Oracle Applications Cloud
Extending the Applications

This guide also applies to on-premise implementations

Release 8

April 2014
Customizing the Springboard for Simplified Pages: Points to Consider .......................... 5-1
Menu Customization ........................................................................................................ 5-2

6 Customizing Reports and Analytics
Analytics Customization ................................................................................................. 6-1
Reports Customization ..................................................................................................... 6-6

7 Customizing Help
Help File Customization .................................................................................................. 7-1
Embedded Help Customization ....................................................................................... 7-12
This Preface introduces the guides, online help, and other information sources available to help you more effectively use Oracle Fusion Applications.

**Oracle Fusion Applications Help**

You can access Oracle Fusion Applications Help for the current page, section, activity, or task by clicking the help icon. The following figure depicts the help icon.

*Note*

If you don't see any help icons on your page, then click the Show Help icon button in the global area. However, not all pages have help icons.

You can add custom help files to replace or supplement the provided content. Each release update includes new help content to ensure you have access to the latest information. Patching does not affect your custom help content.

**Oracle Fusion Applications Guides**

Oracle Fusion Applications guides are a structured collection of the help topics, examples, and FAQs from the help system packaged for easy download and offline reference, and sequenced to facilitate learning. To access the guides, go to any page in Oracle Fusion Applications Help and select *Documentation Library* from the *Navigator* menu.

Guides are designed for specific audiences:

- **User Guides** address the tasks in one or more business processes. They are intended for users who perform these tasks, and managers looking for an overview of the business processes. They are organized by the business process activities and tasks.

- **Implementation Guides** address the tasks required to set up an offering, or selected features of an offering. They are intended for implementors. They are organized to follow the task list sequence of the offerings, as displayed within the Setup and Maintenance work area provided by Oracle Fusion Functional Setup Manager.

- **Concept Guides** explain the key concepts and decisions for a specific area of functionality. They are intended for decision makers, such as chief
financial officers, financial analysts, and implementation consultants. They are organized by the logical flow of features and functions.

- **Security Reference Manuals** describe the predefined data that is included in the security reference implementation for one offering. They are intended for implementors, security administrators, and auditors. They are organized by role.

These guides cover specific business processes and offerings. Common areas are addressed in the guides listed in the following table.

<table>
<thead>
<tr>
<th>Guide</th>
<th>Intended Audience</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common User Guide</td>
<td>All users</td>
<td>Explains tasks performed by most users.</td>
</tr>
<tr>
<td>Common Implementation Guide</td>
<td>Implementors</td>
<td>Explains tasks within the Define Common Applications Configuration task list, which is included in all offerings.</td>
</tr>
<tr>
<td>Functional Setup Manager User Guide</td>
<td>Implementors</td>
<td>Explains how to use Oracle Fusion Functional Setup Manager to plan, manage, and track your implementation projects, migrate setup data, and validate implementations.</td>
</tr>
<tr>
<td>Technical Guides</td>
<td>System administrators, application developers, and technical members of implementation teams</td>
<td>Explain how to install, patch, administer, and customize Oracle Fusion Applications.</td>
</tr>
</tbody>
</table>

**Note**
Limited content applicable to Oracle Cloud implementations.

For other guides, go to Oracle Technology Network at http://www.oracle.com/technetwork/indexes/documentation.

**Other Information Sources**

**My Oracle Support**

Oracle customers 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.

Use the My Oracle Support Knowledge Browser to find documents for a product area. You can search for release-specific information, such as patches, alerts, white papers, and troubleshooting tips. Other services include health checks, guided lifecycle advice, and direct contact with industry experts through the My Oracle Support Community.
Oracle Enterprise Repository for Oracle Fusion Applications

Oracle Enterprise Repository for Oracle Fusion Applications provides details on service-oriented architecture assets to help you manage the lifecycle of your software from planning through implementation, testing, production, and changes.

In Oracle Fusion Applications, you can use Oracle Enterprise Repository at http://fusionappsoer.oracle.com for:

- Technical information about integrating with other applications, including services, operations, composites, events, and integration tables. The classification scheme shows the scenarios in which you use the assets, and includes diagrams, schematics, and links to other technical documentation.

- Other technical information such as reusable components, policies, architecture diagrams, and topology diagrams.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/us/corporate/accessibility/index.html.

Comments and Suggestions

Your comments are important to us. We encourage you to send us feedback about Oracle Fusion Applications Help and guides. Please send your suggestions to oracle_fusion_applications_help_ww_grp@oracle.com. You can use Send Feedback to Oracle from the Settings and Actions menu in Oracle Fusion Applications Help.
Customizing and Extending Oracle Fusion Applications

Customizing and Extending Oracle Fusion Applications: Overview

While Oracle Fusion Applications provides robust ready-to-use functionality, there may be areas of the applications that you must change to meet your business needs.

You can:

- **Customize**: Change a standard (existing) artifact. For example, you can add an attribute to an existing object or you can change what is displayed on a standard page.
- **Extend**: Create a completely new artifact, such as a custom object.

For customizations and extensions, there are three basic scenarios:

- Personalization
- Run time customizations and extensions
- Design time customizations and extensions

What You Can Change

Oracle Fusion Applications is based on Oracle Fusion Middleware. Most user interfaces are implemented using Oracle Application Development Framework (Oracle ADF) and standard Java technologies. The foundation of the applications includes the service-oriented architecture (SOA) business processes. Business intelligence frameworks provide a number of reporting capabilities. Identity management works at every level to control access. Each of these areas of an application can be customized and extended to suit your business needs.

Additionally, Oracle Fusion Applications is built using a common data model. Because of this commonality, when you make a customization in one area, that customization is available to all objects in the application. For example, if you add an attribute to an object, you can easily add that attribute to the web-based view page, to an associated mobile page, and to any associated reports.

Generally, the tools and processes you use to customize one application are the same tools and processes to customize all applications.

Personalization: Explained

Personalization refers to the changes that every user of Oracle Fusion Applications can make to certain artifacts in the user interface (UI) at run
time. These changes remain for that user each time that user signs in to the application. Personalization includes changes based on user behavior (such as changing the width of a column in a table), changes the user elects to save, such as search parameters, or composer-based personalizations, where a user can redesign aspects of a page.

For composer-based personalizations, Oracle Fusion Applications includes Page Composer, which allows users to change certain UI pages to suit their needs. For example, they can rearrange certain objects on a page, and add or remove designated content.

### Run Time Customizations and Extensions: Explained

Run time customizations and extensions include those that a business analyst can make to Oracle Fusion Applications at run time using browser-based composers and other tools. These customizations and extensions are visible and usable by all or by a subset of Oracle Fusion Applications users. The types of run time customizations and extensions range from changing the look and feel of a page, to customizing standard objects, adding a new object and associated pages and application functionality, changing workflows, defining security for new objects, and customizing reports.

Access to run time customization tools depends on your roles. If you are assigned a role with an administrative privilege, you can access most run time customization tasks.

---

**Note**

Some customization tools are available only for specific product families.

With run time customization tools, you can:

- Personalize and customize the UI
- Customize menus
- Create and customize objects
- Create and customize business flows for custom objects
- Add custom attributes to business objects using flexfields
- Customize reports and analytics
- Customize help

### Personalizing and Customizing the UI

Both personalization and customization involve using Page Composer to make changes to an application page. For personalization, any user can drag and drop fields, rearrange regions, and add approved external content.

For customization, you also use:

- Page Composer to customize simplified and desktop pages for other users. You can add fields, add validation, change defaults, rearrange regions, add external content, and save queries. Page Composer allows you to work in a WYSIWYG view, and, in some cases, Source view.
- The Customize User Interface Text page to edit text that appears on multiple pages. For example, you can change the term supplier to vendor if that is your preferred term, and the change affects all pages where the term is displayed.
- The Settings functional area to change the:
  - Look and feel of simplified UI.
• Announcements on the simplified home page.

**Customizing Navigation**

Use the Manage Menu Customizations task in the Setup and Maintenance work area to customize the Navigator menu. You can also determine which dashboards to include on the home page (desktop UI only).

For simplified UI, use the Structure page in the Settings functional area to customize the springboard.

**Creating and Customizing Objects**

Application Composer lets you make more complex run time customizations to Oracle Sales Cloud applications. In addition to customizing pages, you can customize objects and all the artifacts that support them (such as fields, pages, buttons and links, security, server scripts, and saved searches), and can also create completely new objects and artifacts. For more information on defining objects, see Oracle Sales Cloud: Extending Sales.

When new objects are created, you often also create associated work area pages for those objects. Using the Manage Menu Customizations task, you can add those pages to the Navigator menu so that they can be accessed in the same way as standard objects.

When you create a new object in Application Composer, you can define security policies for it. A security policy defines the end user’s level of access to the data records of the object. For more information about securing custom objects, see Oracle Sales Cloud: Extending Sales.

**Creating and Customizing Business Process Flows for Custom Objects**

When you create a new object that is not a subclass of another object, you can also create a new object workflow to manage any business processes associated with it. For example, say you used Application Composer to create a marketing object and you want to create an associated approval flow. From within Application Composer, you can access Business Process Composer and create the process that defines that flow. For more information on using the Business Process Composer, see Oracle Sales Cloud: Extending Sales.

---

**Note**

Business Process Composer is not used in Oracle Fusion Human Capital Management.

**Adding Custom Attributes to Business Components Using Flexfields**

Most business components, except those in Oracle Sales Cloud products, support the use of flexfields to extend the object with custom attributes. Using a flexfield, you can create custom attributes without programming. The flexfield captures data that is related to a specific purpose, such as information about job positions or inventory items. Each attribute is a segment of the flexfield and corresponds to a pre-reserved column in the application database. Roles with administrative privileges provide access for viewing, configuring, and deploying flexfields using tasks available in the Setup and Maintenance work area. The configuration
of the flexfield is stored in the Oracle Metadata Services (MDS) repository and preserved across patches and upgrades.

**Customizing Reports and Analytics**

Oracle Fusion Applications comes with a complete set of reports. You can customize these reports (for example, change the layout) to fit your particular business needs. Additionally, if you customize or create a business object, you can create a new report for that object.

**Customizing Help**

If you have the appropriate job roles, then you can customize the help files in Oracle Fusion Applications Help. You can also determine which help files to show in specific help windows. Use the Manage Custom Help page to maintain both predefined and custom help files. You can open this page from any help window, or from the help site itself. Aside from help files, you can also customize help that appear on the UI, for example hint text.

**Design Time Customizations and Extensions: Highlights**

Design time customizations and extensions include complex changes that require deployment into the runtime environment. Design time customizations and extensions are most often done by developers and are not available in Oracle Cloud implementations.

**Design Time Customizations and Extensions Resources**

- Design time customizations are most often done using Oracle JDeveloper (a comprehensive integrated development environment), or other tools such as Oracle SOA Composer. Refer to the Oracle Fusion Applications Extensibility Guide for Developers.
  
  See: Design Time Customizations and Extensions
- You can determine what users can personalize.
  
  See: Configuring End-User Personalization
- You can also create a complete Java EE application and integrate it into Oracle Fusion Applications.
  
  See: Oracle Fusion Applications Developer's Guide

**Customization Layers**

**Customization Layers: Explained**

Oracle Fusion Applications contains built-in customization layers that allow you to make customizations that affect only certain instances or users of an
Before you create customizations, select the layer to which you want your customizations to be applied. Most of the tools that you use to create your customizations provide a dialog box where you can pick the layer for your customizations. You must be careful to choose the correct layer.

**Available Layers**

The exact customization layers available for an application depend on that application family. For information on product-specific customization layers, see assets with the Customization Layer type in Oracle Enterprise Repository for Oracle Fusion Applications (http://fusionappsor.oracle.com). Use product-specific layers appropriately as documented.

However, all applications have the following customization layers:

- **Site layer:** Customizations made in the site layer affect all users.
- **User layer:** The user layer is where all personalizations are made. Users do not have to explicitly select this layer. It is automatically selected when users personalize the application.

**Note**

If you are not given the option to choose a layer before you customize, then by default your customizations are made to the site layer.

**Layer Hierarchy**

These layers are applied in a hierarchy, and the highest layer in that hierarchy in the current context is considered the tip layer. With the default customization layers, the user layer is the tip layer. If customizations are done to the same object, but in different layers, at run time, the tip layer customizations take precedence. For example, if you customize in the site layer using Page Composer and hide a region, and a user personalizes the same page to have the region displayed, then the user layer will take affect for that user at run time.

**Where Customizations and Layer Information Are Stored**

Customizations you make are not saved to the base standard artifact. Instead, they are saved to an Extensible Markup Language (XML) file that is stored in an Oracle Metadata Services (MDS) repository. This XML file acts like a list of instructions that determines how the artifact looks or behaves in the application, based on the customization layer. The customization engine in MDS manages this process.

Because customizations are saved in these XML files, when Oracle Fusion Applications is patched or upgraded, the base artifacts can be updated without touching your changes. The base artifact is replaced, and when the application is run after the patch or upgrade, the XML files are simply layered on top of the new version. You do not need to redo your customizations.

**Customization Layers: Examples**

The following scenarios illustrate how customization layers work so that the correct customizations or personalizations are available at run time to the appropriate users.
For example, the Sales application has a layer for job role. When you customize an artifact, you can choose to make that customization available only to users of a specific job role, for example, a sales representative.

**Customization**

You want to customize the Sales home page by removing the Quick Create panel, but only for users with the Sales Representative role. Before you make your customization, you first select the layer in which to make your customization, in this case the role layer whose value is Sales Representative. When you make your customization by removing that pane from the page, an XML file is generated with the instructions to remove the pane, but only in the role layer, and only when the value is Sales Representative. The original page file remains untouched. The customization engine in MDS then stores the XML file in an MDS repository.

Now, whenever someone signs in to the application and requests an artifact, the customization engine in MDS checks the repository for XML files that match the requested artifact and the given context, and if there is a match, it layers the instructions on top of the base artifact. In this example, whenever the Sales home page is requested (the artifact) by someone who is assigned the role of Sales Representative (the context), before the page is rendered, the customization engine in MDS pulls the corresponding XML file from the repository, layers it on top of the standard Sales home page, and removes that pane. Whenever someone who is not a Sales Representative signs in (for example, someone with the role of Sales Manager), the XML file with your changes is not layered on top, and so the Quick Create panel is displayed.

This figure shows how the customization XML file is applied to the base document and is visible only to a sales representative.

**Personalization**

All users of Oracle Fusion applications can personalize certain pages using the Personalization menu. Users can move elements around on a page, hide
elements, and even add available elements to their page. When they do this personalization, the customization engine in MDS creates an XML file specific to that user, for the user layer.

For example, say User 1 (who has the role of Sales Representative) personalizes the Sales home page. There will then be an XML file stored in the repository, noting the changes that user made. When User 1 signs in, as in the previous example, the customization engine in MDS pulls the XML file with the sales representative customizations from the repository and layers it on top of the standard Sales home page. In addition, the engine pulls the XML file with the User 1 personalizations, allowing the user to see the personalization changes along with the Sales Representative changes. When other Sales Representatives log in, they do not see the User 1 personalization changes, as shown in this figure.

Selecting Customization Layers to Include: Examples

When you use the dialog box to select which customization layer to customize, you can also include lower layers, to view customizations from those layers while you customize.

The following scenarios explain what happens based on your selected layers. For these examples, the available layers are Site, Territory, and Job Role.

What You See While Customizing

You choose to edit the Job Role layer and select Sales Representative as the value for that layer. You also choose to include the Territory layer and select Southwest as the value. The Site layer is automatically included because it applies to everyone.

While you are customizing in Page Composer, you see customizations that apply to sales representatives in the Southwest territory, based on what was defined
What Your Customizations Apply To

No matter what you see while customizing, the customizations you are making apply only to the edit layer, Job Role.

For example, a field is hidden in the Site layer but displayed in the Territory layer for Southwest. No customization exists for the field in the Job Role layer for Sales Representative.

Because Territory is higher than Site, you see the field displayed while you are customizing in Page Composer. However, you choose to hide the field as part of your customization, in which case, that customization applies to the Job Role layer, for sales representatives.

Users with other job roles in the Southwest territory might still see the field. However, because Job Role is a higher layer than Territory, the field is hidden for all sales representatives in any territory, unless a layer higher than Job Role applies to any of these users and has the field displayed.

What You Can Customize and Extend and with Which Tool: Explained

There are many scenarios for which you can customize and extend Oracle Fusion Applications. The following tables identify for each scenario the artifacts that you can customize or create in Oracle Fusion Applications, and what tool you use:

The following tables identify for each scenario the artifacts that you can customize or create in Oracle Fusion Applications, and what tool you use:

- Page customization
- Branding customization
- Object customization
- Business process customization
- Security customization
- Business intelligence customization
- Help customization

Note

- Presented in the following tables are the top customization tasks, not a comprehensive list.

- Application Composer is available only if you want to make changes in the following Oracle Sales Cloud applications:
• Marketing
• Sales
• Customer Center
• Trading Community Architecture (TCA)
• Order Capture

• Application Composer and other Oracle Sales Cloud customizations and extensions are described in Oracle Sales Cloud: Extending Sales.

The following tables list the types of customizations and extensions that business analysts can make. For more information about design time customizations and extensions by developers and administrators, see the Oracle Fusion Applications Extensibility Guide for Developers.

Note
Design time customizations and extensions are not available in Oracle Cloud implementations.

Page Customization

This table shows some types of customizations you can make to pages and the corresponding tools to use.

Note
While you can customize pages in Page Composer and Application Composer, only certain pages are configured to allow it. If the customization that you want to make is not available in Page Composer, then developers can use JDeveloper to make the customization (not available in Oracle Cloud implementations).

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add, move, delete, show, or hide components on a page.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Change a page layout.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Create a site-level search for all users.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Customize a page title.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Customize a task list menu.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Customize dialog box content.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Add fields, buttons, and links to a standard page (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Customize attributes for a flexfield on a page.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Customize properties for user interface (UI) components on a standard page.</td>
<td>Page Composer</td>
</tr>
</tbody>
</table>
Customize properties for UI components on a standard page (Oracle Sales Cloud).

- **Tool**: Application Composer

Customize the UI Shell template.

- **Tool**: Page Composer

Customize a text string wherever it appears across all pages.

- **Tool**: Customize User Interface Text page

Customize the look and feel of simplified pages.

- **Tool**: Appearance page in the Settings functional area

Change the announcements on the simplified home page.

- **Tool**: Announcements page in the Settings functional area

## Branding Customization

This table shows some types of customizations you can make to use your own branding logo, and the corresponding tools to use.

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customize the UI Shell template.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Change the logo and application name in simplified UI</td>
<td>Appearance page in the Settings functional area</td>
</tr>
<tr>
<td>Customize report layouts.</td>
<td>Oracle BI Publisher</td>
</tr>
</tbody>
</table>

## Object Customization

This table shows some types of customizations you can make to objects and the corresponding tools to use.

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customize objects (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Add an attribute to a business object using flexfields (not Oracle Sales Cloud).</td>
<td>Setup and Maintenance work area</td>
</tr>
<tr>
<td>Create objects (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Add a business object page to the Navigator menu.</td>
<td>Setup and Maintenance work area</td>
</tr>
<tr>
<td>Add custom object work area pages to the Navigator menu (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Add validation to an object (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Customize saved searches for a custom object (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Customize object workflows for an object (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Create object workflows for an object (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
</tbody>
</table>

## Business Process Customization

This table shows some types of customizations you can make to business processes and the corresponding tools to use.
<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a BPMN process in a BPM project.</td>
<td>Business Process Composer</td>
</tr>
<tr>
<td>Create a BPMN approval process in a BPM project (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Customize custom BPM projects.</td>
<td>Business Process Composer</td>
</tr>
<tr>
<td>Customize custom BPM projects (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
</tbody>
</table>

**Security Customization**

This table shows some types of security customizations you can make to objects and the corresponding tools to use.

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add data security to a custom object.</td>
<td>Setup and Maintenance work area</td>
</tr>
<tr>
<td>Grant access to custom objects (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Enable elevated privileges customization.</td>
<td>Application Composer</td>
</tr>
</tbody>
</table>

**Business Intelligence Customization**

This table shows some types of customizations you can make to business intelligence analytics and reports, and the corresponding tools to use.

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create report layout.</td>
<td>Oracle BI Publisher</td>
</tr>
<tr>
<td>Customize report layouts.</td>
<td>Oracle BI Publisher</td>
</tr>
<tr>
<td>Customize style templates.</td>
<td>Oracle BI Publisher</td>
</tr>
<tr>
<td>Create a report.</td>
<td>Oracle BI Publisher</td>
</tr>
<tr>
<td>Translate a report.</td>
<td>Oracle BI Publisher</td>
</tr>
<tr>
<td>Create a report subject area (Oracle Sales Cloud).</td>
<td>Application Composer</td>
</tr>
<tr>
<td>Customize analytics.</td>
<td>Reports and Analytics pane</td>
</tr>
</tbody>
</table>

**Help Customization**

This table shows some types of customizations you can make to help and the corresponding tools to use.

<table>
<thead>
<tr>
<th>Customization</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customize text that is displayed when the user hovers over a button, link, icon button, or tab title.</td>
<td>Page Composer</td>
</tr>
<tr>
<td>Customize help files and determine the help links to show on help windows.</td>
<td>Oracle Fusion Applications Help</td>
</tr>
</tbody>
</table>
All customizations and extensions to Oracle Fusion Applications should be done in a full test environment. Typically, this environment contains one or more Oracle Fusion applications that will then be moved to a production environment after all customizations and extensions are complete and tested.

Business analysts using Page Composer and Application Composer can make application customizations in a sandbox. Sandboxes store the customizations in Extensible Markup Language (XML) files in a separate Oracle Metadata Services (MDS) repository that is available only when you work in that particular sandbox. The changes can be done in a test-only sandbox (that is, the code in the sandbox is for testing only, and is never deployed), or they can be done in a sandbox that is then published to the full test environment.

Developers using design time tools, such as Oracle JDeveloper, can deploy their customizations directly to that environment, or they can publish to a sandbox. For more information on design time customization workflow, see the Oracle Fusion Applications Extensibility Guide for Developers.

Project managers can monitor, import, and export customizations. The entire environment with all customizations can then be tested, as shown in the figure below.
Tip

Depending upon the need, developers may allow users having access to the Oracle Fusion Functional Setup Manager, to configure the customizations and extensions made to Oracle Fusion Applications.

Run Time Customizations

Run Time Customization Workflow: Explained

When you use Application Composer and Page Composer to make run time customizations to Oracle Fusion applications, you can use sandboxes to save your changes in a segregated environment. For example, before you begin making customizations, you create a sandbox named MySandbox and make your customizations in that sandbox. If others want to see the customizations, then they would use MySandbox.

Note

There are restrictions when more than one user works in a sandbox.

You can also use a sandbox when you define security policies for custom objects that you have created using Application Composer. A security sandbox stores the security information in new database tables that are available only when you choose to work in that sandbox.
After you complete your customizations, the sandbox can be reviewed and approved by others, and then published to the full test environment where your customizations become part of that repository.

**Note**
A flexfield sandbox is for testing only and cannot be published. Instead, you deploy a flexfield to the full test environment using the flexfield UI. To test a flexfield configuration before deploying it to the full test environment, deploy it to a flexfield sandbox. The changes that you deploy to a sandbox are isolated from the full test environment and can be seen only by those who make the flexfield sandbox active in their session. After you are satisfied with the changes in the sandbox, you can deploy the changes to the full test environment.

When you publish a sandbox, the published customizations are labeled. Labeling can act as a save point, meaning that if a future customization causes issues, you can use the Manage Customizations dialog box to remove all customizations done after that point by promoting the last known good label back to the tip.

You can also use the Manage Customizations dialog box to view others’ customization metadata files, and to download those files to manually move them to another environment or to diagnose any issues. You can also upload others’ customization metadata files to your environment.

**Note**
The navigator menu and report customizations do not use an Oracle Metadata Service (MDS) repository.

This figure illustrates the use of sandboxes when customizing pages, objects, and security using Page Composer and Application Composer and when configuring flexfields.
Viewing and Diagnosing Run Time Customizations: Points to Consider

Use the Manage Customizations dialog box to view and diagnose run time customizations that have been made to application pages. Customizations are role-dependent and by default, the Manage Customizations dialog box displays the customizations that have been performed by the signed-in user.

If you are unable to display the page that contains the customizations, choose **Manage Customizations** from the **Administration** menu, and then, in the Manage Customizations dialog box, type the page, page fragment, or task flow in the **Search** text field, and click the **Search** icon.

---

**Note**

Before you begin viewing customizations, ensure that you have administrative privileges to access the Manage Customizations dialog box.

Customizations for a user are visible under the **Current Context** column on the Manage Customizations dialog box.

---

**Tip**

After you are in the Manage Customizations dialog box, you can change the page, page fragment, or task flow for which you are viewing customizations using the **Search** field.

Developers too may be assigned to specific roles and can view only those customizations that are permitted for that role. However, administrators can view all the customizations made at the site level, and for any user, under the **All Layers** column on the Manage Customizations page. To view customizations made by more than one user, administrators can select multiple users.

Sometimes, an administrator might need to view a personalization that was made by another end user. For example, a user might have made an error while personalizing a page and that page is no longer displayed for the user. Because the user cannot access the page, the user cannot correct the error. In this case, the administrator can access the page, request to see the user’s changes, and delete those changes to restore the page to its original settings.

**Logging Page-Level Customizations**

You can use logging that is applied to a page or you can use the Manage Customizations dialog box to diagnose customization issues to determine whether customizations have been applied to a page.

If you suspect that a problem might have been caused by a customization on a page, such as a user interface component disappearing from a page, you can export the page’s customizations and examine the document file.

To turn on run time logging for customizations that are applied to a page, set the log level for the **oracle.mds.custmerge** module to **FINEST**. You can set the application’s log level by choosing **Troubleshooting** from the **Help** menu. You might need to ask your administrator to give you privilege to set the log level.

If you have administration privileges, you can also use Fusion Applications Control to set the log level.
Managing Customizations Using Sandboxes: Explained

Different types of customizations can be applied to an application such as changes to an application’s metadata stored in the metadata services repository or changes related to data security of the application. All such customizations are stored in sandboxes and are validated before applying them to an application.

Environment
To customize an Oracle Fusion application in run time, you must first create a sandbox and then use Page Composer or Application Composer to make the customizations. These changes remain within the sandbox and do not affect the mainline code. You can test and validate the changes by publishing the sandbox to a full test environment. After the application has been tested, it can then be moved to the production environment. The customizations created in the sandbox will be migrated to the production environment and will be available to the users of the system.

It is recommended that customizations are never made directly to the mainline code and have to be first done in the sandbox. You make changes to an application at run time in a sandbox so that the changes are isolated from the mainline code. The mainline code is a branch of data that serves as a single source. After the changes in the sandbox are complete and verified and you want to commit them, you can publish the metadata or security-enabled sandbox to the mainline code. If customizations existed in the mainline code before the sandbox was created, you will see the customization information in the sandbox.

To view the customizations that were newly added to the mainline code, you will need to exit, publish, or delete the sandbox and create a new sandbox. Flexfield sandboxes are for testing only and cannot be published. You make flexfield configurations that are stored in a database, and then deploy those configurations to a sandbox to see the resulting deployment artifacts in a sandbox environment. Flexfields are deployed directly to the mainline code using the flexfield UI.

Tools
You can use several run time tools to customize the application. The sandbox manager works with Application Composer and Page Composer to customize objects and pages. The Oracle Business Process Composer and Oracle SOA Composer are also run time customization tools, but they do not use the sandbox manager. They have their own mechanisms for handling customization changes.

For on-premise implementations, a metadata sandbox that you create using the sandbox manager is available in JDeveloper when you are creating and deploying customizations intended for a deployed Oracle Fusion Application in Oracle WebLogic Server. The available sandboxes except security sandboxes appear in a selection list in JDeveloper during deployment.

Note
For cloud implementations, you can use Application Composer to work with sandboxes. For more information on using sandboxes in Application Composer, refer to the Oracle Sales Extensibility Guide.
The metadata sandbox sessions can be saved, downloaded, and imported as files into other Oracle Fusion applications.

**Using Sandboxes: Points to Consider**

In the customization run time workflow, you use sandboxes to isolate the changes from the mainline code for testing and validating. After you are satisfied with the changes, you can publish the changes back to the mainline code. You can create two types of sandboxes:

- Sandboxes that are intended for testing purposes only.
- Sandboxes that are intended to be published.

The testing sandboxes are never published and therefore produce no concurrency conflicts between sandboxes. You can have several testing sandboxes at the same time. But if you have multiple users working on the same testing sandbox, then they must adhere to the prescribed guidelines. Customizations in the sandboxes that are published are merged back to the mainline code. The following figure illustrates the two types of sandboxes and their relationship to the mainline code.

**Working with a Single Sandbox**

When multiple users are customizing an application using the same sandbox at the same time, conflicts within a sandbox can arise. It is because more than one user may be attempting to customize the same artifact or performing a customization task that indirectly affects other shared files. An example of a direct conflict is when different users attempt to customize the same page, the same fragment, or the same metadata file within the same layer. An example of an indirect conflict is when two users, each creating their own object, cause a conflict in the metadata file that tracks which new objects have been created by both saving their changes around the same time.
Conflicts may also arise when users are editing a shared artifact, such as when a user performs an operation that adds or edits a translatable string. For example, a user edits a field’s display label or help text, or a validation rule’s error message, while another user performs an operation around the same time that similarly affects translatable strings. Another example of a shared artifact conflict is when two or more users are working in navigator menus that are shared across applications. Whenever there is a customization conflict with another user, the application displays concurrency warning messages.

Whether the sandbox is meant for testing or production, if multiple users work with a single sandbox, follow these guidelines to avoid conflicts.

- Multiple concurrent users in the same sandbox must operate only on different and unrelated objects.
  For example, if user1 updates object1, then user2 can update object2 but should not update object1. Remember that if both modifications involve changes to translatable strings, then saving changes to separate objects around the same time may still cause a conflict in the resource bundle that stores the translatable strings.

- If multiple users update the same artifact concurrently (either the same object or the same underlying frequently modified file), then they will get a concurrent update error. In this case, the second user’s updates will not be saved (the Save button will be disabled) and one of the users will have to cancel and try again.

**Working with Multiple Sandboxes**

Multiple sandboxes are used when customizations are stored in testing as well as production sandboxes.

If there is a concurrent change made in the mainline code after the sandbox was created and the user attempts to publish that sandbox, then such conflicts are detected at publication time and error messages are displayed.

**Tip**

If you encounter a message showing a conflict on oracle/apps/menu/fnd/applcore/dataSecurity/dataSecurityService/mds/DSMO.xml when you publish your sandbox, it indicates that the security changes that you made in your sandbox conflict with other security changes in the mainline code. Delete the sandbox and recreate your changes in a new sandbox.

If multiple users are permitted to work in multiple sandboxes at the same time, follow these guidelines to avoid conflicts:

- There can be any number of test-only sandboxes operating concurrently. That is, multiple users can use multiple sandboxes concurrently for testing if these sandboxes will never be published. Sandboxes that are used for testing only, and that are not published, cause no conflicts with each other. Be aware, however, that all modifications will be lost when the sandboxes are destroyed.

- For sandboxes that are not for test-only and will be published, multiple concurrent sandboxes can be used only if they operate on mutually exclusive artifacts. For example, you can have one sandbox that contains a page that is being customized to add a task flow, and another sandbox that contains a different page from a different application.
• If an artifact is updated in both the mainline code and in one sandbox or in two different sandboxes, when the sandbox is published, such conflicts are detected and an error is displayed. At this point, cancel publishing the sandbox to avoid overwriting previous changes.

Note
For a sandbox that contains ADF Business Components customizations, sign out and sign in again after switching in or out of this sandbox to avoid any inconsistencies between the run time caches and the ADF Business Components definitions.

Setting Up Sandboxes: Procedures

To make customizations to the application artifacts you need to first store them in an active sandbox. You can either create a sandbox or select an existing sandbox, and designate it as an active sandbox. The active sandbox holds the context for all the changes. The sandbox uses a database to store the actual changes. After you examine the changes and are certain about them, you can publish the sandbox, or deploy the flexfield, and the changes are merged into the mainline code. Eventually, the sandbox is archived.

Note
The following procedure is for setting up non-flexfield sandboxes. For flexfields, use the Manage Descriptive Flexfields task or the Manage Extensible Flexfields task.

To create a new sandbox and set it up, perform the following steps:

1. In the global area of the Oracle Fusion Applications, select Administration - Manage Sandboxes.

2. On the Manage Sandboxes dialog box, create a new sandbox.

3. To make it a security-enabled sandbox, select the Create Data Security Sandbox check box on the Create Sandbox dialog box. If it is not required, skip this step.

Note
Setting up the security sandbox requires duplicating the schema for Oracle Fusion Data Security tables. Therefore, this will always be a lengthy operation in Application Composer. Allow sufficient time for the process to be completed and do not terminate it in a hurry. You may want to defer customizing security and enabling the security sandbox until you are sure that customizations are required.

4. Click Save and Close.

5. On the Manage Sandboxes dialog box, select the newly created sandbox or an existing sandbox and click Set as Active. The sandbox is designated as the active sandbox.

6. Perform the following optional tasks, if required:
• To export the sandbox to a file so that it can be transported or shared, click the sandbox. On the Sandbox Details dialog box, click **Download All**.
• To import a sandbox from a file, click Import, upload the file and click **OK**.

7. Close the Manage Sandboxes dialog box.

**Publishing Sandboxes: Procedures**

After you have made the customizations in the sandbox, you need to publish them to make them available in the application. Before publishing the customizations, test or validate the changes at run time using test systems and any combination of the validation setups.

If there are changes to the mainline code from another source and you publish your sandbox data, then the mainline code is not overwritten. However, if there are conflicts you are notified, and you need to fix the conflicts before publishing.

To publish a sandbox:
1. In the global area of the Oracle Fusion Applications, select **Administration - Manage Sandboxes**.
2. On the Manage Sandboxes dialog box, select the sandbox and click **Publish**. The Publish confirmation message box appears.
3. Click **Yes**. The sandbox is published to the mainline code.
4. Close the Manage Sandboxes dialog box.

**Moving Customizations**

**Using Customization Migration to Move Customizations: Points to Consider**

The Customization Migration page enables you to create a set of all customizations and extensions that have been made to an Oracle Fusion Applications environment, download that customization set, and upload it into another environment. The customization set includes customizations across all Oracle Fusion Applications product families, such as Oracle Fusion Financials, Oracle Sales Cloud, and Oracle Fusion Human Capital Management (Oracle Fusion HCM). The customization set includes only the customizations and extensions that you make using the following tools and features. Personalizations are not included in the set.

To access the Customization Migration page, navigate to **Tools - Customization Migration** from the Navigator menu.

**Contents of the Customization Set**

The customization set includes only the customizations that you make using the following tools, features, or tasks:
• Application Composer, with the exception of the following customizations:
• Object artifacts that were generated from the Import and Export page in Application Composer to make extensions available for importing and exporting.
• User names and passwords for secured SOAP web service connections.
• The enabled attachment feature for custom objects.
• Page Composer
• Content developed using Oracle Business Intelligence Enterprise Edition features including but not limited to Oracle Business Intelligence Answers, Oracle Business Intelligence Delivers, Oracle BI Composer, and Oracle Business Intelligence Interactive Dashboards.

Note
You can move these customizations only if the Disable BI for CSM profile option is set to No.

• Tasks and dialog boxes for configuring descriptive and extensible flexfields and value sets
• Manage Menu Customizations task
• Manage Oracle Social Network Objects task
• Manage Standard Lookups task
• Manage Common Lookups task
• Security settings and changes made in the Application composer including associating privileges to duty roles, which provide access to custom objects are migrated and overwritten to the target. Enterprise roles, new duty roles, and role hierarchy changes, which are made directly in Oracle Authorization Policy Manager (Oracle APM) will not be migrated.
• For non-Cloud implementations, customization metadata that is created from JDeveloper using the Oracle Fusion Applications Administrator Customization role and then packaged and deployed to the source Oracle Fusion Applications environment.

Warning
While an upload or restore activity processes Presentation Services toolset customizations, the following can occur:
• Reports that were submitted by Oracle Enterprise Scheduler to Oracle BI Publisher and were scheduled to execute during the process, will fail.
• The Reports and Analytics pane does not display.
• Oracle Business Intelligence Publisher reports may not display on Oracle Business Intelligence Presentation Services analyses or dashboard pages.
• Users may not be able to access Oracle Business Intelligence Enterprise Edition features, such as Oracle Business Intelligence Answers, Oracle Business Intelligence Delivers, Oracle BI Composer, and Oracle Business Intelligence Interactive Dashboards.

Note
The Customization does not include code extensions, such as managed beans, that you implement in JDeveloper using the Oracle Fusion Applications Developer role. These code extensions are stored in the app-inf/lib and web-inf/lib directories and you must manually move the extensions.
The Customization Migration page preselects the type of customizations across all applications that will be added to the customization set. You cannot change this selection.

**Tip**
To prevent in-progress customizations from getting included in the customization set, make your customizations in a sandbox. The customization set does not include customizations that are in sandboxes.

While you can use the Customization Migration page to move customizations and extensions from any source environment to any target environment, you should always perform your customizations and extensions in a full test environment and use the Customization Migration page to move these changes to a production environment. As customization set migration does not provide a merge capability, never customize or extend a production environment.

When you import a customization set, the rows in the database that are not preconfigured are updated if a matching record exists. Otherwise a record is inserted.

**Note**
The customization set does not include all deletions. For example, the set does not include the removal of a customization document using the Manage Customizations dialog box. After you import a customization set into the target environment, you must examine the environment for any deletions that you must make manually. Similarly, the customization set does not include roles or role hierarchy changes. Changes made to Oracle APM have to be manually updated in the target environment.

**Creating and Applying Customizations Using a Customization Set: Procedures**

Create a customization set to move customizations across all the product families of Oracle Fusion Applications from one environment to another environment. Creating a customization set is beneficial in moving customization in a batch instead of moving customizations one by one.

You need to ensure the following before creating a customization set:

- The source and target Oracle Fusion Application environments are of the same release and that the same standard and one-off patches have been applied to both environments.
- All Application Composer, Page Compose, and JDeveloper customizations made in sandboxes are complete before they are published. All complete customizations must be published before the export process begins.
- All customizations and extensions made using the Manage Menu Customizations task, the Manage Standard Lookups task, and the Oracle Authorization Policy Manager, are complete.
- To move content that was created using Oracle Business Intelligence Enterprise Edition features, ensure that the Disable BI for CSM profile option is set to No in both the source and target environments. To view this profile option, access the Setup and Maintenance work area in the global area of Oracle Fusion Applications and search for the Manage Profile Options task.
- You have been granted access to the FND_CUSTOMIZATION_SET_MANAGEMENT_DUTY role, which
enables you to access the Customization Migration page. Contact your security administrator for details.

• Users never make customizations in the target or production environment while applying customizations.

Note
If users must make customizations to the production environment in emergency circumstances, remember to make the same customizations to the test environment to ensure that they are included in the next customization migration.

• Users do not perform customizations in the source environment during the export process.

To create customizations:

1. In the source environment, from the Navigator menu, select Tools - Customization Migration .
2. From the Outgoing tab of the Customization Migration page, click Create Customization Set.

Tip
If the Delete button appears for an existing customization set, click the button. This removes the temporary files that are on the server from the previous customization set creation. You will not be able to create a customization set until the previous set has been deleted.

3. Provide a name for the customization set.
4. Optionally, type a description of the set.
5. Click Save and Close.
6. Periodically, click Refresh to view the current status of the set creation. Eventually, the status changes to Ready for Download.

