Oracle Fusion Cloud SCM

Configuring and Extending Product Lifecycle Management

23D

Oracle Fusion Cloud SCM
Configuring and Extending Product Lifecycle Management

23D

F85805-02

Author: Divya Begur, Usha Pereira

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Use help icons to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessibility Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, suggest *ideas* for product enhancements, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 Extend Product Lifecycle Management

About This Guide

You can configure and extend the application interfaces and the business objects in Oracle Product Lifecycle Management (PLM) Cloud to support your business needs. The recommended tool for such configuration is Application Composer.

This guide describes the types of configuration that you can perform, and includes frequently asked questions about how to effectively use Application Composer.

Audience

This guide provides information on how administrators and implementors can make application changes using Application Composer.

Additional Information

The table provides a list of the guides and videos that have detailed information on the capabilities of Application Composer.

Related Information

Guide/Video	Description
Configuring Applications Using Application Composer	This guide provides information on how administrators and implementors can make application changes using Application Composer.
Groovy Scripting Reference	This document explains the basics of how to use Groovy scripting language to enhance the functionality of the application.
Watch Videos: Extending PLM Applications	These videos show how to extend and configure PLM objects.

Overview of Application Composer

Application Composer is a browser-based tool that you as an administrator can use to configure applications. You can use this tool to make data model changes.

Administrators can create and configure layouts to meet business requirements. For example, you can create a new object and related fields, and then create new interface pages to expose that object to users. Application Composer is a design-at-runtime tool, which means that you can navigate to Application Composer directly from a Cloud application, make your changes, and see that most changes take immediate effect, without having to sign back into the application.

For Quality Management, Innovation Management, and Product Development, you can use Application Composer to extend the application interface.

Note: To configure change order attributes, use Extensible Flexfields (EFFs).

Note: Application Composer is supported for use only in English. Additionally, Application Composer isn't supported for use with iPad devices.

Overview of Extensible Flexfields

An extensible flexfield provides a configurable expansion space that implementers, such as Oracle Fusion Applications customers, can use to configure additional attributes (segments) without additional programming.

Note: This configuration applies to change orders, change requests, problem reports, corrective actions, items, and manufacturer parts. You can define fields for other objects in Application Composer.

Related Topics

- Extensible Flexfields

What You Can Configure

Use Application Composer to:

- Edit the display label and help text of standard fields;
- Create conditional layouts;
- Assign fields to layouts;
- Create fields of different types (such as text, long text, number, date, choice list, and check box) and add them to standard and administrator-defined objects;
- Define application actions using validation rules, triggers, and functions;
- Set field-level and object-level validation rules;
- Create new objects and child objects;

  Note: You can’t create child objects for change orders.

- Create new subtabs and connect to external applications;
- Hide or show objects and tabs;
- Use web services to create and update objects;
- Delete configured attributes that are created in a sandbox.

  Note: Ensure that the configured attributes you're deleting aren't in use or have never been published.

Attributes, or fields, must be assigned to a layout for the application user to see and work with them. A conditional statement assigned to a layout determines when it's displayed and who can see it.

Configuration Scenarios

Let's look at some scenarios where you can tailor the application interface to achieve a specific requirement.

Scenario 1

Your business requires that the description field of all change orders must be automatically updated with the department and location entered by the user.

Solution:

1. In the Setup and Maintenance work area, navigate to:
    - Offering: Product Management
    - Functional Area: Change Orders
    - Task: Manage Change Order and New Item Request Header Descriptive Flexfields
2. Create global descriptive flexfields entitled **Department** and **Location**.
3. In Application Composer, create a script that copies the values of the **Department** and **Location** fields into the **Description** field.

Scenario 2

When users create a quality action, you want to make sure that the triage date is automatically set to 3 days after the creation date.

Solution: Within Application Composer, create a new date field called Triage Date. Define the default value of Triage Date using a Groovy script expression that captures the creation date and adds 3 days. Define an expression in a way that doesn't allow the field to be updated.

