Oracle
Primavera
Gateway Provider Customization Guide

Version 21
February 2021

Contents

Overview

Primavera Gateway is an application that facilitates sharing project, resource, and other data between your enterprise application and Primavera applications, enabling you to combine the management and scheduling power for projects, portfolios, and resources of Oracle Primavera applications with other enterprise software.

Within our documentation, some content might be specific for cloud deployments while other content is relevant for on-premises deployments. Any content that applies to only one of these deployments is labeled accordingly.

The *Primavera Gateway Customization Guide* describes how to customize an integration using a provider that has been created for Primavera Gateway. Also see the *Provider's Reference Guide* for a list of business objects supported by each Primavera provider in the data dictionary.

Primavera Gateway Architecture

Primavera Gateway is a web application that is deployed on a WebLogic application server. The following components are required to load data into a Primavera application and the Gateway database:

- **Seed Data,** XML files that provide flows and other data to illustrate best practices, which can be used as a starting point for your implementation.
- **Primavera Gateway Framework**, which includes:
  - Primavera Gateway user interface
  - The following providers to support integrations with Primavera applications:
    - EnterpriseTrack provider

      The EnterpriseTrack provider enables you to share data with Oracle Instantis EnterpriseTrack application. Primavera Gateway supports Oracle Instantis EnterpriseTrack integration with a Sample provider.
    - P6 provider

      The P6 provider enables you to share data with P6 EPPM. Primavera Gateway supports P6 EPPM integration with the Oracle Primavera Cloud, and Unifier applications, and a Sample provider.
    - Primavera Cloud provider

      The Primavera Cloud provider enables you to share data with the Oracle Primavera Cloud application. Primavera Gateway supports Oracle Primavera Cloud integration with the P6 EPPM and Unifier applications.
    - Sample provider

      The Sample provider is a provider for demonstration purposes only. The purpose of the Sample provider is to illustrate how to use Primavera Gateway to synchronize data between a Primavera application and the Sample provider. Primavera Gateway supports a Sample provider integration with P6 EPPM and Oracle Instantis EnterpriseTrack applications.
    - Unifier provider

The Unifier provider enables you to share data with the Primavera Unifier application. Primavera Gateway supports Primavera Unifier integration with P6 EPPM and Oracle Primavera Cloud applications.

- File provider

  The File provider allows you to export or import data in XML and CSV file formats. When using the XML format, the XML file must conform to the Gateway schema. Primavera Gateway supports a File provider integration with all Primavera applications listed above. For more details, see the *Primavera Gateway File Provider Setup Guide*.

What can be Customized?

With a provider, an integration between Gateway and your application can be customized to define additional:

- fields
- custom field mappings using groovy code
- custom parameters
- custom cross-references
- custom steps
- direct field mapping templates
- groovy field mapping templates
- flows

In an on-premises environment, you can also define:

- custom Java field mappings, and Java custom steps
- flow parameters and the Java code can be extended to use new parameters

As with most application development work, when you customize a provider, you will generally need to involve people that have various skills and that perform various roles: programmers, analysts, users, and administrators.

Where are the Customizations Stored?

Customizations that you make by interacting with Primavera Gateway are stored in an Oracle database. The following items are stored in the database:

- Cross references (Xref), data value mappings (DVM), flows, metadata (data dictionary), and customizations
- Audit information, logs, and intermediate artifacts
- Configuration settings
- Schedules

Final Customization Artifacts

The following files must be created and deployed with each provider customization:

- A customization description XML file
- A .jar file containing any Java code to implement the customization interfaces. The jar file is not needed if the customization does not have any java code. (on-premises only)

Customizing an Integration

To customize an integration, you will need to customize the source, destination provider applications to support additional objects, fields, flows, flow parameters, custom steps, etc.

Customize a provider application by creating one or more customization description XML files as follows:

For Cloud

To customize an integration, complete the following steps:

1) Create the customization description XML file.

   You can also download a sample customization file from http://docs.oracle.com/cd/F12018_01/customization_xml/customization.zip
2) Deploy your project and customize the Primavera Gateway database schema by importing the customization description XML file from the **Import/Export** tab of the **Configuration** page of the Primavera Gateway user interface.

For more details on each step, see the corresponding section in this guide.

For On-Premises

To customize an integration, complete the following steps:

1) Set up your environment.
2) Create the customization description XML file.

   If you have Java code, then implement the customization interfaces and build your project.

   > **Note**: Usually, building the customization interfaces and creating the customization XML file occurs simultaneously.
3) Deploy your project and customize the Primavera Gateway database schema by any of the following methods:
   - Import the customization description XML file from the **Import/Export** tab of the **Configuration** page of the Primavera Gateway user interface
   - Using the Gateway Setup Configuration Utility
4) Restart the WebLogic domain if the customization includes a .jar file.

For more details on each step, see the corresponding section in this guide.

In This Section

Setting Up Your Environment for On-Premises

Set up your environment as follows:

1) Create a folder to contain your customization files. This folder is known as the <project_home> folder.
2) Locate the **<Primavera_Gateway_home>\gateway\app** folder, extract the gateway.ear file and copy the **pdi.jar** file to an appropriate folder in your <project_home> folder.
3) Put the **pdi.jar** file in your project's classpath.

Creating the Customization XML File

With a provider, an integration between Gateway and your application can be customized to define additional:

- fields
- custom field mappings using groovy code
- custom parameters
- custom cross-references
- custom steps
- direct field mapping templates
- groovy field mapping templates
- flows

In an on-premises environment, you can also define:

- custom Java field mappings, and Java custom steps
- flow parameters and the Java code can be extended to use new parameters
- As with most application development work, when you customize a provider, you will generally need to involve people that have various skills and that perform various roles: programmers, analysts, users, and administrators.

You can download a sample customization XML file from:
http://docs.oracle.com/cd/F25735_01/customization_XML/customization.zip

Entrance to the Java Code for On-Premises

The CustomizationHandler interface (`com.oracle.pgbu.gateway.customization.CustomizationHandler`) provides the entry point into all of the Java code in the customization jar file and therefore must be implemented.

In the sample customization project, the Customization class (under `com.mycompany.primavera.integration.custom` package) implements the CustomizationHandler interface. The Gateway loads this class through reflection, and retrieves all other feature implementations by querying this class.