Tip
To see the detailed status of each customization type, expand Customization Details.

Note
The process runs asynchronously. You can exit the dialog box and return to it at a later time.

7. Click Download and specify the name and location for the file that will be created (ensure that the downloaded file is a JAR file), and click Save.
8. After the file is successfully downloaded on your local file system, click Delete to remove the temporary files that were created on the server.

To apply customization to the target environment:

Tip
Apply customizations when a few people are signed into the environment, because end users must sign out and sign in again to see any changes that are made.

1. Open the Customization Migration page in the target environment.

2. From the Incoming tab, click Browse, specify the name and location of the customization set file, and click Open.

**Note**
If the Browse button appears disabled, click Delete to remove the previously uploaded customization set from the environment and enable the Browse button.

3. When the status for the customization set is Ready to Apply Customizations, click Apply.

4. Periodically, click Refresh to view the current status of the Apply action.

**Tip**
The process runs asynchronously. You can exit the dialog box and return to it at a later time.

**Note**
For Oracle Cloud implementations, if problems occur during an Apply action, log a service request using My Oracle Support at https://support.oracle.com.

5. Access the target environment and examine the environment for any deletions that you must manually make.

6. Deploy all flexfields that display a Patched status.

7. Perform the following steps to send the new and updated Oracle Social Network definitions to the Oracle Social Network server:
   a. In the Setup and Maintenance work area, access the Manage Oracle Social Network Objects task.
   b. For an object that was created or updated as part of the applying customizations process, if its Enabled value is anything other than No, trigger the process of sending its definition to the Oracle Social Network server. You can do it by disabling the object and enabling it again with its original status. For example, if the Enabled value is Manual, disable the object, enable the object, and select the value Manual.
   c. Click OK and save the changes.

8. If you applied Oracle Sales Cloud application customizations, you must perform the following tasks from Application Composer to complete the movement of Oracle Sales Cloud customizations and extensions:
   - Generate the artifacts that are required to register the migrated extensions. Click **Import and Export** in the Common Setup region.
and click **Generate**. This makes the migrated extensions available for importing and exporting.

- From the Common Setup region, click Web Services and complete the following substeps for every secured SOAP web service connection that uses either the **Invoke with separate user credentials over SSL** or the **Invoke with separate user credentials and message protection** authentication schemes to manually migrate the user names and passwords:

  1. Make a note of the name and WSDL URL for the web service and delete the web service.
  2. Click **Create** to recreate the secured web service.
  3. In the Create SOAP Web Service Connection dialog box, type the name and WSDL URL for the web service. The details must match with those of the web service that you deleted.
  4. Click **Read WSDL**.
  5. Ensure that the appropriate authentication scheme is selected.
  6. Click **New** for the Credential Key field.
  7. In the Create Key dialog box, type the credential key, user name and password, and click **OK**.
  8. Click **Save and Close**.
  9. If attachments were enabled on run time pages using Applications Composer, complete the following steps to enable attachments for those pages in the target environment:

    a. In the source environment, in the Setup and Maintenance work area, access the Manage Implementation Projects task.
    b. On the Manage Implementation Projects page, click **Actions - Create**.
    c. On the Enter Basic Information page, either accept the default values or replace them with the required values.

**Tip**

The following points must be adhered to:

- Provide a meaningful name for the project. The code and description fields change automatically.
- If you modify the Code value, replace it with a unique code.
- Use the default value for the Start Date and leave the Finish Date blank.
- The user in the **Assigned To** field must have access to Customization Set Migration functionality.

  d. Click **Save and Open Project**.
  e. On the Implementation Project page, click **Actions - Select and Add**.
f. Search for the Manage Attachment Categories and Manage Attachment Entities tasks, click Apply, and click Done.

g. In the Tasks region, click Manage Configuration Packages.

h. On the Manage Configuration Packages page, click Actions - Create.

i. In the Source Implementation Project region, select the implementation project that you created.

j. Click Next and on the consequent page, click Next.

k. On the Create Configuration Package: Schedule page, click Submit. On the warning message box, click Yes to proceed with the export process.

Note

Periodically click Refresh to display the status.

l. After the export process is complete, download the configuration package.

m. In the target environment, create a similar implementation project and a configuration project and click Save and Close.

n. On the Manage Configuration Packages page, select the configuration package that you just created and click Upload.

o. Select the configuration package that you downloaded from the source environment and click Open.

p. Click Details and click Submit.

q. Select the configuration package, click Import Setup Data and click Submit.

Note

Periodically click Refresh to display the status.

r. After the import is complete, click Done.

10. As an optional step, identify and reschedule any Oracle Sales Cloud processes that failed while applying customizations. To do this, from the Navigator menu, select Tools - Scheduled Processes and look for processes with the following statuses:

• Blocked

• Error

• Error Auto-Retry

• Error Manual Recovery

• Paused

• Validation Failed
11. Manually migrate all business processes that were created in the source environment to the target environment. Alternatively, you can export them from the Process Composer. To access the Process Composer, navigate to the Application Composer and select the business process that you want to migrate.

12. When the customizations are successfully applied, perform functional testing to verify the changes. If testing exposes problems with the customizations, such as importing more than you intended, or the changes were not what you expected, access the customization set in the Incoming tab of the Customization Migration page, and click Restore to revert to the state before the customization set was applied. In such cases, skip the next step.

**Note**

Users can monitor the progress of the download or the applying process by viewing the process log. This process takes approximately 15 minutes. If it takes any longer and you do not see any progress, click Refresh. You can either let the server take its time and click Continue or click Restart to restart the export process.

**Important**

- After an environment upgrade, any previous imports which were performed in an earlier release cannot be reverted. However if a new import is submitted in the upgraded instance, then the most recent import can be reversed.

- Lookup values for lookup fields that exist in both source and target are not overwritten during the customization import. The lookup values from source are added to the target and all the lookup values coexist for the same field. For example: Status field in its source environment has values Open, Closed. In the target environment it has values Yes, No. After the import, the Status field in target environment has values Open, Closed, Yes, and No.

- After the import, perform the following steps in the target environment to send the new and updated Oracle Social Network definitions to the Oracle Social Network server.

  1. In the Setup and Maintenance work area, access the Oracle Social Network Objects task.
  2. On the Oracle Social Network Objects page, click Synchronize to synchronize a selected object or click Synchronize All to synchronize all objects together.

- During customization import, the data security privileges are not automatically revoked in the target environment. For example, if a specific privilege is granted in the target environment but the corresponding privilege does not exist in source environment, during import, the privilege in the target environment will not be automatically revoked. To address this manually, add such a privilege to the source environment and revoke it. The revoke action is picked up...
as a customization instance during the customization import process and applied to the target environment.

**Note**
- You can create custom reports directly in the target environment. However, ensure that you create the custom reports and reference them to the already existing custom subject areas (do not create the custom subject areas directly in the target environment).
- Customization export and import tasks can only be initiated from the mainline code. If they are initiated from a sandbox, the process does not execute.

**Caution**
All user personalizations that are performed after a customization set is applied are lost when you perform a restore action on that customization set.

13. Broadcast information to the users that they must sign out and sign in to view the most recent changes.

**Exporting and Moving Customizations: Points to Consider**

Customizations are stored in XML files. This enables you to export customizations for the following reasons:
- To move customizations and extensions to another Oracle Fusion Applications environment, such as the production environment.
- To diagnose issues noticed in the test environment.
- To send files to Oracle Support Services for further diagnosing.
- To import a customization into another environment. For example, a customization developer using JDeveloper might need to see customizations done by someone else.

Exporting and importing customization files also help in backing up and restoring customizations.

The following table lists the tools to use to export and move customizations and extensions.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Tools to Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move all customizations to another Oracle Fusion Applications environment.</td>
<td>Customization Set Migration.</td>
</tr>
<tr>
<td>Move only MDS customizations made to pages and the user interface to another Oracle Fusion Applications environment.</td>
<td>Oracle Enterprise Manager Fusion Applications Control (Fusion Applications Control).</td>
</tr>
</tbody>
</table>

**Note**
You can also use Fusion Applications Control to download and upload a set of customizations.
Move only descriptive flexfield configurations to another Oracle Fusion Applications environment.

Oracle Fusion Functional Setup Manager (Functional Setup Manager). Moves configurations for a specified module. To move configurations for all modules, use Customization Set Migration.

Move only extensible flexfield configurations to another Oracle Fusion Applications environment.

Functional Setup Manager. Moves configurations for a specified module. To move configurations for all modules, use Customization Set Migration.

Move only value set configurations to another Oracle Fusion Applications environment.

Functional Setup Manager. Moves configurations for a specified module. To move configurations for all modules, use Customization Set Migration.

Move only lookups to another Oracle Fusion Applications environment.

Functional Setup Manager. Move application standard lookups, application common lookups, or both.

Move only data security policies to another Oracle Fusion Applications environment.

Functional Setup Manager.

Note

It does not move Oracle Fusion Human Capital Management roles.

Export customizations to a file to help diagnose an issue.

Manage Customizations dialog box.

Export customizations to import them into an application workspace in JDeveloper.

Manage Customizations dialog box.

Note

Fusion Applications Control is not available in Oracle Cloud implementations. Therefore, in Oracle Cloud implementations, to perform tasks that require use of Fusion Applications Control, log a service request using My Oracle Support at https://support.oracle.com.

Downloading Customizations

On the Manage Customizations dialog box, from the Administration - Manage Customizations menu, you can download customization files for a given page. You can use these files for diagnosing customization issues.

You can also download all customizations of the page for all layers (AllCustomization.zip), using the Download Customizations for All Layers link located at the bottom of the dialog box. The file contains all the customization XML files for the page.

Managing Customizations: Points to Consider

The Manage Customizations dialog box displays the customizations of the task flows in a page. You can access the Manage Customizations dialog box from Page Composer and from the Administration menu in the global area of Oracle Fusion Applications. You can use this dialog box to perform the following activities:
• Backup and roll back customizations
• Delete page customizations

**Backing Up and Rolling Back Customizations**

Metadata labels are created when you save customizations made in Page Composer. Labels identify the state of the objects in an MDS repository at a given point in time and can serve as save points to which you can roll back your customizations, if the customizations create problems. Therefore, before you make customizations, create a backup of a known good state by creating a label in Page Composer.

For a specific page, to revert to a label that you backed up, you roll back the customization to that label by making that label become the latest version. This action is often referred to as promoting the label to the tip. You can perform this task in the Promote Documents dialog box accessible from the Manage Customizations dialog box.

**Note**

When you use the Manage Customizations dialog, you are rolling back only the customizations for the page and its `pageDef` file. You are not rolling back the other customizations made at the label’s save point.

**Deleting Customizations**

You can use the Manage Customizations dialog box to delete customizations. In the page that contains the customizations, select the page fragment or task flow and choose **Manage Customizations** from the **Administration** menu. In the **Name** list, select the correct layer, find the page, task flow, or fragment that contains the customizations, and click **Delete** for the customization document that you want to delete.

**Advanced Customization Life Cycle Tasks: Highlights**

Move customizations from one environment to another by exporting, downloading, and uploading customizations. You can preserve customizations by backing them up or delete customizations if you do not need them. Also, configure log settings and examine log files to diagnose problems associated with the movement of customizations.

**Customization Tasks**

• The logging functionality for customization set migration is different from the standard logging functionality for Oracle Fusion Applications. You can adjust the amount of detail to log customization migration without requiring a server restart. For more information, refer to the Oracle Fusion Applications Administrator's Troubleshooting Guide.
• Customizations that are created in a sandbox are automatically labeled when the sandbox is published. You need to identify such labels to be able to promote them to the tip. For more information, refer to the Oracle Fusion Applications Administrator’s Guide.

See: Managing Oracle Fusion Applications-Specific Labels in the Oracle Metadata Repository

• For non-Cloud implementations, customization metadata is created from JDeveloper using the Oracle Fusion Applications Administrator Customization role and then packaged and deployed to the source Oracle Fusion Applications environment. For more information, refer to the Oracle Fusion Applications Extensibility Guide for Developers.

See: Design Time Customization Workflow

See: Using Oracle JDeveloper for Customizations
Customizing Pages

Overview of Page Customization Tools

Page Composer Views: Explained

Page Composer provides two options for viewing and customizing page content and layout. Design view and Source view. To open a specific option, you select it from the View menu. Unique features exist in each view, however, both views share many common customization features.

Design View

When you work in Design view, you see one region that shows a WYSIWYG rendering of the page and its content. You work with components directly and access their properties from the toolbar in the chrome of the individual component.

Source View

When you work in Source view, you see the two regions. The Selection pane shows a WYSIWYG rendering of the page and its content. The Source pane contains a hierarchical ordering of the page components, including some components that otherwise do not appear on the page. Source view enables the selection and configuration of such components. You click a component in the Selection pane to highlight the component in the hierarchical list. The cursor turns to a magnifier and a blue outline appears around the component selection. You can also traverse the hierarchy and select components directly.

You work with components indirectly and access their properties by clicking Edit on the Source view header. You can also right click the object in the hierarchy and click Edit.

Tip

In Source view, controls on individual components are inactive, but you can click an individual component to select it. The Source position option on the View menu is available for specifying whether the page source appears above, below,
left, or right of the Selection pane. The **Source position** option is active only when the page is open in Source view.

## Selecting a Text Customization Tool: Points to Consider

You can modify and replace words or phrases that appear on pages in messages, and other areas of the user interface using several tools or tasks. Text customization tools include Application Composer, the Customize User Interface Text page, the Manage Menu Customizations task, and Page Composer. Multiple factors such as the offering you use, the extent and scope of your customizations, and the components that you customize influence the option you select.

This table presents the navigation and offering availability options associated with the tools you can use to customize user interface text.

<table>
<thead>
<tr>
<th>Tool or Task</th>
<th>Navigation</th>
<th>Offering Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Composer</td>
<td>In the Navigator, select <strong>Tools</strong> - <strong>Customization</strong> - <strong>Application Composer</strong></td>
<td>Oracle Sales Cloud</td>
</tr>
<tr>
<td>Customize User Interface Text page</td>
<td>In the Navigator, select <strong>Tools</strong> - <strong>Customization</strong> - <strong>User Interface Text</strong></td>
<td>All Oracle Fusion Applications</td>
</tr>
<tr>
<td>Manage Menu Customization task</td>
<td>In the Navigator, select <strong>Tools</strong> - <strong>Setup and Maintenance</strong>. Then, access the task from the Overview page.</td>
<td>All Oracle Fusion Applications</td>
</tr>
<tr>
<td>Page Composer</td>
<td>Select <strong>Settings and Actions</strong> - <strong>Customize Pagename Pages</strong></td>
<td>All Oracle Fusion Applications</td>
</tr>
<tr>
<td>Projects Custom Objects</td>
<td>In the Navigator, select <strong>Tools</strong> - <strong>Customization</strong> - <strong>Projects Custom Objects</strong></td>
<td>Oracle Fusion Projects</td>
</tr>
</tbody>
</table>

### Text Customization Scenarios

The following table includes possible scenarios for customizing user interface text. Compare your situation to the scenario in the table to determine the most appropriate tool for customizing text in your application.

<table>
<thead>
<tr>
<th>Task</th>
<th>Scope</th>
<th>Tool or Task</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simultaneously replace multiple occurrences of a word or phrase that appears on multiple pages in multiple contexts</td>
<td>Comprehensive The customization affects multiple pages throughout your application.</td>
<td>Customize User Interface Text tool</td>
<td>Change the word Employee to Associate on every page associated with Employee Self Service, Benefits, and Payroll.</td>
</tr>
<tr>
<td>Simultaneously replace multiple occurrences of a word or phrase that appears in parts of messages in the message dictionary</td>
<td>Comprehensive The customization affects multiple messages throughout your application.</td>
<td>Customize User Interface Text tool</td>
<td>Change the word Employee to Associate in every message associated with Employee Self Service, Benefits, and Payroll.</td>
</tr>
</tbody>
</table>
Simultaneously replace multiple occurrences of the singular and plural forms of a word or phrase that appears in messages and on pages | Comprehensive  
The customization affects multiple pages and multiple messages throughout your application. | Customize User Interface Text tool | The phrase Bonus Amount appears on several pages and in multiple messages. You want to replace the phrase with Incentive Compensation on all pages and in all messages except one specific message. |
---|---|---|---|
Replace a word or phrase that appears on a specific page | Targeted: A page  
The customization affects user interface text on a specific page or page fragment. | Page Composer | Change the word customer to account on two specific pages. |
Replace a word or phrase that appears in a specific message in the message dictionary | Targeted: A message  
The customization affects part of a specific message in the message dictionary. | Manage Messages task | Change the word recruit to potential employee, but only in two specific messages. All other messages continue to use the word recruit. |
Simultaneously replace a word or phrase associated with a specific object wherever the object appears | Targeted: A business object  
The customization affects a specific component of a specific message in the message dictionary. | Application Composer  
Customize User Interface Text tool | In the Oracle Sales application, change the label of the opportunity business object, from opportunity to deal. You want the change to affect the business object wherever it appears. |
Replace words or phrases that appear in menus and menu items | Targeted: Navigator menu item text | Manage Menu Customization task | Change the menu item label from Total Compensation Statements to Compensation Package Statements |

Regardless of the changes and the tool you use to make them, text changes that you make with Page Composer and Application Composer always supersede the changes you make using the Customize User Interface Text tool. Targeted and specific changes that you make with the composers take precedence over generic, comprehensive changes that the Customize User Interface Text tool makes.

**Note**

The following exception exists.

You use a tool other than the Customize User Interface Text tool to perform a targeted replacement of one specific instance of OLDWORD with NEWWORD. Then, you use the Customize User Interface Text tool to perform a comprehensive replacement of all remaining instances of OLDWORD with SOME NEWWORD. The targeted replacement supersedes the comprehensive replacement. However, if you then perform a comprehensive replacement of NEWWORD with ANOTHERWORD, then all instances of NEWWORD—including the targeted instance—change and have one instance of SOME ANOTHERWORD.
Elements of Page Design

Page Component Properties: Explained

All components have configurable properties that control, or express, their appearance and behavior. Many properties are common to all component types and some properties are unique to one component type. You access the properties of a component by opening the Component Properties dialog box. The dialog box organizes properties of similar function under tabs that name the category of the properties. Properties and tabs can vary from component to component.

Component Property Tabs

This table describes the tabs that you might see in a component properties dialog box:

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
<td>Settings that can control component aspects that are specific, or often unique to the component.</td>
</tr>
<tr>
<td></td>
<td>For example, on a page that contains a map, a component might have a parameter that provides a choice between units of measurement.</td>
</tr>
<tr>
<td>Display Options</td>
<td>Settings that affect the chrome of a component, including:</td>
</tr>
<tr>
<td></td>
<td>• Header, header text, and border</td>
</tr>
<tr>
<td></td>
<td>• Actions menu</td>
</tr>
<tr>
<td></td>
<td>• Edit, Remove, Expand, Collapse and other icons</td>
</tr>
<tr>
<td></td>
<td>• Tooltips</td>
</tr>
<tr>
<td></td>
<td>For example, display options on image layout components specify the image source URL and its optional link target.</td>
</tr>
<tr>
<td>Child Components</td>
<td>The list of all the components contained within and under the control of the parent component including controls for ordering the child components.</td>
</tr>
<tr>
<td>Style</td>
<td>Settings that affect the look and feel of the component chrome or the component contents and that override corresponding values from a parent object, such as a component, page, and application providing an opportunity to fine-tune appearance.</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Content Style</td>
<td>Note</td>
</tr>
<tr>
<td></td>
<td>Some style properties might be disabled at the component level if other page or application elements (such as the skin) do not support modification to the property.</td>
</tr>
<tr>
<td></td>
<td>For example, font, color, and dimension.</td>
</tr>
</tbody>
</table>

| Events | Events and event handlers for all the components on the current page that the current component can consume. For example, an event could be selecting a check box within the current context. The code that runs and drives the result of a event, such as making another component visible, is an event handler. |

**Customizing Objects That Appear on Multiple Pages: Points to Consider**

You can use Page Composer to customize objects that appear on multiple pages. Whether your customizations affect one or more pages depends on the way you include the object on the page and the Oracle Applications product that you use. Consider these points when you customize objects that appear on multiple pages.

**Customization Appearance Based on Shared and Non Shared Task Flows**

The following table briefly describes the task flow scenarios in which object customizations appear on one or more pages.

<table>
<thead>
<tr>
<th>If the object is...</th>
<th>Then the customizations...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not part of a shared task flow</td>
<td>Do not appear on other pages</td>
</tr>
<tr>
<td>Part of a shared task flow</td>
<td>Do appear on all pages that include the shared task flow</td>
</tr>
</tbody>
</table>

**Warning**

When shared task flows include embedded logic that uses data from the page, the logic can override the customizations that you make in Page Composer.

**Customizing the Same Page with Multiple Tools: Points to Consider**

Using both Oracle JDeveloper and Page Composer, design time customizations and run time customizations can coexist on a page.
Note

Design time customizations are not available in Oracle Cloud implementations.

When you use multiple tools to edit the same page, consider these points:

- You cannot use both tools at the same customization layer.
- If you customize the same component with both tools, the customization at the higher level layer takes precedence.

  For example, let’s say you use Page Composer to change a field label from Employee to Associate in the site layer. Meanwhile someone uses JDeveloper, to change the same label from Employee to Worker in the global layer. The global layer is the base customization layer, as it is only for design time customizations and applies to all users. Your users see Associate, not Worker, because the site layer is at a higher level than the global layer.

- Run time customizations are not automatically visible in JDeveloper. To view them, you must:
  a. Export the customizations from the run time environment to a Java archive (JAR) file.
  b. Import the customizations into the JDeveloper customization application workspace.

Note

For more information on importing run time customizations into Oracle JDeveloper, see the Oracle Fusion Applications Extensibility Guide for Developers.

Customizing Simplified Pages Using Page Composer: Points to Consider

On a simplified page, you can click your name and select Customize User Interface to customize the UI using Page Composer. When customizing a simplified page, consider the customization layer to choose, the types of customizations you can make, and labels for your saved changes.

Customization Layers

The customization layer that you select before making changes to the page determines the scope of users impacted by your customizations. If you are not presented with customization layers to choose from after you select Customize User Interface, then your changes are made to the site layer. For more information to understand customization layers, see the Oracle Fusion Applications Extensibility Guide for Business Analysts.

Types of Customizations

In simplified pages, customization using Page Composer is limited to what you can change with component properties. For example, you can show or hide
fields or make a check box required, but you cannot add new components or change the layout of the page.

After you select a customization layer, if available, you can click:

- **Design** to navigate around and get to the components you want to customize. You cannot make any customizations in this mode.
- **Select** to select a component on the page and open its properties.

Each component has its own set of properties, which may include some of the properties in this table.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Text used by screen readers, for information in addition to what is provided in the Short Desc property.</td>
</tr>
<tr>
<td>Label</td>
<td>Display text for the component, for example the field prompt or the single prompt for a group of check boxes</td>
</tr>
<tr>
<td>Read only</td>
<td>Whether users can edit the component, for example if a check box can be selected or not</td>
</tr>
<tr>
<td>Rendered</td>
<td>Whether the component is visible or hidden to users on the page</td>
</tr>
<tr>
<td>Required</td>
<td>Whether users must enter something for the component before saving the page</td>
</tr>
<tr>
<td>Short Desc</td>
<td>Text that appears when users hover or focus on the component, for example hover over a field label or click in the text box</td>
</tr>
<tr>
<td>Show Required</td>
<td>Whether an asterisk is displayed to indicate that the component is required</td>
</tr>
</tbody>
</table>

When you access component properties on a workstation page using Page Composer, more properties are available.

**Save and Label**

When you save, you can label your changes so that you can later revert to your saved customizations. Labels are stored with a prefix of `composer_`. For example, if you enter `myLabel`, then the label is `composer_myLabel`.

As needed later, you can click your name in the global area and select Manage Customizations. Click Promote for the desired component and select the label to revert to.

**Page Customization Tasks**

**Before You Customize Existing Pages: Points to Consider**

Before you start your customizations, consider these recommended prerequisite tasks:
1. Familiarize yourself with the Oracle Fusion application architecture that enables customization.
2. Become knowledgeable of the typical workflows for working with runtime customizations.
3. Verify that the page is customizable in your Oracle Fusion application.
4. Confirm that your privileges are sufficient for customizing the page.
5. Set up a sandbox for working on your customizations.
6. Determine the appropriate layer for the customization.

**Changing the Look and Feel of Simplified Pages: Points to Consider**

Use the Appearance page to apply predefined themes to all simplified pages, or customize these themes to create your own themes. To open the Appearance page, click Settings in the springboard and then select the Appearance tab.

Use themes to determine:
- The shape of buttons, menus, tabs, and pages
- The style of icon buttons in the springboard
- The branding above every page
- The background of every page

**Branding**

You can define the branding logo and application name above all pages.

**Tip**

For the logo, use an image that is as close to 400 by 100 pixels as possible. In general, an image that is wider than it is tall works best.

**Background**

The base color and watermark appear in the background of all simplified pages.

**Tip**

For the watermark, use an image that is as close to 1024 by 768 pixels as possible.

**Changing the Layout of a Page: Explained**

Page layout defines the number, placement, and orientation of content regions on a page. You set the layout style when you create the page. Some layouts allow choosing a different layout even after you add content to the page; others do not support switching layouts after creation.

To change the layout of a page:
1. Access the page in Design view.
2. Click **Change Layout**.
3. Select the new layout.

**Working With Components in Page Customizations: Procedures**

The Resource Catalog provides a selection of task flows, portlets, and layout components that you can use to customize pages. You access the Resource Catalog in Page Composer from either Design view or Source view. Aspects of components pertinent to page customizations include:

- Accessing the Resource Catalog
- Adding components
- Moving components
- Hiding components manually
- Hiding components programmatically

**Accessing the Resource Catalog**

To access the Resource Catalog when in Design view:

1. Access the page you want to customize.
2. Open the page in Page Composer.
3. From the existing components, select the one that you want to be the parent component. Alternatively, insert a box component and select it as the parent component.
4. Click the **Add Content** button associated with the parent component. The Resource Catalog appears.

**Adding Components**

To add components when in Design view:

1. Access the Resource Catalog.
2. Navigate through the catalog of components until you find the component you want.
3. Click the **Add** icon associated with the component.
4. Cut and paste, or drag and drop the component to place it.

In Source View, you:

1. Select the container component in the selection pane.
2. In the Source view toolbar, click **Add Content**.
3. When the Resource Catalog appears, find the component you want.
4. Click the **Add** icon.

**Moving Components**

Move page components in these ways:

- In Design view, drag and drop the component.
• In Source view, you can:
  • Cut and paste.
  • Drag and drop.
  • Access the Component Properties for the container component and rearrange the components on the Child Components tab.

**Hiding Components Manually**

The *Show Component* property is a display option property that determines whether the component appears to users at run time. The default state of a component is visible. To manually hide a component, you deselect the *Show Component* check box in the Component Properties dialog box. If the component is a child component, then deselecting the *Show Component* property affects only the child component itself. However, deselecting the *Show Component* property of the parent component, hides the parent and all child components it contains. When you hide a parent component, you automatically hide all child components.

You can hide components using any of the following three procedures:

- Hide a child component directly
- Hide a child component from within the parent component
- Hide parent component and all child components

To hide a child component directly:
1. Click the *Edit* icon in the header of the child component. This opens the Component Properties dialog box.
2. Click the *Display Options* tab.
3. Deselect the *Show Component* check box.
4. Click *OK*.

To hide the child component from within the parent component:
1. Click the *Edit* icon on the containing box’s toolbar.
2. Click the Child Components tab.
3. Deselect the box next to the component you want to hide.
4. Click *OK*.

To hide the parent component and all child components:
1. Click the *Edit* icon in the box header.
2. Click the Display Options tab.
3. Deselect the *Show Component* check box.
4. Click *OK*.

**Hiding Components Programmatically**

You can add an Expression Language (EL) expression to a component that enables you to set a condition for hiding the component.

For example, suppose you have two check boxes (1 and 2) on a page. You also have a button (B) that you want to be visible only if check box 2 is checked. To
step through the logic, ask yourself questions such as the ones in the following table.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
<th>Purpose of Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is the condition?</td>
<td>Check box 2 is selected</td>
<td>Determines what the occurrence, or event, is.</td>
</tr>
<tr>
<td>What action or event must happen?</td>
<td></td>
<td>Determines the component that triggers the event.</td>
</tr>
<tr>
<td>What happens when the condition is met?</td>
<td>Button &quot;B&quot; appears.</td>
<td>Determines the effect of the action.</td>
</tr>
<tr>
<td>What happens when the event happens?</td>
<td>Show the component: Button B</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note</td>
<td>The implication is that button B is hidden until the event occurs.</td>
</tr>
<tr>
<td>What property determines whether a component</td>
<td>The Show Component property</td>
<td>Determines the property the code affects.</td>
</tr>
<tr>
<td>is visible?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

So the logic is: If 2 is checked, then the Show Component property of B is activated.

You place the expression on the component that receives the action.

Here is sample code that we might add to the component.

```java
#{if checkbox2.selected = true}
```

After you think through the logic and find the correct expression, you add it to the property.

To access the EL Editor and add an expression to a property:

1. Click the **Edit** icon in the component header.
2. Click the **Display Options** tab.
3. Find the **Show Component** property and click the chevron to open the EL Editor.
4. Add EL to check for an event or condition and set the property, and based on the result, turn the property on or off.

To hide a parent component and all child components programmatically:

1. Click the **Edit** icon in the box’s header.
2. Click the **Display Options** tab.
3. Find the **Show Component** property and click the chevron to open the EL Editor.
4. Add EL to check for an event or condition and set the property and, based on the result, turn the property on or off.

**Deleting Components**

Delete a component only if you are certain that no other components or processes depend on the component you delete. If you have reservations or
are unsure whether any dependencies exist, then hide the component instead of deleting it. If you are sure no other components or processes are dependent on a component, click the **Delete** icon in the component header to delete the component from the page.

**Warning**
If you delete a parent component, you delete all of the child components automatically.

**Note**
Some components, such as mandatory or indexed fields or components that are installed as part of the Oracle Fusion Applications, cannot be deleted using Page Composer.

**Bulk Text Customizations: Explained**

Use the Customize User Interface Text tool when you want to simultaneously customize multiple occurrences of entire words or phrases in the user interface. This tool includes features such as sandbox integration; case sensitive, whole word searches; singular and plural text replacement; and contextual previews all of which make this tool suited for bulk text customization. You access the Customize User Interface Text tool from the Navigator. Under the Tools menu, select Customization, User Interface Text, and then press the **Search and Replace** button.

**Restriction**
The Customize User Interface Text tool searches text on pages and in messages in the message dictionary. This includes user assistance only if the user assistance text is in the message dictionary. Customization functionality for this tool does not extend to text in Business Intelligence (BI) reports, custom Enterprise Scheduler jobs, menus, or service oriented architecture (SOA) processes.

Features of the Customize User Interface Text tool include:

- Search and replace
- Preview and adjust
- Save and publish

**Restriction**
You must work in an active sandbox to access and use Customize User Interface Text tool.

**Search and Replace**

After you activate a sandbox and click the **Search and Replace** button, you enter the search text and the replacement text. You can enter the singular and plural forms of whole words or phrases. As an option, you can perform case sensitive searches. You cannot perform partial word searches, nor can you use wildcard characters as part of the search text.

Use the sample values in this table as a guide when you enter search text.
<table>
<thead>
<tr>
<th>Search Text</th>
<th>Expected Match</th>
<th>Match?</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empl</td>
<td>empl Empl</td>
<td>Yes</td>
<td>Unless you select the Case Sensitive check box, both matches are considered exact.</td>
</tr>
<tr>
<td>empl</td>
<td>Employee EmplID employment</td>
<td>No</td>
<td>The application treats your search text value as a whole word. The text empl is not the same as the text Employee, EmplID, or employment.</td>
</tr>
</tbody>
</table>

After you enter the search text and replacement text, click the **Preview Customizations** button. The tool looks for exact whole word matches.

### Preview and Adjust

The preview divides the search results and presents the matches on tabs based on the match category: user interface text and multipart message text. Data grids on each tab present the matches in rows. You can adjust each row independently. The grids on each tab are similar, but not identical.

On the **User Interface Text** tab, each row includes:

- A view of the existing text and the immediately surrounding text for context. You cannot edit the existing view.
- A preview of the replacement text and the immediately surrounding text for context. You can edit the preview.
- An option to exclude the row and the specific match you see in the row from the customization.

On the **Multipart Message Text** tab, each row includes the previously mentioned elements and:

- An indicator that identifies when the search text appears in a subcomponent of the message, not necessarily in the message body you see in the row.
- The ability to expand the row and view subordinate rows that display the message subcomponent that contains the match and the preview, in context as previously described.

For each row in the preview, you can:

- Do nothing to accept the customization as you see it.
- Select **Exclude** to eliminate the row from the batch customization and maintain the existing text.
- Modify the replacement text to fine tune the customization for the specific match in the row. The row remains part of the batch customization, even if the actual customization differs from the other rows.

### Save and Publish

After you review and adjust the matches, save, thoroughly test, and validate the customizations.

---

**Important**
You should not publish a sandbox before you visually inspect and validate all pages and messages that contain text that you customize.

When the customizations pass your validation process, then publish the sandbox. Message text customizations appear to users when you save, even if you do not publish the sandbox. Page text customizations appear to users when you publish the sandbox.

**Customizing Dialog Box Content: Procedures**

You can use Page Composer to customize the content in your dialog boxes.

Complete these steps in Page Composer to customize the content in dialog boxes.

1. Open the page that gives access to the dialog box.
2. From the **View** menu, select **Source**.

**Note**

You must be in Source view.

3. Select the button that brings up the dialog box.
4. Open the properties for the dialog box.
5. Click the **Child Components** tab in the Component Properties dialog.
6. Edit the dialog box content.
7. Click **Apply** to save your changes, then click **OK** to save your changes and close the Component Properties dialog.

**Customizing the Global Page Template: Explained**

The global page template, or UI Shell template, provides a common header area and the footer panel for all pages in your application. You can use Page Composer to customize the global page template.

Use either of these methods to open the global page template in Page Composer:

- Click Customize Global Page template in your Settings and Actions menu.
- Open the UI Shell Template task flow.

You can make the following customizations to the global page template in Page Composer:

- Add components
- Edit components
  Example: Add Expression Language to hide the Tags link.
- Delete components
  Example: Removing the Tags link.
Tip
When you move your cursor over the global page template, the areas that you can edit display a blue outline.

Adding Components to the Global Page Template
You can use Page Composer to open and add components to the global page template.

To add components to the global page template:
1. If you have not already opened the global page template in Page Composer, open it now as previously described.
2. Select the portion of the global area to which you want to add a component, and click Add Content.
3. In the component catalog, select Components to display the list of available components.
4. Click the Add button associated with the component you want to add. The component appears in the global area.
5. Change component properties as appropriate. For example, if you added the Text component, enter the text that you want to be displayed.
6. After you finish making changes, click Close. When prompted, click Save to save your changes.

Editing Components in the Global Page Template
You can use Page Composer to open and edit components in the global page template.

To edit components in the global page template:
1. If you have not already opened the global page template in Page Composer, open it now as previously described.
2. Select the component that you want to edit.
3. Click Edit.
4. Edit the component properties, then click OK to save your changes.
5. After you finish making changes, click Close. When prompted, click Save to save your changes.

Deleting Components from the Global Page Template
You can use Page Composer to open and delete components from the global page template.

To delete components from the global page template:
1. If you have not already opened the global page template in Page Composer, open it now as previously described.
2. Select the component that you want to delete.
When you move your cursor over the global page template, the areas that you can edit display a blue outline.

3. Click **Delete**. When prompted, click **Delete** to delete the component.

4. If you are done making changes to the global page template, click **Close**. When prompted, click **Save** to save your changes.

**Editing Footers in the Global Page Template**

You can use Page Composer to open and edit footers in the global page template.

To edit footers in the global page template:

1. If you have not already opened the UI Shell template in Page Composer, open it now as previously described.

2. Select the left region of the footer area.

**Tip**

In the Source region, make sure the `spacer: 40px` node is selected after you select the left region of the footer area.

3. In the Source region, right-click the `panelGroupLayout: horizontal` node that appears above the spacer, and click **Add Content**.

4. In the component catalog, select **Components** to display the list of available components.

5. Click the **Add** button associated with the component you want to add. The component appears in the footer.

6. Change component properties as appropriate.

7. After you finish making changes, click **Close**. When prompted, click **Save** to save your changes.

**Note**

For information on making skin customizations, such as selecting a different color palette, see the Oracle Fusion Applications Extensibility Guide for Developers.

**Changing Application Logo in the Global Page Template**

To change the application logo in the global page template:

1. If you have not already opened the UI Shell template in Page Composer, open it now as previously described.

2. Click on the existing logo.

3. Click **Edit**, and edit the component properties.

**Note**

You can change the text in the Short Desc field. This text will display when you hover over the logo.
4. Click the **Style** tab.

5. In the **Background Image** field, enter the URL to access the logo.

6. In the **Other CSS** field, enter the width and height values of the logo’s image in the format: "background-size: <width>px <height>px;". For example, "background-size: 119px 25px;".

**Note**

Adjust the dimensions of the new logo such that it replaces the existing logo appropriately.

7. Click **OK** to save your changes.

**Customizing Pages Using Design Time Tools: Highlights**

Design time customizations can be made to the UI, for example to change the color theme for all pages. Design time customizations and extensions include complex changes that require deployment into the run time environment. Design time customizations and extensions are most often done by developers and are not available in Oracle Cloud implementations.

**Design Time Tools**

- You can use Oracle Application Development Framework (Oracle ADF) Skin Editor to change the look and feel of Oracle Fusion applications. Refer to the Oracle Fusion Applications Extensibility Guide for Developers.
  
  See: Customizing the Oracle Fusion Applications Skin

- You can edit the UI Shell template in JDeveloper to make customizations that you cannot make through Page Composer. Refer to the:
  
  - Oracle Fusion Applications Extensibility Guide for Developers
  
  See: Editing the UI Shell Template

- Oracle Fusion Applications Developer’s Guide
  
  See: Introduction to Implementing the UI Shell

**Customizing Task Lists: Procedures**

Task lists enable you to organize direct links to task flows in your application and web pages outside your application. You link task flows and web pages to tasks through the Tasks List Task Properties in the Component Properties dialog box. Use the Source view of Page Composer to customize task lists.

To customize task lists, you should know how to:

- Insert tasks into the task list.
• Remove tasks from the task list.
• Reorder tasks in the task list.
• Change properties associated with tasks and task lists.

To perform these customizations:

1. Navigate to the task list that you want to change and launch Page Composer.
2. If you are in Design view, change to Source view.

**Note**

You customize task lists in Source view only.

3. In the Selection region, click the task list.
4. Click **Edit** to confirm your intent and automatically select the task list entry in the Source region.
5. In the Source region, click the **Edit Task Flow** link next to the task list region.
6. Click **Edit** to confirm your intent and automatically select the task list entry in the Source region.
7. In the Source region, right-click the panelFormLayout node, and select **Edit**.

   The Component Properties:panelFormLayout dialog box appears.
8. Click the Tasks List Task Properties tab.
9. Expand the tree to view the hierarchy of items in the task list.
10. Select an item in the task list hierarchy.
11. The toolbar provides multiple customization options. You can edit the selected item or insert a new task in the task list.
12. If you insert a new task in the task list, you must enter values in several properties.

