Oracle® Communications Service Catalog and Design

Design Studio Modeling Basics

Release 8.1
F96243-01
July 2024

Oracle Communications Service Catalog and Design Design Studio Modeling Basics, Release 8.1

F96243-01

Contents

5 Working with Conceptual Models

About Design Studio Help

This Help system contains information about the procedures and tasks that are necessary to configure Oracle Communications applications using Oracle Communications Service Catalog and Design - Design Studio.

Design Studio is a design tool that unifies and accelerates the creation and delivery of services across Oracle Communications and minimizes the cost of ownership for operators and systems integrators. Design Studio simplifies the creation of order management workflows and rule logic, inventory assign and design metadata, and of activation service and network actions. Design Studio enables packaging, versioning, collaboration, and deployment with reduced time to market for new services.

Audience

This guide is intended for business analysts, architects, development managers, developers, and designers who are responsible for system integration or solution development involving the Oracle Communications operational support systems applications.

Ideally, you should be knowledgeable about your company's business processes, the resources you need to model, and any products or services your company offers.

Documentation Accessibility

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

Access to Oracle Support

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

Diversity and Inclusion

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

1

Getting Started with Design Studio

The integrated development environment (IDE) of Oracle Communications Service Catalog and Design - Design Studio provides a user interface to manage and configure data across Oracle Communications Service Fulfillment products.

Using Design Studio, you can:

- Manage and configure products, services, and associated data from a single application while abstracting underlying Business Support Systems (BSS) and Operational Support Systems (OSS) application interfaces during configuration.
- Enable configurable OSS software to accommodate new products that bundle existing services and modify existing configurations.
- Provide customized views to access data in multiple ways, display and modify configurations graphically, provide impact analysis capabilities during modelling to inform users of change impact, validate the integrity of configurations, and assist users with resolutions when issues arise.
- Provide facilities to capture requirements, design, and implementation details.

  Flexibly deploy, test, and document configurations as they evolve.

When getting started with Design Studio, see the following topics:

About Design Studio Platform

Design Studio is built on an extensible platform that provides complementary tools and facilitates the creation of service fulfillment solutions.

About Eclipse

Design Studio uses Eclipse as a product base and as an IDE. Eclipse supports application development tool construction, independent tool vendors, GUI and non-GUI application development, numerous content types (including Java, HTML, C, and XML), tool integration, and use of Java language for writing the tools.

Design Studio uses Eclipse as a product framework and to support plug-in architecture and customizations.

About Java Development Tools

Java Development Tools (JDT) provides a set of workbench plug-ins that add the capabilities of a full-featured Java IDE to the Eclipse platform. JDT plug-ins provide APIs that can be further extended by other tool builders. Additionally, the JDT includes a built-in Java compiler that compiles Java code and creates error messages when compilation fails.

Related Topics

Getting Started with Design Studio

About Design Studio Naming Conventions

When naming Design Studio entities, consider the following:

Namespace

A namespace is a collection of names used in XML documents as element types and attribute names. The namespace functionality in Design Studio enables you to differentiate between elements and attributes that have the same name but come from different sources.

For example, using namespaces in Oracle Communications Order and Service Management (OSM) enables you to separate OSM models (tasks, processes, order templates, workgroups, and worklists) into specific services in your Operational Support System (OSS) environment. Each service can be implemented independently by a different team, then deployed into a single OSM run-time environment.

Folders

Use Design Studio folders to organize entities within a single project. Design Studio folders are not group-specific; rather, they can contain different types of entities. For example, you can create a single folder to contain all of your task and process entities. In this example, the folder appears under the Studio Projects view **Process** directory and under the Studio Projects view **Tasks** directory. If you rename the folder in one location, Design Studio updates the name in all locations. When you are creating new entities, Design Studio uses the last used folder name, irrespective of entity type, as the folder default value.

Entities

When naming entities in Design Studio, you must ensure that the entity names are unique by entity type. For example, you cannot name two task entities with the same name. However, because Design Studio enforces a naming restriction to define name uniqueness by entity name and entity type, you can create identical names for different entity types. For example, you can model a task entity and a process entity with the name **AddDSL**.

If you create, rename, or import an entity with a name that is identical to the same type of existing entity, Design Studio generates a problem marker. You cannot deploy a cartridge until you have corrected all existing problem markers in the cartridge.

Some Design Studio features have additional conventions. Additionally, operating systems may impose naming restrictions. Oracle recommends that you comply with these naming conventions.

> **Note:**
>
> Rename Design Studio entities in the Studio Design perspective only. Do not rename entities from within the Navigator view.

Related Topics

Getting Started with Design Studio

Defining Preferences

Design Studio enables you to define preferences that apply to your entire workspace. For example, you can specify whether to package cartridges during incremental builds and define groups of languages with which you intend to work. When defining preferences, see the following topics:

- Defining Packaging Preferences
- Defining Language Preferences
- Defining Conceptual Model Preferences
- Defining Data Dictionary Preferences
- Defining Dictionary View Preferences
- Defining Local History Preferences
- Retaining Workspace Preferences

Defining Packaging Preferences

Before you can deploy a cartridge project to a run-time environment, you must determine which entities, libraries, and resources to include (or package) in the project. Typically, Design Studio automatically packages your projects during incremental builds. You can disable this functionality and defer packaging until project deployment.

> **Note:**
>
> Defer packaging only for large projects when packaging is slowing incremental builds. Do not enable this option for systems used for command line builds because it prevents builds from producing complete archives.
>
> Design Studio always packages the entities, libraries, and resources for clean builds.

To define packaging preferences:

1. From the **Window** menu, select **Preferences.**

   The Preferences dialog box appears.
2. Select **Oracle Design Studio**.
3. Select **Defer packaging**.

   Design Studio will not package the project during incremental builds; packaging is deferred until you deploy the project.
4. Click **OK**.

Related Topics

Defining Preferences

Defining Language Preferences

Design Studio supports multiple languages for fields in run-time applications. Use the language preference settings to define the languages that you intend to use in your cartridges and to define the language with which you prefer to work.

To define language preferences:

1. From the **Window** menu, select **Preferences**.

   The Preferences dialog box appears.
2. Select **Oracle Design Studio**.
3. Click **New**.

   The Add Language dialog box appears.
4. Select a language.
5. Click **OK**.

   Design Studio adds the language to the Languages group.
6. (Optional) Define the language display priority.

   When multiple languages appear in the Languages group, use the **Up** and **Down** buttons to reposition the language display priority. The language display priority controls the order in which the languages appear in Design Studio language drop-down lists.
7. (Optional) Click **Remove** to delete a language from the Languages group.
8. In **Preferred Language**, select the language in which you prefer to work.
9. Click **OK**.

Related Topics

Defining Preferences

Defining Conceptual Model Preferences

Conceptual model preferences enable you to change some validation errors to warnings. For example, you can change the validation severity during conceptual model design if you are working in a single functional area, and when your design work does not require solution validity across all functional areas. By default, Design Studio displays validation errors for all functional areas.

> **Note:**
>
> The Conceptual Model Preferences page displays only the functional areas that are defined in the workspace. If no functional areas are defined in the workspace, the options defined in this procedure are not available.

To define conceptual model preferences:

1. From the **Window** menu, select **Preferences**.

   The Preferences dialog box appears.

2. In the left-column menu tree, expand the **Oracle Design Studio** folder.
3. Click **Conceptual Model Preferences**.
4. Select one of the following:
   - To relax the validation severity for a functional area, select **Model entities relevant to specific Functional Areas**.
   - To retain the default validation severity for all functional areas, select **Entire Model (all Functional Areas)**. This is the default setting.
5. For each functional area, select one of the following:
   - To relax the validation severity for a functional area, select **Warning**. For example, select this option for functional areas that are not directly relevant to the functional area in which you are working.
   - To retain the default validation severity for a functional area, select **Error**. Select this option for functional areas in which you intend to work.
6. Click **Apply**.
7. Click **OK**.
8. Clean and rebuild the projects in the workspace.

Related Topics

Conceptual Model Preferences Page

About Conceptual Model Preferences

About Conceptual Model Preferences

A conceptual model spans multiple functional areas, and includes configuration related to commercial order management, service order management, and technical order management. Your role determines whether you work on the entire model or in a single functional area. For example, an enterprise architect is concerned with the entire model, and needs to ensure that all products, customer facing services, service actions, components, resource facing services, resources, and other conceptual model entities are fully modeled and include valid references and relationships. Enterprise architects need to see validation errors for all missing conceptual model entities and for unresolved references.

A conceptual model may be developed by a team of developers working on different projects. In this organization, some developers work on projects that include only products and references to customer facing services. Other developers may work on projects that include only customer facing services and service actions. While other developers may work with sets of projects that include the resource facing services and resources for a specific technology.

In this organization, there are multiple projects that comprise the entire solution. Developers working in projects related to the service order management and technical order management layers may not be interested in the projects related to commercial order management. These developers will likely want to include in their workspace only those projects in the model that are relevant to their work.

If you load only a subset of the conceptual model projects into your workspace, Design Studio generates validation errors because the system detects missing entities, references, and relationships. You can reduce the number of errors and relax the validation in areas outside of the scope of your work by defining conceptual model preferences. The Conceptual Model Preferences page in Design Studio enables you to define a validation severity level for each functional area defined in your conceptual model.

Based on the settings that you define for a functional area on the Conceptual Model Preferences page, Design Studio uses the following rules to evaluate the validation severity levels:

- For each functional area, Design Studio determines the provider functions that are impacted.
- Design Studio generates validation errors for poorly-formed named relationships defined between source and target entities in the impacted provider functions. Validation issues in all other provider functions are identified as warnings.
- Design Studio generates action-related validation errors (such as mandatory and multiple action checks) for actions named as a source or target for impacted provider functions and for actions directly associated to the specified functional area.
- Design Studio generates realization validation errors for source entities named in the impacted provider functions, including actions. Realization validation issues in all other provider functions are identified as warnings. Design Studio generates fulfillment pattern reference-related errors for source entities in the impacted provider functions. Fulfillment pattern reference-related issues in all other provider functions are identified as warnings.

Conceptual Model Preferences Page

Use the Conceptual Model Preferences page to select a validation severity level for each functional area.

Field	Use
Enforce referential integrity for	Select one of the following: • To relax the validation severity for a functional area, select **Model entities relevant to specific Functional Areas**. • To retain the default validation severity for all functional areas, select **Entire Model (all Functional Areas)**. This is the default setting.
Select Functional Areas to Validate	For each functional area, select one of the following: • To relax the validation severity for the functional area, select **Warning**. This is the default setting. • To retain the default validation severity for the functional areas, select **Error**.

Defining Data Dictionary Preferences

You define Data Dictionary preference settings to specify the tree depth to which the Data Dictionary tree can expand.

To define Data Dictionary preferences:

1. From the **Window** menu, select **Preferences**.

   The Preferences dialog box appears.
2. In the left-column menu tree, expand the **Oracle Design Studio** folder.
3. Click **Data Dictionary**.

   See "Data Dictionary Preferences Editor" for more information.
4. In the **Expansion Level** field, specify the tree depth to which the Data Dictionary can expand.

5. Click **Apply**.
6. Click **OK**.

   Design Studio restarts.

Data Dictionary Preferences Editor

Use the Data Dictionary preferences page to specify the expansion level of the Data Dictionary.

Field	Use
Expansion Level	Enter the tree depth to which the Data Dictionary tree can expand. The default is 6. If you change the default setting, you must restart Design Studio for the change to take effect. **Note**: Expanding the tree depth above 9 may adversely affect system performance.

Related Topics

Defining Data Dictionary Preferences

Defining Dictionary View Preferences

The Dictionary View Preferences page enables you to configure the manner in which Design Studio initially filters the entities that appear in the Dictionary view, when viewing entity types in an editor that is linked to the Dictionary view.

Filtering options enable you to determine the types of entities that initially appear for entities in a specific project type or for all projects in the workspace, and for each entity type within a project.

To define Dictionary view preferences:

1. In Design Studio, from the **Window** menu select **Preferences**.

   The Preferences dialog box appears.
2. In the Preferences dialog box menu tree, select **Oracle Design Studio** and then select **Dictionary view**.

   The Dictionary view preferences page appears.
3. In the **Select Cartridge Project Type** field, select an option to define filter options for entities in a specific project type.
4. In the Entity types column, select an entity type to define the filtering options for that entity type.
5. In the Default Entity Filter Types column, do any of the following:
   - Click **Select** to add additional default entity filter types to the entity type configuration.
   - Select any of the default entity filter types and click **Remove** to remove the filter type from the entity type configuration.
6. Click **OK**.

   Design Studio adds updates to the Default Entity Type Filter Types column. The next time you open the entity type in an editor (and if you have linked the editor with the Dictionary view), the Dictionary view initially displays all available entities of the defined types in the project or in any dependent projects.

Dictionary View Preferences Page

Use the Dictionary View Preferences page to configure the manner in which Design Studio initially filters the entities that appear in the Dictionary view, when viewing entity types in an editor that is linked to the Dictionary view.

Field	Use
Select All Cartridge Project Type	Select an option to define filter options for entities in a specific project type.
Default Entity Filter Types	Displays the entity types that initially appear in the Dictionary view, when viewing entity types in an editor that is linked to the Dictionary view.
Remove	Click to remove a filter type from the entity type configuration.
Select	Click to add additional default entity filter types to the entity type configuration.

Related Topics

Defining Dictionary View Preferences

Defining Local History Preferences

The Eclipse platform includes a Local History feature which maintains copies of saved files. These copies are maintained only in your workspace. The Local History feature provides a recovery file on your local file system for work not yet committed to source control, and you can use this feature to complement your source control system.

> **Note:**
>
> Oracle does not recommend using the Local History feature for primary backup, as it is subject to media failure and is coupled to a specific workspace.

You define preferences for the Local History functionality. For example, Oracle recommends that you change the default settings to define a longer retention period for added security.

To define Local History preferences:

1. From the Design Studio **Window** menu, select **Preferences**.

   The Preferences dialog box appears.

2. Select **General**, then select **Workspace**, and then select **Local History**.

   The Local History preferences page appears.

3. Select **Limit history size**.

4. In **Days to keep files**, enter **28**.

   Increasing the value in this field to **28** enables recoveries even after extended work stoppages (for example following vacations). Oracle recommends, however, that you check changes into source control regularly.

5. In **Maximum entries per file**, enter **50**.
6. In **Maximum file size (MB)**, enter **1**.
7. Click **OK**.

For more information about the Local History feature, see the Eclipse Help.

Retaining Workspace Preferences

The **.metadata** workspace folder contains information about your workspace. You can back up your workspace preferences and recreate workspaces by exporting and then importing workspace preferences.

For more information about importing and exporting workspace preferences, see the *Eclipse Workspace User Guide*.

2

Working with Design Studio Projects

Projects contain folders and files representing entities that you use to model Design Studio cartridges, which you deploy to servers. You can use projects for version management, sharing, and resource organization. All Design Studio configuration is contained in a project.

The most common types of projects you use in Design Studio are:

- Cartridge projects, which contain collections of entities and supporting artifacts that represent a cartridge deployed to a run-time environment.
- Model projects, which contain data models common to multiple cartridge projects.
- Environment projects, which you use to manage attributes associated with your run-time environments.

When working with projects, see the following topics:

Importing Projects

You can import data from external sources into your Design Studio workspaces. For example, if you have purchased cartridges from Oracle, you can import them into Design Studio and reuse their components to create your projects.

> **Note:**
>
> There are two different import methods available: an Eclipse method and a Design Studio method. Always use the Design Studio method. Import projects using the Studio Projects view contextual menu or with the **Import Studio Project** menu action available in the **Studio** menu. Using the Eclipse import functionality may cause unpredictable results and may require that you restart Design Studio.
>
> Do not distribute projects among team members by using archive files that contain a workspace and the set of projects. If you receive from a team member an archive file that contains a workspace and a set of projects, unzip the file and import the projects using the **Import Studio Project** menu action.

When you import a project, it becomes a project in the current workspace. Some projects are sealed, meaning that they are read-only. Sealed projects cannot be modified without first being unsealed. See "Sealing Projects" and "Unsealing Projects" for more information.

When importing projects into Design Studio, see the following topics:

- Importing Projects into Design Studio Using Root Directories
- Importing Projects into Design Studio Using Archive Files

Importing Projects into Design Studio Using Root Directories

To import projects into Design Studio using the root directory:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Studio Projects** tab.

   The Studio Projects view appears.
3. Right-click in the Studio Projects view and select **Import**, and then select **Import Project**.

   The Import Projects dialog box appears.
4. Select **Select root directory**.

   > **Note:**
   >
   > If your project is contained in an archive file (such as a TAR file or ZIP file) select Select Archive File. See "Importing Projects into Design Studio Using Archive Files" for more information.
5. Click **Browse**.
6. Locate the directory containing the project and select it.
7. Click **OK**.
8. In the Projects area, select the project to import.
9. Ensure that the **Copy projects into workspace** check box is selected.
10. Click **Finish**.

    Design Studio adds the project to your workspace.

> **Note:**
>
> If you import projects that have dependencies to other projects that are not in the current workspace, Design Studio displays an error. Import all dependent projects, then clean all projects to remove the errors. See "Running Clean Builds" for more information.

Related Topics

Importing Projects into Design Studio Using Archive Files

Importing Projects

Importing Projects into Design Studio Using Archive Files

To import projects in Design Studio using the archive file:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Studio Projects** tab.

   The Studio Projects view appears.
3. Right-click in the Studio Projects view and select **Import**, and then select **Import Project**.

   The Import Projects dialog box appears.
4. Select **Select archive file**.
5. Click **Browse**.
6. Navigate to archive file and select it.
7. Click **Open**.
8. In the Projects area, select the project to import.
9. Click **Finish**.

   Design Studio adds the project to your workspace.

> **Note:**
>
> If you import projects that have dependencies to other projects that are not in the current workspace, Design Studio displays an error. Import all dependent projects first, then clean all projects to remove the errors. See "Running Clean Builds" for more information.

Related Topics

Importing Projects into Design Studio Using Root Directories

Importing Projects

Exporting Projects

To facilitate sharing projects across teams, you can export projects to archive files. The archive files can be subsequently imported into a different Design Studio workspace.

To export a project from Design Studio:

1. From the **File** menu, select **Export**.

   The Export dialog box appears.
2. Expand the **General** folder.
3. Select **Archive File**.
4. Click **Next**.

   The Archive File dialog box appears.
5. Select the projects and resources to export.
6. In the **To archive file** field, specify the location of the archive file.

   Or click **Browse** to locate the archive file.
7. In the Options area, select the following:
   - The archive file format
   - Your compression preferences
   - The **Create directory structure for files** option
8. Click **Finish**.

   Design Studio creates the archive file in the specified location.

Related Topics

Importing Projects

Closing Projects

To reduce the amount of memory required and to improve build times, you can close projects not in use. When you close a project, the resources no longer appear in the workbench area. The projects remain in your local file system, and you can reopen them at any time.

> **Note:**
>
> Oracle recommends that you close projects from the Studio Projects view context menu, as described below.

To close a project:

1. From the Studio Projects view, right-click an open project.

   The context menu appears.
2. Select **Close Project**.

   Oracle Communications Design Studio prompts you to save any unsaved work in the **resources** directory. You can select specific files to save from a list of unsaved files.

Related Topics

Opening Projects

Opening Projects

You can keep multiple projects open in your workspace. Resources included in open projects are available for modeling.

> **Note:**
>
> Oracle recommends that you open projects from the Studio Projects view context menu, as described below.

To open a project:

1. From the Studio Projects view, right-click a closed project.

   The context menu appears.

2. Select **Open Project**.

   The project becomes active and the project resources are available for use in Design Studio.

Related Topics

Closing Projects

Defining Cartridge Project Target Versions

Cartridge project target versions specify the version of the run-time application instance to which a cartridge project will be deployed. Design Studio builds your project to be compatible with the run-time software version specified in the **Target Version** field on the Project editor **Properties** tab. For example, the **Target Version** field can be defined as **7.3.0** or **7.2.4** for cartridge projects that you deploy to a 7.3.0 or 7.2.4 run-time environment, respectively. Design Studio automatically sets the Java execution environment based on the selected target version.

> **Note:**
>
> Design Studio is compatible with specific Oracle Communications applications releases. See Service Catalog and Design Compatibility Matrix, available at the Oracle Help Center, for more information.

When working with Design Studio for Inventory and Design Studio for Network Integrity cartridge projects, ensure that you import into your workspace the correct version of any required model projects. The target version defined for cartridge projects in any given workspace must be defined with the same version as defined for the required model projects.

For example, Design Studio for Inventory cartridge projects have dependencies on the ora_uim_model project and the ora_uim_mds project. If your workspace contains 7.3.0.0.0 Inventory cartridge projects, you must import the 7.3.0.0.0 versions of the ora_uim_model project and the ora_uim_mds project. See "About Inventory Cartridge Project Dependencies" and the *UIM Cartridge and Technology Guide* for more information about required Inventory

model projects. See "Importing Prerequisite Network Integrity Projects" for more information about required Network Integrity projects.

> **Note:**
>
> Project names must be unique in a workspace. You cannot define a single project with multiple target versions in one workspace. Set up multiple workspaces if you want to deploy cartridge projects to target environments with different versions. For example, set up a workspace for all Design Studio 7.3.0.0.0 cartridge projects, and import the required 7.3.0.0.0 model projects into that workspace. Set up a different workspace to contain 7.2.4.0.0 cartridge projects, and import the required 7.2.4.0.0 model projects into that workspace.

To define cartridge project target versions:

1. From the Studio Projects view, double-click a **Project** entity.

   The Project entity opens in the Project editor.
2. Click the **Projects** tab.
3. In the **Target Version** field, select the highest version number that is equal to or less than the version of the run-time software to which you want to deploy the cartridge project.

   For example, if you are deploying to release 7.3.0, select the highest version number that is equal to or less than version 7.3.0.

   When you select a new value in the **Target Version** field, Design Studio automatically initiates a new build. When changing an existing target version value, some entity configurations may no longer be valid for the new application version.

Related Topics

Project Editor Properties Tab

Managing Project Dependencies

Projects have dependencies on other projects when entities in one project reference entities in a different project. For example, an application project might reference data elements defined in a common model project.

If you configure a project to reference content in other projects without declaring project dependencies, Design Studio creates an error or a warning, depending on how you configured the diagnostic level. Design Studio filters data that appears in dialog boxes and views based on project dependencies.

To manage project dependencies:

1. From the Studio Projects view, double-click a Project entity to open the entity in the Project editor.
2. Click the **Dependency** tab.
3. Click **Add**.

   The Project Selection dialog box appears.
4. Select dependencies to add to the dependency list, and click **OK**.

To locate a specific project in a list of existing project dependencies, enter a partial or full project name in the search field.

5. From the project dependencies list, select a project and do the following:
   - Select a project and click **Move Up** or **Move Down** to change the order of the projects. The order establishes priority when upgrading and deploying projects with dependencies. The lowest project in the dependency hierarchy should appear first.
   - Limit the project dependency to a specific range of versions by defining the **Minimum Version** and **Maximum Version** for the selected project.
   - In the **Dependency Type** field, define whether a project dependency is required in the Design Studio workspace or in both the workspace and the run-time environment. This field is not available for some types of cartridge projects.
   - Click **Remove** to remove the selected from the dependency list.
6. In the **Dependency Violation Diagnostic Level** field, specify whether to generate a warning marker or an error marker when projects reference content in other projects but fail to declare a project dependency.
7. Click **Save**.

Related Topics

Project Editor Dependency Tab

Renaming Design Studio Projects

To rename a project:

1. From the **Window** menu, select **Open Perspective**, and then select **Java**.
2. Click the **Package Explorer** tab.

   The Package Explorer view appears.
3. Right-click a project, select **Refactor**, and then select **Rename**.

   The Rename Java Project dialog box appears.
4. In the **New Name** field, enter the new name for the project.
5. (Optional) To update entities that reference this project, select **Update references**.

   Typically, you will want to do this to avoid compile errors.
6. (Optional) To review the list of changes that will be made as a result of the name change, click **Preview**.
7. Click **OK**.

Related Topics

About Design Studio Naming Conventions

Sealing Projects

Design Studio projects can be sealed to prevent changes to the data. You might seal a project, for example, after the design is complete, debugged, and tested to prevent users who import the project from rebuilding or overwriting the original build artifacts. When a project is sealed, the entities in the project cannot be changed in any Design Studio editor. If you import a sealed project into a different workspace, the project in the target workspace remains sealed.

> **Note:**
>
> Design Studio generates Exchange Format XML files for a sealed project during the initial import if the sealed project directory does not contain any **.studioModel** XML files in the project **generated** folder. Design Studio does not update the **generated** folder XML files for sealed projects during subsequent builds.
>
> Before distributing sealed projects, Oracle recommends that you generate the project Exchange Format XML files to reduce initial build times when team members import the sealed projects.

All editors in a sealed project display **[Sealed]** in the title bar to indicate that the project is sealed and cannot be changed.

To seal a project:

1. In the Studio Projects view, double-click the Project entity for the project that you want to seal.

   The Project entity opens in the Project editor.
2. Click the **Properties** tab.
3. Click **Seal**.

   The confirmation dialog box appears.
4. Click **OK**.

Related Topics

Unsealing Projects

Working with Design Studio Cartridge Projects

Unsealing Projects

Design Studio projects can be sealed to prevent changes to the project data. If you intend to make changes to a sealed project, you must first unseal the project.

To unseal a project:

1. In the Studio Projects view, double-click a Project entity for the project that you want to unseal.

   The Project entity opens in the Project editor.
2. Click the **Properties** tab.
3. Click **Unseal**.

   > **Note:**
   >
   > Before unsealing a project, carefully consider the terms of the license agreement.
4. Click **OK**.

   Design Studio unseals the project and automatically initiates a build.

Related Topics

Sealing Projects

Working with Design Studio Cartridge Projects

Controlling Project Visibility in a Workspace

Design Studio solutions can contain large numbers of productized, sealed, and application-specific projects that are not directly related to your work. You control which projects appear in your workspace by creating and applying a filter, called a working set.

The projects that a working set filters out exist in the workspace and remain open but do not appear in the Studio Projects view. Also, you can define a separate working set for the Solution view to control visibility of project entities at the root level.

For example, you can facilitate design modeling and workspace navigation by creating a working set that displays only those projects related to your present design work. The working set can filter projects based on the project type and based on a tag that you associate with a project. Working set filters are limited to the workspace in which they are defined.

One working set is delivered with Design Studio and is applied to the Studio Projects view when you install Design Studio. This working set is named **Exclude Base Projects**, and it excludes from display all Design Studio base projects, any projects associated with the **Base Project** tag, and all non-Design Studio projects (such as Eclipse projects and Java projects). The **Exclude Base Projects** working set is not editable, but you can deactivate this working set filter. See "Deactivating a Working Set" for information about removing the filter that the **Exclude Base Projects** working set applies to the workspace.

Also, you can use standard Eclipse working set functionality to create a global working set that controls which menus, views, and toolbars appear in a workspace. You use these standard Eclipse working sets to control project visibility in the Package Explorer view. See the Eclipse *Workbench User Guide* for more information about creating and using window working sets.

> **Note:**
>
> You can control whether sealed projects appear in the Studio Projects view by toggling the **Exclude Sealed Projects** and the **Include Sealed Projects** icon, located in the Studio Projects view toolbar. See "Studio Projects View" for more information.

Controlling project visibility in a workspace involves the following tasks:

- Associating projects with a tag.

  You can associate projects with system-defined tags and with tags that you create, and filter the workspace to include or exclude projects associated with this tag. Also, you can create your own tags and include or exclude projects associated with these tags. See "Project Editor Tag Tab" and "Creating Tags" for more information. You cannot associate tags with Environment projects.
- Creating a Working Set
- Activating a Working Set
- Editing a Working Set

- Deactivating a Working Set
- Exporting Working Sets
- Importing Working Sets

Creating a Working Set

You can create a working set that defines a filter to control which projects appear in the Studio Projects view. You can also create a working set that controls visibility of entities of projects (at the root level) in the Solution view. You can create one or more working sets and apply them to a workspace.

> **Note:**
>
> Before you create a working set, you can associate projects with system-defined tags (such as the **Base Project** tag) and with tags that you create, and filter the workspace to include or exclude projects associated with this tag. Also, you can create your own tags. See "Project Editor Tag Tab" and "Creating Tags" for more information.

To create a working set:

1. From the Studio Projects or from the Solution view, click the **View Menu** icon in the view toolbar (the icon is an arrow).

   The context menu options appear.
2. From the context menu list, select the **Select Working Set** option.

   The Select Working Set dialog box appears.
3. Click **New**.

   The New Working Set dialog box appears.
4. Select **Design Studio** and then click **Next**.
5. In the **Working Set Name** field, enter a name to describe the working set.
6. Select a working set filter option.
   - Select **Include** to include specific projects in the workspace.
   - Select **Exclude** to exclude specific projects from the workspace.
7. In the **Project Type** area, select the project types that you want to include in the workspace or exclude from the workspace, depending on your selection in the previous step.
8. In the **Tag** area, select one or multiple tags to include or to exclude projects associated with these tags from the workspace, depending on your selection in the previous step.

> **Note:**
>
> If your working set includes selections in the **Project Type** area and in the **Tag** area, the working set filters only those projects that meet the criteria defined in both areas. For example, if the working set is defined to exclude OSM projects (selected in the **Project Type** area) and base projects (selected in the **Tag** area), the filter hides only those projects that are OSM projects associated with the **Base Project** tag.
>
> Tags do not apply to Environment projects.

9. Click **Finish**.
10. In the Select Working Set dialog box, select the new working set and click **OK**.

    Design Studio applies the working set and updates the Studio Projects view or the Solution view to display only those projects and entities that meet the filter criteria defined in the applied working set.

Related Topics

Activating a Working Set

Editing a Working Set

Controlling Project Visibility in a Workspace

Activating a Working Set

You activate the working set that you want to apply to the workspace.

To activate a working set:

1. From the Studio Projects view or from the Solution view, click the **View Menu** icon in the view toolbar.

   The context menu options appear.
2. From the context menu list, select the **Select Working Set** option.

   The Select Working Set dialog box appears.
3. Do one of the following:
   - Select **Window Working Sets** if you want to select a previously created global working set that controls project visibility in the Package Explorer view. See the Eclipse *Workbench User Guide* for more information about creating and using window working sets.
   - Select **No Working Sets** to remove any workspace filters defined by an active working set.
   - Select **Selected Working Sets** and then select a working set from the list to apply the filter defined by that working set to the workspace.
4. Click **OK**.

   Design Studio applies the working set and updates the Studio Projects view or the Solution view to display only those projects and entities that meet the filter criteria defined in the applied working set.

Related Topics

Creating a Working Set

Editing a Working Set

Deactivating a Working Set

Controlling Project Visibility in a Workspace

Editing a Working Set

You can edit an active working set (a working set whose filter is presently applied to the workspace) to change the projects that appear in your workspace.

> **Note:**
>
> Before you edit a working set, you can associate projects with system-defined tags (such as the **Base Project** tag) and with tags that you create, and filter the workspace to include or exclude projects associated with this tag. Also, you can create your own tags. See "Project Editor Tag Tab" and "Creating Tags" for more information.

To edit a working set:

1. From the Studio Projects view or from the Solution view, click the **View Menu** icon in the view toolbar.

   The context menu options appear.
2. From the context menu list, select the **Edit Active Working Set** option.

   The Edit Working Set dialog box appears.
3. In the **Working Set Name** field, edit the name that describes the working set.
4. Select the working set filter option.
   - Select **Include** to include specific projects in the workspace.
   - Select **Exclude** to exclude specific projects from the workspace.
5. In the **Project Type** area, select the project types that you want to include in the workspace or exclude from the workspace, depending on your selection in the previous step.
6. In the **Tag** area, select one or multiple tags to include or to exclude projects associated with these tags from the workspace, depending on your selection in the previous step.

> **Note:**
>
> If your working set includes selections in the **Project Type** area and in the **Tag** area, the working set filters only those projects that meet the criteria defined in both areas. For example, if the working set is defined to exclude OSM projects (selected in the **Project Type** area) and base projects (selected in the **Tag** area), the filter hides only those projects that are OSM projects associated with the **Base Project** tag.
>
> Tags do not apply to Environment projects.

7. Click **Finish**.

   Design Studio applies the working set and updates the Studio Projects view or the Solution view to display only those projects and entities that meet the filter criteria defined in the applied working set.

Related Topics

Creating a Working Set

Activating a Working Set

Deactivating a Working Set

Controlling Project Visibility in a Workspace

Deactivating a Working Set

You can deactivate a working set to remove the filter from the workspace. You can use this option only when there is an active working set applied to the workspace.

To deactivate a working set:

1. From the Studio Projects view or from the Solution view, click the **View Menu** icon in the view toolbar.

   The context menu options appear.

2. From the context menu list, select the **Deselect Working Set** option.

   Design Studio removes from the Studio Projects view or from the Solution view the filter defined by all previously active working sets.

Related Topics

Creating a Working Set

Activating a Working Set

Deactivating a Working Set

Controlling Project Visibility in a Workspace

Exporting Working Sets

You can export working sets to an external file if you want to use working sets across multiple workspaces.

To export working sets:

1. From the **File** menu, select **Export**.

   The Export dialog box appears.
2. Select **Oracle**, and then select **Design Studio Working Set**.
3. Click **Next**.
4. Select the working sets to export.
5. In **Select Destination**, specify the path and file name for the **.xml** file in which to save the working sets.

   If you export to an existing file, you will overwrite any content defined in the file.
6. Click **Finish**.

Importing Working Sets

You can import working sets from an external file if you want to use working sets across multiple workspaces.

1. From the **File** menu, select **Import**.

   The Import dialog box appears.
2. Expand the **Oracle Communications Design Studio Wizards** folder, and then select **Design Studio Working Set**.
3. Click **Next**.
4. In the **Select Source** field, specify the **.xml** file that contains the saved working sets.

   You must select a valid **.xml** file that contains working sets. The working sets that are saved in the file appear.
5. Select the working sets to import.

   Working sets that exist in the workspace are not available to select.
6. Click **Finish**.

Working with Model Projects

Model projects are collections of data elements that can be referenced by other projects in a workspace. Model projects include business entities and schema entities that are not specific to an Oracle Communications application and enable you to leverage common definitions and share that data across a solution.

When working with model projects, see "Creating Model Projects" for more information.

Creating Model Projects

You can create model projects to represent a collection of data elements within a workspace.

To create a model project:

1. From the **Studio** menu, select **New**, then select **Project** and then select **Model Project**.

   The New Studio Model Project dialog box appears.
2. In the **Project Name** field, enter a name for the project.

   Project names must be unique among Project entity types.

3. (Optional) Select a location for the project.

   By default, Design Studio saves the project to your default workspace location. To identify a location different from the default:

   a. Deselect **Use default location**.

   b. Click **Browse**.

   c. Navigate to the directory in which to save the project.

   d. Click **OK**.

4. In the **Execution Environment** field, specify the Java version to be used.

   The default value that appears is the execution environment that is specified in Preferences, Java, Installed JREs Preferences page. If no default value is defined on the Installed JREs Preferences page, Design Studio uses the execution environment of the primary Eclipse feature. You can select a different supported version from the list.

   When you create the project, Design Studio automatically configures the JRE System Library and the compiler compliance setting.

5. Click **Finish**.

   Design Studio adds the new model project to the Studio Projects view.

Related Topics

Working with Model Projects

Modeling Data

Working with Design Studio Cartridge Projects

In Design Studio, a cartridge is a collection of entities that you deploy to a run-time environment to support your business processes (for example, you deploy cartridges to Oracle Communications Order and Service Management run-time environments to support processes required to provision services requested on incoming sales orders). When modeling application-specific entities in Design Studio, you configure all entities within a project. That collection of entities is packaged into an archive file, which you can deploy to a run-time environment.

When working with Design Studio cartridges, see the following topics:

- Creating New Cartridge Projects
- Defining Project Version Numbers
- Working with Model Variables

Creating New Cartridge Projects

Cartridge projects are collections of entities that you deploy to a run-time environment to support your business processes.