The Java class also must be defined in the XML description file. You can find the following at the top of the customization.xml from the sample customization project.

```
<JavaClassPath>com.mycompany.primavera.Customization</JavaClassPath>
```

Adding Custom Java Field Mapping for On-Premises

When the field mapping is not as simple as one-to-one, you can use a custom Java field mapping mechanism. Using a custom Java field mapping involves creating a class that implements the CustomFieldMap interface. When you implement the CustomFieldMap interface, you have control of the source object and the target object which permits you to write logic that involves more than one field. The SampleResourceCustomFieldMap class (under com.mycompany.primavera.integration.custom.java package) is a simple custom Java field mapping example in the sample customization project that concatenates the ResourceId and EmployeeName fields of Resource in the Host side and puts the concatenated field into the Name field of Resource in the PDI side. This class is defined in the customization.xml in the following section:

```
    <JavaCustomizationMapping>
<App1Name>Sample</App1Name>
<App2Name>P6</App2Name>
<FieldMapping>
    <Description>Resource Java Field Mapping</Description>
    <App1BusinessObjectName>Resource</App1BusinessObjectName>
    <App2BusinessObjectName>Resource</App2BusinessObjectName>
    <Name>SampleResourceCustomFieldMap</Name>
    <PDIBusinessObjectName>Resource</PDIBusinessObjectName>
    <Fields>
        <App1>
            <Name>EmployeeName</Name>
        </App1>
        <App2><Name>Name</Name>
        </App2>
    </Fields>
</FieldMapping>
    </JavaCustomizationMapping>
```

Adding External Custom Steps

External custom steps are available for Gateway cloud service and on-premises installations. You can add an external custom step in a flow to handle specific additional requirements to extend the capabilities of a standard flow provided in Gateway. External custom steps can be created and deployed outside of the Gateway server. The Gateway **Customization SDK** describes how to create and deploy an external custom step for Gateway.

Downloading Customization SDK and JavaDocs

Gateway developer users who need to implement external custom steps must get familiar with the utility classes and methods delivered in the Customization SDK. Download the SDK and JavaDocs for the two utility classes (CustomizationSession and ExternalStepRequest) as follows:

In the Gateway user interface:

1) Select ⓘ **Help**.
2) Select **Customization SDK**.

   For more details how to implement an external custom step, see *gateway_external_custom_step_SDK.HTM* in the SDK download.

   A Sample External Custom Step is also delivered in Gateway. For more details on how to use the Sample External Custom Step in a business flow and a synchronization, see *Readme_Gateway_External_Custom_Step.txt* in the SDK download.

Adding Internal Custom Steps for On-Premises

A custom step is a very powerful tool that enables customers to make big changes to the flow. In addition to the normal Load/Convert/Save/Compare steps, you can insert a custom step anywhere after the Load step and before the Save step. This allows you to change the document before the document is passed on to the next step. Internal custom steps are also referred to as Java Custom Steps in Gateway.

> **Note**: When you use an internal custom step, you have to be very careful not to disrupt other existing flow steps.

In the sample customization project, there is a sample custom step, SummarizePlannedCost, that does a very simple summarization. It adds up the PlannedCost of all of the ResourceAssignments in the project and puts it into a new UDF field of the project object. The new UDF field is the SummarizePlannedCost field. After performing this task, the custom step removes all ResourceAssignment elements from the document. A simple boolean type parameter controls whether the custom step is executed or not.

SummarizePlannedCost can be found under the com.mycompany.primavera.integration.custom.step package and it is defined in the following section in the customization.xml file:

```
<FlowDefinitionCustomization>
        <FlowDefinitionName>Project Data</FlowDefinitionName>
        <!--   <DisableCompare>true</DisableCompare> Uncomment this line to disable compare for this flow type -->
        <FlowStep>
                <Description>A sample custom step</Description>
                <Name>Summarize Planned Cost</Name>
                <OwnerAppName>P6</OwnerAppName>
                <Sequence>15</Sequence>
                <!-- to make sure it is inserted after Load step before the first convert step -->
                <Type>Custom</Type>
        </FlowStep>
...
</FlowDefinitionCustomization>
```

Notice that the sequence number that you set here determines when this step will get called. In this case, this step is inserted right after the P6 Load step.

Adding Flow Parameters for On-Premises

A flow parameter is a mechanism to get user input for the flow. There are some parameters defined already for each flow. The flow parameter allows you to define additional parameters for the Java code to consume. Consider the sample parameter, BusinessUnit, in the sample customization project. BusinessUnit parameter is a custom type parameter which is a much more powerful parameter that requires an implementation class. The BusinessUnitParameterExecutorImpl class (under com.mycompany.primavera.integration.custom.param package) implements the BusinessUnit parameter.

The **BusinessUnit** parameter is defined in the **Customization.xml** file as follows:

```
<FlowDefinitionCustomization>
    <FlowDefinitionName>Project Data</FlowDefinitionName>
...
    <Parameter>
        <DefaultValue/>
        <Description>A sample custom type parameter</Description>
        <Name>BusinessUnit</Name>
        <Sequence>5</Sequence>
        <Title>Business Unit</Title>
        <Type>Custom</Type>
    </Parameter>
</FlowDefinitionCustomization>
```

Defining Additional Fields

You can customize your application by defining additional fields from the Gateway user interface or in the XML description file as shown below.

The following example, defines two fields: CostCode UDF field, and TotalPlannedCost UDF field.

In P6 EPPM, customers might have fields defined for project, activity, or other objects. They can define them using codes (project code, resource code or activity code) or user defined fields (UDF). Similarly, they might have fields defined in their enterprise application as well that carries the same information as the corresponding fields in P6 EPPM. Hence, they would want these fields to be integrated when they implement their integration.

To provide support for these fields, you start by defining them in an XML description file and then reference the fields later in direct field mappings, foreign field mappings, custom Java field mappings, Groovy mappings, or custom steps. Here is a sample from the customization.xml in the sample customization project that defines the CostCode UDF field for Activity and the TotalPlannedCost UDF field for the project on the P6 side. You can define additional fields for the enterprise application side as well.

```
<CustomMetadata>
	<App>
		<Name>P6</Name>
		<BusinessObject>
			<Name>Activity</Name>
			<Field category="UDF">
				<Description>Cost code UDF Field</Description>
				<Name>CostCode</Name>
				<Type>String</Type>
				<MaxLength>255</MaxLength>
			</Field>
		</BusinessObject>
		<BusinessObject>
			<Name>Project</Name>
			<Field category="UDF">
				<Description>Total planned cost UDF Field</Description>
				<Name>TotalPlannedCost</Name>
				<Type>Double</Type>
			</Field>
		</BusinessObject>

	</App>
</CustomMetadata>
```

