Oracle® Identity Manager

Connector Guide for Generic REST

Release 11.1.1
E71482-09
September 2020

Oracle Identity Manager Connector Guide for Generic REST, Release 11.1.1

E71482-09

Primary Author: Gowri.G.R

Contributing Authors: Alankrita Prakash

Contents

List of Tables

Preface

This guide describes the connector that is used to integrate Oracle Identity Manager with any REST-based identity-aware application.

Audience

This guide is intended for resource administrators and target system integration teams. This guide assumes that you are familiar with REST-based architecture and standards.

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.

Related Documents

For information about installing and using Oracle Identity Manager, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E52734_01/index.html

For information about Oracle Identity Manager Connectors documentation, visit the following Oracle Help Center page:

http://docs.oracle.com/cd/E22999_01/index.htm

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.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in Oracle Identity Manager Connector for Generic REST?

This chapter provides an overview of the updates made to the software and documentation for the Generic REST connector in release 11.1.1.5.0.

The updates discussed in this chapter are divided into the following categories:

- Software Updates

  This section describes updates made to the connector software. This section also points out the sections of this guide that have been changed in response to each software update.

- Documentation-Specific Updates

  These include major changes made to this guide. For example, the relocation of a section from the second chapter to the third chapter is a documentation-specific update. These changes are not related to software updates.

Software Updates

The following section discusses software updates:

Software Updates in Release 11.1.1.5.0

This is the first release of the Oracle Identity Manager connector for Generic REST. Therefore, there are no software-specific updates in this release.

Documentation-Specific Updates

The following section discusses documentation-specific updates:

Documentation-Specific Updates in Release 11.1.1.5.0

The following documentation-specific update has been made in revision "09" of this guide:

The "Oracle Identity Governance or Oracle Identity Manager" row of Table 1-1 has been updated to include support for Oracle Identity Governance 12*c* (12.2.1.4.0).

The following documentation-specific update has been made in revision "08" of this guide:

Few editorial changes and minor updates to the document structure have been made for better readability.

The following documentation-specific update has been made in revision "07" of this guide:

Information about refresh tokens has been removed from Introduction to the Generic REST Connector and Custom Authentication.

The following documentation-specific update has been made in revision "06" of this guide:

Minor updates to the document structure have been made for better readability.

The following documentation-specific updates have been made in revision "05" of this guide:

- Information about ADDATTRIBUTE, REMOVEATTRIBUTE, UPDATEOP, and DELETEOP have been included in Additional Configuration Parameters.
- The opTypes row of Table 4-7 has been updated to include a limitation of the connector regarding deleting account operations.
- The "Connector Server" row of Table 1-1 has been updated with certification for 12.2.1.3.0.

The following documentation-specific update has been made in revision "04" of this guide:

The “Oracle Identity Manager” row of Table 1-1 has been renamed as “Oracle Identity Governance or Oracle Identity Manager” and also updated for 12c (12.2.1.3.0) certification.

The following documentation-specific updates have been made in revision "03" of this guide:

- Broken cross-references have been fixed.
- Chapter 6, “Known Issues and Workaround for the Generic REST Connector” has been removed as there are no known issues associated with this connector.

The following documentation-specific updates have been made in revision "02" of this guide:

- Information about packaging a JAR file containing custom implementation has been modified in Implementing Custom Authentication and Implementing Custom Parsing.
- A note regarding the Scheduler Status page has been modified in Configuring Scheduled Jobs.
- Oracle Identity Manager user interface names have been corrected throughout the guide.

1

About the Generic REST Connector

The Generic REST connector integrates Oracle Identity Manager with REST-based target systems.

Oracle Identity Manager is a centralized identity management solution that provides self service, compliance, provisioning and password management services for applications residing on-premise or on the Cloud. Oracle Identity Manager connectors are used to integrate Oracle identity Manager with the external and identity-aware applications.

The following topics introduce the Generic REST connector:

1.1 Introduction to the Generic REST Connector

The Generic REST connector is a solution to integrate OIM with REST-based identity-aware applications. A **REST-based identity-aware application** is any application that exposes its REST APIs or interfaces for identity management.

> **Note:**
>
> In this guide:
>
> - A REST-based identity-aware application has been referred to as the **target system** or **REST-based target system.**
> - *RELEASE_NUMBER* has been used as a placeholder for the current release number of the connector. Therefore, replace all instances of *RELEASE_NUMBER* with the release number of the connector. For example, 11.1.1.5.0.

The Generic REST connector provides a centralized system to streamline delivery of services and assets to your company's consumers, and manage those services and assets in a simple, secure, and cost efficient manner by using automation. The Generic REST connector standardizes service processes and implements automation to replace manual tasks.

In order to connect with a REST-based target system, the Generic REST connector supports HTTP Basic Authentication and OAuth 2.0 authentication mechanisms. This connector also supports authenticating to the target system by using access token as an input from the user. This authentication mechanism can be useful if your target system does not provide a programmatic approach to obtain access tokens.

The connector supports the following OAuth 2.0 grant types:

- JWT
- Client Credentials
- Resource Owner Password

If your target system does not support any of the authentication types supported by this connector, then you can implement the custom authentication that your target system supports. You can connect this custom implementation to the connector by using the plug-ins exposed by this connector.

The Generic REST connector synchronizes data between OIM and REST-based target systems by performing reconciliation and provisioning operations that parse data in the JSON format. If your target system does not support request or response payload in JSON format, then you can create your own implementation for parsing data. You can connect this custom implementation to the connector by using the plug-ins exposed by this connector.

The Generic REST connector is a connector for a discovered target system. This is because the schema of the REST-based target system with which the connector integrates is not known in advance. The Generic REST connector is not shipped with any artifacts. Instead, it is shipped with a set of deployment utilities that help in discovering the schema of the REST-based target system and generating the artifacts.

1.2 Certified Components for the Generic REST Connector

These are the software components and their versions required for installing and using the connector.

Table 1-1 Certified Components

Item	Requirement
Oracle Identity Manager or Oracle Identity Governance	You can use one of the following releases of Oracle Identity Governance or Oracle Identity Manager: • Oracle Identity Governance 12*c* (12.2.1.4.0) • Oracle Identity Governance 12*c* (12.2.1.3.0) • Oracle Identity Manager 11g Release 2 PS3 (11.1.2.3.0) • Oracle Identity Manager 11g Release 2 PS2 (11.1.2.2.0)
Target System	Any target system that supports REST service.
Connector Server	• 11.1.2.1.0 • 12.2.1.3.0
Connector Server JDK	JDK 1.6 or later

1.3 Certified Languages for the Generic REST Connector

The connector will support the languages that are supported by Oracle Identity Manager. Resource bundles are not part of the connector installation media as the resource bundle entries vary depending on the target system being used.

1.4 Features of the Generic REST Connector

The features of the connector include support for full and incremental reconciliation, limited reconciliation, custom authentication, custom parsing, custom payload, handling multiple endpoint URLs, and SSL communication.

The following are the features of the connector:

- Support for Both Trusted Source and Target Resource Reconciliation
- Full and Incremental Reconciliation
- Limited (Filtered) Reconciliation
- Custom Authentication
- Custom Parsing
- Custom Payload
- Support for Additional HTTP Headers
- Support for Handling Multiple Endpoint URLs
- SSL Communication

1.4.1 Support for Both Trusted Source and Target Resource Reconciliation

The Generic REST connector includes a groovy file (a part of the metadata generator) that enables you to configure the connector to run either in the trusted source mode or target resource mode.

1.4.2 Full and Incremental Reconciliation

After you create the connector, you can perform full reconciliation to bring all existing user data from the target system to Oracle Identity Manager. After the first full reconciliation run, you can configure your connector for incremental reconciliation. In incremental reconciliation, only records that are added or modified after the last reconciliation run are fetched into Oracle Identity Manager.

> **Note:**
>
> The connector supports incremental reconciliation if the target system contains an attribute that holds the timestamp at which an object is created or modified.

You can perform a full reconciliation any time. See Full Reconciliation and Incremental Reconciliation.

1.4.3 Limited (Filtered) Reconciliation

You can reconcile records from the target system based on a specified filter criterion. To limit or filter the records that are fetched into Oracle Identity Manager during a reconciliation run, you can specify the subset of added or modified target system records that must be reconciled.

You can set a reconciliation filter as the value of the Filter Suffix attribute of the scheduled jobs. This filter specifies the subset of newly added and modified target system records that must be reconciled.

See Limited (Filtered) Reconciliation.

1.4.4 Custom Authentication

By default, the Generic REST connector supports HTTP Basic Authentication and OAuth 2.0 authentication mechanisms. The connector also supports an authentication mechanism in which the user provides access token as an input. The supported grant types for OAuth 2.0 authentication mechanism are JWT, Client Credentials, and Resource Owner Password. If your target system uses any of the authentication mechanisms that is not supported by the connector, then you can write your own implementation for custom authentication by using the plug-ins exposed by this connector.
See Implementing Custom Authentication for more information about creating your own implementation for the custom authentication.

1.4.5 Custom Parsing

By default, the Generic REST connector supports request and response payloads only in the JSON format. If your target system does not support request or response payload in JSON format, then you can implement a custom parsing logic by using plug-ins exposed by this connector.

See Implementing Custom Parsing for more information about custom parsing.

1.4.6 Custom Payload

The Generic REST connector provides support for handling custom formats for any attributes in the payload that do not adhere to the standard JSON format.

This can be achieved by specifying a value for the customPayload IT resource parameter. See Additional Configuration Parameters for more information about this parameter.

1.4.7 Support for Additional HTTP Headers

If your target system requires additional or custom HTTP headers in any REST call, then you can insert these HTTP headers as the value of the customAuthHeaders or customAuthHeaders IT resource parameters.

See Additional Configuration Parameters for more information about these parameters.

1.4.8 Support for Handling Multiple Endpoint URLs

The Generic REST connector allows you to handle attributes of an object class (for example, a User object class) that can be managed only through endpoints other than the base endpoint URL of the object class.For example, in certain target systems, there are attributes of the User object class that can be managed using the base endpoint URL. However, some attributes (for example, email alias) can be managed only through a different endpoint URL. The connector provides support for handling all endpoint URLs associated with an object class.
This can be achieved by providing endpoint URL details of such attributes in the relURIs IT resource parameter. See Additional Configuration Parameters for more information about this parameter.

1.4.9 SSL Communication

You can configure SSL to secure data communication between Oracle Identity Manager and the REST-based target system.

See Configuring SSL for information about configuring secure communication.

1.5 Use Cases Supported by the Generic REST Connector

The Generic REST connector can be used to integrate OIM with any target system that supports REST services. This connector can be used to load identity data into OIM from a REST service and then efficiently manage identities in an integrated cycle with the rest of the identity-aware applications in your enterprise.
As a business use case example, consider a leading logistics company that has 20+ cloud applications. Most of these cloud applications are now inefficient because data in these applications are manually entered and are managed using spreadsheets or custom-coded process flows. Therefore, this company wants to integrate its cloud applications with OIM to streamline its operations, increase its organizational efficiency, and at the same time, lower its operational costs. There are two approaches for integrating these cloud applications with OIM. One approach would be to deploy a point-to-point connector for each of these applications. The drawbacks of this approach are as follows:

- Increased time and effort to identify and deploy a point-to-point connector for each application.
- Increased administration and maintenance overheads for managing connectors for each application.
- Unavailability of point-to-point connectors for all applications. In such a scenario, one needs to develop custom connectors which increases time and effort to develop, deploy and test the custom connector.

An alternative to this approach is to use the Generic REST connector that can be used to integrate all the cloud applications with OIM. The Generic REST connector provides the ability to manage accounts across all cloud applications without spending additional resources and time on building custom connectors for each cloud application.

The Generic REST connector is a hybrid approach that helps enterprises leverage on-premise OIM deployment to integrate with target systems for identity governance. These targets systems include any application that exposes REST APIs such as SaaS, PaaS, home-grown applications and so on.

The following are some example scenarios in which the Generic REST connector is used:

- **User Management**

  The Generic REST Connector manages individuals who can access Cloud service by defining them as users in the system and assigning them to groups. This connector allows new users to self-provision on a Generic REST Cloud Service, while having it be controlled by IT. Users can request and provision from a catalog of cloud-based resources that is established by OIM administrators. For example, to create a new user in the target system, fill in and submit the OIM process form to trigger the provisioning operation. The connector executes the create operation against your target system and the user is created on successful execution of the operation. Similarly, operations such as delete and update can be performed.

- **Entitlement Management**

  The Generic REST Connector manages Cloud services objects (if exposed by the target system) as entitlements. Depending on the target system being used, this connector can be used to manage entitlements such as Groups, Roles, Licenses, Folders, Collaboration and so on. For example, you can use the Generic REST connector to automatically assign or revoke groups to users based on predefined access policies in OIM. Similarly, you can use the Generic REST Connector to manage role memberships that provide selective access to certain Cloud Service functionality or groups. Therefore, as new users are added to a specific role, they automatically gain corresponding access in the applications.

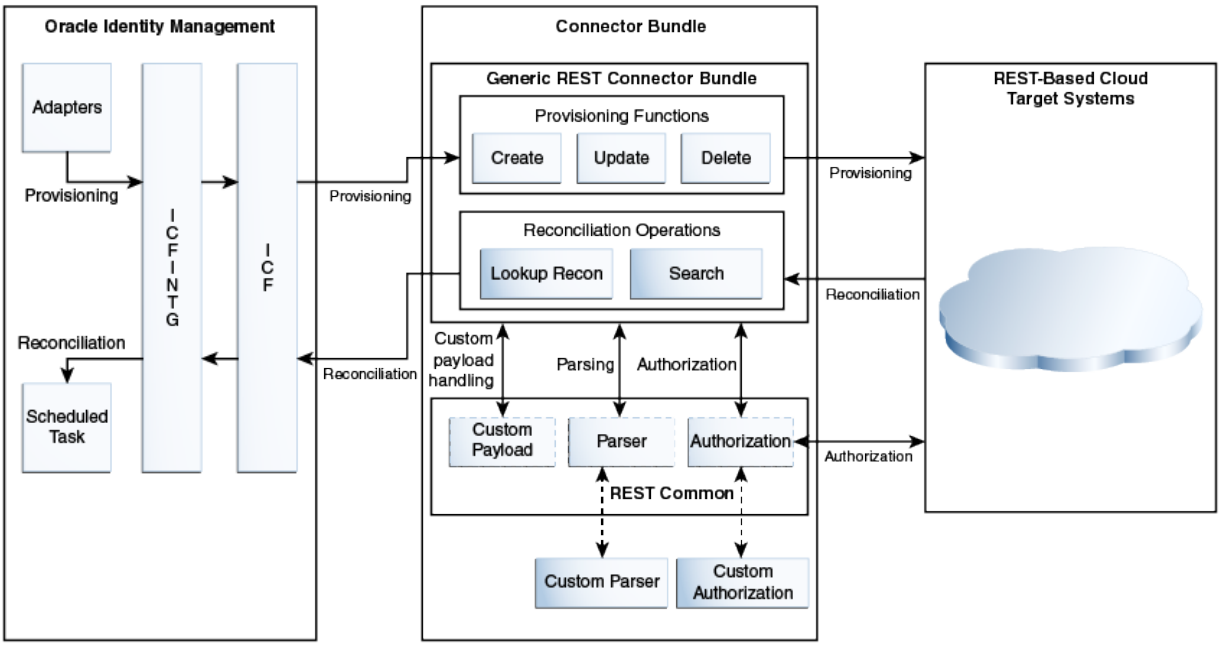
1.6 Architecture of the Generic REST Connector

The Generic REST connector is implemented using the Identity Connector Framework (ICF).

The ICF is a component that provides basic reconciliation and provisioning operations that are common to all Oracle Identity Manager connectors. In addition, ICF provides common features that developers would otherwise need to implement on their own, such as connection pooling, buffering, time outs, and filtering. The ICF is shipped along with Oracle Identity Manager.

Figure 1-1shows the architecture of the connector.

Figure 1-1 Connector Architecture



The primary function of the Generic REST connector is to connect to any application that exposes its REST APIs and then synchronize user identity data between this application and Oracle Identity Manager.

This connector is not shipped with any metadata as it is a connector for target system that is not known in advance. Depending on the schema of your target system, the connector artifacts are generated during connector deployment. Once the connector artifacts are created, Oracle Identity Manager communicates with your target system through the connector bundle by using various adapters and scheduled tasks.

The REST Common layer contains all the plug-ins and logic required by the connector to authenticate to the target system and parse data. Any custom implementation for authorization and data parsing can also be hooked as a plug-in in the REST Common layer.

During provisioning, adapters carry provisioning data submitted through the process form to the target system. The adapters establish a connection with the corresponding Create, Update, or Delete operations in the connector bundle which inturn establishes a connection with a target system by leveraging the REST Common layer. After the adapters establish a connection with the target system, REST calls are made to the endpoints and the required provisioning operation is performed. Subsequently, the response from the target system is returned to the adapters.