   The following table describes the properties that you use to create a task in your task list.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description or Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Application</td>
<td>Select the target web application from the list of web applications that you defined in the deployments tables.</td>
</tr>
</tbody>
</table>

**Note**

Oracle Cloud customers must contact My Oracle Support at https://support.oracle.com to log a service request and obtain a list a valid values.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Focus View Id</td>
<td>Enter the focusViewId value of the target page, for example, <code>/ServiceRequest</code>. Focus View Id and Web Application are mutually inclusive properties.</td>
</tr>
<tr>
<td>Action</td>
<td>Enter the action that occurs when the user selects this item in the task menu.</td>
</tr>
<tr>
<td>Note</td>
<td>• Pages with actions appear in the adfc-config.xml file. These actions can take the user to a particular page.</td>
</tr>
<tr>
<td></td>
<td>• If you specify an action here, then the Web Application and Focus View Id values are ignored.</td>
</tr>
<tr>
<td></td>
<td>• This Action attribute is used in an ADF Controller navigation.</td>
</tr>
<tr>
<td>Label</td>
<td>Enter a name for the new task.</td>
</tr>
<tr>
<td></td>
<td>This name appears in the task list menu and on the task tab when opened if the value of the page attribute <code>isDynamicTabNavigation</code> is true.</td>
</tr>
<tr>
<td>Rendered</td>
<td>Select this to display the item in the task list. Deselect to hide the item.</td>
</tr>
<tr>
<td></td>
<td>To help you identify hidden items quickly, deselected items appear in italics on the customization dialog box.</td>
</tr>
<tr>
<td>Destination</td>
<td>Enter the full URL for the item, such as <a href="http://www.example.com">http://www.example.com</a>. Use to open a new window that takes you out of the Oracle Fusion Middleware UI Shell pages.</td>
</tr>
<tr>
<td></td>
<td>Destination takes precedence over any specified Web Application value.</td>
</tr>
<tr>
<td>Task Type</td>
<td>Select the task type for newly created items from the list of these values:</td>
</tr>
<tr>
<td></td>
<td>• dynamicMain</td>
</tr>
<tr>
<td></td>
<td>• defaultMain</td>
</tr>
<tr>
<td></td>
<td>• defaultRegional</td>
</tr>
<tr>
<td></td>
<td>• taskCategory</td>
</tr>
</tbody>
</table>
13. Save any changes that you made to the properties, and then save the page in Page Composer.

**Making Customizations Visible Based on User Roles: Explained**

Controlling page components conditionally based on user role requires that you create security privileges and add an Expression Language expression to the property that you want to control.

For example, you want to show the Reports link to only your hiring managers, Sherry Callaway and Terrance Whitaker.

First, you create a privilege. Next, you can either assign it to an existing role that the hiring managers already have or you can create a new role that you assign to Sherry and Terrance only.

Next, open the page in Page Composer and select the Reports link component. Edit the component and find the Show Component property.

After you find the property, you must open Expression Builder and enter an expression that verifies whether the user has the appropriate privilege. Use this sample code as a guide:

```#{securityContext.userGrantedPermission['MANAGER_REPORTS_LINK_PRIV']}#``

Save the property changes and close Page Composer.

Now, when users open this page, the application evaluates the expression. Because Sherry and Terrance have the correct permission, the expression and the Show Component property evaluate to Yes; they see the Reports link. For all other users, the expression and the Show Component property evaluate to No.

**Note**

Other methods exist for creating role-based customizations:

- Oracle Sales Cloud features a role layer that enables role-based component customization in Page Composer. The role layer is unique to Oracle Sales Cloud.

- Use JDeveloper to define custom permissions that render UI components based on the user’s access rights. (Not available in Oracle Cloud implementations.)

**Customizing Page Titles: Procedures**

Page titles appear in multiple places in the application: the browser header, page headings, tab titles, and as items in task lists and the navigator. Three separate title and label properties control the different places where the title appears. To customize a page title, you must enter the same value for all three. You change two properties using Page Composer and one using the Manage Menu Customizations task.

**Note**
Page Composer enables page title customization in all Oracle Fusion Applications except Oracle Fusion Sales Cloud.

The properties are:
- Task list Page Title property
- Task list task Label property
- Navigator item node Label property

**Note**

If your application requires language localization, you must provide the translated custom text. For more information on translating custom text, see the Oracle Fusion Applications Extensibility Guide for Developers.

## Task List Page Title Property

The task list Page Title property controls the text that appears in the browser header.

To change the task list Page Title property:

1. Access the page you want to customize and open it in Page Composer.
2. From the **View** menu, select **Source**.
3. In the Selection pane, click a task list link. An edit warning confirmation dialog box appears.
4. Check the **Do not ask next time** check box.
5. Click **Edit**.
6. In the Source region, right-click the **panelFormLayout** node, and select **Edit**. The panelFormLayout Component Properties dialog box appears.
7. Select the **Tasks List Properties** tab.
8. Enter the new value in the **Page Title** field.
9. Click **Apply** to save the changes, if you are changing the next property now.

**Note**

If you keep the dialog box open, skip the next step and then start at step 7 in the next section.

10. Click **OK** to save the changes and close the Component Properties dialog box.

## Task List Task Label Property

The task list task Label property controls the text that appears in the page headings, tab titles, and as menu items in task lists.

To change the task list task Label property:
1. Access the page and open Page Composer in Source view.
2. In the Design pane, select the task list.
   
   A confirmation dialog box appears.
3. Click Edit to edit the task flow.
4. In the Source pane, click the Edit Task Flow link next to the first subordinate region node.
   
   A confirmation dialog box appears.
5. Click Edit to edit the task flow.
6. In the Source region, right-click the panelFormLayout node, and select Edit.
   
   The panelFormLayout Component Properties dialog box appears.
7. Click the Tasks List Task Properties tab.
8. Expand the tree to display the child items in the task list hierarchy.
9. Right-click the child item that you want to customize and click Edit in the toolbar.
10. Enter the new value in the Label field.
11. Click OK to save the changes and close the Component Properties dialog box.

**Navigator Item Node Label Property**

The navigator item node Label property controls the text that appears as menu item links in the Navigator.

To change the navigator item node Label property:

1. Use the Setup and Maintenance Overview page to find and go to the Manage Menu Customizations task.

**Note**

The navigator menu groups appear as a list of expandable nodes.

2. Expand and traverse the nodes until you find the item that you want to change.
3. Select the item, and click Edit.
4. Enter the new label, and click Save and Close.
5. Close the task.

**Making Saved Searches Available to All Users: Procedures**

You can use Page Composer to create saved searches that appear to all users at the site layer.
Note
Create saved searches for all users at the site layer only. Do not use other layers, such as the role layer.

Creating Saved Searches for All Users
To create site-level saved searches:

1. Navigate to the search component.
2. Under your Settings and Actions, click Customize <Page Name> Pages to open Page Composer.
3. Select the Site layer.
4. In Design view, enter the search criteria.
5. Click Search.

Note
You must perform the search before you save.

6. Save the search component.
7. Name your search and optionally select from these options:
   - Set as Default - the search criteria appear when users view the search component.
   - Run Automatically - the search criteria and results appear when users view the search component.
8. Close the dialog box.
9. Save and close Page Composer.
10. Sign out and sign in again to refresh the current list of saved searches.

Editing Saved Searches for All Users
To delete, rename, or change the search options for a site-level saved search, perform the following steps:

1. Navigate to the search component.
2. Under your Settings and Actions, click Customize <Page Name> Pages to open Page Composer.
3. Select the Site layer.
4. In the Design view, select Personalize from the Saved Search list of values.
5. Select a saved search, then you can:
   - Delete
   - Rename
   - Change search options
Note

You must perform the search before you save.

Note

You can’t change the search criteria of a saved search in Page Composer. You can delete and then re-create it if you need to change the search criteria.

6. Save the changes and close Page Composer.
7. Sign out and sign in again to refresh the current list of saved searches.

Adding Translations of Customized Text at Run Time: Procedure

If you customize text in your application and you have users who sign on in multiple locales, you can enter translations of customized text so that it appears to end users in their own language. Several methods exist for adding the translated version of the customized text. The method that you select depends on the number of languages that you install for your application. If you run your application with only a small number of languages installed, you can manually add translations of the customized text for each language at run time.

Important

The following run time method works for text customizations that you make using these tools only:

• Application Composer
• Customize User Interface Text page
• Page Composer
• Projects Custom Objects

Entering the Translations

To enter custom text translations at run time, follow this process.

1. Sign in to your application in the base language, English.
2. Access the sandbox that you are using for the batch of text customizations.
3. Using the Customize User Interface Text page, enter the English word or phrase that you want to customize and enter the English language replacement text.
4. Preview and adjust the matches as necessary.
5. Verify and save your text changes.
6. Publish the sandbox that contains the changed text.
Your replacement text is now available to all users who sign in using English.

Note
Custom text in messages appears to users when you save; custom text in the user interface does not appear until you publish the sandbox.

7. Sign out.
8. Sign in again, but this time select a language other than the base language, for example Portuguese.
9. Access the sandbox that you are using for the text customizations.
10. Using the Customize User Interface Text page, search for the word or phrase in either English or Portuguese and enter the Portuguese language replacement text.
11. Preview and adjust the matches as necessary.
12. Verify and save your text changes.
13. Publish the sandbox that contains the changed text.
   Your replacement text is now available to all users who sign in using Portuguese.

Note
Custom text in messages appears to users when you save; custom text in the user interface does not appear until you publish the sandbox.

Repeat the process in each installed language.

Note
If you run your applications in many locales with many languages, it might be more efficient to edit the XML localization interchange file format (XLIFF) documents instead of adding translations one language at a time. For more information about using XLIFF files to translate custom text, see the Oracle Fusion Applications Extensibility Guide for Developers.

FAQs for Customizing Pages

How can I work on customizations, but prevent users from seeing them until the customizations are complete?

You should create or select an appropriate sandbox. Set it as active to capture your customizations using Page Composer. When you are ready, publish the sandbox to make your changes available to users.
What happens if my customizations or personalizations make the page inaccessible?

You must contact an administrator and she can use the Manage Customizations task to view and, if necessary, delete your changes.

How can I reset a page or task flow to a previously saved version?

Press the Reset Page button in Page Composer (Design or Source view) to reset pages to a previously saved version or to the original out-of-the-box state.
Press the Reset Task Flow button in Page Composer (Source view only) while you have the task flow open to reset it to a previously saved version or to the original out-of-the-box state.

How do I back out or roll back customizations I made to a specific page?

You should use Save and Label to save your work. The label stores the metadata for the page at a specific point in time. Then, open the page in Page Composer and access the Manage Customizations dialog box. Select a label that is older than the current label and promote the older label to the tip layer. This effectively rolls back the customizations.

Note
When you use the Manage Customizations dialog, you roll back the customizations for the page and its pagedef file only. You do not roll back the other customizations made at the label save point.

Can I undo text customizations that I made using the Customize User Interface Text tool if I haven't published the sandbox?

It depends on the type of text that you customize. You must publish the sandbox before your user interface customizations become visible to users. If you haven’t published the sandbox, then destroying it removes the customizations. Message text customizations, however, are visible as soon as you save the preview. After you save, destroying the sandbox has no effect on message text customizations; the changes cannot be undone.

Can I get a report of all customized text if I want to analyze, troubleshoot, and diagnose the cause of unexpected action?

No, but you can use the Customization Setup Manager to export all of your application customizations to a ZIP file that contains the FAOverrideBundle. The
file lists of all text customizations in your application. You can use the contents of this file for diagnosis and troubleshooting purposes. Do not edit or modify the contents of this file. The file is only a read-only version of all text customizations made using browser based tools, such as Application Composer, Page Composer, and the Customize User Interface Text tool.

Why didn’t text in my BI reports, SOA processes, and menus change when I used the Customize User Interface Text tool to perform comprehensive text customizations?

The bulk customizations performed by the Customize User Interface Text tool affect only the text that appears on application pages and in message dictionary messages.

How do I use metadata to perform customization-related tasks that cannot be done in Page Composer?

In Page Composer, you can access the Manage Customizations dialog box, where you can download the metadata. You then manipulate the metadata to perform direct customization and upload the changed metadata using the same dialog box.

How can I post announcements on the simplified home page?

On any simplified page, click Settings in the springboard, and then select the Announcements tab. Use the Announcements page to create, edit, and delete announcements.

Only the content (not the subject) of announcements appears on the home page.
Using Flexfields for Custom Attributes

Flexfields: Overview

A flexfield is an extensible set of placeholder fields in application pages that are associated with a business object. Each segment of the flexfield corresponds to a single application field, such as a segment of a key identifying a particular purchase, or the components of a student’s contact information, or the features of a product in inventory.

Using descriptive and extensible flexfields, you can extend business objects to capture data that wouldn’t otherwise be tracked by the application. If you need to add custom fields to a business object to meet your enterprise-specific requirements, configure the flexfield to have one segment for each needed field.

Using key flexfields, you can configure intelligent key codes comprised of meaningful parts according to your business practices. You configure the key flexfield to have one segment for each part that makes up your key code.

Flexfields let you meet enterprise requirements without changing the data model. Different data can be captured on the same database table. Each segment captures a single atomic value, has a name, and maps to a pre-reserved column in the application database.

You can use a flexfield to extend a business object if it has been registered for use on that object. Application developers create a flexfield and register it so that it is available for configuration. Administrators and implementation consultants set up or configure segments and other properties of the available flexfields. End users see flexfield segments as fields or attributes of information displayed in the application user interface. They enter a value for the attribute. The value may be selected from a list of valid values or entered as free-form text that complies with formatting rules.

The following aspects provide an overview of flexfields:

- Accessing flexfields and flexfield management tasks
- Types of flexfields
- Flexfield segments
- Value sets
- Structure and context
• Deployment
• Run time appearance

Accessing Flexfields and Flexfield Management Tasks

You can view flexfields on a page where they occur using the Highlight Flexfields feature. You can access flexfield management tasks directly from a highlighted flexfield, through product-specific flexfield management tasks, or by starting in the Setup and Maintenance Overview page which is available from the Navigator or the Administration menu.

For lists of flexfields, see assets with the Flexfield: Descriptive, Flexfield: Extensible, or Flexfield: Key type in Oracle Enterprise Repository for Oracle Fusion Applications (http://fusionappsoer.oracle.com).

Types of Flexfields

The following three types of flexfields are available in Oracle Fusion Applications and provide a means to customize applications features without programming.

• Key
• Descriptive
• Extensible

For example, in Oracle Fusion Financials, key flexfields represent objects such as accounting codes and asset categories. Generally, correct operations of a product depend on key flexfield setup. In Oracle Fusion Payables, a descriptive flexfield lets you collect custom invoice details fields on an invoices page. You can implement these fields, which are descriptive flexfield segments, as context-sensitive so they appear only when needed on a row-by-row basis when specific contextual information is met. Extensible flexfields are similar to descriptive flexfields, but provide additional advanced features. Generally, setup of descriptive and extensible flexfields is optional because their segments capture custom fields needed beyond the predefined fields.

Segments

Each field that you configure using flexfields is a flexfield segment. Segments represent attributes of information. They can appear globally wherever the flexfield is implemented, or based on a structure or context.

You define the appearance and meaning of individual segments when configuring a flexfield.

A key flexfield segment commonly describes a characteristic of the entity identified by the flexfield, such as a part number structured to include information about the type, color, and size of an item. A descriptive flexfield segment represents an attribute of information that describes a characteristic of the entity identified on the application page, such as details about a device containing components, some of which are globally present on the page while others are contextually dependent on the category of the device.

Value Sets

A value set is a named group of values that can be used to validate the content of a flexfield segment.
You configure a flexfield segment with a value set that establishes the valid values that an end user can enter for the segment. You define the values in a value set, including such characteristics as the length and format of the values. You can specify formatting rules, or specify values from an application table or predefined list. Multiple segments within a flexfield, or multiple flexfields, can share a single value set.

**Structure and Context**

Key flexfields have structure. Descriptive flexfields and extensible flexfields have context.

Each key flexfield structure is a specific configuration of segments. Adding or removing segments, or rearranging their order, produces a different structure. The database columns on which segments in different structures are based can be reused in as many structures as desired.

Descriptive flexfield segments can be context-sensitive, which means available to an application based on a context value rather than globally available wherever the flexfield appears. A descriptive flexfield context is a set of context-sensitive segments that store information related to the same context value. You define contexts as part of configuring a descriptive flexfield. End users see global segments, as well as any context-sensitive segments that apply to the selected context value.

Extensible flexfield segments are made available to an application based upon a category value. An extensible flexfield context serves as a container for related segments, used to organize the various segments that are applicable to a category value. You define contexts with context-sensitive segments and associate them to categories as part of configuring an extensible flexfield. End users see the segments displayed in subregions, one for each context associated to the selected category value.

In descriptive flexfields and extensible flexfields, the database columns on which context-sensitive segments are based can be reused in as many contexts as desired.

**Deployment**

A flexfield must be deployed to display its current definition in a run time application user interface. For example, if the deployment status is Edited, the flexfield segments may appear in the UI based on the flexfield definition at the time of last deployment, rather than the current definition.

**Run time Appearance**

In an application user interface, descriptive flexfield segments appear as label and field pairs or as a table of fields where the column headers correspond to the labels. The fields represent the flexfield segments and accept entered input or a selection from a list of choices that correspond to the segment's assigned value set. Extensible flexfield segments appear grouped within labeled regions, where each grouping is a context and the region labels are the context names.

Use the **Highlight Flexfields** command in the Administration menu of the Setup and Maintenance work area to identify the location of the flexfields on the run time page. Flexfields in highlight mode display an **Information** icon button to access details about the flexfield, an **Edit** icon button to manage the flexfield, and an **Add Segment** icon button to add flexfield segments.
All segments of a single flexfield are grouped together by default. The layout and positions of the flexfield segments depend on where the application developer places the flexfield on the page. Flexfields may also be presented in a separate section of the page, in a table, or on their own page or subwindow.

You can use Oracle Composer to edit the layout, position, or other display features of the flexfield segments.

**Configuring Flexfields: Overview**

Configuring a flexfield ranges from identifying the need for extending a business object with custom attributes to integrating the custom attributes into the deployment. In the case of key flexfields, configuring the flexfield involves identifying value set assignments and determining segment structures.

**Overall Process for Configuring Custom Attributes**

For descriptive and extensible flexfields, the overall configuration process involves the following:

1. Use the Highlight Flexfields feature from the Administration menu to find flexfields on pages associated with business objects.
2. Plan the flexfield configuration.
3. Plan flexfield validation.
4. Define the attributes by configuring the flexfield segments.
   a. Use the Manage Extensible Flexfields or Manage Descriptive Flexfields tasks, or use the **Configure** icon button directly on the page where the flexfield is highlighted. For simple configurations, use the **Add Segment**, **Add Context Value**, and **Edit Segment** icon buttons directly on the page where the flexfield is highlighted.
   b. Optionally, validate the flexfield configuration.
   c. Optionally, deploy the flexfield to a sandbox for initial testing.
5. Deploy the flexfield to the mainline to display the custom attributes on the application pages and to make them available for integration with other tools such as Oracle Business Intelligence.
6. Perform the necessary steps to integrate the custom attributes into the technology stack.

A simple configuration is limited to such actions as adding a format-only field or adding a field with a basic list of values.

**Overall Process for Configuring Custom Keys**

Using key flexfields, you can configure intelligent key codes comprised of meaningful parts according to your business practices. You configure the key flexfield to have one segment for each part that makes up your key code.

For key flexfields, the overall configuration process involves the following:

1. Use the Highlight Flexfields feature from the Administration menu to find flexfields on pages associated with business objects.
2. Plan the flexfield configuration.

3. Plan the flexfield validation.

4. Define the value sets before configuring the key flexfield segments by going to the Manage Value Sets task.

5. Define the key flexfield structures and their segments, and define structure instances for each structure.
   a. Use the Manage Key Flexfields task or the **Configure** icon button directly on the page where the flexfield is highlighted.
   b. Optionally, validate the flexfield configuration.
   c. Optionally, deploy the flexfield to a sandbox for initial testing.

6. Deploy the flexfield to the mainline to display it on the application pages and to make it available for integration with other tools such as Oracle Business Intelligence.

7. Perform the necessary steps to integrate the flexfield into the technology stack.

**Flexfields at Run Time: Explained**

Many business objects in Oracle Fusion applications have an associated descriptive or extensible flexfield with which you can create custom attributes for the business object. Some business objects have an associated key flexfield for configuring flexible multiple part keys.

The following aspects are important in understanding flexfields at run time:

- Finding flexfields on a page
- Why no flexfields are on a page

**Finding Flexfields on a Page**

At run time, the custom attributes you define as extensible and descriptive flexfield segments appear in the application page just like any other attribute. Key flexfields typically appear in the application page as a field with a key flexfield icon, where the field’s value is actually a collection of segments. In some pages, one or more key flexfield segments may be displayed in the page like any other attribute. Thus, when viewing the page in standard mode, in many cases you may not be able to discern which fields are flexfield segments, or whether flexfields are available to configure on the page.

Use the Highlight Flexfields feature to render the page in a special mode that lets you view:

- Where, if any, flexfields are available on your page
- Which, if any, of the fields on your page are flexfield segments rather than out-of-the-box fields

To obtain information about the flexfields on a page, open the page and choose **Highlight Flexfields** from the **Administration** menu. Hover over the **Information** icon button next to the highlighted fields to display information.
about the flexfield. Choose Unhighlight Flexfields from the Administration menu when you no longer want to see the highlighted flexfields.

When you click the **Configure Flexfield** icon button for a highlighted flexfield, the applicable Manage Flexfields task is displayed for that flexfield. For simple configurations, you can click the **Add Context Value** icon button to add a context value, or click the **Add Segment** or **Edit Segment** icon buttons to add or edit a global segment or a context-sensitive segment that doesn't require advanced configuration.

**Note**

Not all flexfields are available for creating custom attributes. Consult the product-specific documentation in Oracle Fusion Applications Help to verify whether there are any restrictions on using the flexfield.

---

**Why No Flexfields Are on a Page**

For a flexfield to be available in the page, it must be registered by developers. If a flexfield is available, you may configure segments. The segments appear on the page only after you have successfully deployed the flexfield. For information about registering flexfields, see the Oracle Fusion Applications Developer’s Guide. Some business objects haven’t been designed to support flexfields. For information about how to enable business objects with flexfield capability, see Getting Started with Flexfields in the Oracle Fusion Applications Developer’s Guide.

**Note**

The following Oracle Sales Cloud applications don’t support flexfields:

- Sales
- Marketing
- Customer Center
- Trading Community Architecture
- Order Capture

To add custom attributes to these applications, use Application Composer. For more information, see the "Editing an Object: Explained" section in Oracle Sales Cloud: Extending Sales.

---

**Customizing Flexfields Using Page Composer: Explained**

Using Page Composer, you can create customizations to flexfields that are specific to a page.

In Page Composer, to customize:

- Extensible flexfields, open the page in Source view, and look for a region that is bound to an EffContextsPageContainer task flow. This is the container for the extensible flexfield attributes and contexts. To view the flexfield code and identifying information, open the properties panel for the region. To customize any component within the region, select the desired tag and click Edit.
• Descriptive flexfields, open the page in Source view, and look for `<descriptiveFlexfield>` elements. Open the properties panel for the element to view the flexfield code and identifying information. Within the properties panel, you may customize properties for the global and context-sensitive segments or re-order the segments on the page.

**Accessing Flexfield Management Tasks: Procedures**

You can configure and manage flexfields by highlighting them on an application page and using the available on-screen configuration tools. Alternatively, you can access product-specific flexfield tasks or global flexfield management tasks.

**Accessing Flexfield Management Tasks through the Run time Page**

You can identify flexfields on the run time application page where they are implemented.

1. Navigate to an application page.
2. Choose **Highlight Flexfields** from the **Administration** menu in the global area of Oracle Fusion Applications.
3. View the available flexfields highlighted on the page. If any of the fields on the page are custom fields configured as part of a flexfield, they also appear highlighted.
4. To edit a flexfield, use the:
   - **Configure Flexfield** icon button to access the flexfield management task pages for extensive configuration to the flexfield and its segments.
   - **Add Segment** icon button to access the subwindow for adding segments with limited configuration to descriptive flexfields.
   - **Edit Segment** icon button to access the subwindow for limited configuration changes to descriptive flexfield segments.

**Accessing Flexfield Management Tasks through Setup and Maintenance**

Manage flexfields using tasks you access by starting in the Setup and Maintenance Overview page which is available from the Navigator or the Administration menu.

To access tasks for configuring flexfields and value sets, you must be provisioned with roles that entitle you to access the Define Flexfields task list or tasks for managing product-specific flexfields. Contact your security administrator for details. For information about product-specific flexfield tasks, such as Manage Purchasing Descriptive Flexfields, consult the product-specific documentation in Oracle Fusion Applications Help.

To access the flexfield management tasks and search for existing flexfields, perform the following steps:

1. Choose **Setup and Maintenance** from the **Administration** menu in the global area of Oracle Fusion Applications.
2. Search for Define Flexfields in the All Tasks tab.

**Tip**
- Use the Business Object parameter to search:
  - Application Key Flexfields, Application Descriptive Flexfields, and Application Extensible Flexfields to find all tasks related to flexfields.
  - Application Flexfield Value Set to find all tasks related to value sets.
- To manage any:
  - Flexfield across all Oracle Fusion Applications products, search for the Define Flexfields task list and access the Manage Descriptive Flexfields, Manage Extensible Flexfields, and Manage Key Flexfields tasks.
  - Value set across all Oracle Fusion Applications products, search for the Define Flexfields task list and access the Manage Value Sets task.

**Restriction**
If you are configuring key flexfields, search for and access the Manage Value Sets task to set up value sets before accessing the Manage Key Flexfields task.

3. Expand the task list to view the included tasks.
4. Click the Task icon button to open the manage flexfield pages.
5. Search for all or specific flexfields.
6. In the search results, select the flexfield.
7. Use the Edit action to open pages for viewing and configuring the flexfield. Access to managing value sets is available within the tasks for managing descriptive and extensible flexfields.

**Note**
Access to managing value sets is:
- Available within the tasks for managing descriptive and extensible flexfields.
- Not available within the tasks for managing key flexfields. Therefore, configure value sets prior to configuring your key flexfield.

**Flexfields and Oracle Fusion Application Architecture: How They Work Together**

Administrators configure flexfield segments to capture data that represents the values of attributes. Flexfield segments represent attributes of entities (business
objects). Most business objects are enabled for descriptive flexfields. Some business objects are enabled for extensible flexfields.

For example, an airline manufacturer might require very specific attributes for their orders that aren’t provided by the out-of-the-box implementation of an order. Because a flexfield exists for the order business object, you can use it to create and configure the desired attribute.

The figure shows the layers of a flexfield: the business entity table and metadata in the database, business components that are Application Development Framework (ADF) objects or ADF business component (ADFbc) objects derived from the metadata and stored in the Metadata Services Repository (MDS), and the user interface where the input fields defined by the flexfield segments are rendered. The flexfield definition consists of all the metadata defined during configuration and stored in the database.

Application developers create a flexfield and register it so that it is available for configuration. Administrators and implementation consultants configure segments and other properties of the available flexfields. This information is stored as additional flexfield metadata in the database. Deploying the flexfield generates ADF business components based on the flexfield metadata in the database.

The following aspects are important in understanding how flexfields and Oracle Fusion Applications architecture work together:

• Integration

Using Flexfields for Custom Attributes 4-9
• Deployment
• Import and Export
• Run time
• Patching

Integration

The attributes that you add by configuring flexfields are available throughout the Oracle Fusion Middleware technology stack, allowing the flexfields to be used in user interface pages, incorporated into the service-oriented architecture (SOA) infrastructure, and integrated with Oracle Business Intelligence. You identify flexfield segments for integration by the segment’s Application Programming Interface (API) name.

A flexfield affects the Web Services Description Language (WSDL) schemas exposed by ADF services and used by SOA composites. The Web services that expose base entity data also expose flexfield segment data.

Attributes incorporate into SOA infrastructure (BPEL, Rules) and integrate with business intelligence (Oracle Business Intelligence, Extended Spread Sheet Database (ESSbase)).

Flexfield configurations are preserved across Oracle Fusion Applications updates.

Deployment

The metadata for the flexfield is stored in the application database as soon as you save your configuration changes. Deploying the flexfield generates the ADF business components so that the run time user interface reflects the latest definition of the flexfield in the metadata.

Importing and Exporting

You can export and import flexfields with a deployment status of Deployed or Deployed to Sandbox across instances of Oracle Fusion Applications using the Setup and Maintenance Overview page. Ensure a flexfield is eligible for migration (by verifying that it has successfully deployed) prior to attempting the migration.

Run time

For a flexfield to reflect the latest flexfield definition at run time it must be deployed. The user interface accesses a business object and the deployed flexfield definition indicates which business object attributes the flexfield captures values for. If you add display customizations for a flexfield using Oracle Composer, these are customizations on the page so that the same flexfield segments can appear differently on various different pages.

Values entered for segments are validated using value sets.

Patching

Flexfield configurations are preserved during patching and upgrading.
Flexfields and Value Sets: Highlights

Before you use flexfields to create custom attributes, you should be familiar with the Oracle Fusion application architecture that enables customization, customization layers, and the customization lifecycle.

In addition to the extensive information in the Oracle Fusion Applications Help about configuring flexfields that are already available for configuration, consider the resources below for adding flexfields to business components and alternatives to flexfields where flexfields cannot be enabled.

To assess the flexfields available in a deployment of Oracle Fusion Applications, see assets of type: flexfield in the Oracle Enterprise Repository at http://fusionappser.oracle.com.


Restriction
Don’t use Oracle JDeveloper to customize flexfields.

Before Configuring Flexfields

You can add custom attributes to a business object using a flexfield, if a flexfield has been registered for that object by developers.

• For information about registering flexfields to business objects, refer to the Oracle Fusion Applications Developer’s Guide.

See: Getting Started with Flexfields

• The user interface page for a business object that a developer extends to support a flexfield must be enabled to display the custom attributes defined by the flexfield.

See: Adding Descriptive Flexfield UI Components to a Page

See: Employing an Extensible Flexfield on a User Interface Page

• For Sales, Marketing, Customer Center, Trading Community Architecture, and Order Capture applications, use Application Composer to add custom attributes instead of using descriptive and extensible flexfields. For more information, refer to Oracle Sales Cloud: Extending Sales.

See: Application Composer: Introduction

• For information about displaying translated values of a table-validated value set from the value column for the run time session’s locale, refer to the Oracle Fusion Applications Developer’s Guide.

See: Using Multi-Language Support Features
Security

- For an understanding of data security when considering the consequences of applying data security to value sets, refer to the Oracle Fusion Applications Security Guide.

See: Data Security

Deploying Flexfields

- To examine the artifacts of a deployed flexfield configuration that you exported using the exportMetadata WLST command, refer to the Oracle Fusion Applications Extensibility Guide.

See: Exporting Customizations

- For information about synchronizing the updated XML schema definition (XSD) files in the metadata services (MDS) repositories for each service-oriented architecture (SOA) application, refer to the Oracle Fusion Applications Extensibility Guide.

See: Customizing SOA Composite Applications

- For information about incorporating a deployed flexfield into the technology stack, such as customizing the pages, integrating with Oracle Business Intelligence, or integrating into Web Services and service-oriented architecture SOA infrastructure, refer to the Oracle Fusion Applications Concepts Guide.

See: Oracle Fusion Middleware Components

- Oracle ADF services used by SOA composites expose the Web Services Description Language (WSDL) schemas where deployed flexfields are stored.

See: Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite

Oracle Business Intelligence

- For more information about importing and propagating your flexfield changes, refer to the Oracle Fusion Applications Extensibility Guide.

See: Customizing the Oracle BI Repository (RPD)

- For information about importing business intelligence-enabled flexfield changes into the Oracle Business Intelligence repository, refer to the Oracle Transactional Business Intelligence Administrator’s Guide.

Flexfield Management

Managing Flexfields: Points to Consider

Managing flexfields involves registering, planning, and configuring flexfields.
You plan and configure the registered flexfields provided in your applications by applications developers. How you configure flexfield segments determines how the flexfield segments appear to end users. Optionally, you can customize the UI page to change how the flexfield segments appear to end users on that page.

The figure shows the processes involved in making flexfields available to end users. The tasks in the Define Flexfields activity let administrators configure and deploy flexfields. If you deploy a flexfield to a sandbox and decide to apply the configuration to the mainline, select the flexfield in the Manage Flexfields tasks of the Define Flexfields activity and deploy the flexfield in the mainline so that it is available to users.

Consider the following aspects of managing flexfields:

- Registering flexfields
- Planning flexfields
- Configuring flexfields
- Enabling a flexfields segment for business intelligence
- Deploying flexfields
- Optionally changing a flexfield segment’s appearance in a user interface page
- Identifying flexfields on a run time page and troubleshooting
Registering Flexfields

Application development registers flexfields so they are available to administrators and implementation consultants for configuration.

As part of registering a flexfield, application development reserves columns of entity tables for use in flexfields so an enterprise can capture segments to meet their business needs. Many flexfields are registered in Oracle Fusion Applications.

A flexfield must be registered before it can be configured.

For more information on registering flexfields, see Oracle Fusion Applications Developer's Guide.

Planning Flexfields

Before you begin planning flexfields, determine what type is appropriate to your needs, and which business objects are available for customizing flexfields.

All flexfields consist of segments which represent attributes of an entity. The values an end user inputs for an attribute are stored in a column of the entity table.

Carefully plan flexfields before configuring them. Before configuring new segments for your flexfields, be sure to plan their implementation carefully.

If you have determined that a business object supports flexfields, and those flexfields have been registered, you can begin planning how to configure the flexfield for your needs. Note the code name of the flexfield you intend to configure so you can find it easily in the Define Flexfield activity.

In some cases you can customize how the flexfield appears on the page.

See Oracle Fusion Applications Help for specific products to determine any restrictions on using product-specific flexfields.

Configuring Flexfields

Administrators or implementers configure flexfields so they meet the needs of the enterprise. Some flexfields require configuration to make an application operate correctly.

You can configure flexfields using the following methods:

- Go to the manage flexfield tasks in the Setup and Maintenance work area.
- Use the Highlight Flexfields command in the Administration menu while viewing a run time page.
- Use the **Configure Flexfield** icon button to manage a flexfield, such as change a segment’s sequence number, or configure a flexfield segment’s business intelligence label.
• Use the **Add Segment** icon button to add descriptive flexfield segments and context values, or extensible flexfield segments.

Configuring a flexfield includes the following:

• Defining value sets against which the values entered by end users are validated

• Defining the structure or context of the segments in the flexfield

• Specifying the identifying information for each segment

• Specifying the display properties such as prompt, length and data type of each flexfield segment

• Specifying valid values for each segment, and the meaning of each value within the application

**Tip**

You can create value sets while creating descriptive and extensible flexfield segments. However, define value sets before configuring key flexfield segments that use them, because you assign existing value sets while configuring key flexfield segments.

When creating table-validated, independent, dependent, or subset value sets while creating descriptive and extensible flexfield segments, you can optionally specify to display the description of the selected value to the right of the segment at run time.

You can assign sequence order numbers to global segments and to context-sensitive segments in each context. Segment display is always in a fixed order based on the segments’ sequence numbers. You cannot enter a number for one segment that is already in use for a different segment.

**Tip**

Consider numbering the segments in multiples, such as 4, 5, or 10, to make it easy to insert new attributes.

A flexfield column is assigned to a new segment automatically, but you can change the assignment before saving the segment. If you need to set a specific column assignment for a segment, create that segment first to ensure that the intended column isn’t automatically assigned to a different segment.

**Enabling a Flexfield Segment for Business Intelligence**

You can enable flexfield segments for business intelligence if the flexfield is registered in the database as an Oracle Business Intelligence-enabled flexfield.

For more information on enabling segments for business intelligence, see points to consider when enabling key and descriptive flexfield segments for business intelligence.

For extensible flexfield segments, you can’t assign labels and use equalization to prevent duplication.
Deploying Flexfields

Once you have configured a flexfield, you must deploy it to make the latest definition available to runtime users.

In the Define Flexfields tasks, you can deploy a flexfield using either of the following commands:

- The Deploy Flexfield command to deploy a flexfield to mainline. This is for general use in a test or production environment.
- The Deploy to Sandbox command to deploy a flexfield to sandbox. This is to confirm that the flexfield is correctly configured before deploying it to the mainline.

When using the Add Segment and Edit Segment tools for descriptive flexfields in Highlight Flexfields mode, you can use the Save and Deploy command to save your changes and deploy the flexfield to mainline.

Once deployed, the deployment status indicates the state of the currently configured flexfield relative to the last deployed definition.

Optionally Changing a Flexfield Segment Appearance

The flexfield attributes that you define integrate with the user interface pages where users access the attributes’ business object. Application development determines the UI pages where business objects appear and the display patterns used by default to render flexfield segments.

After a flexfield has been deployed to a mainline metadata services (MDS) repository so that it appears on application pages, you can customize it on a per-page basis using Page Composer. For example, you can hide a segment, change its prompt or other properties, or reorder the custom global attributes so that they are interspersed with the core attributes in the same parent layout.

You can only customize the appearance of descriptive and extensible flexfield segments in the UI page using Page Composer once the flexfield is deployed to the mainline.

If the Oracle Fusion applications are running in different locales, you can provide different translations for translatable text, such as prompts and descriptions. Enter translations by signing in using the locale that requires the translated text. You do this by selecting Settings and Actions - Personalization - Set Preferences in the global area and changing the text to the translated text for that locale.

Identifying Flexfields on a Run Time Page and Troubleshooting

The Highlight Flexfields command in the Administration menu of the Setup and Maintenance work area identifies the location of flexfields on the run time page by displaying an Information icon button for accessing details about each flexfield.

Even if a descriptive or extensible flexfield hasn’t yet been deployed and no segments appear on the run time page in normal view, the flexfield appears in
the Highlight Flexfield view for that page. In the case of descriptive flexfields, the segments as of the last deployment appear. **Highlight Flexfields** accesses the current flexfield metadata definition.

Use the highlighted flexfield’s **Edit** icon button to manage flexfields directly. Alternatively, note a highlighted flexfield’s name to search for it in the tasks for managing flexfields.

To examine a flexfield’s configuration, export the deployed artifacts using the `exportMetadata WLST`.

For more information on creating flexfields and adding them to a UI page, see the Oracle Fusion Applications Developer’s Guide.

For more information about customizing flexfield segment appearance with Oracle Composer, see guidance on customizing existing pages in the Oracle Fusion Applications Extensibility Guide.

**Flexfield Segment Properties: Explained**

Independent of the value set assigned to a segment, segments may have properties that affect how they are displayed and how they behave.

The following aspects are important in understanding:

- Display properties
- Properties related to segment values
- Properties related to search
- Range validation segments
- Rule validation of segment values
- Naming conventions

**Display Properties**

The following table summarizes display properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled</td>
<td>Whether the segment can be used.</td>
</tr>
<tr>
<td>Sequence</td>
<td>The order the segment appears in relation to the other configured segments.</td>
</tr>
<tr>
<td>Prompt</td>
<td>The string to be used for the segment’s label in the user interface.</td>
</tr>
<tr>
<td>Display type</td>
<td>The type of field in which to display the segment.</td>
</tr>
<tr>
<td>Checked and unchecked values</td>
<td>If the display type is check box, the actual values to save. For example, Y and N or 0 and 1.</td>
</tr>
<tr>
<td>Display size</td>
<td>The character width of the field.</td>
</tr>
<tr>
<td>Display height</td>
<td>The height of the field as measured in visible number of lines when the display type is a text area.</td>
</tr>
<tr>
<td>Read only</td>
<td>Whether the field should display as read-only, not editable text.</td>
</tr>
</tbody>
</table>
Description help text

The field-level description help text to display for the field. Use description help text to display a field-level description that expands on or clarifies the prompt provided for the field.

If description help text is specified, a Help icon button is displayed next to the field in the run time application. The description help text is displayed when the user hovers over the Help icon button.

Instruction help text

The field-level instruction help text to display for the field.

Use instruction help text to provide directions on using the field. If instruction help text is specified, it is displayed in an in-field help note window that appears when users give focus to or hover over the field.

Properties Related to Search

Extensible flexfield segments can be marked as selectively required in search using the indexed property. The indexed property requires end users to enter a value before conducting a search on the attribute represented by the indexed segment. A database administrator must create an index on the segment column representing the indexed attribute.

Range Validation of Segments

Range validation enables you to enforce an arithmetic inequality between two segments of a flexfield. For example, a product must be ordered before it can be shipped. Therefore, the order date must be on or before the ship date, and consequently the order date segment value must be less than or equal to the ship date segment value. You can use range validation to ensure this relationship.

The conditions for range validation are as follows:

• Segments must be configured for range validation in pairs, one with the low value and one with the high value.

• Both segments must be of the same data type.

• Both segments must be parts of the same structure in a key flexfield or parts of the same context in a descriptive flexfield or extensible flexfield.

• The low value segment must have a lower sequence number than the high value segment.

• Non-range validated segments can exist between a range validated pair, but range validated pairs cannot overlap or be nested.

You can configure as many range validated pairs as you want within the same flexfield. Your application automatically detects and applies range validation to the segment pairs that you define, in sequence order. It must encounter a low value segment first, and the next range validated segment that it encounters must be a high value segment. These two segments are assumed to be a matching pair. The low value and the high value can be equal.
Rule Validation of Segment Values

Validation rules on descriptive and extensible flexfield segments determine how an attribute is validated. The value entered for an attribute on a business object may need to match a specified format or be restricted to a list of values. Use a value set to specify the validation rules.

Value set validation is required for global segments and context-sensitive segments, and optional for context segments. In the case of context segments, the application may validate an input value instead of the value set validating the input value against the context segment. However, the application input values must match exactly the valid context segment values. If the context segment values are a superset or subset of the input values, you must assign a table-validated value set or independent value set to validate context values.

When you configure a descriptive flexfield segment, you can specify a constant to use for setting the initial value. The initial value can be an available parameter. For every planned segment, list the constant value or parameter, if any, to use for the initial value.

Naming Conventions

Enter a unique code, name, and description for the segment. These properties are for internal use and not displayed to end users. You can’t change the code after the segment is created.

The Application Programming Interface (API) name is a name for the segment that isn’t exposed to end users. The API name is used to identify the segment in various integration points including web services, rules, and business intelligence. Use alphanumeric characters only with a leading character. For example, enter a code consisting of the characters A-Z, a-z, 0-9 with a non-numeric leading character. The use of spaces, underscores, multi-byte characters, and leading numeric characters isn’t permitted. You can’t change the API name after the segment has been created.

Flexfields Segments: How They Are Rendered

Flexfield segments appear on pages as attributes of business objects.

Settings That Affect Flexfield Segment Display

When you configure flexfield segments, the value you enter for the segment’s display type determines how the segment appears on the run time page.

How Display Type Values Appear

The figure shows how display types appear at run time.

In the following figure, identify the display type by letter when referring to the table of descriptions for check box, drop-down list, list of values, pop-up list of values, and radio button group.
In the following figure, identify the display type by letter when referring to the table of descriptions for radio button group, text area, text box, and date/time.

E. Radio Button Group

F. Text Area

G. Text Box

H. Date/Time

The table describes each display type. The Example column refers to the figures above.
<table>
<thead>
<tr>
<th>Display Type</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
<td>A</td>
<td>The field is displayed as a check box. If the end user selects the checkbox, the checked value is used. Otherwise, the unchecked value is used.</td>
</tr>
<tr>
<td>Drop-down List</td>
<td>B</td>
<td>The field displays a dropdown list of values from which the end user can select a value.</td>
</tr>
<tr>
<td>List of Values</td>
<td>C</td>
<td>The field displays a dropdown list of values from which the end user can select a value. The user can also click Search to find more values.</td>
</tr>
<tr>
<td>Pop-up List of Values</td>
<td>D</td>
<td>The field displays as a text field with a Search icon button. The end users can type a value in the text field or they can click the Search icon button to open a subwindow for searching.</td>
</tr>
<tr>
<td>Radio Button Group</td>
<td>E</td>
<td>The field is displayed as a set of radio buttons. The end user can select one button. Selecting a button deselects any previously selected button in the set.</td>
</tr>
<tr>
<td>Text Area</td>
<td>F</td>
<td>The field is displayed as a text area in which the end user can type multiple lines of text. The display width and height specify the visible width and number of lines in the text area, respectively.</td>
</tr>
<tr>
<td>Text Box</td>
<td>G</td>
<td>The field is displayed as a text field in which the end user can type a single line of text. The display width controls the width of the text box.</td>
</tr>
<tr>
<td>Date/Time</td>
<td>H</td>
<td>The field enables the end user to enter a date if the data type is Date, or a date and time if the data type is Date Time. The user can select the date from a calendar. If the data type is Date Time, the field also displays fields for specifying the hour, minutes, seconds, AM or PM, and time zone.</td>
</tr>
<tr>
<td>Hidden</td>
<td></td>
<td>The field isn’t displayed.</td>
</tr>
</tbody>
</table>
Flexfields and Value Sets: How They Work Together

Value sets are specific to your enterprise. When gathering information using flexfields, your enterprise’s value sets validate the values that your users enter based on how you defined the value set.

You can assign a value set to any number of flexfield segments in the same or different flexfields. Value set usage information indicates which flexfields use the value set.

The following aspects are important in understanding how flexfields and value sets work together:

- Defining value sets
- Shared value sets
- Deployment

Defining Value Sets

As a key flexfield guideline, define value sets before configuring the flexfield, because you assign value sets to each segment as you configure a flexfield. With descriptive and extensible flexfields, you can define value sets when adding or editing a segment.

Caution

Be sure that changes to a shared value set are compatible with all flexfield segments that use the value set.

Shared Value Sets

When you change a value in a shared value set, the change affects the value set for all flexfields that use that value set. The advantage of a shared value set is that a single change propagates to all usages. The drawback is that the change shared across usages may not be appropriate in every case.

Value Set Values

To configure custom attributes to be captured on the value set values screen in the Manage Value Sets task, configure the Value Set Values descriptive flexfield. The object’s code is FND_VS_VALUES_B. This flexfield expects the context code to correspond to the value set code. For each value set, you can define a context whose code is the value set code, and whose context-sensitive segments will be shown for the values of that value set. By default the context segment is hidden since it defaults to the value set code and is not expected to be changed.

You can also define global segments that will be shown for all value sets. However, this would be quite unusual since it would mean that you want to capture that attribute for all values for all value sets.

Deployment

When you deploy a flexfield, the value sets assigned to the segments of the flexfield provide end users with the valid values for the attributes represented by the segments.
Defaulting and Deriving Segment Values: Explained

To populate a flexfield segment with a default value when a row is created, specify a default type of constant or parameter and a default value.

To synchronize a segment's value with another field's value whenever it changes, specify the derivation value to be the flexfield parameter from which to derive the attribute's value. Whenever the parameter value changes, the attribute's value is changed to match. If you derive an attribute from a parameter, consider making the attribute read-only, as values entered by users are lost whenever the parameter value changes.

When defaulting or deriving a default value from a parameter, only those attributes designated by development as parameters are available to be chosen.

Different combinations of making the segments read only or editable in combination with the default or derivation value or both, have different effects.

Initial run time behavior corresponds to the row for the attribute value being created in the entity table. If the default value is read only, it cannot subsequently be changed through the user interface. If the default value isn't read only, users can modify it. However, if the segment value is a derived value, a user-modified segment value is overwritten when the derivation value changes.

<table>
<thead>
<tr>
<th>Default Type</th>
<th>Default value specified?</th>
<th>Derivation value specified?</th>
<th>Initial run time behavior</th>
<th>Run time behavior after parameter changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No</td>
<td>Yes</td>
<td>No initial segment value</td>
<td>The changed parameter derivation value updates segment value</td>
</tr>
<tr>
<td>Constant</td>
<td>Yes</td>
<td>No</td>
<td>Default segment value</td>
<td>N/A</td>
</tr>
<tr>
<td>Constant</td>
<td>Yes</td>
<td>Yes</td>
<td>Default segment value</td>
<td>The changed parameter derivation value updates segment value</td>
</tr>
<tr>
<td>Parameter</td>
<td>Yes</td>
<td>No</td>
<td>The default segment value is the parameter's default value</td>
<td>N/A</td>
</tr>
<tr>
<td>Parameter</td>
<td>Yes</td>
<td>Yes, and same as default value</td>
<td>The default segment value is the parameter's default and derivation value</td>
<td>The changed parameter derivation value updates segment value</td>
</tr>
<tr>
<td>Parameter</td>
<td>Yes</td>
<td>Yes, and different from default value</td>
<td>The default segment value is the parameter’s default value</td>
<td>The changed parameter default value doesn’t update segment value. Only the changed derivation value updates the segment value.</td>
</tr>
</tbody>
</table>

### Flexfield Usages: Explained

Usage affects various aspects of flexfields. The usage of the flexfield is set when the flexfield is registered and specifies the application and table with which the flexfield is associated.

Entity usage indicates the table containing the segments of a flexfield.

A flexfield can have multiple usages. The first table registered for a flexfield is the master usage. Segments are based on the master usage, and other usages of the same table for the same flexfield use the same segment setup, though the column names optionally may have a differentiating prefix.

### Extensible Flexfields

You can configure different behavior for extensible flexfield contexts at the usage level. The usage of an extensible flexfield context determines in which scenarios or user interfaces the segments of a context appear to end users. For example, if a Supplier page displays an extensible flexfield’s supplier usage and a buyer page displays that same extensible flexfield’s buyer usage, a context that is associated to the supplier usage but not the buyer usage displays only on the supplier page and not the buyer page.

### Value Sets

The usage of value sets specifies the flexfields having segments where the value set is assigned.

### FAQs for Flexfield Management

#### Why did my flexfield changes not appear in the run time UI?

The ADF business components or artifacts of a flexfield, which are generated into an metadata services (MDS) repository when the flexfield is deployed, are cached within a user session. You must sign out and sign back in again to view flexfield definition changes reflected in the run time application user interface page.

A flexfield’s deployment status indicates whether the flexfield segments as currently defined in the metadata are available to end users. The flexfield
segments seen by end users in the run time correspond to the flexfield definition that was last deployed successfully.

**How can I enable flexfield segments for Oracle Social Network Cloud Service?**

Descriptive flexfield segments can be enabled for integration with Oracle Social Network Cloud Service. When you manage Oracle Social Network Objects during setup and maintenance, search for the business object that includes descriptive flexfields, and select the business object attributes that are defined as flexfield segments.

**Flexfield Deployment**

**Flexfield Deployment: Explained**

Deployment generates or refreshes the Application Development Framework (ADF) business component objects that render the flexfield in a user interface. The deployment process adds the custom attributes to the Web Services Description Language (WSDL) schemas that are exposed by Oracle ADF services and that are used by SOA composites. Flexfields are deployed for the first time during the application provisioning process. After you configure or change a flexfield, you must deploy it to make the latest definition available to end users.

If a descriptive flexfield is enabled for business intelligence, the deployment process redeploy the flexfield’s business intelligence artifacts.

You can deploy a flexfield to a sandbox for testing or to the mainline for use in a test or production run time environment. You can deploy extensible flexfields as a background process.

After deployment, the custom attributes are available for incorporating into the SOA infrastructure, such as business process and business rule integration. For example, you can now write business rules that depend on the custom attributes. You must sign out and sign back in to Oracle Fusion Applications to see the changes you deployed in the run time.

The following aspects are important in understanding flexfield deployment:

- Deployment Status
- Initial Deployment Status
- Metadata Validations
- Metadata Synchronization
- Deployment as a Background Process

**Deployment Status**

Every flexfield has a deployment status.
A flexfield can have the following deployment statuses.

<table>
<thead>
<tr>
<th>Deployment Status</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edited</td>
<td>The flexfield metadata definition hasn’t been deployed yet. Updates of the metadata definition aren’t applied in the runtime environment yet.</td>
</tr>
<tr>
<td>Patched</td>
<td>The flexfield metadata definition has been modified through a patch or through a data migration action, but the flexfield hasn’t yet been deployed so the updated definition isn’t reflected in the runtime environment.</td>
</tr>
<tr>
<td>Deployed to Sandbox</td>
<td>The current metadata for the flexfield is deployed in ADF artifacts and available as a flexfield-enabled sandbox. The status of the sandbox is managed by the Manage Sandboxes task available to the Administrator menu of the Setup and Maintenance work area.</td>
</tr>
<tr>
<td>Deployed</td>
<td>The current metadata for the flexfield is deployed in ADF artifacts and available to end users. There haven’t been any changes to the flexfield since it was last deployed in the mainline.</td>
</tr>
<tr>
<td>Error</td>
<td>The deployment attempt in the mainline failed.</td>
</tr>
</tbody>
</table>

**Note**

Whenever a value set definition changes, the deployment status of a flexfield that uses that value set changes to edited. If the change results from a patch, the deployment status of the flexfield changes to patched.

**Initial Deployment Status of Flexfields**

The Oracle Fusion Applications installation loads flexfield metadata into the database. This initial load sets the flexfield status to Edited. The application provisioning process during installation deploys the flexfields of the provisioned applications, which sets their status to Deployed if no errors are encountered.

When accessing a provisioned application, deployed flexfields are ready to use. In some cases, flexfield availability at runtime requires setup, such as defining key flexfields.

**Metadata Validation**

Use the Validate Metadata command to view possible metadata errors before attempting to deploy the flexfield. Metadata validation is the initial phase of all flexfield deployment commands. By successfully validating metadata before running the deployment commands, you can avoid failures in the metadata validation phase of a deployment attempt. The deployment process aborts if it encounters an error during the metadata validation phase. Metadata validation results don’t affect the deployment status of a flexfield.
**Metadata Synchronization**

When an extensible or descriptive flexfield is deployed, the deployment process regenerates the XML schema definition (XSD), which makes the custom attributes available to web services and the SOA infrastructure.

After deploying a flexfield configuration, you must synchronize the updated XML schema definition (XSD) files in the MDS repositories for each SOA application.

---

**Note**

To synchronize the updated XSD files in the MDS repositories in Oracle Cloud implementations, log a service request using My Oracle Support at http://support.com/

---

**Deployment as a Background Process**

You can deploy extensible flexfields or incremental changes made to extensible flexfields as a background process. You must use this action to deploy extensible flexfields that have more than 30 categories. You can also use this action if you want to deploy several extensible flexfields, or if you want to continue working in your session without having to wait for a deployment to complete.

**Flexfield Deployment Status: How It Is Calculated**

Flexfield deployment status indicates how the flexfield metadata definition in the Oracle Fusion Applications database relates to the Application Development Framework (ADF) business components generated into a Metadata Services (MDS) repository.

The following aspects are important in understanding how flexfield deployment status is calculated:

- Settings that affect flexfield deployment status
- How deployment status is calculated

**Settings That Affect Flexfield Deployment Status**

If you have made a change to a flexfield and expect a changed deployment status, be sure you have saved your changes. No settings affect flexfield deployment status.

**How Deployment Status Is Calculated**

If the flexfield definition has been edited through the Define Flexfields activity task flows, the status is Edited. The latest flexfield metadata definition in the Oracle Fusion application diverges from the latest deployed flexfield definition. Any change, including if a value set used in a flexfield changes, changes the
deployment status to Edited. If a flexfield has never been deployed, its status is Edited.

**Note**

When an application is provisioned, the provisioning framework attempts to deploy all flexfields in that application.

If you deploy the flexfield to a sandbox successfully, the status is Deployed to Sandbox. The latest flexfield metadata definition in the Oracle Fusion application matches the metadata definition that generated ADF business components in a sandbox MDS repository. Whether the sandbox is active or not doesn’t affect the deployment status. If the flexfield was deployed to a sandbox and hasn’t been edited or redeployed to the mainline since then, the status remains Deployed to Sandbox independent of whether the sandbox is active, or who is viewing the status.

If you deploy the flexfield successfully to the mainline, the status is Deployed. The latest flexfield metadata definition in the Oracle Fusion application matches the metadata definition that generated ADF business components in a mainline MDS repository. Change notifications are sent when a flexfield is deployed successfully to the mainline.

If either type of deployment fails so that the current flexfield definition isn’t deployed, the status is Error. The deployment error message gives details about the error. The latest flexfield metadata definition in the Oracle Fusion application likely diverges from the latest successfully deployed flexfield definition.

If the flexfield definition has been modified by a patch, the status is Patched. The latest flexfield metadata definition in the Oracle Fusion application diverges from the latest deployed flexfield definition. If the flexfield definition was Deployed before the patch and then a patch was applied, the status changes to Patched. If the flexfield definition was Edited before the patch and then a patch was applied, the status will remain at Edited to reflect that there are still changes (outside of the patch) that aren’t yet in effect.

When a deployment attempt fails, you can access the Deployment Error Message for details.

**Deploying a Flexfield-Enabled Sandbox: How It Works With Mainline Metadata**

The flexfield definition in a sandbox corresponds to the flexfield metadata definition in the Oracle Fusion Applications database at the time the flexfield was deployed to the sandbox. When the flexfield is ready for end users, the flexfield must be deployed to the mainline.

A flexfield-enabled sandbox uses the following components.

- Flexfield metadata in the Oracle Fusion Applications database
- Flexfield business components in a sandbox Metadata Services (MDS) repository
- User interface customizations for the flexfield in the mainline MDS repository
The figure shows the two types of deployment available in the Manage Flexfield tasks of the Define Flexfields activity. Deploying a flexfield to a sandbox creates a sandbox MDS repository for the sole purpose of testing flexfield behavior. The sandbox is only accessible to the administrator who activates and accesses it, not to users generally. Deploying a flexfield to the mainline applies the flexfield definition to the mainline MDS repository where it is available to end users. After deploying the flexfield to the mainline, customize the page where the flexfield segments appear. Customization of the page in the sandbox MDS repository cannot be published to the mainline MDS repository.

**Sandbox Metadata Services Repository Data**

Deploying the flexfield to a sandbox generates the Application Development Framework (ADF) business components of a flexfield in a sandbox MDS repository for testing in isolation.

**Warning**

Don’t customize flexfield segment display properties using Page Composer in a flexfield-enabled sandbox as these changes will be lost when deploying the flexfield to the mainline.

**Mainline Metadata Services Repository Data**

The Oracle Fusion Applications database stores the single source of truth about a flexfield. When the flexfield is deployed, the ADF business component objects...
that implement the flexfield in the run time user interface are generated in the mainline MDS repository from this source.

Deploying a Flexfield to a Sandbox: Points to Consider

Deploying a flexfield to a sandbox creates a flexfield-enabled sandbox. Each flexfield-enabled sandbox contains only one flexfield.

You can test the run time behavior of a flexfield in the flexfield-enabled sandbox. If changes are needed, you return to the Define Flexfield tasks to change the flexfield definition.

When you deploy a flexfield to sandbox, the process reads the metadata about the segments from the database, generates flexfield Application Development Framework (ADF) business component artifacts based on that definition, and stores in the sandbox only the generated artifacts derived from the definition.

When you deploy a flexfield sandbox, the process generates the name of the flexfield sandbox, and that flexfield sandbox is set as your current active sandbox. When you next sign in to the application, you can see the updated flexfield configurations. The Oracle Fusion Applications global area displays your current session sandbox.

**Note**

Unlike a standalone sandbox created using the Manage Sandboxes tool, the sandbox deployed for a flexfield contains only the single flexfield. You can manage flexfield sandboxes, such as setting an existing flexfield sandbox as active or deleting it, using the Manage Sandboxes tool.

When you deploy a flexfield to the mainline after having deployed it to the sandbox, the sandbox-enabled flexfield is automatically deleted.

**Sandbox MDS Repository Data**

The sandbox data lets you test the flexfield in isolation without first deploying it in the mainline where it could be accessed by users.

**Warning**

Don’t customize flexfield segment display properties using Page Composer in a flexfield-enabled sandbox as these changes will be lost when deploying the flexfield to the mainline.

**Managing a Flexfield-Enabled Sandbox**

When you deploy a flexfield as a sandbox, that flexfield-enabled sandbox automatically gets activated in your user session. When you sign back in to see the changes, the sandbox is active in your session.
You can only deploy a flexfield to a sandbox using the Define Flexfields task flow pages.

You also can use the Manage Sandboxes feature in the Administration menu of the Setup and Maintenance work area to activate and access a flexfield-enabled sandbox.

**Note**

Whether you use the Define Flexfields or Manage Sandboxes task flows to access a flexfield-enabled sandbox, you must sign out and sign back in before you can see the changes you deployed in the run time.

You cannot publish the flexfield from the sandbox to the mainline. You must use the Define Flexfields task flow pages to deploy the flexfield for access by users of the mainline because the flexfield configuration in the mainline is the single source of truth.

### Deploying Flexfields Using the Command Line: Explained

You can use the Manage Key Flexfields, Manage Descriptive Flexfields, and Manage Extensible Flexfields tasks to deploy flexfields. You can also use WebLogic Server Tool (WLST) commands for priming the Metadata Services (MDS) repository with predefined flexfield artifacts and for deploying flexfields.

The table describes the available commands.

<table>
<thead>
<tr>
<th>WebLogic Server Tool Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| deployFlexForApp             | Deploys all flexfields for the specified enterprise application. Only flexfields whose status is other than deployed are affected by this command unless the option is enabled to force all flexfields to be deployed regardless of deployment status.  
Initial application provisioning runs this command to prime the MDS repository with flexfield artifacts. |
| deployFlex                  | Deploy a single flexfield regardless of deployment status |
| deployPatchedFlex           | Deploys flexfield changes that have been delivered using a flexfield Seed Data Framework (SDF) patch. Deploys flexfields that have a Patched deployment status. |
| deleteFlexPatchingLabels    | Displays MDS label of flexfield changes for viewing and deleting patching labels. |
| ValidateFlexDeploymentStatus| Displays list containing flexfields that aren't deployed or failed deployment. |

Executing these commands outputs a report at the command line. The report provides the following information for every flexfield that is processed:

- Application identity (APPID)
• Flexfield code
• Deployment result, such as success or error

In case of errors, the report lists the usages for which the errors were encountered. If a run time exception occurs, the output displays the traceback information. For each WLST flexfield command, adding the \texttt{reportFormat='xml'} argument returns the report as an XML string.

Consider the following aspects of command line deployment.

• Preparing to use the WLST flexfield commands
• Using the \texttt{deployFlexForApp} command
• Using the \texttt{deployFlex} command
• Using the \texttt{deployPatchedFlex} command
• Using the \texttt{deleteFlexPatchingLabels} command
• Using the \texttt{validateFlexDeploymentStatus} command
• Exiting the WLST and checking the results

\textbf{Preparing To Use the WLST Flexfield Commands}

You can only execute the WLST flexfield commands on a WebLogic Administration Server for a domain that has a running instance of the Oracle Fusion Middleware Extensions for Applications (Applications Core) Setup application.

For more information on deploying the Applications Core Setup application, see the Oracle Fusion Applications Developer's Guide.

Ensure that the AppMasterDB data source is registered as a JDBC data source with the WebLogic Administration Server and points to the same database as the ApplicationDB data source.

Start the WebLogic Server Tool (WLST) if it isn’t currently running.

UNIX:

\texttt{sh $JDEV\_HOME/oracle\_common/common/bin/wlst.sh}

Windows:

\texttt{wlst.cmd}

Connect to the server, replacing the user name and password arguments with your WebLogic Server user name and password.

\texttt{connect('wls\_username', 'wls\_password', 'wls\_uri')}

The values must be wrapped in single-quotes. The \texttt{wls\_uri} value is typically T3://localhost:7101.

For more information on the WLST scripting tool, see the Oracle Fusion Middleware Oracle WebLogic Scripting Tool.
Using the deployFlexForApp Command

The **deployFlexForApp** command translates the product application's predefined flexfield metadata into artifacts in the MDS repository.

**Important**

This command is run automatically when you provision applications. However, after custom applications development, you must run the **deployFlexForApp** command after you configure your application to read the flexfield artifacts from the MDS repository and before you log into the application for the first time, even if there is no predefined flexfield metadata.

This command doesn't deploy flexfields that have a status of Deployed unless the force parameter is set to 'true' (the default setting is 'false').

For more information on priming the MDS partition with configured flexfield artifacts, see the Oracle Fusion Applications Developer's Guide.

From the WLST tool, execute the following commands to deploy the artifacts to the MDS partition, replacing `product_application_shortname` with the application's short name wrapped in single-quotes.