Adding Direct Field Mapping

A direct field mapping template defines a series of simple one-to-one field mapping for one object. The integration product that is built on Primavera Gateway should come with a list of direct field mapping templates already, but you can add more if this becomes necessary either from the Gateway user interface or in the relevant XML description file. Here is an example from the customization.xml in the sample customization project that maps the CostCode field of the Activity business object and the TotalPlannedCost field of the Project business object:

```
<FieldMapTemplates>
        <App1Name>Sample</App1Name>
        <App2Name>P6</App2Name>
        <FieldMapTemplate>
                <Description>Cost ceveode mapping for activity object</Description>
                <App1BusinessObjectName>Operation</App1BusinessObjectName>
                <Name>Activity Cost Code Mapping</Name>
                <PDIBusinessObjectName>Activity</PDIBusinessObjectName>
                <FieldMap>
                        <App1>SampleCostCode</App1>
                        <App2>CostCode</App2>
                        <PDI>CostCode</PDI>
                </FieldMap>
        </FieldMapTemplate>
        <FieldMapTemplate>
                <Description>Total planned cost for project object</Description>
                <Name>Project Customization Mapping</Name>
                <PDIBusinessObjectName>Project</PDIBusinessObjectName>
                <FieldMap>
                        <App1>TotalPlannedCost</App1>
                        <App2>TotalPlannedCost</App2>
                        <PDI>TotalPlannedCost</PDI>
                </FieldMap>
        </FieldMapTemplate>
</FieldMapTemplates>
```

Adding Foreign Key Mapping

Foreign keys can now be defined and added in the data dictionary of the source and destination providers, including Gateway from the Gateway user interface or in the XML description file as shown below. Once defined, the mapping can then be defined in a mapping template.

```
<FieldMapTemplates>
    <App1Name>Sample</App1Name>
    <App2Name>P6</App2Name>
    <FieldMapTemplate>
        <Description>Demonstrating mapping foreign key in a mapping template</Description>
        <App1BusinessObjectName>Operation</App1BusinessObjectName>
        <Name>Sample Mapping Template</Name>
        <PDIBusinessObjectName>Activity</PDIBusinessObjectName>
        <FieldMap>
            <App1>SampleWorkOrderElementId</App1>
            <App2>SampleWBSObjectId</App2>
            <PDI>SampleWBSObjectId</PDI>
        </FieldMap>
    </FieldMapTemplate>
</FieldMapTemplates>
```

In the metadata XML file or the customization XML file, mark the field as a foreign key and set the JoinTo to the name of the object that this field is joining to as in the example below:

```
    <Field>
        <Description>Sample WBS Object IDs</Description>
        <Name>SampleWBSObjectId</Name>
        <JoinTo>WBS</JoinTo>
        <Type>ForeignKey</Type>
    </Field>
```

Adding Custom Groovy Field Mapping

Custom Groovy Field Mapping is similar to Custom Java Field Mapping in that they both can handle more complex logics and allow multiple fields to participate at the same time. Custom Groovy Field Mapping is easier to use because the Groovy script is embedded in the customization description XML file, and therefore will not require a separate jar file. Groovy code can be defined either from the Gateway user interface or in the XML description file as shown below.

The syntax of Custom Groovy Field Mapping in the customization description XML file is also similar to Custom Java Field Mapping. The following is an example from Customization.xml in SampleCustomization project.

```
<GroovyFieldMappingTemplates>
   <App1Name>Sample</App1Name>
   <App2Name>P6</App2Name>
   <GroovyFieldMappingTemplate>
      <Description>Sample Groovy resource field mapping</Description>
      <App1BusinessObjectName>Resource</App1BusinessObjectName>
      <App2BusinessObjectName>Resource</App2BusinessObjectName>
      <Name>SampleGroovyResourceFieldMap</Name>
      <PDIBusinessObjectName>Resource</PDIBusinessObjectName>
      <GroovyFieldMapping>
         <Direction>GuestToPDI</Direction>
         <SourceFields>EmployeeName,SampleDate</SourceFields>
         <TargetFields>Name,PDISampleDate</TargetFields>
         <RequireAllFields>true</RequireAllFields>
         <Script>
               <![CDATA[
                     Name = EmployeeName.toUpperCase();
                     if (containsField("EmployeeName")) {
                      {Name} = [EmployeeName].toUpperCase();
                     }
                     if (containsField("SampleDate")) {
                        def cal = new GregorianCalendar();
                        cal.setTime([SampleDate]);
                        cal.add(Calendar.DATE, -1);
                        cal.add(Calendar.HOUR, 2);
                        {PDISampleDate} = cal.getTime();
                         }
                      }
                              ]]>
         </Script>
      </GroovyFieldMapping>
   </GroovyFieldMappingTemplate>
</GroovyFieldMappingTemplates>
```

1) At the top, declare the two applications involved in the integration.

2) Declare what business object from each side is involved. In this example, it is Resource object for all 3 sides.
3) Within one Groovy mapping (GroovyFieldMapping tag), specify the following tags:
   - Direction: The direction of the mapping, it could be App1ToPDI, App2ToPDI, PDIToApp1 or PDIToApp2.
   - SourceFields: Comma-separated field names from the source object.
   - TargetFields: Comma-separated field names from the target object.
   - Script: The script in Groovy code.
   - RequireAllFields: When set to True, this mapping will be skipped unless all the source fields are present in the source object.

Within the Groovy script, use brackets to surround a source field, and curly brackets for a target field as in the following example:

```
{Name} = [EmployeeName].toUpperCase();
```

Where `EmployeeName` is a field from the source object, and `Name` is a field from the target object.

You can also use the `containsField` method to test whether a field exists in the source object. In the above example, the script uses `containsField` to test whether `EmployeeName` or `SampleDate` fields are there, before it executes the logic. This is important to know so as to avoid null pointer exceptions.

When the `RequireAllFields` tag is set to true, the script will only be called when all source fields are present in the source object; no possibility for null pointer exception there. But when the RequireAllFields tag is set to false, the script will be executed even when some source fields are not present in the source object. In the case when a source field is not present in the source object, for a primitive type field, such as integer, long, double types, the value will be set to default value 0; for a string type field, it will be set to default value ""; for a date type field, it will be set to null.

Limitations

For security reasons, the following limitations have been enforced on Groovy capability:

- Loops are not allowed.
- Closure is not allowed
- Class loading or reflection is not allowed
- New classes cannot be defined
- File system access is not allowed
- Network access is not allowed
- Classes under java.lang and java.util only can be accessed, but not any other packages
- Writing one or more expressions and calculating one or more target fields from one or more source fields is allowed

Adding Custom XRefs

Add custom XRef elements in the source application and create the business object mappings between the source provider and the destination provider either using the Object and Fields wizard in the Gateway user interface or in the XML description file as shown below.

For example, the Role object in the Sample provider is mapped to the Role object in Gateway and P6 providers. To create a new flexible object mapping to map the Role object in the Sample provider to the Resource object in Gateway and P6 providers, define a new XRef mapping between the Role object on the Sample (source) side and the Resource object on the Gateway (source) side using the CustomXRef element as shown below:

```
<CustomXRef>
  <XRefDefinition>
    <AppName>Sample</AppName>
    <XRefObjectDefinition>
      <AppObjectName>Role</AppObjectName>
      <PDIObjectName>Resource</PDIObjectName>
      <AppKeyFieldName>ElementId</AppKeyFieldName>
    </XRefObjectDefinition>
  </XRefDefinition></CustomXRef>
```

Adding Flexible Object Mappings

Primavera Gateway providers can be customized to add custom field-mappings, parameters, and steps in business flows. Flexible object mapping is another powerful provider customization feature that allows you define and create new business object mappings between the source and destination provider applications based on your organization's data requirements.