During reconciliation, a schedule task is run which calls the SearchOp operation of the connector bundle. The connector bundle establishes a connection with the target system by using the REST Common layer. Then, the connector retrieves all records that match the reconciliation criteria by calling the specific REST endpoint. This result is then passed to Oracle Identity Manager.

1.7 Generic REST Connector Concepts

Learn about the basic concepts behind the components used for generating the Generic REST connector.

Connector Generation

The Generic REST connector installation package is not shipped with any metadata or connector artifacts. It is shipped only with a set of deployment utilities that help in generating the metadata based on your target system schema. Therefore, understanding the schema of your target system is one of the important aspects in generating the connector. You must create a schema file describing the attributes of your target system to help the connector know your target system. The Generic REST connector installation package includes a Groovy file in which you can specify information about your target system. This information is used by the metadata generator to generate the connector based on the target system schema.

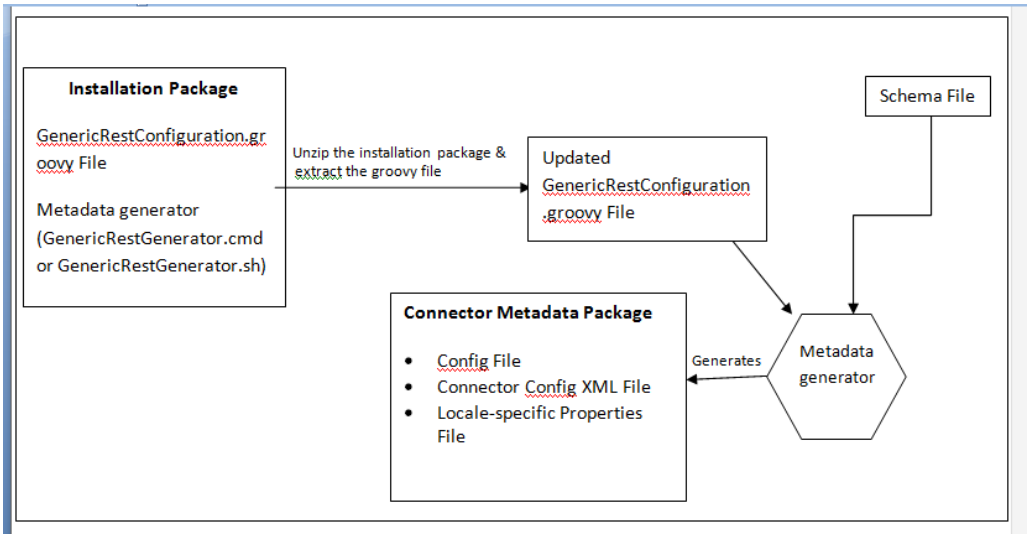
To generate the connector, you must create a schema file describing the attributes of your target, configure the Groovy file and generate the connector metadata package. Then, you must install the connector installation package to upload the connector bundle (org.identityconnectors.genericrest-1.0.1115.jar) to the Oracle Identity Manager database, and then install the connector metadata package containing the metadata specific to your target system..

The key to understanding the architecture of the connector generation and installation is to first understand its components. The Generic REST connector includes the following key components:

- Schema File — a user-created properties file
- Groovy File — GenericRestConfiguration.groovy
- Metadata Generator — GenericRestGenerator.cmd or GenericRestGenerator.sh

Figure 1-2 shows the architecture of the connector installation and the overall flow of the connector generation process.

Figure 1-2 Connector Generation Process Flow



Connector Installation Package

The connector installation package contains all files required by the Connector Installer to install this connector. The connector installation package also contains files that you must update to include your target system details and files that let you generate the metadata for your connector. The connector installation package for this connector is a ZIP file named GenericREST-*RELEASE_NUMBER*.zip that you can download from the Oracle Technology website (OTN).

For information about the files and directories in the connector installation package, see Table C-1.

Schema File

The schema file is a properties file that must contain entries pertaining to your target system in the name-value pair format. This file is not available in the connector installation package as the target system is not known in advance. You need to create a schema file representing the structure of your target system.

The schema file must define a list of all target system fields and their details that the connector needs to perform connector operations. It must also define the target system field that must be used to identify records to be fetched during reconciliation operations.

Groovy File

The Groovy file, GenericRestConfiguration.groovy, is located in the GenericREST-*RELEASE_NUMBER*/metadata-generator/resources directory of the connector installation package. You use the GenericRestConfiguration.groovy file to specify values for properties that can store basic information about your target system schema.

The metadata generator uses the GenericRestConfiguration.groovy file to perform the following tasks:

- Understand the schema
- Configure the mode (trusted source or target resource) in which you want to run the connector
- Generate the connector metadata package specific to your target system

The Groovy file has one or more configuration sections. Each section begins with *`CONFIG_NAME`* `{`, where *CONFIG_NAME* is the name of the configuration. Depending on whether you want to configure the connector to run in the trusted source or target resource mode, a section in the Groovy file can be either a trusted source type or target resource type.

The GenericRestConfiguration.groovy file provided in the connector installation package contains sample configuration (one each for trusted source and target resource) with prepopulated values for most of the entries. Depending upon your requirements, you can add or modify values for entries in the sample configuration or create new sections for your configuration. The following are the sample configuration sections in the GenericRestConfiguration.groovy file:

- trusted

  You specify values for the entries in this section if you want to configure the connector for the trusted source mode.

- target

  You specify values for the entries in this section if you want to configure the connector for the target resource mode.

Metadata Generator

The metadata generator creates files that are needed during the connector installation process and contain definitions of all connector components such as process tasks, scheduled tasks, lookup definitions and so on. The metadata generator is an executable file, GenericRestGenerator.cmd or GenericRestGenerator.sh file, that is located in the genericrest-*RELEASE_NUMBER*/metadata-generator/bin directory.

The metadata generator uses the specified configuration section from the Groovy file as an input and creates configuration files containing connector component data specific to your target system. The configuration section in the Groovy file in turn references the schema file.

The output connector metadata package is a ZIP file containing metadata files. For a description about each of the files in the connector metadata package, see Table C-2.

2

Prerequisites for Installing the Generic REST Connector

Learn about the tasks that you must complete before you install the Generic REST connector.

Topics

2.1 Downloading the Connector Installation Package

You can obtain the installation package for the Generic REST connector on the Oracle Technology Network (OTN) website.

To download the connector installation package:

1. Navigate to the OTN website at http://www.oracle.com/technetwork/middleware/id-mgmt/downloads/connectors-101674.html.
2. Click **OTN License Agreement** and read the license agreement.
3. Select the **Accept License Agreement** option.

   You must accept the license agreement before you can download the installation package.
4. Download and save the installation package to any directory on the computer hosting Oracle Identity Manager.

2.2 Creating a Schema File

Create a schema file to help the connector understand the underlying structure of your target system.

The schema file contains entries that you add to it by using Account qualifiers and Field qualifiers. Account qualifiers help you list certain fields of an account in your target system that the connector uses for performing reconciliation and provisioning operations. You can use Field qualifiers to further describe the fields listed using account qualifiers. For example, using field qualifiers, you can specify whether a given field is mandatory or a multivalued field. You can also specify the datatype of the field.

You must specify field qualifiers in one of the following formats:

- The following is the format for describing parent form fields:

  <FIELDNAME>.<FIELDQUALIFIER>=<VALUE>

  Example: `UserId.Required=true`

- The following is the format for describing complex child form fields:

  <*FIELDNAME*>.<*SUBFIELDNAME*>.<*FIELDQUALIFIER*>=<*VALUE*>.

  Example: `Roles.fromdate.DataType=Long`

The schema file that you create must adhere to the following guidelines that are recognized by the metadata generator:

- Each entry must be in a single line in the following format:

  `PropertyName=PropertyValue`

- Any space characters that appear between the property name and property value is ignored. For example, the following three entries are all the same:
  - `StatusAttribute=AccountStatus`
  - `StatusAttribute = AccountStatus`
  - `StatusAttribute= AccountStatus`
- Comments help you and others referring the schema understand its entries better. Any entry or line that begins with the number sign (#) is ignored by the metadata generator.
- Unless specified, the default data type for all fields is String.

> **Note:**
>
> You must create the schema.properties file on the computer on which you intend to run the metadata generation utility.

To create the schema file:

1. Create a .properties file.
2. Add entries to the file by using the account qualifiers listed in the following table to define the schema of your target system.

Table 2-1 Account Qualifiers

Account Qualifier	Mandatory?	Description
FieldNames	Yes	Enter a comma-separated list of attributes that the connector must fetch from the target system. You must specify all single-valued and multivalued attributes, including the attribute used for performing incremental reconciliation here. You must also specify any child form names here. However, do not enter child form field names here as you specify them by using the Subfields qualifier which is discussed later in this guide. **Note:** For target system attributes that have corresponding attributes predefined in ICF, Oracle recommends to use the predefined ICF attribute naming convention in the FieldNames qualifier. Suppose you want the connector to fetch groups from the target system, then you must include the group attribute in the FieldNames qualifier. ICF has a corresponding predefined attribute for groups named __GROUP__ . Therefore, we recommend that you specify __GROUP__ instead of Groups. For a complete list of predefined ICF attributes, see *Oracle Fusion Middleware Java API Reference for Identity Connector Framework*. Sample entry: `FieldNames=id,primaryEmail,password,name.familyName,name.givenName,alias,phones,__GROUP__,isAdmin,changePasswordAtNextLogin,orgUnitPath,suspended`
UidAttribute	Yes	Enter the name of the attribute that corresponds to the unique ID of the account. The connector uses this value to uniquely identify target system user accounts that it needs to fetch during reconciliation. The connector also uses this value to uniquely identify target system user accounts during update and delete provisioning operations. The value of this qualifier corresponds to the __UID__ attribute of the connector. Sample entry: `UidAttribute=id`
NameAttribute	Yes	Name of the attribute that corresponds to a descriptive name of the account in the target system that the connector uses for performing reconciliation and update provisioning operations. This value corresponds to the __NAME__ attribute of the connector and is used to generate the reconciliation rule. Sample entry: `NameAttribute=UserName`
PasswordAttribute	Yes, if your target system supports password management of user accounts	Name of the password attribute of the account that the connector uses during Change Password provisioning operations. The value of this qualifier corresponds to the __PASSWORD__ attribute of the connector. Sample value: `PasswordAttribute=accountPwd`

Table 2-1 (Cont.) Account Qualifiers

Account Qualifier	Mandatory?	Description
StatusAttribute	Yes, if your target system supports management of user account statuses	This qualifier refers to the attribute which denotes the status of the account. The connector uses this attribute during provisioning operations to enable or disable user accounts. In addition, the connector uses this attribute to fetch the status of an account during Status reconciliation. The value of this qualifier corresponds to the __ENABLE__ attribute of the connector. Sample value: `StatusAttribute=status`

3. If required, using the field qualifiers, add entries to the file to include details about fields added in Step 2.

> **Note:**
>
> All the field qualifiers are optional.

Table 2-2 Field Qualifiers

Field Qualifier	Description
Required	This field qualifier specifies if the mentioned attribute is mandatory. If the value of this qualifier is set to `true`, the parser will skip processing the records that do not contain this field name. For example: `UserId.Required=true`
Multivalued	This field qualifier specifies if the mentioned attribute is a multivalued field. For example: `Roles.Multivalued=true`
DataType	This field qualifier is used to specify the datatype of the field name. If you do not specify the data type for any field, then it is considered as a String data type by default. The following are the possible values for this qualifier: • String • Long • Character • Double • Float • Integer • Boolean • Byte • BigDecimal • BigInteger • Date For example: `startDate.DataType=Date`

Table 2-2 (Cont.) Field Qualifiers

Field Qualifier	Description
EmbeddedObjectClass	This field qualifier specifies the object class name of child forms that have more than one subfield. The value of this qualifier is used internally by ICF and is mandatory for all complex child forms. For example: `Roles.EmbeddedObjectClass=Roles`

4. Save the created .properties file.

> **See Also:**
>
> Sample Schema Filefor a sample ACME schema file

2.3 Updating the Groovy File

Update the Groovy file by specifying values for properties that store information about your target system schema and the mode in which you want to configure the connector.

To update the GenericRestConfiguration.groovy file:

1. Extract the contents of the connector installation package to any directory on the computer hosting OIM. This creates a directory named GenericREST-*RELEASE_NUMBER*. See Files and Directories of the Generic REST Connector for information about all the files and directories in the connector installation package.
2. In a text editor, open the GenericRestConfiguration.groovy file located in the GenericREST-*RELEASE_NUMBER*/metadata-generator/resources directory.
3. Depending on whether you want to run the connector in the trusted source mode or target resource mode, enter values for the entries in the configuration section. You can use the existing sample configuration or create your own section.

> **Note:**
>
> - The presence of the trusted=true entry in a configuration section indicates that the section is for configuring the connector to run in the trusted source mode. If trusted=true does not exist or trusted=false, then the configuration section is for running the connector in the target resource mode.
> - If you do not want to specify a value for any of the optional entries or attributes in the GenericRestConfiguration.groovy file, then comment out that entry or attribute by prefixing it with the double-slash symbol (//).

Table 2-3 Entries in the GenericRestConfiguration.groovy File

Entry	Applicable to trusted, target, or both?	Mandatory?	Description
trusted	Trusted source	Yes	This entry indicates whether the configuration section is for trusted source mode or target resource mode. Set the value of this entry to `true`, if you are configuring the connector to run in the trusted source mode.
itResourceDefName	Both	Yes	Enter the name of the IT resource type for the target system. Note that the value that you specify for this entry determines the name of the connector package, connector configuration file, and connector installer file. For example, if you specify `GenRestTrusted` as the value of this entry, then the name of the connector package directory is `GenRestTrusted.zip`. See Understanding the Generator Connector Package for the directory structure of the connector package.
itResourceName	Both	No	Enter the name of the IT resource for the target system. If this entry is commented, then the IT resource name will be the same as the value of the ITResourceDefName entry. Default value: `"$itResourceDefName"` **Note:** The value of this entry must be unique for each connector that you create for your target system, if you plan to install or use two or more Generic REST connectors in the same OIM environment. In addition, this value will be a part of the names for all connector components (defined in the connector configuration XML file, which is created after you run the metadata generator) such as lookup definitions, resource objects, process forms, and scheduled tasks. For example, if you specify `GenRestTrusted` as the value of itResourceName entry, then after you deploy the connector, the configuration lookup definition is created and its name will be Lookup.GenRestTrusted.Configuration.
connectorDir	Both	No	This entry is the complete path to the directory that must contain the connector package that is generated when you run the metadata generator. By default, the name of the directory containing the generated connector package is the same as the value of the itResourceDefName entry. Sample value: `"/scratch/jdoe/OIMPS3/mw4318/idm7854/server/ConnectorDefaultDirectory/GenRestTrusted"`

Table 2-3 (Cont.) Entries in the GenericRestConfiguration.groovy File

Entry	Applicable to trusted, target, or both?	Mandatory?	Description
xmlFile	Both	No	Enter the name and relative path of the XML file that must contain definitions of the connector objects. If you do not specify a value for this entry, then the file name is generated in the following format: *IT_RES_DEF_NAME*-ConnectorConfig.xml In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry. For example, if you have not specified a value for this entry and `GenRestTrusted` is the value of the itResourceDefName entry, then the name of the XML file that is generated is `GenRestTrusted-ConnectorConfig.xml.` **Tip:** To easily identify files of a specific target system installation, prefix the names of this generated XML file with the name of the IT resource for the target system. Sample value: `GenRestTrusted-ConnectorConfig.xml`
configFileName	Both	No	Enter the name and relative path of the XML file that contains the configuration information of the connector objects. If you do not specify a value for this entry, then the file name is generated in the following format: *IT_RES_DEF_NAME*-CI.xml In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry. For example, if you have not specified a value for this entry and `GenRestTrusted` is the value of the itResourceDefName entry, then the name of the XML file that is generated is `GenRestTrusted-CI.xml.`
propertiesFile	Both	No	Enter the name and relative path of the .properties file which contains the resource bundle translations. If you do not specify a value for this entry, then the file name is generated in the following format: *IT_RES_DEF_NAME*-generator.properties In this format, *IT_RES_DEF_NAME* is the value of the itResourceDefName entry. For example, if you have not specified a value for this entry and `GenRestTrusted` is the value of the itResourceDefName entry, then the name of the properties file that is generated is `GenRestTrusted-generator.properties.`
version	Both	No	Enter the release number of the connector. Sample value: `11.1.1.5.0`

Table 2-3 (Cont.) Entries in the GenericRestConfiguration.groovy File

Entry	Applicable to trusted, target, or both?	Mandatory?	Description
bundleJar	Both	Yes	The value of this entry indicates the name and relative path of the JAR file containing the ICF bundle that the metadata generator will use. Default value: `../lib/org.identityconnectors.genericrest-1.0.11150.jar` Do *not* change the value of this entry.
config	Both	Yes	Specify the manner in which the connector must behave and connect to the target system. By default, the config entry contains only the schemaFile property. The schemaFile property is a mandatory property. Enter the name and relative path of the schema file that you want to use.
dateAttributeList	Both	No	Enter the list of attributes that must be handled as date on the process form. Ensure that the data type of the attributes listed here is set to Long in the schema file. For more information about the format in which you must enter values for this entry, see About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File.
alias	Both	Yes	Depending on whether you are configuring the Groovy file for the target resource mode or trusted source mode, you must specify a value for this entry. The metadata generator uses aliases to create relationships between the attributes in the target system and resource object field names in Oracle Identity Manager. In addition, the metadata generator uses aliases to shorten long database names to meet the character-length restrictions on form names and form field names in Oracle Identity Manager. For detailed information about this entry and the values that you can enter, see About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File.
applicationInstanceName	Target resource	No	Enter the name of the application instance for your target system that the connector must generate. If this entry is commented, then the application instance name will be the same as the value of the ITResourceDefName entry. Default value: `"$itResourceDefName"`

Table 2-3 (Cont.) Entries in the GenericRestConfiguration.groovy File

Entry	Applicable to trusted, target, or both?	Mandatory?	Description
entitlementAttributeList	Target resource	No	Enter the list of attributes in the target system that must be tagged as entitlements. For more information about the format in which you must enter values for this entry, see About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File.
lookupAttributeList	Target resource	No	Enter the list of attributes in your target system that must be handled as lookup fields. For more information about the format in which you must enter values for this entry, see About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File.
prepopulate	Target resource	No	Specify a value for this entry if you want Oracle Identity Manager to prepopulate connector's process form fields from OIM User fields while provisioning an enterprise target system resource. The default value of this entry is as follows: `['__NAME__':'User Login', 'FIRST_NAME':'First Name', 'LAST_NAME':'Last Name', '__PASSWORD__':'Password']` This means that the groovy file is configured to prepopulate the following fields by default: • User Login • First Name • Last Name • Password You can add fields to or remove fields from the preceding list. The following is the format in which you must specify values for the prepopulate entry: ['*ATTR1*': '*OIM_FIELD1*', '*ATTR2*': '*OIM_FIELD2*', . . . '*ATTRn*': '*OIM_FIELDn*'] In this format: • *ATTR* can be either the connector attribute name or the target system attribute name. • *OIM_FIELD* is the name of the field on the OIM User form. See Working with Prepopulate Adapters in *Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager* for more information about attaching and removing prepopulate adapters.