To create a Cartridge project:

1. From the **Studio** menu, select **New**, then select **Project**, and then select the type of cartridge project to create.

   The New Project dialog box appears.

2. In the **Project Name** field, enter a name for the project.

Project names must be unique in the workspace. The cartridge and resultant archive file use the name that you enter here. Do not enter spaces or periods in this field. Oracle recommends a naming convention of lowercase letters separated by underscores. For example, **my_cartridge**.

3. (Optional) Select a location for the project.

   By default, Design Studio saves the project to your default workspace location. To identify a location different from the default:

   a. Deselect **Use default location**.

   b. Click **Browse**.

   c. Navigate to the directory in which to save the project.

   d. Click **OK**.

4. In the **Target Version** field, indicate the version of the run-time software to which you will deploy the cartridge.

   > **Note:**
   >
   > The value in the **Target Version** field is used by Design Studio to build your cartridge project to be compatible with the run-time version to which you want to deploy the project. Select the highest version number that is equal to or less than the version of the run-time software to which you want to deploy the project.

5. In the **Execution Environment** field, select an execution environment for the target version.

   Design Studio pre-populates this field (based on the selection in the **Target Version** field) with the recommended environment for the target version. You can select a different supported version from the list. When you create the project, Design Studio automatically configures the JRE System Library and the compiler compliance setting.

   Design Studio obtains the list of execution environments from the Eclipse workspace configuration. To view the list, from the **Windows** menu select **Preferences**, then expand **Java**, then expand **Installed JREs**, then select **Execution Environments**.

6. In the **Package Name** field, define the default implementation package name to be used as a prefix for generated code.

7. Click **Finish**.

   Design Studio adds the new Cartridge project to the Studio Projects view. In addition to containing the Project entity, the project may also include system-supplied entities, such as a Data Schema entity.

Related Topics

About Design Studio Naming Conventions

Working with Design Studio Cartridge Projects

Defining Project Version Numbers

After you create a project, you define the version number. You can change the version at a later time. When changing the version of deployable cartridge projects, clean and build the

project (and any dependent projects) before deploying the cartridge projects or the dependent cartridge projects.

To define a project version number:

1. In the Studio Projects view, double-click any Project entity.

   The entity opens in the Project editor.
2. Click the **Properties** tab.
3. Enter values for the **Major Version Number**, **Minor Version Number**, **Maintenance Pack**, **Generic Patch**, and **Customer Patch** fields.

   The combination of these version numbers (with the value in the **Build** field, which is automatically generated) represents the cartridge version number. Some cartridges support 3 digit version numbers (using **Major Version Number**, **Minor Version Number**, and **Maintenance Pack**). Version support is determined by the Design Studio feature and by the selected target version. See "Product Editor Properties Tab" for more information about these fields.
4. Select **File**, then select **Save**.

Related Topics

About Project Version Numbers

Working with Design Studio Cartridge Projects

About Project Version Numbers

Design Studio supports both a three-segment and a five-segment release version. You can define release version numbers for any unsealed and editable Design Studio project.

Projects must always have a valid version number. When you first create a project, Design Studio applies the following default values:

- Major Version Number: 1
- Minor Version Number: 0
- Maintenance Pack: 0
- Generic Patch: 0
- Customer Patch: 0

The build number is automatically generated through the build process.

When you edit any of these fields, you create a new version of the project.

> **Note:**
>
> Modifying these field values does not create a separate instance of the project in Design Studio, and Design Studio cannot support multiple versions of a project in the same workspace. Multiple versions of a project in the same workspace creates conflicting model entities.
>
> When changing version numbers, Oracle recommends that you use a source control system to ensure that you are able to return to the previous version.

Related Topics

Defining Project Version Numbers

Project Editor Properties Tab

Working with Model Variables

When you create projects, some of the information you provide may depend on a specific environment. If you have environment-specific values for variables that you will need at run time, you can create tokens for the variables and later define values for each environment in which you will use the variable. Tokens are placeholders for environment-specific values that can be defined at the time of deployment.

When modeling model variables, see the following topics:

- About Model Variables
- Creating Model Variables
- Defining Model Variables

About Model Variables

Model variables are placeholders for environment-specific values that you intend to define at the time of deployment. You can create, add, or remove model variables, as necessary.

For example, consider that you must define the credentials used for running automated tasks in two different environments—your testing environment and your production environment—and that the value required by the testing environment is different than that required by the production environment. Rather than editing the variable value in the source code each time you deploy to one of these environments, you can create a model variable, then define environment-specific values for that variable.

> **Note:**
>
> Some Oracle Communications features do not support model variables.

Related Topics

Creating Model Variables

Defining Model Variables

Project Editor Model Variables Tab

Creating Model Variables

You create model variables to represent environment-specific values that you intend to define at the time of deployment.

To create model variables:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Studio Projects** tab.

The Studio Projects view appears.

3. Double-click any Project entity.

   The project opens in the Project editor.

4. Click the **Model Variables** tab.

5. Click **Add**.

   Design Studio adds a new variable, **VAR_1**, to the Name column.

6. Select the table row that contains **VAR_1**.

7. In the **Name** field (below the table), replace **VAR_1** with a new name.

   For example, you might change the default variable name to **automationUser** if you were defining the credentials with which your automated tasks run.

8. (Optional) In the **Value** field, enter a default value for the variable.

   You can provide a default value for the variable if, for example, you have multiple environments and many share the same variable value.

9. (Optional) Select **Sensitive** to secure the default value in the user interface and on disk.

   This option enables you to define default values with a layer of security. If you deselect the **Sensitive** option, Design Studio clears the **Value** field to protect the existing value. See "Project Editor Model Variables Tab" for more information about this field.

10. (Optional) Click **Add** to create additional variables.

11. (Optional) Select a variable and click **Remove** to delete a variable.

> **Note:**
>
> Oracle recommends that you do not remove any of the default variables.

12. Click **Save**.

Related Topics

About Model Variables

Defining Model Variables

Project Editor Model Variables Tab

Defining Model Variables

You define model variables before you deploy a cartridge to a specific run-time environment.

To define model variables:

1. Create model variables.

   See "Creating Model Variables" for more information.

2. Populate a field with the model variable.

   When populating fields with model variables, use the following syntax:

   %{*VariableName***}**

where *VariableName* is the name of the model variable as defined on the Project editor **Model Variables** tab.

For example, Oracle recommends that you populate the **Run As** field on the Automation Plug-in Properties view **Details** tab with the default automation user variable. To use the delivered sample model variable (DEFAULT_AUTOMATION_USER), you populate the **Run As** field with the value **%{DEFAULT_AUTOMATION_USER}**.

3. Save, clean, and build the project.

   See "Running Clean Builds" for more information.

4. Select **Studio**, then select **Show Environment Perspective**.

5. In the **Environments** tab, select the environment to which you want to deploy the cartridge.

   See "Deploying Cartridge Projects from the Environment Perspective" for more information about deploying cartridges. See "Creating Run-Time Environments" for information about creating new environments.

6. Connect to the environment.

   See "Testing Run-Time Environment Connectivity" for more information about connecting to the environment.

7. Define environment-specific values for model variables.

   You define these values on the **Model Variables** tab. The system displays a list of all the model variables defined in the workspace (and any corresponding default values that you defined in the Project editor **Model Variables** tab. See "Studio Environment Editor Model Variables Tab" for more information.

   Environment-specific model variable values override those that are defined in the Project editor. If you define fields with model variables but you do not define the model variable with a default value (in the Project editor **Model Variables** tab) or with an environment-specific value (in the Studio Environment editor **Model Variables** tab), Design Studio creates a problem marker. You cannot deploy a cartridge until you resolve all problem markers.

Related Topics

About Model Variables

Creating Model Variables

Project Editor Model Variables Tab

Working with Environment Projects

Environment projects contain Environment entities. You work with Environment projects and entities in the Environment perspective, which enables you to manage the attributes associated with your run-time environments, including connection attributes, projects ready to be deployed, projects previously deployed, and associated project attributes such as the version and build numbers.

When working with Environment projects, see "Creating Environment Projects" for more information.

Creating Environment Projects

You create environment projects to contain the attributes associated with your environment, including connection attributes, projects to be deployed and previously deployed, and associated project attributes.

To create environment projects:

1. From the **Studio** menu, select **New**, then select **Project**, and then select **Environment Project**.

   The New Studio Environment Project dialog box appears.

2. In the **Name** field, enter a name for the environment.

   The name must be unique among environment entities in the same namespace.

3. (Optional) Select a location for the project.

   By default, Design Studio saves the project to your default workspace location. To identify a location different from the default:

   a. Deselect **Use default location**.

   b. Click **Browse**.

   c. Navigate to the directory in which to save the project.

   d. Click **OK**.

4. Select an execution environment.

   Specify the Java version that you are using. The preferred java version appears by default. The default value that appears is the execution environment that is specified in Preferences, Java, Installed JREs preferences page. If no default value is defined on the Installed JREs preferences page, Design Studio uses the execution environment of the primary Eclipse feature.You can select a different supported version from the list.

   When you create the project, Design Studio automatically configures the JRE System Library and the compiler compliance setting.

5. Click **Finish**.

   Design Studio saves the changes and adds a new environment project to the Studio Projects view.

Related Topics

Working with Environment Projects

Project Editor

Use the Project editor to define project configuration details. To access the Project editor, double-click any Project entity in the Studio Projects view to display the entity in the Project editor. The tabs in the Project editor depend on the type of project.

When using the Project editor to configure projects, see the following topics:

- Project Editor Properties Tab
- Project Editor Copyright Tab
- Project Editor Dependency Tab

- Project Editor Tag Tab
- Project Editor Packaging Tab
- Project Editor Model Variables Tab
- Project Editor Cartridge Management Variables Tab

Project Editor Properties Tab

Use the **Properties** tab to define the build, version, and target properties of the project.

Field	Use
Description	Enter the name for the project as it should appear in Design Studio and in the run-time environment.
Provider	Enter a name or description of the project to help identify the project within the Design Studio environment. If you have purchased a Cartridge project from a third party, this field may contain the name of the third-party provider.
Identifier	Enter a unique string to identify the project. This is a fully qualified dot separated name. Typically, this value includes segments indicating provider and technology.
Package Name	Enter a default implementation package name to use as a prefix for generated code.
Major Version Number, **Minor Version Number**, **Maintenance Pack**, **Generic Patch**, **Customer Patch**	Define values to create a five-segment release version. You can define release version numbers for any unsealed and deployable Design Studio project. Cartridge projects must always have a valid version number. When you first create a Cartridge project, Design Studio applies the following default values: • Major Version Number: 1 • Minor Version Number: 0 • Maintenance Pack: 0 • Generic Patch: 0 • Customer Patch: 0 You can edit these fields to uniquely identify the cartridge versions. **Note:** Modifying these field values does not create a separate instance of the project in Design Studio. When changing version numbers, Oracle recommends that you use a source control system to ensure that you are able to return to the previous version. **Important:** Design Studio cannot support multiple versions of a project in the same workspace. Multiple versions of a project in the same workspace creates conflicting model entities.
Build Number	Indicates which version of the metadata is used by the corresponding project. If you have enabled the automatic build feature, Design Studio increases the build number automatically every time you save. To enable the automatic build feature, select **Project**, then **Build Automatically**.

Field	Use
Target Version	Select the version of the run-time application instance to which this Cartridge project will be deployed. Design Studio builds your project to be compatible with the run-time software version you specify here. Select the highest version number that is equal to or less than the version of the run-time software to which you want to deploy the project. For example, if you are deploying to release 7.2.0, select the highest version number that is equal to or less than version 7.2.0. For OSM projects, the target version that you specify must match the version of the installed SDK that you specify in the **OSM SDK Home** field on the Order and Service Management Preferences page. See "Defining Order and Service Management Preferences" for more information. **Note:** When you select a new value in the **Target Version** field, Design Studio automatically initiates a new build. Some entity configurations may no longer be valid for the new application version. This field appears only for Cartridge projects that are deployable to run-time environments. See "Defining Cartridge Project Target Versions" for more information.
State	Click **Unsealed** if you want to make changes to the project, then rebuild it to obtain a new archive file. **Note:** To modify files defined as read-only, you must edit the entity read-write properties. See "Defining Entity Read-Only Properties" for more information. Click **Seal** to prevent changes to the project. You might seal a project, for example, after the project design is completed, debugged, and tested to prevent other users from rebuilding or overwriting the original build artifacts.
Namespace	Enter the name of the namespace in which the project exists. Some Design Studio features do not support this field.
Standalone	(Used in OSM only) Indicates whether the cartridge is to be included in the composite cartridge as part of the solution or is to be used as a standalone cartridge with no project-level or entity-level dependencies. This field is not applicable to composite cartridges.
Common Model Entity Container	(Used in OSM only) Specify the Model project in which you want to save the associated conceptual model entities, for example, actions, resources, data structure definitions, and so on.

Related Topics

Defining Project Version Numbers

About Project Version Numbers

Project Editor

Project Editor Copyright Tab

Use the **Copyright** tab to review or edit copyright and license information and to edit the message that appears when users seal and unseal projects.

Field	Use
Description	Enter the name for the project as it should appear in Design Studio and in run-time environments.
Copyright Information	For third-party projects that you have purchased, displays the copyright information. If you are developing your own projects, enter the project copyright information. Copyright information is included in the project archive (in **cartridgeBin**) as **copyright.txt**.
License Information	For third-party projects that you have purchased, you can review the license agreement information. If you are developing your own projects, enter the project license information. License information is included in the project archive (in **cartridgeBin**) as **license.txt**.
Seal Message and **Unseal Message**	Do one of the following to define the message that appears when users click **Seal** and **Unseal**: • Select **Default** to display the default message that appears. • Select **Custom** to change the default message for specific cartridge versions. See "Project Editor Properties Tab" for more information about sealing and unsealing projects.

Related Topics

Project Editor

Project Editor Dependency Tab

Use the **Dependency** tab to manage all project dependencies. Projects have dependencies when they reference entities or data elements defined in other projects. For example, if project A references an element or entity from project B, then project A has a dependency on project B.

The project dependencies that you define here control the selections that are available when you model projects. Design Studio restricts these selections based on the configured dependencies. Oracle recommends that you plan relationships between projects and configure project dependencies early in your development cycle.

Design Studio saves project dependency information in the **MANIFEST.MF** file, and uses this information to validate dependencies at deployment.

> **Note:**
>
> You must declare all project dependencies on the **Dependency** tab. Design Studio creates a problem marker if you reference entities in projects that are not defined as dependencies on this tab.

Field	Use
Search	Enter text to search for a specific project dependency in the project list, which displays an ordered list of project dependencies.

Field	Use
Move Up	Click to move the selected project up in the order of dependencies. The order of the dependencies determines the processing sequence during any upgrade process.
Move Down	Click to move the selected project down in the order of dependencies. The order of the dependencies determines the processing sequence during any upgrade process.
Remove	Click to remove the selected project from the list.
Add	Click to add dependent projects to the list.
Project Location	Displays the workspace location of the selected project. This field is read-only.
Project Name	Displays the name of the selected project. This field is read-only.
Minimum Version and **Maximum Version**	Specify a range of versions for the selected dependent project. During project upgrades, Design Studio searches for dependencies that fall within the specific range. You can refine the criteria by selecting to include the instances of that version (inclusive) or to exclude instances of that version (exclusive). You can define only a minimum version to indicate that all later versions are valid, or define only a maximum version to indicate that all previous versions are valid. Oracle recommends that you define specific ranges, when possible.
Dependency Type	Define whether a project dependency is required in the Design Studio workspace or in both the workspace and the run-time environment. Select: • **Design** to indicate that the project dependency is required in the Design Studio workspace only. • **Runtime** to indicate that the project dependency is required in the Design Studio workspace and in the target run-time environment. The default option is **Design** for all cartridge projects developed in previous Design Studio versions. Dependencies to non-deployable projects (for example, to Model projects) are always defined as **Design** and cannot be changed.

Field	Use
Dependency Violation Diagnostic Level	Do one of the following: • Select **Warning** to generate a warning marker when projects reference content in other projects but fail to declare a project dependency. This is the default value for all upgraded projects. • Select **Error** to generate an error marker when projects reference content in other projects but fail to declare a project dependency. This is the default value for all new cartridge projects. You cannot deploy a cartridge when error markers exist. The default diagnostic level for Activation SRT projects is **Warning**. The default diagnostic level for all other project types is **Error**.

Related Topics

Project Editor

Managing Project Dependencies

Project Editor Tag Tab

Use the **Tag** tab to associate projects with a predefined tag, called the **Base Project** tag, or with tags that you create. You can filter your workspace to include or exclude projects associated with these tags. See "Controlling Project Visibility in a Workspace" for more information.

Field	Use
Add	Click to associate the project with a tag.
Remove	Select a tag in the table and click to remove the association from the project.

Related Topics

Project Editor

Project Editor Packaging Tab

Use the **Packaging** tab to specify which entities will be deployed to the run-time environment.

> **Note:**
>
> Some Oracle Communications features automatically manage the content of the deployment archive. These features do not include the **Packaging** tab in their Project editor.

Field	Use
Packaging Instructions	Select which entities you want to include for each of these categories.
Include all from Project	Select this option if you want to include all entities from a specific resource. For example, if you want to include all of your Java libraries, select **Java Libraries** from the left-side column, then select **Include all from Project**. The system will include all libraries in the package file.

Related Topics

Project Editor

Project Editor Model Variables Tab

Use the **Model Variables** tab to create and define variables that you can use when you require different values for the same field, depending on the environment.

Some applications require specific model variables. See "About OSM Model Variables" for more information.

Model Variables Section

Field	Use
Add	Click to add a new variable to the Model Variables table. Design Studio adds the value **VAR_1** to the Name column. This table is read-only. To change the value, select the table row that contains the new variable, and enter a new value in the **Name** field.
Remove	Select a row in the Model Variables table and click **Remove** to delete the variable.

Model Variable Details Section

Field	Use
Name	Displays the name of the variable in the selected table row.
Sensitive	Select to secure the variable defined in the selected table row and to hide the default value from users. When you select this option, the default value defined for the cartridge model variable is obfuscated in the user interface and on disk. By default, cartridge model variables are not sensitive. **Important:** If you deselect the **Sensitive** option, Design Studio clears the **Value** field to protect the existing value.
Value	Displays the default value for the variable in the selected table row. You can provide a default value for variables when multiple environments share the same variable value. **Note:** When defining default values for variables, employ the same default value for a variable across all projects in a workspace. If a variable defined in multiple projects does not share the same variable value, a warning appears in the Problems view.

Related Topics

Project Editor

Working with Model Variables

About OSM Model Variables

In OSM, you can set model variables in both composite and component cartridges.

> **Note:**
>
> Ensure that you define any model variables that are common to both a composite cartridge and its component cartridges so that they have the same value. If you define the same cartridge management variable with different values on both a composite cartridge and its component cartridge, the value that will be used in OSM depends on how they are deployed. If you deploy them as a single composite cartridge, the variable is derived from the composite cartridge, but if you then deploy the component cartridge alone, the variable defined on the component cartridge are used instead. This can cause unexpected changes to cartridge behavior.

Design Studio for OSM provides the following default model variables:

Default Variable	Use
DEFAULT_AUTOMATION_USER	Use this model variable to represent the security principal whose credentials are used to run the automation plug-ins. By default, the value of this variable is **oms-automation**.
DEFAULT_REQUEST_QUEUE	Use this model variable to identify the JMS destination to which automation plug-ins will send messages by default. By default, this variable has no value.
DEFAULT_RESPONSE_QUEUE	Use this model variable to identify the JMS destination from which automation plug-ins will receive messages by default. By default, the value of this variable is **mslv/oms/oms1/internal/jms/events**.
DEFAULT_MESSAGE_PROPERTY_SELECTOR	Use this model variable to identify the messages for which automation plug-ins listen. By default, this variable has no value.
ACTIVATION_ENVIRONMENT_ID	Use this model variable to represent the Oracle Communications ASAP environment ID to which service action requests are sent. By default, the value of this variable is **ENV1**.

Related Topics

Working with Model Variables

Project Editor Model Variables Tab

Project Editor Cartridge Management Variables Tab

Use the **Cartridge Management Variables** tab to define attributes of the project behavior after you deploy to the target environment.

> **Note:**
>
> When defining default values for variables, employ the same default value for a variable across all projects in a workspace. If a variable defined in multiple projects does not share the same variable value, a warning appears in the Problems view.

Some applications require specific cartridge management variables. For more information, see:

- About OSM Cartridge Management Variables
- About Network Integrity Cartridge Management Variables

Field	Use
Add	Click to add a new variable to the Cartridge Management Variables table. Design Studio adds the value **VAR_1** to the Name column. This table is read-only. To change the value, select the table row that contains the new variable, and enter a new value in the **Name** field.
Remove	Select a row in the Cartridge Management Variables table and click **Remove** to delete the variable.
Name	Displays the name of the variable in the selected table row.
Sensitive	Select to secure the variable defined in the selected table row and to hide the default value from users. When you select this option, the default value defined for the cartridge management variable is obfuscated in the user interface and in memory. By default, cartridge management variables are not sensitive. **Important:** If you deselect the **Sensitive** option, Design Studio clears the **Value** field to protect the existing value.
Value	Displays the default value for the variable in the selected table row. You can provide a default value for variables when multiple environments share the same variable value. **Note:** When defining default values for variables, use the same default value for a variable across all projects in a workspace. If a variable defined in multiple projects does not share the same variable value, a warning appears in the Problems view.

> **Note:**
>
> Oracle recommends that you do not remove any of the default cartridge management variables.

Related Topics

Project Editor

About OSM Cartridge Management Variables

Design Studio for OSM provides the following default cartridge management variables:

> **Note:**
>
> Use the same default value for variables that are defined in multiple cartridges in a workspace. If different values are used, a warning appears in the Problems view.

Default Variable	Use
DEFAULT_CARTRIDGE	This variable indicates whether the current version of the cartridge is the default version in the namespace. Cartridges that you deploy to OSM must have a default version. Specifying a default cartridge ensures that orders submitted by Task web client users who are not defined with permission to create versioned orders are handled correctly in the OSM run-time environment. At run time, if a user who does not have permission to create versioned orders enters an order, the OSM server submits the order to the default cartridge. See "Role Editor Role Tab" for more information about granting permissions to roles. Set this variable to one of the following values: • Enter **true** to indicate that this version of the cartridge is the default version in the namespace. This is the default value. • Enter **false** if there are multiple versions of this cartridge deployed to the OSM environment and one of the deployed versions has this set to **true**. **Note**: If you have multiple versions of a composite cartridge, for each composite cartridge you should set this variable to the same value for the composite cartridge and all of the component cartridges that it references. Only one of the composite cartridges should have this variable set to **true**.
ENTITY_CONFLICT_ACTION_ON_DEPLOY	This variable determines system behavior when cartridges are being updated with a new version and conflicting entities are found. It applies only if PURGE_CARTRIDGE_BEFORE_DEPLOY is set to **false**. Set this variable to one of the following values: • Enter **replace** to replace the old entities with the new. This is the default value. • Enter **ignore** to add the new entities and retain the old. • Enter **abort** to stop the deployment process.
FAST_CARTRIDGE_UNDEPLOY	This variable determines system behavior when undeploying cartridges. Set this variable to one of the following: • Enter **true** to undeploy a cartridge from OSM without purging cartridge metadata or order data. This is the default value. • Enter **false** to purge cartridge metadata and order data during the undeploy operation.
NODE_REMOVAL_ALLOWED_ON_REDEPLOY	This variable determines whether to allow users to remove a node from an order template and redeploy cartridges. Set this variable to one of the following: • Enter **true** to enable users to remove nodes from order template and redeploy cartridges. • Enter **false** to prevent users from removing nodes from order template and redeploy cartridges. This is the default value.

Default Variable	Use
PURGE_CARTRIDGE_BEFORE_DEPLOY	This variable determines system behavior when you are deploying a new version of a cartridge to an OSM environment where an older version of the cartridge is already deployed. Set this variable to one of the following values: • Enter **true** to undeploy the previous version of a cartridge before deploying the new version. Do not use this value if your cartridge has pending or completed orders that you do not want to purge. Instead, do not undeploy the cartridge and set this value to **false**. If both PURGE_CARTRIDGE_BEFORE_DEPLOY and FAST_CARTRIDGE_UNDEPLOY are set to **true**, the cartridge is undeployed using the fast undeploy option before it is redeployed. This is referred to as a fast redeploy. • Enter **false** to update the previous version of the cartridge with the changes in the new version. This is the default value.
PURGE_ORDER_ON_UNDEPLOY	This variable determines system behavior when you try to undeploy cartridges that have pending orders. Set this variable to one of the following values: • Enter **true** to purge all existing orders associated with the cartridge. This is the default value. If both PURGE_ORDER_ON_UNDEPLOY and FAST_CARTRIDGE_UNDEPLOY are set to **true**, the operation is a forced fast undeploy, which stops open orders; neither the cartridge nor associated orders are purged. If PURGE_ORDER_ON_UNDEPLOY is set to **true** and FAST_CARTRIDGE_UNDEPLOY is set to **false**, the operation is a forced undeploy, which purges the cartridge and associated orders. • Enter **false** if you do not want the system to undeploy the cartridge if it has pending orders. If you attempt to undeploy a cartridge that is not present in your workspace, and if the PURGE_ORDER_ON_UNDEPLOY option is defined as **Inherit** on the **Cartridge Management Variables** tab of the Studio Environment editor, then Design Studio defines the PURGE_ORDER_ON_UNDEPLOY option as **false**, even if the deployed cartridge includes a PURGE_ORDER_ON_UNDEPLOY option that is defined as **true** (on the Order and Service Management Project editor **Cartridge Management Variables** tab). To change this behavior, select the PURGE_ORDER_ON_UNDEPLOY option in the **Cartridge Management Variables** tab of the Studio Environment editor, then select **Override**, and then define the option as **true**. **Caution:** Undeploying a cartridge using the forced undeploy option purges all existing orders for that cartridge.
SUSPEND_DB_JOBS_TIMEOUT	This variable determines the amount of time the database jobs remain in a suspended state. Before importing a cartridge, the OSM server ensures that the OSM database jobs have stopped. If a variable is not specified, the default value of 0 is used, which indicates infinite time.

Default Variable	Use
UNDEPLOY_SHARED_CARTRIDGE	(Composite cartridges only.) This variable determines system behavior when undeploying a composite cartridge. See "Undeploying Composite Cartridges with Shared Component Cartridges" for more information. Set this variable to one of the following values: • Enter **true** to force the undeployment of shared component cartridges within a composite cartridge. • Enter **false** to prevent the undeployment of cartridges that are shared. This is the default value.
OSM_RUNTIME_TYPE	This variable indicates whether the cartridge was configured with 7.5.0 features such as, TMF cartridge, System Interactions, Fallout and so on. By default, the variable is set to the value `WLS`. You can set the variable to one of the following values: • For Non-TMF cartridges, you can use `WLS`. This means that cartridges can be deployed both in cloud native and WebLogic Server environments. • For TMF cartridges, the value of the variable **OSM_RUNTIME_TYPE** is `MultiService`. This indicates that the cartridge configuration requires a cloud native environment. Here OSM would be running in an integrated environment with cloud native capabilities. WLS indicates OSM is running as a standalone application. Cloud native indicates that OSM is running in an integrated environment with cloud native capabilities.

> **Note:**
>
> Oracle recommends that you do not remove any of the default cartridge management variables.

You can define and use the following optional cartridge management variables:

Optional Variable	Use
BUILD_DEPLOY_MODE	This variable determines how the system will build and deploy cartridges targeted at versions of OSM prior to 7.3. **Do not use this variable for OSM 7.3 or later OSM servers.** Set the variable to one of the following values. • Enter **optimized** if you want to build the automation components for automation plug-ins to be able to run within a common .ear file (**oms.ear**) at run time. This is the value that will be used if this variable is not set. • Enter **legacy** if you want to build the automation components for each automation plug-in to run in its own .ear file at run time. • Enter **both** if you want to build the automation components for automation plug-ins to run in Optimized and Legacy modes. If you select **both**, OSM processes the automation plug-ins according to the automation plug-in dispatch mode that is set on the OSM server. See "Defining Order and Service Management General Preferences" for information about automation plug-in build-and-deploy modes. **Tip:** How you build and deploy automation plug-ins (the build-and-deploy mode) indicates whether you want the plug-ins to be able to run within the **oms.ear** file or not. The automation plug-in dispatch mode set on the OSM server indicates the ability for OSM to invoke an automation plug-in that is running within the bounds of the **oms.ear** file.
XML_CATALOG_SUPPORT	This variable determines whether XML Catalog support is enabled. It is enabled by default and is required to be enabled. When you deploy to some target run-time software versions, you can disable XML Catalog support. See "Working with XML Catalogs" for more information on using XML Catalogs. • Enter **enable** if you want to enable XML Catalog support for this cartridge. This is the value that will be used if this variable is not set. • Enter **disable** if you want to disable XML Catalog support for this cartridge.

Related Topics

Project Editor Cartridge Management Variables Tab

About Network Integrity Cartridge Management Variables

Design Studio for Network Integrity provides the following cartridge management variables:

Value	Description
wladmin.host.name	Enter the host name of the server where the WebLogic Server resides.
wladmin.host.port	Enter the WebLogic Server port number.
wladmin.server.name	Enter the WebLogic Server Administration Server name.

Related Topics

Project Editor Cartridge Management Variables Tab

3

Working with the Design Studio User Interface

The Design Studio interface includes a number of components to assist you with configuration. Interface components include workspaces, perspectives, views, and editors.

When working with the Design Studio user interface, see:

- Working with Workspaces
- Working with Perspectives
- Working with Views
- Working with Design Studio Menus
- Working with the Design Studio Toolbar
- Working with Editors

Working with Workspaces

Workspaces are directories on your local machine that contain all of your Design Studio projects and entities.

When working with workspaces, see:

- About Workspaces
- Defining Workspace Preferences
- Switching Workspaces

About Workspaces

When you first open Design Studio, you identify a workspace to use in the Workspace Launcher dialog box. If the workspace that you specify does not exist, Design Studio creates it on your local machine. Always use Design Studio to create your workspaces. Do not, for example, copy workspaces between Design Studio installations. You can use more than one workspace, but you can have only one workspace open at a time.

To ensure that you do not adversely impact Design Studio performance, do not select a network location as your workspace and do not share workspaces and projects by referencing shared network file locations. When working in a multiuser environment, use a recommended source control system. See the *Design Studio Developer's Guide* for more information about working with source control.

Do not distribute workspaces using archive files. For example, if you need to distribute projects among team members, do not create an archive file that contains a workspace and the set of projects for distribution. If you receive a workspace contained in an archive file from a team member, unzip the file and import the projects using the **Import Studio Projects** menu action.

> **Note:**
>
> After you install Oracle Enterprise Pack for Eclipse 12.1.1.2, the first time that you open Design Studio (and every time you create a new workspace), the system displays a message if it cannot locate the Git source control installation directory. If your team does not use Git as a source control, select **Do not warn again if Git cannot be found** and click **OK**. When the system displays a second prompt, select **Do not show again** and click **OK**.
>
> If your team does use GIT as your source control, ensure that you define the source control settings as appropriate to your environment. See the *Design Studio Developer's Guide* for more information about working with source control.

Defining Workspace Preferences

You can define certain behavior preferences of your workspace, such as whether to build, refresh, or save automatically.

To define workspace preferences:

1. From the **Window** menu, select **Preferences**.

   The Preferences dialog box appears.
2. Expand the **General** folder.
3. Click **Workspace**.

   The Workspace preferences page appears. For more information about this page, see the Eclipse Help.
4. Click **OK**.

Related Topics

Switching Workspaces

Working with Workspaces

Switching Workspaces

To improve Design Studio performance during modeling and import, save your data in multiple workspaces, and switch workspaces when necessary.

To switch workspaces:

1. Save any changes to your present workspace.
2. From the **File** menu, select **Switch Workspace**, then select **Other**.

   The Workspace Launcher dialog box appears.
3. In the **Workspace** field, select the workspace to which you want to switch.
4. In the **Copy Settings** field, determine whether you want to copy either of the workbench settings.

   Do one of the following:

   - Select **Workbench Layout** if you have customized your perspective and want to retain the layout in the new workspace.

- Select **Working Sets** if you are filtering resources and would like to include the same selected resources in the new workspace.

See the Eclipse Help for more information about the **Copy Settings** options.

5. Click **OK**.

   Design Studio closes, and it restarts using the new workspace.

Related Topics

Defining Workspace Preferences

Working with Workspaces

Working with Perspectives

Perspectives determine how information appears in the workbench, in menus, and in toolbars. Perspectives are task oriented. For example, you use the Design perspective to model the entities in your cartridges. You use the Environment perspective to create and manage the attributes associated with your environment and to deploy and undeploy cartridges. Each perspective contains a default layout and a set of views which you can customize.

When working with perspectives, see:

- About Design Studio Perspectives
- Switching Perspectives

About Design Studio Perspectives

Perspectives define your Workbench layout and provide different functionality for working with different types of resources. The following are the most common perspectives in Design Studio:

- The Design perspective is a layout containing a collection of views that you use for cartridge modeling and configuration.

> **Note:**
>
> The Design perspective default layout is best viewed using a wide screen monitor format. Oracle recommends that you use a monitor with a 16:9 aspect ratio.

- The Environment perspective is a layout containing a collection of views that enable you to create and manage the attributes associated with your environment. Use this perspective to deploy or undeploy cartridges to an environment and to control and manage all of your environments.
- The Java perspective is a layout containing a collection of views that contain project folders and files that are generated in builds.

The Design Studio perspectives work together with perspectives that are not native to Design Studio but are commonly used for tasks such as implementation, debugging and version control (for example, the perspectives for Debug, Resource, and so forth).

Related Topics

Switching Perspectives

Working with Perspectives

Switching Perspectives

You can switch perspectives to display different sets of default views and editors.

To switch perspectives:

1. From the **Window** menu, select **Open Perspective**.

   A list of recent selections appears at the top of the menu.
2. Select **Other**.

   The Open Perspective dialog box appears.
3. Select a perspective.
4. Click **OK**.

Working with Views

Views enable you to customize the manner in which information is presented in the Workbench. When working with views, see the following topics:

- About Views
- About Fast Views
- Opening Views
- Minimizing and Maximizing Views
- Studio Projects View
- Data Elements View
- Dictionary View
- Notes View
- Outline View
- Overview View
- Problems View
- Relation Graph General View
- Solution View
- Studio Projects View
- Structure View

About Views

Views provide access to specific sets of functions, available through the toolbars and context menus. For example, the Problems view displays errors that exist in the model entities, so you use the Problems view to locate and resolve entity errors. You use the Dictionary view to model

and review data in your workspace. The Dictionary view and the Problems view each provide access to a different set of Design Studio functions.

A view can appear by itself or it can be stacked with other views. Additionally, you can undock a view from the workbench. You can change the layout of a perspective by opening and closing views and by docking them in different positions in the workbench. Within a given perspective, views further define your Workbench layout and provide different presentations of resources.

The most common views in the Java perspective are the Problems, Outline, and Package Explorer views. The most common views in the Studio Design perspective are the Solution, Studio Projects, Dictionary, and Data Elements views. The most common views in the Environment perspective are the Cartridge Management view, Console view, and Environment view.

Many views are available that are not native to Design Studio but are required for a variety of tasks, including implementation and builds of cartridges. These include various General views, Team views, Debug views, Java views, and others.

Related Topics

Working with Views

About Fast Views

You can right-click on the title bar of a view and select **Fast View** to create space on your workbench while keeping views minimized but easily accessible. When you select **Fast View**, the corresponding view temporarily displays on top of the other views. The fast view minimizes again when you click outside of it. Additionally, you can create a fast view by dragging a view onto the Fast View bar in the bottom left corner.

Related Topics

Working with Views

Opening Views

Design Studio perspectives have default combinations of views and editors. You can open a Design Studio view that is not included in the current perspective, minimize views to increase the working space of your monitor, and restore the view to the original size.