By using this feature, you are no longer limited to only working with the business object mappings supported and delivered within Gateway. Define flexible object mapping to create custom business object mappings to deliver customized solutions to support extensive use-case scenarios to manipulate and share data between applications at the *business object* level.

For example, you can use this feature to convert specific roles in the Sample to Resources in P6 by defining a custom object mapping between a Role object in Sample provider and the Resource object in P6.

To add flexible object mappings:

1) Adding new objects or fields in the Gateway data dictionary and the relevant provider dictionary from the Gateway user interface.

   In an on-premises environment, you can also modify the provider code to support the new object mappings to ensure Gateway pushes the data to the destination provider.

2) Create a customization XML file containing the new object mappings for the business objects on the source side as follows:

   a. Add custom XRef elements and create the business object mappings between the source provider and the destination provider.

For example, the Role object in the Sample provider is mapped to the Role object in Gateway and P6 providers. To create a new flexible object mapping to map the Role object in the Sample provider to the Resource object in Gateway and P6 providers, define a new XRef mapping between the Role object on the Sample (source) side and the Resource object on the Gateway (source) side using the CustomXRef element as shown below:

```
<CustomXRef>
  <XRefDefinition>
    <AppName>Sample</AppName>
    <XRefObjectDefinition>
      <AppObjectName>Role</AppObjectName>
      <PDIObjectName>Resource</PDIObjectName>
      <AppKeyFieldName>ElementId</AppKeyFieldName>
    </XRefObjectDefinition>
  </XRefDefinition></CustomXRef>
```

b. Import the customization file into Gateway.

c. Create field-mapping templates to support the custom XRefs defined above in the Gateway user interface as shown in the code snippet below:

```
<FieldMapTemplates>
  <App1Name>Sample</App1Name>
  <App2Name>P6</App2Name>
  <FieldMapTemplate>
    <Description>Role To Resource Field Mappings</Description>
    <App1BusinessObjectName>Role</App1BusinessObjectName>
    <App2BusinessObjectName>Resource</App2BusinessObjectName>
    <Name>Role To Resource Mappings</Name>
    <PDIBusinessObjectName>Resource</PDIBusinessObjectName>
    <FieldMap>
      <App1>RoleId</App1>
      <App2>Id</App2>
      <PDI>Id</PDI>
    </FieldMap>
    <FieldMap>
      <App1>RoleName</App1>
      <App2>Name</App2>
      <PDI>Name</PDI>
    </FieldMap>
  </FieldMapTemplate>
</FieldMapTemplates>
```

d. Add or edit business flows that use the field-mapping templates containing the flexible object mappings.

e. Create synchronizations that use the business flows.

For a detailed example, see the *Primavera Gateway Online Help*.

Deploying a Customization

Deploy a customization using the following methods:

- ***Deploying Customization Files from Gateway User Interface*** (on page 20)
- ***Deploying Customization Files Using the Configuration Utility for On-Premises*** (on page 20) if your customization has a JAR file (on-premises only)

  If your final customization artifacts contain a JAR file, then deploy it using the Gateway Setup Configuration utility. The configuration utility is a desktop application, and needs direct access to the database. After deploying the customization, you will need to restart the WebLogic domain.

Notes:

- When you install Primavera Gateway, you have the option to install a sample customization project that contains examples that demonstrate how each additional feature can be defined. You will find the XML description file called Customization.xml in the **data** folder. Java source code and Groovy code is included as well.
- You can configure how the Primavera Gateway synchronizes data by interacting with the Primavera Gateway user interface. Additionally, you can create a provider, or customize an existing provider, to further control how the adapter loads the data. We supply a Sample provider that you can use to get started.

Deploying Customization Files from Gateway User Interface

To deploy customization files, import the configuration data files to a Gateway database as follows:

1) Sign in toGateway as a Gateway administrator or Gateway developer:
   http://<host name>:<port>/gateway
2) In the sidebar, select **Configuration**.
3) Select the **Import/Export** tab.
4) Select **Select File**, to browse and select a .zip file or an XML file to be imported.
5) Select **Import**.
   A success message displays.

Notes: Only one zip file can be imported at a time.

Deploying Customization Files Using the Configuration Utility for On-Premises

Use this procedure if a customization includes XML files, jar files, and other files such as Java customizations. If you have installed Gateway on a managed server, see ***Updating the gateway.ear File for On-Premises*** (on page 22) for more details.

Note: If a customization does not include any jar files, and uses Groovy customization, then upload the XML file from the Gateway user interface.

Prerequisites

Ensure the following:

- The **bin** folder of the supported Java JDK is included in the PATH.
- If there are other JDK bin folders in the PATH, it should be listed first.
- Sign in to WebLogic as an administrator and stop the Gateway domain before adding customizations.
- If you are using a non-Microsoft Windows system, type the following command for execute privileges:

  chmod 755 Gateway-Configuration.sh

Procedure

To add customization files in Gateway:

1) Navigate to any of the following folder locations:
   - **<Primavera_Gateway_Home>/gateway/app/utilities** folder

     where

     <Primavera_Gateway_Home> is the installed location of Gateway.
   - **<Gateway Download Folder>/GenericInstaller/GatewayUtility** folder.
2) Run the following command:
   - For Windows installations, run **Gateway-Configuration.bat**
   - For Linux and Solaris installations, run ./**Gateway-Configuration.sh**
3) In the **Primavera Gateway Configuration Utility** dialog box:
   a. Select **Manage Customizations** to update the gateway.ear file and the Gateway database with custom metadata from the customization XML files.

      Note: Select **Next** on each screen to advance to the next step.

   b. In the **Select Gateway ear file (gateway.ear) location** field, enter or select **Browse** to locate the .ear file in the **<Primavera_Gateway_Home>/gateway/app/utilities** folder.
   c. In the **Installed Gateway Customizations** field, select **Add Customization** and navigate to the folder location of the customization file.
   d. Enter the following database connection details:
      - **DBA User Name:** Enter the name of the database administrator.
      - **DBA Password**: Enter the password for the database administrator.
      - **Database Host:** Enter the host name of the Oracle database on which you will be updating the Primavera Gateway database.
      - **Database Host Port**: Enter the port number of the Oracle database.
      - **Database Name:** Enter the Gateway database name and select any of the following methods to connect to the database.
      - **SID**: Enter the SID of the Oracle database.
      - **Service:** Enter the service name of the Oracle database.
      - **Schema Owner:** Enter the name of Gateway schema owner. (This name should match the name that was entered when you installed Primavera Gateway.)