```python
deployFlexForApp('product_application_shortname', ['enterprise_id'], ['force'])
```

In a multi-tenant environment, replace `enterprise_id` with the Enterprise ID to which the flexfield is mapped. Otherwise, replace with 'None' or don't provide a second argument.

To deploy all flexfields regardless of their deployment status, set force to 'true' (the default setting is 'false'). If you want to deploy all flexfields in a single-tenant environment, you either can set `enterprise_id` to 'None', or you can use the following signature:

```python
deployFlexForApp(applicationShortName='product_application_shortname', force='true')
```

**Tip**

The application's short name is the same as the application's module name.

For more information about working with application taxonomy, see the Oracle Fusion Applications Developer's Guide.

Using the deployFlex Command

From the WLST tool, execute the following command to deploy a flexfield, replacing `flex_code` with the code that identifies the flexfield, and replacing `flex_type` with the flexfield's type, which is either DFF, KFF, or EFF. The values must be wrapped in single-quotes.

```python
deployFlex('flex_code', 'flex_type')
```

Optionally, execute the following command if the flexfield is an extensible flexfield, and you want to deploy all the flexfield's configurations.
By default, extensible flexfields are partially deployed. That is, only the pages, contexts, or categories that had recent changes, are deployed.

deployFlex('flex_code', 'flex_type', ['force_Complete_EFF_Deployment'])
where, forceCompleteEFFDeployment=None

Using the deployPatchedFlex Command

Use the deployPatchedFlex command for situations where the patching framework doesn't invoke the command, such as when an application has been patched offline.

If the installation is multi-tenant enabled, the command deploys all patched flexfields for all enterprises. This command isn't intended to be invoked manually.

Check with your provisioning or patching team, or the task flows for managing flexfields, to verify that the flexfield has a Patched deployment status.

From the WLST tool, execute the following command to deploy the artifacts to the MDS partition.

deployPatchedFlex()

Execute the following command to deploy all flexfields that have either a READY status or an ERROR status.

deployPatchedFlex(mode='RETRY')

Using the deleteFlexPatchingLabels Command

Whenever you deploy flexfield changes to MDS using the deployPatchedFlex() WLST command, an MDS label is created in the format FlexPatchingWatermarkdate+time. Use the deleteFlexPatchingLabels command to inquire about and delete these labels.

From the WLST tool, execute the deleteFlexPatchingLabels () command with no arguments to delete the flexfield patching labels.

To output a list of flexfield patching labels, execute the command with the infoOnly argument, as follows:

deleteFlexPatchingLabels(infoOnly='true')

Using the validateFlexDeploymentStatus Command

The validateFlexDeploymentStatus() WLST command checks the deployment status of all flexfields in an Oracle Fusion Applications deployment.

validateFlexDeploymentStatus()

Use this command to verify that all flexfields in the current instance of provisioned Java EE applications are deployed.
Exiting the WLST and Checking the Results

To exit the tool, execute the following command.

disconnect()

Optionally, sign into the application, access user interface pages that contain flexfields, and confirm the presence of flexfields for which configuration exists, such as value sets, segments, context, or structures.

Manage Value Sets

Value Sets: Explained

A value set is a group of valid values that you assign to a flexfield segment to control the values that are stored for business object attributes.

An end user enters a value for an attribute of a business object while using the application. The flexfield validates the value against the set of valid values that you configured as a value set and assigned to the segment.

For example, you can define a required format, such as a five digit number, or a list of valid values, such as green, red, and blue.

Flexfield segments are usually validated, and typically each segment in a given flexfield uses a different value set. You can assign a single value set to more than one segment, and you can share value sets among different flexfields.

Caution

Be sure that changes to a shared value set are compatible with all flexfields segments using the value set.

The following aspects are important in understanding value sets:

- Managing value sets
- Validation
- Security
- Precision and scale
- Usage and deployment

Managing Value Sets

To access the Manage Value Sets page, use the Manage Value Sets task, or use the Manage Descriptive Flexfields and Manage Extensible Flexfields tasks for configuring a segment, including its value set. To access the Manage Values page, select the value set from the Manage Value Sets page, and click Manage Values. Alternatively, click Manage Values from the Edit Value Set page.
Validation

The following types of validation are available for value sets:

- Format only, where end users enter data rather than selecting values from a list
- Independent, a list of values consisting of valid values you specify
- Dependent, a list of values where a valid value derives from the independent value of another segment
- Subset, where the list of values is a subset of the values in an existing independent value set
- Table, where the values derive from a column in an application table and the list of values is limited by a WHERE clause

A segment that uses a format only value set doesn’t present a list of valid values to users.

Note

Adding table validated value sets to the list of available value sets available for configuration is considered a custom task.

Security

Value set security only works in conjunction with usage within flexfield segments.

You can specify that data security be applied to the values in flexfield segments that use a value set. Based on the roles provisioned to users, data security policies determine which values of the flexfield segment end users can view or modify.

Value set security applies at the value set level. The value set is the resource secured by data security policies. If a value set is secured, every usage of it in any flexfield is secured. It isn’t possible to disable security for individual usages of the same value set.

Value set security applies to independent, dependent, or table-validated value sets.

Value set security applies mainly when data is being created or updated, and to key flexfield combinations tables for query purposes. Value set security doesn’t determine which descriptive flexfield data is shown upon querying.

Security conditions defined on value sets always use table aliases. When filters are used, table aliases are always used by default. When predicates are defined for data security conditions, make sure that the predicates also use table aliases.

For key flexfields, the attributes in the view object that correspond to the code combination ID (CCID), structure instance number (SIN), and data set number (DSN) cannot be transient. They must exist in the database table. For key flexfields, the SIN segment is the discriminator attribute, and the CCID segment is the common attribute.
**Precision and Scale**

If the data type of a value set is Number, you can specify the precision (maximum number of digits user can enter) or scale (maximum number of digits following the decimal point).

**Usage and Deployment**

The usage of a value set is the flexfields where that value set is used. The deployment status of flexfields in which the value set is used indicates the deployment status of the value set instance.

The figure shows a value set used by a segment in a key flexfield and the context segment of a descriptive flexfield.

For most value sets, when you enter values into a flexfield segment, you can enter only values that already exist in the value set assigned to that segment.

Global and context-sensitive segment require a value set. You can assign a value set to a descriptive flexfield context segment. If you specify only context values, not value sets for contexts, the set of valid values is equal to the set of context values.

**Defining Value Sets: Critical Choices**

Validation and usage of value sets determine where and how end users access valid values for attributes represented by flexfield segments.
Tip

As a flexfield guideline, define value sets before configuring the flexfield, because you can assign value sets to each segment as you configure a flexfield. With descriptive and extensible flexfield segments, you can create value sets when adding or editing a segment on the runtime page where the flexfield appears.

The following aspects are important in defining value sets:

- Value sets for context segments
- Format-only validation
- Interdependent value sets
- Table validation
- Range
- Security
- Testing and maintenance

Value Sets for Context Segments

When assigning a value set to a context segment, you can only use table-validated or independent value sets.

You can use only table and independent value sets to validate context values. The data type must be character and the maximum length of the values being stored must not be larger than the context’s column length. If you use a table value set, the value set cannot reference flexfield segments in the value set’s WHERE clause other than the flexfield segment to which the value set is assigned.

Format Only Validation

The format only validation type enables end users to enter any value, as long as it meets your specified formatting rules. That is, the value must not exceed the maximum length you define for your value set, and it must meet any format requirements for that value set.

For example, if the value set allows only numeric characters, users can enter the value 456 (for a value set with maximum length of three or more), but can’t enter the value ABC. A format only value set doesn’t otherwise restrict the range of different values that users can enter. For numeric values, you can also specify if a numeric value should be zero filled or how many digits should follow the radix separator.

Interdependent Value Sets

Use an independent value set to validate input against a list that isn’t stored in an application table, and not dependent on a subset of another independent value set.
You cannot specify a dependent value set for a given segment without having first defined an independent value set that you apply to another segment in the same flexfield. Use a dependent value set to limit the list of values for a given segment based on the value that the end user has chosen for a related independent segment. The available values in a dependent list and the meaning of a given value depend on which value was selected for the independently validated segment.

For example, you could define an independent value set of U.S. states with values such as CA, NY, and so on. Then you define a dependent value set of U.S. cities, with values such as San Francisco and Los Angeles that are valid for the independent value CA, and New York City and Albany that are valid for the independent value NY. In the UI, only the valid cities can be selected for a given state.

Because you define a subset value set from an existing independent value set, you must define the independent value set first. End users don't need to choose a value for another segment first to have access to the subset value set.

Independent, dependent, and subset value sets require a customized list of valid values. Use the Manage Values page to create and manage a value set's valid values and the order in which they appear.

Tip

You can customize the Manage Value Sets page to capture additional attributes for each valid value by adding context-sensitive segments in a new context for FND_VS_VALUES_B descriptive field.

Table Validation

Typically, you use a table-validated set when the values you want to use are already maintained in an application table, such as a table of vendor names. Specify the table column that contains the valid value. You can optionally specify the description and ID columns, a WHERE clause to limit the values to use for your set, and an ORDER BY clause.

If you specify an ID column, then the flexfield saves the ID value, instead of the value from the value column, in the associated flexfield segment. If the underlying table supports translations, you can enable the display of translated text by basing the value set's value column on a translated attribute of the underlying table. You should also define an ID column that is based on an attribute that isn't language-dependent so that the value's invariant ID (an ID that doesn't change) is saved in the transaction table. This allows the run time to display the corresponding translated text from the value column for the run time session's locale.

Table validation lets you enable a segment to depend upon multiple prior segments in the same context structure. You cannot reference other flexfield segments in the table-validated value set's WHERE clause. That is, the WHERE clause cannot reference SEGMENT.segment_code or VALUESET.value_set_code.

Table-validated value sets have unique values across the table, irrespective of bind variables. The WHERE clause fragment of the value set is considered if it doesn't have bind variables. If it has bind variables, the assumption is that the values are unique in the value set.
Range

In the case of format, independent, or dependent value sets, you can specify a range to further limit which values are valid. You can specify a range of values that are valid within a value set. You can also specify a range validated pair of segments where one segment represents the low end of the range and another segment represents the high end of the range.

For example, you might specify a range for a format-only value set with format type Number where the user can enter only values between 0 and 100.

Security

In the case of independent and dependent values, you can specify that data security be applied to the values in segments that use a value set. Based on the roles provisioned to users, data security policies determine which values of the flexfield segment end users can view or modify.

To enable security on a value set, specify a database resource, typically the code value for the value set. Using the Manage Database Security Policies task, specify conditions, such as filters or SQL predicates, and policies that associate roles with conditions. You can use a filter for simple conditions. For more complex conditions, use a SQL predicate.

Value set data security policies and conditions differ from data security conditions and policies for business objects in the following ways:

- You can grant only read access to end users. You cannot specify any other action.
- When defining a condition that is based on a SQL predicate, use VALUE, VALUE_NUMBER, VALUE_DATE, VALUE_TIMESTAMP, or VALUE_ID to reference the value from a dependent, independent, or subset value set. For table value sets, use a table alias to define the table, such as &TABLE_ALIAS category=70.

When you enable security on table-validated value sets, the security rule that is defined is absolute and not contingent upon the bind variables (if any) that may be used by the WHERE clause of the value set. For example, suppose a table-validated value set has a bind variable to further filter the value list to x, y and z from a list of x, y, z, xx, yy, zz. The data security rule or filter written against the value set shouldn't assume anything about the bind variables; it must assume that the whole list of values is available and write the rule, for example, to allow x, or to allow y and z. By default in data security, all values are denied and show only rows to which access has been provided.

Testing and Maintenance

There is no need to define or maintain values for a table-validated value set, as the values are managed as part of the referenced table or independent value set, respectively.

You cannot manage value sets in a sandbox.
When you change an existing value set, the deployment status for all affected flexfields changes to Edited. You must redeploy all flexfields that use that value set to make the flexfields reflect the changes. In the UI pages for managing value sets, the value set's usages show which flexfields are affected by the value set changes.

If your application has more than one language installed, or there is any possibility that you might install one or more additional languages for your application in the future, select Translatable. This doesn't require you to provide translated values now, but you cannot change this option if you decide to provide them later.

**Planning Value Sets: Points to Consider**

The value sets you create and configure depend on the valid values on the business object attributes that will use the value set. When creating value sets, you first give the value set a name and description, and then define the valid values of the set.

The following aspects are important in planning value sets:

- List of values
- Plain text input
- Value ranges
- Value format specification
- Security

**List of Values**

You can use one of the following types of lists to specify the valid values for a segment:

- Table column
- Custom list
- Subset of custom list
- Dependent custom list

If the valid values exist in a table column, use a table value set to specify the list of values. To limit the valid values to a subset of the values in the table, use a SQL WHERE clause. Table value sets also provide some advanced features, such as enabling validation depending on other segments in the same structure.

Use an independent value set to specify a custom set of valid values. For example, you can use an independent value set of Mon, Tue, Wed, and so forth to validate the day of the week. You can also specify a subset of an existing independent value set as the valid values for a segment. For example, if you have an independent value set for the days of the week, then a weekend subset can be composed of entries for Saturday and Sunday.
Use a dependent value set when the available values in the list and the meaning of a given value depend on which independent value was selected for a previously selected segment value. For example, the valid holidays depend on which country you are in. A dependent value set is a collection of value subsets, with one subset for each value in a corresponding independent value set.

For lists of values type value sets, you can additionally limit the valid values that an end user can select or enter by specifying format, minimum value, and maximum value. For list of values type value sets, you can optionally implement value set data security. If the Oracle Fusion applications are running in different locales, you might need to provide different translations for the values and descriptions.

**Plain Text Input**

Use a format-only value set when you want to allow end users to enter any value, as long as that value conforms to formatting rules. For example, if you specify a maximum length of 3 and numeric-only, then end users can enter 456, but not 4567 or 45A. You can also specify the minimum and maximum values, whether to right-justify, and whether to zero-fill. With a format-only value set, no other types of validation are applied.

**Value Ranges**

You can use either a format-only, independent, or dependent value set to specify a range of values. For example, you might create a format-only value set with Number as the format type where the end user can enter only the values between 0 and 100. Or, you might create a format-only value set with Date as the format type where the end user can enter only dates for a specific year, such as a range of 01-JAN-93 to 31-DEC-93. Because the minimum and maximum values enforce these limits, you need not define a value set that contains each of these individual numbers or dates.

**Value Format**

Flexfield segments commonly require some kind of format specification, regardless of validation type. Before creating a value set, consider how you will specify the required format.

The following table shows options for validation type and value data type.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value data type</td>
<td>Character, Number, Date, Date Time.</td>
</tr>
<tr>
<td>Value subtype</td>
<td>Text, Translated text, Numeric digits only, Time (20:08), Time (20:08:08).</td>
</tr>
<tr>
<td></td>
<td>An additional data type specification for the Character data type for the Dependent, Independent, and Format validation types.</td>
</tr>
<tr>
<td>Maximum length</td>
<td>Maximum number of characters or digits for Character data type.</td>
</tr>
<tr>
<td>Precision</td>
<td>Maximum number of digits the user can enter.</td>
</tr>
</tbody>
</table>
Using Flexfields for Custom Attributes

<table>
<thead>
<tr>
<th>Scale</th>
<th>Maximum number of digits that can follow the decimal point.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uppercase only</td>
<td>Lowercase characters automatically changed to uppercase.</td>
</tr>
<tr>
<td>Zero fill</td>
<td>Automatic right-justification and zero-filling of entered numbers (affects values that include only the digits 0-9).</td>
</tr>
</tbody>
</table>

**Caution**

You cannot change the text value data type to a translated text value subtype after creating a value set. If there is any chance you may need to translate displayed values into other languages, choose Translated text. Selecting the Translated text subtype doesn’t require you to provide translated values.

**Value Sets for Context Segments**

You can use only table and independent value sets to validate context values. The data type must be character and the maximum length of the values being stored must not be larger than the context’s column length. If you use a table value set, the value set cannot reference flexfield segments in the value set’s WHERE clause other than the flexfield segment to which the value set is assigned.

**Security**

When enabling security on a value set, the data security resource name is an existing value set or one that you want to create. The name typically matches the code value for the value set.

**Restriction**

You cannot edit the data security resource name after you save your changes.

**Table-Validated Value Sets and Bind Variables: Points to Consider**

After you assign a value set to a flexfield, you can use bind variables in the WHERE clause.

The following bind variables refer to flexfield elements:

- :{SEGMENT.<segment_code>}
- :{CONTEXT.<context_code>;SEGMENT.<segment_code>}
- :{VALUESET.<value_set_code>}
- :{FLEXFIELD.<internal_code>}
- :{PARAMETER.<parameter_code>}

Using Flexfields for Custom Attributes 4-43
Segment Code

`:{SEGMENT.<segment_code>}`

This bind variable refers to the ID or value of a segment where `<segment_code>` identifies the segment. Where referring to the ID, the value set is ID-validated. Where referring to the value, the value set isn't ID-validated. The data type of the bind value is the same as the data type of the segment's column.

For both descriptive and extensible flexfields, the segment must be in the same context as the source segment. The source segment contains the WHERE clause. For descriptive flexfields, if the segment is global, then the source segment must be global.

The segment must have a sequence number that is less than the sequence number of the target segment with this bind variable. A matching segment must exist in the current flexfield context.

This bind variable is useful when the set of valid values depends on the value in another segment. For example, the values to select from a CITIES table might depend upon the selected country. If SEGMENT1 contains the country value, then the WHERE clause for the CITIES table might be `<country_code> = :{SEGMENT.SEGMENT1}`.

Context Code

`:{CONTEXT.<context_code>;SEGMENT.<segment_code>}`

This bind variable, which is valid only for extensible flexfields, refers to the ID (if the value set is ID-validated) or value (if not ID-validated) of a segment that is in a different context than the target segment (the segment with the WHERE clause).

- The `<context_code>` identifies the context and must be in the same category or in an ancestor category. It cannot be a multiple-row context.
- The `<segment_code>` identifies the segment. The data type of the bind value is the same as the data type of the segment's column.

Tip

The target segment should appear in the UI after the source segment to ensure the source segment has a value. If the target segment's context is a single-row context, the source and target segments must be on separate pages and the target page must follow the source page.

This bind variable is useful when the set of valid values depends on the value of a segment in another context. For example, the values to select from a CERTIFICATION table for a segment in the Compliance and Certification context might depend on the value of the country segment in the Manufacturing context.

Value Set Code

`:{VALUESET.<value_set_code>}`
This bind variable refers to the ID (if the value set is ID-validated) or value (if not ID-validated) of the segment that is assigned to the value set that is identified by the `value_set_code`. The data type of the bind value is the same as the data type of the segment's column.

The segment must have a sequence number that is less than the sequence number of the segment with this bind variable. If more than one segment is assigned to the value set, the closest prior matching segment will be used to resolve the bind expression. A matching segment must exist in the current flexfield context.

This bind variable is useful when the set of valid values depends on the value in another segment and that segment code can vary, such as when the value set is used for more than one context or flexfield. For example, the values to select from a CITIES table might depend upon the selected country. If the value set for the segment that contains the country value is COUNTRIES, then the WHERE clause for the CITIES table might be `<county_code> = :{VALUESET.COUNTRIES}`.

**Flexfield Internal Code**

`: {FLEXFIELD.<internal_code>}`

This bind variable refers to an internal code of the flexfield in which the value set is used, or to a validation date. The `internal_code` must be one of the following:

- **APPLICATION_ID** - the application ID of the flexfield in which this value set is used. The data type of APPLICATION_ID and its resulting bind value is NUMBER.

- **DESCRIPTIVE_FLEXFIELD_CODE** - the identifying code of the flexfield in which this value set is used. The data type of DESCRIPTIVE_FLEXFIELD_CODE and its resulting bind value is VARCHAR2. Note that you use this string for both descriptive and extensible flexfields.

- **CONTEXT_CODE** - the context code of the flexfield context in which this value set is used. The data type of CONTEXT_CODE and its resulting bind value is VARCHAR2.

- **SEGMENT_CODE** - the identifying code of the flexfield segment in which this value set is used. The data type of SEGMENT_CODE and its resulting bind value is VARCHAR2.

- **VALIDATION_DATE** - the current database date. The data type of VALIDATION_DATE and its resulting bind value is DATE.

**Flexfield Parameters**

`: {PARAMETER.<parameter_code>}`

This bind variable refers to the value of a flexfield parameter where parameter_code identifies the parameter. The data type of the resulting bind value is the same as the parameter's data type.

**Note**

You cannot assign a table value set to a context segment if the WHERE clause uses VALUESET.value_set_code or SEGMENT.segment_code bind variables.
Table-Validated Value Set: Worked Example

In an application user interface, you want to display a list of values that allow customers to enter satisfaction scores. The value column name is 1, 2, 3, 4, 5 and the value column description is Extremely Satisfied, Satisfied, and so on. Users can pick the appropriate value or description which stores the corresponding name so the name value can be used in a calculation expression.

In this case, you can use the FND_LOOKUPS table as the basis for a table-validated value set. The lookup meaning corresponds to the Value Column Name and the lookup description corresponds to the Description Column Name. The properties of the value set are as follows:

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>FROM clause</td>
<td>FND_LOOKUPS</td>
</tr>
<tr>
<td>WHERE clause</td>
<td>lookup_type = 'CN_XX_CUST_SATISFACT_SCORE'</td>
</tr>
<tr>
<td>ID column</td>
<td>lookup_code</td>
</tr>
<tr>
<td>Value column</td>
<td>meaning</td>
</tr>
<tr>
<td>Description column</td>
<td>description</td>
</tr>
<tr>
<td>Enable Flag column</td>
<td>enabled_flag</td>
</tr>
<tr>
<td>Start Date column</td>
<td>start_date_active</td>
</tr>
<tr>
<td>End Date column</td>
<td>end_date_active</td>
</tr>
<tr>
<td>Order by</td>
<td>display_sequence</td>
</tr>
</tbody>
</table>

After completing this task, you should have created your customer satisfaction value set for the Incentive Compensation page of your implementation project.

Creating a Value Set Based on a Lookup

1. From the Setup and Maintenance work area, find the Manage Value Sets task and click the Go to Task icon button.
2. On the Manage Value Sets page, click the Create icon button.
3. On the Create Value Set page, enter the following values:
   a. In the Value Set Code field, enter CN_XX_CUSTOMER_SATISFACTION_SCORES
   b. In the Description field, enter Customer satisfaction score.
   c. In the Module field, select Search....
   d. In the Search and Select: Module subwindow, enter Incent in the User Module Name field
   e. Select Incentive Compensation.
   f. Click OK.
4. On the Create Value Set page, enter the following values:
   a. In the Validation Type field, select Table.
   b. In the Value Data Type field, select Character.
c. In the Definition section FROM Clause field, enter FND_LOOKUPS.
d. In the Value Column Name field, enter DESCRIPTION.
e. In the Description Column Name field, enter MEANING.
f. In the ID Column Name field, enter LOOKUP_CODE.
g. In the Enabled Flag Column Name field, enter 'Y'.
h. In the Start Date Column Name field, enter START_DATE_ACTIVE.
i. In the End Date Column Name field, enter END_DATE_ACTIVE.
j. In the WHERE Clause field, enter LOOKUP_TYPE = 'CN_XX_CUST_SATISFACT_SCORE'.

5. Click Save and Close.
6. In the Manage Value Sets page, click Done.

**Adding Attributes to the Manage Value Sets Page: Procedures**

For independent, dependent, and subset value sets, you can add attributes to a value set. The attributes appear in the Manage Value Sets UI for capturing additional information about each valid value, such as its purpose.

Typically, these attributes are used to capture internal information. To display attributes on an application page, you must programmatically modify the application to access them.

1. Find the FND_VS_VALUES_B flexfield using the Manage Descriptive Flexfields task.
2. Open FND_VS_VALUES_B for editing.
3. Click Manage Contexts.
4. Create a new context and use the value set code for the context code.
5. Add the new attributes as context-sensitive segments.
6. Deploy FND_VS_VALUES_B to the run time.
7. Sign out and sign back in.
8. Open the Manage Value Sets page to view the new attributes.

**Translating Flexfield and Value Set Configurations: Explained**

When you first configure a flexfield or segment, the translatable text that you enter, such as prompts and descriptions, is stored as the text for all installed locales. You may then provide a translation for a particular locale. If you don’t provide a translation for a given locale, then the value that was first entered is used for that locale.

To translate the text for a particular locale, log in with that locale or specify the locale by selecting Settings and Actions - Personalization - Set Preferences
in the global area. Then, update the translatable text in the flexfield using the Manage Descriptive Flexfields task, Manage Key Flexfields task, or Manage Extensible Flexfields task. Your modifications change the translated values only for the current session's locale.

After you complete the translations, deploy the flexfield.

You can define translations for a dependent value set or an independent value set, if it is of type Character with a subtype of Translated text. You define the translations by setting the current session to the locale for which you want to define the translation and using the Manage Value Sets task to enter the translated values and descriptions for that locale.

For a table value set for which the underlying table supports multiple languages and for which the value set's value column is based on a translated attribute of the underlying table, you can define translated values using the maintenance task for the underlying table. For more information on using multilanguage support features, see the Oracle Fusion Applications Developer's Guide.

**FAQs for Manage Value Sets**

**What happens if a value set is security enabled?**

Value set security is a feature that enables you to secure access to value set values based on the end user's role in the system.

As an example, suppose you have a value set of US state names. When this value set is used to validate a flexfield segment, and users can select a value for the segment, you can use value set security to restrict them to selecting only a certain state or subset of states based on their assigned roles in the system.

For example, Western-region employees may choose only California, Nevada, Oregon, and so on as valid values. They cannot select non-Western-region states. Eastern-region employees may choose only New York, New Jersey, Virginia, and so on as valid values, but cannot select non-Eastern-region states. Value set security is implemented using Oracle Fusion Applications data security.

**How can I set a default value for a flexfield segment?**

When you define or edit a flexfield segment, you specify a default value from the values provided by the value set assigned to that segment.

You can set the default value for a descriptive flexfield segment to be a parameter, which means the entity object attribute to which the chosen parameter is mapped provides the initial default value for the segment.

You can set the default value to be a constant, if appropriate to the data type of the value set assigned to the segment.

In addition to an initial default value, you can set a derivation value for updating the attribute's value every time the parameter value changes. The parameter you choose identifies the entity object source attribute. Any changes in the value of the source attribute during run time are reflected in the value of the segment.
If the display type of the segment is a check box, you can set whether the default value of the segment is checked or unchecked.

Manage Descriptive Flexfields

Descriptive Flexfields: Explained

Descriptive flexfields provide a way to add custom attributes to entities, and define validation and display properties for them. These attributes are generally standalone. They don’t necessarily have anything to do with each other and aren’t treated together as a combination.

All Oracle Fusion Applications business entities that you can access are enabled for descriptive flexfields. Descriptive flexfields are optional. You can choose whether or not to configure and expose segments for the descriptive flexfield defined and registered in your database. For lists of descriptive flexfields, see assets with the Flexfield: Descriptive type in Oracle Enterprise Repository for Oracle Fusion Applications (http://fusionappsoer.oracle.com).

A descriptive flexfield provides a set amount of segments for an entity. You make the segments of a descriptive flexfield available to end users as individual fields in the application user interface.

Context

A descriptive flexfield can have only one context segment to provide context sensitivity.

The same underlying column can be used by different segments in different contexts. For example, you can define a Dimensions context that uses the ATTRIBUTE1 column for height, the ATTRIBUTE2 column for width, and the ATTRIBUTE3 column for depth. You can also define a Measurements context that uses the same columns for other attributes: the ATTRIBUTE1 column for weight, the ATTRIBUTE2 column for volume, and the ATTRIBUTE3 column for density.

Segments and Contexts

Descriptive flexfield segments are of the following types.

<table>
<thead>
<tr>
<th>Segment Type</th>
<th>Run Time Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global segment</td>
<td>Always available</td>
</tr>
<tr>
<td>Context segment</td>
<td>Determines which context-sensitive segments are displayed</td>
</tr>
<tr>
<td>Context-sensitive segment</td>
<td>Displayed depending on the value of the context segment</td>
</tr>
</tbody>
</table>

In the figure, a descriptive flexfield has one context segment called Category for which there are three values: Resistor, Battery, and Capacitor. In addition, the descriptive flexfield consists of two global segments that appear in each of the contexts, and three context-sensitive segments that only appear in the context in which they are configured.

Application development determines the number of segments available for configuring. During implementation, you configure the flexfield by determining the following:

- Which attributes to add using the available segments
- The context values
- The combination of attributes in each context

A segment can be used for different attributes, such as Height in Context1 and Color in Context2. Each segment of a descriptive flexfield that you make available to end users is exposed in the user interface as an individual field.

**Value Sets**

For each global and context-sensitive segment, you configure the values allowed for the segment and how the values that end users enter are validated, including interdependent validation among the segments.

**Planning Descriptive Flexfields: Points to Consider**

Once you have identified a flexfield to configure, plan the configuration in advance. Compile a list of the UI pages and other artifacts in your deployment that are affected by the configuration. Verify that you are provisioned with
the roles needed to view and configure the flexfield. View the flexfield using the Highlight Flexfields command in the Administration menu while viewing the run time page where the flexfield appears. Plan how you will deploy the flexfield for test and production users. Review the tools and tasks available for managing flexfields, such as the Define Flexfields task list, Manage Sandboxes, and Highlight Flexfields for adding and editing flexfield segments.

Planning a descriptive flexfield can involve the following tasks:

1. Identify existing parameters.
2. Identify existing context values and whether the context value is derived.
3. Identify custom attributes and plan the descriptive flexfield segments, segment properties, and structure.
5. Plan initial values.
6. Plan attribute mapping to Oracle Business Intelligence objects.

**Identify Existing Descriptive Flexfield Parameters**

Some descriptive flexfields provide parameters that can be used to specify the initial value of a descriptive flexfield segment. The parameter is external reference data, such as a column value or a session variable. For example, if a flexfield has a user email parameter, you can configure the initial value for a customer email attribute to be derived from that parameter.

Review the list of available parameters in the Derivation Value field in the Create Segment page for a descriptive flexfield. If you decide to use one of the parameters to set an initial value, select that parameter from the Derivation Value drop-down list when you add the descriptive flexfield segment.

**Evaluate Whether the Context Value Is Derived**

The context value for a descriptive flexfield might have been preconfigured to be derived from an external reference. For example, if the context is Marriage Status, then the value might be derived from an attribute in the employee business object. When the context value is derived, you might need to take the derived values and their source into consideration in your plan.

To determine whether the context value is derived, access the Edit Descriptive Flexfield task to view the list of configured context values for the flexfield. The Derivation Value field in the Context Segment region displays a list of available parameters.

If context values have been preconfigured, see Oracle Fusion Applications Help for product-specific information about the use of those values.

**Plan the Segments, Segment Properties, and Structure**

Identify the custom attributes you need for a business object to determine the segments of the descriptive flexfield. Determine the segment properties such as the prompt, display type, or initial value.
The structure of the descriptive flexfield is determined by its global, context, and context-sensitive segments. Plan a global segment that captures an attribute for every instance of the business object. Plan a context for segments that depend on a condition of situation applying to a particular instance of the business object. Plan context-sensitive segments to capture attributes that are relevant in the context.

There is only one context segment available for descriptive flexfields. If you have more than one group of custom attributes where you could use the context segment, you will have to pick one group over the others, based on your company's needs and priorities, and add the other custom attributes as global segments.

**Plan Validation Rules**

Define each segment's validation rules and check if value sets exist for those rules or you must create new ones. If you must create a value set, you can create it either before configuring the flexfield or while creating or editing a segment.

When determining a segment's validation rules, consider the following questions:

- What is the data type - character, date, date and time, or number?
- Does the segment require any validation beyond data type and maximum length?
- Should a character type value be restricted to digits, or are alphabetic characters allowed?
- Should alphabetic characters automatically be changed to uppercase?
- Should numeric values be zero-filled?
- How many digits can follow the radix separator of a numeric value? In base ten numerical systems the radix separator is decimal point.
- Does the value need to fall within a range?
- Should the value be selected from a list of valid values? If so, consider the following questions:
  - Can you use an existing application table from which to obtain the list of valid values, or do you need to create a custom list?
  - If you are using an existing table, do you need to limit the list of values using a WHERE clause?
  - Does the list of valid values depend on the value in another flexfield segment?
  - Is the list of valid values a subset of another flexfield segment's list of values?

**Plan Initial Values**

For every segment, list the constant value or SQL statement, if any, to use for the initial value of the custom attribute.
Plan How Segments Map to Oracle Business Intelligence Objects

If a descriptive flexfield has been enabled for Oracle Business Intelligence, you can make it available for use in Oracle Business Intelligence applications. You can use segment labels to map segments to logical objects. Plan to map segments to logical objects before deploying the flexfield as a way to streamline the import into Oracle Business Intelligence.

Use the Manage Segment Labels page to view preconfigured segment labels. If a segment label doesn’t exist for the logical object, then decide on a code, name, and description in preparation for adding that label. Choose a code, name, and description that will help end users select the correct label.

The mapping equalizes similar context-sensitive attributes that are from different contexts but are mapped to a single logical object. For information about objects in the logical model, see the “Working with Logical Tables, Joins, and Columns” chapter in the Oracle Fusion Middleware Metadata Repository Builder’s Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition).

Managing Descriptive Flexfields: Points to Consider

Configuring descriptive flexfields involves managing the available flexfields registered with your Oracle Fusion Applications database and configuring their flexfield-level properties, defining and managing descriptive flexfield contexts, and configuring global and context-sensitive segments.

Every descriptive flexfield is registered to include a context segment, which you may choose to use or not.

In general, configuring descriptive flexfields involves:

1. Creating segment labels for business intelligence enabled flexfields.
2. Configuring global segments by providing identity information, the initial default value, and the display properties.
3. Configuring the context segment by specifying the prompt, whether the context segment should be displayed, and whether a value is required.
4. Configuring contexts by specifying a context code, description, and name for each context value, and adding its context-sensitive segments, each of which is configured to include identifying information, the column assignment, the initial default value, and the display properties.

The following aspects are important in understanding descriptive flexfield management:

- Segments
- Adding Segments to a Highlighted Flexfield
- Usages
- Parameters
- Delimiters
You can assign sequence order numbers to global segments and to context-sensitive segments in each context. Segment display is always in a fixed order. You cannot enter a number for one segment that is already in use for a different segment.

Value sets are optional for context segments. The value set that you specify for a context segment consists of a set of context codes, each of which corresponds to a context that is appropriate for the descriptive flexfield. The value set must be independent or table-validated. If table-validated, the WHERE clause must not use the VALUESET.value_set_code or SEGMENT.segment_code bind variables. The value set must be of data type Character with the maximum length of values being stored no larger than the context's column length.

If you don't specify a value set for a context segment, the valid values for that context segment are derived from the context codes. The definition of each context segment specifies the set of context-sensitive segments that can be presented when that context code is selected by the end user.

For reasons of data integrity, you cannot delete an existing context. Instead, you can disable the associated context value in its own value set by setting its end date to a date in the past.

You can configure the individual global segments and context-sensitive segments in a descriptive flexfield. These segment types are differentiated by their usage, but they are configured on application pages that use most of the same properties.

**Adding Segments to a Highlighted Flexfield**

When you highlight flexfields on a run time page and use an Add Segment icon button to create a segment, the segment code, name, description, table column, and sequence number are set automatically. If you use an Add Segment icon button to configure descriptive flexfield segments, you cannot use an existing value set. Value sets are created automatically when you add the segments. You can enter the valid values, their descriptions, and the default value or specify the formatting constraints for the value set, such as minimum and maximum values.

Depending on display type, the value set you create with the Add Segment icon button is either an independent value set or a format-only value set. The table shows which type of value set is created depending on the segment display component you select.

<table>
<thead>
<tr>
<th>Display Component</th>
<th>Value Set Created with Add Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check box</td>
<td>Independent</td>
</tr>
<tr>
<td>Drop-down list</td>
<td>Independent</td>
</tr>
<tr>
<td>List of Values</td>
<td>Independent</td>
</tr>
<tr>
<td>Radio Button Group</td>
<td>Independent</td>
</tr>
</tbody>
</table>
### Tip
After you add a context value, refresh the page to see the new value.

### Usages
Descriptive flexfield usages allow for the same definition to be applied to multiple entities or application tables, such as a USER table and a USER_HISTORY table. Descriptive flexfield tables define the placeholder entity where the flexfield segment values are stored once you have configured the descriptive flexfield. When you configure a flexfield, the configuration applies to all its usages.

### Parameters
Some descriptive flexfields provide parameters, which are attributes of the same or related entity objects. Parameters are public arguments to a descriptive flexfield. Parameters provide outside values in descriptive flexfield validation. You use parameters to set the initial value or derivation value of an attribute from external reference data, such as a column value or a session variable, rather than from user input. Parameters can be referenced by the logic that derives the default segment value, and by table-validated value set WHERE clauses.

### Delimiters
A segment delimiter or separator visually separates segment values when the flexfield is displayed as a string of concatenated segments.

### Initial Values
The SQL statement defining an initial value must be a valid statement that returns only one row and a value of the correct type.

You can use two types of SQL statements:

- SQL statement with no binding. For example, select MIN(SALARY) from EMPLOYEES.

- SQL statement with bind variables. You can use the following bind variables in the WHERE clause of the SQL statement.

  - `{SEGMENT.<segment_code>}`: Identifies a segment in the same context.
  - `{CONTEXT.<context_code>;SEGMENT.<segment_code>}`: Identifies a segment in a different context. The context must be in the same
category or in an ancestor category, and it cannot be a multiple-row context.

- :{VALUESET.<value_set_code>}: Identifies the closest prior segment in the same context that is assigned to the specified value set.
- :{FLEXFIELD.<internal_code>}: Identifies a flexfield.

For more information about using bind variables, see the help for value sets.

**Business Intelligence**

Selecting a global, context, or context-sensitive segment’s BI Enabled checkbox specifies that the segment is available for use in Oracle Business Intelligence.

When the flexfield is imported into Oracle Business Intelligence, the label you selected from the BI Label dropdown list equalizes the segment with segments in other contexts, and maps the segment to the logical object represented by the label.

**Enabling Descriptive Flexfield Segments for Business Intelligence: Points to Consider**

A descriptive flexfield that is registered in the database as enabled for Oracle Business Intelligence (BI) includes a BI Enabled setting for each of its segments. When a global, context, or context-sensitive segment is BI-enabled, it is available for use in Oracle Business Intelligence.

The following aspects are important in understanding BI-enabled flexfield segments:

- Flattening business components to use BI-enabled segments in Oracle BI
- Equalizing segments to prevent duplication and complexity in the flattened component
- Mapping attributes of flattened business components to logical objects in Oracle BI
- Managing the labels that map segments to logical objects in Oracle BI

After you deploy a business intelligence-enabled flexfield, use the Import Oracle Fusion Data Extensions for Transactional Business Intelligence process to import the flexfield changes into the Oracle Business Intelligence repository. Users can make use of the newly-generated attributes in business intelligence applications. For example, a user can generate a report that includes attributes added by the descriptive flexfield. For additional information about logical objects and import, refer to the Oracle Transactional Business Intelligence Administrator’s Guide.

**Flattening**

When you deploy a business intelligence-enabled descriptive flexfield, the deployment process generates an additional set of flattened Application Development Framework (ADF) business components in addition to the usual ADF business components and ADF faces run time artifacts that are
generated during deployment. The flattened business components include attributes for business intelligence-enabled segments only. Flattening means each custom column in each context shows up as an attribute in an Oracle Business Intelligence folder.

Flattened components include one attribute for the BI-enabled context-segment, and one attribute for each business intelligence-enabled global segment. For BI-enabled context-sensitive segments, consider the following:

- If you assigned a label to the segment, the flattened components include an additional single attribute representing segments with that label.
- If you didn’t assign a label, the flattened components include a discrete attribute for each BI-enabled context-sensitive segment in each context.

**Mapping to Logical Objects in Business Intelligence**

You can simplify reporting by representing similar segments as a single logical object in Business Intelligence.

If you assign a label to any set of context-sensitive segments that serve the same purpose in different contexts, you can consolidate or equalize the segments into a single attribute. This prevents duplication and the extra workload and complexity that result from the flattening process. For example, a United States context might have a Passport segment and a Canada context might have Visa segment. If you assign the NationalID segment label to both the Passport and Visa segments, they are equalized into the same NationalID attribute in the flattened business component.

Non-labeled context-sensitive segments aren’t equalized across context values, so the flattened components include a separate attribute for each context-sensitive segment for each context value.

**Note**

It may not be possible to equalize similarly labeled segments if they have incompatible data types or value set types.

Assign a label to a global segment, context segment, or context-sensitive segment to map the corresponding attribute in the flattened components to a logical object in Oracle Business Intelligence. Using labels to map segments to BI logical objects minimizes the steps for importing the flexfield into Oracle Business Intelligence.

**Note**

Assigning a label to a context-sensitive segment serves to equalize the attribute across contexts, as well as map the equalized attribute to business intelligence.

**Managing Labels**

You may assign a predefined label (if available) to segments or create new labels for assignment, as needed. Specify a code, name, and description to identify each label. In the BI Object Name field, enter the name of the logical object in Oracle Business Intelligence to which the segment label should map during import. Specifying the BI logical object minimizes the steps for importing the flexfield into Oracle Business Intelligence and helps to equalize context-sensitive segments across contexts.
If no labels are assigned to a BI-enabled segment, or the BI Object Name on the assigned label doesn’t exist in business intelligence, you must manually map the segment to the desired logical object when importing into Oracle Business Intelligence. In addition, context-sensitive segments without labels cannot be equalized across context values. The flattened components include a separate attribute for each non-labeled context-sensitive segment in each context.

**Importing to Oracle Business Intelligence Repository**

After you deploy a business intelligence-enabled flexfield, import the flexfield changes into the Oracle Business Intelligence repository to make use of the newly flattened business components in business intelligence and then propagate the flexfield object changes. When you import the metadata into the Oracle Business Intelligence repository, you must do so as the FUSION_APPS_BI_APPID user.

---

**Note**

To import flexfield changes into the Oracle Business Intelligence repository in Oracle Cloud implementations, run the Import Oracle Fusion Data Extensions for Transactional Business Intelligence process. For additional information about import, refer to the Oracle Transactional Business Intelligence Administrator’s Guide.

---

**Tip**

When you import a flexfield into the Oracle Business Intelligence repository, you see both `<name>` and `<name>_c` attributes for each segment, along with some other optional attributes. The `<name>` attribute contains the value. The `<name>_c` attribute contains the code of the value set that the value comes from, and is used for linking to the value dimension. You must import both attributes.

---

**Manage Extensible Flexfields**

**Extensible Flexfields: Explained**

Extensible flexfields are like descriptive flexfields, with some additional features:

- You can add as many context-sensitive segments to the flexfield as you need. You aren’t restricted by the number of columns predefined and registered for the flexfield.
- You can configure a one-to-many relationship between the entity and its extended attribute rows.
  - A row of data can have multiple contexts associated with it.
  - A row of data can have multiple occurrences of the same context.
- You can configure contexts in groups so the attributes in the context always appear together in the user interface.
• You can use existing hierarchical categories so that entities inherit the contexts that are configured for their parents. Contexts are reusable throughout categories.

When you configure a context for multiple rows per entity, the segments are displayed as a table.

Unlike descriptive flexfields, the extension columns corresponding to extensible flexfields segments are part of extension tables, separate from the base application table. Unlike descriptive flexfield contexts, the set of attributes in an extensible flexfield context remains constant and doesn’t differ by context value.

An extensible flexfield describes an application entity, with the run time ability to expand the database that implementation consultants can use to define the data structure that appears in the application.

Extensible flexfields support one-to-many relationships between the entity and the extended attribute rows.

For lists of extensible flexfields, see assets with the Flexfield: Extensible type in Oracle Enterprise Repository for Oracle Fusion Applications (http://fusionappsoer.oracle.com).

The following aspects are important in understanding key flexfields:

• Usages
• Categories
• Pages
• Security

Usages

As with descriptive flexfields, you can define multiple usages for an extensible flexfield, which enables several application tables to share the same flexfield.

For example, a flexfield for shipping options can be used by both a Supplier table and a Buyer table. In addition, you can associate a context with one, some, or all of the flexfield’s usages. Thus, with the shipping information example, you can associate a warehouse context with the Supplier usage, a delivery location context with the Buyer usage, and a ship-via context with all usages.

Usages include security information for applying no security to user access or enforcing view and edit privileges. Some product-specific extensible flexfields have specialized usage fields beyond those for security.

Categories

You can configure multiple extensible flexfield contexts and group the contexts into categories. All extensible flexfields have at least one category. For some extensible flexfields, you can configure a hierarchy of categories. A child category in the hierarchy can inherit contexts from its parent category.

You can define categories for extensible flexfields, and you can associate any combination of contexts with a given category.
For example, the Electronics and Computers category hierarchy might include a Home Entertainment category, which in turn might include an Audio category and a TV category, and so on. The Home Entertainment product might have contexts that specify voltage, dimensions, inputs and outputs. Contexts are reusable within a given extensible flexfield. For example, the dimensions context could be assigned to any category that needs to include dimensional information.

Pages

Extensible flexfields let you combine contexts into groups known as pages, which serve to connect the contexts so they will always be presented together in the application user interface.

Each application page corresponds to one extensible flexfield category, with a separate region of the page for each associated context.

Security

When you configure a flexfield, you set the privileges for a context at the usage level by selecting actions for the view and edit privileges of a context usage.

When an end user performs a search, the user interface displays only the attribute values of the contexts for which the user has view privileges. The user is able to perform a search using all attributes for all contexts, regardless of view privileges.

If end users access a context through a web service, an exception is thrown if they perform an action for which they don't have privileges.

All extensible flexfields have a base data security resource. Some data security resources for extensible flexfields are preconfigured with actions that you can use to specify access privileges. If no action is preconfigured, a security administrator can create actions and policies to support access control on the extensible flexfield attributes.

Some extensible flexfields have a translatable option; these flexfields also have a translation data security resource.

Planning Extensible Flexfields: Points to Consider

Once you have identified a flexfield to configure, plan the configuration in advance. Compile a list of the UI pages and other artifacts in your deployment that are affected by the configuration. Verify that you are provisioned with the roles needed to view and configure the flexfield. View the flexfield using the Highlight Flexfields command in the Administration menu while viewing the run time page where the flexfield appears. Plan how you will deploy the flexfield for test and production users. Review the tools and tasks available for managing flexfields, such as the Define Flexfields task list, Manage Sandboxes, and Highlight Flexfields for adding and editing flexfield segments.

Planning an extensible flexfield can involve the following tasks:

1. Identify a hierarchical structure of categories.
2. Identify existing context values.
3. Identify custom attributes and plan the extensible flexfield segments, segment properties, and structure.
5. Plan initial values.
6. Plan security.
7. Plan attribute mapping to Oracle Business Intelligence objects.

**Category Hierarchy Structure**

Existing category hierarchy structures provide the framework for planning what segments to add to an extensible flexfield as custom attributes of an entity.

Some Oracle Fusion applications provide user interfaces to create and manage an extensible flexfield’s category hierarchy.

**Contexts and Existing Context Values**

If related custom attributes can be grouped together, plan adding the attributes as a context of segments, and plan the order in which the attributes should appear.

Some extensible flexfields have preconfigured context values. Region headers displayed in a the user interface page or pages that contain the flexfield segments identify existing contexts. Using the Manage Extensible Flexfields task, find and open the flexfield for editing to view the list of configured context values.

See product-specific information for guidance in using preconfigured context values.

**Plan the Segments and Segment Properties**

List all the custom attributes that you want to add as extensible flexfield segments.

For each segment, define properties, including the indexed property.

**Plan Validation Rules**

Define each segment’s validation rules and check if value sets exist for those rules or you must create new ones. If you must create a value set, you can create it either before you configure the flexfield or at the same time that you create or edit a segment.

When determining a segment’s validation rules, consider the following questions:

- What is the data type - character, date, date and time, or number?
- Does the segment require any validation beyond data type and maximum length?
• Should a character type value be restricted to digits, or are alphabetic characters allowed?

• Should alphabetic characters automatically be changed to uppercase?

• Should numeric values be zero-filled?

• How many digits can follow the radix separator of a numeric value? In base ten numerical systems the radix separator is decimal point.

• Does the value need to fall within a range?

• Should the value be selected from a list of valid values? If so, consider the following questions:
  • Can you use an existing application table from which to obtain the list of valid values, or do you need to create a custom list?
  • If you are using an existing table, do you need to limit the list of values using a WHERE clause?
  • Does the list of valid values depend on the value in another flexfield segment?
  • Is the list of valid values a subset of another flexfield segment's list of values?

Plan Initial Values

For every segment, list the constant value or SQL statement, if any, to use for the initial value of the custom attribute.

Plan Security

Determine what privileges to set for view and edit access to context attributes, such as providing all end users with view access but only managers with edit access.

If your security restrictions apply to several contexts, you can create generic actions. At a minimum, create the generic actions for the base data security resource. If the flexfield has a translatable option and you plan to use translatable contexts, then also create the generic actions for the translation data security resource. For example, if the Item flexfield supports the translatable option and has a data security resource ITEM_EFF_VL in addition to the base data security resource ITEM_EFF_B, then create actions for both data security resources, such as EDIT_NONTRANS_ATTRS for ITEM_EFF_B and EDIT_TRANS_ATTRS for ITEM_EFF_VL.

If your security restrictions are more fine-grained, such as needing to secure each context with a different privilege, then you can create more fine-grained actions.

Plan Which Segments Map to Oracle Business Intelligence Objects

If an extensible flexfield has been enabled for Oracle Business Intelligence, you can make the attributes available for use in Oracle Business Intelligence applications.
Managing Extensible Flexfields: Points to Consider

Configuring extensible flexfields involves managing the available flexfields registered with your application database.

The following sequence describes how to configure extensible flexfields:

1. Configuring contexts by creating each context segment and the context-sensitive segments for each context segment, and providing the following for each segments:
   a. Identifying information
   b. Column assignment
   c. Initial default value
   d. Display properties
2. Configuring context usages and usage security by selecting actions to which users should have access:
   • View
   • Edit
   • None, if no special privileges should be enforced.
3. Configuring categories and category details.
4. Associating contexts with a category.
5. Creating logical pages for a category.

The following aspects are important in understanding extensible flexfield management:

• Contexts
• Categories
• Initial values
• Indexed segments
• Pages
• Security
• Deployment

Contexts

Each context is displayed to end users as a region containing its context-sensitive segments. You can specify instruction help text to display instructions that explain how to use the region and its attributes to end users. Instruction help text is displayed at the top of the context region. A context can be defined as single
row or multi row. Single row contexts are the same as descriptive flexfields contexts. A single row context has only one set of context-sensitive segments. A multi-row context enables you to associate multiple sets of values with the same object instance.

For example, for a BOOK table, you could create a multi row context named chapters that contains a segment for chapter and a segment for number of pages. Multiple chapters can then be associated with each book in the BOOK table.