To open a view:

1. From the **Window** menu, select **Show View**.

   A list of commonly used views appears.
2. Do one of the following:
   - If the view you want to open appears in the list, select it.
   - If the view that you want to open does not appear in the list, select **Other** In the Show View dialog box, expand the appropriate directory, and select the view.
3. Click **OK**.

   Design Studio adds the view to the Workbench.
4. (Optional) Click the **Minimize View** button for a view to minimize the view and increase the amount of viewable space.

   An icon representing the view will appear in the status bar of the view area. Click the **Restore** button to restore the view to its normal size.

Related Topics

Working with Views

Minimizing and Maximizing Views

You can minimize and maximize views. When minimized, the view (or view stack) is collapsed to a series of icons on the workbench frame. Minimizing views enables you to retain the collection of views open in a perspective while temporarily creating more space for an editor. You can click the **Restore** icon to restore the views to the initial size.

To maximize views, double-click the view's title bar. The view fills the entire Workbench. Double-click the title bar a second time to return to the original size.

Related Topics

Working with Views

Data Elements View

Use the Data Elements view to view the simple and structured data elements for a selected entity and to perform operations on those data elements.

The Data Elements view is linked to the Solution view, the Studio Projects view, and the Package Explorer view. When you make a selection in any of these views, Design Studio updates the content in the Data Elements view based on the active selection. The Data Elements view enables you to view all data elements associated with the selected entity and to perform operations on the data elements, without needing to open the related editor.

For example, when a Data Schema entity is selected in the Solutions view, you can right-click in the Data Elements view to add data elements to the selected data schema. If a Product specification is selected in the Studio Projects view, you can view the data elements defined on the product in the Data Elements view, and perform refactoring operations. See "Refactoring Entities and Data Elements" for more information about the operations that you can perform on data elements in this view.

Related Topics

Solution View

Dictionary View

Studio Projects View

Working with Views

Dictionary View

Use the Dictionary view to display all data elements for every entity that contributes to the Data Dictionary. The root element is read-only.

You can use this view to create and manage data elements. For example, you can use this view to add elements to data schemas or to refactor data elements.

Additionally, you can use this view to model entities by dragging data from this view into an entity editor. See "Dragging Elements from the Dictionary View" for more information. Also, you can double-click the data element to open the element in the source entity editor.

Action	Location	Use
Show Only Root Simple Element	Toolbar	Click to display all root level simple elements. Disable to display root level simple and root level structured data elements.
Filters elements inherited through entity extension	Toolbar	Click to display only data elements that inherit from a base type. Inherited elements (structured or simple) are defined using a base type as the data type.
Filters elements not visible to the active editor based on project dependencies	Toolbar	Click to display only those elements available to the active editor, based on the project dependencies. See "Project Editor Dependency Tab" for information about defining project dependencies.
Clear Filter	Toolbar	Click to remove all previously defined filters: all data elements in the workspace appear in the view.
Filter Menu	Toolbar	Click to access two sets of view filters: the Tags filter and the Entity Type filter. In the Tags area, select: • **Any** to display only those data elements that have associated tags defined. • **None** to display only those data elements that have no associated tags defined. • A specific tag definition to display only those data elements defined with that tag. When multiple tags are selected, data elements defined by one or more of the selected tags appear in the view. In the Entity Type area, select: • **All** to display data elements associated with any entity type. • A specific entity type to display only data elements associated with the selected entity. When multiple entity types are selected, data elements defined by any of the selected types appear in the view.

Action	Location	Use
Element **Entity Name**	Toolbar	Search for specific data elements, such as a specific address or person, or for specific entities. Enter terms into these fields and click the **Filter** button to search for the term. Click the **Filter** button again to clear the search term. When filtering, consider the following: • Separate terms by a space to filter for both terms. Add **OR** between terms to search for either one or the other. For example, entering **32 OR aa** returns occurrences of terms starting with **32** and **aa**. • Searches return all elements and entities that begin with the search string. • Use an asterisk followed by a term to match the endings of terms. • Use an asterisk to match any number of characters. • Use asterisks on both sides of a term to return strings that contain a match anywhere in the string. • Use a question mark to match single characters. • Variations of strings are returned. For example, a search for ***ID** returns **productID**, **customerID**, and **orderID**. • Use a hyphen (-) before a term to omit specific variations of words (ensure there is a space before the hyphen). For example, a search for ***ID -product** returns **customerID** and **orderID**, but not **productID**.
Add Simple Schema Element	context menu	Right-click a Schema entity to add a new global simple schema element (an element that has no associated parent structure) to a data schema. A simple schema element cannot contain any child elements. See "Creating Simple Data Elements" for more information.
Add Structured Schema Element	context menu	Right-click a Schema entity to add a new global structure (a structure that has no associated parent structure) to a data schema. Structures are complex data types that include embedded data types. See "Creating Structured Data Elements" for more information.
Add Structured Child Schema Element	context menu	Right-click a structured data element to add a new child structure (a structure that is included in a parent structure) to a data schema. Structures are complex data types that include embedded data types. See "Creating Structured Data Elements" for more information.
Add Simple Child Schema Element	context menu	Right-click a structured data element to add a new child schema element (an element that is included in a parent structure) to a data schema. See "Creating Simple Data Elements" for more information.
Delete	context menu	Select to permanently remove the schema element from a data schema. This action is not available for root level elements (those that represent the entity name).

Action	Location	Use
Move Up	context menu	Select to move a data element (and the children, if applicable) to a higher position within the set of child data elements. You can move data elements when: • The entity that contains the element is writable. See "Defining Entity Read-Only Properties" for more information. • The entity that contains the element exists in a project that is unsealed. See "Unsealing Projects" for more information. • The element is not at the root level.
Move Down	context menu	Select to move a data element (and the children, if applicable) to a lower position within the set of child data elements. You can move data elements when: • The entity that contains the element is writable. See "Defining Entity Read-Only Properties" for more information. • The entity that contains the element exists in a project that is unsealed. See "Unsealing Projects" for more information. • The element is not at the root level.
Refactoring	context menu	Select to access a menu of options that enable you to propagate data model changes across a solution without sacrificing model integrity. See "Refactoring Entities and Data Elements" for more information.

Related Topics

Working with Views

Dragging Elements from the Dictionary View

You can add elements to data schemas and to modeling entities by dragging elements from the Dictionary view to an entity editor.

You can drag elements from the Dictionary view to an entity when:

- The target and source entities are writable. See "Defining Entity Read-Only Properties" for more information.
- The target and source entities exist in projects that are unsealed. See "Unsealing Projects" for more information.
- The element is not in an unresolved state. See "Refactoring Entities and Data Elements" for more information about resolving elements.
- The target entity supports the selected entity type.

Notes View

Use the Notes view to provide documentation for the entity or data element selected in the Solution view. You can annotate entities and data elements when you want to communicate to other team members information about the solution.

For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML. See "Notes Tab" for more information.

The Notes view is linked to the Solution view, the Studio Projects view, and the Package Explorer view. When you make a selection in any of these views, Design Studio updates the content in the Notes view based on the active selection.

Outline View

Use the Outline view to view relationships, entities, and data elements related to the entity in the active editor. Also, you can perform operations on those relationship folders, entities, and elements.

The Outline view is linked to the editor in focus. When you switch editors, Design Studio updates the content in the Outline view based on the active selection.

The Outline view enables you to navigate through and to perform operations on relationships, entities, and data elements while maintaining focus on the active entity or data element. See "Refactoring Entities and Data Elements" for more information about the operations that you can perform on data elements in this view.

Overview View

Use the Overview view to review the entire content of any Oracle Communications Order and Service Management (OSM) process, if supported by the active editor. A process may contain hundreds of tasks and subprocesses; the Overview view enables you to navigate throughout a process quickly and to select specific sections of a process for view in the Process editor.

The Overview view always displays the entire diagram. The rectangle represents the position and size of the current view in the Process editor. Move the rectangle to change the section of the diagram displayed in the Process editor. Resize the rectangle to zoom in or zoom out of a specific section of the diagram.

Related Topics

Working with Views

Problems View

Use the Problems view to review short descriptions of each problem marker in a project. Design Studio creates problem markers on data elements in modeling or in the code during implementation and displays warnings, errors, and informational messages that are generated as you work on projects. For example, if a Java source file contains syntax errors, these errors appear in the Problems view. Similarly, if you make configuration errors while modeling entities, error messages appear in this view.

By default, the problems are logged by severity. You can also group the problems by type. The first column of the view displays an icon that denotes the message type (warning, error, or informational) and the description. The remaining columns display the name of the resource that generated the problem, its path, and its directory location.

> **Note:**
>
> Design Studio updates problem markers during project builds. Oracle recommends that you enable the automated build feature to ensure that problem markers are current to the most recent saved content. Problem markers are based on saved content; Design Studio does not update problem markers to reflect unsaved work.

Problem markers indicate that changes are required before the project can deploy successfully. You can double-click any error in the view to open the editor of the affected entity and resolve the problem. In code files, the line containing the problem is highlighted. Design Studio includes multiple layers of validation. Therefore, a single error may generate multiple markers.

Click the **View Menu** button in the Problems view toolbar and select **Configure Contents** to define Problem view configuration.

For best results, select **Errors/Warnings on Selection** to ensure that errors and warnings for the current selection (and children, if applicable) appear.

Additionally, you can filter the Problems view to display only warnings and errors that are associated with a particular resource or group of resources.

See the Eclipse *Workbench User Guide* for information about using the Problems view.

Related Topics

Working with Views

Relation Graph General View

Use the Relation Graph General view to display graphical representations of entity-to-entity type relationships, such as parent-child relationships. With the Relation Graph view open, you can verify parent-child entity relationships after defining the relation.

Action	Location	Use
Link	Toolbar	Click to link or unlink the view to the active editor. When you link views to active editors, Design Studio updates the contents of the view based on the entity displayed in the active editor. For example, when the Relation Graph view is linked, Design Studio updates the contents of the view each time you toggle to a different editor, displaying the relationship graph that is specific to the entity currently displayed in the active editor. You can unlink the view to prevent Design Studio from updating the contents of the Relation Graph view when the entity is no longer displayed in the active editor.
Zoom In **Zoom Out** **Zoom To**	Context Menu	Increase and decrease the size of the objects representing the elements in the view.
Increase Scope **Decrease Scope**	Context Menu	Increase and decrease the area of the relation graph that appears, relative to the active entity.

Related Topics

Working with Views

Solution View

Use the Solution view as an entry point to your service fulfillment solutions, to view the relationships among the elements in a solution, and to adjust the level of detail for efficient navigation and design. The Solution view enables you to explore the associations between entities used to model the service domain in the conceptual model and in application projects.

From the Solution view, you can model, run design patterns against, and refactor entities. The Solution view enables you to organize and view your solutions through products, services, resources, orders, and other entities that implement the solution. This view displays relationships among entities in a workspace and includes child folders that represent relationships defined in your solution.

You can filter the Solution view so that child folders appear even if the relationships are not yet established (the folder will be empty; the Solution view will include these folders to indicate that these are important relationships that can be defined).

For example, when you filter the Solution view to show all folders and to display the orders in your workspace, all Order entities include a **Creation Tasks** folder, a **Default Process** folder, a **Permissions** folder, and so forth. Even when no creation task is associated with the order, the **Creation Task** folder still appears, indicating that a creation task can (and should) be associated with an order.

You can also filter the view to display or hide actions, realizations, and components associated with the entities in the Solution view.

> **Note:**
>
> The Solution view is linked to other views and to the active editor. When you select an entity in the Solution view, all linked views are updated to display information about the selected entity.
>
> See *Design Studio Concepts* for more information about navigating across solutions using the Solution view.

Field/Icon	Use
Category Filters	Click to define options for filtering the Solution view. You can select options to display actions, realizations, and components associated with products, services, and resources. You can also display or hide Functional Area entity realizations and order item parameter bindings.
Add Relationship	Click to add relationships to the selected entity.
New Category Items	Click to add a new category item to the Solution view. The options that are available depend on the category selected. For example, if you select an Order entity in the Solution view **Category** field, you can click the **New Category Item** arrow to create a new order.
Show/Hide Folders	Click to display only primary relationships for the entities that appear in the Solution view.
Category	Filter for entities that are associated with a specific category. Entities associated with the category display at the root level of the Solution view.

Field/Icon	Use
Types	Filter for entity types that have been registered for the selected category, which include all types of entities or elements, text documents, Word documents, and any other types contained in your workspace.
Filter	Enter an entity name and click the **Filter** button (represented by a flashlight icon) to search for a specific entity. Click the **Filter** button again to clear the search term. When searching, consider the following: • Separate terms by a space to filter for both terms. Add **OR** between terms to search for either one or the other. For example, entering **32 OR aa** returns occurrences of terms starting with **32** and **aa**. • Searches return all elements and entities that begin with the search string. • Use an asterisk followed by a term to match the endings of terms. • Use an asterisk to match any number of characters. • Use asterisks on both sides of a term to return strings that contain a match anywhere in the string. • Use a question mark to match single characters. • Variations of strings are returned. For example, a search for ***ID** returns **productID**, **customerID**, and **orderID**. • Use a hyphen (-) before a term to omit specific variations of words (ensure there is a space before the hyphen). For example, a search for ***ID -product** returns **customerID** and **orderID**, but not **productID**.
Link to Editor	Click to link or unlink the view to the active editor. When you link views to active editors, an entity is highlighted in the view when the corresponding entity editor is active. Also, when you select an entity in the view, the corresponding editor becomes active (assuming that editor is open). Unlink the view to prevent Design Studio from updating the contents views when an entity is no longer displayed in the active editor.
View Menu	Click to access the following options: • **Select Working Set**: Select to apply a working set filter to the workspace, or to create a new working set. • **Deselect Working Set**: Select to remove an active working set filter from the workspace. This option is available only when you have previously applied a working set to the workspace. • **Edit Active Working Set**: Select to edit an active working set filter. This option is available only when you have previously applied a working set to the workspace.

Related Topics

Working with Views

Modeling Data Using Context Menus

Structure View

Use the Structure view to view relationships for a selected entity or data element and perform operations on the relationship contents. For some relationships, these folders appear even if the relationships are not yet established (the folder will be empty; the Solution view will include these folders to indicate that these are important relationships that can be defined).

For example, if you select an order entity in the Solution view, the Structure view displays a **Creation Tasks** folder, a **Default Process** folder, a **Permissions** folder, and so forth. Even when no creation task is associated with the order, the **Creation Task** folder still appears in the Structure view, indicating that a creation task can (and should) be associated with an order.

The Structure view is linked to the Solution view, the Studio Projects view, and the Package Explorer view. When you make a selection in any of these views, Design Studio updates the content in the Structure view based on the active selection. The Structure view enables you to navigate through and to perform operations on relationship folder contents while maintaining focus on the active entity. See "Refactoring Entities and Data Elements" for more information about the operations that you can perform on data elements in this view.

Related Topics

Working with Views

Modeling Data Using Context Menus

Solution View

Studio Projects View

Use the Studio Projects view to perform modeling activities and other project tasks. This view has various functions available for project tasks, which are accessible from the Studio Projects view toolbar and context menu.

The tree structure displays Design Studio components organized by project. Each project you are working on is represented by one folder as the top level of a tree, with logical groups of entities below it.

The actions available in the Studio Projects view context menu depend on your current selection in the Studio Projects view. For example, Design Studio filters the list of entity creation actions based on the project type associated with the current selection.

In addition to commands for copying, pasting, and deleting entities in the workspace, the Studio Projects view toolbar and context menu also provide the following commands:

Action	Location	Use
Include Sealed Projects **Exclude Sealed Projects**	Toolbar	Toggle to show and hide sealed projects in the view.
Hierarchical view format **Flat view format**	Toolbar and Context Menu	Toggle to Hierarchical view format to display the Studio Projects view contents in a folder structure. Toggle to the Flat view format to display the Studio Projects view contents with a dot-separated name that includes each folder in the path (if you are using folders to organize your entities).
Include Empty Folders **Exclude Empty Folders**	Toolbar	Toggle between these filter settings to hide or display empty folders.

Action	Location	Use
Link to Editor	Toolbar	Click to link or unlink the view to the active editor. When you link views to active editors, an entity is highlighted in the view when the corresponding entity editor is active. Also, when you select an entity in the view, the corresponding editor becomes active (assuming that editor is open). Unlink the view to prevent Design Studio from updating the contents views when an entity is no longer displayed in the active editor.
View Menu	Toolbar	Click to access the following options: • **Select Working Set**: Select to apply a working set filter to the workspace, or to create a new working set. • **Deselect Working Set**: Select to remove an active working set filter from the workspace. This option is available only when you have previously applied a working set to the workspace. • **Edit Active Working Set**: Select to edit an active working set filter. This option is available only when you have previously applied a working set to the workspace.
Folder Name	Toolbar	Filter entities by location and name combinations based on the pattern and matching position. Enter filter terms into these fields and click the **Filter** button to search for the term. Click the **Filter** button again to clear the search term. When filtering, consider the following: • Separate terms by a space to filter for both terms. Add OR between terms to search for either one or the other. For example, entering **32 OR aa** returns occurrences of terms starting with **32** and **aa**. • Searches return all elements and entities that begin with the search string. • Use an asterisk followed by a term to match the endings of terms. • Use an asterisk to match any number of characters. • Use asterisks on both sides of a term to return strings that contain a match anywhere in the string. • Use a question mark to match single characters. • Variations of strings are returned. For example, a search for ***ID** returns **productID**, **customerID**, and **orderID**. • Use a hyphen (-) before a term to omit specific variations of words (ensure there is a space before the hyphen). For example, a search for ***ID -product** returns **customerID** and **orderID**, but not **productID**.
New	context menu	Select to create new projects and entities. Design Studio filters the options based on the project type associated with the current selection.
Copy	context menu	Select to copy a project or entity.
Paste	context menu	Select to paste a copy of a project or entity into the workspace. Depending on the location, you may be required to edit the name for uniqueness.

Action	Location	Use
Delete	context menu	Select to delete a project or entity from the workspace. **Note:** Oracle recommends that you use the Studio Projects view to delete projects.
Move	context menu	Select to move an entity to a different project.
Rename	context menu	Select to rename an entity.
Open **Open Project**	context menu	Select **Open** to display the editor for the project or entity that you have selected in the Studio Projects view. Select **Open Project** if there is no active selection in the view.
Open Project **Close Project**	context menu	Select to open or close a project in the workspace. See "Closing Projects" and "Opening Projects" for more information. **Note:** Though you can open and close projects using other views, Oracle recommends that you perform these operations from the Studio Projects view.
Expand	context menu	Expands the selected entity to display all child elements of the structure.
Collapse	context menu	Collapses a structured data element and hides all child elements of the structure.
Import	context menu	Select to import a project into the workspace. Multiple import options may appear, depending on the plug-ins installed. See "Importing Projects" for more information. **Note:** Though you can import projects from other views, Oracle recommends that you perform this operation using the Studio Projects view.
Refresh	context menu	Select to refresh the Studio Projects view.
Design Pattern	context menu	Select to apply a design pattern using the Design Pattern wizard. See "Working with Design Patterns" and *Design Studio Developer's Guide* for more information.
Guided Assistance	context menu	Select to open the Design Studio Guided Assistance dialog box, which provides a range of context-sensitive learning aids mapped to the editor or view in focus. See "Using Guided Assistance" for more information.
Deploy	context menu	Select to deploy a cartridge using Optimize Deploy. See "Deploying Cartridge Projects with Optimize Deploy" for more information.
Go Into **Go Back** **Go Home**	context menu toolbar	Select: • **Go Into** to make the current selection the root of the view. Use this action to focus on a branch of the tree, and to reduce the view to only the content currently of interest. • **Go Back** to return to the previous level in the hierarchy. • **Go Home** to return to the default Studio Projects view hierarchy.
Properties	context menu	Select to view the properties of the entity and define preferences.

The Studio Projects view context menu also contains additional actions based on the current selection, as well as actions provided by Eclipse, such as actions that enable you to share

projects and apply patches, compare documents, use local history, and so forth. See the Eclipse *Workbench User Guide* for more information.

Related Topics

Working with Views

Working with Design Studio Menus

Design Studio has two types of menus: the Studio menu and the context menu. Both provide access to many of the same actions. The Studio menu is in the Design Studio Workbench menu that appears at the top of the Design Studio interface. The Studio menu enables you to create new Design Studio entities. The context menu is accessed by right-clicking in a Design Studio view or editor, and enables you to configure Design Studio entities.

The **Studio** menu contains the following:

Menu Option	Use
New	Select to access: • The five most recently used entity creation wizards, which enables you to quickly create additional entities of similar types. • A list of Design Studio project creation actions, which enables you to create new Design Studio projects. • A list of actions that enable you to create application-specific entities. • An **Other** action, which enables you to create an entity of a different type within another project.
Show Environment Perspective	Select to switch to the Environment perspective.
Show Design Perspective	Select to switch to the Design perspective.
Guided Assistance	Select to open the Design Studio Guided Assistance dialog box, which provides a range of context-sensitive learning aids mapped to the editor or view in focus. See "Using Guided Assistance" for more information.
Import Studio Project	Select to import data from external sources into your Design Studio workspaces. See "Importing Projects" for more information.
Design Pattern	Select to apply a design pattern using the Design Pattern wizard. See "Working with Design Patterns" and *Design Studio Developer's Guide* for more information.
Deploy	Perform deploy actions directly from the Design perspective. This enables you to deploy cartridges without needing to switch to the Environment perspective. Additionally, you can select to access actions for Optimize Deploy. See "Deploying Cartridge Projects with Optimize Deploy" for more information.

Related Topics

Working with Design Studio Menus

Working with the Design Studio Toolbar

The buttons on the Design Studio Workbench toolbar enable you to quickly create projects and entities.

> **Note:**
>
> See "Deploying Cartridge Projects with Optimize Deploy" for information about the **Deploy** button. See "Importing Projects" for more information about the **Import Studio Project** button.

Menu Option	Use
New Studio Project	Click to create Design Studio projects. When you click this button, the New Studio Project dialog box appears, containing a list of all Design Studio project types that you can create. Clicking the arrow next to the button displays a menu that contains actions to create Design Studio project types.
New Studio Entity	Click to create Design Studio entities. When you click this button, the New Studio Entity dialog box appears, containing a list of all Design Studio entities that you can create. Clicking the arrow next to the button displays a list of the most recent entities that you created and a complete list of all Design Studio entities. See "Selecting Entity Types" for more information.
Import Studio Project	Select to import data from external sources into your Design Studio workspaces. See "Importing Projects" for more information.
Deploy	Click to perform deploy actions directly from the Design perspective. This enables you to deploy cartridges without needing to switch to the Environment perspective. See "Deploying Cartridge Projects with Optimize Deploy" for more information. Additionally, you can define packaging preferences here. See "Defining Packaging Preferences" for more information.
Guided Assistance	Click to open the Design Studio Guided Assistance dialog box, which provides a range of context-sensitive learning aids mapped to the editor or view in focus. See "Using Guided Assistance" for more information.
Design Pattern	Select to apply a design pattern using the Design Pattern wizard. See "Working with Design Patterns" and *Design Studio Developer's Guide* for more information.

Selecting Entity Types

In some scenarios, you must select an entity type to access the appropriate Design Studio entity creation wizard. You select entity types in the New Studio Entity wizard.

To select an entity type for creation:

1. In the New Studio Entity wizard, select an entity type in the **Entity Type** field.
2. Click **Next**.

   The appropriate entity wizard appears. For example, if you select **Process** in the **Entity Type** field, the Process wizard appears.

Working with Editors

An editor is a special type of view that enables you to edit data, define parameters, and configure settings. Editors contain menus and toolbars specific to that editor and can remain open across different perspectives. You can open entities in editors at any time to modify

existing projects and elements. An asterisk in the editor title bar indicates that the changes you made in an editor are unsaved.

When working with Design Studio editors, see the following topics:

- Defining Editor Preferences
- Displaying Editors
- Using Drag and Drop to Open Editors
- Navigating Among Multiple Editors
- Defining Entity Notes
- Defining Entity Read-Only Properties
- Displaying Editor Help
- Using Guided Assistance
- Using Cheat Sheets
- Design Studio Common Editor Tabs

Defining Editor Preferences

The Design Studio Workbench supports multiple open editors at any one time. You can define preferences to control editor behavior across a workspace. For example, you can define the number of open editors allowed to ensure that Design Studio does not run out of available resources.

To define Design Studio editor preferences:

1. From the **Windows** menu, select **Preferences**.

   The Preferences dialog box appears.
2. Expand the **General** directory.
3. Click **Editors**.

   The Editors preferences page appears. See the Eclipse Help for more information about the fields on this page.
4. (Optional) Select **Close editors automatically**.
5. (Optional) In the **Number of opened editors before closing** field, indicate the number of editors that can remain open at any one time.

   The system automatically closes a previously opened editor if you exceed this number of open editors.
6. Click **OK**.

   Design Studio saves your changes and closes the Preferences dialog box.

Related Topics

Working with Editors

Displaying Editors

Design Studio editors are associated with entities. Many Design Studio views enable you to double-click on entities to open the entity in the associated editor. For example, you can double-click an entity in the Solution view or Studio Projects view to open the associated editor.

Additionally, you can double click on table entries that reference entities to open the entity in the associated editor.

> **Note:**
>
> See the Eclipse *Workbench User Guide* for more information about associating editors with file types.

You can open entity editors at any time to modify existing projects and elements.

To display the editor for a project or element:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Studio Projects** tab.

   The Studio Projects view appears.
3. Double-click a project or entity.

   The editor associated with the entity appears, enabling you to view and edit the information. For example, if you double-click a Project entity, the entity opens in the Project editor (you can also double-click a project folder to open the Project entity in the Project editor).

Related Topics

Working with Editors

Using Drag and Drop to Open Editors

Design Studio supports drag and drop functionality, enabling you to drag files or entities from the Studio Projects view to editors. Additionally, you can open an editor associated with an entity by dragging the entity from a view into the editor area.

Related Topics

Working with Editors

Navigating Among Multiple Editors

When you have multiple editors open, select **Navigate**, then select **Back** to access a list of previously used editors (**Alt** key + **E**). Also, you can navigate through the editor list by pressing the **Alt** key and the left arrow to return to a previously used editor and the **Alt** key and the right arrow to move back.

By default, the Studio Projects view is linked to the currently open editor. For example, closing or switching editors does not change the highlighted selection in the Studio Projects view. To always display in the Studio Projects view the file currently being edited, ensure that the **Link to Editor** button in the Studio Projects view toolbar is enabled.

Related Topics

Studio Projects View

Working with Editors

Defining Entity Notes

Some Design Studio editors include a **Notes** button that you can use to attach documentation to an entity. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML.

To define notes for an entity:

1. From the Studio Projects view, double-click an entity.

   The entity opens in the appropriate editor.
2. Click the **Notes** button.

   This button is located in the top right corner of Design Studio editors and is represented by a note pad, pencil, and lowercase letter "i" icon.
3. Enter information about the entity.
4. Do one of the following:
   - Press **F2** to apply the changes.
   - Press **Esc** to close the text box without saving changes.

Related Topics

Working with Editors

Notes Tab

Defining Entity Read-Only Properties

You can create read-only entities by changing the read-write property of the entity. Read-only entities can exist independently of sealed projects. For example, you can have read-only entities in unsealed projects, and sealing a project does not change any of the project entity read-write properties to read-only. When you define an entity's read-write property as read-only, the editor for that entity displays **[Read-only]** in the title bar.

To define read-write properties:

1. In the Studio Projects view, right-click the entity and select **Properties**.

   The Properties dialog box appears.
2. In the left column, click **Resource**.
3. Select **Read only**.
4. Click **Apply**.
5. Click **OK**.

   Design Studio prevents you from editing any of the data in the entity. To change the read-write property, deselect **Read-only** in the Properties for dialog box.

Related Topics

Working with Editors

Displaying Editor Help

Some Design Studio editors include an editor help button that you can use to open the Help view. The view contains a list of topics relevant to the corresponding entity. You can use the Help view to review Design Studio Help without leaving the workbench.

The help button is located at the upper right corner of an editor and is represented by a question mark.

Related Topics

Working with Editors

Using Guided Assistance

Design Studio guided assistance provides a range of context-sensitive learning aids mapped to specific editors and views in the user interface. For example, when working in editors, you can open the Guided Assistance dialog box for Help topics, cheat sheets, and recorded presentations that are applicable to that editor.

You can access guided assistance from the Design Studio toolbar, from the **Studio** menu, and in the Studio Projects view context menu. The **Guided Assistance** button in the toolbar is represented by a question mark and play button combination.

Using Cheat Sheets

Cheat Sheets are XML documents that can be interpreted by the Eclipse Cheat Sheet framework to provide user assistance. For example, when developing design patterns, you can include a cheat sheet to describe the resources added to a workspace, and to assist users with any manual steps required after a design pattern is applied. Cheat sheets are not mandatory for design patterns, but recommended.

You can edit cheat sheets using any XML editor. Eclipse provides a Cheat Sheet editor that facilitates cheat sheet development.

For information about creating and developing cheat sheets, see "Building Cheat Sheets in Eclipse" on the Oracle Technology Network:

http://www.oracle.com/technetwork/articles/entarch/eclipse-cheat-sheets-092351.html

Design Studio Common Editor Tabs

This section describes editor tabs and functionality used by multiple Oracle Communication features. For information about tabs and functionality specific to a feature or entity, see the section of the Help for the editor type.

When working with common editors and functionality, see the following topics:

- About Design Studio Common Editor Tabs
- About Control Types
- Details Tab or Attributes Tab
- Enumerations Tab
- Tags Tab

- Usage Tab
- Notes Tab
- Settings Tab

About Design Studio Common Editor Tabs

Some application editors utilize common Design Studio components to enable you configure entities by modeling a data tree to hierarchically represent all associated data elements. These common components facilitate the reuse of data elements within a modeling solution and provide tools for locating and using existing data elements.

These components consist of a data tree and a collection of supporting tabs. You can select a data element in the data tree to review and model details of the selected data element. Additionally, you can access a context menu from the data tree to perform various types of refactoring operations on the data elements.

Related Topics

Details Tab or Attributes Tab

Enumerations Tab

Tags Tab

Settings Tab

About Control Types

You can define the manner in which run-time application users work with data elements by specifying a control type. Control types have specific options that define or limit the information stored for the data element:

Control Type	Description
Text	Enables users to enter characters. The properties you specify for the text field data element determine what users can enter in the field. This control type is available for elements defined with the following primitive data types: **dateTime**, **decimal**, **double**, **float**, **hexBinary**, **string**, and **time**.
Numeric	Enables users to enter integers. This control type is available for elements defined with the following primitive data types: **int** and **long**.
Check Box	Provides a Boolean data type with true and false values. This control type is available for **boolean** elements only.
Drop Down List	Displays a list of values. In Design Studio, you can define the possible values for the list in several different ways. This control type is available for elements defined with the following primitive data types: **decimal**, **double**, **float**, **hexBinary**, **int**, **long string**, and **time**.

Control Type	Description
Calendar	Enables users to enter or select date and time. The properties you specify for the data element define a range of valid dates, time, and a default date. This control type is available for elements defined with the **date** and **dateTime** primitive data types.
URL	Displays a URL. This control type is available for **string** elements.

Related Topics

Settings Tab

Details Tab or Attributes Tab

Use the **Details** tab and the **Attributes** tab to define specific constraint values for the selected data element. You use the Attributes tab when working with conceptual model entities.

Field	Use
Type or **Source**	Click the adjacent **Select** button to select a data element as the data type and to inherit from that base type. When working in the Data Schema editor, if the data element initially inherits data from a base type, click the **Clear** button to remove the link (on refresh, Design Studio removes the read-only data). Click the label link to navigate to the base type of a data element (you can also double-click a data element to navigate to its base type, when defined). See "Leveraging Existing Data Information" and "Deriving from Base Type Elements" for more information.
Primitive Type	Displays the data type for simple data elements, and displays **Structure** for structured data elements. If this element inherits from a base type, this field displays the type of the base type. Design Studio supports the following primitive data types: • string • int • long • decimal • hexBinary • date • dateTime • boolean • float • double • time Additionally, you can select **Structure** from the list to convert simple data elements to structures. **Note:** Design Studio clears the **Type** field if you change the **Primitive Type** field to a value that is not compatible with the base type. You cannot change the **Primitive Type** field value when working in the **Attributes** tab of a conceptual model editor.

Field	Use
Name	Displays the name of the element as saved in the file system, given to the element at the time of creation. This field is read-only. See "Refactoring Entities and Data Elements" for more information about changing data element names.
Display Name	Edit the data element display name. The Data Schema editor supports multiple languages for this field. The field adjacent to **Display Name** displays your language. You can define a **Display Name** field value for any language you select from the list. If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.
Path	Displays an XPath expression to define the location of the node in the schema relative to the root. This field is read-only.
Namespace	Identifies the namespace in which the selected data element exists, and identifies the version within the namespace, if applicable. Every data element belongs to a single namespace. If the data element is inherited from a base element, the namespace of the base element appears in this field.
Multiplicity	Use these fields to define the data element cardinality. The **Minimum** field indicates the minimum number of times the data element can appear in an instance document, and the **Maximum** field indicates the maximum number of times the data element can appear. When defining data element cardinality, you indicate whether the element is **Required** or **Optional** or whether there can be multiple occurrences of the element (**Range**). If you select **Range**, you can define the cardinality using one of the following configurations: • *Range with at least one occurrence*: Select the value **1** in the **Minimum** field and select **Unbounded** (no explicit limit) in the **Maximum** field. • *Range with no required minimum number of occurrences*: Select the value **0** in the **Minimum** field and select **Unbounded** in the **Maximum** field.
Suppress	Select to suppress an inherited data element. Suppressed elements remain in the Data Elements area data tree but are not included in the parameter set. This field is not available on the **Attributes** tab.
Abstract	Select to indicate that the data element is intended to be inherited and not referenced. If abstract data elements are referenced, a warning marker will appear in the Problems view. This field appears only in the Schema editor and in data element creation dialog boxes.
Internal	Select to indicate that the data element cannot be used as a base type from which other elements inherit. This field appears only in entity editors that can contribute to the Data Dictionary (for example, the Activation Atomic Action editor, the Inventory Specification editors, and the Inventory Configuration Specification editors).

Field	Use
Deprecated	Select to discourage use of the data element. If the data element is used, a warning marker will appear in the Problems view. Data elements can be marked as deprecated at any level of the hierarchy.
Sensitive	Select to protect the contents of a text or numeric data element that contains sensitive information. For example, select this option for fields that are used as password fields.
Length	Specify the minimum and maximum length of String and HexBinary data types. This field is read-only for data elements that inherit from a base type. You must define the minimum and maximum lengths with a non-negative integer between 0 and 9999. Select **Unbounded** to define the maximum length as 9999. For HexBinary data types, you must define the length as an even integer. Design Studio generates an error marker if you define HexBinary types with odd-numbered minimum or maximum lengths. **Note:** Oracle Communications features may include application-specific limitations on the length, and may generate problem markers when a data element exceeds this limitation. For example, Oracle recommends that you do not define data elements tagged as characteristics with maximum length values greater than 255. Design Studio for Inventory stores all characteristic values as strings of length 255. Also, Design Studio for Order and Service Management does not support simple or structured data element length values greater than 1000 characters, except in specific circumstances (see *OSM Developer's Guide* for more information).
Unit	Displays the unit of measure. Click **Unit** to define a new unit of measure for any integer data element types. Click **Select** to select from existing units of measure.
Default	(Optional) Assign a default value for simple data elements. The default value must be of the same type as the data element. For example, if the data element is a String, the default value must be a String. The length of the default value must be equal to or less than the value defined for the maximum length of the data element.

Related Topics

Design Studio Common Editor Tabs

Enumerations Tab

Use the **Enumerations** tab to define sets of valid enumeration values for elements. Enumerations (also referred to as look-up or drop-down values) represent actual values that elements can take as valid values.