- **Schema Password:** Enter the password for the schema owner.
- Select **Test Connection**. Modify the applicable fields if the connection fails and repeat as necessary.
- Select **Update**. The status field displays a success message.

e. Select **Finish** to exit the configuration utility.

- Redeploy gateway.ear in WebLogic to ensure the changes are reflected in Primavera Gateway. For more details, see ***Updating the gateway.ear File for On-Premises*** (on page 22).

Updating the gateway.ear File for On-Premises

Upgrade Primavera Gateway by updating the gateway.ear in WebLogic.

Prerequisites

Backup the current database and gateway.ear file before copying the new .ear file.

Procedure

Complete the following steps to update the gateway.ear file in WebLogic:

1) Ensure the Primavera Gateway domain is running.
2) Sign in to the WebLogic Administration Server Console for Primavera Gateway domain:
   http://<hostname>:<port>/console
   Where, <*hostname*> and <*port*> is that of the Primavera Gateway domain in WebLogic. The default port is **7001**.
3) Enter the WebLogic administrator user name and password.
4) In the **Change Center** pane, select **Lock & Edit**.
5) In the **Domain Structure** pane, select **Deployments**.
6) Stop the Gateway deployment as follows:
   a. In the **Deployments** tab, select *gateway* in the **Name** column.
   b. Select the **Control** tab.
   c. Select **gateway**.
   d. Select **Stop** to stop the current Gateway deployment.
7) Update the gateway.ear file as follows:
   a. Select the **Configuration** tab.
   b. Select the check box corresponding to *gateway* in the **Name** column.
   c. Select the **Update** button.
   d. In the **Source Path** field, select the **Change Path** button to browse and locate the updated gateway.ear file.
      In the downloaded Gateway software, gateway.ear file is located in the **/GenericInstaller/GatewayUtility/bin** folder.
   e. Select **Finish**.
8) In the **Change Center** pane, select **Activate Changes**.
9) Restart **gateway** deployment.

Restarting the WebLogic Domain for On-Premises

Finally, complete the customizations by restarting the WebLogic domain. Depending on your operating system, proceed as follows to restart the Primavera Gateway domain.

On Windows System for On-Premises

Start and stop the Gateway domain on Windows installations as follows:

Primavera Gateway Installed on an Admin Server

To start Primavera Gateway, from the Windows **Start** menu, select **Oracle, Primavera Gateway,** and then **Start Primavera Gateway.** This invokes `startGatewayAS.cmd`.

To stop Primavera Gateway, from the Windows **Start** menu, select **Oracle, Primavera Gateway,** and then **Stop Primavera Gateway.** This invokes the `stopGatewayAS.cmd`.

Primavera Gateway Installed on a Managed Server

To start Primavera Gateway, from the Windows **Start** menu, select **Oracle, Primavera Gateway,** and then **Start Primavera Gateway.** This invokes `startGatewayMS.cmd`.

To stop Primavera Gateway, from the Windows **Start** menu, select **Oracle, Primavera Gateway,** and then **Stop Primavera Gateway.** This invokes `stopGatewayMS.cmd`.

On Linux and Solaris Systems for On-Premises

Start and stop the Gateway domain on Linux and Solaris installations as follows:

Primavera Gateway Installed on an Admin Server

To start Primavera Gateway on an Admin server:

1) Enter `cd <Primavera Gateway_Home>/WLST/adminserver`
2) Run `./startGatewayAS.sh`

> - **Notes**:
> - You may need to set permissions on the startGatewayAS.sh file. If so, enter the following command: `chmod 755 startGatewayAS.sh`**.**
> - You must use the <PrimaveraGateway_home>\WLST\startGateway*.sh scripts to start the Primavera Gateway domain because these scripts contain the specific memory and timezone settings required by Primavera Gateway.

To stop Primavera Gateway:

1) Enter `cd <Primavera Gateway_Home>/WLST/adminserver`
2) Run `./stopGatewayAS.sh`

> **Note**: You may need to set permissions on the stopGatewayAS.sh file. If so, enter the following command: `chmod 755 stopGatewayAS.sh`

Primavera Gateway Installed on a Managed Server

To start Primavera Gateway on a managed server:

1) Enter: `cd <Primavera Gateway_Home>/wlst/managedserver.`
2) Run: `./startGatewayMS.sh`

> **Note**: You may need to set permissions on the startGatewayMS.sh file. If so, enter the following command: `chmod 755 startGatewayMS.sh`

To stop Primavera Gateway on a managed server:

1) Enter: `cd <Primavera Gateway_Home>`/wlst/managedserver
2) Run: `./stopGatewayMS.sh`

> **Note**: You may need to set permissions on the `stopGatewayMS.sh` file. If so, enter the following command: `chmod 755 stopGatewayMS.sh`

Removing Customization Files

You can remove customization files using any of the following methods:

- ***Removing Customization Files Using the Configuration Utility for On-Premises*** (on page 24)
- ***Removing Customization Files Using Gateway User Interface*** (on page 25)

Removing Customization Files Using the Configuration Utility for On-Premises

If a customization uses a JAR file, then you will need to remove the customization using the Gateway configuration utility only. Otherwise, you can delete it using the Gateway user interface.

Prerequisites

Ensure the following:

- The **bin** folder of the supported Java JDK is included in the PATH.
- If there are other JDK bin folders in the PATH, it should be listed first.
- Sign in to WebLogic as an administrator and stop the Gateway domain before removing customizations.
- If you are using a non-Microsoft Windows system, type the following command for execute privileges:

  chmod 755 Gateway-Configuration.sh

Procedure

To remove customization files using the Gateway configuration utility:

1) Navigate to any of the following folder locations:
   - **<Primavera_Gateway_Home>/gateway/app/utilities** folder
     where **<Primavera_Gateway_Home>** is the installed location of Gateway.
     OR
   - **<Gateway Download Folder>/GenericInstaller/GatewayUtility** folder
2) Run the following command:
   - For Windows installations, run **Gateway-Configuration.bat**
   - For Linux and Solaris installations, run ./**Gateway-Configuration.sh**
3) In the **Primavera Gateway Configuration Utility** dialog box:
   a. Select **Manage Customizations.**

      Note: Select **Next** on each screen to advance to the next step.

   b. In the **Select Gateway ear file (gateway.ear) location** field, select **Browse** to locate the .ear file in the **<Primavera_Gateway_Home>/gateway/app/utilities** folder.
   c. In the **Installed Gateway Customizations** section, select the **Remove** check box for the customization you want to remove.
   d. Enter the following database connection details:
      - **DBA User Name:** Enter the name of the database administrator.
      - **DBA Password**: Enter the password for the database administrator.
      - **Database Host:** Enter the host name of the Oracle database on which you will be updating the Primavera Gateway database.
      - **Database Host Port**: Enter the port number of the Oracle database.
      - **Database Name:** Enter the Gateway database name and select any of the following methods to connect to the database.
      - **SID**: Enter the SID of the Oracle database.
      - **Service:** Enter the service name of the Oracle database.
      - **Schema Owner:** Enter the name of Gateway schema owner. (This name should match the name that was entered when you installed Primavera Gateway.)
      - **Schema Password:** Enter the password for the schema owner.
      - Select **Test Connection**. Modify the applicable fields if the connection fails and repeat as necessary.
      - Select **Update**. The status field displays a success message.
   e. Select **Finish** to exit the configuration utility.