Scenario 3

Your business process requires that a requirements specification is created for every idea that has an Approved status. Also, the requirements specification must have the same name and description as the original idea.

Solution A: Within Application Composer, you can add a button to an idea that calls a web service. The web service will create a requirements specification and copy the name and description of the idea to the new requirements specification.

Solution B: Within Application Composer, you can create a trigger that calls a web service whenever an idea is set to Approved status, which will create a requirements specification and copy the name and description of the idea to the new requirements specification.

Scenario 4

On a quality issue, your users must select a service type and then a service center. The selection of the service center is dependent upon the selection of the service type.

Solution A: Within Application Composer, create two fixed choice list fields with appropriate values for service type and service center. Designate the service type field as the parent of the service center field and map the values as wanted so that when a user selects a service type, only the applicable service centers are available for selection.

Solution B: Within Application Composer, create two fixed choice list fields with appropriate values for service type and service center. Use Groovy scripting to create a trigger that will automatically select the service center based on the selection of service type.

> **Note:** Refer to the Triggers section in Create Automation with Scripting.

Related Topics

- Triggers in Create Automation with Scripting

Configuration Nodes

When you open Application Composer, each object type that's available to you is displayed, along with a set of configuration nodes. Here's what you can do in each node:

- Fields: modify standard fields and create new fields.
- Pages: create and modify page layouts.
- Actions and Links: create internal actions or links to external applications with Groovy scripts.
- Server Scripts: create validation rules, triggers, and object functions with Groovy scripts.

  > **Note:** Supported functionality for change orders differs from other objects.

PLM Objects You Can Configure

PLM uses various objects in its processes, and you can configure or extend all of them to some extent. When you plan your configuration requirements, be aware of what's supported for each object type.

Object Types

You can configure the landing page, the Create dialog box, and the Details page of the following objects in Application Composer:

- Idea
- Requirements Specification and Requirements
- Proposal
- Quality Action
- Quality Issue
- Change Order
- Change Request
- Problem Report
- Corrective Action

- Concept

Note: A Create dialog box must include all fields that are tagged as Required. This includes Required fields that you've renamed. If not, an error message appears as you set up the Create dialog box for a business object.

What You Can Configure For Each Object Type

This table summarizes the specific kinds of configuration that you can perform on each PLM object type.

Configuration Types and Objects

Configuration Type	Change Order	Quality Issue and Quality Action	Idea	Proposal	Concept	Concept Component	Requirement	Requirement Specification
Additional Attributes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Page Layouts	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Subtabs	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Show or Hide Tabs	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Configure Buttons or Actions	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Field Groups	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Configure URL Tabs	Yes	No	Yes	Yes	Yes	Yes	No	Yes
Configure first-level Objects	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Configure Child Objects	No	Yes	Yes	Yes	Yes	Yes	No	Yes

Note:

- In Oracle PLM Cloud applications, you're always in an active environment. When you plan to create user interface entities in Application Composer, you must first open a sandbox. After you do that, it's safe to open Application Composer and work without changing the production interface pages until you're completely prepared.
- All page layouts created using Application Composer must be duplicates of the standard template not a copy of another configured page layout.
- Concept structures don't support long text and formulas.
- In the **Pages** node for **Change Orders**, you can make page layouts for the Details page, but you can't modify the fields on the page layout. You can't configure the landing page or the Create dialog box.
- You can’t create child objects for change orders. The Create Child Object button is no longer available for change order objects in Application Composer.
- You can define conditions in Application Composer to display the same page layout with different fields for different roles or users.

Enable Auditing for Configured Objects

You can configure specific objects and attributes using audit policies, which can then be audited for configuration changes.

Once you enable the objects for audit, navigate to Audit Reports, and view changes such as when the object or its configured attributes were created, updated, or deleted.