4. Save and close the GenericRestConfiguration.groovy file.

2.3.1 About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File

Learn about the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias entries of the Groovy file. You use the information provided here while updating the Groovy file.

dateAttributeList

This entry holds the list of attributes that the connector must handle as date on the process form. Ensure that the data type of the attributes listed here is set to `Long` in the schema file.

The connector creates a date editor for each of the attributes specified in this entry.

If you want to the connector to handle single-valued or multivalued fields as date, then enter the value for this entry in the following format:

`["`*`FIELD_NAME`*`"]`

In this format, replace *FIELD_NAME* with the name of the single or multivalued field.

If you want to handle an embedded multivalued field as date, then enter the value in the following format:

["*OBJ_CLASS*.*SUB_FIELD_NAME*"]

In this format, replace:

- *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
- *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

Default value: `["JoiningDate"]`

You can modify the default value to meet the requirements in your environment.

The following is a sample value for handling embedded multivalued fields as date:

```
["MyRole.StartDate", "MyRole.EndDate"]
```

entitlementAttributeList

This entry must include the list attributes in the target system that must be tagged as entitlements.

The connector creates a lookup field for each of the attributes specified in this entry, assigns the lookup fields to a process form, and adds all the required properties of entitlements.

If you want to tag entitlements for multivalued fields, then enter the value in the following format:

["*MULTIVALUED_FIELD_NAME*"]

If you want to tag entitlements for a multivalued field that is embedded, then enter the value in the following format:

["*OBJ_CLASS*.*SUB_FIELD_NAME*"]

In this format, replace:

- *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
- *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

Default value: `['Roles.RoleName']`

In this value, `Roles.RoleName` is a multivalued field that is embedded. In other words, `Roles` is the EmbeddedObjectClass name for roles child form as specified in the schema file (that is, roles.EmbeddedObjectClass=Roles) and `RoleName` is one of the subfields for the roles child form as specified in the schema file (that is roles.Subfileds=ROLENAME).

You can modify the default value to meet the requirements in your environment.

lookupAttributeList

This entry holds the list of attributes in your target system that the connector must be handle as lookup fields.

The connector creates a lookup field for each of the attributes specified in this entry and associates it with the corresponding lookup fields on the OIM User process form.

If you want to create a lookup field for a single-valued or multivalued field, then enter the value in the following format:

['*FIELD_NAME*']

In this format, replace *FIELD_NAME* with the name of the single or multivalued field.

If you want create a lookup field for a multivalued field that is embedded then, enter the value in the following format:

['*OBJ_CLASS*.*SUB_FIELD_NAME*']

In this format, replace:

- *OBJ_CLASS* with the EmbeddedObjectClass name for the child form as specified in the schema file.
- *SUB_FIELD_NAME* with the subfield name for the child form as specified in the schema file.

The default value of this entry is:

```
['Currency']
```

In this value, `Currency` is a multivalued field.

You can modify the default value to meet the requirements in your environment.

For each attribute listed in the lookupAttributeList entry, the connector creates a lookup definition and scheduled job in the following format:

- Lookup definition format:

  Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*}

  This lookup definition holds the lookup values reconciled from the target system.

- Scheduled job format:

  IT_RES_NAME Target *FIELD_NAME* Lookup Reconciliation

  This scheduled job is used to load or reconcile lookup values from your target system. See Scheduled Job for Lookup Field Synchronization for more information about the attributes of the scheduled job for lookup reconciliation.

In both the formats, the connector replaces:

- *IT_RES_NAME* with the value of the itResourceDefName entry.
- *FIELD_NAME* with the name of the field for which the lookup field is created.

alias

The metadata generator uses aliases to create relationships between the attributes in the target system and resource object field names in Oracle Identity Manager. In addition, the metadata generator uses aliases to shorten long database names to meet the character-length restrictions on form names and form field names in Oracle Identity Manager. Aliasing can be used on column name, form name, and form field name levels. Note that the target system attributes are represented as connector attributes.

Depending on the type of configuration, specify values for one of the following sections:

- For trusted source configuration

  In the trusted source configuration section, you use the alias entry to map connector attributes or target system attributes to the OIM User form field names. The mappings that you specify here are used to populate entries in the Recon Attribute map lookup definition for trusted source reconciliation.

  Note that some of the OIM User form field names do not have the same display name internally. For such fields, you must ensure that you map the connector attribute or target system attribute to the internal name rather than the display name. The following table lists the names of the OIM User form display names and their corresponding internal names:

Display Name	Internal Name
Organization	Organization Name
Manager	Manager Login
E-mail	Email

  The following is the default value of the alias entry:

```
['__NAME__':'User Login', 'LastName':'Last
Name','Organization':'Organization Name', 'Employee Type':'Xellerate
Type', 'Role':'Role']
```

  In the default value, note that the "Organization" connector attribute has been mapped to "Organization Name", which is the internal name.

  You cannot delete existing mappings in the default value. However, you can modify these mappings.

  If you want to add mappings for fields other than the ones already present in the alias entry, then you can add them either to the existing values in the alias entry, or add them to the alias + entry.

The following is the default value of the alias + entry:

```
['__ENABLE__':'Status', 'FirstName':'First Name', 'email':'Email',
'JoiningDate':'Start Date']
```

The following is the format in which you must specify values for the alias and alias + entry:

['*CONN_ATTR1*': '*OIM_FIELD1*', '*CONN_ATTR2*': '*OIM_FIELD2*', . . . '*CONN_ATTRn*': '*OIM_FIELDn*']

In this format:

- *CONN_ATTR* is the connector attribute name.
- *OIM_FIELD* is the name of the field on the OIM User form.

- For target resource configuration

  In the target resource configuration section, you use the alias entry for one or all of the following purposes:

  - To map connector attributes or target system attributes to fields of the process form. The mappings that you specify here are used to populate entries in the Recon Attribute map and Prov Attribute map lookup definitions that are used during target resource reconciliation.
  - To set an alias (a unique and shortened name) for the IT resource name specified in the itResourceName entry.
  - To specify a short name for a lengthy process form field name.

    When the number of characters in a process form is more than 11, the metadata generator automatically truncates the process form name to 10 characters and then suffixes it with the digit 0. Subsequently, for every process form that results in the same name after truncating, the suffix is incremented by 1. The metadata generator prevents any two process forms from having the same name by using autonumbering. To gain control over the autogenerated form name and to have meaningful form names, you can use an alias to specify a shortened process form name.

    This is illustrated by the following example:

    Assume that the resource name is GENDB and contains child data that is represented as USER_ROLES in the schema.

    When you run the metadata generator, the process form is created and the form name is UD_GENDB_USER_ROLES. As the number of characters in this process form name is more than 11, the metadata generator automatically truncates it to UD_GENDB_U0. The truncated form name, UD_GENDB_U0, is not meaningful.

    To avoid encountering such issues or forms with autogenerated names, you can use the alias entry to specify short and meaningful process form names.

    The following is the default value of the alias entry in the target resource configuration section:

    ```
    ['__UID__':'id', '__NAME__':'primaryEmail']
    ```

    You cannot delete existing mappings in the default value as they are mandatory. However, you must modify the default value to match the values of the UidAttribute and NameAttribute qualifiers in the schema file. For example, in the schema file, if you have set the values of the UidAttribute

and NameAttribute qualifiers to EmpId and UserName respectively, then you must set the value of the alias entry to the following:

```
['__UID__':'EmpId', '__NAME__':'UserName']
```

If you want to add mappings for fields other than the ones already present in the alias entry (in other words, optional aliases), then you can add them either to the existing values in the alias entry, or add them to the alias + entry.

The following is the default value of the alias + entry in the target resource section:

```
['USERROLERELATIONSHIP':'USRROL', 'comments':'Description', 'Family Name':'Last Name', 'Visibility':'Status']
```

The following is the format in which you must specify values for the alias and alias + entries:

['*CONN_ATTR1*': '*ALIAS_FIELD1*', '*CONN_ATTR2*': '*ALIAS_FIELD2*', . . . '*CONN_ATTRn*': '*ALIAS_FIELDn*']

In this format:

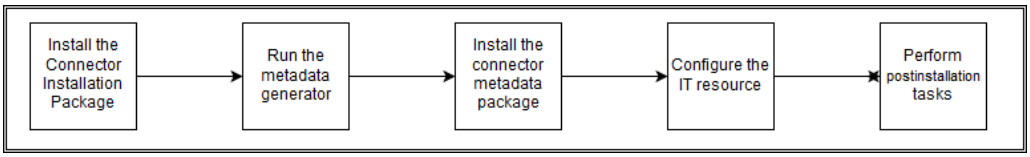
* *CONN_ATTR* is the connector attribute name.
* *ALIAS_FIELD* is the alias corresponding to the connector attribute or target system attribute.

3
Installing the Generic REST Connector

You must install the connector in Oracle Identity Manager. If necessary, you can also deploy the connector in a Connector Server.

The Generic REST connector installation can be divided into various stages, as illustrated in the following figure:

Figure 3-1 Stages of Connector Installation



Installation information is divided across the following sections:

- Installing the Connector Installation Package
- Generating and Installing the Connector Metadata Package

3.1 Installing the Connector Installation Package

You must install the connector installation package by running the Connector Installer. When you run the Connector Installer, it automatically copies configuration files to Oracle Identity Manager and uploads the connector bundle, org.identityconnectors.genericrest-1.0.1115.jar, to the Oracle Identity Manager database.

If your target system uses an authentication mechanism that is not supported by this connector, then before you install the connector installation package, you must implement the authentication that your target system uses as described in Implementing Custom Authentication.Similarly, if the reconciliation data from your target system is not in JSON format, then you must write a custom parser implementation for your data format as described in Implementing Custom Parsing.

To install the connector installation package:

1. Copy the contents of the unzipped connector installation media to the following directory:

   OIM_HOME/server/ConnectorDefaultDirectory
2. Log in to Oracle Identity System Administration.
3. In the left pane, under Provisioning Configuration, click **Manage Connector**.
4. In the Manage Connector page, click **Install**.
5. From the Connector List, select **Generic REST Connector-*RELEASE_NUMBER*.**.

This list displays the names and release numbers of connectors whose installation files you copy into the default connector installation directory in Step 1

If you have copied the installation files into a different directory, then:

a. In the **Alternative Directory** field, enter the full path and name of that directory.

b. To repopulate the list of connectors in the Connector List list, click **Refresh**.

c. From the Connector List list, select the relevant connector name depending on whether you are installing the connector included in the connector installation media or the generated connector.

6. Click **Load**.

7. To start the installation process, click **Continue**.

The following tasks are performed in sequence:

a. Configuration of connector libraries

b. Import of the connector XML files (Using Deployment Manager).

c. Compilation of Adapter Definitions

On successful completion of a task, a check mark is displayed for the task. If a task fails, then an X mark and a message stating the reason for failure are displayed. Depending on the reason for the failure, make the required correction and then perform one of the following steps:

- Retry the installation by clicking **Retry**.

  Cancel the installation and begin again from Step 3.

If all three tasks of the connector installation process are successful, then a message indicating successful installation is displayed.

8. Click **Exit** to close the installation page.

3.2 Generating and Installing the Connector Metadata Package

You must run the Metadata Generator to generator the connector metadata package and then install it.

Topics

3.2.1 Running the Metadata Generator

You generate a connector metadata package for your target system by running the metadata generator.

You invoke the metadata generator from the command line by using the executable file named GenericRestGenerator.cmd or GenericRestGenerator.sh.

To run the metadata generator, in a command window, change to the genericrest-*RELEASE_NUMBER*/metadata-generator/bin directory (for example,

genericrest-11.1.1.5.0/bin) and run one of the following commands depending on the operating system that you are using:

- **For Microsoft Windows**

  ```
  GenericRestGenerator.cmd CONFIG_FILE CONFIG_NAME
  ```

- **For UNIX**

  ```
  GenericRestGenerator.sh CONFIG_FILE CONFIG_NAME
  ```

In this command, replace:

- *CONFIG_FILE* with the absolute or relative path name of the GenericRestConfiguration.groovy file.
- *CONFIG_NAME* with the name of the configuration section within the GenericRestConfiguration.groovy file, being used for the target system. The sample configuration within this file are trusted and target. You can create additional custom configurations with different names depending on your requirements.

The following is a sample command:

```
GenericRestGenerator.cmd ..\resources\GenericRestConfiguration.groovy
target
```

In this command, "target" denotes the name of the section in the GenericRestConfiguration.groovy file for which you have specified values. .

If you encounter any errors while running the metadata generator, then you must fix it and then resume running the metadata generator.

3.2.2 Installing the Connector Metadata Package

You must install the connector metadata package by running the Connector Installer. When you run the Connector Installer, it automatically imports the metadata files into Oracle Identity Manager, generates all the connector components, and compiles adapters used for provisioning.

To install the connector metadata package:

1. Copy the contents of the unzipped connector metadata package to the following directory:

   OIM_HOME/server/ConnectorDefaultDirectory

2. Log in to Oracle Identity System Administration.
3. In the left pane, under Provisioning Configuration, click **Manage Connector**.
4. In the Manage Connector page, click **Install**.
5. From the Connector List, select the name of the connector metadata package that you generated.

   This list displays the names and release numbers of connectors whose installation files you copy into the default connector installation directory in Step 1

   If you have copied the installation files into a different directory, then:

   a. In the **Alternative Directory** field, enter the full path and name of that directory.

   b. To repopulate the list of connectors in the Connector List list, click **Refresh**.

c. From the Connector List list, select the relevant connector name depending on whether you are installing the connector included in the connector installation media or the generated connector.

6. Click **Load**.
7. To start the installation process, click **Continue**.

   The following tasks are performed in sequence:

   a. Configuration of connector libraries
   b. Import of the connector XML files (Using Deployment Manager).
   c. Compilation of Adapter Definitions

   On successful completion of a task, a check mark is displayed for the task. If a task fails, then an X mark and a message stating the reason for failure are displayed. Depending on the reason for the failure, make the required correction and then perform one of the following steps:

   - Retry the installation by clicking **Retry**.

     Cancel the installation and begin again from Step 3.

   If all three tasks of the connector installation process are successful, then a message indicating successful installation is displayed.
8. Click **Exit** to close the installation page.

3.2.3 Understanding the Generated Connector Metadata Package for the Generic REST Connector

The connector metadata package is a ZIP file that is generated in the GenericRest-*RELEASE_NUMBER*/metadata-generator/ directory.

For example, if you have specified GenRest as the value of the itResourceDefName entry in the GenericRestConfiguration.groovy file, then the connector package ZIP (GenRest.zip) file is generated in the GenericRest-11.1.1.5.0/metadata-generator/ directory. The directory structure of the connector metadata package is as follows:

```
CONNECTOR_METADATA_PCKG/
        configuration/
                IT_RES_DEF-CI.xml
        resources/
                genericrest-generator.properties
        xml/
               IT_RES_DEF-ConnectorConfig.xml
```

In this directory structure:

- *CONNECTOR_METADATA_PCKG* is replaced with the name of the IT resource definition specified as the value of the itResourceDefName entry in the GenericRestConfiguration.groovy file.
- *IT_RES_DEF* is replaced with the name of the IT resource definition specified as the value of the itResourceDefName entry in the GenericRestConfiguration.groovy file.