For contexts that store multiple rows, you can uniquely identify each row by having the values in each row form a unique key.

If flexfield has a category hierarchy, then you can leverage the hierarchy to reuse contexts for similar entities, such as similar items in a product catalog.

Set the context to translatable so that free-form text entered by end users is stored in the language of the user’s locale, and different translations of that text can be stored in other languages. Segments in the translated contexts should utilize format-only value sets for storing free-form, user-entered text.

Set the context security to give an end user view or edit access to a context. The context’s task flow and region appear in the user interface only for users with view access. With edit access, an end user can edit the context’s attribute values. With no action specified for a usage, no special privileges are enforced through the context’s configuration.

**Categories**

A category is a grouping of related data items that can be considered to belong together. You can associate any combination of contexts with a given category. Extensible flexfields with more than 30 categories must be deployed as a background process.

A category hierarchy logically organizes a set of categories. For example, the Electronics and Computers category hierarchy might include a Computer category and a Home Entertainment category, which in turn might include an Audio category and a TV category, and so on.

A category can be a child or sibling of an existing category. The hierarchy can be as simple or as complex as desired, with any combination of zero or more sibling categories and zero or more child categories. If no category is defined, the data items are grouped under a single predefined default category.

Each category has associated contexts that store relevant information about a data item in that category. For example, a Home Entertainment product has contexts that specify Voltage, Dimensions, Inputs and Outputs. Contexts are reusable within a given extensible flexfield; the Dimensions context could be assigned to any category that needs to include dimensional information.

If a hierarchy includes child categories, each child category inherits the contexts from its parent category; for example, the Home Entertainment category inherits Voltage and Dimensions from the Electronics and Computers category.

Each extensible flexfield is associated with a particular category hierarchy. Consider category hierarchies to be defining framework for extensible flexfields and their contexts. A category hierarchy specifies which contexts are valid for each category.
An extensible flexfield can include multiple contexts which you define to support a given category. These contexts can be suitable for various purposes, but within a particular category, some contexts might be considered to be related to, or dependent on, each other. You can combine these contexts into groups known as logical pages, and determine the sequence in which the pages appear. This serves to connect the contexts so they will always be presented together and in a particular order in the application user interface.

For example, the Home Entertainment category might have an Electrical Specifications page that contains the Voltage, Inputs and Outputs contexts, and a Physical Specifications page that contains the Dimensions and Form Factor contexts.

**Initial Values**

The SQL statement defining an initial value must be a valid statement that returns only one row and a value of the correct type.

You can use two types of SQL statements:

- SQL statement with no binding. For example, select MIN(SALARY) from EMPLOYEES.

- SQL statement with bind variables. You can use the following bind variables in the WHERE clause of the SQL statement.
  
  - :{\text{\texttt\{SEGMENT.\texttt<segment_code>\}}}: Identifies a segment in the same context.
  
  - :{\text{\texttt\{CONTEXT.\texttt<context_code>\};SEGMENT.\texttt<segment_code>\}}: Identifies a segment in a different context. The context must be in the same category or in an ancestor category, and it cannot be a multiple-row context.

  - :{\text{\texttt\{VALUESET.\texttt<value_set_code>\}}}: Identifies the closest prior segment in the same context that is assigned to the specified value set.

  - :{\text{\texttt\{FLEXFIELD.\texttt<internal_code>\}}}: Identifies a flexfield.

For more information about using bind variables, see the help for value sets.

**Indexed Segments**

You can designate an extensible flexfield segment as indexed so that it is one of the selectively required attributes an end user can use in an attribute search. If you indicate in the Manage Extensible Flexfield UI page that a segment should be indexed, the column representing the segment must be added to the database index. Commonly, a database administrator (DBA) adds columns to the database index.

When an extensible flexfield with indexed segments is deployed, search task flows are generated along with the other flexfield artifacts and specify the indexed attributes as selectively required. In the deployed extensible flexfield’s search task flow, an end user must specify at least one of the indexed attributes in the search criteria. This prevents non-selective searches, which could cause performance issues.
For example, if you index the memory and processor attributes and ensure that the corresponding columns in the database are indexed, an end user can search an item catalog for computers by entering processor or memory or both as a search criteria. No search is performed if an end user enters an attribute that isn’t indexed as a search criterion.

Pages

Define logical pages to group contexts together in the user interface. For a given category, you may create one or more logical pages. You may add one or more of the category’s associated contexts to each of the category’s logical pages.

You can specify:

- The sequence of the contexts within each page.
- The sequence in which the logical pages appear.
- Instruction help text to display instructions that explain how to use the page to end users. Instruction help text is displayed at the top of the logical page, preceding all of its context regions.

Security

An extensible flexfield's base data security resource typically has a name with an _B suffix. The translation data security resource is a view of a translation table that typically has a name with an _VL suffix.

If a flexfield supports the translatable option and has a translation data security resource, make sure that you create the action for the appropriate data security resource.

- If you create a context-specific action for a nontranslatable context, add it to the base data security resource.
- If you create a context-specific action for a translatable context, add it to the translation data security resource.

Deployment

You can only deploy extensible flexfields using the Manage Extensible Flexfields task. You can deploy extensible flexfields offline as a background process and continue working in the session without having to wait for the deployment to complete. You can add one after another extensible flexfield to your deployment queue when you deploy as a background process. The flexfields are deployed, one at a time, in the order that you deploy them to the queue. You must deploy extensible flexfields with more than 30 categories as a background process.

You can remove an extensible flexfield from the deployment queue with the Cancel Background Deployment command.

When an extensible flexfield is deployed in a background process, its offline status indicates that the flexfield is in a background deployment process. A
flexfield’s offline status is cleared and its deployment status updated when the background deployment process has completed.

**Note**

The **Offline Status** column refreshes when you perform a new search in the **Manage Extensible Flexfields** task.

## Managing Extensible Flexfield Categories: Points to Consider

Categories are a way of extending the number of context-sensitive segments for a flexfield beyond the columns reserved for flexfield segments.

An Items extensible flexfield has a category for each item and each category can have one or more contexts. The laptop item belongs to the Computers category. Since extensible flexfields are mapped to separate extension tables, not just to columns as with descriptive flexfields, the thirty reserved columns on the extensible flexfield table let you define up to thirty context-sensitive segments for each context.

If you add a Dimensions context to the Computers category, thirty segments are available. But if you need to add more than thirty attributes, create another context and associate it to the same category. You could now add an Electronics Attributes context to the same Computers category in which you create another thirty segments.

You can continue creating more contexts and adding them to the Computers category. In this way your laptop computer item can be extended with as many attributes as you need, because it is mapped to a category and you can keep adding contexts to that category.

A descriptive flexfield on an items table with thirty columns reserved for segments can only have a single context. Once you configure the columns for that one context, you cannot create any more segments.

### Predefined and Preconfigured Categories

How you structure the flexfield configuration depends on how categories are defined for the flexfield. If the extensible flexfield is preconfigured with one category, associate all your contexts and pages with that category. If a product-specific extensible flexfield is preconfigured with several categories, associate your contexts and pages with those categories. If the extensible flexfields provide user interfaces for configuring multiple categories, associate a context with more than one category using inheritance.

Some products provide an activity or task for creating and maintaining categories for an extensible flexfield. See product-specific information to determine if you can create categories for the flexfield.

You can view a flexfield’s category hierarchies by using either the Highlight Flexfields feature or the Manage Extensible Flexfields task to find and open the flexfield for editing.
Disabling Categories

While configuring an extensible flexfield, you can disable a category. The Enabled column in the Category table of the Edit Extensible Flexfield page, indicates which categories are enabled.

Warning

When you deploy an extensible flexfield that has a disabled category, that category and its descendant categories aren’t deployed. Contexts and their segments are deployed only if they belong to at least one enabled category.

Contexts

Group similar custom attributes into contexts. The group is displayed together in a region. The region’s header is the context value.

If a category hierarchy exists for the flexfield, then you can leverage the hierarchy to reuse contexts for similar entities, such as similar items in a product catalog.

The figure shows the Item Extended Attributes flexfield, which uses the category hierarchy feature to reuse contexts. The flexfield’s Electronics and Computers category contains contexts for compliance and certification, voltage, and materials and substances. The TV and Video subcategory and the Computer Products subcategory inherit the Electronics and Computer contexts in addition to having their own contexts. The Materials and Substances context belongs to both the Electronics and Computer Products category and the Tools, Auto, and Industrial Products category.

The table shows an example of category hierarchy for an extensible flexfield.

<table>
<thead>
<tr>
<th>Display Name</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electronics and Computers</td>
<td>PROD_ELECTRONICS</td>
<td>Electronics and Computers</td>
</tr>
<tr>
<td>• TV and Video</td>
<td>PROD_TV_VIDEO</td>
<td>Television and Video</td>
</tr>
<tr>
<td>• Computers</td>
<td>PROD_COMPUTERS</td>
<td>Computers</td>
</tr>
<tr>
<td>Office Products and Supplies</td>
<td>PROD_OFFICE_PRODUCTS_SUPPLIES</td>
<td>Office Products and Supplies</td>
</tr>
</tbody>
</table>
To store voltage information for all electronic and computer items, associate a Voltage context with the Electronics and Computers category. Both the TV and Video subcategory and the Computers subcategory then inherit the Voltage context from the parent Electronics and Computers category.

**Configuring an Item Extended Attributes Flexfield: Example**

The Item Extended Attributes flexfield provides segments for extending the Item business object. In the Manage Extensible Flexfields task, you configure your product business object to include a Technical Specifications logical page in the user interface for the Electronics and Computers category of items.

In this example, your configuration of this flexfield groups custom attributes into the following contexts:

- Materials and Substances
- Compliance and Certification
- Voltage

**Scenario**

The following list shows an example plan for custom computer attributes for the Item Extended Attributes flexfield. In this example, the Electronics Information page is inherited from the parent Electronics and Computers category.

- Page: Electronics Information
  - Context: Compliance and Certification, single row
    - ISO 14001 (International Organization for Standardization for an Environmental Management System)
    - ENERGY STAR (energy efficiency guidelines)
    - ROHS (Restriction of the use of certain hazardous substances in electrical and electronic equipment)
  - Context: Voltage, single row
    - Minimum voltage
    - Maximum voltage
    - Current type
  - Context: Materials and Substances, multiple rows
    - Material
    - Contain recyclate
The following table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which extensible flexfield is available for configuring a hierarchy of categories?</td>
<td>Item Extended Attributes flexfield</td>
</tr>
</tbody>
</table>

**Collecting Technical Specifications**

Your product inventory pages for electronics and computers require a technical specifications page. Your product inventory pages for furniture require a furniture specifications page and an assembly instructions page. Items in both the electronics and computer category, and in the furniture category, share attributes for specifying materials and substances.

The figure shows a Technical Specifications logical page in the user interface for the Electronics and Computers category, with attributes in the context of Recovery and Recycling, Compliance and Certification, Operating Conditions, and Materials and Substances. The Materials and Substances context is configured for multiple rows so your users can select all the materials and substances required to make a single product, displayed as attribute values in a table.
Analysis

You use logical pages to arrange how the contexts appear in the user interface. Use a context to store all the materials and substances required to make a single product. You can configure a context to store multiple rows per entity. The multiple rows are displayed in a table, as for the Materials and Substances context.

The Technical Specifications logical page contains the attributes for the four contexts.

- Recovery and Recycling
- Compliance and Certification
- Operating Conditions
- Materials and Substances

In the figure, the Furniture category is configured to include a Furniture Specifications logical page and an Assembly Instructions logical page. The two categories (Electronics & Computers and Furniture) share the Materials & Substances context.

Configure Security for the Item Flexfield Configuration

The following table shows an example of data security policies for the Item flexfield.

<table>
<thead>
<tr>
<th>Data Security Resource</th>
<th>Policy</th>
<th>Role</th>
<th>Actions</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITEM_EFF_B</td>
<td>A</td>
<td>VOLTAGE_SPEC</td>
<td>edit_nontrans_voltage</td>
<td>All values</td>
</tr>
<tr>
<td>ITEM_EFF_VL</td>
<td>B</td>
<td>COMPLIANCE_SPEC</td>
<td>edit_trans_compliance</td>
<td>All values</td>
</tr>
<tr>
<td>ITEM_EFF_VL</td>
<td>C</td>
<td>COMPUTER_SPEC</td>
<td>edit_trans_attrs</td>
<td>ComputerCategoryFilter</td>
</tr>
<tr>
<td>ITEM_EFF_VL</td>
<td>D</td>
<td>TELEVISION_SPEC</td>
<td>edit_trans_attrs</td>
<td>TVCategoryFilter</td>
</tr>
</tbody>
</table>

The following table shows the privileges for three of the flexfield’s contexts.
<table>
<thead>
<tr>
<th>Context</th>
<th>Edit Privilege</th>
<th>View Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>edit_nontrans_voltage_ctx</td>
<td>NONE</td>
</tr>
<tr>
<td>Compliance and Certification</td>
<td>edit_trans_compliance_ctx</td>
<td>NONE</td>
</tr>
<tr>
<td>Materials and Substances</td>
<td>edit_trans_attrs</td>
<td>NONE</td>
</tr>
</tbody>
</table>

In this example, anyone can view the contexts’ attributes, but the edit privileges are restricted as follows:

- Voltage: Editable only by voltage specialists.
- Compliance and Certification: Editable only by compliance specialists.
- Materials and Substances: Only computer specialists can edit these attributes for items in the computer category. Only television specialists can edit these attributes for items in the TV category.

In this example, the Materials and Substances context is secured by a generic action with a condition applied to restrict access by category. Voltage and Compliance and Certification are secured by actions specific to each context.

**FAQs for Manage Extensible Flexfields**

**Why did the extensible flexfield context not appear at run time?**

If a deployed extensible flexfield context doesn’t appear in the user interface, verify that the context is associated with one of the category’s pages defined for the extensible flexfield.

**Manage Key Flexfields**

**Key Flexfields: Explained**

Key flexfields provide a means to capture a key such as a part number, a job code, or an account code. A key flexfield consists of one or more segments, where each segment can have a meaning.

For example, a part number 10-PEN-BLA-450 might correspond to a black pen from vendor #450 sold by division #10 (office supplies). Behind the scenes, the application uses a unique number, 13452, for this part, but the end user always sees the 10-PEN-BLA-450 part number.

The following aspects are important to understanding key flexfields:

- Architecture
- Segments and segment labels
- Structures
Using Flexfields for Custom Attributes

- Segment and structure instances
- Combinations
- Dynamic combination creation
- Security

Key flexfields aren’t optional. You must configure key flexfields to ensure that your applications operate correctly. You configure and maintain key flexfield definitions with the Manage Key Flexfields task.

For lists of key flexfields, see assets with the Flexfield: Key type in Oracle Enterprise Repository for Oracle Fusion Applications (http://fusionappsoer.oracle.com).

Note

For information about specific key flexfields, see the Oracle Fusion Applications Help for the product where the associated business component is implemented.

Architecture

When you configure a key flexfield, you define metadata about the key flexfield such as how many segments are in a structure, how many structures the flexfield uses, what value sets each segment uses, and so on. Flexfield metadata is stored in flexfield metadata tables.

Based on the flexfield metadata, actual part numbers are captured at run time as a combination of segment values and stored in a combinations table. A combinations table contains all the segment columns for a flexfield, plus a unique ID column and a structure instance number column that differentiates multiple arrangements of the segment columns.

For example, a part number that can be comprised of multiple segments can be represented by a key flexfield. A part number key flexfield has a corresponding combinations table, where the flexfield stores a list of the complete codes, with one column for each segment of the code, together with the corresponding unique ID and structure instance number for the code. When users define a new part number or maintain existing part numbers in the parts catalog, they directly maintain rows in the combinations table.

The foreign key table contains a different business entity than the combinations table. For example, the business entity in the foreign key table is order lines or invoice lines that contain foreign key references to parts for ordering and so on. Any number of foreign key tables can reference a particular entity represented by a key flexfield.

Segments and Segment Labels

A key flexfield consists of segments. Segments consist of a prompt, a short prompt, display width, a number that determines where in the sequence of a key flexfield structure the segment exists, the range type and the column name of the attribute being captured by the segment, a default value set and a label for
the segment. A segment label identifies a particular segment of a key flexfield. Segment labels are defined and made available by applications development.

Applications identify a particular segment for some purpose such as security or computations. Segment name or segment order cannot reliably identify a segment because key flexfield segments can be configured to appear in any order with any prompts. A segment label functions as a tag for a segment.

For example, Oracle Fusion General Ledger needs to identify which segment in the Accounting Flexfield contains balancing information and which segment contains natural account information. General Ledger uses a segment label to determine which segment you are using for natural account information. When you define your Accounting Flexfield, you must specify which segment label apply to which segments.

Some labels must be unique, and cannot be applied to more than one segment in each structure. Other labels are required, and must be applied to at least one segment in each structure.

A segment label orients an end user’s search of segments, such as the Cost Center label for all segments across key flexfields that capture a value for cost center.

**Structures**

A key flexfield structure definition includes the number of segments and their order.

In some applications, different users need to see different segment structures for the same flexfield. A key flexfield can have multiple structures if registered to support more than one structure.

The flexfield can display different fields for different end users based on a data condition in your application data, such as the value of another field entered by the end user or the user’s role. For example, the correctly formatted local postal address for customer service inquiries differs based on locale. A postal address key flexfield could display different segments and prompts for different end users based on a location condition in your application data, such as the user’s role or a value entered by the user.

Each structure can have one or more segments. Thus a segment is a child of a structure. If you want to store a particular segment, such as Cost Center, in two different structures, you must define the segment separately in each structures.

Each structure may have one or more structure instances. Each instance of a structure shares the same number and order of segments, but differs in the allowable values or value sets that validate the segments.

**Structure and Segment Instances**

You can define multiple configurations of a key flexfield structure. These structure instances have the same segment structure, in the same sequence order. They differ primarily in how each segment is validated. You define a structure instance for each key flexfield and each key flexfield structure instance.

The segments in a key flexfield structure instance are segment instances. A segment instance is a segment with a specific value set assigned to it.
If a key flexfield has been registered with a tree structure, you can specify a tree code for a segment instance, where the tree code defines a hierarchical relationship between the segment values.

Combinations

A combination is a complete code, or combination of segment values that makes up the code, that uniquely identifies an object.

For example, each part number is a single combination, such as PAD-YEL-11x14 or 01-COM-876-7BG-LTN. In these combinations, the hyphen is the segment separator. If you have ten parts, define ten combinations. A valid combination is simply an existing or new combination that can currently be used because it isn't out of date or disabled, and doesn’t violate cross-validation or security rules. A combination has different segments depending on the flexfield structure being used for that combination. Any combination is associated with only one particular flexfield structure.

Many Oracle Fusion Applications products refer to a key flexfield combination by using the name of the entity or the key flexfield itself. For example, Oracle Fusion Assets uses the asset key flexfield and refers to one of its combinations as an asset key or asset key flexfield. In another example, other Oracle Fusion Applications products including Oracle Fusion General Ledger (GL) refer to combinations of the accounting flexfield as account or GL account.

Each key flexfield has one corresponding table, known as the combinations table, where the flexfield stores a list of the complete codes, with one column for each segment of the code, together with the corresponding unique ID number (a code combination ID number or CCID) for that code. Then, other tables in the application have a column that stores just the unique ID for the code. For example, you may have a part number code, such as PAD-YEL-11x14. The Parts combinations table stores that code along with its ID, 57494. If your application lets you take orders for parts, you might then have an Orders table that stores orders for parts. That Orders table would contain a single column that contains the part ID, 57494, instead of several columns for the complete code PAD-YEL-11x14.

Typically, one combinations page maintains the key flexfield, where the key flexfield is the representation of an entity in your application. Maintain individual combinations, such as part numbers in the combinations page.

Dynamic Combination Creation

Dynamic combination creation is the insertion of a new valid combination into a combinations table from a page other than the combinations page.

Dynamic combination creation may be enabled at the following levels.

<table>
<thead>
<tr>
<th>Level Of Dynamic Combination Creation</th>
<th>Controlled By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexfield</td>
<td>Application development</td>
</tr>
<tr>
<td>Each usage or reference to the key flexfield</td>
<td>Application development</td>
</tr>
<tr>
<td>Structure instance</td>
<td>Administrators and implementation consultants</td>
</tr>
<tr>
<td>Other</td>
<td>Administrators and implementation consultants</td>
</tr>
</tbody>
</table>
If your key flexfield or certain usages or references of the key flexfield don’t permit dynamic combination creation, you may control whether dynamic combination creation is enabled for each structure instance. If enabled, a user can enter a new combination of segment values using the flexfield window from a foreign key page. For example, when entering a transaction, a GL user can enter a new expense account code combination for an account that doesn’t yet exist. Your application creates the new account by inserting the new combination into the combinations table behind the scenes. Assuming that the new combination satisfies any existing cross-validation rules, the flexfield inserts the new combination into the combinations table, even though the combinations table isn’t the underlying table for the foreign key page.

Planning Key Flexfields: Points to Consider

Your first step in planning your key flexfields is to determine which key flexfields your application requires.

Your plan should include:

• The purpose of the key flexfield
• The number and length of its available segment columns
• Whether your key flexfield allows more than one structure
• Whether more than one structure must be defined
• The number, order and length of your segments for each structure

Consider the following aspects in planning flexfields:

• Before you begin
• Access to flexfield-related tasks
• Restrictions
• Validation rules for flexfield segments

Before You Begin

Once you have identified a flexfield to configure, plan the configuration in advance. Compile a list of the UI pages and other artifacts in your deployment that are affected by the configuration. Verify that you are provisioned with the roles needed to view and configure the flexfield. View the flexfield using the Highlight Flexfields command in the Administration menu while viewing the run time page where the flexfield appears. Plan how you will deploy the flexfield for test and production users.

Review the tools and tasks available for managing flexfields, such as the Define Flexfields task list and Manage Sandboxes.
If you plan to use value sets, create them before configuring the key flexfield. You cannot create value sets for key flexfields at the time that you add and configure key flexfield segments.

**Access to Flexfield-Related Tasks**

To access tasks for configuring flexfields and value sets, you must be provisioned with roles that entitle you to access the tasks in the Define Flexfields task list or tasks for managing product-specific flexfields. Contact your security administrator for details. For information about product-specific flexfield tasks, such as Manage Fixed Assets Key Flexfields, consult the product-specific documentation in Oracle Fusion Applications Help.

**Restrictions**

If you plan to use value sets, create them before configuring the flexfield.

Plan your key flexfield configuration to scale to your enterprise needs. For example, if you expect to disable old cost centers and enable new ones frequently, plan a larger maximum size for your cost center value set so that you can have more available values. One thousand available values for a 3-character value set provides more room for changes than 100 available values for a 2-character value set.

Note the code name of the flexfield you intend to configure so you can find it easily in the Define Flexfield task list or tasks for managing product-specific key flexfields.

In some cases you can customize how the flexfield appears on the page.

See Oracle Fusion Applications Help for specific products to determine any restrictions on using product-specific key flexfields.

**Reporting**

If you want to report on your data by certain criteria or sub-entities, such as account number or project or region, consider making that sub-entity a distinct segment, rather than combining it with another sub-entity, so that you can categorize and report on smaller discrete units of information.

**Managing Key Flexfields: Points to Consider**

Consider the plans for a key flexfield, security, and resulting run time pages when configuring key flexfields.

**Planning**

Plan structures carefully and allow for future needs.

---

Caution
Don’t change the number, order, and maximum length of segments once you have acquired flexfield data.

**Structure Delimiters**

A delimiter separates the segments when they appear to end users. The delimiter value of a structure specifies the character used to visually separate segment values when the key flexfield is displayed as a string of concatenated segments in the UI.

**Tip**

Choose the delimiter value of your key flexfield carefully so that it doesn’t conflict with the flexfield data. For example, if your data frequently contains periods, such as in monetary or numeric values, don’t use a period as your segment separator. Any character you expect to appear frequently in your segment values or descriptions isn’t a good choice for the delimiter.

If you change the configuration of a key flexfield, such as the delimiter, the change affects the previously stored key flexfields with that structure.

**Security**

Oracle Fusion data security enforces value set security.

Within key flexfields, value set security applies to the selection of the individual segment values in the segment list of values. When selecting a key flexfield segment value from the combinations table, data security allows display of only the combinations whose segment values you have access to. Applications development controls whether or not value set security rules propagate to the foreign key table. By default they do.

**Run time Pages**

Application development determines the user interface (UI) pages used to render flexfields. The types of key flexfield UI pages are as follows:

- Combinations pages where the underlying entity objects use the combinations table itself
- Foreign key pages where the underlying entity objects contain a foreign key reference to the combinations table
- Partial usage pages where some or all of the key flexfield’s segment columns are in a product table

The same key flexfield can be used in different ways on different pages.

A page with a foreign key reference has a base table or view that contains a foreign key reference to a combinations table with the actual flexfield segment columns. This lets you manipulate rows containing code combination IDs (CCID).

A page with partial usage of a key flexfield presents segments that are defined on a product’s transactional table in addition to being defined on a combinations table. In the case of a partial usage page, it is possible that only part of the
configuration is visible. This enables the key flexfield to behave more like a descriptive flexfield.

A code combination maintenance page or combinations page presents the combinations table. This enables directly creating and maintaining code combinations. The combinations table contains all key flexfield segment columns and a unique ID column.

A typical application has only one combinations page. An application might not have a combinations page if it doesn't support maintenance by administrators.

A page containing a search region enables end users to select which attributes of the key flexfield view object to use as criteria to search for flexfield metadata.

For example, you can configure seven segments for the Account key flexfield. In a foreign key reference page, end users see the typical key flexfield picker with all seven segments where they can search for combinations. In a partial usage page using the same key flexfield, end users potentially could see only a single segment such as the Cost Center labeled segment, or they might see multiple segments but displayed as individual segments rather than as a picker for choosing combinations.

For more information on key flexfield pages, see the Oracle Fusion Applications Developer's Guide.

Key Flexfield Structures: Explained

A key flexfield structure arranges the segments of a key so that you can reuse a single key flexfield in multiple combinations of the same segments or a subset of those segments. Multiple instances of a single structure can accommodate differences in the value sets assigned to the structure's segments.

The structure determines the following aspects of a key flexfield:

- The segments to include
- The order of the segments
- Segment labels on the included segments
- Properties for each segment applied to the instances of the segments in an instance of the structure

Managing Key Flexfield Structures

All the segments defined for a key flexfield are available to be included in a key flexfield structure.

You can define as many segments as there are defined segment columns in your key flexfield combinations table.

Restriction
Be sure to add segments in the order that your key requires. Once deployed, the order cannot be changed.

Enable segments to indicate that they are in use. A flexfield doesn't display disabled segments in run time.

Tip
To protect the integrity of your data, disable a segment if you have already used it to enter data.

Key Flexfield Structure Instances and Segment Instances: Explained

A key flexfield structure can have one or more alternate structure instances.

The instances of a key flexfield structure share the following aspects of the structure:

- The same set of segments
- The same arrangement of segments
- The same properties at the segment and structure levels

At the structure level, differences among structure instances include whether dynamic combination creation is allowed.

At the structure instance level, differences among segment instances include the following:

- Value set
- Default type and default value
- Tree code
- Whether the segment is any of the following:
  - Required
  - Displayed
  - Enabled for business intelligence
  - Optional or required as a query criterion

For example, you can use one group of value sets for the US and another for France.

The figure shows two structures instances for a part number structure. The structures differ in the number of segments and the segment separators used. The structure instances of a structure share all properties that are defined for the structure, but can vary in the properties defined at the structure instance or segment instance level, such as the value set assigned to the segment instances.
Query Required Segment Instances

You can designate a key flexfield segment instance as query required so that it is one of the selectively required attributes an end user can use in a key flexfield combination search. If you indicate in the Manage Key Flexfields UI page that a segment instance should be indexed, the column representing the segment must be added to the database index. Commonly, a database administrator (DBA) adds columns to the database index.

Following deployment, the combination picker of the key flexfield displays the query required attributes as selectively required. An end user must specify at least one of the query required attributes in the search criteria. This prevents non-selective searches that could cause performance issues.

For example, if you mark the cost center and account attributes as query required and ensure that the corresponding columns in the database are indexed, an end user can search for combinations by entering cost center or account or both as search criteria. No search is performed if an end user doesn’t enter at least one query required attribute as search criteria.

Tip

Index the Structure Instance Number column on your combinations table to improve run time performance.
**Dynamic Combinations**

If a key flexfield supports dynamic combination creation, you can choose to enable this feature by selecting *Dynamic Combination Creation Allowed*. This lets end users enter values at run time that produce new code combinations for the flexfield. If *Dynamic Combination Creation Allowed* isn’t enabled, new valid combinations can only be entered using the combinations table for the flexfield.

**Trees**

If a tree code has been defined for the value set assigned to the segment instance, and you assign the tree code to the segment instance, tree hierarchy search operations are available on the segment values.

For a segment instance to be based on a tree, the following must be true.

- Application development registered the key flexfield with a tree structure.
- A tree code for that tree structure exists.
- The tree code includes tree versions containing the values of the value set assigned to the segment instance.
- You assign the desired tree code directly to the segment instance.

Provided these conditions are satisfied, different segment instances that use the same value set can be assigned the same or different tree codes, meaning they use a different hierarchy definition over the same values.

**Cross Validation Rules: Explained**

You can control the creation of new key flexfield code combinations by defining cross validation rules. A cross-validation rule defines validation across segments and enforces whether a value of a particular segment can be combined with specific values of other segments to form a new combination.

The table compares segment validation to cross-segment validation:

<table>
<thead>
<tr>
<th>Type of validation</th>
<th>Type of control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment validation</td>
<td>Controls the values you can enter for a particular segment</td>
</tr>
<tr>
<td>Cross-segment validation</td>
<td>Controls the combinations of values that administrators and end users can create for key flexfields</td>
</tr>
</tbody>
</table>

**Note**

You can use cross-validation rules for any key flexfield that has cross-validation enabled. See the documentation for your key flexfield to determine if it supports cross validation.

Cross-validation rules prevent the creation of combinations with values that shouldn’t coexist in the same combination. For example, your company requires that all revenue accounts must have a specific department. Therefore, account...
combinations that have revenue account values, such as all values between 4000 and 5999, must have a corresponding department value other than 000, which indicates no department is specified. You can define cross validation rules that disallow creation of combinations with incompatible segments, such as 4100-000 or 5000-000.

Alternatively, suppose your accounting key flexfield has an Organization segment with two possible values, 01 and 02. You also have a Natural Account segment with many possible values, but company policy requires that Organization 01 uses the natural account values 001 to 499 and Organization 02 uses the natural account values 500 to 999. You can create cross-validation rules to ensure that users cannot create a general ledger account with combinations of values such as 02-342 or 01-750.

The following aspects are important to understanding cross validation rules:

- Rule Definitions
- Enforcement
- Timing

**Rule Definitions**

Cross validation rules consist of the following information:

<table>
<thead>
<tr>
<th>Rule Feature</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Uniquely identifies cross validation rules in a deployment.</td>
</tr>
<tr>
<td>Description</td>
<td>Helps administrators identify the purpose of the rule.</td>
</tr>
<tr>
<td>Error message</td>
<td>Explains why the attempted combination violates the rule.</td>
</tr>
<tr>
<td>Start Date, End Date</td>
<td>Indicates the period of time when the rule is in effect.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the rule is enforced.</td>
</tr>
<tr>
<td>Condition filter</td>
<td>Determines the conditions under which an enabled cross validation rule should be evaluated.</td>
</tr>
<tr>
<td>Validation filter</td>
<td>Determines the validation that the rule enforces when that condition is met.</td>
</tr>
</tbody>
</table>

When the event specified in the condition filter is applicable, the validation filter condition must be satisfied before the combination can be created. If the event specified in the condition filter isn’t applicable, then the combination is considered to pass the rule and the rule won’t be evaluated even if it is enabled.

**Note**

If you don’t specify any statement in the condition filter, then the condition is always true and the rule is always evaluated.

**Enforcement**

Cross-validation prevents creation of invalid combinations by administrators using maintenance pages and end users using dynamic insertion in foreign key pages.
Enabled rules are enforced when there is an attempt to create a new combination of segment values. Disabled rules are ignored. Deleting the rule has the same effect, but you can re-enable a disabled rule.

**Timing**

When users attempt to create a new combination, the key flexfield evaluates any cross-validation rules that are enabled and in effect.

**Warning**

Cross-validation rules have no effect on combinations that already exist. The flexfield treats any existing invalid combinations that pre-date the rule as valid.

If you want to prevent users from using previously existing combinations that are no longer valid according to your cross-validation rules, manually disable those combinations using the combinations page for that key flexfield.

When defining a cross-validation rule, specify a start and end date to limit the time when the rule is in effect. The rule is valid for the time including the From and To dates.

### Cross Validation Rules: Points to Consider

When you need key flexfield combinations of segment values validated across segments, you can optimize your cross validation rules to improve the experience of administrators and end users.

Consider the following when defining cross validation rules:

- Filters
- Rule Complexity
- Maintenance

**Filters**

A cross validation rule includes a condition filter and a validation filter.

The rule is evaluated using the following logic: If the condition filter is satisfied, then validate that the validation filter is satisfied.

1. The condition filter describes the event under which the rule will be evaluated. If the event specified in the condition filter isn’t applicable, then the rule won’t be evaluated even if it is enabled.

2. When the event specified in the condition filter is applicable, the validation filter condition must be satisfied before the combination can be created.

For example, if your organization has determined that a certain company value, Operations, cannot use a specific cost center, Marketing, you can define a cross validation rule to validate your combinations.

1. The rule evaluates the company condition filter.

2. When company is equal to Operations, the rule evaluates the cost center validation filter.
3. When cost center is equal to Marketing, the rule prevents a combination from being created.

4. The error message you defined for the rule displays to inform the user that the attempted combination violates the rule.

Note
This rule doesn’t affect the creation of combinations with Marketing cost center and company values other than Operations.

Rule Complexity
For optimal performance and ease of understanding, define several simple validation rules instead of using one complex rule. Simple validation rules let you provide a more specific error message and are easier to maintain over time. Avoid rules that control validation across more than two segments, where possible. While you can define cross validation rules that span two or more segments, keep in mind that it becomes more difficult to interpret cross validation error messages and correct invalid key flexfield combinations as your rules encompass more segments.

Maintenance
To maintain consistent validation, review existing key flexfields when you update your cross validation rules. Regardless of your current validation rules, Oracle Fusion Applications accept a key flexfield combination if the combination already exists and is enabled. Therefore, to ensure accurate validation, you must review your existing combinations and disable any combinations that don’t match the criteria of your new rules.

Tip
To keep this type of key flexfield maintenance to a minimum, decide upon your cross validation rules when you first set up your key flexfield structure. Define cross validation rules before creating combinations and before combinations are used in transactions.

If you want to prevent users from using previously existing combinations that are no longer valid according to your cross validation rules, disable those combinations using the combinations page.

Creating a Cross Validation Rule: Example
Create cross validation rules to prevent specific combinations of segment values in your account combinations, for example, preventing a particular cost center from being combined with a specific company value. Cross validation rules only affect the creation of new account combinations.

Scenario
Enter a new cross validation rule to prevent your InFusion America Inc. company value 01 from being combined with your marketing department value
300 in an account combination. Your company, InFusion America Inc. does not have a marketing department.

1. Navigate to the Manage Cross-Validation Rules task from within your implementation project, and then click the Go to Task icon.
2. Select your InFusion America chart of accounts.
3. Click the Create icon.
4. Specify a unique rule Name, IFAM01, and an optional Description, Do not combine Marketing Department, 300 with InFusion America, company 01.
5. Enter an optional effective From Date of today. Check Enabled.
6. Click the Change filter condition on the Condition Filter. Enter Company equal to 01. The cross validation rule evaluates if Company 01 was entered and if it was entered, then the validation process continues to evaluate the rule.

Note

If you do not specify any statement in the condition filter, then the rule is always evaluated.

7. Click on the Change filter condition on the Validation Filter. Enter Cost Center is not equal to 300. When the rule is evaluated, an account combination must contain a cost center other than 300 before it can be created.
8. Enter an Error Message: Cost Center 300 is not allowed with Company 01. The message displays in the relevant user interfaces and processes when an account combination cannot be created because it violates the rule.
9. Click Save and Close.

Enabling Key Flexfield Segments for Business Intelligence: Points to Consider

A key flexfield that is registered in the database as enabled for Oracle Business Intelligence (BI) includes a BI Enabled setting for each of its segment instances. When a segment instance is BI-enabled, it is available for use in Oracle Business Intelligence.

The following aspects are important in understanding BI-enabled key flexfield segments.

- Flattening business components to use BI-enabled segments in Oracle BI
- Equalizing segments to prevent duplication and complexity in the flattened component
- Mapping attributes of flattened business components to logical objects in Oracle BI
- Managing the labels that map segments to logical objects in Oracle BI
After you deploy a business intelligence-enabled flexfield, use the Import Oracle Fusion Data Extensions for Transactional Business Intelligence process to import the flexfield changes into the Oracle Business Intelligence repository. Users can make use of the newly-generated attributes in business intelligence applications. For additional information about logical objects and import, refer to the Oracle Transactional Business Intelligence Administrator’s Guide.

**Flattening**

When you deploy a business intelligence-enabled key flexfield, the deployment process generates an additional set of flattened business components for use in business intelligence. The flattened business components include attributes for business intelligence-enabled segment instances only.

If you assigned a label to a segment, the flattened components include a single attribute representing all segment instances with that label. If you didn’t assign a label, the flattened components include a discrete attribute for each BI-enabled segment instance in each structure.

**Mapping to Logical Objects in Business Intelligence**

You can simplify reporting by representing similar segments as a single logical object in Business Intelligence.

If you assign a label to segments that serve the same purpose in different structures, you can consolidate or equalize the segments into a single attribute. This prevents duplication and the extra workload and complexity that result from the flattening process. For example, an organization may have more than one definition of its key accounting flexfield to support different requirements for accounting reporting, or due to chart of accounts definitions from acquired organizations. A US Accounting Flexfield structure may have a segment called Subaccount to track project expenditures. The same type of information may be tracked in a UK accounting flexfield structure with a segment called Project. Equalize these two segments to create a single list of values for reporting.

Non-labeled segments aren’t equalized across context values, so the flattened components include a separate attribute for each segment for each structure.

---

**Note**

It may not be possible to equalize similarly labeled segments if they have incompatible data types or value set types.

Assign a label to a segment to map the corresponding attribute in the flattened components to a logical object in Oracle Business Intelligence. Using labels to map segments to BI logical objects minimizes the steps for importing the flexfield into Oracle Business Intelligence.

---

**Note**

Assigning a label to a segment serves to equalize the attribute across structures, as well as map the equalized attribute to business intelligence.

---

**Managing Labels**

You may assign a predefined label (if available) to segments or create new labels for assignment, as needed. Specify a code, name, and description to identify
each label. In the BI Object Name field, enter the name of the logical object in Oracle Business Intelligence to which the segment label should map during import. Specifying the BI logical object minimizes the steps for importing the flexfield into Oracle Business Intelligence and helps to equalize context-sensitive segments across structures.

If no labels are assigned to a BI-enabled segment, or the BI Object Name on the assigned label doesn’t exist in business intelligence, you must manually map the segment to the desired logical object when importing into Oracle Business Intelligence.

In addition, segments without labels cannot be equalized across structures. The flattened components include a separate attribute for each non-labeled segment in each structure.

---

**Note**

Segment labels serve other functions as well, as presented in Key Flexfields: Explained.

---

**Importing to Oracle Business Intelligence Repository**

After you deploy a business intelligence-enabled flexfield, import the flexfield changes into the Oracle Business Intelligence repository to make use of the newly flattened business components in business intelligence and then propagate the flexfield object changes. When you import the metadata into the Oracle Business Intelligence repository, you must do so as the FUSION_APPS_BI_APPID user.

---

**Note**

To import flexfield changes into the Oracle Business Intelligence repository in Oracle Cloud implementations, run the Import Oracle Fusion Data Extensions for Transactional Business Intelligence process. For additional information about import, refer to the Oracle Transactional Business Intelligence Administrator’s Guide.

---

**Tip**

When you import a flexfield into the Oracle Business Intelligence repository, you see both `<name>` and `<name>_c` attributes for each segment, along with some other optional attributes. The `<name>` attribute contains the value. The `<name>_c` attribute contains the code of the value set that the value comes from, and is used for linking to the value dimension. You must import both attributes.

---

**Key Flexfields: Example**

A key flexfield can capture expense account information.

**Scenario**

When entering details for each expense, the user specifies an account to which the expense is charged.
Entering Expense Accounts

A user interface for entering expenses gives the user the option of selecting an expense account that identifies the cost center and other details needed for processing the expense.

Analysis

The expense account field is a foreign key reference to a code combination (EXPENSE_LINES.EXPENSE_ACCOUNT = ACCOUNTS.CCID).

Code Combinations Table for Entering Accounts and Employees

The code combinations table supports entering account information, such as for expense accounts.

The figure shows the origin in the code combinations table of the account specified by the user. The code combination ID record stores the information of the key flexfield segments used to assemble the expense account based on the key flexfield configuration.

The combinations page, which is the maintenance page for the key flexfield, is for managing rows in the combinations table. In this example, managing the combinations means adding or editing account numbers that adhere to the key flexfield metadata rules.
The figure shows the code combination details for the example expense account reflected in the flexfield configuration and the code combinations table.

If dynamic combination creation isn't enabled, then when entering an expense line, the user can only select an account that already exists in the ACCOUNTS (combinations) table. If they require an account that doesn't exist, they must consult with the appropriate application administrator who can add the account to the combinations table.

If dynamic combination creation is enabled, then when entering an expense line, the user can either select a pre-existing account, or type in a new account that is created dynamically on the fly in the ACCOUNTS (combinations) table. Once the new combination is created, the same user can refer to it on the expense line.

When managing employee information, the user specifies the cost center that the employee belongs to. The cost center field corresponds to a single, labeled segment of the Account Key Flexfield and has metadata defined such as the allowable value set for that segment.

In this figure, instead of specifying a cost center ID reference to an account, only the Cost Center segment is used and the value is stored directly on the employee table.
Customizing the Springboard for Simplified Pages: Points to Consider

The springboard is the area on the simplified user interface home page, and above all simplified pages, that contains a set of functional area icon buttons. You can define which functional areas and pages within these areas are available to users.

Click **Settings** in the springboard, and then select the **Structure** tab. The Structure page displays all the functional areas and pages that are currently selected to appear on the springboard.

**Displaying or Hiding Functional Areas**

To display functional areas on the springboard, click **Add Item** and select the functional areas to include.

To hide a functional area, hover over that functional area and click **Hide Item**.

**Displaying or Hiding Pages**

To display a page in a functional area, hover over that functional area and click **Add Tab**, if available.

To hide a page, hover over that page and click **Hide Tab**.

**Changing the Order of Functional Areas and Pages**

Drag and drop functional areas and pages to set the order.

You can also use the **Organize** menu and select:

- **Site**: To reorder functional areas.
- **Any functional area**: To reorder pages within that area.
Renaming Functional Areas and Pages

Click the name of a functional area or page to modify its name.

Menu Customization

Menu Customization: Explained

You use the Manage Menu Customizations task to customize the navigator and home page menus. This task is available from the Setup and Maintenance work area, which is accessible from the Administration menu in the Oracle Fusion Applications global area. Select either Customize - Navigator or Customize - Homepage to proceed with the customization activity.

Note

To perform menu customization at run time, it is important that you have the required privileges.

You customize the menus at the site level and your changes affect all users (or all users of a tenant if in a multi-tenant environment).

Tip

If you are making minor changes, such as adding or editing one or two nodes, then you can hide the changes until you have completed your customizations. However, if you are making more than minor changes, such as rearranging several nodes, you might want to instead create a sandbox before customizing menus.

Navigator Menu Configuration

The navigator menu is the global menu that is accessible from the Oracle Fusion Applications global area. It allows users to navigate directly to the pages inside Oracle Fusion Applications as well as to outside web pages. The menu is composed of links (items) that are organized in a hierarchy of groups.

You can customize the navigator menu to address needs that are specific to your organization. For example, you might want to add specialized groupings for cross-functional teams or add links to web pages or external applications. You can add groups and links to the navigator menu, as well as hide and show them.

The Manage Menu Customizations task displays the menu groups as expandable nodes, with which you can traverse the menu hierarchy.

Note
Not all Oracle Fusion Applications pages appear in the navigator menu, because some pages are accessible from a work area or from other links in the global area such as the Home link.

The following table lists the Navigator menu customization tasks that you can perform at run time as well as the tasks that you cannot perform.

<table>
<thead>
<tr>
<th>Permitted Tasks</th>
<th>Restricted Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Add and delete custom groups.</td>
<td>• You cannot add menu items (links) as top-level nodes. You can add nodes to only the groups in the top level and subgroups.</td>
</tr>
<tr>
<td>• Edit any group.</td>
<td>• You cannot delete nodes that are delivered with the product. Instead, you can hide them.</td>
</tr>
<tr>
<td>• Add and delete custom items.</td>
<td>• You cannot move nodes. Instead, you must duplicate the node and hide the original node.</td>
</tr>
<tr>
<td>• Edit any item.</td>
<td></td>
</tr>
<tr>
<td>• Specify navigation for an item:</td>
<td></td>
</tr>
<tr>
<td>• Specify navigation to a UI Shell page in an Oracle Fusion application.</td>
<td></td>
</tr>
<tr>
<td>• Specify navigation to an external web page.</td>
<td></td>
</tr>
<tr>
<td>• Hide or show groups and items.</td>
<td></td>
</tr>
</tbody>
</table>

**Home Page Menu Configuration**

The home page menu is the set of tabs that are displayed in the Oracle Fusion Applications global area. The home page menu displays tabs for all the items in the menu for which the end user has access privileges. You can add tabs to the home page menu, as well as hide and show them.

The following table lists the Home page menu customization tasks that you can perform at run time as well as the tasks that you cannot perform.

<table>
<thead>
<tr>
<th>Permitted Tasks</th>
<th>Restricted Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Add and delete custom items.</td>
<td>• You cannot add menu items (links) as sub-nodes. All nodes are top-level nodes.</td>
</tr>
<tr>
<td>• Edit any item.</td>
<td>• You cannot delete nodes that are delivered with the product. Instead, you can hide them.</td>
</tr>
<tr>
<td>• Specify navigation to a UI Shell page in an Oracle Fusion application.</td>
<td>• You cannot move nodes. Instead, you must duplicate the node and hide the original node.</td>
</tr>
<tr>
<td>• Hide or show items.</td>
<td></td>
</tr>
</tbody>
</table>

**Adding Navigator Menu Group: Points to Consider**

You arrange the navigator menu by building a hierarchy of nested groups.

Use the View menu to expand or collapse a group of nodes. You can also right-click a node and access similar actions to facilitate tree navigation.
Adding Groups

To add a group, you can insert a group above or below a peer group or insert a child group. You edit a group by defining a label and specifying whether the group should be rendered. You typically hide the group until all changes have been completed.

Adding Menu Items: Points to Consider

The home page menu items are URL links to home pages in Oracle Fusion applications. The Navigator menu items can either be links to UI Shell pages or links to external applications and web sites.

In the menu hierarchy, the home page menu items are always top-level items. Whereas, you can add Navigator menu items to top-level groups and to their subgroups but you cannot add navigator menu items as top-level nodes.

Adding a Home Page Menu Item

To add a home page menu item, navigate to the place where you want the item to appear and insert it above or below the existing item. You can also duplicate an existing menu item and position it at the required location. You must provide a label for the menu and link the menu item to a UI Shell page.

Adding a Navigator Menu Item

To add a Navigator menu item, you navigate to the item's group and insert the item above or below another item.

You can also duplicate an existing item. You must provide a label for the menu and either link the menu item to a UI Shell page or link it to an external web site or application.

You can link a Navigator menu item to the following:

- A UI Shell page in an Oracle Fusion application.
- A dynamic URL of a page outside of Oracle Fusion Applications where the host, port, or context root might change.
- A Static URL of a page outside of Oracle Fusion Applications where the host, port, or context root does not change.

Linking to a UI Shell Page

If the new item points to a UI Shell page in an application, then you must provide the name of the web application and the view ID of the target page. The web application name and view ID can be obtained from an existing menu item that links to the same UI Shell page.

In a non-Cloud implementation, you also can obtain the web application name from the context root for the application, and you can obtain the view ID from
the id attribute for the page’s <view> tag in the product’s public_html/WEB-INF/adfc-config.xml file.

If you want secure access to the target UI Shell page from the menu item, then you must provide the name of the secured resource and the name of the policy store’s application stripe. When an end user clicks the link, the Oracle Fusion Applications checks the secured resource and the Lightweight Directory Access Protocol (LDAP) policy store to determine whether the user has the privilege to view the page.

If there is another menu item that points to the same page, then you can get the secured resource name and application stripe from that item. In a non-Cloud implementation, you also can obtain the application stripe from the jps.policystore.applicationid parameter in the application’s weblogic-application.xml file. Examples of application stripes are crm, fscm, and hcm.

For non-Cloud applications, you can determine the secured resource name by obtaining the name of the web page’s page definition file. By default, the page definition files are located in the view.PageDefs package in the Application Sources directory of the view project. If the corresponding JavaServer Faces (JSF) page is saved to a directory other than the default (public_html), or to a subdirectory of the default, then the page definition will also be saved to a package of the same name. An example of a secured resource name is oracle.apps.view.pageDefs.CaseList_Form_Attach_UIShellPagePageDef.

A UI Shell page might take parameters and display or act differently based on the parameters that are passed in. For example, if accessing a page from one group in the menu hierarchy, the parameter might be set to status=Open and if accessing the page from a different group, the parameter might be set to status=Closed. If the page takes parameters, you can use the Page Parameters List text box to provide a semicolon-delimited string of name-value pairs, such as org=m1;context=s1. You can use expression language (EL) to specify the parameters. If the EL evaluates to an Object, the toString value of that Object is passed as the value of the parameter.

**Linking to the Dynamic URL of an External Web Site or Application**

Linking a menu item to a dynamic URL is beneficial in cases where the host, port, or context root to which you point frequently changes. Instead of updating the link to each application, you can update the details of the web application in the topology registration, and that change affects all menu items that contain dynamic links pointing to that web application. For example, you would need a dynamic URL to link to a test version of an application and you will need to change the host and port when you move the application from a test environment to a production environment.

To link to a page outside of Oracle Fusion Applications where the host, port, or context root might change, you must first register the web application in the topology using the Register Enterprise Applications task.

While creating a new menu item on the Create Item Node dialog box, select the Dynamic URL option and provide the details of the web application as per the following example.

When the complete URL to be linked is: http://example:9011/myApp/faces/Page1,
• The name of the web application added to topology would be: myApp (the value that would eventually appear in the Web Application list) and the protocol host, port, and context root values of the URL would be: http://example:9011/myApp