Field	Use
Included table	Click the placeholder text inside the **Code** and **Description** columns to change the code name and description for the enumeration. The value you enter in the **Code** field is stored in the database. The value you enter in the **Description** field is displayed to the user. Enumerations that are inherited from a base type are read-only. Click inside the Default column and select **Yes** to make the selected value the default value in the run-time environment.
Add	Click **Add** to add an enumeration for the data element you selected in the Included table.
Excluded table	Displays inherited enumerations that you want to exclude from the list of values in the run-time environment. Move enumerations between the Included and Excluded tables using the arrow buttons. Use this capability to display a subset of the inherited enumerations as valid values for the element in the run-time application.
Language	Specify the language in which to display the selected enumeration value. Select a language in the adjacent field to create a different value for the selected language. If you create no specific value for a selected language, the default value is displayed. If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.

Related Topics

Design Studio Common Editor Tabs

Tags Tab

Use the **Tags** tab to characterize data elements with keywords. A set of tags is delivered with Design Studio. These tags cannot be inherited.

Field	Use
Name	Displays the list of tags associated with this data element.
Remove	Click to remove the association of the selected tag.

Field	Use
Add	Click to associate a system-defined tag with the data element. The following tags are available: • **Changeable**: The data element value changes frequently. For example, you might apply this tag to the **DownloadSpeed** data element because a customer can upgrade their service to a higher download speed at any time, and you want the ability to track this value and maintain the history. Tag data elements as **Changeable** if you want the data element to be realized as a characteristic on a Design Studio for Inventory Service Configuration specification. • **Characteristic**: The data element is relevant to Design Studio for Inventory and Design Studio for Network Integrity data models. Tag data elements as **Characteristic** if you want the data element to be realized as a characteristic on a Design Studio for Inventory Service specification. • **Control Data**: The data element is data that OSM requires to perform orchestration. See "About Modeling Control Data" for more information. • **Fulfillment Function**: The data element is relevant to an OSM fulfillment function. See "Adding a New Fulfillment Function" for more information. • **Ignore Characteristic in Network Integrity**: The data element is used during Design Studio for Inventory modeling but is not used in Design Studio for Network Integrity data modeling. • **Implicit Parameter**: The data element is a default data element defined in a functional area and inherited by service actions. • **Order Item Property**: The data element is a property associated with an OSM order item. See "Working with Order Items" for more information. • **Persisted**: The data element is saved to a conceptual model entity and realized to an Inventory specification. • **Realization Item**: The data element is realized from a conceptual model Resource specification to an Inventory configuration item. • **Target**: The data element is mapped to the **Atomic Action Label** field with the value **MCLI**. When creating technical actions, you are required to tag one simple data element with the **Target** tag. See "Configuring Actions" for more information.

Related Topics

Design Studio Common Editor Tabs

Working with Tags

Working with Conceptual Models

Working with Characteristics

Working with Configurations

Usage Tab

Use the **Usage** tab to review the projects and entities in which a data element is used. Additionally, this tab displays all references to a specified data element.

Related Topics

Design Studio Common Editor Tabs

Notes Tab

Use the **Notes** tab to add documentation to an entity or data element. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML.

Design Studio supports multiple languages for this tab. The field at the top of this tab displays your list of languages. If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.

Inherited data elements are read-only.

Settings Tab

Use the **Settings** tab to configure the presentation of data elements in run-time applications.

Field	Use
Display Name	Enter the field label that appears for the data element in the run-time application. Design Studio supports multiple languages for this field. The adjacent field displays your language. You can define a value for any language you select from the list. If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.
Tool Tip	Enter the text that appears for a data element when a user moves the mouse over the corresponding field in a run-time application. Design Studio supports multiple languages for this field. The adjacent field displays your language. You can define a value for any language you select from the list. If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.

Field	Use
Control Type	Specify the manner in which run-time application users interact with the corresponding data element. The values that are available in this field depend on the **Primitive Type** value defined for the data element on the "Details Tab or Attributes Tab". Additionally, the value that you select in this field determines the subsequent options that appear. • Select **Text** to enable the user to enter alphanumeric text into this field. • Select **Numeric** to enable the user to enter numeric characters (integers) into this field. • Select **Calendar** if you intend for the user to select a date and time for this field. • Select **DropDown** if you intend the user to select from a list of enumerations. • Select **CheckBox** to indicate that the field is a Boolean option. • Select **URL** to indicate that the field is a uniform resource locator.
Default Value	Enter a value to populate the data element with a default setting in a run-time application.
Required	Select if a user must enter information in this field when working in a run-time application.
Sensitive	Select to protect the contents of a text or numeric data element that contains sensitive information. For example, select this option for fields that are used as password fields. When you select this option, the **Default Value**, **Edit Mask**, and **Display Mask** fields are disabled. Default values and masks are not supported for passwords. If the data element is marked as **Sensitive** on the **Details** subtab or on the **Attributes** subtab, the **Sensitive** option is disabled in the **Settings** tab. See "Details Tab or Attributes Tab" for more information.
Read Only	Select to make the data element a read-only field in the run-time environment.
Display Mask	Enter a mask to control how text in read-only fields is formatted and displayed in a run-time environment. You use Java regular expressions to define display masks.
Edit Mask	Enter a mask to control how text in editable fields is formatted and displayed in a run-time environment. You use Java regular expressions to define masks.
Case	Select the display format for text when the element appears in the run-time environment. You can format the text using all uppercase, all lowercase, or mixed case.
Numeric Range	Define a range of values (for example, 1-9) that a user can enter in the field. Select **Unbounded** to specify no limitations on the **Upper** value. This option appears only when you select **Numeric** in the **Control Type** field.
Date Range	Select a date range between which the field values are valid. This option appears only when you select **Calendar** in the **Control Type** field.
Sort	Select to sort the enumerations list by alphanumeric characters (strings) or by numeric characters (integers). This option appears only when you select **DropDown** in the **Control Type** field.

Field	Use
Enable Search	Select to provide a browse option in the run-time environment, which opens a pop-up window. You can customize the pop-up window as per the requirement. See *UIM Developer's Guide* for more information on customizing the user interface of pop-up window.

4

Modeling Data

Oracle Communications business solutions require definitions of data types and structures that are managed within and passed among multiple applications. Modeling these data types and structures consistently across products enables reuse of data modeling constructs and simplifies product integration.

A design-time data model represents the data configuration required by your solution design. When designing your data model, you define simple and structured data elements once (in data schemas) and leverage those definitions across multiple products. Design Studio refers to the conceptual collection of all data schemas and data types in the workspace as the Data Dictionary. The Data Dictionary is a logical repository of data types and structures and the key reference point for consistent modeling of data within the solution design.

When modeling data in Design Studio, see the following topics:

About Data Modeling

There are two basic approaches for modeling data in Design Studio:

- Data-centric

  In the data-centric approach, you first model data for a cartridge project and then model the cartridge project entities using specific data, as needed. To model the data, you use the Data Schema editor, which you access by double-clicking a Data Schema entity icon in one of several Design Studio views. For example, you can filter the Solution view to display data schemas and you can view data schemas in the Studio Projects view. Also, you can double-click data elements in the Dictionary view to open Data Schema editors. See "Solution View" and "Dictionary View" for more information.

- Entity-centric

In the entity-centric approach, you first model your business process and entities, and then model the data specifically required by the entities used by the business process. The data for these entities can be modeled directly within their respective editors.

Oracle recommends that you use the Data Schema editor for all of your data modeling activities. Third-party XML Schema editors (including the Eclipse editor) may not support all Design Studio data types and may not fully support refactoring throughout the entire model. See "Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions" for information about modeling data. See *Design Studio Developer's Guide* for information about working with externally created schemas.

> **Note:**
>
> Do not edit Design Studio entities directly in XML. For example, do not open and edit model entities in a text editor. Create and edit Design Studio entities using Design Studio editors only.

About the Data Dictionary

When you design your fulfillment solution, you need to develop a model comprised of data for use in multiple applications. For example, you may want to create order templates, atomic actions, and service specifications, and share the data defined for those entities across your OSM, ASAP, and Inventory applications.To facilitate this data sharing across products, Design Studio includes the concept of the Data Dictionary.

The Data Dictionary is a logical repository of data elements and data types in a workspace. Data types can be contributed from entities in the workspace, such as Activation service actions. The data elements are created and saved in data schemas, which can be accessible across all projects in a workspace. Design Studio automatically creates a project-specific data schema when you create a cartridge project. You can use this default schema to contain the data you require to model the project, you can create multiple schemas in the same project, or you can create schemas in common projects. You can model your cartridge project using data from any combination of these data schemas.

Creating Data Schema Entities

Design Studio enables you to leverage the data elements (simple and structured) contributed by various entities and available for use in the workspace. When modeling data, you can create common data schemas (for example, schemas in Model projects) to enable product-agnostic data type modeling. Also, you can create product-specific data schemas (for example, data schemas in OSM, Network Integrity, and Inventory projects).

> **Note:**
>
> When using data schemas created outside of Design Studio, ensure that the schema defines a target namespace.

To create a new Data Schema entity:

1. From the **Studio** menu, select **New**, then select **Model**, and then select **Data Schema**.

The Data Schema wizard appears.

2. In the **Project** field, select the project in which to save this entity.
3. In the **Name** field, enter a name for the data schema.

   The name must be unique among Data Schema entity types in the same namespace.
4. (Optional) Select a location for the data schema.

   By default, Design Studio saves the Data Schema entity to the root level of the project **dataDictionary** folder. You can enter a folder name in the **Folder** field or select a location different from the default if you want to create additional subfolders. To select a different location:

   a. Click the **Folder** field **Browse** button.
   b. Navigate to the directory in which to save the entity.
   c. Click **OK**.
5. Click **Finish**.

   Design Studio adds the new Data Schema entity to the selected project in the Studio Projects view.

Related Topics

Data Schema Editor

Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions

When modeling data for a project, you can create simple data elements, structured data elements, and data structure definitions that you can reuse throughout your model. Also, you can define values for data elements, called enumerations, that are available for selection in a run-time environment.

When creating simple data elements, structured data elements, and data structure definitions, see the following topics:

- About Simple and Structured Data Elements
- About Data Structure Definitions
- Creating Simple Data Elements
- Creating Structured Data Elements
- Creating Data Structure Definitions
- Creating Data Element Enumerations

About Simple and Structured Data Elements

Simple data elements are reusable data types that contain no child dependencies. A simple data element has no structure, and is associated—directly or indirectly—to a primitive type (int, boolean, char, and so forth).

Structured data elements are reusable, complex data types that include embedded data types. Structured data elements contain simple data elements and other structured data elements.

For example, you might create a structured data element called **building** that contains the **floor**, **room**, **aisle**, **rack**, and **shelf** child elements. Structured data elements can also contain other structured data elements. For example, a structured data element called **person** might contain the child elements **firstName**, **lastName**, and the child structured data element **address**.

Related Topics

Creating Simple Data Elements

Creating Structured Data Elements

About Data Structure Definitions

Data structure definitions are structured data elements that you can use to create generic model definitions that are instantiated in the run-time environment.

Data structured definitions can be defined as **abstract** or **final**. A data structure definition that is defined as **abstract** cannot be instantiated; it is only intended to be extended by other entities. For example, an abstract data structure definition called **vehicle** can be extended by non-abstract definitions called **car** and **motorcycle**.

A data structure definition defined as **final** cannot be extended by other data structure definitions. For example, if the data structure definition **car** is defined as **final** it cannot be extended by other data structures. If it is not defined as **final**, the **car** data structure can be extended, for example, by the **coupe** data structure and by the **convertible** data structure.

You create data structure definitions in Model projects. After you create a data structure definition, open the entity in the Data Structure Definition editor to define its attributes, details, behaviors, and so on. You can define a key for each data structure definition, which uniquely identifies each instance of the data structure definition. See "About OSM Data in Model Projects" for more information.

Related Topics

Creating Data Structure Definitions

Creating Data Structure Definitions from Existing Data Elements

Data Structure Definition Editor

Creating Simple Data Elements

Simple data elements are simple data types. A simple data element cannot contain any child elements.

To create simple data elements:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Dictionary** tab.
3. In the Dictionary view, right-click and select **Add Simple Schema Element**.

   The Create Data Schema Element dialog box appears.
4. In the **Entity** field, select the schema entity into which the new element will be saved.
5. In the **Type** field, do one of the following:
   - Click **Select** to select an existing simple data element as the base type for the new element. If you do not see any existing data elements in the resulting window, deselect

the **Filter Project Dependencies** check box. The new element becomes a subtype of the element you select here, and inherits the base type attributes. See "Leveraging Existing Data Information" and "Deriving from Base Type Elements" for more information about extending existing elements.

- Click **Clear** to remove the association of the base type element and to use a primitive type as the base type.

6. In the **Primitive Type** field, select a data type for the new element.

   If the new element inherits from a base type element, the primitive type of the base type element displays in this field. Base type and subtype elements must both be defined by the same primitive type.

7. In the **Name** field, enter the name for the data element.

   Design Studio uses the value defined in the **Type** field (if specified) as the default value in the **Name** field. Oracle recommends that you use the default value.

8. In the **Display Name** field, enter the name for the new element that appears in Design Studio editors and in the run-time environments.

   Design Studio supports multiple languages for this field. The field adjacent to **Display Name** displays your list of languages. You can define a display name for any language you select from the list.

   If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.

9. In the **Multiplicity** field, define the minimum and maximum number of times the data element can appear in an instance document.

   See "Details Tab or Attributes Tab" for more information about defining values for this field.

10. (Optional) Select **Abstract** to indicate that the data element is intended to be inherited and not referenced.

    If abstract data elements are referenced, a warning marker will appear in the Problems view.

11. In the **Length** field, specify the minimum and maximum lengths for primitive types that support length restriction.

    See "Details Tab or Attributes Tab" for more information about defining values for this field.

12. In the **Default** field, assign a default value for simple data elements.

    The default value must be of the same type as the data element. For example, if the data element is a string, the default value must be a string.

13. Click **OK**.

    The new simple element appears in the Dictionary view.

Related Topics

About Simple and Structured Data Elements

Data Schema Editor

Creating Structured Data Elements

Structured data elements are reusable, complex data types that include embedded data types. Structured data elements contain simple data elements and other structured data elements.

To create structured data elements:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Dictionary** tab.
3. Right-click in the Dictionary view and select **Add Structured Schema Element**.

   The Create Data Schema Structure dialog box appears.
4. In the **Entity** field, select the schema entity into which the new data element will be saved.
5. In the **Type** field, do one of the following:
   - Click **Select** to select an existing structured data element as the base type for the new element. The new element becomes a subtype of the element you select here, and inherits the base type attributes. See "Leveraging Existing Data Information" and "Deriving from Base Type Elements" for more information about extending existing elements.
   - Click **Clear** to remove the association of the base type and to use **Structure** as the base type.
6. In the **Name** field, enter the name for the data element.

   Design Studio uses the value defined in the **Type** field (if specified) as the default value in the **Name** field. Oracle recommends that you use the default value.
7. In the **Display Name** field, enter the name for the new data element that appears in Design Studio editors and in the run-time environments.

   Design Studio supports multiple languages for this field. The field adjacent to **Display Name** displays your list of languages. You can define a display name for any language you select from the list.

   If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.
8. In the **Multiplicity** field, define the minimum and maximum number of times the data element can appear in an instance document.

   See "Details Tab or Attributes Tab" for more information about defining values for this field.
9. (Optional) Select **Abstract** to indicate that the data element is intended to be inherited and not referenced.

   If abstract data elements are referenced, a warning marker will appear in the Problems view.
10. Click **OK**.

    The new structured element appears in the Dictionary view.

Related Topics

About Simple and Structured Data Elements

Data Schema Editor

Creating Data Structure Definitions

Data structure definitions are structured data elements that you can use to create generic model definitions that are instantiated in the run-time environment.

To create a data structure definition:

1. From the **Studio** menu, select **Show Design Perspective**.

2. In the Studio Projects view, right-click a model project, click **New**, and then select **Data Structure Definition**.

   The Data Structure Definition dialog box appears.

3. (Optional) Click the **Extends** field **Select** button.

   In the Matching items area, select a data structure definition that you want the new data structure definition to extend, or click **New** to create a new data structure definition that you want this data structure definition to extend.

4. In the **Name** field, enter a name for the data structure definition.

   The name must be unique among data structure definition entities.

5. (Optional) Select a location for the data structure definition.

   By default, Design Studio saves the data structure definition entity to the root level of the project folder. You can enter a folder name in the **Folder** field or select a location different from the default if you want to create additional subfolders. To select a different location:

   a. Click the **Folder** field **Browse** button.

   b. Navigate to the directory in which to save the entity.

   c. Click **OK**.

6. Click **Finish**.

   The new data structure definition entity appears under the model project in the Studio view.

Related Topics

About Data Structure Definitions

Data Structure Definition Editor

Creating Data Structure Definitions

Creating Data Element Enumerations

You can define values for data elements that are available for selection in a run-time environment. For example, you can define a set of values for data elements that appear as drop down lists in the run-time environment.

To create data element enumerations:

1. From the **Studio** menu, select **Show Design Perspective**.
2. In the Studio Projects view, double-click any Data Schema entity.

   The entity appears in the Data Schema editor.

3. In the Data Element tree, select a data element.

   The data element subtabs appear in the Element area.

4. Click the **Enumerations** tab.
5. Click **Add**.

   Design Studio creates a new row in the Included table and uses placeholder text for the **Code** and **Description** values.

6. Click the placeholder text in the **Code** or **Description** column.

   The placeholder text becomes editable.

7. Edit the placeholder text.
8. (Optional) To make the enumeration the default value in the run-time environment, click inside the Default column and select **Yes**.
9. (Optional) To specify the language in which to display the selected enumeration value, enter the name in the **Language** field.

   If your preferences are set up to work in one language only, the system displays only the **[default]** option. See "Defining Language Preferences" for more information.

Related Topics

Enumerations Tab

Adding Existing Simple and Structured Data Elements to Entities

You can add existing data simple and structured data elements to entities when modeling data in an entity editor. This procedure is initiated from editors capable of contributing data type information to the Data Dictionary. These editors enable you to create and model persistent data elements outside of the Data Schema editor.

To add existing simple and structured data elements to entities:

1. Right-click in an editor data tree area and select an option appropriate to the editor.

   For example, from the ASAP Atomic Action editor, you would select **Select Simple Data Element** or **Select Structured Data Element**. From an Inventory Specification editor, you would select **Select Characteristic**, and so forth.

   A selection dialog box appears.

   > **Note:**
   >
   > The elements that appear in the list are filtered based on the dependencies defined for the project and based on the target entity.

2. (Optional) In the **Entity Name** field, enter the name of an entity to display and select from only those data elements that are contained in the specified entity.
3. (Optional) In the **Element** field, enter the name of and search for a specific data element.
4. Select elements from the search results, and click **OK**.

   The elements are added to the editor entity.
5. If you are configuring conceptual model entities and you are unable to locate the data elements that you require, you can create new data elements.

   Do the following:

   a. Select **Create New Element**.
   b. Click **Next**.
   c. See "Creating Simple Data Elements" for more information.

Related Topics

About the Data Dictionary

Leveraging Existing Data Information

To increase modeling efficiency, Design Studio enables you to create new data elements that obtain attributes from other data elements. In Design Studio, this is called deriving from a base type element, where the new element automatically obtains the information in the base element.

Leveraging information already defined for base types enables you to define attributes once, share the common attributes among multiple entities, and edit those entities in a single location. Changes that you make to a base type automatically cascade to all entities that derive from that base type.

See "Deriving from Base Type Elements" for more information. See *Design Studio Concepts* for more information about leveraging information defined for existing data elements.

Related Topics

Extending Design Studio Entities

Deriving from Base Type Elements

When modeling simple and structured data elements in Design Studio, you can create new data elements that derive from existing base types. Rather than referencing one of the primitive types (int, boolean, char, and so forth), you reference another data element as their data type.

To create data elements that derive from existing base types:

1. From the **Studio** menu, select **Show Design Perspective**.
2. Click the **Dictionary** tab.
3. Right-click in the Dictionary view area and select the appropriate menu option.
   - To create a data element that will contain no child elements, select **Add Simple Schema Element**.
   - To create a data element that will contain child elements, select **Add Structured Schema Element**.
4. In the **Type** field, click **Select**.

   The Data Element Selection dialog box appears.
5. Select an existing data element as the base type for the new element.

   The new element becomes a subtype of the element you select here, and inherits some of the base type element attributes.
6. Enter information for all required fields.

   See "Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions" for more information.

Extending Design Studio Entities

You can increase your modeling efficiency by extending entities. Data extensibility enables you to leverage data when building new, similar entities. When you extend one entity from another, the target entity inherits all of the data elements defined for the extended entity. For example, you can extend orders and tasks.

Inherited data elements are read-only. If you extend an entity that includes structured data elements, you can add any number of additional simple and structured child elements. In Design Studio, you can extend conceptual model entities, tasks, and orders.

For more information, see the following topics:

- Conceptual Model Editor Properties Tab
- About Order Extensions and Inheritance
- About Task Extensions and Inheritance

Refactoring Entities and Data Elements

Design Studio enables you to propagate data model changes across a solution without sacrificing model integrity. Using the **Refactoring** menu, Design Studio enables you to rename and change the location of entities and data elements. Also, you can create similar data entities, modular and reusable data structures, and so forth. When you refactor entities and data elements, Design Studio updates all references to the entity or data element.

> **Note:**
>
> Close editors prior to refactoring. Editors that remain open during refactoring may display incorrect data or may display data marked with errors. Restart Design Studio to correct these issues.

> **Note:**
>
> When you make design-time changes to existing cartridges and want to deploy those changes to a production environment, Oracle recommends that you increment the cartridge version number before re-deploying the cartridge.

When refactoring entities and data elements, see the following topics:

- Renaming Entities and Data Elements
- Moving Entities and Data Elements to Different Schemas
- Changing Data Element Base Type References
- Making Data Elements Modular and Reusable
- Creating Data Structure Definitions from Existing Data Elements
- Referencing New Base Types for Unresolved Data Elements
- Design Studio Refactoring Menu

Renaming Entities and Data Elements

You can rename entities and data elements and update all references to the new name.

> **Note:**
>
> See "Renaming Conceptual Model Entities and Realized Application Entities" for information about renaming conceptual model entities that you have previously realized as application entities.

To rename entities and data elements:

1. Select an entity from a Design Studio view or a data element from an editor data tree or from a Design Studio view.

   See "Design Studio Common Editor Tabs" for more information about editor data tree areas.
2. Right-click and select **Refactoring**, and then select **Rename**.

   The Refactoring dialog box appears.
3. In the **Name** field, enter a new name for the data element.
4. Click **Next**.

   Design Studio displays a list of all the changes to be performed.
5. Deselect any changes to be performed to exclude specific instances from the refactoring.

   > **Note:**
   >
   > Deselecting elements that are referenced can cause model integrity violations.
6. Click **Finish**.
7. Design Studio updates the data element name for all selected references.

Related Topics

Refactoring Entities and Data Elements

Renaming Conceptual Model Entities and Realized Application Entities

This topic describes how to change the name of a Design Studio conceptual model entity (for example, a customer facing service, a resource facing service, a resource, and so forth) after you have realized the entity and created application entities.

See *Design Studio Concepts* for more information about working with conceptual models.

You can rename conceptual model entities using the refactoring context menu options. However, if the conceptual model entity has been previously realized and is associated with one or multiple application entities, you must also update the existing realized application entity names before using the Design Studio Synchronize feature. If you do not, the Design Studio Synchronize feature creates new application entities with the new names, rather than updating the original application entities. This scenario results in multiple realized application entities for the same conceptual model entity, those created initially with original names, and those created after synchronization with the new names.

To rename a conceptual model entity and the realized application entities:

1. Right-click a conceptual model entity and select **Rename**.

The Rename Studio Model Entity dialog box appears.

2. Enter the new name for the entity, and then click **Next**.

   Design Studio displays a list of all the changes to be performed.

3. Deselect any changes to be performed to exclude specific instances from the refactoring.

4. Click **Finish**.

   Design Studio updates the entity name for all selected references.

5. From the Solution view, expand the conceptual model entity.

6. Right-click a realized application entity, select **Refactoring**, and then select **Rename**.

7. Rename all realized application entities to match the new conceptual model entity name.

   For example, if you renamed a Customer Facing Service entity from **Broadband** to **Broadband_new**, change the name of the realized Service Specification entity to **Broadband_new** and make similar changes to the realized Service Configuration Specification (from example, change **Broadband_Configuration_v1-0-0** to **Broadband_new_Configuration_v1-0-0**).

> **Note:**
>
> If you are renaming a conceptual model Resource entity that realizes an Inventory entity that does not support configuration specifications, you must also update the names of the realized Logical Device specification, the realized Logical Device Configuration specification, and the realized container specification. See *Design Studio Concepts* for more information about Design Studio for Inventory realizations.

You can now use the Synchronize feature when you make subsequent configuration changes.

Related Topics

Working with Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Synchronizing Conceptual Model Entities with Application Entities

Moving Entities and Data Elements to Different Schemas

You can move entities to a different folder in the same project or into a different project, and you can move data elements from one data schema to another. Design Studio updates all references with the new location.

To move data elements to a different schema:

1. Select an entity or data element from an editor data tree or from a Design Studio view.

   See "Design Studio Common Editor Tabs" for more information about editor data tree areas.

2. Right-click and select **Refactoring**, and then do one of the following:

   - Select **Move To** to move data elements.

The Open Resource dialog box appears, displaying a list of all data schemas in the workspace.

- Select **Move** to move entities.

  The Move Studio Model Entity dialog box appears, displaying a list of all open projects in the workspace.

3. Select a data schema or a project folder from the list and click **OK**.

   Design Studio displays all of the changes to be performed.

4. Deselect any changes to be performed to exclude specific instances from the refactoring.
5. Click **Finish**.

Related Topics

Renaming Entities and Data Elements

Changing Data Element Base Type References

You can change the base type for all data elements that reference that type.

To change the base type for data elements:

1. From an editor data tree area, select a data element.

   See "Design Studio Common Editor Tabs" for more information about editor data tree areas.

2. Right-click and select **Refactoring**, and then select **Replace With**.

   The Refactoring dialog box appears, displaying a list of all base types in the workspace.

3. Select a base type from the list and click **Next**.
4. Deselect any changes to be performed to exclude specific instances from the refactoring.
5. Click **Finish**.

   The data elements that derive from the base type are now updated.

Related Topics

Refactoring Entities and Data Elements

Making Data Elements Modular and Reusable

You can copy a data element, and any children, to the root level of the same data schema or to a different data schema. The new data element created is defined as the base type for the original element. For example, if you make a structure reusable, references to the original location do not change. However, Design Studio changes the original child structure to a reusable base type structure in the Data Dictionary. You can now use this structure as a base type from which you can derive other structures.

To make modular, reusable structures:

1. From an editor data tree area, select a data element.

   See "Design Studio Common Editor Tabs" for more information about editor data tree areas.

2. Right-click and select **Refactoring**, and then select **Make Reusable**.

The Open Source dialog box appears, displaying a list of all data schemas in the workspace.

3. Select a schema from the list and click **OK**.
4. Review all impacted entities.
5. Click **Next**.
6. Click **Finish**.

   Design Studio moves the child structure to the root level of the selected entity.

Related Topics

Refactoring Entities and Data Elements

Creating Data Structure Definitions from Existing Data Elements

You can create a data structure definition from an existing data element and add it to an existing model project.

To create a data structure definition from an existing data element:

1. From an editor data tree area, select a data element.

   See "Design Studio Common Editor Tabs" for more information about editor data tree areas.
2. Right-click and select **Refactoring**, and then select **Create Data Structure Definition**.

   The Data Structure Definition wizard appears.
3. Select the model project where you want to add the data structure definition.
4. (Optional) Click the **Extends** field **Select** button.

   In the Matching items area, select a data structure definition that you want the new data structure definition to extend, or click **New** to create a new data structure definition that you want this data structure definition to extend.
5. In the **Name** field, enter a name for the data structure definition.

   The name must be unique among data structure definition entities.
6. (Optional) Select a location for the data structure definition.

   By default, Design Studio saves the data structure definition entity to the root level of the project folder. You can enter a folder name in the **Folder** field or select a location different from the default if you want to create additional subfolders. To select a different location:

   a. Click the **Folder** field **Browse** button.
   b. Navigate to the directory in which to save the entity.
   c. Click **OK**.
7. Click **Finish**.

   The new data structure definition entity appears under the model project in the Studio view.

Related Topics

About Data Structure Definitions

Creating Data Structure Definitions

Data Structure Definition Editor

Referencing New Base Types for Unresolved Data Elements

When you must manually resolve references that Design Studio cannot automatically resolve, you can reference new base types for multiple unresolved element references in entities, groups of entities, and projects.

For example, when working in the Studio Projects view, you can resolve all unresolved data elements in all Order entities that appear in a project's Order folder.

> **Note:**
>
> A data element is considered unresolved when Design Studio can no longer automatically resolve the reference to its base type. For example, if data element A in an Order entity references (as the base type) data element B in a model project data schema, and if element B is moved or deleted, data element A is considered unresolved.

To reference new base types for unresolved data elements:

1. From the Solution view context menu or from the Studio Projects view context menu, select and right-click:
   - A Project entity (any Project entity other than an Environment Project entity)
   - An Order entity
   - A Mapping Rule entity
   - Any entity that can contribute to the Data Dictionary
   - Any folder that contains groups of entities that can contribute to the Data Dictionary

   The context menu appears.
2. Do one of the following:
   - If you are working in the Solution view, select **Refactoring** and then select **Resolve Node**.
   - If you are working in the Studio Projects view, select **Resolve Node**.

   The Resolve Studio Model Elements dialog box appears.

> **Note:**
>
> The **Resolve Node** menu option is not available when you select:
>
> - An OSM project with any other type of project.
> - An OSM Order entity or a Mapping Rule entity with any other entity. For example, the menu option is not available if you select one Order entity and one Data Schema entity, or if you select one Mapping Rule entity and one Data Schema entity.
> - Any combination that includes an OSM Order entity and a Mapping Rule entity.

3. (Optional) In the **Name** field, enter the name or the partial name of an entity and click the **Filter** button.

   Design Studio filters the table to include only those elements that meet the filter criteria.

4. Select an entity in the table.

5. In the **Type** field, click **Select**.

   The Select Source Element to Resolve dialog box appears.

6. Select a new base element from the list.

   If a base element does not appear in the list, you may need to deselect the **Filter Project Dependencies** option.

7. Click **OK**.

   The new base type appears in the **Type** column.

8. (Optional) Select another entity in the table, click **Select** in the **Type** field and select a new base type.

   Repeat these steps until you select base types for all unresolved elements in the table.

9. Click **Finish**.

Related Topics

Refactoring Entities and Data Elements

Design Studio Refactoring Menu

Design Studio Refactoring Menu

Use the Design Studio **Refactoring** menu to propagate data model changes across a solution without sacrificing model integrity.

You can access the **Refactoring** menu from Design Studio editor data trees and from some Design Studio views. You can override the settings used to configure the behavior of Design Studio when refactoring data elements and entities. See "Refactoring Preferences Page" for more information.

Option	Use
Rename	Select to rename an entity or data element. An entity or data element can be renamed when: • The entity is writable. See "Defining Entity Read-Only Properties" for more information. • The entity or element exists in a project that is unsealed. See "Unsealing Projects" for more information. • A single entity or data element is selected in the view or editor. Before renaming the entity or element, Design Studio displays a list of the impacted references, selected for rename by default. If you exclude a reference from the rename process, you must resolve that data element reference. After renaming, all selected references and extensions to the entity are updated. See "Renaming Entities and Data Elements" for more information.

Option	Use
Remove from Workspace	Select to delete data elements from the workspace. You can remove elements from the workspace when: • The element is writable. See "Defining Entity Read-Only Properties" for more information. • The element exists in a project that is unsealed. See "Unsealing Projects" for more information. Before removing the element, Design Studio displays a list of all changes to be performed. After removal, all references and extensions to the deleted element are updated. **Note:** Carefully consider the impact when removing from the workspace data elements that impact multiple systems.
Create Data Structure Definition	Select to create a data structure definition from an existing data element. See "Creating Data Structure Definitions" for more information.
Move To	Select to move a global data element to a different data schema. You can move elements when: • The source and target schemas are writable. See "Defining Entity Read-Only Properties" for more information. • The source and target schemas exist in projects that are unsealed. See "Unsealing Projects" for more information. Before moving the element, Design Studio displays a list of all changes to be performed. After moving, all references and extensions to the elements are updated. When moving structured elements, all child elements are moved to the target schema. See "Moving Entities and Data Elements to Different Schemas" for more information.
Move	Select to move an entity to a different folder in the same project or to a different project. This action is available from the Solution view, the Studio Projects view, the Structure view, and the Outline view. You can move an entity when: • The entity is writable. See "Defining Entity Read-Only Properties" for more information. • The entity exists in a project that is unsealed. See "Unsealing Projects" for more information.

Option	Use
Resolve	Select to reference a new base type for a data element that Design Studio cannot automatically resolve. For example, consider that you have data element A that references data element B as the base type. If element B is moved or deleted, data element A is considered unresolved. You can resolve elements when: • The entities that contain the source and target elements are writable. See "Defining Entity Read-Only Properties" for more information. • The entities that contain the source and target elements exist in projects that are unsealed. See "Unsealing Projects" for more information. • The target elements do not exist in the base hierarchy of the selected element. • The target element is resolved.
Resolve Node	Select to reference new base types for multiple unresolved element references in entities, groups of entities, and projects. For example, when working in the Studio Projects view, you can use the **Resolve Node** menu option to resolve all unresolved data elements in all Order entities that appear in a project's **Order** folder. The **Resolve Node** menu option is available from the Solution view context menu and from the Studio Projects view context menu when you select: • Project entities (except for Environment projects) • Order entities • Mapping Rule entities • Entities that can contribute to the Data Dictionary • Groups of entities that can contribute to the Data Dictionary See "Referencing New Base Types for Unresolved Data Elements" for more information.

Option	Use
Replace With	Select to change the base type for all data elements that reference the selected base type (this action is applicable only to data schema elements). You can replace the base type for multiple data element references when: • The entities that contain the target elements are writable. See "Defining Entity Read-Only Properties" for more information. • The entities that contain the target elements exist in projects that are unsealed. See "Unsealing Projects" for more information. • The selected base element is a root element or a child simple element. • The selected base element is resolved. • The target elements do not exist in the base hierarchy of the selected base element. For example, you can use this action when consolidating data models, as it enables you to replace proprietary types with common types.
Copy Schema Element To	Select to copy a structured or simple data element to another data schema. When copying structured elements, all child elements are copied to the target schema. Elements are copied to the target schema as global elements. You can copy child structures and child elements (independent of the parent element) to create new global elements.
Make Reusable	Select to copy a data element (and any children, if applicable) to the root level of the same entity or to a different entity. The new data element created is defined as the base type for the original element (at the original location). For example, consider that a structure **employee** includes child structure **address**. If you make **address** reusable, order template and atomic action references to the original location **employee\address** do not change. However, the original child structure **address** is now of type **address** and a reusable base type structure **address** is created in the Data Dictionary. You can make an element reusable when: • The source and target entities are writable. See "Defining Entity Read-Only Properties" for more information. • The source and target entities exist in projects that are unsealed. See "Unsealing Projects" for more information. • The data element name must remain unique in the target entity. See "Making Data Elements Modular and Reusable" for more information.

Refactoring Preferences Page

Use the Refactoring Preferences page to override the settings used configure the behavior of Design Studio when refactoring entities and data elements.

For each refactoring menu action, select one of the following values:

Field	Use
Prompt	Select to instruct Design Studio to alert you when refactoring data elements associated with entities and references. You can do one of the following: • Make the entity or reference writable and then complete the refactoring action. • Leave the entity or reference as read-only, and cancel the refactoring action.
Always	Select to instruct Design Studio to automatically perform refactoring actions against data elements even when the elements are associated with read-only entities. In this scenario, Design Studio automatically changes associated read-only entities to writable. Design Studio prompts you before making read-only references writable.
Never	Select to prevent users from refactoring data elements associated with references (defined as writable or read-only) or read-only entities.