The following behavior is observed after generation of the connector configuration XML file:

The length of a field (column) from the target system is not fetched into the process form. Therefore, except for the Unique ID and Password fields, the length of all other data fields (of the String data type) on the process form is always set to 255 characters. The length of the Unique ID and Password fields is set to 40 characters.

4
Configuring the Generic REST Connector

You must configure the connector to let Oracle Identity Manager connect to the target system. You must also configure the connector to set it up to perform reconciliation and provisioning operations.

Topics

4.1 Configuring the IT Resource for the Target System

The IT resource for your target system is created after you install the connector. An IT resource is composed of parameters that store connection and other generic information about a target system. Oracle Identity Manager uses this information to connect to a specific installation or instance of your target system and perform reconciliation and provisioning operations.

The list of IT resource parameters for the Generic REST connector can be grouped into the following categories:

- Connection-related parameters
- Authentication parameters
- Parser parameters
- Additional configuration parameters

This section provides information about the following topics related to IT resource configuration:

4.1.1 Specifying Values for the IT Resource Parameters

The IT resource for the target system contains connection information about the target system. Oracle Identity Manager uses this information during provisioning and reconciliation.

When you run the metadata generator, the IT resource corresponding to this connector is automatically created in Oracle Identity Manager. You must specify values for the parameters of this IT resource as follows:

1. Log in to Oracle Identity System Administration.

2. In the left pane, under Configuration, click **IT Resource.**
3. In the IT Resource Name field on the Manage IT Resource page, enter the name of the IT resource, and then click **Search.** The name of the IT resource is the value of the itResourceName property in the GenericRestConfiguration.groovy file.
4. Click the edit icon for the IT resource.
5. From the list at the top of the page, select **Details and Parameters**.
6. Specify values for the parameters of the IT resource. See Configuring the IT Resource for the Target System for information about IT resource parameters.
7. To save the values, click **Update.**

4.1.2 Connection Parameters

Connection parameters are used by the connector to establish a connection between Oracle Identity Manager and your target system for exchange of identity information.

Table 4-1 lists the connection-related IT resource parameters.

Table 4-1 Connection IT Resource Parameters

Parameter	Description
Configuration Lookup	Name of the lookup definition that holds connector configuration entries that are used during connector operations. **Note:** This is a mandatory parameter.
Connector Server Name	If you have deployed the Generic RESTconnector in the Connector Server, then enter the name of the IT resource for the Connector Server.
host	Host name or IP address of the computer hosting the target system. Sample value: `www.example.com` **Note:** This is a mandatory parameter.
port	Port number at which the target system is listening. Sample value: `80` **Note:** This is a mandatory parameter.
proxyHost	Name of the proxy host used to connect to an external target system. Sample value: `www.example.com`
proxyPort	Proxy port number Sample value: `80`
proxyUser	Proxy user name of the target system user account that Oracle Identity Manager uses to connect to the target system.
proxyPassword	Password of the proxy user ID of the target system user account that Oracle Identity Manager uses to connect to the target system.

Table 4-1 (Cont.) Connection IT Resource Parameters

Parameter	Description
connectionTimeOut	An integer value that specifies the number of milliseconds after which an attempt to establish the connection between the target system and Oracle Identity Manager times out. Sample value: `100000`
socketTimeOut	An integer value that specifies the number of milliseconds after which the wait for a response from the target system times out. Sample value: `100000`

4.1.3 Authentication Parameters

Authentication parameters are used by the target system to authenticate an application. The set of IT resource parameters for which you must specify values depends on the value that you enter for the authenticationType parameter.

The authenticationType parameter holds the type of authentication used by your target system. By default, the connector supports the following types of authentication:

- HTTP Basic Authentication
- OAuth 2.0 JWT
- OAuth 2.0 Client Credentials
- OAuth 2.0 Resource Owner Password
- Manually input access token and refresh token

Apart from the authentication types listed, if you target system uses any other authentication type, then you must write your own implementation which requires development effort. The following are the possible values for the authenticationType parameter:

- For HTTP Basic Authentication: `basic`
- For OAuth 2.0 JWT: `jwt`
- For OAuth 2.0 Client Credentials: `client_credentials`
- For OAuth 2.0 Resource Owner Password: `password`
- For manual input of access token and refresh token: `other`
- For custom authentication implementation: `custom`

> **Note:**
>
> This section provides information about IT resource parameters for all authentication types. Enter values only for IT resource parameters corresponding to the authentication type you specify.

HTTP Basic Authentication

Table 4-2 lists the set of IT resource parameters for which you must enter values when the authenticationType parameter is set to `basic`.

Table 4-2 HTTP Basic Authentication IT Resource Parameters

Parameter	Description
username	User name or User ID of the account that Oracle Identity Manager must use to connect to and access the target system during reconciliation and provisioning operations. Sample value: `johnsmith`
password	Password of the account that Oracle Identity Manager must use to connect to and access the target system during reconciliation and provisioning operations. Sample value: `password`

OAuth 2.0 JWT

Table 4-3 lists the set of IT resource parameters for which you must enter values when the authenticationType parameter is set to `jwt`.

Table 4-3 OAuth 2.0 JWT IT Resource Parameters

Parameter	Description
aud	Enter the intended audience of the JWT. The value can either be a URI or token endpoint URL of the authorization server. Sample value: `https://www.example.com/oauth2/v3/token`
iss	Enter a value that uniquely identifies the entity that issued the JWT. Sample value: `527901474-ugnvd5uh21p598cf9h6cd@developer.example.com`
scope	Enter the scope of the access token being issued. Sample value: `https://www.example.com/auth/adm.direct.group`, `https://www.example.com/auth/adm.direct.user`
sub	Enter a value that identifies the principal to which the JWT is being issued. Sample value: `admin@example.com`
privateKeyLocation	Enter the absolute path to the private key used to sign the access token. Sample value: `C:\Users\jdoe\Desktop\Connector_Server_111210\connector_server_java-1.4.0\bundles\googleapps.p12`

Table 4-3 (Cont.) OAuth 2.0 JWT IT Resource Parameters

Parameter	Description
privateKeySecret	Enter the secret key for the private key that is being used to sign the access token.
tokenLifespan	Enter the life span of the access token in milliseconds. Sample value: `3600`
signatureAlgorithm	Enter the algorithm used for signing the access token. Sample value: `RS265`
privateKeyFormat	Enter the format of the private key used to sign the access token. Sample value: `PKCS12`

OAuth 2.0 Client Credentials

Table 4-4 lists the set of IT resource parameters for which you must enter values when the authenticationType parameter is set to `client_credentials`.

Table 4-4 OAuth2.0 Client Credentials IT Resource Parameters

Parameter	Description
clientId	Enter the client identifier (a unique string) issued by the authorization server to the client during the registration process. Sample value: `XDWTh0r2eWuULCDVt`
clientSecret	Enter the value used to authenticate the identity of your client application. Sample value: `clZsdZisTOoYN5NITirarIDepDkiJTGHdzNFT0m`
authenticationServerURL	Enter the URL of the authorization server that authenticates the client (by validating the client ID and client secret), and if valid, issues an access token. Sample value: `https://api.example.com/oauth2/token`

OAuth 2.0 Resource Owner Password

Table 4-5 lists the set of IT resource parameters for which you must enter values when the authenticationType parameter is set to `password`.

Table 4-5 OAuth 2.0 Resource Owner Password IT Resource Parameters

Parameter	Description
username	Enter the user name or user ID of the resource owner. Sample value: `johnsmith`
password	Enter the password of the resource owner. Sample value: `password`
clientId	Enter the client identifier issued to the client during the registration process. Sample value: `XDWTh0r2eWuULCDVt` **Note:** This is an optional parameter.
clientSecret	Enter the client secret used to authenticate the identity of the client application. Sample value: `clZsdZisTOoYN5NITirarIDepDkiJTGHdzNFT0m` **Note:** This is an optional parameter.
authenticationServerUrl	Enter the URL of the authorization server (token endpoint) that authenticates the client (by validating client ID and client secret) and the resource owner credentials, if valid, issues an access token. Sample value: `https://api.example.com/oauth2/token`

Manual Input of Access Tokens and Refresh Tokens

This section discusses the IT resource parameter for which you must enter a value when the authenticationType parameter is set to `other`.

In this authentication mechanism, the connector expects the value of the access token and refresh token to be directly passed through the customAuthHeaders IT resource parameter. The customAuthHeaders parameter must hold the access token and refresh token values that must be passed through an HTTP authorization header.

Custom Authentication

This section discusses the IT resource parameter for which you must enter a value when the authenticationType parameter is set to `custom`.

If you have implemented custom authentication, then you must enter a value for the customAuthClassName parameter. The customAuthClassName parameter must hold the name of the class implementing the custom authentication logic that you created while performing the procedure described in Implementing Custom Authentication.

4.1.4 Parser Parameters

By default, the Generic REST connector supports only JSON parsing during reconciliation runs. If the reconciliation data from your target system is not in JSON format, then you must write a custom parser implementation for your data format.

Table 4-6 lists the IT resource parameters related to parsing

Table 4-6 Parser IT Resource Parameters

Parameter	Description
jsonResourcesTag	Enter the JSON tag value that is used for parsing a response payload. The connector will consider the value that you enter in this parameter as an unwanted outer tag while parsing responses. You can skip entering a value for this parameter if there is no unwanted outer tag in your response payload. Enter a value for this parameter in the following format: *OBJ_CLASS=OUTER_ATTR_NAME* In this format, *OBJ_CLASS* is the name of the object class for which a response payload is being parsed. *OUTER_ATTR_NAME* is the name of the outer tag in the response payload. For example, consider the following JSON value for a User object: `"Resources":"{` `"user1":"{value1}",` `"user2":"{value2}"` `}` Because the name of the object class for a User object is __ACCOUNT__, for the given example, the value of the jsonResourcesTag parameter is `__ACCOUNT__=Resources`. **Note:** You must enter a value for this parameter only if the data from your target system is in JSON format. For more than one JSON tag, the values must be comma separated.
customParserClassName	Enter the name of the class implementing the custom parser logic that you created while performing the procedure described in Implementing Custom Parsing. **Note:** Enter a value for this parameter only if you are using a custom parser implementation.

4.1.5 Additional Configuration Parameters

All additional configuration IT resource parameters are target system specific. Table 4-7 lists the IT resource parameters related to target system configuration. The supported operation types for all the parameters listed in this table are CREATEOP, DELETEOP, SEARCHOP, UPDATEOP, ADDATTRIBUTE, and REMOVEATTRIBUTE. You must use the ADDATTRIBUTE or REMOVEATTRIBUTE operations only if you want to use a separate request payload for each child table add and remove operation. Otherwise, use the UPDATEOP or DELETEOP operations.

> **Note:**
>
> In this guide, attributes in an object class that can be managed only through a separate rest endpoint rather than the same endpoint of the base object class have been referred to as **special attributes**.

Table 4-7 Configuration IT Resource Parameters

Parameter	Description
simpleMultivaluedAttributes	Enter a comma-separated list of simple multivalued attributes to be managed by the connector. You must enter values for this parameter in the following format: *OBJ_CLASS.ATTR_NAME* Sample value: `"__ACCOUNT__=alias","__GROUP__=alias"`
opTypes	Enter the HTTP operation type for each object class that is to be managed by the connector. Values must be comma separated and in the following format: *"OBJ_CLASS.OPERATION=HTTP_OPERATION"* In this format, *OBJ_CLASS* is the connector object class, *OPERATION* is the connector operation (for example, CreateOp, UpdateOp, or SearchOp), and *HTTP_OP* is the HTTP operation (for example, GET, PUT, or POST). Sample value: `"__ACCOUNT__.CREATEOP=POST","__ACCOUNT__.DELETEOP=DELETE","__GROUP__.CREATEOP=POST","__ACCOUNT__.alias.CREATEOP=POST","__ACCOUNT__.alias.UPDATEOP=POST","__ACCOUNT__.alias.DELETEOP=DELETE","__GROUP__.alias.CREATEOP=POST","__GROUP__.alias.UPDATEOP=POST","__GROUP__.alias.DELETEOP=DELETE","__ACCOUNT__.__GROUP__.CREATEOP=POST","__ACCOUNT__.__GROUP__.UPDATEOP=POST","__ACCOUNT__.__GROUP__.DELETEOP=DELETE"`

Table 4-7 (Cont.) Configuration IT Resource Parameters

Parameter	Description
relURIs	Enter the relative URLs for all operations of each object class. Enter a value for this parameter in one of the following formats: • For attributes: *OBJ_CLASS.OP=REL_URL* • For special attributes: *OBJ_CLASS.ATTR_NAME.OP=REL_URL* • For attributes that have the same relative URL for multiple operations: *OBJ_CLASS=REL_URL* or *OBJ_CLASS.ATTR_NAME=REL_URL* • If you want to filter records based on a filter criteria during reconciliation, then use *$(Filter Suffix)$*. Here, replace *$(Filter Suffix)$* with the filter criteria that you specify as part of the Filter Suffix attribute of the scheduled job. • If you have to pass the unique ID of the user as part of endpoint URL, use *$(__UID__)$*. Here, replace *$(__UID__)$* with the unique ID of the user. • If you have to pass any attribute other than the unique ID of the user, then represent it in one of the following formats: – For a single-valued attribute: *$(attr_name)$* Here, replace *$(attr_name)$* with the name of the attribute in the target system. – For an embedded object: $(*OBJ_CLASS.ATTR_NAME*)$ For example, $(__GROUP__.id)$. Sample value: `"__ACCOUNT__.CREATEOP=/admin/directory/v1/users","__ACCOUNT__.SEARCHOP=/admin/directory/v1/users/$(Filter Suffix)$","__ACCOUNT__=/admin/directory/v1/users/$(__UID__)$","__GROUP__.CREATEOP=/admin/directory/v1/groups","__GROUP__=/admin/directory/v1/groups/$(__UID__)$","__GROUP__.SEARCHOP=/admin/directory/v1/groups/$(Filter Suffix)$","__ACCOUNT__.alias=/admin/directory/v1/users/$(__UID__)$/aliases","__ACCOUNT__.alias.DELETEOP=/admin/directory/v1/users/$(__UID__)$/aliases/$`

Table 4-7 (Cont.) Configuration IT Resource Parameters

Parameter	Description
	`(alias)$","__GROUP__.alias.DELETEOP=/admin/directory/v1/groups/$(__UID__)$/aliases/$(alias)$","__ACCOUNT__.__GROUP__=/admin/directory/v1/groups/$(__GROUP__.id)$/members","__ACCOUNT__.__GROUP__.DELETEOP=/admin/directory/v1/groups/$(__GROUP__.id)$/members/$(__UID__)$","__ACCOUNT__.__GROUP__.SEARCHOP=/admin/directory/v1/groups?userKey=$(__UID__)$"`
nameAttributes	Enter the name attribute for all object classes that are handled by the connector. This value specifies the mapping between the _NAME_ connector attribute and the corresponding target system attribute for each object class that the connector handles. Format: *`OBJ_CLASS.ATTR_NAME`* **Note:** All values in this parameter must be comma separated.
uidAttributes	Enter the __UID__ attribute for each object class that the connector handles. A __UID__ attribute is a target system attribute that uniquely identifies an account in the target system. This target system attribute name must be unique and need not be autogenerated. Format: *`OBJ_CLASS.ATTR_NAME`* In this format, *OBJ_CLASS* is the connector object class and *ATTR_NAME* is the name of the attribute that uniquely identifies an account in the target system. **Note:** All values in this parameter must be comma separated.
statusAttributes	Enter the mapping between the _ENABLE_ connector attribute the target attribute that holds the status for each object class this connector handles. Format: *`OBJ_CLASS.ATTR_NAME`* Sample value: `"__ACCOUNT__.suspended"` **Note:** All values in this parameter must be comma separated.

