ .output }"
              }
            }
          ],
          "end": true
        }
      ]
    },
    "functionIdFlow": {
      "id": "functionIdFlow",
      "description": "functionIdFlow",
      "specVersion": "0.8",
      "version": "0.1",
      "start": "startState",
      "functions": [
        {
          "name": "generalRestFunc",
          "type": "custom",
          "operation": "connectivity::rest"
        }
      ],
      "states":[
        {
          "name":"startState",
          "type":"operation",
          "actions":[
            {
              "functionRef": {
                "refName": "generalRestFunc",
                "arguments": {
                  "uri": "${ \"https://functions.\"
+ .connectionProperties.region + \".oraclecloud.com/20181201/
functions\" }",
                  "method": "GET",
                  "parameters": {
                    "applicationId": "${ .configuration.applicationId }"
                  }
```

```
                }
              },
              "actionDataFilter": {
                "results": "${ .body | map({keyName:.id,
displayName:.displayName}) }",
                "toStateData": "${ .output }"
              }
            }
          ],
          "end": true
        }
      ]
    }
}
```

3. In the `actions` code section of the document, define an action with a flow that retrieves configuration values from design time.

   Sample code:

```
{
 "actions":{
        "invokeFunctionAction": {
      "description": "Invoke Function",
      "summary": "Invoke Function.",
      "group": "function",
      "urn": "flow:invokeFunctionFlow",
      "input": {
        "userDefined": true
      },
      "output": {
        "userDefined": true
      },
      "configuration": [
        {
          "name": "tenantId",
          "displayName": "Override Tenant OCID",
          "description": "",
          "type": "TEXT_BOX",
          "required": false
        },
        {
          "name": "compartmentId",
          "displayName": "Compartment Name",
          "description": "",
          "type": "COMBO_BOX",
          "urn": "flow:compartmentIdFlow",
          "required": true
        },
        {
          "name": "applicationId",
          "displayName": "Application Name",
          "description": "",
          "type": "COMBO_BOX",
          "urn": "flow:applicationIdFlow",
          "required": true,
```

```
            "dependencies": {
              "compartmentId": {
                "values": []
              }
            }
          },
          {
            "name": "functionId",
            "displayName": "Function Name",
            "description": "",
            "type": "COMBO_BOX",
            "urn": "flow:functionIdFlow",
            "required": true,
            "dependencies": {
              "applicationId": {
                "values": []
              }
            }
          }
        ]
      }
    }
  }
```

15

Update Category Definitions

In the `categories` section of an adapter definition document, you can group actions and triggers for a better user experience.

What Do You Want to Do?

Task	More information
Get oriented	Introduction to Categories Where Categories Appear in Oracle Integration
Review the properties to define and see sample code	Categories Properties and Sample Code

Learn About Categories

Before updating the `categories` section of an adapter definition document, familiarize yourself with this section and its properties.

Topics:

- Introduction to Categories
- Where Categories Appear in Oracle Integration
- Categories Properties and Sample Code

Introduction to Categories

In the `categories` section of an adapter definition document, you can categorize actions and triggers, so that it becomes easier to choose the required actions or triggers while configuring an adapter in Oracle Integration.

While modeling an adapter's capabilities, you may have to define a large number of actions and triggers depending on the requirements. As actions and triggers appear in a drop-down list in the design-time user interface of Oracle Integration, it becomes difficult to choose the required value from a long list.

Using the Rapid Adapter Builder, you can categorize actions and triggers, so that the number of items in an actions or triggers drop-down list are reduced and easy to view.

For example, if you are building an adapter for an application with multiple modules (like CRM, Financials, ERP, and HCM), you may have to define several actions for each module. In this case, you can define each module as a category.

Here's a code snippet for this example scenario:

```
"categories": {
        "displayName": "Select Module",
        "description": "Enter Module to filter.",
        "groups": [
```

```
{
    "name": "crm",
    "displayName": "CRM",
    "description": "Customer Relationship Management"
},
{
    "name": "financials",
    "displayName": "Financials",
    "description": "Financials"
},
{
    "name": "erp",
    "displayName": "ERP",
    "description": "Enterprise Resource Planning"
},
{
    "name": "hcm",
    "displayName": "HCM",
    "description": "Human Capital Management"
}
```

You can then associate an action or trigger with one of the `groups` defined in the `categories` section of the adapter definition document.

The following code snippet shows how to associate an action with a group defined earlier:

```
"actions": {
    "createEmployee": {
        "description": "Create Employee",
        "displayName": "Create Employee in HCM system",
        "group": "hcm",
        "execute": "flow:createEmployeeInHCM",
```

> **Note:**
>
> Categorization of actions and triggers using the category and group mechanism is optional.

Where Categories Appear in Oracle Integration

In Oracle Integration, you can view the category information while configuring an adapter in an integration flow.

The categories appear in an actions or triggers drop-down list in the Adapter Endpoint Configuration Wizard.

Categories Properties and Sample Code

The `categories` section of an adapter definition document typically contains the properties listed here and has the following sample structure.

- Properties

- Sample Code

Properties

Property	Description
`displayName`	The display name of the group.
`description`	The description of the group.
`groups`	Defines a set actions or triggers. Each group consists of the following properties: • `name` The unique name of the group that is a selectable item value in the design-time user interface. • `displayName` The display name of the group that appears as a selectable item value in the design-time user interface. • `description` The description of the group that appears as a tooltip in the user interface.

Sample Code

```
{
  "categories": {
    "displayName": "Select Service",
    "description": "Enter service name to filter.",
    "groups": [
      {
        "name": "simpleAction",
        "displayName": "Simple Actions",
        "description": "Simple Actions."
      },
      {
        "name": "advancedAction",
        "displayName": "Advanced Actions",
        "description": "Advanced Actions."
      }
    ]
  }
}
```