Related Topics

Refactoring Entities and Data Elements

Working with Design Patterns

In Design Studio, design patterns are wizards that can be used to create preconfigured entities (and the relationships among the entities) in a workspace. Design patterns automate complex, repeatable tasks, and enable team members with varying levels of skill to complete these tasks using a wizard interface. Your teams can use design patterns to reduce errors, simplify modeling, and increase productivity. Solution design teams install design patterns as Design Studio features and, using wizards, apply the patterns to their workspace. These wizards ensure compliance with the best practices and reduce the need for coding and complex configuration.

See "Applying Design Patterns" for more information. See the *Design Studio Developer's Guide* for more information about developing your own design patterns.

Applying Design Patterns

You apply design patterns using the Design Pattern wizard.

> **Note:**
>
> Design patterns can overwrite existing resources in the workspace, and the results of a design pattern can vary, depending on the current state of the workspace and the configuration details defined in the design pattern. See *Design Studio Developer's Guide* for more information about design patterns. See "Realizing Conceptual Model Entities into Application Entities" for information about using design patterns with conceptual model entities.

To apply a design pattern:

1. Do one of the following:
   - In Design Studio, from the **Studio** menu select **Design Pattern**.

     The Design Pattern wizard appears.
   - Select an entity or data element from a Design Studio view, right-click and select **Design Pattern**.

     The Design Pattern wizard appears, displaying the context in focus. For example, if you selected and right-clicked an Order entity in the Solution view, the wizard displays design patterns that are related to OSM orders. If no design patterns are related to the selected entity or data element, Design Studio displays design patterns that are similar. For example, if there are no design patterns registered for OSM orders, the wizard displays patterns related to OSM projects.
2. Do one of the following:
   - Select a design pattern from the list that appears in the dialog box.
   - Select a design pattern from your local file system.
3. Click **Next**.

   The Design Pattern Wizard Introduction page appears.
4. Read the information about the contents of the design pattern and click **Next**.

   The wizard displays the first group of fields defined in the design pattern. These fields may be populated with a default value if the design pattern was launched from a selected entity or data element. For more information about token groups, tokens, and input, see the *Design Studio Developer's Guide*.
5. Enter information into the wizard, as prompted.

   Each page in the wizard displays a set of fields defined in the design pattern. The Design Pattern wizard prompts you for all information required to apply the pattern.
6. Navigate through the wizard to completion, then click the **Summary** button.

   The **Summary** page displays the following:
   - All field values that you provided.
   - All resources to be copied to the workspace. The original name, new name, resource type, and target project displays for each resource. This section also indicates whether any resources with identical names exist in the workspace and whether the design pattern will overwrite the existing values. For example, if a resource with the same name and type exist in the workspace and the resource override value is defined as **true**, when the design pattern is applied the local resource file will be overwritten.

- All actions. The source entity, target entity, and action type appear for each action. This information includes whether the relationships and parameters in existing entities can be overwritten by the design pattern. Restricted actions that cannot be performed because of the configuration appear in the Restricted Actions section.
- All inputs. The input entity or element appears for each input.

7. Click **Finish**.

   After the wizard collects all of the information, Design Studio copies the resources generated by the design pattern into your workspace.

8. (Optional) Review the design pattern log file.

   The log file contains the same information that appears on the **Summary** page. Design Studio copies the log file to the workspace when the design pattern processing completes.

9. (Optional) In the Help view, use the cheat sheet to review the contents of the design pattern, or to complete any remaining manual steps.

   Design Studio may launch an Eclipse cheat sheet after the design pattern is applied, if a cheat sheet is included in the pattern. Design Patterns are built automatically after you apply them.

After you apply the design pattern, you can modify the configuration the design pattern generated and you can modify the resources the design pattern copied to your workspace. See the *Design Studio Developer's Guide* for more information about design patterns.

Modeling Data Using Context Menus

A common Design Studio context menu enables you to model data within some Design Studio application editors and views. This context menu contains actions specific to simple and structured data elements.

The context menu actions that are available depend on the editor or view in focus, and on the selection in the view or editor. For example, the list of context menu actions that appear in the Solution view are different than those that appear in the Studio Projects view, and the list of actions that appear when you have a structured data element selected in the OSM Order editor is different than the list that appears when no data element in the view is selected.

A view or editor may contain only a subset of the following actions:

Command	Use
Add Element, **Add Structure**, and **Add Characteristic**	Select to add simple or structured elements to an entity. See "Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions" and "Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions" for more information. See "Working with Characteristics" for more information about characteristics.
Add Child Element and **Add Child Structure**	With a structured data element selected, adds a new child element or child structure to the selected structured data element. See "Creating Simple Data Elements, Structured Data Elements, and Data Structure Definitions" for more information.

Command	Use
Add	Right-click an entity in the Solution view and select this option to associate the entity with another entity. Right-click the associated entity in the Solution view and select **Remove** *relationship* to disconnect the association. The type of relationship you add or remove is determined by the entity selected in the Solution view. For example, if you right-click a Customer Facing Service (CFS) specification in the Solution view, you can select **Add** to associate with the CFS specification a component, a realization entity, an action, and so forth.
Add *relationship* **Remove** *relationship*	Select **Add** *relationship* to associate two entities or to associate an element with entity. Select **Remove** *relationship* to disconnect the relationships. The type of relationship you add or remove is determined by the relationship folder selected in the Structure view or Outline view. For example, if you select the **Customer Facing Service** folder in the Structure view, you can select the **Add Customer Facing Service** command to add a relationship between a customer facing service specification that you select or create with the entity or element selected in the linked editor or view. These commands are available only in the Structure view and Outline view.
Delete	Deletes data elements and entities.
Expand **Collapse**	Expands and collapses a structured data element to display or hide all child elements of the structure. Or, you can do the following: • Select a structured data element and press the Left Arrow to collapse the selected structured data element, which hides all child elements of the structure. • Select a structured data element and press the Right Arrow to expand the selected structured data element, which displays all child elements of the structure.
Export	Select to export projects to archive files. See "Exporting Projects" for more information.
Import Product Specification	Select to import a Product specification into a PSR model project. This command is available in the Solution view only.
Move Up	Repositions a data element in the view by moving it up in the list.
Move Down	Repositions a data element in the view by moving it down in the list.
Refactoring	Enables you to propagate data model changes across a solution without sacrificing model integrity. See "Refactoring Entities and Data Elements" for more information.
Refresh	Refreshes the view.
Select Simple Data Element, **Select Structured Data Element**, and **Select Characteristic**	Select to add existing simple or structured elements to an entity. See "Working with Characteristics" for more information about characteristics.

Related Topics

About the Data Dictionary

Working with Tags

Tags are keywords that you can use to categorize data elements, instances of entities, and projects. Tags help you filter and search for data elements that are associated with specific Oracle Communications applications. Tags also help you control project visibility in a workspace.

A set of predefined tags is delivered with Design Studio. For example, you can tag a data element as a **Characteristic**, which means it is relevant to Design Studio for Inventory and Design Studio for Network Integrity data models. Or, you can tag a data element as **Control Data** to identify the data element as data that OSM requires to perform orchestration.

If a data element changes frequently, you can associate the data element with the **Changeable** tag. For example, customers often upgrade services for higher download speeds. You can tag the **DownloadSpeed** data element with the **Changeable** tag and track the data element's history for auditing purposes.

You can also create your own tags, and you can use these tags to search for and filter data elements. For example, you can search for tagged data elements using the Dictionary view Filter menu.

You can create your own tags and associate your tags to configuration items that you add to Service, Place, Logical Device, Logical Device Account, and Network entity Configuration specifications. Associating tags with configuration items that you add to Configuration specifications enables UIM users to search for tags using the Tags Search page. At run time, the tag is visible (on the Configuration Summary page) for all instances of entities that were created from the Configuration specification. These tags are also visible in the **getConfiguration** and **getConfigurationDifferences** web service operations, which are included in the **Service Fulfillment** web service. The **getConfiguration** web service operation response contains the tag information from the request. See "Associating Tags with Configuration Items" for more information.

See the following topics for more information:

- Creating Tags
- Tags Tab

Creating Tags

You can create your own tags that you can use to search for and filter data elements and to control project visibility in a workspace.

To create tags:

1. From the **Studio** menu, select **New**, then select **Model**, and then select **Tag**.

   The Tag wizard appears.

2. In the **Project** field, select a project in which to save the tag.

   You can create tags in Model projects only.

3. Enter a name for the new tag.

4. (Optional) Select a location for the tag.

   By default, Design Studio saves the entity to the root level of the project folder. You can enter a folder name in the **Folder** field or select a location different from the default if you want to create additional subfolders. To select a different location:

a. Click the **Folder** field **Browse** button.

b. Navigate to the directory in which to save the entity.

c. Click **OK**.

5. Click **Finish**.

   The Tag editor opens.

6. Enter a description for the new tag.

7. (Optional) In **Build Directives** area, click **Add**.

   A dialog box appears that includes the **Run-time Classification** tag, which is a system-provided tag that you can associate with the new tag. Associating your new tag with the **Run-time Classification** tag ensures that you can use the new tag to group configuration items in Design Studio and to enable UIM users to search for configuration specifications using tags. See "Associating Tags with Configuration Items" for more information.

8. Click **Save**.

Related Topics

Working with Tags

Tags Tab

Tag Editor

Use the Tag editor to define descriptions and annotations for your tags, and to associate your tags with system-delivered Inventory run-time tags.

Field	Use
Description	Enter a description of the tag. **Note:** The description that you enter here appears in the UIM run-time application Tag Summary page. See "Associating Tags with Configuration Items" for more information.
Runtime Directives	Annotate the tag with information relevant to solution modelers.
Build Directives	Click **Add** to associate your tag with the system-delivered **Run-time Classification** tag. Associating your tag with the **Run-time Classification** tag ensures that you can use the new tag to group configuration items in Design Studio and to enable UIM users to search for configuration specifications using tags. See "Associating Tags with Configuration Items" for more information. Click **Remove** to delete the association.

Related Topics

Working with Tags

Creating Tags

Data Schema Editor

Use the Data Schema editor to model data element information. Double-click any data schema entity in the Dictionary view to open the Data Schema editor.

> **Note:**
>
> When using data schemas created outside of Design Studio, ensure that the schema defines a target namespace.

When modeling data in the Data Schema editor, see the following topics:

- About the Data Schema Editor Context Menu
- Data Schema Editor Data Element Tab

About the Data Schema Editor Context Menu

The Data Schema editor context menu contains actions specific to simple and structured data elements. To access these actions, you right-click in the Data Schema editor Dictionary tree area. The context menu options that are available depend on the selection in the view. For example, the list of context menu options that appear when you have a structured data element actively selected is different than the list that appears when no data element in the editor is selected.

See "Modeling Data Using Context Menus" for more information.

Related Topics

Data Schema Editor

Data Schema Editor Data Element Tab

Use the Data Schema editor **Data Element** tab to view a hierarchical representation of data elements in the schema, and to access a common set of subtabs that enable you further refine entities and data elements.

When modeling data in the Data Schema editor **Data Element** tab, see the following topics:

- Details Tab or Attributes Tab
- Enumerations Tab
- Tags Tab
- Settings Tab
- Usage Tab
- Notes Tab

Field	Use
Type	Select a specific data type to limit the nodes that appear in the Dictionary area to structured data elements or simple data elements.
Filter	Search for specific data nodes, such as a specific address or person. When using the **Filter** field, consider the following: • The filter is not case sensitive. • Use a space between terms to filter for multiple terms. • Add OR between terms to filter for either one or the other. For example, entering **32 OR aa** returns occurrences of **32** or **aa**. • Variations of a word are returned. For example, a search query for **ID** returns **productID**, **customerID**, and **orderID**. • Use a hyphen (-) before a second filter term to omit specific variations of words (ensure there is a space before the hyphen). For example, a filter for **ID -product** returns **customerID** and **orderID**, but not **productID**.
Clear Filter button	Remove all criteria from the **Filter** field.

Related Topics

Data Schema Editor

About the Data Schema Editor Context Menu

Data Structure Definition Editor

Use the Data Structure Definition editor to model structured data element information.

When modeling data in the Data Structure Definition editor, see the following topics:

- Data Structure Definition Editor Data Elements Tab
- Data Structure Definition Properties Tab

Data Structure Definition Editor Data Elements Tab

Use the Data Structure Definition editor **Data Elements** tab to view a hierarchical representation of data elements in the data structure definition, and to access a common set of subtabs that enable you further refine data elements.

When modeling data in the Data Structure Definition editor **Data Elements** tab, see the following topics:

- Details Tab or Attributes Tab
- Enumerations Tab
- Tags Tab
- Usage Tab
- Notes Tab

Data Structure Definition Properties Tab

Use the Data Structure Definition **Properties** tab to set values that determine how a data structure definition is defined and used in a Model project.

Field	Use
Extends	You can select an existing data structure definition to extend this data structure definition by clicking the **Select** button. To create a new data structure definition, click **Extends**. When you extend, details are inherited from the parent data structure definition, such as the following: Attributes (for example, simple and structured data elements), and data element details (for example, behaviors, significance, key, default value, and so on).
Abstract	Select to indicate that this data structure definition can be extended by another data structure definition.
Final	Select to indicate that this data structure definition cannot be extended by another data structure definition.
Key Path	Define a key element for a data structure definition.

Related Topics

Data Structure Definition Editor

About Data Structure Definitions

Creating Data Structure Definitions

Creating Data Structure Definitions from Existing Data Elements

5

Working with Conceptual Models

Conceptual models are high-level, abstract representations of service domains. They define the relationships between your commercial products, the services that they represent, the resources that are required to implement the services, and the actions that must be performed in a run-time environment to fulfill a service order request.

Conceptual models define how commercial products and technical services are related, and they enable you to associate the products that you sell with the technical services and resources that are required to fulfill orders.

Conceptual models include entities that represent components of a service (such as customer facing services, resource facing services, products, resources, and so forth), but that contain no application-specific information.

See *Design Studio Concepts* for more information.

When working with conceptual models, see the following topics:

- About Conceptual Model Entities
- Implementing Conceptual Models
- Designing Conceptual Models
- Realizing Conceptual Model Entities into Application Entities
- Synchronizing Conceptual Model Entities with Application Entities
- Conceptual Model Editors

About Conceptual Model Entities

A conceptual model includes the following entities that you configure to support the behavior for a specific domain:

- Customer facing services, which represent your services from a customer perspective. See "About Customer Facing Services" for more information.
- Resource facing services, which represent a technical view of a service. See "About Resource Facing Services" for more information.
- Resources, which represent the entities that are required to configure the service. See "About Resources" for more information.
- Products, which represent your commercial products. See "About Products" for more information.
- Locations, which represent physical locations. For example, in a Broadband service, the **DSL** RFS for a DSL service can be associated with a location to represent the address of the customer.
- Service actions, which are requests to update the design of a customer facing service, a resource facing service, or a resource. See "About Conceptual Model Actions" for more information.

- Technical actions, which are requests to downstream delivery systems to make changes in the network (the downstream delivery systems are represented by an application role). Technical actions are associated with resource facing services or resources. See "About Conceptual Model Actions" for more information.
- Action parameter bindings, which enable you to:
  - Identify the conceptual model entities, such as a resource or a resource facing service, that contribute to the creation of technical actions.
  - Bind the attributes defined for a conceptual model entity or the entity itself (the source data) to parameters defined for technical actions (the target data).

  See "About Action Parameter Bindings" for more information.
- Domains, which are groups of entities and actions that you can use to organize and filter conceptual models. See "About Domains" for more information.

The following conceptual model entities help to define the model infrastructure and rarely require updates. You can extend these entities to meet the requirements of a specific deployment:

- Application roles, which represent types of downstream delivery systems that are responsible for specific types of delivery, such as activation, supply chain management, work force management, and so forth.

  See "About Application Roles" for more information.
- Functional areas, which are the logical layers of an installation and can be commercial, service, and technical layers. These layers are supported by an Order and Service Management order type or by an external order management system. See "About Functional Areas" for more information.
- Provider functions, which are processing components that perform a defined set of tasks based on their role in a solution. Design Studio includes the configuration for some provider functions, such as Calculate Service Order, Design and Assign, Calculate Technical Order, and Activation. See "About Provider Functions" for more information.
- Fulfillment patterns, which determine the high-level functions that are required to process an action or conceptual model entities. Fulfillment patterns include any number of fulfillment functions. See "About Fulfillment Patterns" for more information.
- Fulfillment functions, which represent the work to be performed against an action. Fulfillment functions can be augmented with conditions to determine whether the function is to be performed against an action. See "About Fulfillment Functions" for more information.

About Customer Facing Services

Customer facing services represent the commercial view of the services that you provide to your customer (a service represents the way that a product is realized and delivered to a customer).

You can use the same customer facing service to fulfill different but similar product offers. For example, the same **Broadband_Internet_Access** service can be used to fulfill a **Broadband** product and a **Broadband_Bandwidth** product. You map the data elements defined for customer facing services to data elements defined on products. Additionally, you associate customer facing services with resource facing services (as components). For example, you can associate with a CFS the resource facing services available to fulfill the service, such as **DSL** or **DOCSIS**.

See *Design Studio Concepts* for more information about customer facing services.

Related Topics

Designing Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Customer Facing Service Editor

About Resource Facing Services

Resource facing services describe how customer facing services are configured. For example, you can provision a customer facing service named **Broadband_Internet_Access** using multiple resource facing services, such as **DSL**, **Fiber**, or **DOCSIS**. You determine the resource facing service used to provide the commercial-level services during service design.

Resource facing services are technology-specific; however, resource facing services are not specific to a vendor. They can include resources or finer-granulated resource facing services. Resource facing services are represented in Design Studio for Inventory as Service specifications and as Service Configuration specifications.

See *Design Studio Concepts* for more information about resource facing services.

Related Topics

Designing Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Resource Facing Service Editor

About Resources

Resources define the technical components of a solution. A resource is a specific object in the network and in the inventory that can be consumed, referenced, or shared by a service when provisioning a resource facing service. Resources can be physical, such as a port, or logical, such as bandwidth. Examples of resources include IP addresses, VoIP phones, and DSLAM ports.

Resources are realized as Design Studio for Inventory resource entities. See *Design Studio Concepts* for more information about resources.

Related Topics

Designing Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Resource Editor

About Products

Products are entities that represent something that your business sells. A product type defines a set of product characteristics, validation rules, and relationships. For example, you might create products for **Broadband**, **Broadband_Bandwidth**, and **Email** products.

You can create products in Design Studio or (if your products exist in a separate product catalog) you can import products into Design Studio. After you create or import products, you can create or review the associated attributes in the Product editor.

See *Design Studio Concepts* for more information about products.

When working with products, see the following topics:

- Importing Products
- Product Editor

About Locations

Locations define geographic references that are relevant to services or resources. Locations can be specific places, such as a residence or a business, or more general places, such as a city. Locations are realized as Design Studio for Inventory Place specifications.

See *Design Studio Concepts* for more information about locations.

Related Topics

Designing Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Location Editor

About Conceptual Model Actions

Design Studio includes the definitions for two action families, service and technical:

- Service Actions are the signatures of design operations that apply to customer facing services, resource facing services, or to resources. Actions are run at run-time to initiate design and assign activities. Service actions are grouped into families that represent the range of operations that can be invoked on the entity, and each action consists of a specific set of parameters.
- Technical actions are requests to downstream delivery systems to make changes in the network (the downstream delivery systems are represented by an application role). Technical actions are associated with resource facing services or with resources.

Conceptual model entities are the targets, or subjects, of actions. You associate actions with entities to indicate that the action or group of actions can be performed against the associated entity.

You can create your own actions, or your can configure Design Studio to create actions automatically when you create new conceptual model entities. Actions that Design Studio create automatically are considered mandatory.

Customer facing services are associated with service actions only. When you create customer facing services, Design Studio automatically creates a mandatory service action and associates the service action with the customer facing service. The service action inherits all of the data elements defined on the customer facing service. Also, the service action includes a set of default data elements that are inherited from the associated functional area. These default data elements are associated with the **Implicit Parameter** tag (in the Functional Area editor) and they are not editable.

You can associate resource facing services and resources with one service action or with multiple technical actions.

Conceptual model service actions are realized in Design Studio for Inventory projects as rulesets. Conceptual model technical actions are realized in Design Studio for ASAP project as service actions (CSDLs) or in Design Studio for Network Integrity projects as scan actions.

See *Design Studio Concepts* for more information about actions.

Related Topics

Designing Conceptual Models

Creating Actions

Configuring Actions

Realizing Conceptual Model Entities into Application Entities

Action Editor

About Application Roles

An application role represents a type of downstream delivery system that is responsible for a specific type of delivery, such as activation, supply chain management, work force management, and so forth. Design Studio includes a set of predefined application roles. You can create your own application roles using the Application Role editor.

When you create a new technical action, Design Studio prompts you to select an action type (service or technical) and an application role. Design Studio populates the Action editor **Action Codes** tab with the specialized action code names defined for the application role. A specialized action code is an action code that you rename to align a technical action with a downstream fulfilment system.

For example, you may need to rename the **Create** action code to **Activate** to better align with code defined in a downstream activation system. For a supply chain management system, you may need to rename the **Create** action code to **Ship**. You may need to rename the **Remove** action code to **Uninstall** to better align with a downstream workforce management system.

Additionally, the Application Role editor enables you to define multiple specialized action code names for each default action code. For example, a downstream fulfillment system may require multiple versions of the **Modify** action code. You can differentiate between these versions by defining two unique specialized action code names, such as **Change** and **Revise**.

Related Topics

Application Role Editor

About Conceptual Model Actions

Configuring Actions

Action Editor

About Action Parameter Bindings

Action parameter bindings represent the aggregate of an entity and its parameters in the context of an application role. Action parameter bindings enable you to map source data (a conceptual model entity, attributes defined for a conceptual model entity, and components defined for a conceptual model entity) to target data (parameters defined for technical actions).

The mapping information, or binding, is contained in an Action Parameter Binding (APB) entity. APB entities enable you to identify source data elements in a conceptual model and define how they contribute to the generation of technical action data elements.

For example, you may have a Network Address Template Resource entity defined with multiple technical actions. You can create an action parameter binding to map the source data from the

conceptual model (data defined on Network Address Template Resource entity or elsewhere in the conceptual model) to the data that is required by a set of technical actions.

At run-time, action parameter binding configuration facilitates the analysis required during the Calculate Technical Actions (CTA) provider function, enabling CTA to determine the differences between a requested service configuration and the current service configuration. After the delta is determined, CTA can identify the technical actions required to affect the necessary changes in the network.

See *Design Studio Concepts* for more information about how action parameter bindings contribute to the Calculate Technical Actions provider function. See the *Oracle Communications RSDOD Reference Solution Developers Guide* for more information about generating CTA metadata.

Related Topics

Creating Action Parameter Bindings

Action Parameter Binding Editor

About Domains

A conceptual model domain is a group of entities and actions that you can use to organize and filter conceptual models. For example, you can create a domain called Alcatel DSLAMs that contains the resources (the Alcatel devices) that can be used as a DSLAM. You can include conceptual model entities in multiple domains, and you can create a hierarchy of domains that include subdomains. Subdomains can decompose into smaller groupings (for example, into broadband products and broadband services). Domains can be used as subdomains, and subdomains can be shared across multiple domains.

You can filter the Solution view to display domains and view and navigate among only those entities that are associated with domains.

You do not convert Domain entities into application entities in Design Studio. Rather, Domain entities help you organize, filter, and navigate conceptual models.

See *Design Studio Concepts* for more information about domains.

Related Topics

Domain Editor

About Functional Areas

Functional areas are the logical layers of an installation and can be commercial, service, and technical layers. These layers are supported by an Order and Service Management order type or by an external order management system.

When configuring functional areas, you specify whether the functional area supports actions. If it does, you indicate which conceptual model entities support the type of actions defined by the functional area and whether these actions are multi-instance. For example, the **Service** functional area supports actions, but only on customer facing services, resource facing services, and resources. The **Technical** functional area supports actions also, but those actions can be associated only to resource facing services and resources.

You can define default data elements for service actions associated with a functional area. These elements are automatically tagged with the **Implicit Parameter** tag, which cannot be removed. When a service action is created, the default data elements defined in the Functional Area editor appear as read-only on the Service Action editor **Data Elements** page.

Functional areas are associated with a list of action codes. Each action code, such as **Remove**, **Modify**, **Add**, and so forth, represents an action that can be performed on a conceptual model entity. When a new action of the type defined by the functional area is created, all of the action codes defined in the functional area are added to the new action. You can define the applicability of each default data element to the service action codes associated the functional area.

Related Topics

Creating Functional Areas

Realizing Conceptual Model Entities into Application Entities

Functional Area Editor

About the Service Functional Area

About the Service Functional Area

The **Service** functional area includes default data elements that are inherited by associated service actions. These default data elements are tagged with the **Implicit Parameter** tag, which cannot be removed. When a service action is created, the default data elements defined in the Service Functional Area editor appear as read-only on the Service Action editor **Data Elements** page. You can change the action code applicability settings on the Action editor **Data Map** tab.

The following data elements are defined on the **Service** Functional Area entity:

- **Subject_ID**: Defines the customer facing service or resource facing service instance. The value of this data element originates as an internal instance ID in the Unified Inventory Management.
- **ServiceAddress**: Defines where the service instance is geographically located. This structured data element contains string values that originate in the CRM system.
- **Customer_ID**: Defines the service instance. This value originates in the CRM system and is used to look up or create a corresponding subscriber instance (a UIM party).
- **Commercial_ID**: Defines the product or asset instance related to the service instance. This value originates in the CRM system as an Asset ID.

Related Topics

About Functional Areas

About Conceptual Model Actions

About Provider Functions

Provider functions are processing components that perform a defined set of tasks based on its role in a solution. Design Studio includes the configuration for some provider functions, such as Calculate Service Order, Design and Assign, Calculate Technical Order, and Activation.

Provider functions accept conceptual model entities or the actions that are associated with those entities as input and generate conceptual model entities or their actions as output. The output that provider functions generate is used as input by other, downstream provider functions. Calculate Service Order accepts Product entities as input and generates the service actions associated with customer facing service entities as output. This output is required as input by the Design and Assign provider function, which will generate output that is required by the Calculate Technical Actions provider function.

The types of components that you can define for a conceptual model entity are determined by the input and output types defined in all of the provider function definitions in a workspace.

See *Design Studio Concepts* for more information about provider functions.

Related Topics

Provider Function Editor

About Fulfillment Patterns

Fulfillment patterns define a set of high-level functions that can be performed to process an action or conceptual model entity. For example, a fulfillment pattern can define the functions that are required to process conceptual model entities and actions for provisioning, billing, or installation. In OSM, you associate fulfillment patterns with the processes that process the order items.

You can associate any conceptual model entity or action to a fulfillment pattern. For example, you can associate multiple products to one fulfillment pattern, which enables you to introduce new products with minimal configuration in Design Studio. Fulfillment patterns realize as OSM fulfillment patterns.

Fulfillment patterns contain any number of fulfillment functions, which are functional-level order components.

Related Topics

Fulfillment Pattern Editor

Realizing Conceptual Model Entities into Application Entities

About Fulfillment Functions

Fulfillment functions are operations that can be performed to process an order item. In the context of the conceptual model, the line item is an action or a product. After you create fulfillment functions, you associate them with fulfillment patterns. You also associate actions with fulfillment patterns, and the fulfillment pattern associated with any action determines which fulfillment functions can be associated with the action.

Fulfillment functions can be realized as OSM Order Component specifications. In OSM, line items are sent to a fulfillment pattern, which includes a set of order components that can be used to process the line item. The order components represent work that needs to be done against the line item. The fulfillment pattern determines which order components are to be used based on the conditions and dependencies defined in OSM. Each order component can have dependencies to other components (for example, one component may require that another be completed first).

Related Topics

About Conceptual Model Entities

Fulfillment Function Editor

Configuring Actions

Importing Conceptual Models from External Catalogs

You can import conceptual models from external catalogs using Exchange Format XML. The Exchange Format XML file is a consistent representation of Design Studio entities and external catalog system entities. You can import multiple entities across multiple projects with a single input XML file.

The import operations do not update sealed projects, read-only projects, or read-only entities.

You can import the following conceptual model entities, along with supporting data elements, data schemas, and model projects, into Design Studio:

- Product
- Customer facing service
- Resource facing service
- Resource
- Service action
- Technical action

You can perform a partial or complete import of the XML file.

> **Note:**
>
> **Partial** import is the default option.

A partial import:

- Creates new projects, entities, and elements if they are not present in the workspace
- Appends new information and updates the existing information for entities or elements
- Leaves existing information on entities and elements, if information is not included in the import file
- Renames the existing projects, entities, or elements

> **Note:**
>
> Partial import does not support the removal of information from existing entities or elements.

A complete import:

- Creates new entities or elements if they are not present in the workspace
- Replaces existing information on entities and elements with information that is provided in the import file
- Removes information from existing entities or elements if that information is not included for these entities or elements in the input file
- Renames the existing projects, entities, or elements

- Deletes the existing projects, entities, or elements
- Handles enumerations as follows:
  - Removes existing information of entities or elements if they are missing from the input file
  - Updates existing entities or elements with information that is modified in the input file
- Handles non-list or elements (for example, Copyright information, project name) as follows:
  - Defaults existing entities or elements with information that is missing from the input file
  - Updates existing entities or elements with information modified in the input file

> **Note:**
>
> Unique external identifiers must be provided for new instances of projects, entities, and elements that are imported.
>
> Identifiers for existing or imported projects, entities, and elements cannot be modified by import operations.

Importing Exchange Format Data from External Catalog

To import exchange format data from an external catalog:

1. Open **Design Studio**.
2. Ensure all changes made in workspace are saved.
3. Run a full build to ensure all model files are up-to-date.
4. Open **Studio Projects** or **Solution** view.
5. Open the Context menu by right-clicking the page.
6. Select **Import** > **Conceptual Models** > **Import Exchange Format**.
7. Click **Browse** and select the required exchange format file (.xml file).
8. (Optional) Select **Complete** for a complete import.

   > **Note:**
   >
   > For more information on Partial and Complete import, see "Importing Conceptual Models from External Catalogs".

9. Click **Next**.

   The Summary Page appears.
10. Check all the entities and elements that are imported, renamed, and deleted.

> **Note:**
>
> The Summary Page of Partial import does not display the deleted entities and elements. Use Complete import to check the deleted entities and elements.

11. Click **Finish**.

    A dialog box appears to open the generated import log.

12. Click **Yes** to open the log file, or click **No** to go back to the previous view.

13. Run Clean Build to update the references of entities and elements.

    See "Running Clean Builds" for more information.

Implementing Conceptual Models

The following procedure describes how to create a conceptual model in Design Studio and implement it in a run-time environment.

> **Note:**
>
> Before implementing conceptual models, you must first:
>
> - Generate the Common Model Base Data project into your workspace. See "Generating the Common Model Base Data Project" for more information.
> - Import any existing products. See "Importing Products" for more information.
> - Complete product-to-service mapping. See "Working with Mapping Rules" for more information.

When implementing conceptual models, you:

1. Create a Model project to contain the conceptual model.

   See "Creating Model Projects" for more information.

2. Define a dependency between the Model project and the Common Model Base Data project.

   See "Managing Project Dependencies" for more information.

3. Design the conceptual model.

   See "Designing Conceptual Models" for more information.

4. Realize the conceptual model.

   See "Realizing Conceptual Model Entities into Application Entities" for more information.

5. Synchronize the conceptual model entities and application entities throughout the design cycle.

   See "Synchronizing Conceptual Model Entities with Application Entities" for more information.

6. Build and package the application projects that contain the realized application entities.

   See "Building and Packaging Projects" for more information.

7. Deploy the application projects to a run-time environment.

   See "Deploying Cartridge Projects" for more information.

Generating the Common Model Base Data Project

To build a representation of a service domain in Design Studio, you must first generate the Common Model Base Data project. The Common Model Base Data project contains predefined rules and data for processing the entities and actions in your conceptual model, such as action codes, relationship rules, and entities that support conceptual modeling. This data is foundational to the conceptual model design required for service fulfillment solutions.

> **Note:**
>
> Only one Common Model Base Data project can exist in a single workspace.

To generate the Common Model Base Data project:

1. In the Solution view, right-click and select **Design Pattern**.

   The Design Pattern wizard appears.

2. Expand the **Model Project** folder.
3. Select **Common Model Base Data**, and click **Next**.

   The Introduction page appears.

4. Read the information about the Common Model Base Data cartridge project, and click **Next**.

   The Select Project Name page appears.

5. In the **Base Project Name** field, enter a name that begins with **OracleComms_Model_**.

   The name that you enter here must include the prefix **OracleComms_Model_**.

6. Click **Summary**.

   The Summary page appears.

7. Review the information on the Summary page and click **Finish**.

   Design Studio creates a new project in the workspace. The project contains data and entities that you can use as a starting point for your conceptual model.

Importing Products

You import products into Design Studio when new products are added to your product catalog. See "About Importing Products from AIA Servers" for more information about importing products from an Application Integration Architecture (AIA) server.

To import products:

1. Define web service endpoints for the products that you must import.

   See "Defining Web Service Endpoints" for more information.

2. In the Solution view, right-click and select **Import**, and then select **Import Product**.

   The Import Product wizard appears.

3. From the **Address URL** list, select an endpoint.
4. (Optional) To include the parent specifications of the products you import, select **Import Parent Product**.
5. Click **Add**.
6. In the **Product Name** field, enter the name of the product to import.
7. Click **Next**.
8. (If prompted) Enter the user name and password required to access the web service endpoint and click **OK**.

   See "Clearing Web Service Security Credentials" for information on resetting the web service credentials if needed during a Design Studio session.

   The **Product Import result** page appears. This page includes a table that lists all of the products for import, the ID, and the status.
9. Deselect the **Import** check box for any products that you do not want to import.
10. In the **Data Dictionary** field, select the data schema into which you will add the product attributes.
11. In the **Project** field, select the project in which to include the product.
12. Click **Finish**.

Related Topics

Defining Web Service Endpoints

Clearing Web Service Security Credentials

Defining Web Service Endpoints

A web service endpoint is a URL where files or services can be accessed by client applications. In Design Studio, you define web service endpoints to specify the web service IP address (or the fully qualified domain name) and port where your products are located. This information is required to import products.

To define the endpoints of web services for importing products:

1. From the **Window** menu, select **Preferences**, then select **Oracle Design Studio**, and then select **Product Import Endpoints**.

   The Product Import Endpoints preference page appears.
2. In the Endpoints table, select **New**.
3. In the **Name** field, enter a descriptive name.
4. In the **Endpoint** field, enter the web service IP address (or the fully qualified domain name) and port where the product is located.
5. Select the **Secure Web Service** check box to require a user name and password for access.

   See "Clearing Web Service Security Credentials" for more information.
6. Create additional endpoints for each web service that is to be called by the Product Import wizard.

   Typically, one or two endpoints are defined.
7. Click **Apply**.

8. Click **OK**.

After you define the web service endpoints, you use them in the Import wizard when selecting which endpoint to use to import products.

Related Topics

Importing Products

Clearing Web Service Security Credentials

When Design Studio calls the web service where your product is located, it may be required to pass credentials for web service security. You specify whether the web service requires credentials when you define the web service endpoint in the Product Import Endpoints preference page. See "Defining Web Service Endpoints" for more information.

If you specify that the endpoint supports security, users are prompted for a user name and password the first time the web service is called during a Design Studio session. The credentials are stored for the duration of the Design Studio session. If the credentials change during a session, you can clear and reset the credentials during the session.

To clear and reset web service security credentials during a Design Studio session:

1. From the **Window** menu, select **Preferences**, then select **Oracle Design Studio**, and then select **Product Import Endpoints**.

   The Product Import Endpoints preference page appears.
2. Select the web service endpoint for which to clear credentials.
3. Click **Clear Credentials**.

   The next time the Product Import wizard calls this web service endpoint, you will be prompted to re-enter the credentials.

Related Topics

Importing Products

About Importing Products from AIA Servers

You import products from Application Integration Architecture (AIA) servers using the same process as when you import products from other types of servers. See "Importing Products" for more information.