Table 4-7 (Cont.) Configuration IT Resource Parameters

Parameter	Description
statusDisableValue	Enter the boolean value that indicates the value that must be sent to the target system during a disable operation. **Note:** You must enter a value for this parameter only if the target system expects a different value for a disable operation, from what OIM sends by default. Sample values: `true` or `false` or `0` or `1`
statusEnableValue	Enter the boolean value that indicates the value that must be sent to the target system during an enable operation. **Note:** You must enter a value for this parameter only if the target system expects a different value for an enable operation, from what OIM sends by default. Sample values: `true` or `false` or `0` or `1`
specialAttributeHandling	Enter the list of special attributes whose values must be sent to the target system in separate calls, one at a time. If you do not specify a value for this parameter, then the connector will send all values for a given special attribute in a single call. Format: *`OBJ_CLASS.ATTR_NAME.OP`*`=SINGLE` **Note:** All values in this parameter must be comma separated. Sample value: `"__ACCOUNT__.alias.CREATEOP=SINGLE","__GROUP__.alias.CREATEOP=SINGLE","__ACCOUNT__.alias.UPDATEOP=SINGLE","__GROUP__.alias.UPDATEOP=SINGLE","__ACCOUNT__.__GROUP__.CREATEOP=SINGLE","__ACCOUNT__.__GROUP__.UPDATEOP=SINGLE"`

Table 4-7 (Cont.) Configuration IT Resource Parameters

Parameter	Description
specialAttributeTargetFormat	Enter the format in which a special attribute is present in the target system. Format: *OBJ_CLASS.ATTR_NAME=TARGET_FORMAT* In this example, *TARGET_FORMAT* is the format in the format of the special attribute in the target system. For example, consider the following target endpoint value: `{ "kind": "admin#directory#aliases", "etag": etag, "aliases": [ "kind": "admin#directory#alias", "id": string, "etag": etag, "primaryEmail": string, "alias": string ] }` In this example, the alias attribute is present as aliases.alias in the target system endpoint. Therefore, you must set the value of this parameter to `__ACCOUNT__.alias=aliases.alias`. Sample value: `"__ACCOUNT__.alias=aliases.alias","__GROUP__.alias=aliases.alias","__ACCOUNT__.__GROUP__=groups"`
httpHeaderContentType	This parameter holds the content type expected by the target system in the header. The content type can be `application/json`.
httpHeaderAccept	This parameter holds the accept type expected by the target system in the header. The content type can be `application/json`.
sslEnabled	If the target system requires SSL connectivity, set the value of this parameter to `true`.
customHeaders	Enter any custom or additional header values that must be sent to the target system. Format: *HEADER_NAME1=VALUE1, HEADER_NAME2=VALUE2, . . . HEADER_NAMEn=VALUEn*

Table 4-7 (Cont.) Configuration IT Resource Parameters

Parameter	Description
customAuthHeaders	Enter any additional header values that must be sent to the target system only during authentication.
customPayload	Enter a comma-separated list of request payload formats for target system attributes that do not adhere to the standard JSON format. Format: *OBJ_CLASS.ATTRNAME.OP=PAYLOAD_FORMAT* **Note:** If you must pass the unique ID of the user as part of a custom payload, then represent it as `$(__UID__)$`. If you must pass the value of any other attribute, then represent it as `$(ATTRIBUTE_NAME)$` . Sample value: `"__ACCOUNT__.__GROUP__.UPDATEOP={ \"user\": { \"id\": \"$(__UID__)$\"}, \"group\": { \"id\": \"$(id)$\" } }"`
targetObjectIdentifier	Enter the name of the attribute used for identifying the required target object during reconciliation. Format: *OBJ_CLASS.ATTR_NAME=TARGET_OBJ_ATTR_NAME;VAL*
passwordAttribute	Enter the name of the target system attribute that is mapped to the __PASSWORD__ attribute of the connector in OIM.
uriPlaceHolder	Enter a comma-separated list of key-value pairs for replacing place holders in the relURIs. Format: *KEY;VALUE*
customAuthConfigParams	Enter any configuration parameters that you may use in the custom authentication class. Format: *"PARAM_NAME1=VAL1,PARAM_NAME2=VAL2, . . . PARAM_NAMEn=VALn"*
customParserConfigParams	Enter any configuration parameters that you may use in the custom parser class. Format: *"PARAM_NAME1=VAL1,PARAM_NAME2=VAL2, . . . PARAM_NAMEn=VALn"*
enableEmptyString	Enter a boolean string which indicates that an empty string must be sent to the target system, rather than a null value.

4.2 Performing Postinstallation Tasks

Postinstallation for the connector involves configuring Oracle Identity Manager, enabling logging to track information about all connector events, and configuring SSL.

It also involves performing some optional configurations such as localizing the user interface.

This section discusses the following postinstallation procedures:

- Configuring Oracle Identity Manager
- Managing Logging for the Generic REST Connector
- Configuring SSL
- Localizing Field Labels in UI Forms
- Clearing Content Related to Connector Resource Bundles from the Server Cache

4.2.1 Configuring Oracle Identity Manager

You must create a UI form and an application instance for the resource against which you want to perform reconciliation and provisioning operations. In addition, you must run the entitlement and catalog synchronization jobs.

> **Note:**
>
> Perform the procedures described in this section *only* if you are using the connector in the target resource configuration mode.

These procedures are described in the following sections:

- Creating and Activating a Sandbox
- Creating a New UI Form
- Associating the Form with the Application Instance
- Publishing a Sandbox
- Harvesting Entitlements and Sync Catalog

4.2.1.1 Creating and Activating a Sandbox

You must create and activate a sandbox to begin using the customization and form management features. You can then publish the sandbox to make the customizations available to other users.

See Creating a Sandbox and Activating and Deactivating a Sandbox in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

4.2.1.2 Creating a New UI Form

See Creating Forms By Using the Form Designer in *Oracle Fusion Middleware Administering Oracle Identity Manager* for instructions on creating a new UI form. While creating the UI form, ensure that you select the resource object corresponding to the Generic REST connector that you want to associate the form with. In addition, select the Generate Entitlement Forms check box.

4.2.1.3 Associating the Form with the Application Instance

By default, an application instance is automatically created after you install the connector. The name of this application instance is the one that is specified as the value of the applicationInstanceName entry in the GenericRestConfiguration.groovy file. If you did not specify a value for the applicationInstanceName entry, then the application instance name will be the same as the value of the ITResourceDefName entry. You must associate this application instance with the form created in Creating a New UI Form.

See Managing Application Instances in *Oracle Fusion Middleware Administering Oracle Identity Manager* for instructions on modifying an application instance to associate it with a form.

After updating the application instance, you must publish it to an organization to make the application instance available for requesting and subsequent provisioning to users. However, as a best practice, perform the following procedure before publishing the application instance:

1. In Identity System Administration, deactivate the sandbox.
2. Log out of Identity System Administration.
3. Log in to Oracle Identity Self Service and activate the sandbox that you deactivated in Step 1.
4. In the Catalog, check for the Application Instance UI (form fields) and ensure that it appears correctly.
5. Publish the application instance only if everything appears correctly. Otherwise, fix the issues and then publish the application instance.

   See Publishing an Application Instance to Organizations in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

4.2.1.4 Publishing a Sandbox

Before you publish a sandbox, perform the following procedure as a best practice to validate all sandbox changes made till this stage as it is hard to revert changes once a sandbox is published:

1. In Identity System Administration, deactivate the sandbox.
2. Log out of Identity System Administration.
3. Log in to Identity Self Service using the xelsysadm user credentials and then activate the sandbox that you deactivated in Step 1.
4. In the Catalog, ensure that the Generic REST application instance form appears with correct fields.
5. Publish the sandbox. See Publishing a Sandbox in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager*.

4.2.1.5 Harvesting Entitlements and Sync Catalog

To harvest entitlements and sync catalog:

1. Run the scheduled jobs for lookup field synchronization discussed in Scheduled Job for Lookup Field Synchronization.

2. Run the Entitlement List scheduled job to populate Entitlement Assignment schema from child process form table.
3. Run the Catalog Synchronization Job scheduled job.

> **See Also:**
>
> Predefined Scheduled Tasks in *Oracle Fusion Middleware Administering Oracle Identity Manager*

4.2.2 Managing Logging for the Generic REST Connector

You can set a log level based on Oracle Java Diagnostic Logging and enable logging in the Oracle WebLogic Server.

The following topics contain detailed information:

- Understanding Log Levels
- Enabling Logging

4.2.2.1 Understanding Log Levels

Oracle Identity Manager uses Oracle Java Diagnostic Logging (OJDL) for logging. OJDL is based on java.util.logger. To specify the type of event for which you want logging to take place, you can set the logs to one of the following available levels:

- SEVERE.intValue()+100

  This level enables logging of information about fatal errors.
- SEVERE

  This level enables logging of information about errors that might allow Oracle Identity Manager to continue running.
- WARNING

  This level enables logging of information about potentially harmful situations.
- INFO

  This level enables logging of messages that highlight the progress of the application.
- CONFIG

  This level enables logging of information about fine-grained events that are useful for debugging.
- FINE, FINER, FINEST

  These levels enable logging of information about fine-grained events, where FINEST logs information about all events.

These log levels are mapped to ODL message type and level combinations as shown in Table 1.

Table 4-8 Log Levels and ODL Message Type:Level Combinations

Log Level	ODL Message Type:Level
SEVERE.intValue()+100	INCIDENT_ERROR:1
SEVERE	ERROR:1
WARNING	WARNING:1
INFO	NOTIFICATION:1
CONFIG	NOTIFICATION:16
FINE	TRACE:1
FINER	TRACE:16
FINEST	TRACE:32

The configuration file for OJDL is logging.xml, which is located at the following path:

DOMAIN_HOME/config/fmwconfig/servers/*OIM_SERVER*/logging.xml

Here, *DOMAIN_HOME* and *OIM_SERVER* are the domain name and server name specified during the installation of Oracle Identity Manager.

4.2.2.2 Enabling Logging

To enable logging in Oracle WebLogic Server:

1. Edit the logging.xml file as follows:
   a. Add the following blocks in the file:

```
<log_handler name='genericrest-handler' level='[LOG_LEVEL]'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
<property name='logreader:' value='off'/>
     <property name='path' value='[FILE_NAME]'/>
     <property name='format' value='ODL-Text'/>
     <property name='useThreadName' value='true'/>
     <property name='locale' value='en'/>
     <property name='maxFileSize' value='5242880'/>
     <property name='maxLogSize' value='52428800'/>
     <property name='encoding' value='UTF-8'/>
   </log_handler>

<logger name="ORG.IDENTITYCONNECTORS.GENERICREST" level="[LOG_LEVEL]"
useParentHandlers="false">
     <handler name="genericrest-handler"/>
     <handler name="console-handler"/>
   </logger>
```

   b. Replace both occurrences of **[LOG_LEVEL]** with the ODL message type and level combination that you require. Table 3-2 lists the supported message type and level combinations.

      Similarly, replace **[FILE_NAME]** with the full path and name of the log file in which you want log messages specific to connector operations to be recorded.

      The following blocks show sample values for **[LOG_LEVEL]** and **[FILE_NAME]** :

```
<log_handler name='genericrest-handler' level='NOTIFICATION:1'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
<property name='logreader:' value='off'/>
     <property name='path' value='/<%OIM_DOMAIN%>/servers/oim_server1/
logs/genericrestLogs.log>"
     <property name='format' value='ODL-Text'/>
     <property name='useThreadName' value='true'/>
     <property name='locale' value='en'/>
     <property name='maxFileSize' value='5242880'/>
     <property name='maxLogSize' value='52428800'/>
     <property name='encoding' value='UTF-8'/>
   </log_handler>

<logger name="ORG.IDENTITYCONNECTORS.GENERICREST" level="NOTIFICATION:1"
useParentHandlers="false">
     <handler name="genericrest-handler"/>
     <handler name="console-handler"/>
   </logger>
```

With these sample values, when you use Oracle Identity Manager, all messages generated for this connector that are of a log level equal to or higher than the `NOTIFICATION:1` level are recorded in the specified file.

2. Save and close the file.
3. Set the following environment variable to redirect the server logs to a file:

   For Microsoft Windows:

   ```
   set WLS_REDIRECT_LOG=FILENAME
   ```

   For UNIX:

   ```
   export WLS_REDIRECT_LOG=FILENAME
   ```

   Replace ***FILENAME*** with the location and name of the file to which you want to redirect the output.
4. Restart the application server.

4.2.3 Configuring SSL

You must configure SSL to secure data communication between Oracle Identity Manager and your target system.

To configure SSL:

1. Obtain the SSL public key certificate for the REST-based target system.
2. Copy the public key certificate of the REST-based target system to the computer hosting Oracle Identity Manager.
3. Run the following `keytool` command to import the target system certificate into the Oracle WebLogic Server keystore:

   ```
   keytool -import -keystore KEYSTORE_NAME -storepass PASSWORD -file
   CERT_FILE_NAME -alias ALIAS
   ```

   In this command:

   - *KEYSTORE_NAME* is the full path and name of the DemoTrust keystore.
   - *PASSWORD* is the password of the keystore.
   - *CERT_FILE_NAME* is the full path and name of the certificate file.

- *ALIAS* is the target system certificate alias.

The following is a sample value for this command:

```
keytool -import -keystore WEBLOGIC_HOME/server/lib/DemoTrust.jks -
storepass DemoTrustKeyStorePassPhrase –file /home/target.cert –alias
serverwl
```

> **Note:**
>
> - Change the parameter values passed to the `keytool` command according to your requirements. Ensure that there is no line break in the keytool arguments
> - Ensure that the system date for Oracle Identity Manager is in sync with the validity date of the SSL certificate to avoid any errors during SSL communication.

4.2.4 Localizing Field Labels in UI Forms

You can localize UI form field labels by creating and using a file containing localized versions for your target system fields.

To localize field label that you add to in UI forms:

1. Create a properties file (for example, GR_ja.properties) containing localized versions for the column names in your target system (to be displayed as text strings for GUI elements and messages in Identity System Administration and Identity Self Service).
2. Log in to Oracle Enterprise Manager.
3. In the left pane, expand **Application Deployments** and then select **oracle.iam.console.identity.sysadmin.ear.**
4. In the right pane, from the Application Deployment list, select **MDS Configuration.**
5. On the MDS Configuration page, click **Export** and save the archive to the local computer.
6. Extract the contents of the archive, and open the following file in a text editor:

   SAVED_LOCATION\xliffBundles\oracle\iam\ui\runtime\BizEditorBundle_en.x
7. Edit the BizEditorBundle.xlf file in the following manner:
   a. Search for the following text:

   ```
   <file source-language="en"
   original="/xliffBundles/oracle/iam/ui/runtime/
   BizEditorBundle.xlf"
   datatype="x-oracle-adf">
   ```

   b. Replace with the following text:

   ```
   <file source-language="en" target-language="LANG_CODE"
   original="/xliffBundles/oracle/iam/ui/runtime/
   ```

```
BizEditorBundle.xlf"
datatype="x-oracle-adf">
```

In this text, replace *LANG_CODE* with the code of the language that you want to localize the form field labels. The following is a sample value for localizing the form field labels in Japanese:

```
<file source-language="en" target-language="ja"
original="/xliffBundles/oracle/iam/ui/runtime/
BizEditorBundle.xlf"
datatype="x-oracle-adf">
```

c. Search for the application instance code. This procedure shows a sample edit for Generic Rest application instance. The original code is:

```
<trans-unit id="$
{adfBundle['oracle.adf.businesseditor.model.util.BaseRuntimeResou
rceBundle']
['persdef.sessiondef.oracle.iam.ui.runtime.form.model.user.entity
.userEO.UD_GENERIC_NAME_GIVEN_NAME__c_description']}">
<source>Name Givenname</source>
<target/>
</trans-unit>
<trans-unit
id="sessiondef.oracle.iam.ui.runtime.form.model.GRGAForm1.entity.
GRGAForm1EO.UD_GENERIC_NAME_GIVEN_NAME__c_LABEL">
<source>Name Givenname</source>
<target/>
```

d. Open the properties file created in Step 1 and get the value of the attribute, for example, global.udf.UD_GENERIC_NAME_GIVEN_NAME = \u4567d.

e. Replace the original code shown in Step c with the following:

```
<trans-unit id="$
{adfBundle['oracle.adf.businesseditor.model.util.BaseRuntimeResou
rceBundle']
['persdef.sessiondef.oracle.iam.ui.runtime.form.model.user.entity
.userEO.UD_GENERIC_NAME_GIVEN_NAME__c_description']}">
<source>Name Givenname</source>
<target>\u4567d</target>
</trans-unit>
<trans-unit
id="sessiondef.oracle.iam.ui.runtime.form.model.GRGAForm1.entity.
GRGAForm1EO.UD_GENERIC_NAME_GIVEN_NAME__c_LABEL">
<source>Name Givenname</source>
<target>\u4567d</target>
```

f. Repeat Steps 7.a through 7.d for all attributes of the process form.

g. Save the file as BizEditorBundle_*LANG_CODE*.xlf. In this file name, replace *LANG_CODE* with the code of the language to which you are localizing. Sample file name: BizEditorBundle_ja.xlf.

8. Repackage the ZIP file and import it into MDS.

> **Note:**
>
> See Deploying and Undeploying Customizations in *Oracle Fusion Middleware Developing and Customizing Applications for Oracle Identity Manager* for more information about exporting and importing metadata files

9. Log out of and log in to Oracle Identity Manager.

4.2.5 Clearing Content Related to Connector Resource Bundles from the Server Cache

When you deploy the connector, the resource bundles are copied from the resources directory on the installation media into the Oracle Identity Manager database. Whenever you add a new resource bundle to the connectorResources directory or make a change in an existing resource bundle, you must clear content related to connector resource bundles from the server cache.