Removing Customization Files Using Gateway User Interface

A customization file can be removed only after you have deleted all instances where it has been used in Gateway. For example, to remove a custom step that has been used in three business flows, you will need to remove it from all the business flows before deleting it from Gateway entirely.

To remove customization files using the Gateway user interface:

1) Sign in to Gateway as a Gateway administrator or Gateway developer: http://<host name>:<port>/gateway
2) In the sidebar, select **Configuration**.
3) Select the **Add-Ins** tab.
4) In the **Name** column, select the customization file you want to remove.
5) Select **Delete**.
6) In the **Confirmation** dialog box, select **Confirm**.

> **Note**: If a customization uses a JAR file, then you can remove the customization using the Gateway configuration utility only.

Appendix - Contents of Customization.XML

In This Section

Customization XML File

This file is loaded into the system by the data loader utility and is used to extend the system by adding any of the following customizations to the Source application, Gateway or the destination application:

- User-defined fields (UDFs)
- Mapping templates
- Groovy field mappings
- Foreign field mappings
- Flow Definition customizations
- Custom metadata
- Custom XRefs
- External custom steps

In an on-premises environment, you can also define:

- Java custom mappings
- Java custom steps
- Custom parameters

Schema File

Customization.xsd

Contents

A **Customization** element containing the following elements:

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	Customization	The optional element that specifies the name of the customization.
Version	string restricted to maxLength(60)	Customization	The optional element that specifies the version of the customization.

Element	Type	Parents	Description
JavaClassPath	string restricted to maxLength(255)	Customization	(On-premises only) The optional element that specifies the path to the customization file that loads the customization functions. This element is required if you are using Java customization.
External	boolean	Customization	The flag to determine if an external custom step is external to Gateway.
GUID	string restricted to maxLength(60)	Customization	The Global Unique Identifier for the custom step.
CustomMetadata	CustomMetadataType	Customization	The optional element that contains zero to three App elements. An example use is to add references to user defined fields (UDFs).
CustomXRef	CustomXRefType	Customization	The optional element that contains zero to many custom XRef elements.
FieldMapTemplates	FieldMapTemplatesType	Customization	The optional element that contains zero to many FieldMapTemplate elements that can be used to map additional fields.
JavaCustomizationMapping	JavaCustomizationMappingType	Customization	(On-premises only) The zero to many optional elements that contain the following elements: ▸ GuestAppName (required if the parent is present). ▸ HostAppName (required if the

Element	Type	Parents	Description
			parent is present). ▸ Zero to many FieldMapping elements.
GroovyFieldMappingTemplates	GroovyFieldMappingTemplatesType	Customization	The optional element that contains zero to many GroovyFieldMappingTemplate elements.
FlowDefinitionCustomization	FlowDefinitionCustomizationType	Customization	The zero to many optional elements that allow you to add custom parameters to a particular flow. Contains the following elements: ▸ FlowDefinitionName element (required if the parent is present). ▸ DisableCompare boolean flag. ▸ Zero to many FlowStep elements. ▸ Zero to many Parameter elements.

CustomMetadataType Table

Element	Type	Parent	Description
App	CustomMetadataAppType For more details, see the CustomMetadataAppType table below.	CustomMetadata	The application to which you are adding a user defined field (UDF).

CustomXRefType Table

Element	Type	Parent	Description
XRefMap	XRefMapType For more details, see the XRefMapType table below.	CustomXRef	The element that contains one or more XRef maps. An XRef map contains a mapping of business objects cross-referenced between two applications.
XRefDefinition	XRefDefinitionType For more details, see the XRefDefinitionType table below.	CustomXRef	The element that contains one or more XRef definitions. An XRefDefinition contains the definitions of business objects cross-referenced between two applications in the XRef map.

XRefMapType Table

Element	Type	Parent	Description
App1Name	string	XRefMap	The name of the application.
App2Name	string	XRefMap	The name of the application.
XRefObject	XRefObjectType For more details, see the XRefObjectType table below.	XRefMap	One or more XRef objects in the XRef map.

XRefDefinitionType Table

Element	Type	Parent	Description
AppName	string	XRefDefinition	The element that contains the application name associated with the XRefDefinitions.

Element	Type	Parent	Description
XRefObjectDefinition	XRefObjectDefinitionType For more details, see the XRefObjectDefinitionType table below.	XRefDefinition	The element that contains the business object associated with the XRefDefinition.

XRefObjectDefinitionType Table

Element	Type	Parent	Description
AppObjectName	string	XRefDefinitionType	The element that contains the application object name associated with the XRefObjectDefinition.
PDIObjectName	string	XRefDefinitionType	The element that contains the corresponding PDI object name in Primavera Gateway associated with the AppObjectName for the XRefObjectDefinition.
AppKeyFieldName	string	XRefDefinitionType	The element that contains the unique AppKeyFieldName for the XRefObjectDefinition.

XRefObjectType Table

Element	Type	Parent	Description
Name	string	XRefObject	The name of the cross-reference object.
App1Name	string	XRefObject	The App1 business object name if different from the Name element.

Element	Type	Parent	Description
PDIName	string	XRefObject	The PDI business object name if different from the Name element.
App2Name	string	XRefObject	The App2 business object name if different from the Name element.
App1PrimaryKeyFieldName	string	XRefObject	The name of the primary key field in the database of the application.
PDIPrimaryKeyFieldName	string	XRefObject	The name of the primary key field in the Gateway database.
App2PrimaryKeyFieldName	string	XRefObject	The name of the primary key field in the database of the application.

CustomMetadataAppType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	App	The name of the application to which you are adding a user-defined field (UDF).
ObjectCategory	ObjectCategoryType	App	Provides a mechanism for classifying objects.
FieldCategory	FieldCategoryType	App	Provides a mechanism for classifying fields. It is defined here and referenced by attribute "category" when a field is defined within a business object. For example, in the P6 provider, "udf" is defined as a field category to indicate that this field is an actual user-defined field in the P6 EPPM side.