AIA servers, however, enable you to pass into Design Studio enriched metadata when this metadata is defined for a product or for data elements associated with a product. When you import products from AIA servers, Design Studio:

- Displays languages defined for a product in the Product editor and supports language options in the **Display Name** and **Description** fields. See "Defining Language Preferences" for more information.
- Displays product start date and end date information on the Product editor **Properties** tab, if that information is defined for the imported product. See "Product Editor Properties Tab" for more information.
- Displays product version information in the Notes dialog box. See "Defining Entity Notes" for more information.

> **Note:**
>
> The documentation defined in the Notes dialog box is overwritten if you re-import the same product.

- Saves the version and start date information for data elements defined for a product on the Data Schema editor **Notes** tab. See "Notes Tab" for more information.
- Saves the attribute definitions defined in the source system as data elements in the target data schema.Displays the product key on the Product editor **Properties** tab, if that information is defined for the imported product. The **Key** field value contains the product name that is defined in the source system.

Related Topics

Importing Products

Designing Conceptual Models

To design a conceptual model, you:

1. Create the conceptual model entities.

   See "Creating Conceptual Model Entities" for more information.
2. Configure the conceptual model entities.

   See "Configuring Conceptual Model Entities" for more information.
3. Create the actions associated with the conceptual model entities.

   See "Creating Actions" for more information.
4. Configure the actions associated with the conceptual model entities.

   See "Configuring Actions" for more information.
5. Map source data elements in a conceptual model to target data elements in technical actions.

   See "Creating Action Parameter Bindings" for more information.

Creating Conceptual Model Entities

Conceptual models include the following core entities: customer facing services, resource facing services, resources, products, and locations. Conceptual models also include domain, action code, fulfillment pattern, provider function, and functional area entities.

You create all conceptual model entities using the same procedure.

To create conceptual model entities:

1. From the **Studio** menu, select **New**, and then select **Model**, then select the type of entity that you want to create.
2. In the **Project** field, select the project in which to save this entity.
3. (For core entities only) To select an existing entity as the base type for the new entity, in the **Extends** field click **Select** and navigate to the existing entity.

The new entity becomes a subtype of the entity that you select here, and it inherits the base type attributes.

> **Note:**
>
> The **Extends** field appears only for customer facing services, locations, products, resources, and resource facing services.

4. In the **Name** field, enter a name for the entity.

   The name must be unique among entity types in the same namespace.

5. (Optional) Select a location for the entity.

   You can enter a folder name in the **Folder** field or select a location if you want to create additional subfolders. To select a different location:

   a. Click the **Folder** field **Browse** button.

   b. Navigate to the directory in which to save the entity.

   c. Click **OK**.

6. Click **Finish**.

   The new entity appears in the Solution view.

   If you created a new conceptual model core entity, Design Studio automatically creates any mandatory actions for that conceptual model entity and associates the actions to the entity.

   Design Studio:

   - Defines the conceptual model entity as the subject of the action.
   - Defines the functional area of the action (the action type).
   - Adds the action codes based on those defined in the functional area.
   - Configures the action to inherit data elements from its subject if this configuration is defined in the functional area.

   The new entity and the associated actions appear in the Solution view.

Related Topics

Designing Conceptual Models

Configuring Conceptual Model Entities

Your business requirements will dictate the order in which you configure your conceptual model entities. For example, you might define the technical solution first, where you define the resource facing services and downstream resources before you define the customer facing services and products.

Because a service must be available before you can offer it to customers, another approach is to model customer facing services first, then resource facing services, then resources, and then products.

To configure conceptual model entities:

1. In the Solution view, double-click a conceptual model entity to open the entity in a conceptual model editor.

For example, if you double-click a CFS entity, Design Studio opens the CFS entity in the Customer Facing Service editor.

2. On the **Data Elements** tab, define the simple and structured data elements required by the entity.

   For example, a CFS might require data elements that represent the customer ID, the customer location, the asset integration ID, and so forth.

3. On the **Components** tab (in the CFS, RFS, and Resource editors) and on the **Derivation** tab (in the Product editor), associate the entity with conceptual model components.

   You define relationships between conceptual model entities by adding components to conceptual model entities. See "Defining Conceptual Model Components" for more information.

4. On the **Properties** tab, specify the application entity to which the conceptual model entity will convert and associate a fulfillment pattern to the conceptual model entity.

   > **Note:**
   >
   > Products are not converted into application entities.

5. On the **Other Relationships** tab, associate conceptual model entities with any other entity, even if there is no implied relationship in the delivered service fulfillment definitions.

   This tab enables you to extend the delivered conceptual model.

6. On the **Categorization** tab, tag data elements and specify which data elements are persisted to a Customer Facing Service, a Resource Facing Service, a Resource, or a Location entity.

Related Topics

Designing Conceptual Models

Realizing Conceptual Model Entities into Application Entities

Conceptual Model Editors

Defining Conceptual Model Components

You define relationships between conceptual model entities by adding components to conceptual model entities. A component is a container that represents all of the viable configurations that can be defined for the relationship. For example, the **Broadband_Internet_Access** customer facing service can include the **Access** component. The **Access** component represents all of the resource facing services that can be used to deliver the **Broadband_Internet_Access** service.

See *Design Studio Concepts* for information about components and conceptual model entity relationships.

To define conceptual model components:

1. In the Solution view, double-click a conceptual model entity to open the entity in a conceptual model editor.

   For example, if you double-click a CFS entity, Design Studio opens the CFS entity in the Customer Facing Service editor.

2. Click the **Components** tab.

3. In the Components area, click **Add**.

   The Create Components Element dialog box appears.

4. In the **Name** field, enter a name for the component.

   Name your components to emphasize the role that the component serves (rather than naming the component with the name of associated entity). For example, if you intend to associate the **Broadband_Internet_Access** Customer Facing Service with the **Access_Node** Resource, you might name the component **ActivationTarget**. This naming convention enables you to replace the conceptual model entity defined for the component with a different conceptual model entity without needing to rename the role of the component.

   The component name must not be identical to any of the data elements defined for the same entity.

5. In the **Component Type** field, select an entity type.

   The types of components that are available in this field are determined by all associated provider functions. Associated provider functions are those that define the conceptual model entity as an input type.

   For example, consider that you are defining components for a Customer Facing Service entity, and that there exists in the workspace one provider function, named **DesignAndAssign**. And, consider that this provider function defines the Customer Facing Service entity as input and defines the Resource Facing Service entity and the Location entity as output.

   In this example, there are two available options in the **Component Type** field, the **RFS** value (resource facing service) and the **Location** value.

   If you need to add resources as components of a customer facing service (such as when modeling a Carrier Ethernet) you can edit the **DesignAndAssign** provider function definition to add the Resource entity as an output type.

6. In the **Options** field, define specific details about the components.

   Do one of the following:

   - Click **Add** to define the component option using a new conceptual model entity.
   - Click **Select** to define the component option using an existing conceptual model entity.

7. Define the option cardinality.

   The **Minimum** field indicates the minimum number of option instances, and the **Maximum** field indicates the maximum number of option instances.

   Enter **0** in the Minimum field (or select **Optional**) to indicate that the option is optional. Select **Unbounded** to define the maximum number of occurrences with no explicit limit.

8. In the **Relationship Type** field, define how the entity is related to the component.

   See *Design Studio Concepts* for information about the types of relationships you can define in between entities and components.

9. Click **OK**.

Related Topics

Conceptual Model Editor Components Tab

Configuring Conceptual Model Entities

Relationship Type Editor

Creating Functional Areas

You create new functional areas if you are not using the data delivered in the Common Model Base Data project or if you want to create action types in addition to the **Service** and **Technical** types delivered with Design Studio.

To create new functional areas:

1. From the **Studio** menu, select **New**, then select **Model**, and then select **Functional Area**.

   The Functional Area wizard appears.

2. Using the wizard, create a new Functional Area entity.

   See "Creating Conceptual Model Entities" for more information.

3. From the Solution view, double-click the Functional Area entity.

   The Functional Area entity opens in the Functional Area editor.

4. Click the **Data Elements** tab.

5. In the **Data Elements** area, define any default data elements for service actions associated with a functional area.

   When you add new data elements to a Functional Area entity, these elements are automatically tagged with the **Implicit Parameter** tag, which cannot be removed. When a service action is created, the default data elements defined in the Functional Area editor appear on the Service Action editor **Data Elements** page as read-only.

6. Click the **Action Support** tab.

7. If the functional area supports actions, select the **This Functional Area Supports Actions** option.

8. In the **Action Prefix** field, enter a value to define the naming convention of the functional area actions.

9. In the Supported Entity Types area, click **Add**.

   The Create Supported Entity Types Element dialog box appears.

10. In the **Action Entity Type** field, select the type of conceptual model entity that can be the subject of the actions supported in the functional area.

11. If the conceptual model entity requires an association with the type of action defined by the functional area, select **Mandatory**.

    When this option is selected for an entity in the Supported Entity Types table, Design Studio automatically creates an action entity and associates the action entity with any new entities that you create.

12. If multiple actions can be associated with the conceptual model entity, select **Allow Multiple Instances**.

    Conceptual model entities can have only one associated service action. Resource facing services and resources can be associated with multiple technical actions.

13. If the data elements defined on the conceptual model entity should be inherited by associated action families, select **Extend Subject by Default**.

14. Click **OK**.

    The conceptual model entity appears in the Supported Entity Types table.

15. Define the action codes associated with the functional area.

The action codes appear in the Functional Area editor **Action Codes** area. After you define the action codes, you can select a code and open it in the Action Code editor.

16. Click the **Data Map** tab and define the applicability of each default data element to the action codes associated the functional area.
17. Click the **Realization** tab.
18. Specify how the Functional Area entity is realized into an application entity.
19. Click **Save**.

Related Topics

Functional Area Editor

About Functional Areas

Defining New Action Codes

About Conceptual Model Actions

Defining New Action Codes

You create new action codes if you are not using the data delivered in the Common Model Base Data project or if you want to create action types in addition to the **Service** and **Technical** types delivered with Design Studio.

Action codes enable you to map product action order lines on incoming orders to service action order lines required to provision the order. See "Working with Mapping Rules" for more information about mapping rules.

To create new action codes:

1. From the **Studio** menu, select **New**, then select **Model**, and then select **Action Code**.

   The Action Code wizard appears.
2. Using the wizard, create a new Action Code entity.

   See "Creating Conceptual Model Entities" for more information.
3. From the Solution view, double-click the Action Code entity.

   The Action Code entity opens in the Action Code editor.
4. In the **Functional Area** field, click **Select**.

   The Select Functional Area dialog box appears.
5. Select a functional area from the list and click **OK**.
6. Click **Save**.

   The action code now appears in the Functional Area editor **Action Codes** area.

Related Topics

Creating Functional Areas

About Conceptual Model Actions

Creating Actions

You can create your own actions manually, or you can configure Design Studio to create actions automatically when you create new conceptual model entities.

When creating actions, see the following topics:

- Creating Actions Manually
- Creating Actions Automatically

Creating Actions Manually

To create new actions manually:

1. From the Solution view, select a conceptual model entity.
2. Right-click the entity and select **Add Action**.

   The Add Actions dialog box appears.
3. Click **New**.

   The Action wizard appears.
4. In the **Project** field, select the project in which to save the action family.
5. In the **Action Type** field, click **Select**, and then select an action type (a functional area) to which this action family belongs.

   For example, select **Technical** if you are associating an entity with a technical action. When you select an action type, Design Studio automatically populates the **Name** and **Location** fields. Oracle recommends that you use these naming conventions. To use a different name or location, deselect the **Use recommended name and location** option.
6. (For technical actions) In the **Application Role** field, select the downstream fulfillment system where the technical actions will be sent.
7. Click **Finish**.

   The new action appears in the Solution view.

   For service actions, Design Studio associates with the action all action codes defined on the specified functional area. For technical actions, Design Studio associates the default set of action codes and specialized technical actions defined for the application role. If the related functional area specifies that data elements are inherited by the action family, Design Studio adds the data elements defined for the conceptual model entity subject to the action parameter set.

Related Topics

Creating Actions

About Conceptual Model Actions

Creating Actions Automatically

You can configure Design Studio to create actions automatically when you create new conceptual model entities. Actions that Design Studio creates automatically are considered mandatory.

To create actions automatically:

1. In the Solution view **Category** field, select **Functional Areas**.

   Design Studio filters the Solution view to display Functional Area entities only.
2. Double-click a Functional Area entity to open the entity in the Functional Area editor.
3. Ensure that the **This Functional Area Supports Actions** option is selected.
4. In the **Action Prefix** field, enter the naming convention that Design Studio uses when creating new actions of this action type.
5. In the Supported Entity Types table, select the conceptual model entity for which Design Studio will automatically create actions when you create an entity of this type.
6. In the Supported Entity Type Details table, select **Mandatory**.
7. If you want the new action to inherit all of the data elements defined for the associated entity, select **Extend Subject by Default**.
8. Click **Save**.

   When you create a new conceptual model entity, Design Studio will create a new action automatically, and associate the new action to the entity.

Related Topics

Creating Actions

About Conceptual Model Actions

Configuring Actions

To configure actions:

1. From the Solution view, double-click an Action entity to open the entity in the Action editor.
2. On the **Data Map** tab, define the applicability of the data elements to the action.

   You can specify whether a data element is required by an action by defining applicability to specific action codes (for service actions) or by defining applicability to specialized aliases defined for the action codes (for technical actions). For example, you can specify whether a data element value must be supplied to or returned by each action in an associated action family.
3. Click the **Data Elements** tab and configure the data elements required by the action.

   The data elements that have applicability to the action appear on this tab. You cannot add or remove data elements from the Service Action editor **Data Elements** tab.
4. Click the **Action Codes** tab.
5. (Optional) Add or remove action codes.

   Do one of the following:

   - For service actions, add or remove action codes. The action codes that initially appear are those defined by the action type (or functional area). For example, when modeling service actions, the action codes default from the **Service** functional area.
   - For technical actions, add or remove action codes and define a corresponding specialized action name that is specific to the application role. The action codes that initially appear are those defined by the action type (or functional area). For example, when modeling technical actions, the action codes default from the **Technical** functional area.

> **Note:**
>
> You can add only those action codes that are included in the associated functional area. The functional area is specified in the **Action Type** field.
>
> You can define multiple specialized technical actions against a single action code. Each specialized technical action can define a unique parameter selection on the **Data Map** tab.

6. Click the **Properties** tab.
7. If the action should inherit the data elements defined for the associated conceptual model entity (the subject of the action), select **Extend Subject**.
8. In the **Target** field, enter the resource or resource facing service that represents an instance of an entity in the network against which the action runs.
9. In the **Key** field, enter a value to enable the Order and Service Management order transformation manager to identify order line items of this type at run time.
10. In the **Implementation System** field, define the system where the application entity is used.
11. In the **Realization Design Pattern** field, select a design pattern to realize the action.

    You can create your own design patterns if no patterns are available. See the *Design Studio Developer's Guide* for more information.
12. In the **Implementation Method** field, define how the action realizes.

    Service actions realize in Design Studio for ASAP projects as service actions. Technical actions can realize in Design Studio for Inventory projects as rulesets or in Design Studio for Network Integrity projects as scan actions.
13. If you want to synchronize the action and the realized application entity automatically, select the **Run Realization Design Pattern Automatically** option.
14. (Optional) To extend an existing action, click **Select** in the **Extends** field.

    When you extend one entity from another, the target entity inherits all of the data elements defined for the extended entity. This option is not available if you select the **Extend Subject** field.
15. Click **Save**.

Related Topics

Action Editor

About Conceptual Model Actions

Creating Action Parameter Bindings

Action parameter bindings enable you to bind attributes defined for conceptual model entities and components (the source data) to parameters defined for technical actions (the target data). See "About Action Parameter Bindings" for more information.

You create a single, reusable action parameter binding for a subject when the action parameter binding applies to the subject wherever the subject appears in the conceptual model. In this scenario, the subject entity and its child components contain all of or most of the data that is

required by technical actions and the run-time application uses this binding in every context in which the subject appears.

You create service-specific action parameter bindings when you want the binding to apply only when the subject appears in a specific service. In this scenario, the subject contains little or no source data. Rather, the source data required by the action family is defined in various data elements throughout the service.

> **Note:**
>
> Action parameter bindings do not automatically facilitate the metadata that the Calculate Technical Actions provider function requires at run-time. Rather, Design Studio enables you to complete the modeling necessary to facilitate the CTA processes. To automate the creation of CTA metadata, you must develop a CTA metadata generator. This generator explores the conceptual model configuration and generates the necessary metadata.
>
> You can leverage the Design Studio Exchange Format and create your own CTA metadata generator, or you can use the example that is included in the Oracle Communications RSDOD Reference Solution, which is available on the Oracle Technology Network.
>
> See the *Design Studio Developer's Guide* for more information about the Design Studio Exchange Format. See the *Oracle Communications RSDOD Reference Solution Developers Guide* for more information about generating CTA metadata.

To create a new action parameter binding:

1. In the Solution view, right-click the conceptual model entity (the subject) for which you want to create an action parameter binding.
2. From the context menu, select **Add**, then select **Add Action Parameter Binding**.

   The Add Action Parameter Binding dialog box appears.
3. Click **New**.

   The Action Parameter Binding wizard appears.
4. Accept the default values prepopulated in the **Subject** field.
5. In the **Context** field, do one of the following:
   - Accept the default to create a single, reusable action parameter binding for a subject when the action parameter binding applies to the subject wherever the subject appears in the conceptual model.
   - Click the drop down menu to select a context that is specific to a service, or enter a new context that is specific to a service.

     By default, the context is defined using the relative path of the subject, and indicates that the action parameter binding set is to be used wherever the subject is used. You can change the context if the action parameter binding set is applicable for a more specific context.

     You are not required to define the context at the root level of the service. For example, you can define the context at the resource level.
6. In the **Action Family** field, select the actions to include in the action family.

By default, all actions associated with the subject are included in the action parameter binding. Select **Select Specific Actions** to limit the action family to a subset of actions. For example, you might have technical actions to create, modify, and delete entries in the network address template. You can create a single action parameter binding for all of these technical actions, or you can limit the scope if you require alternate bindings for specific scenarios.

7. (Optional) In the **Name** field, edit the default name.

   This field displays the Action Parameter Binding entity name, which is automatically populated with the name of the subject entity and an **APB** suffix. You can edit the default value if you want to support multiple action parameter bindings for a single subject; for example, if you are creating an APB for the Network Address Template Resource entity, you might want to create two bindings, one named NetworkAddressTemplate_CustomerEdge and the other named NetworkAddressTemplate_ProviderEdge.

8. Click **Finish**.

   The Action Parameter Binding entity is created. Design Studio opens the entity in an editor.

9. Click **Save**.

Related Topics

About Action Parameter Bindings

Action Parameter Binding Editor

Realizing Conceptual Model Entities into Application Entities

Conceptual model entities represent abstractions of services, so you cannot deploy conceptual model entities to run-time environments. Rather, you convert conceptual model entities into detailed application entities. This conversion process is called realization, because the conversion starts with an abstract conceptual model entity and creates a real application entity. Application entities realize conceptual model entities. For some conceptual model entities, you can use delivered design patterns to convert conceptual model entities into application entities.

See *Design Studio Concepts* for more information about realizing conceptual model entities.

When realizing conceptual model entities, see the following topics:

- Setting Up Conceptual Model Entity Realization
- Realizing Conceptual Model Entities
- Realizing Conceptual Model Entities Manually

Setting Up Conceptual Model Entity Realization

To set up conceptual model realization:

1. In the Solution view, double-click a conceptual model entity to open the entity in a conceptual model editor.

   For example, if you double-click a CFS entity, Design Studio opens the CFS entity in the Customer Facing Service editor. Clicking an Action entity opens the Action editor.

2. Click the **Properties** tab.

3. In the **Implementation System** field, specify the system where the application entity is used.

4. In the **Realization Design Pattern** field, select the Design Studio design pattern that you want to run to realize the conceptual model entity.
5. In the **Implementation Method** field, specify how the conceptual model entity is realized
6. Select the **Run Realization Design Pattern Automatically** option.
7. Click **Save**.

Related Topics

Realizing Conceptual Model Entities into Application Entities

Realizing Conceptual Model Entities

The first time that you realize a conceptual model entity, you must manually enter the data required by the design pattern. The design pattern prompts you for the information it requires to realize the selected entity and to build out the application configuration in the workspace. After you run the design pattern once, Design Studio saves the values that you supply in a synchronization record. It uses these values to re-run the design pattern automatically, if you have defined the conceptual model entity to run automatically.

> **Note:**
>
> Configuration generated by Design Studio on realized entities is overwritten when you re-run an associated design pattern.
>
> See "Renaming Conceptual Model Entities and Realized Application Entities" for information about renaming conceptual model entities after you have realized application entities.

To realize conceptual model entities:

1. In the Solution view, right-click a conceptual model entity and select **Synchronized Realization**, and then select **Run Realization Design Pattern**.

   The design pattern that you specified for the entity (on the conceptual model entity editor **Properties** tab) appears.
2. Complete the information required by the wizard, then click **Finish**.

   Design Studio creates the appropriate configuration in your workspace and saves the design pattern values in a synchronization record. Design Studio uses the values in the synchronization record to re-rerun the design pattern automatically (if you have defined the conceptual model entity to run automatically) and to synchronize the conceptual model entity and the realized application entity.

   After Design Studio generates the realization, it adds the realized entity to the Solution view as a child of the conceptual model entity. Double-click the realized entity to open and modify the entity in the appropriate editor.

> **Note:**
>
> Changes that you make to realized entities are not propagated to the associated conceptual model entity. Oracle recommends that, when changes are required, you make changes to the conceptual model entity and then re-run the design pattern to update the realized entity.

Related Topics

Realizing Conceptual Model Entities into Application Entities

Realizing Conceptual Model Entities Manually

If the default design pattern associated with a conceptual model entity is not suitable to your business needs, you can realize conceptual model entities manually.

To realize conceptual model entities manually:

1. In the Solution view, right-click a conceptual model entity and select **Add**, and then select **Add Realization**.

   The Add Realization dialog box appears.

2. Do one of the following:

   - Select an existing application entity to realize the conceptual model entity and click **OK**.
   - Click **New** to create a new entity to realize the conceptual model entity.

   Design Studio creates the application entity in your workspace and adds the realized entity to the Solution view as a child of the conceptual model entity. Double-click the realized entity to open and modify the entity in the appropriate editor.

> **Note:**
>
> The synchronization features are not available for entities that you realize manually.

Synchronizing Conceptual Model Entities with Application Entities

You synchronize conceptual model entities with application entities to ensure that the configuration of the application entities and the related conceptual model entities remains aligned. See *Design Studio Concepts* for more information about synchronizing conceptual model entities with application entities.

> **Note:**
>
> You must run the realization design pattern manually at least once before you can synchronize conceptual model entities and realized application entities. See "Realizing Conceptual Model Entities" for more information.
>
> See "Renaming Conceptual Model Entities and Realized Application Entities" for information about renaming conceptual model entities after you have realized application entities.

To synchronize conceptual model entities with realized application entities:

1. In the Solution view, right-click a conceptual model entity, select **Synchronized Realization**, and then select one of the following:
   - **Synchronize**: Use to run the design pattern that you specified for the conceptual model entity (on the conceptual model entity editor **Properties** tab) in the background, and to add the configuration defined by the design pattern to the workspace.
   - **Synchronize All**: Use to instruct Design Studio to analyze all relationships between the conceptual model entity and the child entities and to run design patterns for all conceptual model entities that support realization (and for those entities that have valid synchronization records).
2. Review the configuration that the design pattern added to the workspace.

   You can filter the Solution view to display all realized application entities of a specific type. For example, you can filter for service realizations to list all realized Design Studio for Inventory Service specifications and their associated Configuration specifications.
3. Ensure that the data saved in the synchronization records is correct.

   Do the following:
   a. From the Package Explorer view, expand the **Model** project folder in which you saved your conceptual model entities.
   b. Expand the **synchronizationRecords** folder.
   c. Double-click any synchronization record to open the record in the Synchronization Record editor.

Conceptual Model Editors

When working with conceptual model editors, see the following topics:

- Conceptual Model Editor Common Tabs
- Product Editor
- Customer Facing Service Editor
- Resource Facing Service Editor
- Resource Editor
- Location Editor
- Action Editor
- Action Code Editor

- Application Role Editor
- Action Parameter Binding Editor
- Relationship Type Editor
- Domain Editor
- Functional Area Editor
- Provider Function Editor
- Fulfillment Pattern Editor
- Fulfillment Function Editor
- Conceptual Model Unit of Measure Editor
- Synchronization Record Editor

Conceptual Model Editor Common Tabs

Conceptual model editors use a set of common Design Studio tabs to enable you to configure conceptual model entities.

When working with common editor tabs, see the following topics:

- Conceptual Model Editor Data Map Tab
- Conceptual Model Editor Data Elements Tab
- Conceptual Model Editor Components Tab
- Conceptual Model Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Conceptual Model Editor Data Map Tab

Use the **Data Map** tab to define the applicability of data elements to the actions in an action family. For example, you can specify whether a data element value must be supplied to or returned by each action in an associated action family.

Field	Use
Element	Displays all of the top-level data elements defined on and inherited from the associated conceptual model entity. For service actions, this column also includes default data elements that are defined on and inherited from the associated functional area. For an action code, leave this row blank if the data element is not required by the action.
Persisted	Specify whether the data element is persisted on the conceptual model entity subject. For service actions, only persisted data elements are added to Inventory specifications when you realize the conceptual model entity. See "Conceptual Model Editor Categorization Tab" for information.

Field	Use
Action Codes or **Specialized Action Code**	The column headings display all action codes (for service actions) or specialized action codes (for technical actions) associated with the conceptual model entity. For each action code and data element combination, select one of the following values: • **Required In**: the data element value must be supplied for the associated action. • **Required Out**: the data element value must be returned by the associated action. • **Required In/Out:**: the data element value must be supplied for and returned by the associated action. • **Optional In**: the data element value can be optionally supplied for the associated action. • **Optional Out**: the data element value can be optionally returned by the associated action. • **Optional In/Out**: the data element value can be optionally supplied for or optionally returned by the associated action. • Blank: the data element is not applicable to the action.

Related Topics

Conceptual Model Editor Data Elements Tab

About Functional Areas

About Conceptual Model Actions

Conceptual Model Editor Data Elements Tab

Use the **Data Elements** tab to add simple and structured data elements to a conceptual model entity from a data schema. For example, a customer facing service that you model for a broadband service might include the **UploadSpeed** and **DownloadSpeed** data elements to represent the service speeds requested by a customer order.

The **Data Elements** area data tree displays a hierarchical representation of data elements modeled for the entity.

Conceptual model editors include context menu actions that you access by right-clicking in the **Data Elements** area data tree. The context menu options that are available depend on the selection in the view. See "Modeling Data Using Context Menus" for more information.

When adding data elements to a conceptual model entity, see the following topics:

- Details Tab or Attributes Tab
- Enumerations Tab
- Tags Tab
- Usage Tab
- Notes Tab
- Conceptual Model Editor Transformation Tab

Conceptual Model Editor Transformation Tab

Use the **Transformation** tab to determine which data elements used by Order and Service Management order transformation manager support data propagation. Data propagation impacts performance.

Field	Use
Supports Forward Propagation	Select to propagate data changes from the original order item to the transformed order item. Data element changes on the original order item are transformed according to the mapping configuration in the OSM mapping rule, and the result is populated to the transformed order item.
Supports Reverse Propagation	Select to propagate data changes from the transformed order item to the original order item. Data element changes on the transformed order item are transformed in reverse according to the mapping configuration in the OSM mapping rule and the result is populated to the original order item. For example, if there exists unit-of-measure mapping between data elements on the original and transformed order items, with the original value being in hours and the transformed value being in minutes, the changed value on the transformed order item will be divided by 60 before being updated on the original order item.
Key Path	Define a key element for a multi-instance node. This value is used during order transformation management. See "Working with Mapping Rules" for more information. This field appears when you select a structured data element in the Data Elements area.

Related Topics

Conceptual Model Editor Data Elements Tab

Conceptual Model Editor Components Tab

Use the **Components** tab to associate a conceptual model entity to other conceptual model entities. For example, on the Resource Facing Service editor **Components** tab, you associate with the RFS all resources that are needed to configure the associated service.

When modeling data in the **Components** tab, see the following topics:

- Conceptual Model Editor Components Tab Components Area
- Conceptual Model Editor Components Tab Component Detail Area

Conceptual Model Editor Components Tab Components Area

Use the **Components** area to associate the conceptual model entity with other conceptual model entities.

Field	Use
Remove	Click to remove the relationship between the conceptual model entities.
Add	Click to create a relationship between conceptual model entities. See "Defining Conceptual Model Components" for more information.

Related Topics

Conceptual Model Editor Common Tabs

Conceptual Model Editor Components Tab Component Detail Area

Use the **Component Detail** area to define the component options, the number of allowable instances, and the type of relationship the conceptual model entity has with each component.

Details Tab

Field	Use
Name	Edit the name of the component selected in the **Components** area. Name your components to emphasize the role that the associated entity serves (rather than naming the component to be the name of the associated entity). This convention enables flexibility and reuse.
Component Type	Displays the type of entity that the selected component represents. The component types that are available is determined by the configuration of the provider function and functional areas with which the entities are associated. You can change the relationship type, as needed.
Options	Displays the conceptual model entities that the component can represent. For example, the Internet Access Resource Facing Service component can have multiple options defined, such as **DSL**, **Satellite**, **Fiber**, and so forth. Do one of the following: • Select a value and click **Open** to open the entity in an editor. • Click **Add** to create a new entity and associate the new entity with the component. • Select a value and click **Remove** to remove the association between the component and the entity. • Click **Select** to associate an existing entity to the component.

Field	Use
Minimum and **Maximum**	The **Minimum** field indicates the minimum number of option instances, and the **Maximum** field indicates the maximum number of option instances. Enter **0** in the **Minimum** field (or select **Optional**) to indicate that the option is optional. Select **Unbounded** to define the maximum number of occurrences with no explicit limit.
Relationship Type	Displays the type of association between the conceptual model entity and the component. The values in this field are defined in the Common Model Base Data project. See "Defining Conceptual Model Components" for more information.

Used By Tab

Use the **Used By** tab to review the projects and entities in which the entity is used.

Notes Tab

Use the **Notes** tab to annotate entities with descriptions or with other applicable information to support the entity. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML markup.

See "Notes Tab" for more information.

Related Topics

Conceptual Model Editor Components Tab

Conceptual Model Editor Properties Tab

Use the **Properties** tab to define how the conceptual model entity is realized, to associate fulfillment patterns to the conceptual model entity, and to extend a conceptual model entity from another conceptual model entity.

Field	Use
Extends	Extends the conceptual model entity from another conceptual model entity. When you extend one entity from another, the target entity inherits all of the data elements defined for the extended entity. See "Extending Design Studio Entities" for more information.

Field	Use
Implementation System	Define the system where the application entity is used. Select one of the following: • Select **None** if the conceptual model entity will not be realized as an application entity. You select this option, for example, if the conceptual model entity is intended for informational use only. • Select **Unified Inventory Management** if the conceptual model entity will be realized as an entity used in the Design Studio for Inventory application. • Select **Other System** if the conceptual model entity will be realized as an application entity that will be used in a system other than Unified Inventory Management. **Note**: The design patterns that are delivered with Design Studio and that realize the conceptual model entities add configuration to your workspace only if you define this field with the **Unified Inventory Management** value.
Realization Design Pattern	Select which design pattern converts the conceptual model entity into an application entity and creates the application entity configuration.
Implementation Method	Specify how the conceptual model entity realizes as an application entity. For example, a customer facing service realizes as a Service specification in a Design Studio for Inventory project. If you change the value in this field and re-run a Design Studio default design pattern, the design pattern does not delete any previously generated entities. See "Realizing Conceptual Model Entities into Application Entities" for information about how each conceptual model entity realizes.
Run Realization Design Pattern Automatically	Select to synchronize conceptual model entities and application entities automatically. When you select this option, Design Studio runs a design pattern automatically when you: • Save of the editor of the conceptual model entity. • Change the conceptual model entity using an option that you select from the context menu. • Manually synchronize or realize a direct child of the conceptual model entity. For example, when this option is selected, Design Studio automatically runs a design pattern for an RFS if you manually synchronize a child resource. See "Synchronizing Conceptual Model Entities with Application Entities" for more information.

Field	Use
Strict Configuration Checking	Deselect to relax the configuration warnings associated with entity realization. For example, consider that your model includes a conceptual model specification that is realized by a pre-existing Inventory Service specification and Service Configuration specification, and that these specifications include a configuration that violates the Design Studio model validation rules. You can deselect the Strict Configuration option to limit the validation severity to a warning.
Realized By	Displays the name of the application entity realized from the conceptual model entity. This field is blank if the design pattern selected in **Realization Design Pattern** has not yet run.
Fulfillment Pattern	Associate fulfillment patterns to the conceptual model entity. A fulfillment pattern provides an order and service management system with the steps required to orchestrate the service order at run-time. Do one of the following: • Click **Select** to associate an existing fulfillment pattern to the conceptual model entity. • Click **Add** to create a new fulfillment pattern and associate the new fulfillment pattern to the conceptual model entity. • Select a value and click **Open**, which opens the fulfillment pattern in the Fulfillment Pattern editor. • Select a value in the **Fulfillment Pattern** field and click **Remove** to remove the association between the fulfillment pattern and the conceptual model entity.
Key	Enter a value to enable the Order and Service Management order transformation manager to identify the payload of a line item at run time. The default value for this field is the name of the entity. You can define a different key value for an entity if your business processes require a naming convention less restrictive than then entity name requirements.

Related Topics

Conceptual Model Editor Common Tabs

Conceptual Model Editor Other Relationships Tab

Use the **Other Relationships** tab to associate conceptual model entities with any other entity, even if there is no implied relationship in the delivered service fulfillment definitions. This tab enables you to extend the delivered conceptual model.

When working with the **Other Relationships** tab, see the following topics:

- Other Relationships Tab Other References Area
- Other Relationships Tab Relationship Detail Area

Other Relationships Tab Other References Area

Use the **Other References** area to associate the conceptual model entity with other conceptual model entities.

Field	Use
Remove	Click to remove the reference between the entity and the selected component.
Add	Click to create a relationship between the entity and the selected component.

Related Topics

Conceptual Model Editor Other Relationships Tab

Other Relationships Tab Relationship Detail Area

Use the **Relationship Detail** area to associate the conceptual model entity with other conceptual model entities.

Details Tab

Use the **Details** tab to further define entity information.

Field	Use
Name	Edit the name of the target entity.
Option	Displays the conceptual model entities that the associated component can represent. Do one of the following: • Select a value and click **Open** to open the entity in an editor. • Click **Add** to create a new entity and associate the new entity with the component. • Select a value and click **Remove** to remove the association between the component and the entity. • Click **Select** to associate an existing entity to the component.

Field	Use
Minimum and **Maximum**	Define the entity cardinality. The **Minimum** field indicates the minimum number of times the entity can appear, and the **Maximum** field indicates the maximum number of times the entity can appear. Also, indicate whether the entity is **Required** or **Optional** or whether there can be multiple occurrences of the entity. To define a range, do one of the following: • Define a range with at least one occurrence. Select the value **1** in the **Minimum** field and select **Unbounded** (no explicit limit) in the **Maximum** field. • Define a range with no required minimum number of occurrences. Select the value **0** in the **Minimum** field and select **Unbounded** in the **Maximum** field.

Used By Tab

Use the **Used By** tab to review the projects and entities in which the target component is used.

Notes Tab

Use the **Notes** tab to annotate entities with descriptions or with other applicable information to support the entity. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML markup.

See "Notes Tab" for more information.

Related Topics

Conceptual Model Editor Other Relationships Tab

Conceptual Model Editor Categorization Tab

Use the **Categorization** tab to tag data elements with Inventory-specific tags and to specify which data elements are persisted to a Customer Facing Service, a Resource Facing Service, a Resource, or a Location entity.