To clear content related to connector resource bundles from the server cache you can either restart Oracle Identity Manager or run the PurgeCache utility. The following is the procedure to clear the server cache by running the PurgeCache utility:

1. In a command window, switch to the *OIM_HOME*/server/bin directory.
2. Enter one of the following commands:
   - On Microsoft Windows: `PurgeCache.bat All`
   - On UNIX: `PurgeCache.sh All`

   When prompted, enter the user name and password of an account belonging to the SYSTEM ADMINISTRATORS group. In addition, you are prompted to enter the service URL in the following format:

   `t3://OIM_HOST_NAME:OIM_PORT_NUMBER`

   In this format:

   - Replace *OIM_HOST_NAME* with the host name or IP address of the Oracle Identity Manager host computer.
   - Replace *OIM_PORT_NUMBER* with the port on which Oracle Identity Manager is listening.

   You can use the PurgeCache utility to purge the cache for any content category.

5
Using the Generic REST Connector

You can use the connector for performing reconciliation and provisioning operations after configuring it to meet your requirements.

This chapter is discusses the following topics:

> **Note:**
>
> These sections provide both conceptual and procedural information about configuring the connector. It is recommended that you read the conceptual information before you perform the procedures.

- Configuring Reconciliation
- Scheduled Jobs
- Performing Provisioning Operations
- Uninstalling the Connector

5.1 Configuring Reconciliation

You can configure the connector to specify the type of reconciliation and its schedule.

This section discusses the following topics related to configuring reconciliation:

- Reconciliation Rules for the Generic REST Connector
- Full Reconciliation and Incremental Reconciliation
- Updating the User Reconciliation Scheduled Job for Incremental Reconciliation
- Limited (Filtered) Reconciliation
- Lookup Field Synchronization
- Reconciling Large Number of Records

5.1.1 Reconciliation Rules for the Generic REST Connector

Reconciliation rules are automatically created when you generate the Generic REST connector.

The following is the format of the rule element:

```
User Login Equals NameAttribute
```

In this rule element:

- User Login is the User ID field on the OIM User form.

- NameAttribute is the value of the account qualifier in the schema.properties file that you created in Creating a Schema File.

For example, if the value of the NameAttribute account qualifier is __NAME__, then the rule element is as follows:

```
User Login Equals__NAME__
```

5.1.2 Full Reconciliation and Incremental Reconciliation

Full reconciliation involves reconciling all existing user records from the target system into Oracle Identity Manager. In **incremental reconciliation**, only records created or modified after the latest date or timestamp the last reconciliation was run are considered for reconciliation.

After you deploy the connector, you must first perform full reconciliation.

You can perform a full reconciliation run by removing or deleting any value currently assigned to the Filter Suffix attribute and then run the scheduled job for user data reconciliation. See Scheduled Jobs for Reconciliation of User Records for more information about the user reconciliation scheduled job and Filter Suffix attribute.

If the target system contains more number of records than what it can return in a single response, then use the Flat File connector to perform full reconciliation. See Reconciling Large Number of Records.

To perform incremental reconciliation, you must update and run the scheduled job for user data reconciliation to include the following attributes:

- Incremental Recon Attribute — Name of the target system attribute that holds the time stamp at which the record was last modified. The value in this attribute is used to determine the newest or latest record reconciled from the target system.
- Latest Token — Holds the value of the attribute that is specified as the value of the Incremental Recon Attribute attribute. The Latest Token attribute is used for internal purposes. Do *not* enter a value for this attribute. The reconciliation engine automatically enters a value in this attribute. Sample value: `1354753427000`

See Updating the User Reconciliation Scheduled Job for Incremental Reconciliation for more information about the scheduled job for incremental reconciliation.

5.1.3 Updating the User Reconciliation Scheduled Job for Incremental Reconciliation

If your target system contains an attribute that holds the timestamp at which an object is created or modified, then you must manually update the user reconciliation schedule job to include attributes for incremental reconciliation.

> **Note:**
>
> See Exporting Deployments and Importing Deployments in *Oracle Fusion Middleware Administering Oracle Identity Manager* for detailed instructions on performing each of the steps discussed in this procedure.

To create a scheduled job for incremental reconciliation:

1. Log in to Identity System Administration.
2. Add the user reconciliation scheduled job file to the Deployment Manager for export.
3. Edit the user reconciliation scheduled job file to include the Incremental Recon Attribute and Latest Token attributes.
4. Import the user reconciliation scheduled job file into Oracle Identity Manager.

5.1.4 Limited (Filtered) Reconciliation

Limited or filtered reconciliation is the process of limiting the number of records being reconciled based on a set filter criteria.

By default, all target system records that are added or modified after the last reconciliation run are reconciled during the current reconciliation run. You can customize this process by specifying the subset of added or modified target system records that must be reconciled. You do this by creating filters for the reconciliation module.

You can perform limited reconciliation by creating filters that your target system supports. This connector provides the Filter Suffix attribute (scheduled task attributes) that allows you to use any of the attributes of the target system to filter target system records.

5.1.5 Lookup Field Synchronization

Lookup field synchronization involves obtaining the most current values from specific attributes in the target system to the lookup definitions (used as an input source for lookup fields) in Oracle Identity Manager.

You can perform lookup field synchronization by configuring and running the scheduled jobs for lookup field synchronization.

Scheduled jobs for lookup field synchronization are created only if you have specified a value for the lookupAttributeList entry in the GenericRestConfiguration.groovy file. The names of these scheduled jobs are in the following format:

IT_RES_NAME Target *FIELD_NAME* Lookup Reconciliation

For every attribute specified in the lookupAttributeList entry, a corresponding scheduled job for reconciling lookup values from the target system is created. This is illustrated by the following example:

Suppose the value of the itResourceDefName entry is GenRest. If the value of the lookupAttributeList entry is `['Roles', 'Groups']`, then the connector creates the following scheduled jobs:

- GenRest Target Roles Lookup Reconciliation
- GenRest Target Groups Lookup Reconciliation

> **Note:**
>
> See Scheduled Job for Lookup Field Synchronization.

5.1.6 Reconciling Large Number of Records

During a reconciliation run, if the target system contains more number of records than what it can return in a single response, then you must use the Flat File connector to fetch all the records into Oracle Identity Manager.

To reconcile a large number of records from the target system into Oracle Identity Manager:

1. Export all users in the target system to a flat file.
2. Copy the flat file to a location that is accessible from Oracle Identity Manager.
3. Create a schema file representing the structure of the flat file. See Creating a Schema File in *Oracle Identity Manager Connector Guide for Flat File*.
4. Install the Flat File connector. See Running the Connector Installer in *Oracle Identity Manager Connector Guide for Flat File*.
5. Configure the Flat File IT resource. See Configuring the IT Resource in *Oracle Identity Manager Connector Guide for Flat File*.
6. If you want to perform trusted source reconciliation, then configure and run the Flat File Users Loader scheduled job.

   While configuring this scheduled job, ensure that you set the value of the **Target IT Resource Name** attribute to the name of the IT resource for the target system installation from which you want to reconcile user records and **Target Resource Object Name** to the name of the resource object used for trusted source reconciliation.

   See Flat File Users Loader and IT_RES_NAME Flat File Users Loader in *Oracle Identity Manager Connector Guide for Flat File* for information about the attributes of the Flat File Users Loader scheduled job.
7. If you want to perform target resource reconciliation, then configure and run the Flat File Accounts Loader scheduled job.

   While configuring this scheduled job, ensure that you set the value of the **Target IT Resource Name** attribute to the name of the IT resource for the target system installation from which you want to reconcile user records and **Target Resource Object Name** to the name of the resource object used for target resource reconciliation.

   See Flat File Accounts Loader and IT_RES_NAME Flat File Accounts Loader in *Oracle Identity Manager Connector Guide for Flat File* for information about the attributes of the Flat File Users Loader scheduled job.

5.2 Scheduled Jobs

When you run the Connector Installer, reconciliation scheduled jobs are automatically created in Oracle Identity Manager. You must configure these scheduled jobs to suit your requirements by specifying values for its attributes.

This section discusses the following scheduled jobs that you can configure for reconciliation:

- Scheduled Job for Lookup Field Synchronization
- Scheduled Jobs for Reconciliation of User Records

- Configuring Scheduled Jobs

5.2.1 Scheduled Job for Lookup Field Synchronization

After you generate the connector, scheduled jobs for lookup field synchronization are created only if you have specified a value for the lookupAttributeList entry in the GenericRestConfiguration.groovy file. For every attribute specified in the lookupAttributeList entry, a corresponding scheduled job for reconciling lookup values from the target system is created.

Table 5-1 describes the attributes of the scheduled job for lookup field synchronization.

Table 5-1 Attributes of the Scheduled Job for Lookup Field Synchronization

Attribute	Description
Code Key Attribute	Enter the name of the attribute that is used to populate the Code Key column of the lookup definition (specified as the value of the Lookup Name attribute).
Decode Attribute	Enter the name of the attribute that is used to populate the Decode column of the lookup definition (specified as the value of the Lookup Name attribute).
IT Resource Name	Name of the IT resource for the target system installation from which you want to reconcile records. The default value of this attribute is the same as the value of the ITResourceDefName entry in the GenericRestConfiguration.groovy file.
Lookup Name	Name of the lookup definition in Oracle Identity Manager that must be populated with values fetched from the target system. The value for this attribute is populated automatically if you have specified a value for the lookupAttributeList entry while configuring the GenericRestConfiguration.groovy file. The value of this attribute is in the following format: Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*} For example, if you have specified Roles as the value of the lookupAttributeList entry, then the value of this attribute is Lookup.GenRestTrusted.Roles.
Object Type	Enter the type of object you want to reconcile. Default value: `OTHER` **Note:** For lookup field synchronization, the object type must be any object other than "User."

5.2.2 Scheduled Jobs for Reconciliation of User Records

After you generate the connector, the scheduled task for user data reconciliation is automatically created in Oracle Identity Manager. A scheduled job, which is an instance of this scheduled task is used to reconcile user data from the target system.

The following scheduled jobs are used for user data reconciliation:

- *RESOURCE* Target Resource User Reconciliation

  This scheduled job is used to reconcile user data in the target resource (account management) mode of the connector.

- *RESOURCE* Trusted Resource User Reconciliation

  This scheduled job is used to reconcile user data in the trusted source (identity management) mode of the connector.

Table 5-2 describes the attributes of both scheduled jobs.

Table 5-2 Attributes of the User Reconciliation Scheduled Jobs

Attribute	Description
Filter Suffix	Enter the search filter for fetching user records from the target system during a reconciliation run. See Limited (Filtered) Reconciliation.
IT Resource Name	Name of the IT resource for the target system installation from which you want to reconcile user records. Sample value: `GenRestTrusted`
Object Type	Type of object you want to reconcile. Default value: `User` **Note:** User is the only object that is supported. Therefore, do not change the value of this attribute.
Resource Object Name	Name of the resource object that is used for reconciliation. Sample value: `GenRestTrusted User`
Scheduled Task Name	Name of the scheduled task that is used for reconciliation. The default value of this attribute in the *RESOURCE* Target Resource User Reconciliation scheduled job is *RESOURCE* `Target Resource User Reconciliation.` The default value of this attribute in the *RESOURCE* Trusted Resource User Reconciliation scheduled job is *RESOURCE* `Trusted Resource User Reconciliation.`

5.2.3 Configuring Scheduled Jobs

You configure scheduled jobs to perform reconciliation runs that check for new information on your target system periodically and replicates the data in Oracle Identity Manager.

To configure a scheduled job:

1. Log in to Oracle Identity System Administration.
2. In the left pane, under System Management, click **Scheduler.**

3. Search for and open the scheduled task as follows:
   a. On the left pane, in the Search field, enter the name of the scheduled job as the search criterion. Alternatively, you can click **Advanced Search** and specify the search criterion.
   b. In the search results table on the left pane, click the scheduled job in the Job Name column.
4. On the Job Details tab, you can modify the following parameters:
   - **Retries:** Enter an integer value in this field. This number represents the number of times the scheduler tries to start the job before assigning the Stopped status to the job.
   - **Schedule Type:** Depending on the frequency at which you want the job to run, select the appropriate schedule type.

   > **Note:**
   >
   > See Creating Jobs in *Oracle Fusion Middleware Administering Oracle Identity Manager* for detailed information about schedule types.

   In addition to modifying the job details, you can enable or disable a job.
5. On the Job Details tab, in the Parameters region, specify values for the attributes of the scheduled task.

   > **Note:**
   >
   > - Attribute values are predefined in the connector XML file that you import. Specify values only for those attributes that you want to change.
   > - Values (either default or user-defined) must be assigned to all the attributes. If even a single attribute value is left empty, then reconciliation is not performed.
   > - Attributes of the scheduled job are discussed in Scheduled Jobs.
6. Click **Apply** to save the changes.

   > **Note:**
   >
   > You can use the Scheduler Status page in Identity System Administration to either start, stop, or reinitialize the scheduler.

5.3 Performing Provisioning Operations

You create a new user in Oracle Identity Self Service by using the Create User page. You provision or request for accounts on the Accounts tab of the User Details page.

To perform provisioning operations in Oracle Identity Manager:

1. Log in to Identity Self Service.
2. Create a user. See Creating a User in *Oracle Fusion Middleware Performing Self Service Tasks with Oracle Identity Manager.*
3. On the Account tab, click **Request Accounts.**
4. In the Catalog page, search for and add to cart the application instance created for the IT resource (in Associating the Form with the Application Instance), and then click **Checkout**.

   > **Note:**
   >
   > Ensure to select proper values for lookup type fields as there are a few dependent fields. Selecting a wrong value for such fields may result in provisioning failure.

5. Click Ready to **Submit.**
6. Click **Submit.**
7. If you want to provision entitlements, then:
   a. On the Entitlements tab, click **Request Entitlements.**
   b. In the Catalog page, search for and add to cart the entitlement, and then click **Checkout.**
   c. Click **Submit.**

5.4 Uninstalling the Connector

Uninstalling the connector involves deleting data related to the connector from Oracle Identity Manager Database. You use the Uninstall Connectors utility to uninstall a connector.

If you want to uninstall the connector for any reason, see Uninstalling Connectors in *Oracle Fusion Middleware Administering Oracle Identity Manager*.

6

Extending the Functionality of the Generic REST Connector

After you generate and install the connector, you can extend its functionality to address your specific business requirements.

This chapter discusses the following optional configuration procedures:

- Implementing Custom Authentication
- Implementing Custom Parsing
- Adding Custom OIM User Fields for Trusted Source Reconciliation
- Adding Custom Fields for Target Resource Reconciliation
- Adding Custom Fields for Provisioning
- Configuring Transformation of Data During User Reconciliation
- Configuring Validation of Data During Reconciliation and Provisioning

6.1 Implementing Custom Authentication

If your target system uses an authentication mechanism that is not supported by this connector, then you must implement the authentication that your target system uses and then attach it to the connector by using the plug-ins exposed by this connector. Implementing custom authentication involves creating a Java class, overriding the Map<String, String> getAuthHeaders(Map<String, Object> authParams) method that returns the authorization header in the form of a map, and updating the connector installation media to include the new Java class.

All the target system configuration and authentication details that may be required for obtaining the authorization header are passed to the Map<String, String> getAuthHeaders(Map<String, Object> authParams) method through specific IT resource parameters. All the configuration properties exposed by this connector are accessible within this method as a part of "authParams".

To implement a custom authentication:

1. Create a Java class for implementing custom authentication. This class must implement the org.identityconnectors.restcommon.auth.spi.AuthenticationPlugin interface.

   Note down the name of this Java class. You will provide the name of the Java class while configuring the IT resource for your target system which is described later in this guide.

2. Override the **Map<String, String> getAuthHeaders(Map<String, Object> authParams)** method in the custom Java class.

   This method must implement the custom authentication logic that returns the authorization header in the form of a map For example, `{ Authorization =`

`Bearer XXXXXXXXXX }`. The authorization header contains the access token received from the target.

3. Package the Java class implementing the custom authentication into a JAR file.
4. Package the JAR file containing the custom authentication implementation with the connector bundle JAR as follows:

> **Note:**
>
> Ensure to package all the JARs for any other custom implementations that you may have.

   a. Extract the contents of the org.identityconnectors.genericrest-1.0.1115.jar file into a temp directory. This file is located in the GenericREST-***RELEASE_NUMBER***\bundle directory.
   b. Copy the JAR file containing the custom authentication (from Step 3) to the lib directory.
   c. Regenerate the connector bundle (org.identityconnectors.genericrest-1.0.1115.jar) by running the following command:

```
jar -cvfm org.identityconnectors.genericrest-1.0.1115.jar META-
INF/MANIFEST.MF *
```

> **Note:**
>
> While updating the connector bundle, ensure that META-INF\MANIFEST.MF file is unchanged.

5. Run the Oracle Identity Manager Delete JARs utility to delete any existing JARs in Oracle Identity Manager database before you upload the regenerated connector bundle. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

- For Microsoft Windows:

  OIM_HOME/server/bin/DeleteJars.bat

- For UNIX:

  OIM_HOME/server/bin/DeleteJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being deleted, and the location

from which the JAR file is to be deleted. Specify 4 (ICF Bundle) as the value of the JAR type.

6. Run the Oracle Identity Manager Upload JARs utility to upload the regenerated connector bundle to Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - For Microsoft Windows:

     OIM_HOME/server/bin/UploadJars.bat

   - For UNIX:

     OIM_HOME/server/bin/UploadJars.sh

   When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 4 (ICF Bundle) as the value of the JAR type.