Element	Type	Parent	Description
BusinessObject	BusinessObjectType	App	The zero to many business object elements that are containers for field elements that relate to the business object specified by the Name element. All custom UDF fields defined in this element will be included when updating the business object.

ObjectCategoryType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	ObjectCategory	The name of the object category.
Description	string restricted to maxLength(255)	ObjectCategory	The description of the object category.

FieldCategoryType Table

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	FieldCategory	The name of the field category.
Description	string restricted to maxLength(255)	FieldCategory	The description of the field category.

BusinessObjectType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	BusinessObject	A description of the business object type.

Element	Type	Parent	Description
Name	string restricted to maxLength(60)	BusinessObject	The name of the business object to which you are adding the user defined field.
Field	FieldType	BusinessObject	All custom UDF fields defined inside this element will be included when updating the business object.

FieldType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	Field	The element that describes the field. This description is presented in the Fields table on the **Data Dictionary** tab in the Primavera Gateway user interface.
Name	string restricted to maxLength(60)	Field	The element that determines the name of the field. This name is presented in the Fields table of the **Data Dictionary** tab in the Primavera Gateway user interface.
JoinTo	string restricted to maxLength(60)	Field	The element that contains the name of the object to which it is being joined to.

Element	Type	Parent	Description
Type	string restricted to maxLength(10) 'Boolean' 'DateTime' 'Double' 'Int' 'String' 'ForeignKey' 'Enum'	Field	The element that defines the data type of the value of the field's data when it is synchronized.
MaxLength	positiveInteger	Field	The optional element that defines the maximum length of the value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is String to limit the field's data values.
MaxValue	double	Field	The optional element that defines the maximum value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is Int to limit the field's data values.
MinValue	double	Field	The optional element that defines the minimum value of the field's data when it is synchronized. This element can be used with the **Type** element whose content is Int to limit the field's data values.

Element	Type	Parent	Description
FieldValue	FieldValueType	Field	An optional element that defines the value of an enum field type. Field elements can contain zero or more FieldValue elements if the Field element contains a **Type** element whose content is Enum.

FieldValueType Table

Element	Type	Parent	Description
Description	string restricted to maxLength(255)	FieldValue	An optional element that describes the fieldValue.
Value	string restricted to maxLength(60)	FieldValue	An optional element that defines an enumerated value.

FieldMapTemplatesType Table

Element	Type	Parents	Description
App1Name	string restricted to maxLenth(60)	Customization	The name of the application.
App2Name	string restricted to maxLenth(60)	Customization	The name of the application.
GuestAppName	string restricted to maxLength(60)	Customization	The name of the guest application.
HostAppName	string restricted to maxLength(60)	Customization	The name of the host application.

Element	Type	Parents	Description
FieldMapTemplate	FieldMapTemplateType	Customization	The container element that contains zero to many FieldMapTemplates included in the customization.

FieldMapTemplateType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	FieldMapTemplate	The text that is used in the Primavera Gateway user interface to provide a description of the field mapping.
App1BusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The Guest business object name if different from the PDIBusinessObjectName element.
App2BusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	The Host business object name if different from the PDIBusinessObjectName element.
GuestBusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	Guest business object name if different from PDIBusinessObjectName element.
HostBusinessObjectName	string restricted to maxLength(60)	FieldMapTemplate	Host business object name if different from PDIBusinessObjectName element.
Name	string restricted to maxLength(60)	FieldMapTemplate	The text that is used in the Primavera Gateway user interface to provide a name for the field mapping.

Element	Type	Parents	Description
PDIBusinessObject Name	string restricted to maxLength(60)	FieldMapTempl ate	The PDI business object name. Also indicates the guest and\or the host business object name if the GuestBusinessObjectNa me or the HostBusinessObjectNam e elements are omitted.
Condition	ConditionType	FieldMapTempl ate	The condition associated with the FieldMapTemplate.
FieldMap	FieldMapType	FieldMapTempl ate	The element that specifies the field map.

ConditionType Table

Element	Type	Parents	Description
App1Fields	string	Condition	The fields associated with the condition from the application designated as App1.
App1Script	string restricted to maxLength (2048)	Condition	The script associated with the condition for the application designated as App1.
App2Fields	string	Condition	The fields associated with the condition for the application designated as App2.
App2Script	string restricted to maxLength (2048)	Condition	The script associated with the condition for the application designated as App2.

FieldMapType Table

Element	Type	Parents	Description
App1	string restricted to maxLength(60)	FieldMap	The name of the application.
App2	string restricted to maxLength(60)	FieldMap	The name of the application.
Guest	string restricted to maxLength(60)	FieldMap	The element that specifies the name of the field that is involved in the mapping from the provider side of the mapping.
Host	string restricted to maxLength(60)	FieldMap	The element that specifies the name of the field that is involved in the mapping from the Primavera provider side of the mapping.
PDI	string restricted to maxLength(60)	FieldMap	The element that specifies the name of the field that is involved in the mapping from the Gateway side.

JavaCustomizationMappingType Table (On-premises only)

Element	Type	Parents	Description
App1Name	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the application.
App2Name	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the application.
GuestAppName	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the application in the App table in the Gateway database.

Element	Type	Parents	Description
HostAppName	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the Primavera application in the App table in the Gateway database.
Condition	ConditionType	JavaCustomizationMapping	The condition associated with the java customization mapping.
FieldMapping	FieldMappingType	JavaCustomizationMapping	The element that specifies the field mapping.

GroovyFieldMappingTemplatesType Table

Element	Type	Parents	Description
App1Name	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the application.
App2Name	string restricted to maxLength(60)	JavaCustomizationMapping	The name of the application.
GuestAppName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The name of the application designated as Guest application in the App table in the Gateway database.
HostAppName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The name of the application designated as the Host application in the App table in the Gateway database.
GroovyFieldMappingTemplate	GroovyFieldMappingTemplateType	GroovyFieldMappingTemplates	The name of the GroovyFieldMappingTemplate associated with the applications designated as Guest and Host.

GroovyFieldMappingTemplateType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	GroovyFieldMappingTemplates	The name of the application designated as Guest application in the apptable in the Gateway database.
App1BusinessObjectName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The name of the business object in the application.
App2BusinessObjectName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The name of the business object in the application.
GuestBusinessObjectName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The Guest business object name if it is different from the PDIBusinessObjectName element.
HostBusinessObjectName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The Host business object name if it is different from the PDIBusinessObjectName element.
Name	string restricted to maxLength(255)	GroovyFieldMappingTemplates	The name of the Groovy business object.
PDIBusinessObjectName	string restricted to maxLength(60)	GroovyFieldMappingTemplates	The name of the PDIBusinessObjectName in the Gateway data dictionary.
Condition	ConditionType	GroovyFieldMappingTemplates	The Groovy condition associated with the Groovy field mapping template.
GroovyFieldMapping	GroovyFieldMappingType	GroovyFieldMappingTemplates	The GroovyFieldMapping being associated with the GuestBusinessObjectName, HostBusinessObjectName, and PDIBusinessObjectName.