Field	Use
Element	Displays all of the data elements defined for the conceptual model entity on the **Data Elements** tab.
Characteristic	Select **Yes** to tag the data element as a characteristic, which means it is relevant to Design Studio for Inventory and Design Studio for Network Integrity data models. You must tag a data element with the **Characteristic** tag (in the data schema) to add it to an Inventory Service specification during realization.

Field	Use
Changeable	Select **Yes** if the data element changes frequently or if you need to track the life cycle of a service. For example, customers often upgrade services for higher download speeds. You can tag the **DownloadSpeed** data element with the **Changeable** tag and track the data element's history for auditing purposes. When realizing customer facing services and resource facing services, the default design patterns save data elements that are tagged as **Changeable** to the Service Configuration specifications.
Persisted	Select **Yes** to persist the data element on the conceptual model entity. You must tag a data element with the **Persisted** tag to add it to an Inventory Service specification during realization. Data elements that are tagged as **Changeable** and **Persisted**, and structured data elements that are tagged as **Persisted** are added to the Configuration specification during realization.

Related Topics

Working with Tags

Product Editor

Use the Product editor to model entities that represent something ordered by a customer on a commercial order.

When modeling data in the Product editor, you use some editor tabs that are common among multiple conceptual model entity editors, and you use some editor tabs that are specific to the Product editor.

When working with the Product editor, see the following topics:

- Conceptual Model Editor Data Elements Tab
- Product Editor Derivation Tab
- Product Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Product Editor Derivation Tab

Use the **Derivation** tab to associate products to customer facing services and resources.

When modeling data in the **Derivation** tab, see the following topics:

- Product Editor Derivation Tab Customer Facing Services Area
- Product Editor Derivation Tab Customer Facing Service Detail Area
- Product Editor Derivation Tab Resources Area
- Product Editor Derivation Tab Resource Detail Area

Product Editor Derivation Tab Customer Facing Services Area

Use the **Customer Facing Services** area to associate the product with customer service facing components.

Field	Use
Remove	Click to remove the relationship between the product and the selected component.
Add	Click to create a relationship between the product and the selected component. See "Defining Conceptual Model Components" for more information.

Related Topics

Product Editor Derivation Tab

Product Editor Derivation Tab Customer Facing Service Detail Area

Use the **Customer Facing Service Detail** area to define the component options, the cardinality, and the type of relationship the product has with each component.

Details Tab

Field	Use
Name	Displays the name of the component.
Customer Facing Service	Displays the customer facing service that the component can represent. Do one of the following: • Select a value and click **Open** to open the entity in an editor. • Click **Add** to create a new entity and associate the new entity with the component. • Select a value and click **Remove** to remove the association between the component and the entity. • Click **Select** to associate an existing entity to the component.
Relationship Type	Displays the type of relationship defined for the association. Do one of the following: • Click **Select** to select a different type of relationship. • Click **Relationship Type** to open the listed value in the Relation Type editor. See "Defining Conceptual Model Components" for more information. **Note**: Every Customer Facing Service entity must have a **Primary** relationship defined with at least one product.

Used By Tab

Use the **Used By** tab to review the projects and entities in which the component is used.

Notes Tab

Use the **Notes** tab to annotate entities with descriptions or with other applicable information to support the entity. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML markup.

See "Notes Tab" for more information.

Related Topics

Product Editor Derivation Tab

Product Editor Derivation Tab Resources Area

Use the **Resources** area to associate the product with resource components.

Field	Use
Remove	Click to remove the relationship between the product and the selected component.
Add	Click to create a relationship between the product and the selected component. See "Defining Conceptual Model Components" for more information.

Related Topics

Product Editor Derivation Tab

Product Editor Derivation Tab Resource Detail Area

Use the **Resource Detail** area to define the component options, the cardinality, and the type of relationship the product has with each component.

Details Tab

Field	Use
Name	Displays the name of the component.
Resource	Displays the resource that the component can represent. Do one of the following: • Select a value and click **Open** to open the entity in an editor. • Click **Add** to create a new entity and associate the new entity with the component. • Select a value and click **Remove** to remove the association between the component and the entity. • Click **Select** to associate an existing entity to the component.

Field	Use
Relationship Type	Displays the type of relationship defined for the association. Do one of the following: • Click **Select** to select a different type of relationship. • Click **Relationship Type** to open the listed value in the Relation Type editor. See "Defining Conceptual Model Components" for more information.

Used By Tab

Use the **Used By** tab to review the projects and entities in which the component is used.

Notes Tab

Use the **Notes** tab to annotate entities with descriptions or with other applicable information to support the entity. For example, you can contribute content to Design Studio reports by writing your own internal documentation about entities and data elements, and you can format the documentation using plain text or simple HTML markup.

See "Notes Tab" for more information.

Related Topics

Product Editor Derivation Tab

Product Editor Properties Tab

Use the **Properties** tab to define product import information, to associate fulfillment patterns to the product, and to extend a product from another product.

Field	Use
Effective Start Date and **Effective End Date**	Displays start and end date information if included during a product import.
Extends	(Optional) Extend the product from another product. When you extend one entity from another, the target entity inherits all of the data elements defined for the extended entity. Do one of the following: • Click **Extends** to create and extend a new source product. • Click **Select** to extend an existing source product.

Field	Use
Fulfillment Pattern	Associate fulfillment patterns to the product. Fulfillment patterns realize as Oracle Communications Order and Service Management (OSM) fulfillment patterns. See "Working with Fulfillment Patterns" for more information. Do one of the following: • Click **Select** to associate an existing fulfillment pattern to the product. • Click **Add** to create a new fulfillment pattern and associate the new fulfillment pattern to the product. • Select a value in the **Fulfillment Pattern** field and click **Open** to open the fulfillment pattern in the Fulfillment Pattern editor. • Select a value in the **Fulfillment Pattern** field and click **Remove** to remove the association between the fulfillment pattern and the product.
Key	Enter a value to enable the Order and Service Management order transformation manager to identify the payload of a line item at run time. The default value for this field is the name of the entity. You can define a different key value for an entity if your business processes require a naming convention less restrictive than then entity name requirements.

Related Topics

Product Editor

About Fulfillment Patterns

Customer Facing Service Editor

Use the Customer Facing Service editor to model entities that define the characteristics of a service.

When modeling data in the Customer Facing Service editor, you use editor tabs that are common among multiple conceptual model entity editors. See the following topics:

- Conceptual Model Editor Data Elements Tab
- Conceptual Model Editor Components Tab
- Conceptual Model Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Resource Facing Service Editor

Use the Resource Facing Service editor to configure the technical sets of services that describe how a customer facing service is provided.

When modeling data in the Resource Facing Service editor, you use editor tabs that are common among multiple conceptual model entity editors. See the following topics:

- Conceptual Model Editor Data Elements Tab
- Conceptual Model Editor Components Tab
- Conceptual Model Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Resource Editor

Use the Resource editor to configure the technical components of a solution.

When modeling data in the Resource editor, you use editor tabs that are common among multiple conceptual model entity editors. See the following topics:

- Conceptual Model Editor Data Elements Tab
- Conceptual Model Editor Components Tab
- Conceptual Model Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Location Editor

Use the Location editor to configure locations.

When modeling data in the Location editor, you use editor tabs that are common among multiple conceptual model entity editors. See the following topics:

- Conceptual Model Editor Data Elements Tab
- Conceptual Model Editor Properties Tab
- Conceptual Model Editor Other Relationships Tab
- Conceptual Model Editor Categorization Tab

Action Editor

Use the Action editor to configure the data required by, the specific properties of, and the actions for action families.

When working with actions, see the following topics:

- Conceptual Model Editor Data Map Tab
- Conceptual Model Editor Data Elements Tab
- Action Editor Action Codes Tab
- Action Editor Properties Tab

Action Editor Action Codes Tab

Use the **Action Codes** tab to review the action codes associated with an action, to add new action codes, to remove action codes that are not applicable, and to edit specialized action code names.

Field	Use
Application Role	Displays the name of the application role that represents a specific type of downstream delivery system. The application role characterizes the technical action family and the specialized action code names that are associated with a specific edge fulfillment system. Design Studio includes a set of predefined application roles. Click the **Application Role** link to open the application role in the Application Role editor.
Specialized Action Code	Displays the list of action codes associated with the action. The list that initially appears is inherited from the associated application role. You can add or remove **Specialized Action Name** and **Action Code** pairs. The action codes available to add to the action are defined on the functional area associated with the application role (specified in the **Action Type** field). For example, when modeling technical actions, you can add any action code defined on the **Technical** functional area.
Specialized Action Code Details	(Technical actions only) In the **Specialized Action Name** field, you can edit the value to ensure that the Specialized Action Name is specific to the application role and to the fulfillment system to which the name is relevant. Click the **Action Code** link to open the action code in the Action Code editor.

Related Topics

Action Editor

Configuring Actions

About Conceptual Model Actions

About Application Roles

Action Editor Properties Tab

Use the **Properties** tab to review or define information about the action and to define how the action realizes.

Field	Use
Action Type	Displays the functional area to which the action belongs, such as Service or Technical. Click **Select** to define the action with a different action type.
Subject	Displays the conceptual model entity with which the action is associated. Click **Select** to associate the action with a different entity.
Extend Subject	If selected, indicates that the data elements defined on an associated conceptual model entity are inherited by the action.

Field	Use
Target	Select the resource (or, in a limited number of scenarios, the resource facing service) against which the technical action family runs.
Key	Enter a value to enable the Order and Service Management order transformation manager to identify the payload of a line item at run time. The default value for this field is the name of the entity. You can define a different key value for an entity if your business processes require a naming convention less restrictive than then entity name requirements.
Implementation System	Define the system where the application entity is used. Select one of the following: • Select **None** if the action will not be realized as an application entity. You select this option, for example, if the action is intended for informational use only. • Select **Unified Inventory Management** if a service action will be realized as an entity used in the Design Studio for Inventory application. • Select **ASAP** if the technical action will be realized as an entity used in the Design Studio for ASAP application. • Select **Other System** if the action will be realized as an application entity that will be used in a system other than Unified Inventory Management. **Note**: The design patterns that are delivered with Design Studio and that realize the conceptual model entities add configuration to your workspace only if you define this field with the **Unified Inventory Management** value.
Realization Design Pattern	Select which design pattern converts the action into an application entity and creates the application entity configuration in the workspace.

Field	Use
Implementation Method	Specify the application entity into which the action realizes. Select: • **Activation Service Action Structure** to realize technical actions in Design Studio for ASAP projects as service actions (CSDLs). • **None** if the action will not realize as a Design Studio entity. If you select this option, the **Realization Design Pattern** field is not available. • **Other Action** if the action will not realize as one of the available Design Studio entities but if you want to select a custom design pattern from the **Realization Design Pattern** field to generate code. See *Design Studio Developer's Guide* for more information about creating custom design patterns. • **Ruleset Structure** to realize service actions in Design Studio for Inventory project as rulesets. • **Scan Action Structure** to realize technical actions in Design Studio for Network Integrity projects as scan actions.
Run Realization Design Pattern Automatically	Select to synchronize the action and application entities automatically. When you select this option, Design Studio runs a design pattern automatically when you: • Save the Action editor. • Change the action using an option that you select from the context menu. • Manually synchronize or realize a direct child of the action. See "Synchronizing Conceptual Model Entities with Application Entities" for more information.
Realized By	Displays the name of the application entity realized from the action. This field is blank if the design pattern selected in **Realization Design Pattern** has not yet run.

Related Topics

Configuring Actions

Action Editor

About Conceptual Model Actions

Action Code Editor

Use the Action Code editor to review the functional areas (or action types) to which the action code belongs. The action codes delivered with Design Studio are defined in the Common Model Base Data cartridge project. See "Generating the Common Model Base Data Project" for more information.

Field	Use
Functional Area	Displays the list of functional areas in which the action code is included as a default value. Do one of the following: • Click **Open** to open functional area in the Functional Area editor. • Click **Remove** to delete a functional area from the list. • Click **Select** to add an existing functional area to the list.

Related Topics

Action Editor

About Conceptual Model Actions

Application Role Editor

Use the Application Role editor to define specialized action code name and action code pairs. These pairs can be added to technical action families and can be associated with a specific edge fulfillment system.

Field	Use
Short Name	Enter an abbreviation for the application role. This value is required. When you create technical actions, Design Studio uses this value to help generate the default name of the action. The default name that is generated is a combination of the action prefix defined in the functional area, followed by the application role short name, followed by the subject. For example: **TechnicalActions_ACT_DSL_RFS** where **TechnicalActions** is the action prefix defined on the functional area, **ACT** is the short name defined for the Activation application role, and **DSL_RFS** is the resource facing service that is the subject of the action.
Action Type	Displays the functional area associated with the application role. The application role can include any action code defined on the associated functional area. Click the **Action Type** link to open the functional area in the Functional Area editor.
Specialized Action Code	Click **Add** to include an action code and to specify a corresponding specialized action name. The action codes available to add to the application role are defined on the functional area associated with the application role (specified in the **Action Type** field). For example, when modeling technical actions, you can add any action code defined on the **Technical** functional area. Select the **Specialized Action Name** and **Action Code** pair and click **Remove** to remove the pair.

Field	Use
Specialized Action Code Detail	Edit the **Specialized Action Name** to ensure that the value is specific to the application role and to the fulfillment system to which the name is relevant. Click the **Action Code** link to open the action code in the Action Code editor.

Related Topics

About Application Roles

Creating Actions

Action Parameter Binding Editor

You use the Action Parameter Binding editor to map data elements defined for conceptual model entities and components (the source data) to data elements defined for technical actions (the target data).

When working with the Technical Action Parameter Binding editor, see:

- Action Parameter Binding Editor Bindings Tab
- Action Parameter Binding Editor Conditions Tab
- Action Parameter Binding Editor Context Tab
- Action Parameter Binding Editor Binding Details Tab
- Action Parameter Binding Editor Custom Bindings Tab
- Action Parameter Binding Editor Binding Conditions Tab

Action Parameter Binding Editor Bindings Tab

You use the **Bindings** tab to map the simple and structured data elements in the conceptual model tree on the left to the action family simple and structured data elements displayed on the right. In a simple action parameter binding, the subject contains all of the data that the action family requires.

Field	Use
Input	Displays the simple and structured data elements defined for the conceptual model entity. You can select an entity, a simple data element, or a structured data element and drag the selection to a simple or structured data element in the Action Family area to bind the two selections. Binding a data element indicates that the data element defined in the conceptual model entity is the source for the target data element defined in the action family.

Field	Use
Output	Displays all of the data elements defined for all actions in the action family. Right-click a data element in the Action Family area and select **Add Custom Binding** to define a relative path to a data element outside of the conceptual model subject. Design Studio displays a marker next to data elements defined with custom bindings to enable you to identify those bindings quickly. All custom bindings are displayed on the **Custom Bindings** tab.
Contributor from 'Before' Configuration Version	Select to indicate the reference of previous version of the configuration.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Action Parameter Binding Editor Conditions Tab

You use the **Conditions** tab to define the conditions under which an action is created for a subject. You define conditions to ensure that run-time changes to an entity initiate the correct set of technical actions.

Field	Use
Action Conditions	Displays a list of the condition name and action combinations. You can apply conditions to each name and action combination in the list.
Action	Displays the name of the action that is selected in the Action Conditions list. Click **Select** to associate a different action with the condition name.
Action Code	Displays the action code associated with the action selected in the Action Conditions list. Technical actions are associated with a single action code. Click **Select** to associate a different action code with the action.
Condition	Enter the XPath expression to describe the condition under which the technical action is applicable to the action parameter binding.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Action Parameter Binding Editor Context Tab

You use the **Context** tab to define the context in which the action parameter binding is to be used and to define which technical actions to include in the action family.

Field	Use
Subject	Displays the conceptual model entity that contains the source data.
Context	Define the context in which the action parameter binding is to be used. By default, the context is defined using the relative path of the subject, and indicates that the action parameter binding is to be used wherever the subject is used. You can change the context if the action parameter binding is applicable for a more specific context. You may need to define the context more broadly if the information required by the set of technical actions must be sourced from multiple areas of a conceptual model tree.
Action Family	Do one of the following: • Select **Include All Actions** to include all technical actions defined for the subject in the action family. • Select **Select Specific Actions** to limit the scope of the action parameter binding to a subset of actions defined for the subject. The actions appear in the **Action** field. You can add to or remove any technical actions defined for the subject. For example, you might have technical actions to create, modify, and delete entries in the network address template. You can create a single action parameter binding for all of these technical actions, or you can limit the scope if you require alternate bindings for specific scenarios. By default, all actions associated with the subject are included in the action parameter binding.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Action Parameter Binding Editor Binding Details Tab

You use the **Binding Details** tab to review binding details. For example, you can review the bindings on this tab to edit or delete invalid bindings.

Field	Use
Bindings	Displays a list of the bindings. Select a binding from the list to edit the properties. Select a binding and click **Remove** to clear the binding.
Contributor	Displays the path to the source conceptual model entity.
Source	Displays the entity or the data element defined on the conceptual model entity for which the binding is defined.
Source Path	Displays the relative path to the source data element.
Target	Displays the simple or structured data element defined on the technical action for which the binding is defined.
Target Path	Displays the relative path to the target data element.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Action Parameter Binding Editor Custom Bindings Tab

You use the **Custom Bindings** tab to create custom bindings, which enable you to define XPath expressions for target action data.

For example, you can bind the technical action data element to a source data element defined outside of the subject. You can create custom action parameter bindings when a subject does not contain all of the data required by the associated technical actions.

Field	Use
Custom Bindings	Displays a list of the custom bindings. Select a custom binding from the list to edit the properties.
Target Data Element	Displays the data element defined on the technical action for which the custom binding is defined. Click **Select** to associate a different target data element with the custom binding.
Target Data Element Path	Displays the relative path to the target data element.
Custom Binding	Displays the binding defined for the target data element. Click to edit the binding.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Action Parameter Binding Editor Binding Conditions Tab

You use the **Binding Conditions** tab to define the conditions under which a custom binding is created. You define conditions to ensure that run-time changes to an entity initiate the correct set of technical actions.

Field	Use
Bindings Conditions	Displays a list of the custom bindings in the action parameter binding. You can apply conditions to each of the bindings in the list.
Target Data Element	Displays the name of the target data element that is selected in the Binding Conditions list. Click **Select** to associate a different target element with the binding condition.
Target Data Element Path	Displays the relative path to the target data element.
Binding Condition	Enter the XPath expression to describe the condition under which the technical action is applicable to the action parameter binding.

Related Topics

Action Parameter Binding Editor

Creating Action Parameter Bindings

About Action Parameter Bindings

Relationship Type Editor

You use the Relationship Type editor to review the action code mappings that the Order and Service Management order transformation manager requires to transform customer-focused order items (what the customer bought) to service-focused order items (the services that equate to what the customer bought).

The action code mappings delivered with Design Studio describe the behavior of Primary and Auxiliary relationships between products and customer facing services. You can also create your own relationship types.

Field	Use
Add	Click to create a new action code mapping.
Remove	Click to remove the selected action code mapping.
Source Action Code	Displays the requested action to be performed against the product on the order line. Do one of the following: • Click **Source Action Code** to open the code in the Action Code editor. • Click **Select** to select an action code.
Current Target Action Code	Displays the last action requested against a service. Do one of the following: • Click **Current Target Action Code** to open the code in the Action Code editor. • Click **Select** to select an action code.

Field	Use
New Target Action Code	Displays the action to be performed against a service when the associated source action and current target action types are present. Do one of the following: • Click **New Target Action Code** to open the code in the Action Code editor. • Click **Select** to select an action code.

Related Topics

Defining Conceptual Model Components

Domain Editor

Use the Domain editor to organize conceptual model entities in ways that are meaningful to your design. For example, you can create domains that are service or project-based.

Field	Use
Conceptual Model	Select the conceptual model entities that you want to associate with the domain. Do one of the following: • Click **Open** to open the conceptual model entity editor. • Click **Add** to create a new conceptual model entity and associate the new entity to the domain. • Click **Remove** to delete the selected conceptual model entity from the list. • Click **Select** to associate an existing conceptual model entity to the domain.
Functional Area	Associate the domain to a functional area, such as **Service** or **Technical**. Do one of the following: • Click **Open** to open the selected functional area in the Functional Area editor. • Click **Add** to create a new functional area and associate it to the domain. • Click **Remove** to delete the selected functional area from the list. • Click **Select** to associate an existing functional area to the domain.

Field	Use
Sub Domain	Associate the domain to a subdomain. You can create hierarchies of domains by creating and associating subdomains with domains. Do one of the following: • Click **Open** to open the selected domain in the Domain editor. • Click **Add** to create a new domain as a subdomain. • Click **Remove** to delete the selected domain from the list. • Click **Select** to associate an existing domain as a subdomain.

Related Topics

About Domains

Functional Area Editor

Use the Functional Area editor to specify whether the functional area supports actions, to define the entities that can be associated with the functional area actions, to define how the Functional Area entity is realized into an application entity, and to define default data elements for associated service actions.

When working with the Functional Area editor, see the following topics:

- Conceptual Model Editor Data Map Tab
- Conceptual Model Editor Data Elements Tab
- Functional Area Editor Action Support Tab
- Functional Area Editor Realization Tab

Functional Area Editor Action Support Tab

Use the Functional Area editor **Action Support** tab to specify whether the functional area supports actions and to define the conceptual model entities that can be associated with the functional area actions.

Field	Use
Action Support area	Select **This Functional Area Supports Action** if the functional area supports actions. Service order management and technical order management layers support actions. Commercial order management layers do not support actions. Define the naming convention of the functional area actions by specifying a value in the **Action Prefix** field. For example, if configuring a service order management functional area, you might define this value as **service_action**.

Field	Use
Supported Entity Types area	Click **Add** to specify the conceptual model entities with which the functional area actions can be associated. Click **Remove** to remove a conceptual model entity from the list.
Supported Entity Type Details area	For the selected conceptual model entity, define the following: • **Mandatory**: Select to indicate that the conceptual model entity requires an association with the type of action defined by the functional area. When this option is selected for an entity in the Supported Entity Types table, Design Studio automatically creates an action entity and associates the action entity with any new entities that you create. • **Multi Instance**: Select to indicate that multiple actions can be associated with the conceptual model entity. Customer facing services can have only one associated service action. Resources, however, can be associated with multiple technical actions. • **Extend Subject by Default**: If selected, indicates that the data elements defined on the conceptual model entity are inherited by associated actions.
Default Action Codes area	Displays the action codes that are supported in the functional area. Action codes represent the base operation types in a solution. For example, the default values defined for a functional area can include **Add**, **Modify**, and **Delete** action codes. Select an action code and click **Open** to open the code in the Action Code editor.

Related Topics

Creating Functional Areas

Creating Actions Automatically

About Functional Areas

Functional Area Editor Realization Tab

Use the Functional Area editor **Realization** tab to specify how the Functional Area entity is realized into an application entity.

Field	Use
Implementation System	Indicate whether the Functional Area entity will be realized as an Order and Service Management (OSM) order. Each functional area can be realized by a different OSM order, each with a unique orchestration configuration.

Field	Use
Realization Design Pattern	Select which design pattern converts the Functional Area entity into an application entity and creates the application entity configuration.
Run Realization Design Pattern Automatically	Select to synchronize the Functional Area entity and the realized application entity automatically. When you select this option, Design Studio runs a design pattern automatically when you: • Save of the Functional Area entity. • Change the Functional Area entity using an option that you select from the context menu. • Manually synchronize or realize a direct child of the Functional Area entity. See "Synchronizing Conceptual Model Entities with Application Entities" for more information.
Realized By	Displays the name of the application entity realized from the Functional Area entity. This field is blank if the design pattern selected in **Realization Design Pattern** has not yet run.
Provider Functions	Associate the functional area with provider functions. Provider functions are processing components that perform a defined set of tasks based on its role in a solution. Design Studio includes the configuration for some provider functions, such as Calculate Service Order, Design and Assign, Calculate Technical Order, and Activation. Do one of the following: • Click **Open** to open the selected provider function in the Provider Function editor. • Click **Add** to create a new provider function to associate with the functional area. • Click **Remove** to delete the selected provider function from the list. • Click **Select** to associate an existing provider function with the functional area.

Related Topics

Creating Functional Areas

About Functional Areas

Provider Function Editor

Use the Provider Function editor to review the configuration of the provider functions delivered with Design Studio or to create your own provider functions. For example, you might create a provider function that works with your workforce management system.

When working with the Provider Function editor, see the following topics:

- Provider Function Editor Inputs Outputs Action Tab
- Provider Function Editor Relationship Types Tab
- Provider Function Editor Realization Tab

Provider Function Editor Inputs Outputs Action Tab

Use the **Inputs Outputs Action** tab to define the conceptual model entities and the associated action types that the provider function requires as input and generates as output.

Field	Use
Input Types	Displays the types of conceptual model entities that the provider function recognizes as input values. Click the **Add** and **Remove** buttons to extend or filter the list of acceptable entities. For example, the Calculate Service Order provider function only recognizes Product entities as input.
Input Type Details	Displays the conceptual model entity type and associated action type of the conceptual model entity displayed in the Input Types area. • **Entity Type** displays the conceptual model entity type that is selected in the Input Types table. Click the menu to change the value to a different entity type. • **Action Type** displays the type of action associated with the conceptual model entity selected in the Input Types table. The provider function uses the actions associated with the entity type as input. The value that initially appears in this field is determined by the configuration defined in the functional area. Click **Select** to select a different functional area.
Output Types	Displays the types of conceptual model entities that the provider function generates as output. Click the **Add** and **Remove** buttons to extend or filter the list of generated entities. For example, the Calculate Service Order provider function only recognizes products as input, and generates only the actions associated with Customer Facing Service and resources.
Output Type Details	Displays the conceptual model entity type and associated functional area of the conceptual model entity displayed in the Output Types area. • **Entity Type** displays the conceptual model entity type that is selected in the Output Types table. Click the menu to change the value to a different entity type. • **Action Type** displays the type of action associated with the conceptual model entity selected in the Output Types table. The provider function uses the actions associated with the entity type as output. The value that initially appears in this field is determined by the configuration defined in the functional area. Click **Select** to select a different functional area.

Related Topics

About Provider Functions

Provider Function Editor Relationship Types Tab

Use the **Relationship Types** tab to define how the conceptual model entities that the provider function requires as input can be related to the conceptual model entities that the provider function generates as output.

The relationship types that you define here will determine how conceptual model entities are associated with the components in the conceptual model entity editor **Components** tab, **Derivation** tab, and **Other Relationships** tab.

Field	Use
Default Relationship	Displays the default relationship between the input conceptual model entities and the generated output conceptual model entities. For example, the Calculate Service Order provider function recognizes products as input, and generates actions associated with customer facing services as output. The relationship between a product and a CFS entity is derivational, meaning that the CFS entity has a primary relationship to one product. Typically, Calculate Service Order generates a CFS entity (it can also generate resources, but that scenario is less common). Therefore, the default relationship is defined as Primary. See *Design Studio Concepts* for more information.
Relationship Types	Displays all of the relationship types that can exist among the input conceptual model entities and the generated output conceptual model entities. Do one of the following: • Click **Open** to open the entity in Relationship Type editor. • Click **Add** to create a new entity and to include the new entity as an additional relationship type. • Click **Remove** to remove the relationship type from the provider function. • Click **Select** to add an existing relationship to the provider function.

Related Topics

About Provider Functions

Provider Function Editor Realization Tab

Use the **Realization** tab to define how the provider function is converted to an application entity. The provider functions that are delivered with Design Studio are realized as Design Studio for Order and Service Management transformation manager entities.

Field	Use
Implementation System	Specify how the Provider Function entity will realize. Select: • **None** if the Provider Function entity does not realize as an Order Transformation Manager entity. • **Transformation Manager** to specify that the Provider Function entity realizes as a Transformation Manager entity.
Realized By	Displays the application entity into which the Provider Function entity is converted. Click **Open** to open the selected application entity editor.

Related Topics

About Provider Functions

Fulfillment Pattern Editor

Use the Fulfillment Pattern editor to define the functional area (such as customer, service, or technical) that the pattern supports, the provider functions that are associated with the pattern for a group of related products in a product catalog, and the fulfillment functions that the pattern supports.

Field	Use
Implementation System	Define the system where the application entity is used. Select one of the following: • Select **None** if the conceptual model entity will not be realized as an application entity. You select this option, for example, if the conceptual model entity is intended for informational use only. • Select **Order and Service Management** if the conceptual model entity will be realized in a Design Studio for Order and Service Management project as a Fulfillment Pattern entity. • Select **Other System** if the conceptual model entity will be realized as an application entity that will be used in a system other than Order and Service Management.
Implementation Method	Specify how the Fulfillment Pattern entity realizes as an application entity. Select: • **Fulfillment Pattern Structure** if the Fulfillment Pattern entity realizes as an OSM Fulfillment Pattern entity. • **None** if the Fulfillment Pattern entity realizes as an entity other than an OSM Fulfillment Pattern entity.
Realization Design Pattern	Select which design pattern converts the Fulfillment Pattern entity into an application entity and creates the application entity configuration.

Field	Use
Run Realization Design Pattern Automatically	Select to synchronize the Fulfillment Pattern entity and the realized application entity automatically. When you select this option, Design Studio runs a design pattern automatically when you: • Save of the Fulfillment Pattern editor. • Change the Fulfillment Pattern entity using an option that you select from the context menu. • Manually synchronize or realize a direct child of the Fulfillment Pattern entity. See "Synchronizing Conceptual Model Entities with Application Entities" for more information.
Realized By	Displays the name of the application entity realized from the Fulfillment Pattern entity. This field is blank if you have not yet realized the Fulfillment Pattern entity. Click **Open** to open the selected application entity editor.
Functional Area	Associate a functional area to the fulfillment pattern. Do one of the following: • Click **Select** to associate an existing functional area to the fulfillment pattern. • Click **Add** to create a new functional area and associate the new functional area to the fulfillment pattern. • Select a value in the **Functional Area** field and click **Open** to open the functional area in the Functional Area editor. • Select a value in the **Functional Area** field and click **Remove** to remove the association between the functional area and the fulfillment pattern.
Provider Function	Associate a provider function to the fulfillment pattern. Do one of the following: • Click **Select** to associate an existing provider function to the fulfillment pattern. • Click the clear button (represented by a red X) to remove the association.
Fulfillment Function	Associate fulfillment functions to the fulfillment pattern. Do one of the following: • Click **Select** to associate an existing fulfilment function to the fulfillment pattern. • Click **Add** to create a new fulfilment function and associate the new fulfilment function to the fulfillment pattern. • Select a value in the **Fulfilment Function** field and click **Open** to open the fulfilment function in the Fulfilment Function editor. • Select a value in the **Fulfilment Function** field and click **Remove** to remove the association between the fulfilment function and the fulfillment pattern.

Related Topics

About Fulfillment Patterns

Fulfillment Function Editor

Use the Fulfillment Function editor to define the functions that represent the work to be performed against an action.

Field	Use
Implementation System	Define the system where the application entity is used. Select one of the following: • Select **None** if the conceptual model entity will not be realized as an application entity. You select this option, for example, if the conceptual model entity is intended for informational use only. • Select **Order and Service Management** if the conceptual model entity will be realized in a Design Studio for Order and Service Management project as an Order Component specification. • Select **Other System** if the conceptual model entity will be realized as an application entity that will be used in a system other than Order and Service Management.
Implementation Method	Specify how the Fulfillment Function entity realizes as an application entity. Select: • **OSM Order Component** if the Fulfillment Pattern entity realizes as an OSM Order Component specification. • **None** if the Fulfillment Pattern entity realizes as an entity other than an OSM Order Component specification.
Realization Design Pattern	Select the design pattern that converts the Fulfillment Function entity into an application entity and creates the application entity configuration.
Run Realization Design Pattern Automatically	Select to synchronize the Fulfillment Function entity and the realized application entity automatically. When you select this option, Design Studio runs a design pattern automatically when you: • Save of the Fulfillment Function editor. • Change the Fulfillment Function entity using an option that you select from the context menu. • Manually synchronize or realize a direct child of the Fulfillment Function entity. See "Synchronizing Conceptual Model Entities with Application Entities" for more information.
Realized By	Displays the name of the application entity realized from the Fulfillment Function entity. This field is blank if you have not yet realized the Fulfillment Function entity. Click **Open** to open the selected application entity editor.

Conceptual Model Unit of Measure Editor

Use the conceptual model Unit of Measure editor to define units of measure for any integer data element types. You can define units of measure for data elements that you include in conceptual model entities.

A unit of measure is a quantity or increment by which something is divided, counted, or described. For example, Kbps is a unit that measures a bit rate. If you use Order and Service Management Order Transformation Manager, you may need to define units of measure to ensure that your mapping rules are valid.

Field	Use
Unit of Measure area	Click **Add** to define a new unit of measure. Select a unit of measure and click **Remove** to delete the unit of measure from the workspace.
Name	Displays the unit of measure selected in the **Unit of Measure** area table. You can edit the name in this field.
Display Name	Displays the name that appears in Design Studio selection dialog boxes and in run-time environments. You can edit the display name in this field.
Value	Displays the value assigned to the unit of measure. You can edit the value in this field.
Description	Displays a description of the unit of measure. You can edit the description in this field.

Synchronization Record Editor

Use the Synchronization Record editor to review synchronization records and to correct invalid data in records.

When working with the Synchronization Record editor, see the following topics:

- Synchronization Record Editor Synchronization Details Tab
- Synchronization Record Editor Token Values Tab
- Synchronization Record Editor References Tab

Synchronization Record Editor Synchronization Details Tab

Use the **Synchronization Details** tab to review information about the design pattern and the project associated with the synchronization record.

Field	Use
Design Pattern	Displays the design pattern associated with the synchronization record.
Time Stamp	Displays the date and time when the design pattern last ran.
Synchronization Subject	Displays the conceptual model entity against which the design pattern ran.

Field	Use
Projects area	Displays all design pattern project tokens and the values supplied for those project tokens. Tokens are placeholders that represent information to be collected by the Design Pattern wizard from a user applying a design pattern. Tokens ensure that the resources a design pattern copies to a workspace are based on information supplied by the user who applies the design pattern. See the *Design Studio Developer's Guide* for more information.
Project Token	Displays the design pattern project token selected in the Projects area.
Value	Displays the value of the selected token supplied by the user who applied the design pattern.

Related Topics

Synchronization Record Editor

Realizing Conceptual Model Entities into Application Entities

Synchronizing Conceptual Model Entities with Application Entities

Synchronization Record Editor Token Values Tab

Use the **Token Values** tab to review the tokens defined in the design pattern and the values supplied by the user who last applied the design pattern.

Field	Use
Tokens area	Displays all of the design pattern tokens (that do not collect project information) and the values supplied for those tokens. Tokens are placeholders that represent information to be collected by the Design Pattern wizard from a user applying a design pattern. Tokens ensure that the resources a design pattern copies to a workspace are based on information supplied by the user who applies the design pattern. See the *Design Studio Developer's Guide* for more information.
Token	Displays the design pattern token selected in the Tokens area.
Value	Displays the value of the selected token supplied by the user who applied the design pattern.

Related Topics

Synchronization Record Editor

Realizing Conceptual Model Entities into Application Entities

Synchronizing Conceptual Model Entities with Application Entities

Synchronization Record Editor References Tab

Use the **References** tab to review the element and entity reference tokens defined in a design pattern.

Field	Use
Element Tokens area	Displays all of the design pattern tokens and the values supplied for those tokens. Tokens are placeholders that represent information to be collected by the Design Pattern wizard from a user applying a design pattern. Tokens ensure that the resources a design pattern copies to a workspace are based on information supplied by the user who applies the design pattern. See the *Design Studio Developer's Guide* for more information.
Entity Tokens area	The **Token** field displays the design pattern token selected in the Tokens area. The **Value** field displays the value of the selected token supplied by the user who applied the design pattern. Click **Select** to change the value. Click **Value** to open the token value in the appropriate Design Studio editor.

Related Topics

Synchronization Record Editor

Realizing Conceptual Model Entities into Application Entities

Synchronizing Conceptual Model Entities with Application Entities

6

Building and Packaging Projects

During cartridge development, you must run builds against projects to create and modify workspace resources and to detect errors in your projects. Before you can deploy a cartridge to a run-time environment, you must resolve all errors in the project and determine which entities, libraries, and resources to include (or package) in the cartridge project.