7. Restart Oracle Identity Manager.

This completes the procedure for implementing a custom authentication.

6.2 Implementing Custom Parsing

By default, the connector supports only JSON parsing during reconciliation runs. If the reconciliation data from your target system is not in JSON format, then you must write a custom parser implementation for your data format.

To implement custom parsing:

1. Create a Java class for implementing the custom parser. This class must implement the org.identityconnectors.restcommon.parser.spi.ParserPlugin interface.

   Note down the name of this Java class. You will provide the name of the Java class while configuring the IT resource for your target system which is described later in this guide.

2. Override the **String parseRequest(Map<String, Object> attrMap)** and **List<Map<String, Object>> parseResponse(String response, Map<String, String> parserConfigParams)** methods in the custom Java class.

   The String parseRequest(Map<String, Object> attrMap) method implements the logic for parsing an attribute and generates a string request payload.

   The List<Map<String, Object>> parseResponse(String response, Map<String, String> parserConfigParams) method implements the logic for parsing the string response received from the target in this class.

3. Package the Java class implementing the custom parser into a JAR file.

4. Package the JAR file containing the custom parser implementation with the connector bundle JAR as follows:

   > **Note:**
   >
   > Ensure to package all the JARs for any other custom implementations that you may have.

   a. Extract the contents of the org.identityconnectors.genericrest-1.0.1115.jar file into a temp directory. This file is located in the GenericREST-***RELEASE_NUMBER***\bundle directory.

   b. Copy the JAR file containing the custom authentication (from Step 3) to the lib directory.

   c. Regenerate the connector bundle (org.identityconnectors.genericrest-1.0.1115.jar) by running the following command:

   ```
   jar -cvfm org.identityconnectors.genericrest-1.0.1115.jar META-INF/MANIFEST.MF *
   ```

   > **Note:**
   >
   > While updating the connector bundle, ensure that META-INF\MANIFEST.MF file is unchanged.

5. Run the Oracle Identity Manager Delete JARs utility to delete any existing JARs in Oracle Identity Manager database before you upload the regenerated connector bundle. This utility is copied into the following location when you install Oracle Identity Manager:

   > **Note:**
   >
   > Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - For Microsoft Windows:

     OIM_HOME/server/bin/DeleteJars.bat

   - For UNIX:

     OIM_HOME/server/bin/DeleteJars.sh

   When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being deleted, and the location from which the JAR file is to be deleted. Specify 4 (ICF Bundle) as the value of the JAR type.

6. Run the Oracle Identity Manager Upload JARs utility to upload the regenerated connector bundle to Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the `WL_HOME` environment variable is set to the directory in which Oracle WebLogic Server is installed.

   - For Microsoft Windows:

     OIM_HOME/server/bin/UploadJars.bat

   - For UNIX:

     OIM_HOME/server/bin/UploadJars.sh

   When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 4 (ICF Bundle) as the value of the JAR type.

7. Restart Oracle Identity Manager.

This completes the procedure for implementing custom parsers.

6.3 Adding Custom OIM User Fields for Trusted Source Reconciliation

While generating the connector, you create mappings between OIM User fields and the corresponding target system fields by specifying a value for the alias entry. After generating the connector, if there are additional target system fields that you want to use during trusted source reconciliation, then you can extend the set of fields by creating custom or user-defined fields (UDFs).

To add new fields for trusted source reconciliation:

1. Add the new field on the OIM User process form. See Configuring Custom Attributes in *Oracle Fusion Middleware Administering Oracle Identity Manager* for information on creating UDFs.

> **Note:**
>
> If the new field that you want to add is already present on the OIM User field, then skip this step and proceed to the next step.

2. Log in to the Design Console.
3. In the resource object definition, add the reconciliation field corresponding to the attribute as follows:
   a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**

b. Search for and open the resource object corresponding to your target system.

c. On the Object Reconciliation tab, click **Add Field** to open the Add Reconciliation Field dialog box.

d. Specify a value for the field name. For example, `Building.`

e. From the Field Type list, select a data type for the field. In addition, if you want to designate the attribute as a mandatory attribute, then select the check box.

f. Click the Save icon, and then close the dialog box.

g. Click the Save icon.

4. Create a reconciliation field mapping in the process definition as follows:

a. Expand the **Process Management** folder, and then double-click **Process Definition.**

b. Search for and open the process definition for your target system.

c. On the Reconciliation Field Mapping tab, click **Add Field Map.**

d. From the Field Name list in the Add Reconciliation Field Mapping dialog box, select the name that you have assigned to the attribute created in the resource object.

e. Select a value from the **User Attribute** menu and click **OK.**

f. If the field mapping is a key field for matching the process data, check the key Field for Reconciliation matching check box.

g. Click the Save icon.

5. Create a reconciliation profile as follows:

a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**

b. Search for and open the resource object corresponding to your target system.

c. On the Object Reconciliation tab, click **Create Reconciliation Profile.** This copies changes made to the resource object into the MDS.

d. Click the Save icon.

6. Add an entry for the attribute in the lookup definition for reconciliation attribute mapping as follows:

a. Expand the **Administration** folder, and then double-click **Lookup Definition.**

b. Search for and open the Lookup.*RESOURCE*.UM.ReconAttrMap lookup definition.

c. To add a roe, click **Add.**

d. In the **Code Key** column, enter the name that you have set for the attribute in the resource object. For example, `Building.`

e. In the **Decode** column, enter the corresponding name of the target system column. For example, `BUILDING.`

f. Click the Save icon.

6.4 Adding Custom Fields for Target Resource Reconciliation

While generating the connector, you create mappings between OIM User fields and the corresponding target system fields by specifying a value for the alias entry. After generating the connector, if there are additional target system fields that you want to use during target resource reconciliation, then you can extend the set of fields by creating custom or user-defined fields (UDFs).

To add a custom field for reconciliation:

1. Log in to the Design Console.
2. In the resource object definition, add the reconciliation field corresponding to the attribute as follows:
   a. Expand the **Resource Management** folder, and then double-click **Resource Objects.**
   b. Search for and open the resource object corresponding to your target system.
   c. On the Object Reconciliation tab, click **Add Field** to open the Add Reconciliation Field dialog box.
   d. Specify a value for the field name. For example, `Building`.
   e. From the Field Type list, select a data type for the field. In addition, if you want to designate the attribute as a mandatory attribute, then select the check box.
   f. Click the Save icon, and then close the dialog box.
   g. Click the Save icon.
3. Add an entry for the attribute in the lookup definition for reconciliation attribute mapping as follows:
   a. Expand the **Administration** folder, and then double-click **Lookup Definition**.
   b. Search for and open the Lookup.RESOURCE.UM.ReconAttrMap lookup definition.
   c. To add a row, click **Add.**
   d. In the **Code Key** column, enter the name that you have set for the attribute in the resource object. For example, `Building`.
   e. In the **Decode** column, enter the corresponding name of the target system column. For example, `BUILDING`.
   f. Click the Save icon.
4. Add the attribute as a field on the process form as follows:
   a. Expand the **Development Tools** folder, and then double-click **Form Designer**.
   b. Search for and open the process form for your target system.
   c. Click **Create New Version** to create a version of the process form. Then, enter a version name and click the Save icon.
   d. Click **Add**.

e. In the newly added row, enter values for the Name, Variant Type, Field Label, and Field Type columns. If required, enter values for the rest of the columns.

> **Note:**
>
> - If the attribute on the target system is of the Time, or Timestamp format, then set the value of the Variant Type column to **String**.
> - If you want to handle date attributes of the target system as a date editor, then set the value of the Variant Type column to **Date**. Otherwise, set it to **String**.

f. Click the Save icon.

g. Click **Make Version Active** to activate the new version of the process form.

5. Create a reconciliation field mapping in the process definition as follows:

   a. Expand the **Process Management** folder, and then double-click **Process Definition**.

   b. Search for and open the process definition for your target system.

   c. On the Reconciliation Field Mapping tab, click **Add Field Map**.

   d. From the Field Name list in the Add Reconciliation Field Mapping dialog box, select the name that you have assigned to the attribute created in the resource object.

   e. Double-click the Process Data Field, a new pop-up will appear. The entries in the pop-up correspond to the process form fields.

   f. Select the corresponding newly added field from the pop-up.

   g. If the field mapping is a key field for matching the process data, check the key Field for Reconciliation matching check box.

   h. Click the Save icon.

6. Create a reconciliation profile as follows:

   a. Expand the **Resource Management** folder, and then double-click **Resource Objects**.

   b. Search for and open the resource object corresponding to your target system.

   c. On the Object Reconciliation tab, click **Create Reconciliation Profile**. This copies changes made to the resource object into the MDS.

   d. Click the Save icon.

7. Perform all changes made to the Form Designer of the Design Console (in Step 4) in a new UI form as follows:

   a. Log in to Oracle Identity System Administration.

   b. Create and active a sandbox. See Creating and Activating a Sandbox.

   c. Create a new UI form to view the newly added field along with the rest of the fields. See Creating a New UI Form.

   d. Associate the newly created UI form with the application instance of your target system. To do so, open the existing application instance for your

resource, from the Form field, select the form (created in Step 7.c), and then save the application instance.

e. Publish the sandbox. See Publishing a Sandbox.

8. Add the attribute for provisioning. See Adding Custom Fields for Provisioning.

6.5 Adding Custom Fields for Provisioning

While generating the connector, by performing the procedure described in Generating and Installing the Connector Metadata Package you create mappings between the OIM User fields and the corresponding target system fields (columns) by specifying a value for the alias entry. If there are additional target system fields that you want to use during provisioning, then you can extend the existing set of fields by creating custom or user-defined fields (UDFs).

To add a new user-defined field for provisioning:

1. Add the attribute as a field on the process form as follows:

> **Note:**
>
> Directly proceed to the next step if you have already added the field to the process form while performing the procedure described in Adding Custom Fields for Target Resource Reconciliation.

a. Expand **Development Tools**, and then double-click **Form Designer**.

b. Search for and open the process form for your target system.

c. Click **Create New Version** to create a version of the form. Then, enter a version name and click the Save icon.

d. Click **Add**.

e. In the newly added row, enter values for the Name, Variant Type, Field Label, and Field Type columns. If required, enter values for the rest of the columns.

> **Note:**
>
> - If the attribute on the target system is of the Time, or Timestamp format, then set the value of the Variant Type column to **String**.
> - If you want to handle date attributes of the target system as a date editor, then set the value of the Variant Type column to **Date**. Otherwise, set it to **String**.

f. Click the Save icon.

g. Click **Make Version Active** to activate the new version of the process form.

2. Perform all changes made to the Form Designer of the Design Console (in Step 1) in a new UI form as follows:

a. Log in to Oracle Identity System Administration.

b. Create and active a sandbox. See Creating and Activating a Sandbox.

c. Create a new UI form to view the Creating and Activating a Sandbox newly added field along with the rest of the fields. See Creating a New UI Form.

d. Associate the newly created UI form with the application instance of your target system. To do so, open the existing application instance for your resource, from the Form field, select the form (created in Step 2.c.), and then save the application instance.

e. Publish the sandbox. See Publishing a Sandbox.

3. Add an entry in the lookup definition for provisioning attribute mappings as follows:

a. Expand **Administration**, and then double-click **Lookup Definition**.

b. Search for and open the Lookup.RESOURCE.UM.ProvAttrMap lookup definition.

c. To add a row, click **Add**.

d. In the **Code Key** column, enter the field label for the attribute on the process form. See Step 1 for information about this field name.

e. In the **Decode** column, enter the corresponding name of the target system column. For example, BUILDING.

f. Click the Save icon.

4. To enable updates of the attribute, add an update process task in the process definition as follows:

a. Expand **Process Management**, and then double-click **Process Definition**.

b. Search for and open the process definition for your target system.

c. On the Tasks tab, click **Add**.

d. On the General tab of the dialog box that is displayed, enter a name and description for the task, and then select the following fields in the Task Properties section:

- Conditional
- Required for Completion
- Allow Cancellation while Pending
- Allow Multiple Instances

> **Note:**
>
> The name must be in the *PROCESS_FORM_FIELD_NAME* Updated format.

e. Click the Save icon.

f. On the Integration tab, attach the adapter responsible for performing the update account provisioning operations and map the adapter variables as listed in the following table:

Variable Name	Data Type	Map To	Qualifier	Literal Value
processKeyInstance	Long	Process Data	Process Instance	NA
Adapter return value	Object	Response Code	NA	NA
objectType	String	Literal	String	User
attrFieldName	String	Literal	String	Building
itResourceFieldName	String	Literal	String	IT Resource Form Field Name

g. Click the Save icon.

h. On the Response tab, add appropriate responses.

i. Click the Save icon.

j. Click the Save icon and then close the dialog box.

5. Adding the attribute for reconciliation.

   When you add an attribute on the process form, you must also enable reconciliation of values for that attribute from the target system. See Adding Custom Fields for Target Resource Reconciliation .

6.6 Configuring Transformation of Data During User Reconciliation

You can configure transformation of reconciled single-valued data according to your requirements. For example, you can use First Name and Last Name values to create a value for the Full Name field in Oracle Identity Manager.

> **Note:**
>
> This section describes an optional procedure. Perform this procedure only if you want to configure transformation of data during reconciliation.

To configure transformation of data:

1. Write code that implements the required transformation logic in a Java class.

   The following sample transformation class creates a value for the Full Name attribute by using values fetched from the FIRST_NAME and LAST_NAME columns of the target system:

```
package oracle.iam.connectors.common.transform;

import java.util.HashMap;

public class TransformAttribute {

       /*
       Description:Abstract method for transforming the
```

```
attributes

          param hmUserDetails<String,Object>

          HashMap containing parent data details

          param hmEntitlementDetails <String,Object>

          HashMap containing child data details

          */
          public Object transform(HashMap hmUserDetails, HashMap
hmEntitlementDetails,String sField) {
          /*
           * You must write code to transform the
attributes.
           Parent data attribute values can be fetched by
           using hmUserDetails.get("Field Name").
           *To fetch child data values, loop through the
           * ArrayList/Vector fetched by
hmEntitlementDetails.get("Child          Table")
           * Return the transformed attribute.
           */
          String sFirstName= (String)hmUserDetails.get("First
Name");
          String sLastName= (String)hmUserDetails.get("Last
Name");
          String sFullName=sFirstName+"."+sLastName;
          return sFullName;
          }
   }
```

2. Create a JAR file to hold the Java class.
3. Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the WL_HOME environment variable is set to the directory in which Oracle WebLogic Server is installed.

For Microsoft Windows:

OIM_HOME/server/bin/UploadJars.bat

For UNIX:

OIM_HOME/server/bin/UploadJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 1 as the value of the JAR type.

4. Create a lookup definition for transformation and add an entry to it as follows:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.
   c. In the Code field, enter `Lookup.RESOURCE.UM.ReconTransformation` as the name of the lookup definition.
   d. Select the **Lookup Type** option.
   e. On the Lookup Code Information tab, click **Add**.

      A new row is added.
   f. In the **Code Key** column, enter the name of the resource object field into which you want to store the transformed value. For example: `FirstName`.
   g. In the **Decode** column, enter the name of the class that implements the transformation logic. For example, `oracle.iam.connectors.common.transform.TransformAttribute`.
   h. Save the changes to the lookup definition.
5. Add an entry in the Lookup.*RESOURCE*.UM.Configuration lookup definition to enable transformation as follows:
   a. Expand **Administration**, and then double-click **Lookup Definition**.
   b. Search for and open the **Lookup.*RESOURCE*.UM.Configuration** lookup definition.
   c. Create an entry that holds the name of the lookup definition used for transformation as follows:

      Code Key: `Recon Transformation Lookup`

      Decode: `Lookup.RESOURCE.UM.ReconTransformation`
   d. Save the changes to the lookup definition.

6.7 Configuring Validation of Data During Reconciliation and Provisioning

You can configure validation of reconciled and provisioned single-valued data according to your requirements.

For example, you can validate data fetched from the FIRST_NAME column to ensure that it does not contain the number sign (#). In addition, you can validate data entered in the First Name field on the process form so that the number sign (#) is not sent to the target system during provisioning operations.

For data that fails the validation check, the following message is displayed or recorded in the log file:

```
oracle.iam.connectors.icfcommon.recon.SearchReconTask : handle : Recon
event skipped, validation failed [Validation failed for attribute:
[FIELD_NAME]]
```

> **Note:**
>
> This feature cannot be applied to the Locked/Unlocked status attribute of the target system.