1. Navigate to the **Setup and Maintenance > Product Management > Audit Trail** functional area.
2. Click the **Manage Audit Policies** task.
3. Click **Configure Business Object Attributes** to set objects which will be a part of the audit.
4. From the Product list, select **Product Hub** for configured objects.

Set Up Access

You must be assigned certain job roles along with associated privileges to access and work in Application Composer.

Ensure that you've the following roles:

- Any role that contains the Manage Extensible Object privilege, and Administrator Sandbox privilege, for example, Application Implementation Consultant.
- Custom Objects Administration role.

  Note:
  - To access and manage configured child objects created using Application Composer, you must have the Custom Objects Administration role (ORA_CRM_EXTN_ROLE). This role is created dynamically when you create a new object using Application Composer. Next, you can assign users to the role.
  - You can’t create child objects for change orders. The Create Child Object button is no longer available for change order objects in Application Composer.

- Product Manager role.
- Quality Analyst role.

Related Topics
- Get Started

Use a Sandbox

You use sandboxes to make application changes and test them without impacting other users in the environment. Make changes to the application whenever you're in a sandbox rather than making direct changes in the mainline environment.

To access the sandboxes, navigate to **Configuration** > **Sandboxes** work area.

Completed application changes created within a test-only sandbox have to be published if they have to be available to other users in the application in the mainline metadata. While creating the sandbox, in the Publishable field, select **Yes** or **No**. If you set the Publishable option to **No**, you can use your sandbox for testing only, but can't publish it. When you're in a sandbox, you can open Application Composer and work without changing the production interface pages.

Also, a best practice is to create a new sandbox for each process change you intend to implement. As an example, say you want to create a set of new basic fields that fit your business process but have no dependency on the other fields, do that in one sandbox and publish it. If you want to create a validation for field or set of fields, do all of that in one sandbox and publish it. Therefore, you can group your modifications together according to the functionality that you want to provide and release or publish them incrementally one sandbox at a time.

Use the Unified Sandboxes user interface. This is the default feature that you get.

Note:

- Don't create or publish a sandbox for Application Composer or Page Composer configurations while other users or processes are deploying extensible flexfields and descriptive flexfields changes for Product Development objects.
- Don't deploy flexfields while you've an open sandbox in which you're making Application Composer or Page Composer configurations that you want to publish. Don't deploy the flexfields until the publish operation is complete. Ensure that all the Application Composer or Page Composer configurations in sandbox are tested and published before deploying any extensible flexfields and descriptive flexfields configuration changes for Product Development objects.

Related Topics

- Overview of Sandboxes
- Create and Activate Sandboxes

Get Started

The following procedure is an extremely abbreviated sequence to open Application Composer and have a look around.

Access Application Composer

1. Within an active sandbox, navigate to **Tools** > **Application Composer** work area.
2. When you open Application Composer, the Application list offers **CRM Cloud** and **ERP and SCM Cloud** environments. Select **ERP and SCM Cloud**.
3. For each object node, whether Standard or Custom, expand it further to view and edit object details, such as object fields and pages.
4. For both Standard and Custom Objects, you can view and edit the following details:
    - Fields: Add new fields to an object.
    - Pages: Modify the pages on which an object appears.
    - Actions and links: Add actions or links to desktop pages.
    - Server scripts: Write application logic that controls the behavior of an object's records.

    Note: For change order objects, you can't configure fields in Application Composer.
5. In this brief procedure, you can choose a business object, and we're selecting **Idea**. An idea is a standard object, populated with fields, or attributes. Navigate to **Objects** > **Standard Objects** > **Idea** > **Fields** > **Create** > **Select Field Type** to create fields.
6. After creating the field, go to **Pages**, assign the field to a duplicated **Landing** page layout, and click **Save**.
7. Open **Ideas** > **Manage Ideas** and click on an existing Idea to see the new attribute.

Note: This procedure is applicable for all object types, except change order. For change order objects, you can't configure or create fields in Application Composer.