GroovyFieldMappingType Table

Element	Type	Parents	Description
Direction	string restricted to maxLength(10) 'GuestToPDI' 'HostToPDI' 'PDIToGuest' 'PDIToHost' 'App1ToPDI' 'App2ToPDI' 'PDIToApp1' 'PDIToApp2'	GroovyFieldMapping	The direction of the data flow between the source application and destination application in the GroovyFieldMapping.
SourceFields	string	GroovyFieldMapping	The fields in the source application referenced in the GroovyFieldMapping.
TargetFields	string	GroovyFieldMapping	The fields in the target application referenced in the GroovyFieldMapping.
RequireAllFields	boolean	GroovyFieldMapping	The flag to determine if all fields are required for the GroovyFieldMapping.
Script	string restricted to maxLength(1024)	GroovyFieldMapping	The element that contains the Groovy code.

FieldMappingType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	FieldMapping	The description of the custom java field map.
App1BusinessObjectName	string restricted to maxLength(60)	FieldMapping	The name of the business object in the application.
App2BusinessObjectName	string restricted to maxLength(60)	FieldMapping	The name of the business object in the application.

Element	Type	Parents	Description
GuestBusinessObjectName	string restricted to maxLength(60)	FieldMapping	The Guest business object name if different from PDIBusinessObjectName element.
HostBusinessObjectName	string restricted to maxLength(60)	FieldMapping	The Host business object name if different from PDIBusinessObjectName element.
Name	string restricted to maxLength(255)	FieldMapping	The name of the field map that is used to look up the Java class in the code that you have written that contains customization logic.
PDIBusinessObjectName	string restricted to maxLength(60)	FieldMapping	The Gateway business object name in the field mapping.
Fields	FieldsType	FieldMapping	The container element for the fields defined inside this element to be included when loading the business object.

FieldsType Table

Element	Type	Parents	Description
App1	FieldsFieldType	Fields	The name of the application.
App2	FieldsFieldType	Fields	The name of the application.
Guest	FieldsFieldType	Fields	The element that specifies the name of the fields in the mapping from the application designated as the Guest application.

Element	Type	Parents	Description
Host	FieldsFieldType	Fields	The element that specifies the name of the fields in the mapping from the Primavera application designated as the Host application.
PDI	FieldsFieldType	Fields	The element that specifies the name of the fields that is involved in the mapping from the Gateway side.

FlowDefinitionCustomizationType Table

Element	Type	Parents	Description
FlowDefinitionName	string restricted to maxLength(60)	FlowDefinition Customization	The name of the flow to which the parameters are to apply.
Application Name	string restricted to maxLength(60)	FlowDefinition Customization	The name of the application to which the customization applies.
DisableCompare	boolean	FlowDefinition Customization	The flag when set to true, causes compare to be disabled for **all** FlowTypes.
AppType	string restricted to maxLength(10) 'Guest' 'Host' 'PDI'	FlowDefinition Customization	The role of the application in the flow.
FlowSide	string restricted to maxLength(12) 'Source' 'Destination' 'PDI'	FlowDefinition Customization	The role of the application in the data flow.
FlowBusinessObject	FlowBusinessObject Type	FlowDefinition Customization	The business objects associated with the flow.

Element	Type	Parents	Description
FlowStep	FlowStepType	FlowDefinition Customization	The element that contains flow steps that define an end-to-end synchronization.
Parameter	ParameterType	FlowDefinition Customization	The element that specifies the properties of the parameter.

FlowStepType Table

Element	Type	Parents	Description
Description	string restricted to maxLength(255)	FlowStep	The element that describes the purpose of the flow step.
Name	string restricted to maxLength(60)	FlowStep	The element that determines the name of the flow step.
OwnerAppType	string restricted to maxLength(10) 'Guest' 'Host'	FlowStep	The element that the system uses to determine which document format to use when parsing XML document data. For example, specifying the OwnerAppType as Host and P6 is the host of the flow causes the system to ask the P6 provider to provide the implementation of this step.
FlowSide	string restricted to maxLength(12) 'Source' 'Destination'	FlowStep	The element that the system uses to determine whether the data flow originates from the source or the destination application.

Element	Type	Parents	Description
Sequence	int	FlowStep	A number that determines when the step will run relative to the other steps in the flow. When this element is used in a customization.xml file, you can use the following SQL commands to determine the existing sequencing, from which you can determine an appropriate sequence number for a particular flow type. **Note**: This example returns the sequence number for the Export Project Data flow type. Revise the commands as appropriate for the other flow types. select fd.flowdef_name, fs.flowstep_name, fs.flowstep_type, fs.flowstep_seqno from flowstep fs, flowdef fd where fs.flowdef_id = fd.flowdef_id and fd.flowdef_name = 'Export Project Data' order by fd.flowdef_name, fs.flowstep_seqno;
Type	string restricted to maxLength(20) 'Custom' 'Script'	FlowStep	The element that specifies the flow step type. 'Custom' is the only valid value when this element is used in a customization.xml file.
Script	string	FlowStep	

FlowBusinessObjectType

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	FlowBusinessObjectType	The name of the flow business object type.

ParameterType Table

Element	Type	Parents	Description
DefaultValue	string restricted to maxLength(255)	Parameter	The default value for this custom parameter that is used when the parameter is included in the flow and the value is not changed.
Description	string restricted to maxLength(255)	Parameter	The element that provides a description of the parameter.
EnumerationOptions	EnumerationOptionsType For more details see the EnumerationOptionsType Table.	Parameter	The element that contains the list of enumerated values when the Parameter element is specified as an enum.
Name	string restricted to maxLength(60)	Parameter	The element that provides the name of the parameter. The name is used to lookup the parameter in the provider Java code.
Sequence	int	Parameter	The element that provides the mechanism for ordering the parameters.
Title	string restricted to maxLength(255)	Parameter	The element that provides the display title of the parameter.

Element	Type	Parents	Description
Type	string restricted to maxLength(15) Boolean DateTime Double Int String Password Enum Custom	Parameter	The element that defines the data type of the value of the field's data when it is synchronized.

EnumerationOptionsType Table

Element	Type	Parents	Description
Enumeration	EnumerationType	EnumerationOptions	The element that specifies the name of the enumeration.

FieldsFieldType Table

Element	Type	Parents	Description
Name	string restricted to maxLength(60)	Fields	The element that specifies the name of the field.

EnumerationType Table

Element	Type	Parents	Description
Name	string	Fields	The element that specifies the name of the enumeration type.

Copyright

Oracle Primavera Gateway Provider Customization Guide