To configure validation of data:

1. Write code that implements the required validation logic in a Java class.

   The following sample validation class checks if the value in the First Name attribute contains the number sign (#):

```
package com.validate;
import java.util.*;
public class MyValidation {
public boolean validate(HashMap hmUserDetails,
              HashMap hmEntitlementDetails, String field)
{
            /*
         * You must write code to validate attributes.
Parent
         * data values can be fetched by using
hmUserDetails.get(field)
         * For child data values, loop through the
         * ArrayList/Vector fetched by
hmEntitlementDetails.get("Child Table")
         * Depending on the outcome of the validation
operation,
         * the code must return true or false.
         */
         /*
         * In this sample code, the value "false" is returned if
the field
         * contains the number sign (#). Otherwise, the value
"true" is
         * returned.
         */
            boolean valid=true;
            String sFirstName=(String)
hmUserDetails.get(field);
            for(int i=0;i<sFirstName.length();i++){
              if (sFirstName.charAt(i) == '#'){
                    valid=false;
                    break;
              }
            }
            return valid;
       }
}
```

2. Create a JAR file to hold the Java class.
3. Run the Oracle Identity Manager Upload JARs utility to post the JAR file to the Oracle Identity Manager database. This utility is copied into the following location when you install Oracle Identity Manager:

> **Note:**
>
> Before you use this utility, verify that the WL_HOME environment variable is set to the directory in which Oracle WebLogic Server is installed.

For Microsoft Windows:

OIM_HOME/server/bin/UploadJars.bat

For UNIX:

OIM_HOME/server/bin/UploadJars.sh

When you run the utility, you are prompted to enter the login credentials of the Oracle Identity Manager administrator, URL of the Oracle Identity Manager host computer, context factory value, type of JAR file being uploaded, and the location from which the JAR file is to be uploaded. Specify 1 as the value of the JAR type.

4. If you created the Java class for validating a process form field for reconciliation, then:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.
   c. In the Code field, enter `Lookup.`*`RESOURCE`*`.UM.ReconValidation` as the name of the lookup definition.
   d. Select the **Lookup Type** option.
   e. On the Lookup Code Information tab, click **Add**.

      A new row is added.
   f. In the Code Key column, enter the resource object field name. For example, `First Name`.
   g. In the Decode column, enter the class name. For example, `com.validate.MyValidation`.
   h. Save the changes to the lookup definition.
   i. Search for and open the **Lookup.*RESOURCE*.UM.Configuration** lookup definition.
   j. Create an entry with the following values:

      Code Key: `Recon Validation Lookup`

      Decode: `Lookup.`*`RESOURCE`*`.UM.ReconValidation`
   k. Save the changes to the lookup definition.
5. If you created the Java class for validating a process form field for provisioning, then:
   a. Log in to the Design Console.
   b. Expand **Administration**, and then double-click **Lookup Definition**.
   c. In the Code field, enter `Lookup.`*`RESOURCE`*`.UM.ProvValidation` as the name of the lookup definition.
   d. Select the **Lookup Type** option.

e. On the Lookup Code Information tab, click **Add**.

   A new row is added.

f. In the **Code Key** column, enter the process form field name. In the **Decode** column, enter the class name.

g. Save the changes to the lookup definition.

h. Search for and open the **Lookup.RESOURCE.UM.Configuration** lookup definition.

i. Create an entry with the following values:

   Code Key: `Provisioning Validation Lookup`

   Decode: `Lookup.`*`RESOURCE`*`.UM.ProvValidation`

j. Save the changes to the lookup definition.

7

Frequently Asked Questions for the Generic REST Connector

When using the Generic REST connector, you might have one or more questions that are answered here.

What are the various objects that are supported by the Generic REST connector ?

The Generic REST connector supports any object that is exposed by your target system. For example, it can support Users, Groups, Organization, Roles, Licenses, and so on.

A
Sample Schema File

The appendix displays a snippet of a schema file that helps the connector understanding the underlying schema of the target system. Using the information provided in this schema file, the connector creates mappings between the target system and connector attributes. The connector uses these attribute mappings for performing reconciliation and provisioning operations.

```
#Schema file for ACME User

#List of fields
FieldNames=EmpId,UserName,FirstName,LastName,Email,Currency,Salary,Statu
s,JoiningDate,LastUpdated,Groups,Roles

#Unique ID Attribute
UidAttribute=EmpId

#Account Name attribute
NameAttribute=UserName

#Account Status Attribute
StatusAttribute=Status

#Multivalued attributes
Groups.Multivalued=true
Roles.Multivalued=true

#Subfields for complex child form
Roles.Subfields=RoleName,Start_Date,End_Date

#Complex child form objectClass
Roles.EmbeddedObjectClass=MyROLES

#Datatypes (Default:String)
Roles.Start_Date.DataType=Long
Roles.End_Date.DataType=Long
FirstName.DataType=String
JoiningDate.DataType=Long

#Incremental reconciliation attribute with datatype set to Long
LastUpdated.DataType=Long

#Parent and child form mandatory fields
Roles.RoleName.Required=true
```

B

Lookup Definitions Used During Connector Operations

Lookup definitions used during reconciliation and provisioning are either preconfigured or can be synchronized with the target system.

This section discusses the following categories of lookup definitions:

- Predefined Lookup Definitions
- Lookup Definitions Synchronized with the Target System

B.1 Predefined Lookup Definitions

Preconfigured lookup definitions are the other lookup definitions that are created in Oracle Identity Manager when you deploy the connector. These lookup definitions are either prepopulated with values or values must be manually entered in them after the connector is deployed.

The other lookup definitions are as follows:

- Lookup.*RESOURCE*.Configuration
- Lookup.*RESOURCE*.UM.Configuration
- Lookup.*RESOURCE*.UM.ReconAttrMap
- Lookup.*RESOURCE*.UM.ProvAttrMap
- Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

> **Note:**
>
> *RESOURCE* has been used as a place holder text for IT resource name. Therefore, replace all instances of *RESOURCE* in this guide with the value that you specified for the itResourceName entry in the GenericRestConfiguration.groovy file. See About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File for more information about entries in the GenericRestConfiguration.groovy file.

B.1.1 Lookup.*RESOURCE*.Configuration

The Lookup.*RESOURCE*.Configuration lookup definition holds connector configuration entries that are used during reconciliation (both trusted source and target resource) and provisioning operations. Table B-1 lists the entries in this lookup definition.

Table B-1 Entries in the Lookup.*RESOURCE*.Configuration Lookup Definition

Code Key	Decode	Description
Bundle Name	org.identityconnectors.genericrest	This entry holds the name of the connector bundle package. Do *not* modify this entry.
Bundle Version	1.0.11150	This entry holds the version of the connector bundle class. Do *not* modify this entry.
Connector Name	org.identityconnectors.genericrest.GenericRESTConnector	This entry holds the name of the connector class. Do *not* modify this entry.
User Configuration Lookup	Lookup.*RESOURCE*.UM.Configuration	This entry holds the name of the lookup definition that contains configuration information specific to the user object type.

B.1.2 Lookup.*RESOURCE*.UM.Configuration

The Lookup.*RESOURCE*.UM.Configuration lookup definition contains entries specific to the user object type. This lookup definition is preconfigured. Table B-2 lists the default entries in this lookup definition when you have configured your target system as a target resource.

Table B-2 Entries in the Lookup.*RESOURCE*.UM.Configuration Lookup Definition for a Target Resource Configuration

Code Key	Decode
Provisioning Attribute Map	Lookup.*RESOURCE*.UM.ProvAttrMap
Recon Attribute Map	Lookup.*RESOURCE*.UM.ReconAttrMap

Table B-3 lists the default entries in this lookup definition when you have configured your target system as a trusted source.

Table B-3 Entries in the Lookup.*RESOURCE*.UM.Configuration Lookup Definition for a Trusted Source Configuration

Code Key	Decode
Recon Attribute Map	Lookup.*RESOURCE*.UM.ReconAttrMap
Recon Attribute Defaults	Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

B.1.3 Lookup.*RESOURCE*.UM.ReconAttrMap

The Lookup.*RESOURCE*.UM.ReconAttrMap lookup definition holds mappings between resource object fields and target system attributes.

Depending on whether you have configured your connector for the target resource mode or trusted source mode, this lookup definition is used during target resource or trusted source user reconciliation runs, respectively.

If you have configured the connector for target resource mode:

The following is the format of the Code Key and Decode values in this lookup definition:

For single-valued attributes:

- **Code Key:** Reconciliation attribute of the resource object against which target resource user reconciliation runs must be performed
- **Decode:** Corresponding target system attribute name

For multivalued attributes:

- **Code Key:** *RO_ATTR_NAME~ATTR_NAME*[LOOKUP]

  In this format:

  - *RO_ATTR_NAME* specifies the reconciliation field for the child table.
  - *ATTR_NAME* is the name of the multivalued attribute.
  - [LOOKUP] is a keyword that is appended to the code key value if the child data is picked from a lookup or declared as an entitlement.
- **Decode:** Corresponding target system attribute name

  EMBED_OBJ_NAME~RELATION_TABLE_NAME~ATTR_NAME

  In this format:

  - *EMBED_OBJ_NAME* is the name of the object (for example, an account's address) on the target system that is embedded in another object.
  - *RELATION_TABLE_NAME* is the name of child table in the target system.
  - *ATTR_NAME* is the name of the column in the child table corresponding to the multivalued attribute in the Code Key column.

If you have configured your connector for trusted source mode:

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Reconciliation attribute of the resource object against which trusted source user reconciliation runs must be performed
- **Decode:** Corresponding target system attribute name

The entries in this lookup definition depend on the data available in the target system. The entries of this lookup definition are populated based on the values specified for the alias entry in the GenericRestConfiguration.groovy file. See About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File for more information about the alias entry.

B.1.4 Lookup.*RESOURCE*.UM.ProvAttrMap

The Lookup.*RESOURCE*.UM.ProvAttrMap lookup definition holds mappings between process form fields and target system attribute names. This lookup definition is used for performing provisioning operations.

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Name of the label on the process form
- **Decode:** Corresponding target system attribute name

For entries corresponding to child form fields, the following is the format of the Code Key and Decode values:

- **Code Key:** *CHILD_FORM_NAME~FIELD_NAME*

  In this format:

  – *CHILD_FORM_ NAME* specifies the name of the child form.

  – *FIELD_NAME* specifies the name of the label on the child form.

- **Decode:** Combination of the following elements separated by the tilde (~) character:

  EMBED_OBJ_NAME~RELATION_TABLE_NAME~COL_NAME

  In this format:

  – *EMBED_OBJ_NAME* is the name of the object (for example, an account's address) on the target system that is embedded in another object.

  – *COL_NAME* is the name of the column in the child table corresponding to the child form specified in the Code Key column.

  – *RELATION_TABLE_NAME* is the name of child table in the target system.

The entries in this lookup definition depend on the data available in the target system. The values in the lookup definition are populated based on the value specified for the alias entry in the GenericRestConfiguration.groovy file. See About the dateAttributeList, entitlementAttributeList, lookupAttributeList, and alias Entries of the Groovy File for more information about the alias entry.

B.1.5 Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults

The Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults lookup definition holds default values of the mandatory fields on the OIM User form that are not mapped with the target system attributes. This lookup definition is created only if you have configured the connector for the trusted source mode.

The Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults lookup definition is used when there is a mandatory field on the OIM User form, but no corresponding attribute in the target system from which values can be fetched during trusted source reconciliation runs. In addition, this lookup definition is used if the mandatory field on the OIM User form has a corresponding column that is empty or contains null values.

The following is the format of the Code Key and Decode values in this lookup definition:

- **Code Key:** Name of the user field in Identity Self Service.
- **Decode:** Corresponding default value to be displayed.

For example, the Role field is a mandatory field on the OIM User form. Suppose the target system contains no attribute that stores information about the role for a user account. During reconciliation, no value for the Role field is fetched from the target system. However, as the Role field cannot be left empty, you must specify a value

for this field. Therefore, the Decode value of the Role Code Key has been set to `Full-Time`. This implies that the value of the Role field on the OIM User form displays Full-Time for all user accounts reconciled from the target system.

Table B-4 lists the default entries in this lookup definition.

Table B-4 Entries in the Lookup.*RESOURCE*.UM.ReconAttrMap.Defaults Lookup Definition

Code Key	Decode
Role	Full-Time
Organization Name	Xellerate Users
Xellerate Type	End-User

B.2 Lookup Definitions Synchronized with the Target System

During a provisioning operation, you use a lookup field on the process form to specify a single value from a set of values. For example, you may want to select a role from a lookup field (displaying a set of roles) to specify the role being assigned to the user.

While configuring the GenericRestConfiguration.groovy file, if you specified a value for the lookupAttributeList entry, then the connector creates a lookup definition for every target system attribute specified in this entry and then associates it with the corresponding lookup field on the OIM User process form. The connector creates a lookup definition named in the following format:

Lookup.${*IT_RES_NAME*}.${*FIELD_NAME*}

In this format, the connector replaces:

- *IT_RES_NAME* with the value of the itResourceDefName entry in the GenericRestConfiguration.groovy file.
- *FIELD_NAME* with the name of the field for which the lookup field is created.

Lookup field synchronization involves copying additions or changes made to the target system attributes (listed in the lookupAttributeList entry) into corresponding lookup definitions (used as an input source for lookup fields) in Oracle Identity Manager. This is achieved by running scheduled jobs for lookup field synchronization.

The following example illustrates the list of lookup definitions created for a given lookupAttributeList value:

Suppose the value of the itResourceDefName entry is GenRest. If the value of the lookupAttributeList entry is `['Roles', 'Groups']`, then the connector creates the following lookup definitions:

- Lookup.GenRest.Roles
- Lookup.GenRest.Groups

After you perform lookup field synchronization, data in the lookup definition is stored in the following format:

- **Code Key value:** *IT_RESOURCE_KEY~LOOKUP_FIELD_ID*

  In this format:

- *IT_RESOURCE_KEY* is the numeric code assigned to each IT resource in Oracle Identity Manager.
- *LOOKUP_FIELD_ID* is the target system code assigned to each lookup field entry. This value is populated based on the target system attribute name specified in the Code Key attribute of the scheduled job for lookup field synchronization.

Sample value: `1~SA`

- **Decode value:** *IT_RESOURCE_NAME~LOOKUP_FIELD_ID*

In this format:

- *IT_RESOURCE_NAME* is the name of the IT resource in Oracle Identity Manager.
- *LOOKUP_FIELD_ID* is the target system code assigned to each lookup field entry. This value is populated based on the target system attribute name specified in the Decode attribute of the scheduled job for lookup field synchronization.

Sample value: `GenRest~SYS_ADMIN`

> **See Also::**
>
> Scheduled Job for Lookup Field Synchronization for information about the attributes of the scheduled job for lookup field synchronization

C

Files and Directories of the Generic REST Connector

These are the components of the connector installation package and connector metadata package that comprise the Generic REST connector.

Table C-1 describes the files and directories in the connector installation package.

Table C-1 Files and Directories in the Connector Installation Package

File in the Installation Package Directory	Description
bundle/org.identityconnectors.genericrest-1.0.11150.jar	This JAR file is the ICF connector bundle.
configuration/GenericRest-CI.xml	This XML file contains configuration information. The Connector Installer uses this XML file to create connector components.
javadoc	This directory contains information about the Java APIs used by the connector.
metadata-generator/bin/GenericRestGenerator.cmd	This file contains commands to run the metadata generator.
metadata-generator/bin/GenericRestGenerator.sh	Note that the .cmd file is the Microsoft Windows version of the metadata generator. Similarly, the .sh file is the UNIX version of the metadata generator.
metadata-generator/bin/classpath.cmd metadata-generator/bin/classpath-append.cmd	These files contain the commands that add the JAR files (located in the lib directory) to the classpath on Microsoft Windows.
metadata-generator/bin/logging.properties	This file contains the default logging configurations for the metadata generation utility.
metadata-generator/lib/connector-framework-internal.jar	This JAR files contains class files that implement ICF.
metadata-generator/lib/connector-framework.jar	This JAR file contains class files that define the ICF Application Programming Interface (API). This API is used to communicate between Oracle Identity Manager and this connector.
metadata-generator/lib/genericrest-oim-integration.jar	This JAR file contains the class files of the metadata generation utility.
metadata-generator/lib/groovy-all.jar	This JAR file contains the groovy libraries required for running the metadata generator.
metadata-generator/lib/org.identityconnectors.genericrest-1.0.11150.jar	This JAR file is the ICF connector bundle. This file is used during metadata generation.

Table C-1 (Cont.) Files and Directories in the Connector Installation Package

File in the Installation Package Directory	Description
metadata-generator/resources/GenericRestConfiguration.groovy	This file contains properties that store basic information about the target system schema, which is used to configure the mode (trusted source or target resource) in which you want to run the connector. In addition, it stores information about the manner in which the connector must connect to the target system. See Configuring the Groovy File for more information about entries in the RestConfiguration.groovy file.

Table C-2 describes the files and directories in the connector metadata package.

Table C-2 Files and Directories in the Connector Metadata Package

File in the Connector Metadata Package	Description
configuration/*IT_RES_DEF*-CI.xml	This XML file contains configuration information that is used by the Connector Installer during the connector installation process.
resources/genericrest-generator.properties	This property file contains locale-specific properties. You can use this file as a template to add or update locale-related properties.
xml/*IT_RES_DEF*-ConnectorConfig.xml file	This XML file contains definitions for connector components such as IT resource, lookup definitions, scheduled tasks, process forms, and resource objects. This file is also referred to as the connector configuration file.

See Also:

Understanding the Generated Connector Package for the Generic REST Connector for information about the structure of the generated connector package