Related Topics

- PLM Objects You Can Configure
- Define Objects

Review Published Configuration Changes

After you finish configuration tasks in Application Composer, you can review your changes to make sure that everything is in order. In Metadata Manager, click **Generate Configuration Report** to view a summary of configurations made to layout details against a published sandbox.

You can view a summary of modifications across the following in the Application Composer Configuration Report:

- Standard and Custom Objects
- Global and Object Functions
- Standard Fields or Configured Fields
- Custom Relationships
- Validations
- Triggers
- Object Workflows
- Dynamic Layout

Related Topics

- Tools for Moving and Troubleshooting Configurations

Considerations When Using Page Composer and Application Composer Together

Application Composer is the recommended tool for configuring and extending PLM applications. Page Composer can’t be used with all PLM objects. If you attempt to add Page Composer-enabled configurations to existing Application Composer-enabled pages, such configurations can break during an update.

Let's look at PLM objects that you can configure using Page Composer.

Page Composer Support for PLM Objects

Supported	Not Supported
You can configure the following PLM objects with Page Composer: • Change Orders • Change Requests • Problem Reports • Corrective Actions	You can't configure the following objects with Page Composer: • Manufacturers • Quality Actions • Quality Issues • Concepts • Proposals • Requirement Specifications • Ideas

You can use Page Composer to do some configurations for change orders, change requests, problem reports, and corrective actions. Use Page Composer to add components only within the tab regions of the predefined tabs like **General Information**, **Affected Items**, and **Attachments**.

However, you can't make changes to the tabs configured by Application Composer or the tab bar component itself. You can't add, rename, hide, delete, or modify the tabs. This is applicable for actions or buttons on the page.

Upgrade - Safe Page Composer Configurations

Page Composer Configurations	Description and Examples
Rename or hide standard fields in the object pages.	Rename standard fields such as: • 'Description' to 'Description of Change' • 'Requested By' to 'Change Originator' • 'Assigned To' to 'Change Analyst' • Hide the Effective Date field on the Affected Object tab of a Change Order without Revision Control. **Note:** You can use the Show Component property on the Component Properties dialog box to show or hide page components.
Make the fields for standard attributes mandatory.	Make these fields mandatory: • Change Originator • Change Analyst • Reason Code
Reorder the standard and previously configured fields.	Select the page and change the order in which the fields appear in the Component Properties: panelFormLayout page. Add components to the object. Some examples of components: • Box: Use this component to place content on a page. • Image: Use this component to add a picture, a logo, or a linked image to a page. • Hyperlink: Use this component to add a link to a page or a website. • Web Page: Use this component to provide URLs of other web pages within the context of a WebCenter application page.

Page Composer Configurations	Description and Examples
	• HTML Markup: Use this simple editor to enter raw text and HTML tags, including JavaScript embedded in HTML <script> tags. • Text: Use this component to add UI text or any other kind of informative content to a page. • Button: Use buttons on Page Composer only if you don't use Application Composer to add or modify configured buttons, actions, or tabs.
Use Expression Language to show or hide fields and their labels.	Add a condition using an expression to control the visibility of a component. If the condition is met, then the component is displayed, otherwise, the component isn't displayed.
Label and font changes, or any CSS changes.	Configure your page layout to define the number, placement, and orientation of content regions. You can set the layout style while creating a page or you can change the layout style even after adding content to the page. **Note:** You can't change the page layout for all pages.
Add Action, Links, or Buttons.	Add an expression to the **Change Status** button to control the display of change orders and change requests.
Validate the user entry in a field.	Display an error message if a value isn't entered for a specific field. You can do this, by selecting the **Required** check box on the Component Properties page, and entering the message that you want to display.
Configure the Change Overview page.	Hide the **Approve** and **Reject** options in the My Worklist or My Changes on the Overview tab in the Product Development work area.