• The value to be provided in the Destination for Web Application field would be: /faces/Page1

Once the menu item is linked to the dynamic URL, the target page appears in a new browser window or tab when you click the menu item.

Linking to a Static URL of an External Web Site or Application

This option is used when you link a menu item to a page outside Oracle Fusion Applications where the host, port, or context root remains constant. For example, you can use a static URL to link to http://www.oracle.com.

Hiding or Displaying Menu Nodes: Points to Consider

While you are creating or working with a menu group or a menu item, you might want to prevent end users from accessing the node. You can hide the menu group or menu item while you are working with it, and then show the node when you have completed the task.

Working with Nodes

The Manage Menu Customizations page shows all nodes. The Rendered check box is selected by default for all nodes that are added and are visible.

To hide a node, clear the Rendered check box. You can edit the node anytime to either display or hide it.

If you want a menu group or a menu item to appear only if certain conditions are met, you can use an expression language (EL) command to make the node to appear. For example, #{securityContext.userInRole('ADMIN')}.

A node that appears in italics either contains an EL command or the Rendered check box beside it was cleared, and therefore is hidden from end users.

Tip

For major changes that need to be tested and approved, you might want to use the sandbox manager instead of hiding and showing nodes.

Design Time Menu Customizations: Highlights

The menu customization feature provides several options to add, modify, and organize the Navigator and home page menus during design time. You must have developer rights to perform these customizations.
Note

Design time menu customizations are not applicable to Oracle Cloud implementations.

An overview of customizing the Navigator menu and home page is provided in the Oracle Fusion Applications Extensibility Guide.

Customizations

- Use Oracle JDeveloper to customize the Navigator and home page menus at design time.
  
  See: Customizing Menus

- Define translations for your customizations in the locales you support.
  
  See: Translating Menu Customizations

- Customize the page template to display the Navigator menu groups as separate menus, each of them displaying their list of menu items. Refer to the Oracle Fusion Applications Developer's Guide.
  
  See: Rendering the Navigator Menu as Dropdown Buttons

Troubleshooting Navigator Menu: Highlights

If the Navigator menu does not display customizations as expected, use the following troubleshooting tips to verify the changes.

Issues and Resolutions

- If an expected menu item does not appear in the Navigator menu, verify whether the menu item has been hidden from view.

- If a custom menu item was added and the browser does not display the page indicated by the URL, open the Manage Menu Customizations task and verify whether the web application name is the same as the context root for the application, and that the view ID is the id attribute for the page's <view> tag in the product's public_html/WEB-INF/adfc-config.xml file. The URL should not contain the .JSPX suffix.

- If you see a "webApp value not define" error message when you choose an item in the Navigator menu, verify whether the application is in the topology tables. Refer to the Oracle Fusion Applications Administrator's Guide.

  See: Viewing the Routing Topology of an Oracle Fusion Applications Instance, Product Family, or Product
Customizing Reports and Analytics

Analytics Customization

Customizing Analytics: Highlights

Edit and create custom analyses and dashboards to provide ad hoc reporting on your transactional data. Though the predefined analyses and dashboards help answer many of your business questions, you can create your own to meet custom requirements.

Each analysis and dashboard contains a set of attributes. For example, an analysis to provide information about customers would have attributes such as customer name, customer address, contact information, and so on. When users view the analysis, the latest customer information is queried and displayed. The Oracle Business Intelligence (BI) repository contains the data model metadata that support the available attributes.

This table shows what tools you can use to customize which BI object: analyses, dashboards, or the BI repository.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
<th>Business Intelligence Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>BI Composer</td>
<td>A wizard for creating and editing analyses, available within Oracle Fusion Applications.</td>
<td>Analyses</td>
</tr>
<tr>
<td>Oracle Business Intelligence</td>
<td>An application that provides all needed features to customize analyses and dashboards. You can open this application from Oracle Fusion Applications.</td>
<td>Analyses and dashboards</td>
</tr>
<tr>
<td>Manage Descriptive Flexfields</td>
<td>Task lists in the Setup and Maintenance work area that let you capture custom attributes.</td>
<td>Oracle BI repository</td>
</tr>
<tr>
<td>Manage Key Flexfields</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage Extensible Flexfields</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oracle BI Administration Tool (not available in Oracle Cloud implementations)</td>
<td>A feature in Oracle Business Intelligence that lets you customize the Oracle BI repository.</td>
<td>Oracle BI repository</td>
</tr>
</tbody>
</table>
Oracle BI Repository Customization

- Designate key, descriptive, or extensible flexfields as BI-enabled and import the flexfields into the Oracle BI repository. In the Oracle Fusion Transactional Business Intelligence Administrator's Guide, refer to: Designating Flexfields as BI-Enabled; Importing Changes to Flexfields Automatically.

- Update the Oracle BI repository, for example to define that data for certain predefined attributes should come from a specified custom table. Refer to the Oracle Fusion Applications Extensibility Guide for Developers. See: Modifying the Oracle Business Intelligence Repository for Customized Analytics

Catalog Security and the Types of Analytics Customizations Allowed: Points to Consider

Before you customize analyses or dashboards, ensure that you have proper permissions for editing and creating objects in the Oracle Business Intelligence (BI) Presentation Catalog.

Custom Folder

The Oracle BI Presentation catalog includes a Custom folder under Shared Folders. The Custom folder contains a subfolder for each product family, similar to the product family subfolder structure under Shared Folders.

When you edit a predefined analysis or dashboard, you should save the updated version in the Custom folder and leave the original as is. Keeping all custom objects in the Custom folder has the following benefits.

- While patches can update predefined analyses and dashboards outside the Custom folder, you ensure that customized versions of those objects are not affected. You might lose customizations saved outside the Custom folder during upgrades.

- Customized objects are easily located and identified.

- Default security facilitates the editing of objects in the Custom folder without compromising security on the original objects.

Note

When you copy an object into the Custom folder, the copied object inherits the permission settings of the Custom folder. You must manually reset the permissions on the analysis or dashboard and product subfolder structure that you create under the Custom folder to reestablish the security settings of the original object.

BI Author

Users with the BI Author role have full control permissions to the Custom folder located under Shared Folders in the catalog.

This table summarizes the actions allowed for users with the BI Author role.
**Customizing Reports and Analytics**

<table>
<thead>
<tr>
<th>Customization</th>
<th>Action Allowed for BI Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit analysis</td>
<td>Can edit analysis but cannot save to original folder. Must save the edited version to the Custom folder or My Folders.</td>
</tr>
<tr>
<td>Copy analysis</td>
<td>Can copy analysis and save to the Custom folder or My Folders. Cannot save the copy to original folder.</td>
</tr>
<tr>
<td>Create new analysis</td>
<td>Can create a new analysis and save to the Custom folder or My Folders.</td>
</tr>
<tr>
<td>Edit dashboard</td>
<td>Not allowed for predefined dashboards; must first make a copy of predefined dashboards.</td>
</tr>
<tr>
<td>Copy dashboard</td>
<td>Can copy dashboard and save to the Custom folder or My Folders. In the Custom folder the copy can be edited. Cannot save the copy to original folder.</td>
</tr>
<tr>
<td>Create new dashboard</td>
<td>Not allowed.</td>
</tr>
</tbody>
</table>

**BI Platform Administrator**

By default, the BI Platform Administrator role (not available in Oracle Cloud implementations) is granted full control permissions to the Custom folder and product family subfolders under Shared Folders. Users with this role can, for example:

- Edit and delete the objects that are copied to or created in the Custom folder.
- Set permissions on objects in the Custom folder.
- Create new dashboards.
- Directly edit predefined objects in the catalog.

**Important**

Edit predefined analyses or dashboards directly only when necessary to maintain the proper functioning of applications or processes that reference the original objects (such as embedded dashboards or targets of navigation actions). Otherwise, you should copy the predefined objects that you wish to customize into the corresponding product family subfolder under the Custom folder.

**Customizing Analytics: Examples**

There are many ways to customize predefined analyses and dashboards, as well as many options available to you to create custom analyses and dashboards to suit your needs. The following examples illustrate only a few possible types of customizations.

**Create an Analysis**

Your team needs to print a simple list of all your customers, sorted by customer ID. From the Reports and Analytics pane, you click **Create** to create a new analysis. You include only the customer name, ID, and address in the analysis, and add a sort on the customer ID attribute, or column. When users view this analysis, they can sort the results as needed and create a printable PDF file.
Create a View
A predefined analysis shows the number of hires per year in a bar graph. You want the results in a table format view as well. From the Reports and Analytics pane, you click Edit for the analysis and then Select Views. Add a table view and specify for the analysis layout that the table should be displayed below the graph. Save this customized version of a predefined analysis in the Custom folder of the Oracle Business Intelligence Presentation Catalog.

Create a View Selector
For the customized analysis that you added a table to, you later decide that you want to be able to toggle between the table and the graph and not display both at the same time. In the Reports and Analytics pane, find your custom analysis, and click More. Edit the analysis. Select View Selector as a new view to add, and include the table and graph views in the selector.

Edit a Dashboard Prompt
A dashboard can contain many objects including analyses and prompts, which are parameters that determine what is displayed. A predefined dashboard on invoices provides prompts such as Begin Invoice Date and Invoice Type. You want to replace Begin Invoice Date with a date range prompt. First you copy the dashboard and its components (analysis and prompt) into the Custom folder. In the Reports and Analytics pane, you click the Browse Catalog icon and then find the copied prompt. You edit that prompt and also the copied analysis so that both components work together.

Copying Dashboards: Worked Example
This example demonstrates how to copy a predefined dashboard so that you can customize the copy and not the original. The following table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Which predefined dashboard are you copying?</td>
<td>Payables Invoice Audit Listing</td>
</tr>
<tr>
<td>Is the copied version for yourself only or to be shared among multiple users?</td>
<td>Shared</td>
</tr>
</tbody>
</table>

Make a copy of the Payables Invoice Audit Listing dashboard and all its contents, including its analysis and prompt, and place them in the appropriate folders under Shared Folders - Custom. Edit the copied dashboard so that it contains the copied analysis and prompt.

Copying the Dashboard
1. Navigate to the Reports and Analytics work area (Navigator - Tools - Reports and Analytics).
2. Go to Shared Folders - Financials - Payables - Invoices - Payables Invoice Audit Listing - Invoice Audit Listing, and click the More link.
3. In the Folders pane, click the selected Payables Invoice Audit Listing dashboard.
4. Click the **Copy** icon button in the toolbar.

5. In the Folders pane, go to **Shared Folders - Custom - Financials**.

6. In the toolbar, click the **New** icon button and select **Folder**.

7. Enter **Payables** in the **Name** field and click the **OK** button.

8. Click the **Paste** icon button in the toolbar.

**Copying the Prompt and Analysis**

1. With the Payables folder still open, in the toolbar of the Folders pane, click the **New** icon button and select **Folder**.

2. Enter **Prompts** in the **Name** field and click the **OK** button.

3. Create another folder with the name **Report Components**.

4. In the Folders pane, go to **Shared Folders - Financials - Payables - Invoices**, and select the Prompts folder.

5. For the Payables Invoice Audit Listing dashboard prompt, click the **More** link and select **Copy**.

6. In the Folders pane, select **Shared Folders - Custom - Payables - Prompts**, and click the **Paste** icon button in the toolbar.

7. In the Folders pane, go to **Shared Folders - Financials - Payables - Invoices**, and select the Report Components folder.

8. For the Payables Invoice Audit Listing analysis, click the **More** link and select **Copy**.

9. In the Folders pane, select **Shared Folders - Custom - Payables - Report Components**, and click the **Paste** icon button in the toolbar.

**Editing the Copied Dashboard**

1. In the Folders pane, go to **Shared Folders - Custom - Payables - Payables Invoice Audit Listing**.

2. Click the **Edit** link for the Invoice Audit Listing dashboard.

3. Click the **Delete** icon button for the Payables Invoice Audit Listing dashboard prompt within the Search region.

4. In the Catalog pane, select **Shared Folders - Custom - Payables - Prompts - Payables Invoice Audit Listing** and drag it into the Search region.

5. Delete the Payables Invoice Audit Listing compound view within the Section 1 region.

6. In the Catalog pane, select **Folders - Custom - Payables - Report Components - Payables Invoice Audit Listing** and drag it into the Section 1 region.

7. Click the **Save** icon button.
FAQs for Analytics Customization

What happens to customized analyses and dashboards when a patch is applied?

All custom analyses and dashboards in Shared Folders - Custom are preserved during the patch process.

Changes to predefined analyses and dashboards are preserved when the patch does not include a new version of that object. If the patch does include a new version of a predefined object that was customized, then the patch process detects and logs this conflict, and patching will stop. The administrator must resolve any conflicts manually using Catalog Manager and then rerun the patch.

Reports Customization

Customizing Reports: Explained

Reports extract data from your applications and present it in the formats required for your enterprise. The output is optimized for high-fidelity printing. Reports provide the information you need for internal operations and statutory compliance; reports also provide the business documents for communicating with your customers.

Many product-specific reports are provided with Oracle Fusion Applications; for example, the invoice register, the pick slip report, the payroll summary, the journals report, and the customer credit memo. To meet the specific needs of your enterprise, you may need to customize the reports provided or create new reports to capture and present different data.

Report Components

Reports are built with Oracle Business Intelligence (BI) Publisher. A report in Oracle BI Publisher consists of components which can be customized, as described in this table:

<table>
<thead>
<tr>
<th>Report Component</th>
<th>Description</th>
<th>Tool for Customizing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data model</td>
<td>Defines the data source, data structure, and parameters for the report. A data model can be used by multiple reports. Each report has one data model.</td>
<td>Data model editor in Oracle BI Publisher</td>
</tr>
</tbody>
</table>
**Layout**

Defines the presentation, formatting, and visualizations of the data. A report may have multiple layouts of the data model, and there are different types of layout templates, for example Excel and RTF.

Style templates and subtemplates can also be created and applied to layout templates.

- **RTF**: Microsoft Word (Template Builder for Word)
- **XPT (BI Publisher Template)**: Layout editor in BI Publisher
- **PDF**: Adobe Acrobat Professional
- **Excel**: Microsoft Excel (Template Builder for Excel)
- **eText**: Microsoft Word

**Properties**

Specifies run time and formatting options.

- **Report editor in BI Publisher**

### What You Can Customize

This table shows some common report customization scenarios.

<table>
<thead>
<tr>
<th>Customization Use Case</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit the layout of a report provided with an application.</td>
<td>Add your company logo to the Receivables Credit Memo report.</td>
</tr>
<tr>
<td>Add a new layout to a report provided with an application.</td>
<td>Design a new form letter users can select when they run the Receivables Credit Memo.</td>
</tr>
<tr>
<td>Edit a predefined data model.</td>
<td>Add two fields to the data model used by the Receivables Credit Memo report so that those new fields can be incorporated into a custom layout of the report.</td>
</tr>
<tr>
<td>Create a new report based on a new data model.</td>
<td>Create a new data model based on sales order transaction data from an external system, and create new sales order reports using the custom data model.</td>
</tr>
</tbody>
</table>

### Additional Report Customization Tasks

Depending on how a report is implemented in Oracle Fusion Applications and the type of customization you make, you may also have to perform additional tasks to implement your custom report in the system.

- **Scheduled processes**: When you create a new report and you want users to be able to run this report as a scheduled process, you must create an Oracle Enterprise Scheduler job definition for the report.

- **Translation**: If you create a custom layout and you require translations of the layout, you must also provide the translations. Oracle BI Publisher provides a tool for extracting the translation file for some layout types. The translation file can be translated into the required languages then uploaded to the report.

- **Security**: You must ensure that the proper security settings are applied to the report and data model to enable the intended report consumers to run the report.
Accessing Report Components to Customize: Points to Consider

To create or edit reports, you need to access the Oracle Business Intelligence (BI) Presentation Catalog. In the catalog, objects of type Report represent the report definition, including report properties and layouts. Data models are separate objects in the catalog, usually stored in subfolders called Data Models. Style templates and subtemplates are also stored in the catalog.

Accessing the Catalog

To access the catalog, you have the following options:

- In the Reports and Analytics pane, click **Browse Catalog** to open the Oracle BI Presentation Catalog, and find the report in the catalog.
- In the Reports and Analytics pane, find the report and select **More** to go to the report directly in the catalog. The data model associated with the report should be in the Data Models subfolder within the same folder as the report.
- Sign in to the business intelligence application directly (for example: [http://host:port/analytics/saw.dll](http://host:port/analytics/saw.dll)) to open the catalog.
- Sign in to the BI Publisher server directly (for example: [http://hostname.com:7001/xmlpserver](http://hostname.com:7001/xmlpserver)) to open the catalog.
- Alternatively, once you are in the catalog using another method, for example through the Reports and Analytics pane, change the final node of the URL ([http://host:port/analytics/saw.dll](http://host:port/analytics/saw.dll)) to xmlpserver. So the URL you use would be: [http://host:port/xmlpserver](http://host:port/xmlpserver).

**Important**

Save all custom report components in **Shared Folders - Custom** within the catalog. Objects outside the Custom folder are susceptible to patches. Therefore, Oracle recommends that you do not directly edit predefined report components. If you change a predefined object and a subsequent patch includes a new version of it, then the patch overwrites any customizations.

For predefined report objects only (not data models, style templates, or subtemplates), use the Customize option if possible.

Predefined Reports

The Customize option for predefined report objects is only available through direct access to the BI Publisher server using the /xmlpserver URL. When you find your report in the Oracle BI Presentation Catalog, select the Customize option from the **More** menu.

The Customize option automatically creates a custom copy of a predefined report and stores it in the **Shared Folders - Custom** folder within the catalog. The new report is linked to the original, so that when users open or schedule the original, they are actually using the custom version.

If you do not have the Customize option or do not want the original version linked to the new report, then make a copy of the predefined report and save it in the Custom folder.
Predefined Data Models

Don’t customize predefined data models. Instead, copy the data model into the Custom folder and edit the copy.

Using the Customize Option for Predefined Reports: Points to Consider

When you select the Customize option for a report, Oracle Business Intelligence (BI) Publisher creates a copy of the report in Shared Folders - Custom within the Oracle BI Presentation Catalog. The report, folder structure, and original report permissions are copied. For example, if the predefined report is in the Sales folder, the custom report is automatically created and saved in the Sales folder within Custom.

This custom copy is linked internally to the original report. You can customize the custom copy of the report, leaving the original report intact. When users initiate a request to run the original report, whether as a scheduled process, from the BI catalog, or through an application process, BI Publisher detects the customized version and runs your custom version instead.

Benefits of the Customize Option

Aside from the convenience of automatically copying a report to the Custom folder, the Customize option:

- Removes the requirement to edit calling processes or applications to execute the custom report. BI Publisher automatically sends all requests to run the original report to the custom copy. Therefore Oracle Enterprise Scheduler job definitions that point to the original report or applications that call the original report will automatically run your custom report with no additional configuration required.
- Automatically copies the security settings of the original report.
- Removes the risk of patches overwriting your customizations of predefined reports. When a patch is applied that updates the original report, the custom report is not updated in any way.

Note

The custom report still references the original data model. The data model is not copied. A patch that updates the data model may impact the running of your custom report if the data structure is changed.

Accessing the Customize Option

To access the Customize option:

1. Sign in to the BI Publisher server (for example, http://hostname.com:7001/xmlpserver).
2. Go to the predefined report in the catalog.
3. Select Customize from the More menu for the report.
4. The copied report in the Custom folder opens, so proceed to customize this report.

To apply further edits to the report in the Custom folder, perform one of the following:

- Select the **Customize** or **Edit** option for the original report to open the existing customized report.
- Go to the custom report in the Custom folder and select **Edit**.

**Links Between Original and Custom Reports: Points to Consider**

The Customize option for predefined reports creates a custom copy of the report that is linked to the original. Consider the following points when you work with both the original and custom versions.

**Maintaining the Link Between Reports**

The link between the predefined report and the custom report is based on the name of the custom report in the same folder under Custom.

- If you manually create the identical folder path to a report under the Custom folder and create a report with the same name, Oracle Business Intelligence Publisher treats the identically named report as a customized report and will run this report instead of the original report, as if you had used the Customize option to create it.

- The link to the original report is broken if you rename the custom report. You can edit the custom report so that it uses a different data model. However, if the original data model is updated later, for example due to a patch, then your custom report does not benefit from the change.

**Tasks Performed on Original Reports**

When a custom version of a report exists, performing tasks on the original report have the results shown in this table.

<table>
<thead>
<tr>
<th>Tasks Performed on the Original Report</th>
<th>Behavior When a Custom Report Is Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>Opens the custom report.</td>
</tr>
<tr>
<td>Schedule</td>
<td>Creates a report job for the custom report.</td>
</tr>
<tr>
<td>Edit</td>
<td>Edits the custom report.</td>
</tr>
<tr>
<td>Delete</td>
<td>Deletes the original report only. If you delete the custom report, the original report is not deleted.</td>
</tr>
<tr>
<td>Copy</td>
<td>Copies the original report.</td>
</tr>
<tr>
<td>Cut and Paste</td>
<td>Cuts and pastes the original report.</td>
</tr>
<tr>
<td>Rename</td>
<td>Renames the original report. The custom report name is not changed.</td>
</tr>
<tr>
<td>Download</td>
<td>Downloads the custom report.</td>
</tr>
<tr>
<td>Customize</td>
<td>Edits the custom report.</td>
</tr>
<tr>
<td>History</td>
<td>Opens the job history of the custom report.</td>
</tr>
</tbody>
</table>
Roles You Need to Customize Reports: Explained

To view and perform actions on report components in the Oracle Business Intelligence (BI) Presentation catalog, your role must be granted the appropriate combination of functional permissions to enable the actions and access permissions to the object in the catalog. The functional permissions are granted in the definition of the role; the access permissions are granted in the catalog.

Customizing Reports, Style Templates, and Subtemplates

To edit, create, or copy reports, style templates, and subtemplates, you must have the BI Author role.

The following security settings are configured by default in the BI Presentation catalog:

- All roles are granted the Traverse permission on all reports.
- The BI Author role is granted Full Control on the Custom folder and product family subfolders, and along with the Read permission, you can:
  - Make a copy of predefined objects and save the copy in the Custom folder.
  - Edit and delete objects copied to the Custom folder for customization.
  - Create new reports, style templates, and subtemplates in the Custom folder.

Using the Customize Option

To access to the Customize option for predefined reports, you must have:

- The BI Author role (or a role that includes the permission `oracle.bi.publisher.developReport`).
- These permissions on the predefined report in the catalog:
  - Read
  - Run Publisher Report
- These permissions on the Custom folder in the catalog:
  - Read
  - Write

To view a report in the Custom folder that is linked to the predefined report, you must have:

- The BI Consumer role
- These permissions on the predefined report in the catalog:
  - Read
  - Run Publisher Report
- These permissions on the Custom folder in the catalog:
  - Read
• Run Publisher Report
• These permissions on the data model of the predefined report:
  • Read
  • Traverse

**Customizing Data Models**

To create and edit data models, you must be granted the Application Developer role. This role gives you the BI Publisher Data Model Developer role, which allows you to customize data models.

**Important**

• Because the BI Publisher Data Model Developer enables the ability to write and execute SQL, it carries substantial access privileges and should, therefore, only be assigned when needed to the right users.
• This access should be granted only in test environments, not production environments.

**Report Layouts**

**Customizing Report Layouts: Overview**

The layout defines the presentation of the report data. All predefined reports include at least one predefined layout template file that defines the presentation components (such as tables and labeled fields) and maps the elements from the data model to these components. The layout also defines font sizes, styles, borders, shading, and can also include images, such as a company logo. To customize a layout, you edit the layout template.

**Layout Template Types**

Several types of templates support different report layout requirements.

• Most of the predefined templates are rich text format (RTF) templates created using Microsoft Word.
• Some predefined templates are Oracle Business Intelligence (BI) Publisher layout templates created using Oracle BI Publisher's layout editor. These are for interactive and more visually appealing layouts.
• A third type is the eText template, which is used specifically for electronic data interchange (EDI) and electronic funds transfer (EFT).

Oracle BI Publisher templates can also be created using Adobe PDF, Microsoft Excel, Adobe Flash, and XSL-FO.

**Style Templates and Subtemplates**

You can create style templates and subtemplates to apply to your custom layout templates. For more information on creating style templates and subtemplates, see the Oracle Fusion Middleware Report Designer's Guide for Oracle Business Intelligence Publisher.
Customizing Report Layouts: Examples

If the output of predefined reports doesn’t completely meet your requirements, you can customize the layout templates. The following examples illustrate reasons to customize layout.

Style Changes Only, No Changes to Data Mapping
This is the simplest type of customization. Examples are removing the predefined logo from the report and inserting your own or simply modifying colors and font styles. For these changes you can download the predefined template and edit it.

Because there are no changes to the data mapping, style changes do not require sample data from your report; however, having sample data available will enable testing of the template from your desktop.

Changes to Mapped Data Elements Within the Existing Layout
An example of this type of customization is adding or removing a table column or data field from the report layout. For these changes you must have sample data to load to the layout editing tool.

You can download the predefined template, load your sample data, insert the required elements, preview your template, then upload your customized template back to the report definition.

New Presentation of the Data
For example, you want to present a different set of fields in a completely different way. Instead of editing an existing layout, it’s more efficient to just create one. To create a new layout, start by opening the layout editing tool and loading the sample data to begin designing your custom layout.

Customizing Report Layouts: Procedures

Editing or creating a layout template using Microsoft Word or the layout editor involves making the actual changes to the template. However, that task is just one part of the entire process for customizing layouts.

1. Create a custom copy of the original report.
2. Review report settings for online viewing.
3. Generate sample data from the report.
4. Edit or create the layout.
5. Upload the template file to the report definition.
6. Add translations.
7. Configure the layout settings.

1. Create a Custom Copy of the Original Report

Make a copy of the original report using the Customize option, or by manually duplicating a report and placing the copy within Shared Folders - Custom in the Oracle Business Intelligence (BI) Presentation Catalog.
2. Review Report Settings for Online Viewing

Some reports are configured to view only through an external application or through Oracle Enterprise Scheduler. To view your report online while you are customizing it, ensure that the following properties are set as shown in this section. When finished customizing your report, ensure that you reset these properties appropriately for production.

- **Report Properties Settings**
  a. Navigate to your report copy in the catalog and click **Edit**.
  b. In the report editor, click the **Properties** link at the top of the page.
  c. In the Report Properties dialog box, select **Run Report Online** and deselect **Report is Controlled by External Application**.

- **Layout Setting**
  a. In the report editor, click the **View a list** link.
  b. Ensure that the **View Online** check box is selected.

3. Generate Sample Data from the Report

Depending on the type of layout customization you are doing, sample data can be mandatory or helpful. Sample data enables the mapping of data fields to layout components in the report.

You can generate sample data from the:
- **Report data model**
- **Report viewer**
- **Scheduler**

4. Edit or Create the Layout

To design an RTF layout using Template Builder for Word, or design a layout using the Oracle BI Publisher layout editor.

To design one of the other supported layout types, see the corresponding chapter in the Oracle Fusion Middleware Report Designer’s Guide for Oracle Business Intelligence Publisher:
- **PDF templates**
- **Excel templates**
- **eText templates**

5. Upload the Template File to the Report Definition

If you created a layout using the layout editor, the layout is automatically saved to the report definition and you can skip this step. For all other layout types, upload the template file to the report definition.

1. Navigate to your custom report under the Custom folder in the catalog and click **Edit**.
2. On the report definition page, click **View a list**.
3. On the table that lists the layouts, click **Create**.
4. Under Upload or Generate Layout, click **Upload**.
5. In the Upload Template dialog box:
   a. Enter a layout name.
   b. Browse for and select the template file.
   c. Select the template file type.
   d. Select the locale, which is not editable once the template file is saved to
      the report definition.
   e. Click Upload.

6. Save the report definition.

6. Add Translations

Template translation enables the extraction of translatable strings from a single
RTF-based template or a single BI Publisher layout template. Use this option if
you want the report to have output in different languages based on the preferred
language of the user; for example, you need to generate invoices for both
German and French customers.

For information on translating individual templates, see the Oracle Fusion
Middleware Report Designer’s Guide for Oracle Business Intelligence Publisher.

7. Configure the Layout Settings

To edit the layout settings, click View a list in the report editor.

**Note**

From here, you can also delete any layout templates by selecting the
corresponding row and clicking the Delete icon button.

This table describes the properties to set for your custom layout.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Output Formats** | Output formats are the file formats available for the
generated report, such as PDF, HTML, RTF, Excel. Depending on the requirements of a report, you may want to limit the output formats available to users. The output formats available vary according to the template file type. |
| **Default Format** | When multiple output formats are available for the report, the default output format is generated by default when the report is run in the report viewer. |
| **Default Layout** | When multiple layouts are available for the report, the default layout is presented first in the report viewer. Select this check box for your custom layout when you want it displayed first. One layout must be selected as the default layout. |
| **Apply Style Template** | If a style template is assigned to this report, use this field to apply the style template to the layout. |
| **Active** | Active layouts are displayed to report consumers. |

**Tip**

To hide the original layout from users, inactivate it.
View Online | Layouts that can be viewed online are available to report consumers from the report viewer. If this check box is not selected, the layout is available only for scheduled jobs.

Generating Sample Report Data: Procedures

The report layout tools require sample data to enable the mapping of data fields to layout components in the report. You can generate sample data from the:

- Report data model
- Report viewer
- Scheduler

Data Model

To generate sample data from the data model:

1. Navigate to the data model in the catalog and click Edit.

Tip
If you are not sure which data model is the source for a particular report, click the report Edit link. The data model is displayed in the upper left corner of the report editor.

2. In the data model editor, click View Data.
3. Enter values for any required parameters, select the number of rows to return, and click View.
4. To save the sample data to the data model, click Save As Sample Data.
   If you are designing an RTF template, click Export to save the file to a local client for use with the Template Builder for Word.
5. Save the data model.

Report Viewer

To save sample data from the report viewer:

Note
This procedure requires that the report is enabled for online viewing.

1. Navigate to the report catalog.
2. Click Open to run the report in the report viewer with the default parameters.
3. On the Actions menu, click Export, then click Data.
4. Save the data file.

Scheduler

To save sample data from the scheduler:
Note
This procedure requires that the report is enabled for scheduling through Oracle Business Intelligence (BI) Publisher.

1. Navigate to the report in the catalog.
2. Click Schedule to open the BI Publisher scheduler.
3. On the General tab, enter values for any report parameters.
4. On the Output tab, ensure that Save Data for Republishing is selected.
5. Click Submit.
7. On the global header, click Open, then click Report Job History.
8. Locate your report job in the report Job Histories table and click the job name. On the details page, under Output and Delivery, click the XML Data Download icon button.

RTF Templates

Customizing RTF Templates: Procedures

Most predefined layout templates are RTF templates. An RTF template is a rich text format file that contains the layout instructions to use when generating the report. RTF templates are created using Microsoft Word. The add-in to Microsoft Word, Template Builder for Word, facilitates the coding of layout instructions.

Tip
If you are designing a new layout for the report, consider using the BI Publisher layout editor. The layout editor is an online layout editing tool launched from the report editor.

Using Template Builder for Word

To customize an RTF template:
1. Ensure that you have a supported version of Microsoft Word.
2. From the Oracle Business Intelligence home page, under the Get Started pane, click Download BI Desktop Tools. Select Oracle BI for MS Office and install the client.
3. If you are modifying a predefined layout, navigate to the copy of the report under Custom in the catalog and click Edit. In the report editor, click the Edit link of the layout to download the RTF file to your local client.
   If you are creating a new layout, skip this step.
4. Open the downloaded RTF template file in Microsoft Word; or, if you are creating a new template, open Microsoft Word.
5. Load the sample data to the Template Builder for Word add-in.
6. Edit or create the layout following the guidelines in the Template Builder for Word help.
7. Save the file as Rich Text Format (RTF).

## Changing the Branding Logo in a Predefined Template: Worked Example

The Payables Invoice Register report layout includes a standard logo in the report header. You want to replace the Oracle logo with your own logo. The following table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>What version of Microsoft Word am I using?</td>
<td>Microsoft Word 2007</td>
</tr>
<tr>
<td>Do I want to preview the customized template?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Changing the Logo

To change the logo in the report output:

1. Download sample data from the Payables Invoice Register data model (InvoiceRegisterDM).
2. Download the RTF template file and open it in Microsoft Word.
3. In the BI Publisher tab, click **Sample XML** in the Load Data group, and select sample data that was saved from the data model.
4. In the template, delete the Oracle logo.
5. On the Insert tab in the Illustrations group, click **Picture**.
6. Select your company logo image file and insert it into the Word document.
7. Resize the image as necessary.

---

**Tip**
If the template file includes section breaks, you must insert the new logo for each section header.

8. In the Oracle BI Publisher tab, in the Preview group, click PDF. Template Builder for Word applies the sample data you loaded and generates a PDF output document.

Creating an RTF Template Using an Existing Data Model: Worked Example

This example demonstrates creating an RTF template using an existing data model. You want to create a layout that displays invoice summary for each currency.

The following table summarizes the key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>What version of Microsoft Word am I using?</td>
<td>Microsoft word 2007</td>
</tr>
<tr>
<td>Which data model am I using?</td>
<td>Payables Invoice Register</td>
</tr>
<tr>
<td>Which date format do I want displayed in the report output?</td>
<td>M/d/yyyy</td>
</tr>
<tr>
<td>What number format do I want displayed in the report output?</td>
<td>999.00</td>
</tr>
<tr>
<td>Do I want to preview the layout?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Creating the Layout and Inserting Fields

To create the layout of the template and insert data fields:

1. Download to your local client sample data from the Payables Invoice Register data model (InvoiceRegisterDM).
2. Open Microsoft Word and create a blank new document.
3. In the BI Publisher tab, click Sample XML in the Load Data group, and select sample data that was saved from the data model.
4. Using Microsoft Word functionality, insert the page header. Type the text for the header, and insert the field for the Business Unit.
5. In this example, the report will repeat the table of invoices for each occurrence of the currency code. To create this behavior, insert a repeating group based on the element C_CURRENCY_CODE:
   a. On the BI Publisher tab, in the Insert group, click Repeating Group.
b. In the BI Publisher Properties dialog box, select the G_CURRENCY in the For Each field.

c. To display the field value, type Currency: after the for-each tag in the template.

d. To insert the element from the data, on the BI Publisher tab, in the Insert group, click Field. Select the C_Currency_Code element in the Field dialog box.

6. To insert the table, on the BI Publisher tab in the Insert group, click Table Wizard. Make the following selections as shown in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Format</td>
<td>Table</td>
</tr>
<tr>
<td>Data Set</td>
<td>VENDOR</td>
</tr>
<tr>
<td>Which fields do you want to show in your report?</td>
<td>• C Vendor Name</td>
</tr>
<tr>
<td></td>
<td>• C Invoice Num</td>
</tr>
<tr>
<td></td>
<td>• C Invoice Date</td>
</tr>
<tr>
<td></td>
<td>• C Invoice Amount</td>
</tr>
<tr>
<td></td>
<td>• C Amount Rem</td>
</tr>
</tbody>
</table>

Note
The inserted table will display with the column names from the data, and no formatting is applied to number and date fields.

Edit the column names by simply editing the text in the column header row.

Apply Formatting to the Layout

To apply date, number, and other formatting to the layout:
1. Apply formatting in the date field.
a. Right-click the C_INVOICE_DATE field in the table and select **BI Publisher - Properties**.

b. In the BI Publisher Properties dialog box, make the following selections as shown in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text to display</td>
<td>8/7/2011</td>
</tr>
<tr>
<td>Type</td>
<td>Date</td>
</tr>
<tr>
<td>Format</td>
<td>M/d/yyyy</td>
</tr>
</tbody>
</table>

2. Apply formatting to the number field.

a. Right-click the C_INVOICE_AMOUNT field in the table and select **BI Publisher - Properties**.

b. In the BI Publisher Properties dialog box, make the following selections as shown in this table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text to display</td>
<td>999.00</td>
</tr>
<tr>
<td>Type</td>
<td>Number</td>
</tr>
<tr>
<td>Format</td>
<td>#,##0.00</td>
</tr>
</tbody>
</table>
3. To display the total for each currency, enter the text **Currency Total Remaining:** beneath the table, but inside the **for-each** / **end** tags. Insert the field as follows:

   a. On the BI Publisher tab, in the Insert group, click **Field**.
   b. In the Field dialog box, click the **C Amount Rem** field (under the G Vendor group).
   c. In the **Calculation** field, select **Sum**.
   d. Click **Insert**.
   e. Format the field as a number, as described above.
4. Perform any additional formatting of colors, fonts, borders, and text strings using Microsoft Word functionality.

Previewing the Layout

In the Oracle BI Publisher tab, in the Preview group, click PDF.

Customizing BI Publisher Templates: Procedures

Oracle Business Intelligence (BI) Publisher templates are created using the BI Publisher layout editor, a design tool that provides an intuitive, drag-and-drop interface for creating pixel perfect reports in PDF, RTF, Excel, PowerPoint, and HTML. It also provides dynamic HTML output that supports lightweight interaction through a browser.

BI Publisher layouts are best suited for reports of simple to medium complexity. The interactive view is only available for BI Publisher layouts, therefore choose this layout type when you want your report consumers to interact with the report (change sorting, apply filters, and so on).

Using the Layout Editor

To customize BI Publisher templates:

1. Make sure that sample data is generated from the data model that the report is using.
2. Navigate to the report within the catalog and click **Edit**.
3. Under the Create Layout section, click a template to launch the layout editor.
4. Create the layout using the guidelines in the help for the layout editor.
5. Click **Save** to save the layout to the report definition.

**Customizing Data Models: Points to Consider**

A data model defines the source and structure of the data for a report. At run time, Oracle Business Intelligence (BI) Publisher executes the data model to supply the XML data for a report.

Create a custom data model when the predefined data models do not provide the data required in your report.

If you need to customize the data that is captured by the report data model, you can either copy and edit an existing data model or create a new data model.

For more details on customizing data models, see the help on the pages you use to work with data models.

**Predefined Data Models**

In the Oracle BI Presentation Catalog, copy the predefined data model and paste in the corresponding folder within Shared Folders - Custom so that both versions have similar file path. Edit only the new copy in the Custom folder.

**Data Tables**

In BI Publisher the Oracle Fusion Applications tables are provisioned as follows:

- **ApplicationDB_FSCM** - includes the applications data for Oracle Fusion Financials, Supply Chain Management, Project Management, Procurement, and Incentive Compensation
- **ApplicationDB_CRM** - includes the Oracle Sales Cloud applications data
- **ApplicationDB_HCM** - includes the Oracle Fusion Human Capital Management applications data

Typically, you create data sets from SQL queries against your Oracle Fusion application data tables.

**Parameters**

The order of parameters is important if there are Oracle Enterprise Scheduler job definitions defined for reports that use your data model. If you change the order in the data model, the job definitions must also be updated.

**Creating Custom Reports: Procedures**

Create a custom report when the predefined reports do not provide the data you need, or if you want to use a predefined data model but change other properties of the report. Save your custom report to **Shared Folders - Custom** in the Oracle Business Intelligence Presentation Catalog.
Report Creation Process

To create a custom report:

1. From the toolbar of the catalog, select **New - Report**, then select the data model to use as the data source for this report.

2. Continue with the wizard to create the report layout, or select to use the report editor and close the wizard.

3. Define the layout for the report, and add translations for the layouts if needed.

4. Configure a variety of properties to set specific formatting, caching, and processing options for your report.
   To access the Report Properties dialog box, click the **Properties** button in the report editor. For more information on configuring report properties, see the Oracle Fusion Middleware Report Designer’s Guide for Oracle Business Intelligence Publisher.

5. Optionally create an Oracle Enterprise Scheduler job definition so that users can run the custom report as a scheduled process.

6. Optionally enable access to the report through the Reports and Analytics pane.

7. Secure your report.

Setting Reports Up to Run as Scheduled Processes: Points to Consider

You can set up predefined or custom Oracle Business Intelligence (BI) Publisher reports to be run as scheduled processes. Otherwise, users can open reports appropriate to be run online through the Reports and Analytics pane or from the Oracle BI Presentation Catalog.

Create an Oracle Enterprise Scheduler job definition for the report. A job definition contains metadata that determines how the scheduled process works. Use the Define Custom Enterprise Scheduler Jobs task in the Setup and Maintenance work area to create job definitions.

**General Job Definition Information**

This table describes the general information to enter for the job definition.

<table>
<thead>
<tr>
<th>Field</th>
<th>What You Enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job Type</td>
<td>BIPJobType</td>
</tr>
<tr>
<td>Report ID</td>
<td>The path to the report in the catalog, starting with the folder beneath Shared Folders, for example: Custom/Financials/Payables/Payables InvoiceRegisterCustom.xdo.</td>
</tr>
</tbody>
</table>

**Tip**

Ensure that you include the .xdo extension for the report object.
Parameters

You can define parameters to be available to users when they submit scheduled processes based on your job definition. The parameters that you define must be in the same order as parameters in the data model that the report is using.

For example, the data model has parameters in the following order:

- P_BUSINESS_UNIT
- P_VENDOR_ID
- P_INVOICE_TYPE

You create parameters as follows:

- Business Unit
- Supplier ID
- Invoice Type

When users run the scheduled process, the values they enter are passed to the data model and determine the data to be included in the report.

Note

Because you define parameters using the list of values sources from the Define Custom Enterprise Scheduler Jobs task, you should not define lists of values in the data model.

User Property

The only user property you need to define is EXT_PortletContainerWebModule. Only lists of values associated with the application that you select are made available for parameters in this job definition.

Setting Up Reports for the Reports and Analytics Pane: Procedures

Users can access Oracle Business Intelligence (BI) Publisher reports from the Reports and Analytics pane, which is available in various work areas. Reports can be set up to be submitted as scheduled processes. For users to submit these
scheduled processes from the Reports and Analytics pane, you must configure report properties.

Making Reports Available

To determine which reports are available to users in the Reports and Analytics pane, click Edit Settings in the Reports and Analytics pane, or use the Map Reports to Work Areas task in the Setup and Maintenance work area.

Enabling Reports for Scheduling

To enable scheduling in the Reports and Analytics pane:

1. Go to the report in the Oracle BI Presentation Catalog and click Edit.
2. Click Properties.
3. On the General tab in the Properties dialog box, enter the following fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise Scheduler Job Package Name</td>
<td>The path for the job definition, for example: / oracle/apps/ess/financials/payables/invoices/transactions/Jobs</td>
</tr>
<tr>
<td>Enterprise Scheduler Job Definition Name</td>
<td>The job definition name (not display name), for example: APXINRIR</td>
</tr>
</tbody>
</table>

Securing Reports and Related Components: Procedures

When you create a custom report, you must secure it so that only users with appropriate privileges can run it. You can also update security definitions for custom reports that inherited those definitions from predefined reports.

For securing reports, you can use:

- **An existing role:** When a custom report is functionally aligned with a predefined report. For example, if a user who runs the Account Analysis report should also run or have access to the custom report, then you can use the same role and entitlements from the Account Analysis report rather than creating a custom role.

- **A custom role:** Which administrators create, or, in Oracle Cloud implementations, can be requested by logging a service request using My Oracle Support at [https://support.oracle.com](https://support.oracle.com).

Configuring Permissions in the Oracle Business Intelligence Presentation Catalog

For a user to run a report, Read permissions must be granted to that user's role on every component in the Catalog that is used in the report:

- Report
• Data model
• Subtemplate or style template (only if used)

The report object requires additional grants to run, schedule, and view output.

**To grant permissions in the catalog:**

1. Go to the object in the catalog and select **More - Permissions**.
2. In the Permission dialog box, click the **Add users/roles** icon button.
3. In the Add Application Roles, Catalog Groups and Users dialog, select a role from the **Application Roles** list.
4. In the **Set Permission to** field, select **Custom**, and then click **OK**.
5. In the Permission dialog box, click the **Edit Custom Permissions** icon button for the role you added.
6. In the Custom Permissions dialog box, select the permissions to enable.

For all report component types, enable:

• **Read**: To access, but not modify, the object.
• **Traverse**: To copy the object.

For report objects only, you typically also enable:

• **Run Publisher Report**: To read, traverse the folder that contains the report, and run the report.
• **Schedule Publisher Report**: To read, traverse the folder that contains the report, and schedule the report.
• **View Publisher Output**: To view the report output generated by the scheduler.

**Granting Permissions to the Oracle Enterprise Scheduler Job Definition**

If your custom report has an Oracle Enterprise Scheduler job definition defined, then execution permissions must be granted to enable users belonging to the role to submit the corresponding job. For more information on extending custom Oracle Enterprise Scheduler jobs using existing Oracle Fusion applications, see the Oracle Fusion Applications Extensibility Guide for Developers.

**Note**

This task does not apply to Oracle Cloud implementations.
Help File Customization

Help File Customization: Overview

If you have the appropriate job roles, then you can customize the help files in the help site. Use the Manage Custom Help page to maintain both predefined and custom help files. You can create, duplicate, edit, and delete custom files, or set their status to Active or Inactive. For predefined files, you can only duplicate them or set their status.

For each help file, predefined or custom, use help locations to determine where the help file appears in the application and in the help site. You have various options in how you add custom help, for example by uploading a file or specifying a URL. You can upload files of any type, and your URL can also point to files of any type.

Note

To make a copy of all custom help for testing, migration, or other purposes, create a configuration package then use the export and import feature in the Setup and Maintenance work area. The configuration package must use a source implementation project that contains the Define Help Configuration task list and you must select the following objects to export: Help Configuration and Help Topic.

Customizing Help in Help Windows

Many help files can be accessed from help windows in the application. If you want to customize help in the context of a help window, for example create a custom help file and add a link to it from a specific help window, then start by opening that help window. When you click the Manage Custom Help link, you go to the Manage Custom Help page, and the help location fields are automatically populated with values that correspond to the help window. This way you can easily select existing files to add to the same help location, and when you create a new file, the same help location appears by default.
Restriction
Aside from links to help files, some help windows also display informational text. You can't use the Manage Custom Help page to edit this text. Your technical administrators can do so using developer tools (not available in Oracle Cloud implementations).

Customizing Help by Help Location
Open the Manage Custom Help page directly from the home page of Oracle Fusion Applications Help or from search result pages.

Tip
When you search in the Manage Custom Help page, make sure that the Custom Help Only check box is not selected if you are looking for predefined help.

Editing Specific Files
To edit a specific file, you can either find it in the Manage Custom Help page, or open the file itself and click the Edit link.

Customizing Glossary Terms
You can find glossary terms in the Manage Custom Help page, or go to Navigator - Glossary in Oracle Fusion Applications Help to open the Glossary tab, search for the term, and click Edit.

Adding UPK to the Settings and Actions Menu
If your enterprise has purchased Oracle User Productivity Kit (UPK) content, then your administrator can also add a UPK item to the Settings and Actions menu in the global area of Oracle Fusion Applications. When users select this menu item, they access UPK content specific to the page that they are on.

Help Types: Explained

Oracle Fusion Applications Help contains various types of help content, including demos, examples, FAQs, glossary terms, help topics, and PDF guides. A business process or product can be supported by some or all of these help types.

Demo

Demos are Oracle User Productivity Kit (UPK) topics that visually demonstrate how to use the application to complete a short task or portion of a task. Demos can also provide an introduction to complex dashboards and work areas.

Example

Examples provide real use cases of features to illustrate how and when to use the feature, or scenarios to illustrate abstract concepts. Worked examples show
exactly what you need to do to achieve a specific result, emphasizing decisions that you make and values that you enter.

**FAQ**

FAQs, or frequently asked questions, provide brief answers to questions that you might have regarding a task or page. For example, they can briefly explain what a term means, why something happened, how you can perform an action, or what happens if you perform the action.

**Glossary**

Glossary terms provide definitions for words or phrases used in help. You can search or browse glossary terms in the Glossary tab of Oracle Fusion Applications Help. Where the links are available, you can also see the definition when you hover over the term in help content for other help types.

**Help Topic**

Help topics explain key concepts, illustrate how application components work together, or assist in decision-making by explaining points to consider or the options you have. Help topics can also provide reference, overview, and other information.

**PDF Guide**

PDF guides in Oracle Fusion Applications Help provide, in a book format, information usually not found in other help types. There are other guides that present a collection of help content from the other help types, except demos, in an organized and logical format. These guides, for example, address specific business processes and setup offerings. You can find these guides by going to **Navigator - Documentation Library** in Oracle Fusion Applications Help.

**Help Locations: Explained**

Help locations determine where users can find help files, custom or not, from either the application or the help site.

Help locations include:

- Page or section values
- Help hierarchies
- Primary locations

**Page or Section Values**

The value in the **Page or Section** field on the help customization pages represents where users can click a help icon to open a help window that contains...
a link to the help file. In most cases, this value represents a page or region header in the application. Help windows are also available on specific tabs or windows, and in the Setup and Maintenance work area for specific task lists or tasks. You can associate a help file with multiple page or section values, or with none at all.

The page or section value reflects the logical navigation to the help window. For example, Edit Opportunity page, Revenue tab, Recommendations window does not mean that the help file is available in three different places. The help icon is in the Recommendations window, which is accessed from the Revenue tab on the Edit Opportunity page.

If the value suggests multiple locations, for example Create and Edit Opportunity pages, then the help file is available from the page header of both the Create Opportunity and Edit Opportunity pages. If the page or section value is, for example, a dashboard region that appears in multiple dashboards, then the value does not specify the page name but just the region. The help file is available from that region in multiple dashboards.

Help Hierarchies

Help files are associated with help hierarchies, which are used to categorize help files and aid users in finding help. Each help file can have multiple hierarchies, with at least one of type Business Processes. The business process hierarchy is based on the Business Process Management model. Every page or section value is predefined with a specific business process hierarchy. If you select a page or section without entering a business process hierarchy, the predefined hierarchy appears by default.

The Common Tasks navigator is based on the Welcome hierarchy type. The level 1 nodes represent categories of functional areas common to all users.

The Business Processes navigator in the help site is based on the business process hierarchy. For example, if you assign two business process hierarchies to a help file, users can find the file in both locations in the navigator. When the user clicks More Help from a help window, all help files assigned to the same business process hierarchy as the page or section value are returned as search results.

Similarly, the Products navigator is based on the Product hierarchy type, in which level 1 is the product family, level 2 is the product, and level 3 is the business activity owned by that product.

The Functional Setup navigator is based on the Functional Setup hierarchy type. The level 1 nodes for this hierarchy are:

- Functional Setup Manager, which includes help about using the Setup and Maintenance work area.
- Offerings, which contains level 2 nodes for each setup offering, and lower levels for the main task lists in the offerings. Help for the task lists and tasks are included.

Primary Locations

The primary location of a help file designates the hierarchy that is displayed for the help file in search results and within the help content as breadcrumbs.
You cannot change the primary location of a help file that came with your help installation. Primary locations of predefined help are based on the business process hierarchy, while custom help files can have primary locations based on hierarchies of any type.

### Editing Predefined Help and Glossary Terms: Points to Consider

When you open any predefined help file, including glossary terms, that came with Oracle Fusion Applications Help, you can see an edit option if you have roles allowing edit access. When you edit predefined help, keep in mind:

- What happens to the original help file
- Where predefined help appears
- Considerations specific to glossary terms

#### What Happens to the Original Files

When you edit predefined help, you are actually creating a new custom help file based on the original file, with the same help locations. The customized version replaces the original, which becomes inactive and hidden from users. You can display both versions by reactivating the original in the Manage Custom Help page.

**Note**

In the Search Results: Existing Help region on the Manage Custom Help page, there is no option to edit predefined help. You can duplicate a predefined help file, edit the copy, and optionally inactivate the original.

#### Where Predefined Help Appears

All predefined help comes with preassigned help locations, including at least one based on the hierarchy of type Business Processes. Many also have predefined page or section values that indicate where the help can be accessed from help windows in the application.

To change where predefined help appears, either in the help site navigators or in the application, create a duplicate in the Manage Custom Help page. Change or add help locations to your custom copy, and inactivate the original.

Even though glossary terms do not appear in the help site navigators, you still need to enter at least one help location to categorize the glossary term.

#### Considerations Specific to Glossary Terms

When you edit a predefined glossary term, the original term becomes inactive. Existing links to the glossary term, from other predefined and custom help files, will automatically point to your custom version. If you later inactivate the
custom glossary term, make sure to activate the original term so that the links still work.

**Links in Custom Help: Points to Consider**

When you create or edit custom help, follow best practices when you include links to help files or other content. If you are working on custom help created by duplicating a predefined help file, then you may see existing links from the original file in the Help Content section. The types of links that you can work with include:

- Related help links
- Standard hypertext links
- Links to documentation library content
- Glossary term links

For all link types, except the standard hypertext links, you must create or edit custom help with a Text or Desktop source type. In other words, you must type the help content directly in the application or use an HTML file that you upload to help. For standard hypertext links, the source type can also be URL.

**Related Help Links**

Related help is the section at the end of help files that contains links to other help files. The syntax for related help contains a comma-separated list of title IDs that represent help files.

This figure provides an example of related links code.