When building and packaging projects, see the following topics:

About Builds

Builds are processes that update existing resources and create new resources. You run builds against projects to create or modify workspace resources. The type of project determines the type of build. For example, when you run a build for a Java project, the build converts each Java source file (**.java** files) into one or more executable class files (**.class** files).

> **Note:**
>
> See the Eclipse Help for more information about builds.

There are two kinds of builds:

- Incremental builds, which update only the resources that have changed since the previous build.
- Clean builds, which update all resources.

There are three ways to run builds:

- Automatically, whenever resources are saved. Automatic builds are always incremental and always affect all projects in the workspace. You enable automatic builds from the **Project** menu by selecting **Build Automatically**.
- Manually, by explicitly selecting a build menu option. You can run incremental or clean builds manually, for specific projects, or for the entire workspace.
- Using a script via command line.

Related Topics

Running Incremental Builds

Running Clean Builds

Running Incremental Builds

By default, builds are performed automatically when you save resources. You can disable automatic building and manually run builds if, for example, you want to finish implementing cartridge changes before building the project.

> **Note:**
>
> Oracle recommends leaving incremental builds enabled. Incremental builds ensure that problem markers remain current and accurate.

To manually run incremental builds on projects:

1. From the **Project** menu, deselect **Build Automatically**.
2. Do one of the following:
   - Select **Build All** to build all projects in the workspace.
   - Select **Build Project** to clean a specific project.

     These options update only the resources that have changed since the previous build.

> **Note:**
>
> You can also manually run clean builds against specific projects or against all projects in the workspace. See "Running Clean Builds" for more information.

Related Topics

About Builds

Running Clean Builds

You can clean a project to resolve any dependencies or similar error from all previous build results. Clean builds update all resources within the scope of the build.

To clean and build a project:

1. From the **Project** menu, deselect **Build Automatically**.

   When this option is selected, you can clean a project but you cannot perform a clean and build (automatic builds are always incremental).
2. From the **Project** menu, select **Clean**.

   The Clean dialog box appears.
3. Do one of the following:
   - To clean all projects in the workspace, select **Clean all projects**.
   - To clean a specific project or group, select **Clean projects selected below**, then select the projects to clean.

4. Select **Start a build immediately**.

   This option enables you to clean and build the selected projects in a single step. When you select this option, you must determine whether to build all projects after the cleanup, or limit the build to the selected projects.

5. Click **OK**.

   Design Studio cleans and builds the selected projects.

Related Topics

Running Incremental Builds

About Builds

About Design Studio Builder

The Design Studio builder generates several artifacts automatically every time you create a cartridge or make a change to a cartridge. You can access the following artifacts from the Package Explorer view of the Java perspective or from the Navigator view of the Resource perspective:

- The cartridge archive file that Design Studio sends to the run-time server when deploying a cartridge. The file is located in the **cartridgeBin** folder.
- A pre-compressed version of the cartridge archive file, which contains all folders, subfolders, and files in the archive. This directory is contained in the **cartridgeBuild** folder. When you make a change to a cartridge, the Design Studio builder process makes changes in the **cartridgeBuild** folder, then it builds the file.

> **Note:**
>
> The Design Studio Builder process is automated; consequently, you should not make any changes in the **cartridgeBuild** or **cartridgeBin** folders (any changes you make will be overwritten) or check these files into source control.

Packaging Projects

Before you deploy a cartridge, determine which entities, libraries, and resources to include (or package) in the Cartridge project. Typically, Design Studio automatically packages your projects during incremental builds. You can disable this functionality and defer packaging until cartridge deployment.

See "Defining Packaging Preferences" for more information about disabling packaging during incremental builds.

> **Note:**
>
> Some Oracle Communications features do not support all Design Studio packaging functionality, and some provide feature-specific variants.

To package a Cartridge project:

1. Double-click a Project entity in the Studio Projects view.

   The cartridge information appears in the Project editor.
2. Click the **Packaging** tab.
3. In the Cartridge Packaging Instructions area, do one of the following:
   - To include all entities of the corresponding type in the cartridge project, select **Include all from Project**.
   - To include a specific set of entities of the corresponding type, deselect **Include all from Project**.

   For example, select **Resources** in the left column and deselect **Include all from Project** to specify a specific subset of resources defined in the project to include in the packaging.
4. Click **Save**.

Related Topics

Deploying Cartridge Projects

7

Deploying Cartridge Projects

When deploying cartridge projects, see the following topics:

- Deploying Cartridge Projects from the Environment Perspective
- Deploying Cartridge Projects with Optimize Deploy
- Cartridge Management View
- Studio Environment Editor

Deploying Cartridge Projects from the Environment Perspective

You deploy the Design Studio cartridge projects from the Environment perspective.

> **Note:**
>
> Design Studio is not intended for automated deployment or production environment deployment. See *Design Studio Developer's Guide* for more information about deploying cartridge projects to production environments.
>
> Before you can deploy cartridge projects from the Environment perspective, you must create at least one Environment project, which enables you to organize your environment attributes. See "Working with Environment Projects" for more information.

To deploy cartridge projects from the Environment perspective:

1. Build the cartridge project.

   Builds detect errors in cartridge projects. You must resolve all errors in a cartridge project before you can deploy the cartridge project.

2. Determine which entities, libraries, and resources to include in the cartridge project.

   See "Packaging Projects" for more information.

3. Select **Studio**, then select **Show Environment Perspective**.

   An Environment perspective is a collection of views that enable you to create and manage the attributes associated with your environment. You use the Environment perspective to deploy and undeploy cartridges to one or more environments and to control and manage all of your environments. See "Working with Perspectives" for more information.

4. Do one of the following:

   - Create a new run-time environment if no environments exist. See "Creating Run-Time Environments" for more information.
   - Select an existing run-time environment to which to connect.

   The Cartridge Management view displays the selected environment information obtained from the most recent queried state.

5. Test the run-time environment connectivity.

   See "Testing Run-Time Environment Connectivity" for more information.

6. Define any environment-specific variables for the test environment.

   When you create Cartridge projects, some of the information you provide may depend on a specific environment. Model variables are placeholders for environment-specific values that can be defined at the time of deployment. See "Working with Model Variables" for more information.

   Cartridge management variables control attributes of the deployment and attributes of the project behavior after you deploy to the target environment. See the list of application-specific topics in the note below for more information.

   > **Note:**
   >
   > Some Design Studio applications require that you define cartridge management variables before you deploy cartridges. The following topics provide more information:
   >
   > - Design Studio for ASAP Cartridge Management Variables Tab
   > - Design Studio for Inventory Cartridge Management Variables Tab
   > - Design Studio for Network Integrity Cartridge Management Variables Tab
   > - About OSM Cartridge Management Variables

7. In the Cartridge Management view, select the cartridge projects to be deployed.

   If you select multiple projects, they are deployed individually, based on any existing dependencies. Oracle recommends that you deploy all run-time dependent projects to the run-time environment prior to or concurrent with any project that references them.

8. Click **Deploy**.

   Design Studio queries the environment for the current state of the deployed cartridges, and validates dependencies before deploying the cartridge projects. If a cartridge project defined as a run-time dependency does not exist in the target environment, Design Studio displays a warning. See "Project Editor Dependency Tab" for more information about defining dependency types.

   If the deployment fails, a message describing the reason for the failure and how to correct it appears in the Console view. If an error occurs during a deployment in which you have selected multiple projects, the system stops deployment for all subsequent projects. If you cancel the deployment, the system attempts to cancel the current deployment, and then cancels all subsequent deployments.

9. (Optional) Click **Query**.

   You can query the environment at any time to view the current system state.

Creating Run-Time Environments

You create a Design Studio run-time environment to contain the run-time environment connection parameters.

To create a run-time environment:

1. From the **Studio** menu, select **Show Environment Perspective**.

The Environment perspective appears.

2. Right-click in the Environments view and select **New Environment**.

   The Studio Model Entity wizard appears, with the project name selected by default.

> **Note:**
>
> Studio Environment entities must be saved in an Environment project. If no Environment projects exist in the workspace, Design Studio prompts you to create one. See "Creating Environment Projects" for more information.

3. In the **Project** field, select the appropriate project for the environment.

   See "Creating Environment Projects" for more information about creating new projects.

4. In the **Name** field, enter a name for the run-time environment.

   The name must be unique among environment entities within the same namespace.

5. (Optional) Select a location in which to save the run-time environment configuration.

   By default, Design Studio saves the environment configuration to your default workspace location. You can enter a folder name in the **Folder** field, or select a location different from the default. To select a different location:

   a. Click the **Folder** field **Browse** button.

   b. Navigate to the directory in which to save the entity.

   c. Click **OK**.

6. Click **Finish**.

   The new run-time environment entity appears in the Environment view.

7. In the Environment view, right-click the environment to which you want to connect and select **Open**.

   The Studio Environment view appears.

8. In the **Address** field, enter the Oracle WebLogic Server IP address (or the fully qualified domain name if DNS is enabled) and port necessary to connect to the run-time environment.

> **Note:**
>
> If you are using an IPv6 URL address, you must be deploying over a network that supports IPv6. The IP address you enter must be in standard IPv6 form. The final four nodes can be omitted if the nodes are all zeroes. For example:
>
> - **[10:20:0:0:0:0:0:0]** or **[10:20:0:0]**
> - **[26AB:FFFF:800:50:0:0:0:0]** or **[26AB:FFFF:800:50]**
> - **[CD:47:2:9999:0:0:0:0]** or **[CD:47:2:9999]**

See "Studio Environment Editor Connection Information Tab" for more information about the **Address** field.

9. (Optional) In the **Cluster/Server** field, enter the name of the cluster or server on which the target application resides.

   See "Studio Environment Editor Connection Information Tab" for more information.

10. (Optional) Enable SSL connections.

    Before you deploy cartridges from Design Studio using an SSL connection, you must enable SSL in the WebLogic server to ensure that the Cartridge Management web service accepts the SSL connection. See *Design Studio System Administration Guide* for information about enabling SSL.

11. Click **Save**.

Related Topics

Deploying Cartridge Projects from the Environment Perspective

Working with Model Variables

Studio Environment Editor

Testing Run-Time Environment Connectivity

You can test a connection to a run-time environment before you deploy cartridges.

To test a connection to a run-time environment:

1. In the Environment perspective Environment view, right-click the Design Studio environment that you want to test and select **Test Connection**.

   The Test Environment Connection dialog box appears.

2. Enter your WebLogic user name and password.

   See *Design Studio System Administration Guide* for information about setting up Design Studio users.

3. Click **OK**.

   Design Studio displays the login result, which determines what information appears in the Cartridge Management view.

   If the login is successful, Design Studio displays information about the state of synchronization among the cartridges in the workspace and cartridges in the run-time environment. If login fails, it displays only cartridges from the workspace.

Related Topic

Deploying Cartridge Projects from the Environment Perspective

Deploying Cartridge Projects with Optimize Deploy

Optimize Deploy is a method of deployment that, when enabled, attempts to deploy only the changes you have made in your Design Studio cartridge project. For example, you can use Optimize Deploy when testing or debugging changes to your cartridge data.

> **Note:**
>
> Some product cartridges do not support optimized deploy, and only Design Studio for OSM creates optimized archives.

When deploying cartridge projects using Optimize Deploy, see the following topics:

- About Optimize Deploy
- Deploying Optimized Builds

About Optimize Deploy

When you enable Optimize Deploy, Design Studio detects any changes you've made to the cartridge, and deploys to the run-time environment only the artifacts related to those changes, whenever possible.

> **Note:**
>
> Optimized Deploy reduces deploy times by skipping optional steps in the build and deploy process. If Optimized Deploy fails to deploy the complete cartridge, run a clean build, then attempt to deploy.

The cartridge and environment combination is persisted for each workspace. Each time you start Design Studio, the system uses your previous cartridge and environment selection combinations.

Related Topics

Deploying Cartridge Projects with Optimize Deploy

Deploying Optimized Builds

Deploying Optimized Builds

Use Optimize Deploy to deploy cartridge projects to run-time environments.

> **Note:**
>
> Before deploying cartridge projects, ensure that you have configured a run-time environment and that the run-time server is running.

To deploy optimized builds:

1. In the Design Studio toolbar, click the **Deploy a Cartridge** icon menu.
2. Select **Optimize**.

   A check mark appears next to the option.
3. Do one of the following:

- In the Design Studio toolbar, click **Deploy a Cartridge**.

  Design Studio uses the previous cartridge and environment combination selection as the default value when you click **Deploy a Cartridge**. If Design Studio finds no previously deployed cartridge and environment combination, it displays a list of available cartridges. If no cartridges are available, Design Studio displays **<no applicable cartridges>**. Cartridges that contain build errors appear in the list but are not available for selection.

- Select **Studio**, then select **Deploy** to select a cartridge and target environment combination.

  The five most recently deployed cartridge and environment combinations appear at the top of the menu.

4. If prompted, log into the run-time environment.
5. If prompted, decide whether to first deploy a full build to the target environment.

   Oracle recommends that, when first starting Design Studio, you perform a full deployment to ensure that a complete cartridge exists in the run-time environment. A full build of the cartridge must exist in the target environment before you deploy an optimized build. Deploying a partially complete cartridge to the run-time environment may result in run-time errors.

   If prompted, do one of the following:

   - To cancel the deployment, click **Cancel**.
   - To deploy a full build, click **Yes**.
   - To continue with the optimized deployment, click **No**.
6. Review the deployment status in the Environment perspective Console view.

Related Topics

Deploying Cartridge Projects with Optimize Deploy

About Optimize Deploy

Cartridge Management View

Use the Cartridge Management view to deploy cartridges in the Design Studio workspace and undeploy them from run-time environments.

The Cartridge Management view lists all available cartridges in your workspace. A status column indicates which cartridges have been deployed and, if so, whether they are synchronized with the target environment. The Deployed Versions table lists which cartridge version and build combination is currently deployed in the target environment (for the selected cartridge). The last refresh time appears at the bottom of the table. Design Studio refreshes the table after cartridge queries, imports, deploys, and undeploys.

Field	Use
Problem Marker column	The first column of the Cartridges table displays an X if the cartridge contains problem markers. You must resolve all problem markers before you can deploy a cartridge to a run-time environment.

Field	Use
Status	Displays an icon to represent the cartridge project status. Hover over the icon to reveal one of the following status descriptions: • The cartridge does not exist in the workspace, but exists in the run-time environment. • The cartridge exists in the workspace but does not exist in the run-time environment. • The cartridge exists in the workspace and in the run-time environment, but the versions are not synchronized (they are no longer identical due to changes to data in the workspace version). • The cartridge exists in the workspace and in the run-time environment, and the versions are identical. **Note:** You must query the environment for state information before cartridge status is displayed. See "Testing Run-Time Environment Connectivity" for more information.
Cartridge Name	Displays the name of all cartridges defined in the workspace.
Cartridge Display Name	Displays the name defined for the Project entities in the Project editor **Description** field.
Type	Indicates the type of Cartridge project. For example, this column indicates whether the cartridge is an OSM cartridge project, an ASAP cartridge project, a UIM cartridge project, and so forth.
Version	Displays the cartridge version number currently saved in the Design Studio workspace. The number of digits in the version number depends on the number of digits supported by the target environment for the project. Some target environments support five digits, and some support only three digits. Cartridges are synchronized when the version number and the name match.
Build	Indicates which build of the cartridge data is used by the corresponding cartridge. If you have enabled the automatic build feature, Design Studio increases the build number automatically every time you save. To enable the automatic build feature elect **Project**, then **Build Automatically.**
Operation	Displays the operations that the workbench is currently performing against the cartridge. This column does not display any operations initiated by other instances of Design Studio or by the Cartridge Management web service client tools.
Query	Click to refresh the view of the listed cartridges.
Deploy	Click to deploy the selected cartridges to a run-time environment.

When you select a cartridge in the Cartridge area, Design Studio populates the following fields for that cartridge in the Deployed Versions area:

Field	Use
State	Displays the state of the selected cartridge. For example, this field displays **active** if the selected cartridge has been deployed.
Version	Displays the version number of the cartridge that was deployed to the run-time environment.
Build	Displays the build number of the cartridge deployed to the run-time environment.

Field	Use
Deployed On	Displays the date of the last deployment of the cartridge selected in the Cartridge area.
Provider	Displays the cartridge provider, if available. For example, for cartridges that you obtained from the Oracle software delivery website, **Oracle** appears in this column.
Show Details	Click to review copyright and license information for the cartridge selected in the Deployed Versions area. The dialog box that appears when you click **Show Details** also displays the name, type, version, and build number of the selected cartridge. This button is not available for selected cartridges deployed to run-time environments older than 7.3.0, or when cartridges with multiple versions are selected.
Undeploy/Remove	Select one or multiple cartridges in the Deployed Versions area and click **Undeploy/Remove** to undeploy those cartridges from a run-time environment. Additionally, you can select one or multiple cartridges in the Deployed Versions area and click **Undeploy/Remove** to remove from the run-time server those cartridges that failed to deploy successfully. **Note:** This button is not available for all types of cartridges.
Import	Click to import from the run-time environment the cartridge selected in the Deployed Versions area. **Note:** This button is not available for all types of cartridges.

Studio Environment Editor

Use the Studio Environment editor to define the run-time environment connection information, define the Secure Socket Layer (SSL) keystore file location, review and edit the cartridge and model variables defined for cartridge projects, and to define application-specific connection information.

When defining run-time environment connection information, see the following topics:

- Studio Environment Editor Connection Information Tab
- Studio Environment Editor SSL Tab
- Studio Environment Editor Model Variables Tab
- Studio Environment Editor Cartridge Management Variables Tab

Studio Environment Editor Connection Information Tab

Use the **Connection Information** tab to define the connection parameter necessary to connect to the run-time environment.

> **Note:**
>
> In addition to populating Connection Information tab fields, some Design Studio applications require that you also define cartridge management variables before you deploy cartridge projects. The following topics provide more information:
>
> - Design Studio for ASAP Cartridge Management Variables Tab
> - Design Studio for Inventory Cartridge Management Variables Tab
> - Design Studio for Network Integrity Cartridge Management Variables Tab
> - About OSM Cartridge Management Variables

Field	Use
Address	Enter the WebLogic IP address (or the fully qualified domain name if DNS is enabled) and port necessary to connect to the run-time environment. Depending on your product installation, Design Studio may populate this field with a default destination URL for a WebLogic server. However, you must edit the IP address/server name and port number to match your own server address configuration: `https://IPAddressOrQualifiedDomanName:port/cartridge/wsapi` where *IPAddressOrQualifiedDomanName* is the IP address or server name of the WebLogic server that you connected to during installation and *port* is the WebLogic server port number configured to receive web requests. **Note:** If you are deploying to a clustered environment, *IPAddressOrQualifiedDomanName* is the proxy server. See product-specific installation guides for more information about installing Oracle Communications applications and connecting to WebLogic servers. See *Design Studio System Administration Guide* for information about deploying cartridges using SSL connections. **Note:** If you are deploying using an SSL connection, you must provide a keystore. See "Studio Environment Editor SSL Tab" for more information. If you are using an IPv6 URL address, you must be deploying over a network that supports IPv6. The IP address you enter must be in standard IPv6 form. The final four nodes can be omitted if they are all zeroes. For example: **https://[2606:b400:2010:504c:216:3eff:fe6f:6d8d]:6001/cartridge/wsapi?wsdl**
Cluster/server	Specify the name of the server or cluster for the target application. When interacting with a run-time environment, an error is returned if the cluster or server name specified in this field cannot be found. This value is not considered when the target environment does not support the cluster/server parameter. **Note**: You must populate this field with a value when deploying to UIM environments.

Related Topics

Testing Run-Time Environment Connectivity

Studio Environment Editor

Studio Environment Editor SSL Tab

Use the **SSL** tab to encrypt your cartridge data prior to deployment.

> **Note:**
>
> Before you deploy cartridges from Design Studio using an SSL connection, you must enable SSL on the WebLogic server to ensure that the Cartridge Management web service accepts the SSL connection. See *Design Studio System Administration Guide* and *Design Studio Security Guide* for information about enabling SSL.
>
> Oracle recommends that you configure environments with SSL to increase security.

Field	Use
Keystore	Identify the location of your keystore file. The keystore is a file (encrypted with a password) that contains private keys and trusted certificates.

Studio Environment Editor Model Variables Tab

Use the **Model Variables** tab to review and override cartridge-specific model variables defined for all of the cartridges in the workspace.

Column	Use
Model Variables area	Displays the name of the variable and environment-specific values, if defined in the **Override** field. Design Studio displays all model variables that are defined in the workspace. The values in this table are read-only. You can change the default value defined for a variable by selecting the **Override** option and entering a new environment-specific value.
Environment Model Variable Details area	Displays the name and value of the selected variable in the **Model Variables** area. Do any of the following: • Select **Sensitive** to indicate that the variable has been secured in the "Project Editor Model Variables Tab". When this option is selected, the default value defined for the cartridge model variable is obfuscated in the user interface and in memory. If multiple cartridges define a variable with the same name, the Environment editor secures the default value if any of the variables have been configured as sensitive. **Note:** Default values that are defined as sensitive are not displayed when you disable the option. To retain the original value, you can reselect the option prior to entering a new default value. • Select **Override** to define a variable value with an environment-specific value. • Select **Inherit** to use the default value defined for the variable.

Related Topics

Working with Model Variables

Studio Environment Editor

Studio Environment Editor Cartridge Management Variables Tab

Use the **Cartridge Management Variables** tab to review and override cartridge-specific cartridge management variables defined for all of the cartridges in the workspace.

Column	Use
Cartridge Management Variables area	Displays the name of the variable and environment-specific values, if defined in the **Override** field. Design Studio displays all cartridge management variables that are defined in the workspace. The values in this table are read-only. You can change the default value defined for a variable by selecting the **Override** option and entering a new environment-specific value. **Note:** If no override value is defined, Design Studio uses the default value defined for the cartridge selected for deployment.
Environment Cartridge Management Variable Details area	Displays the name and value of the selected variable in the **Cartridge Management Variables** area. Do any of the following: • Select **Sensitive** to indicate that the variable has been secured in the "Project Editor Cartridge Management Variables Tab". When this option is selected, the default value defined for the cartridge management variable is obfuscated in the user interface and in memory. If multiple cartridges define a variable with the same name, the Environment editor secures the default value if any of the variables have been configured as sensitive. **Note:** Default values that are defined as sensitive are not displayed when you disable the option. To retain the original value, you can reselect the option prior to entering a new default value. • Select **Override** to define a variable value with an environment-specific value. • Select **Inherit** to use the default value defined for the variable.

Related Topics

Studio Environment Editor

8

Troubleshooting in Design Studio

When troubleshooting in Oracle Communications Service Catalog and Design - Design Studio, see the following topics:

Resolving Memory Issues

When you launch Design Studio, the Java Virtual Machine (JVM) attempts to allocate the memory necessary to support the Design Studio (and Eclipse) application processes. If there is not enough memory available to allocate from the memory pool, the JVM displays an error message and Design Studio does not start.

Design Studio memory issues can appear erratic because the JVM requires contiguous memory for allocation (the allocation cannot consist of disparate parts of the physical memory pool). The amount of contiguous memory available to the JVM is affected by the number of processes running and the length of time the machine has been running. Even if the operating system reports that there is enough memory to support the allocation request, the allocation will fail unless the memory available is contiguous.

If you are experiencing Design Studio memory issues, make the following changes:

- Close all other open applications.
- Edit the maximum memory settings for the Java heap size.

  If you receive a system error stating:

  ```
  JVM Terminated.Exit code=-1
  ```

  when you attempt to launch Design Studio, it is possible that the heap space requested could not be reserved. Close Design Studio, reduce the `-Xmx` value, and restart Design Studio. Reduce the `-Xmx` value gradually to ensure that Design Studio starts with the greatest heap space possible. Oracle recommends that you begin with decrements of 50.
- If you are using Java 1.5 and getting an `OutOfMemoryError` when opening Design Studio, include the following option (directly after the -Xmx setting) in the **eclipse.ini** file:

  -Dsun.lang.ClassLoader.allowArraySyntax=true
- Add memory to the system.

32-bit systems can support a maximum memory setting of approximately 1GB, depending on current system state.

- Use a 64-bit operating system and a 64-bit version of Design Studio.

  The maximum memory setting for these systems is much greater and can exceed the approximate limit of 1GB on 32-bit systems. 64-bit installations can support maximums of 4GB and higher.

See *Design Studio Installation Guide* for more information about defining JVM flags in Design Studio.

Resolving Cartridge Project Performance Issues

To improve Design Studio performance during import and during modeling, you can separate large cartridges into multiple projects that exist in different workspaces. During the first import, you create the common model project that all of the smaller projects will reference, and select a subset of orders to add to the first project. Create a new workspace for subsequent imports and continue to add subsets of orders. When finished, Design Studio will include multiple projects in multiple workspaces, each sharing the same model project, each with different subsets of orders from the original cartridge. For example, you might include all VOIP orders in a single workspace, all Internet orders in another, and so forth.

You can close projects that you are not actively using to save system resources and prevent unnecessary processing. See "Closing Projects" and "Opening Projects" for more information.

> **Note:**
>
> You cannot close a project if other open projects define it as a dependency. See "Managing Project Dependencies" for more information.

Also, ensure that the recommended `-vmargs` settings are defined in the **eclipse.ini** file. Do not define `-vmargs` settings on the command line or in a desktop shortcut, as these override the **eclipse.ini** settings and can cause performance and memory issues. See *Design Studio Installation Guide* for more information.

Finally, many functions are processor and memory intense. If you are experiencing poor performance, consider more powerful platforms, including faster memory, processors, and disks.

Resolving Plug-in Compatibility Issues

Design Studio does not load Design Studio plug-ins when you are using an unsupported version of the Java Developer Kit and Runtime Environment. If no Design Studio plug-ins load at startup, ensure that you update the Design Studio version to the latest service pack, and that you are using the recommended Java Runtime Environment and Java Developer Kit versions. See *Compatibility Matrix* for more information about system requirements.

Resolving Invalid Problem Markers

Invalid problem markers (for example, those that continue to reappear after you fix the corresponding problem in the configuration), can usually be removed by running a clean and

build. When you run a clean and build, Design Studio discards all build problems and built states of the selected projects and rebuilds those projects from scratch.

See "Running Clean Builds" for more information.

Reviewing the Error Log

The Error Log view captures all warnings and error messages logged by plug-ins. Reviewing the Error Log view can help you find errors in your project builds. The underlying log file resides in the metadata subdirectory of your workspace directory.

Error messages that contain a plus sign preceding the message contain other errors. To display these error messages, click the plus sign to expand the view.

To view the details of any error, double-click the message to display the Event Details dialog box.

Resolving Import Project Errors

If you are not able to import projects into Design Studio successfully, you may be using an incorrect import method. There are two different import methods available in the Design Studio interface: an Eclipse method and a Design Studio method. Always use the Design Studio method. Import projects using the Studio Projects view context menu or with the **Import Studio Project** menu action available in the **Studio** menu. Using the Eclipse import functionality (available by selecting **File**, then **Import**) is not recommended, as it may cause unpredictable results and may require that you restart Design Studio.

Resolving OSM Solution Build Timeout Failures

If the OSM Solution Automation build fails due to a SocketTimeOut exception, increase the value of the **Separate JRE timeout** setting.

To increase the value of the Separate JRE timeout setting:

1. In Design Studio, from the **Window** menu, select **Preferences**.
2. In the Preferences navigation tree, select **Ant**.
3. In the **Separate JRE timeout (ms)** field, enter **40000**.
4. Click **OK**.

Defining Character Encoding

You can define character encoding at the workspace level, at the editor level, or for specific text files. The default character encoding scheme in Eclipse is **cp1252**. You may be required to change this scheme, for example, if you intend to submit orders that contain character sets from languages such as Chinese, Japanese, or Norwegian. In this case, you can define the character encoding scheme as **UTF-8**.

When working with character encoding, see the following topics:

- Defining Character Encoding at the Workspace Level
- Defining Character Encoding for Text Files

Defining Character Encoding at the Workspace Level

To define the character encoding scheme for the entire workspace:

1. From the Design Studio **Window** menu, select **Preferences**.

   The Preferences dialog box opens.

2. Expand **General** and then click **Workspace**.

   The Workspace page appears.

3. In the **Text file encoding** field, select **Other** and then select an encoding value.

4. Click **Apply**, then click **OK**.

Defining Character Encoding for Text Files

Text files do not include encoding declarations. Consequently, Eclipse editors attempt to use the most suitable character encoding, based on preference definitions or on the file content. For example, the default encoding for a Java properties file is **ISO-8859-1**. You may need to change the encoding to **UTF-8** to ensure that the characters display correctly in the Properties File editor.

To define character encoding for text files:

1. In Design Studio, open the text file.

2. From the **Edit** menu, select **Set Encoding**.

   The Set Encoding dialog box appears.

3. Select **Other**.

   A list of available options becomes available.

4. Select a different encoding to associate with the file.

5. Click **Apply**, then click **OK**.

9

Working with Reports

Design Studio enables you to generate reports that include detailed information about an implemented solution. For example, the reports can capture the name, type, description, and relationships of projects, entities, and data elements. You can facilitate information sharing and data reviews by sharing these reports among team members who do not have Design Studio installed locally or who require information about the data model in document form.

Design Studio includes a set of reports that provide a foundational set of capabilities. You can use these report designs as delivered or as a starting point for customizing your own reports. For example, you can customize the delivered report designs for content, layout, or branding.

You can also develop your own report designs using the Eclipse Business Intelligence and Reporting Tools (BIRT) feature. See *Design Studio Developer's Guide* for more information about developing custom reports and packaging custom reports into features.

System administrators can integrate Design Studio report generation into an automated build system. See *Design Studio System Administrator's Guide* for more information.

When working with reports, see the following topics:

About the Design Studio Reports

Design Studio includes a set of report designs that you can use as delivered or as a starting point for customizing your own reports. Most reports can be generated in a number of different output formats, which will appear as options in the Generate Report wizard. Report content may be better suited to one format over another. For example, a report may be available to generate in a number of output formats, such as HTML or Open Office Text (ODT), but designed to display best in PDF format.

Design Studio includes the following reports:

- **Comprehensive Entity Standard Detail Report**: Provides root data element details for entities and entity relationships.
- **Conceptual Model Overview**: Provides an overview of your conceptual model, including information about Product, Customer Facing Service, Resource Facing Service, Resource, and other supporting entities.
- **Data Model Report**: Provides a detailed view of simple and structured data elements.
- **Entity Summary Report**: Provides a summary of entities and entity relationships.
- **Project Summary Report**: Provides an overview of projects and project dependencies.
- **Design Studio Model in XML Report**: Provides XML output of Design Studio projects.
- **Entity and Element Type Reference Report**: Provides a list of entity and element types available for reporting.

Related Topics

Contributing Documentation to Reports

Generating Reports

Viewing the Report Design Example

Contributing Documentation to Reports

You can contribute content to Design Studio reports by writing documentation about entities and data elements in Design Studio editors and views. You can format the documentation using plain text or simple HTML markup.

Reference reports assume that documentation is formatted as HTML markup. If you add documentation in plain text, the documentation appears as plain text in a single paragraph with no formatting.

To contribute documentation to a report:

1. Open an entity in a Design Studio editor and do one of the following:
   - Click the **Data Elements** tab, then click the **Notes** tab.
   - Click the editor **Notes** button. This button is located in the top right corner of Design Studio editors, next to the **Help** button.
2. Add documentation about the entity or data element using your selected markup syntax.

Related Topics

Working with Reports

Project Editor Properties Tab

Notes Tab

Generating Reports

You can generate reports in multiple output formats, including PDF, HTML, Microsoft Word, and so on. Report features may vary depending on the report output format.

The size and complexity of a report impacts the time required to generate the report. Oracle recommends that you limit the scope of a report to the most specific part of the solution as possible, and to define the filtering criteria as narrowly as possible. For example, instead of generating one large report, generate sets of complimentary reports that describe subsets of a solution to improve report generation times.

When reports contain data from sealed projects and from unsealed projects, it may be difficult to distinguish data under active development from static data included in sealed projects. Generate reports that are specific to sealed projects separately from the reports that are specific to the active solution.

System administrators can include report generation in a continuous integration cycle by using Ant tasks. See the *Design Studio System Administrator's Guide* for more information.

> **Note:**
>
> Design Studio reports do not automatically update when changes are made to the report source data.
>
> You can generate reports using data from projects that contain errors, but some content may fail to appear, depending on the error. For example, content defined with an invalid reference will fail to appear in a report.

To generate a report:

1. Build the projects that contain the data to be included in the report.

   Design Studio includes the data from the last project build. To generate a report with the latest model information, perform a clean build. See "Running Clean Builds" for more information.

2. From the **Studio** menu, select **Generate Report**.

   The Generate Report dialog box appears.

3. Do one of the following:

   - To generate a report using a report design that is included in the Design Studio feature installation, select **Select a report design from the list**, and then select the report.
   - To generate a report using a report design that you have saved on a local file system, select **Select a report design from a file**, and then select the report.

4. Click **Next**.

5. In the Report Content area, do one of the following:

   - To generate a report by projects, select **Content by project** and then click **Select**. In the dialog box, select one or multiple projects to include and then click **OK**. When you select this option, all entities in the projects are included. You can refine the scope by using the **Project Dependencies** field.
   - To generate a report by entity, select **Content by entity** and then click **Select**. In the dialog box, select one or multiple entities to include, and then click **OK**. When you select this option, the scope begins from the selected entity and includes all entities related to the selected entity. You can refine the scope by using the **Project Dependencies** field.

6. In the **Project Dependencies** field, select one of the following:

   - **Include all content in referenced reports**: When generating a report by project, select this option to include all content in the selected projects as well as all content in all dependent projects. When generating a report by entity, select this option to include all related entities in the selected entity project and related entities in all dependent projects.
   - **Include only unsealed content in referenced reports**: When generating a report by project, select this option to include all content in the selected projects as well as all content in all unsealed dependent projects. When generating a report by entity, select this option to include all related entities in the selected entity project and related entities in all unsealed dependent projects. When you select this option, no content from sealed dependent projects is included.

     Oracle recommends that you select this option.

- **Exclude content in referenced reports**: When generating a report by project, select this option to include only the content in the selected projects, but no content from any dependent projects. When generating a report by entity, select this option to include only the related entities in the same project as the selected entity. When you select this option, no content from dependent projects is included.

7. In the **Output Format** field, select the format in which to generate the report.

   For installed Design Studio reference reports, this list is filtered to include only output formats that support the report features and that display the report layout properly. If you are generating a report from a local file, the list is not filtered. You must select an output format that supports the features and layout design.

8. In the **Action** field, indicate whether you want to view the report, save the report, or save and view the report.

   You can save reports in any available format. To view a report, you must have an installed version of an application associated with the output format type. For example, you cannot view PDF files if you do not have a PDF reader installed.

9. If you are saving the report, enter the name of the report and the location to which you want to save the report in the **Save As** field.

10. Click **Finish**.

Related Topics

Working with Reports

Contributing Documentation to Reports

Viewing the Report Design Example

Design Studio includes a report design example that you can use as a reference or as a starting point for creating your own custom report designs. This example is included in the Design Studio installation and can be added to your workspace. See the *Design Studio Developer's Guide* for information about all Design Studio example projects.

To view the report design example:

1. From the Design Studio **File** menu select **New**, and then select **Example**.

   The New Example wizard appears.

2. Expand the **Design Studio Examples** folder and select **Design Studio Report Design Example**.

3. Click **Next**.

   The Example Projects page appears, listing each of the projects that will be added to the workspace.

4. Click an example project to view its description:

   - **design.studio.example.report.update.site** creates a project to demonstrate how to export installable features into an update site.
   - **design.studio.example.report.design.feature** creates a project to demonstrate how report designs can be packaged into a feature for installation into Design Studio.
   - **design.studio.example.report.designs** creates a project that contains a sample report design, an XML Schema, a report design library, and other supporting content.

5. Click **Finish**.

The projects are added to the current workspace.