```
OfsRelatedTopics('CREATE_AUTOMATIC_POSTING_CRITERIA_S_0000, JOURNAL_SOURCES_NOM_TABLE_FY_RECORDS_0000')
```

- You can delete this code to remove all related help, or delete title IDs to remove individual links (for example, `CREATE_AUTOMATIC_POSTING_CRITERIA_S_0000`).

- To replace existing links or add new links, you need to retain the code syntax and enter desired title IDs. To find title IDs, search for the help files on the Manage Custom Help page. Title IDs are displayed in the search results, but the **Title ID** column is hidden by default.

**Standard Hypertext Links**

You can create standard hypertext links to any file or Web site as long as you ensure the stability and validity of the links, including links to other help files, custom or not. These links can appear anywhere in the body of your help file as long as they come before any related help links.

In the Help Content section, highlight the text that you want to use as link text and click the **Add Link** icon button.
For links to other help files, open the file to which you want to link, and click the **E-Mail** link. Use the URL in the autogenerated e-mail text as the link to the file.

**Note**

Use the full URL, for example http://www.oracle.com, when creating links.

**Links to Documentation Library Content**

The syntax for links to HTML files in documentation libraries is:

```html
<span class="HP_topic-link_bridgeDocument-linkToSTDoc_"<?ofa
linkToSTDoc(WCSUG4636) ?>\<span class="HP_topic-linktext_"">Understanding
Tags</span></span>.<h2>WCSUG4636 is the anchor ID and Understanding Tags is the link text. You can:

- Modify the link by replacing the existing anchor ID or editing the link
text, or both.
- Remove the link by deleting all the code for it.
- Create links to documentation library content by following the same
syntax. These links can appear anywhere in the body of your help file as
long as they come before any related help links.

**Important**

To ensure that you are linking to a supported documentation library, enter
anchor IDs only from documentation libraries that are linked from predefined
help topics.

**Glossary Term Links**

Glossary term links provide definitions in a note box when users hover over the
term in help files.

This figure shows an example of code for a glossary term link.

```html
OfGlossaryTerm"accounting period", ACCOUNTING_PERIOD_0901
```

In this example, accounting period is the link text, or glossary term, and
ACCOUNTING_PERIOD_001 is the identifier, or title ID.

- To remove the link but retain the text, delete all the code except the term
  itself.
- To add glossary term links, you must follow the link syntax and use the
  correct title ID for the glossary term. You can find title IDs in the search
  results of the Manage Custom Help page.

**Note**

If your custom help has glossary terms and the source type is Desktop File, then
make sure before uploading that the quotes around the glossary term are actual
Customizing PDF Guides: Worked Example

This example demonstrates how to customize a PDF guide that came with Oracle Fusion Applications Help. This guide is currently not available from any help window in the application.

The following table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>What changes do you need to make to the guide?</td>
<td>Change the title of a chapter and remove a section in that chapter, to hide content about a particular subject</td>
</tr>
<tr>
<td>Which help window should the customized guide appear in?</td>
<td>The help window for the entire Welcome dashboard of Oracle Fusion Applications</td>
</tr>
<tr>
<td>Which help navigators should the customized guide appear in, and on which node?</td>
<td>Same as the original guide, plus the path associated with the help window</td>
</tr>
<tr>
<td>Do you want to limit access to the customized guide?</td>
<td>No, same as the original guide</td>
</tr>
</tbody>
</table>

Edit a copy of the original PDF guide, and use the Manage Custom Help page to replace the original PDF guide with your new file.

Copying and Editing the PDF Guide

1. Open the original PDF guide from the help site and save a copy to your desktop. Leave open the help file for the guide.
2. Using a PDF editor application, change the title of the chapter wherever the chapter title appears. Delete the content you want to hide from users.

Replacing the Original PDF Guide

1. In the help file that you still have open for the original PDF guide, click the Edit link.
2. On the Create Help page, use the default values except where indicated.
3. Update the title to the name that you want to display to users.
4. In the File Name field, browse for and select your customized guide.
5. Delete any keywords or parts of the description relevant to the content you removed from the PDF guide.
6. Add a help location with the Business Processes hierarchy type and select Information Technology Management as the level 1 node, Manage Enterprise Application Software as the level 2 node, and Use Applications as the level 3 node.
7. Select Welcome page in the Page or Section column.
8. Click Save and Close. The help file for the original PDF guide is automatically set to inactive.
Adding Custom UPK Content to Help: Worked Example

This example demonstrates how to add custom Oracle User Productivity Kit (UPK) topics as demo help files. These help files function like any predefined help file for demos. You can search and include these files in help windows and navigators as you would other help.

In this scenario, you are adding one demo about activity streams, to appear in a help window on the Welcome dashboard.

**Note**
Your demo must be made with UPK 3.6.1 or later to be added as help.

The following table summarizes key decisions for this scenario.

<table>
<thead>
<tr>
<th>Decisions to Consider</th>
<th>In This Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>What UPK content do you want to add to help?</td>
<td>From a UPK module containing five topics, add one as a custom demo on the help site</td>
</tr>
<tr>
<td>Which help navigators should the demo appear in, and on which node?</td>
<td>Because the demo is about activity streams:</td>
</tr>
<tr>
<td></td>
<td>• Search by Common Tasks navigator, under the Collaboration node</td>
</tr>
<tr>
<td></td>
<td>• Search by Business Process navigator, under Information Technology Management- Manage Collaboration - Manage Collaborative Communities</td>
</tr>
<tr>
<td>Which help window should the demo appear in?</td>
<td>On the Welcome dashboard of Oracle Fusion Applications, in the help window in the Activity Stream region</td>
</tr>
<tr>
<td>Do you want to limit access to the help file for the demo?</td>
<td>No</td>
</tr>
<tr>
<td>Do you want the help file to appear in the New and Updated pane?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Generate a report of UPK document IDs, which you will use when creating custom help, to identify the UPK topic that you want to add. Publish the UPK module as a player package, then create custom help for the UPK topic that you want to use as a help demo.

**Generating a UPK Document ID Report**

1. In the UPK Developer, select **Details View**.
2. Right-click any column header, for example Name, and select **Column Chooser**.
3. In the Column Chooser dialog box, click and drag the Document ID column header and drop it after the Name column. Close the Column Chooser dialog box.
4. From the File menu, select to print, and save the output as a Microsoft Excel file to your desktop.
Creating the Player Package

1. From the UPK Developer, make sure that the topic that you want to add as a demo has the See It play mode. The topic can also have other modes, but only the See It mode is included in the custom help file.

2. Publish the module, specifying any location for the output and selecting to publish the selection only.

3. In the Formats section of the Publish Content window, select the Player check box under the Deployment check box group.

4. In the Player section, select the Include HTML Web Site check box, to ensure that the custom help file includes a text-only version of the UPK topic.

5. Finish the publishing process, after optionally setting other options.

6. Navigate to the location where you specified the output to be generated.

7. In the Publishing Content folder, copy the PlayerPackage folder and add it to the web server where you store UPK content.

Creating Custom Help for Demos

1. Open the help window in the Activity Stream region on the Welcome dashboard of Oracle Fusion Applications, and click Manage Custom Help.

2. On the Manage Custom Help page, the page or section and hierarchy values are populated with the values for the Activity Stream region.

3. Click Create.

4. On the Create Help page, complete the fields in the General Information section, as shown in this table. Use the default values except where indicated.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>The name of the UPK topic.</td>
</tr>
<tr>
<td>Source Type</td>
<td>Oracle User Productivity Kit</td>
</tr>
<tr>
<td>File Location</td>
<td>The full URL of the player package folder on the Web server, for example, http://&lt;your domain&gt;.com/UPKcontent/PlayerPackage.</td>
</tr>
<tr>
<td>Document ID</td>
<td>The document ID of the UPK topic to add to the help window in the Activity Stream region. You can copy and paste this ID from the Microsoft Excel file that you generated earlier.</td>
</tr>
<tr>
<td>Help Type</td>
<td>Demo</td>
</tr>
<tr>
<td>Help Security Group</td>
<td>Unsecured</td>
</tr>
<tr>
<td>Keywords</td>
<td>Terms relevant to the demo.</td>
</tr>
<tr>
<td>Description</td>
<td>Summary of the demo.</td>
</tr>
<tr>
<td>Include in New and Updated pane</td>
<td>Selected</td>
</tr>
</tbody>
</table>
The Help Location section contains values for the help window in the Activity Stream region. This help file will also appear in the Search by Business Process navigator under this predefined hierarchy.

5. Click Save and Close.

6. On the Manage Custom Help page, open the help locations for the help file that you just created.

7. Add a help location with the Welcome hierarchy type and select Collaboration Features as the level 1 node.

8. Click Save and Close.

FAQs for Help File Customization

**How can I add a Youtube video to custom help?**

On the Create Help page, select URL as the Source Type. Find the video in Youtube and click the Share button then click the Embed button. Copy the URL within the embed code, for example http://www.youtube.com/embed/<unique ID>, and paste it in the URL field of the Create Help page.

**How can I restrict help content to specific user roles?**

When you create or edit help, select a help security group that represents the set of roles that you want to have access to the help. If you do not see the Security Group field, then your administrator has not selected the Custom Help Security feature choice. The Unsecured group has no associated roles, so anyone can view the help. The predefined Secured group includes all internal employees and contingent workers, unless this group has been edited. You can create security groups and associate roles using the Manage Help Security Groups page, which you can access by starting in the Setup and Maintenance Overview page and searching for the Manage Help Security Groups task. Your new security groups are immediately available for use to secure new or edited help files.

**Why can't I select and add help to a location?**

You must specify a page or section to add the existing help to. To ensure that help is added to the correct help window, go to the page or section in the application, click the Help icon, and click the Manage Custom Help link in the help window. Alternatively, in the Manage Custom Help page, search for at least a page or section and a level 1 value for the Business Processes hierarchy type before selecting the Select and Add option.

You cannot select and add help to a particular hierarchy, on the Manage Custom Help page, without a page or section. To add just a hierarchy, search for the help file, add a new help location, and specify only the hierarchy information.

**What happens to custom help when a help patch is applied?**

Oracle Fusion Applications Help patches update all help files, both active and inactive, except custom help. Custom help files are not affected by patches.
Consider reviewing inactive files to see if you want to activate the updated version, or to make similar edits to the custom versions of those files, if any.

Embedded Help Customization

Customizing Embedded Help: Highlights

You can customize help that is embedded in the application, for example hints, for all users of Oracle Fusion Applications. There are different types of embedded help.

Creating, Editing, or Deleting Help

- Use Page Composer to edit, create, or delete hint text that appears on hover over buttons, links, icons, or tab titles. Open the properties of the UI element to define the help text in the shortDesc field. Usually, the value resolves to a key in a resource bundle.

- Edit, create, or delete other types of embedded help. Refer to the Customizing or Adding Static Instructions, In-Field Notes, and Terminology Definitions section.

See: Oracle Fusion Applications Extensibility Guide for Developers
abstract role
A description of a person’s function in the enterprise that is unrelated to the person’s job (position), such as employee, contingent worker, or line manager. A type of enterprise role.

accounting flexfield
The chart of accounts that determines the structure, such as the number and order of individual segments, as well as the corresponding values per segment.

action
The kind of access named in a security policy, such as view or edit.

activity stream
A feature that tracks and displays actions and messages from people whom you are connected to in your social network, as well as activities from the application.

ADF
Acronym for Application Developer Framework. A set of programming principles and rules for developing software applications.

analysis
A graph or table that displays data resulting from queries on real-time, transactional, or operational data, to provide answers to business questions.

application identity
Predefined application level user with elevated privileges. An application identity authorizes jobs and transactions for which other users are not authorized, such as a payroll run authorized to access a taxpayer ID while the user who initiated the job is not authorized to access such personally identifiable information.

BPEL
Business Process Execution Language; a standard language for defining how to send XML messages to remote services, manipulate XML data structures, receive XML messages asynchronously from remote services, manage events and exceptions, define parallel sequences of execution, and undo parts of processes when exceptions occur.

business object
A resource in an enterprise database, such as an invoice or purchase order.
chrome
The set of visual elements around the perimeter of a component or task flow that enables users to act directly on the object. Elements that make up the chrome of a component include the header; border; resize handle; edit, collapse, expand, and remove icons; and Actions menu.

collection
An XML filter or SQL predicate WHERE clause in a data security policy that specifies what portions of a database resource are secured.

collection segment
A grouping of flexfield segments to store related information.

collection segment
The flexfield segment used to store the context value. Each context value can have a different set of context-sensitive segments.

collection-sensitive segment
A flexfield segment that may or may not appear depending upon a context such as other information that has been captured. Context-sensitive segments are custom attributes that apply to certain entity rows based on the value of the context segment.

collection
A change to standard, predefined Oracle Fusion Applications artifacts. Customizations impact multiple users.

collection layer
A level that represents the types of users impacted by a customization, for example all users or only those that meet specific criteria.

dashboard
A collection of analyses and other content, presented on one or more pages, to help users achieve specific business goals.

data dimension
A stripe of data accessed by a data role, such as the data controlled by a business unit.

data model
Metadata defining the data source, data structure, and parameters to be used by reports.
**data role**

A role for a defined set of data describing the job a user does within that defined set of data. A data role inherits job or abstract roles and grants entitlement to access data within a specific dimension of data based on data security policies. A type of enterprise role.

**data security**

The control of access to data. Data security controls what action a user can take against which data.

**data security policy**

A grant of entitlement to a role on an object or attribute group for a given condition.

**database resource**

An applications data object at the instance, instance set, or global level, which is secured by data security policies.

**descriptive flexfield**

Customizable expansion space, such as fields used to capture additional descriptive information or attributes about an entity, such as customer cases. Information collection and storage may be configured to vary based on conditions or context.

**design time**

The type of activities performed by developers at the code or data model level.

**desktop page**

A page in a user interface that is optimized for extended periods of use with monitors.

**desktop user interface**

A standard user interface that is optimized for extended periods of use with monitors.

**enterprise**

An organization with one or more legal entities under common control.

**enterprise role**

Abstract, job, and data roles are shared across the enterprise. An enterprise role is an LDAP group. An enterprise role is propagated and synchronized across Oracle Fusion Middleware, where it is considered to be an external role or role not specifically defined within applications.
entitlement
Grants of access to functions and data. Oracle Fusion Middleware term for privilege.

extensible flexfield
Customizable expansion space, as with descriptive flexfields, but able to capture multiple sets of information within a context and multiple contexts grouped to appear in a named region of a user interface page. Some extensible flexfields let you group contexts into categories.

extension
A completely new artifact in addition to what is predefined in Oracle Fusion Applications, for example a custom business object or page.

feature choice
A selection you make when configuring offerings that modifies a setup task list, or a setup page, or both.

flexfield
Grouping of extensible data fields called segments, where each segment is an attribute added to an entity for capturing additional information.

flexfield segment
An extensible data field that represents an attribute on an entity and captures a single atomic value corresponding to a predefined, single extension column in the Oracle Fusion Applications database. A segment appears globally or based on a context of other captured information.

functional area
A set of simplified pages containing content that a user needs to accomplish a business goal. Examples of this content include searches, data entry fields, and analytics.

global area
The region across the top of the user interface. It provides access to features and tools that are relevant to any page you are on.

job role
A role for a specific job consisting of duties, such as an accounts payable manager or application implementation consultant. A type of enterprise role.

key flexfield
Configurable key consisting of multiple parts or segments, each of which may be meaningful individually or in combination with the others. Key flexfields are commonly implemented to represent part numbers and account numbers.
**key flexfield segment instance**

A single occurrence of a key flexfield segment in a key flexfield structure instance.

**key flexfield structure**

The arrangement of segments in a key flexfield. In some cases, multiple structures can be defined for a single key flexfield.

**key flexfield structure instance**

A single occurrence of a key flexfield structure that shares the same order of segments as every other instance of the key flexfield structure, but uses different value sets to validate the segments.

**mainline**

A branch of data that serves as a single source of truth.

**offering**

A comprehensive grouping of business functions, such as Sales or Product Management, that is delivered as a unit to support one or more business processes.

**Oracle Business Intelligence Presentation Catalog**

The repository where all business intelligence objects, including analyses, dashboards, and reports, are stored. The catalog contains folders to separate personal and shared objects, as well as custom objects.

**personalization**

A change users make to control the look or behavior of the application. Personalizations impact only the user making the change.

**privilege**

A grant or entitlement of access to functions and data. A privilege is a single, real world action on a single business object.

**process**

A program that you schedule and run to process data and, if appropriate, generate output as a report. Also known as scheduled process.

**role**

Controls access to application functions and data.

**run time**

The type of activities performed by users while they are in a running application.
sandbox
A run time session that commits changes out of reach of mainline users.

segment
See

equipped page
A page in a user interface that is optimized to provide quick access to high-volume, self-service tasks from any device.

simplified user interface
A user interface that is optimized to provide quick access to high-volume, self-service tasks from any device.

site layer
The customization layer in which customizations that are made affect all users.

SOA
Abbreviation for service-oriented architecture.

springboard
The area on the simplified user interface home page, and above all simplified pages, that contains a set of functional area icon buttons.

SQL predicate
A type of condition using SQL to constrain the data secured by a data security policy.

style template
An RTF template containing style information that is applied to other RTF layouts to achieve a consistent look and feel across reports.

subtemplate
A piece of formatting that is defined once and used multiple times within a single report layout or across multiple layout files.

tree
Information or data organized into a hierarchy with one or more root nodes connected to branches of nodes. A tree must have a structure where each node corresponds to data from one or more data sources.

tree structure
Characteristics applied to trees, such as what data to include or how the tree is versioned and accessed.
user layer

The customization layer in which personalizations are made, which impact only the user making the change.

value set

A set of valid values against which values entered by an end user are validated. The set may be tree structured (hierarchical).

XML filter

A type of condition using XML to constrain the data secured by a data security policy.