Additional Configurations with Application Composer

Existing Page Composer Configuration	Additional Application Composer Configuration	Supported on Upgrade?
Button created on the General Information page to launch a PaaS application.	Action to launch a configured extension on the same page.	No
Button created on the General Information page to launch a PaaS application.	Groovy scripting that validates the presence of value in the **Priority** field of the General Information page after changes are saved. Groovy script that populates Tasks on the Task tab in the Edit Change Order page.	Yes
Attributes made mandatory using Page Composer on General Information page.	Configured action created using Application Composer on General Information page.	Yes
Add hyperlinks to internal sites like confluence.	Configured button to perform **Save As** of certain aspects like General Information or Affected Object of the current object.	Yes

Note: Apart from the combinatorial scenarios stated in the tables, you must use only one of these tools to configure buttons, actions, or tabs. Otherwise, your changes may not be retained or properly upgraded during release or patch updates.If you've used Page Composer to configure buttons, actions, or tabs, avoid using Application Composer to make further changes to these, including adding expressions. Likewise, if you've modified buttons, actions, and tabs using Application Composer, don't use Page Composer to make additional configurations.

2 Extend Change Management

Change Orders

Change Orders let you process changes to user-defined item attributes, structures, packs, associations, and item revisions.

Product data stewards and product managers can manage product change orders. They can create change orders within predefined change types, author product changes, view product changes, submit changes for review and approval, track change statuses, and implement changes on a scheduled date.

You can configure certain aspects of change orders within Application Composer in accordance with your business requirement.

> **Note:** In Application Composer, you can configure problem reports and corrective actions like you configure change orders.

What You Can Configure for Change Orders

Here are the configurations that are applicable for change orders.

- Subtabs that include context links and relationships
- Show or Hide Tabs
- Buttons or Actions
- URL Tabs
- First-level Objects

> **Note:** The Create Child Object button is no longer available for the change order object. However, you can create a one-to-many relationship for your configured object from the Relationships node in Application Composer. Child objects that you've created prior to update 23D are available, and you can edit the attributes of such objects, although usage isn't recommended.

Configure Change Orders

You can configure the change order objects by adding new fields in the Setup and Maintenance work area.

Define new subtabs within the Application Composer to create object relationships on page layouts. Write application logic, such as triggers, validation rules, and object functions that you can use in change order objects.

Let's see how we can configure change orders.

- Set up fields in the Setup and Maintenance work area

- Create new page layouts
- Create new tabs like related objects, context links, and mashup content
- Create actions and links
- Create automations with scripting
- Work with global functions

Related Topics

- Watch Video: Global Functions
- Assign Conditions to Page Layouts
- Use Field Values to Control a Page Display
- Use Advanced Expressions to Control a Page Display
- Control a Page Display Based on a User's Role
- Define Objects
- Related Object Subtabs

Set Up Fields for Change Orders

You can configure new fields for change orders in the Setup and Maintenance work area.

Set Up Fields

You can configure change orders (and other objects that are derived from a change type, such as change requests, problem reports, and corrective actions) to a certain extent using Application Composer. For example, you can define new subtabs to create object relationships on page layouts, or write application logic such as triggers, validation rules and object functions to be used in the change objects. But if you want to configure change attributes or fields, you must set up extensible flexfields from the Setup and Maintenance work area.

Here are the basic configuration steps:

1. To create change order attributes, in the Setup and Maintenance work area, do the following:
    - Offering: Product Management
    - Functional Area: Change Orders
    - Task: Manage Change Order and New Item Request Header Descriptive Flexfields

    Note:
    - Deploy descriptive flexfields before you create a sandbox for working in Application Composer.
    - Attributes can be either Global Segments - applicable to all change types, or Context Segments - applicable to a specific change type.
2. Assign attributes to a change type.
    - To assign attributes to a specific change type, enter the internal name of the change type as the context code of the segment/attribute context group and then add new attributes to the group.

      Attributes can be of the following type:
        - Character (Text)

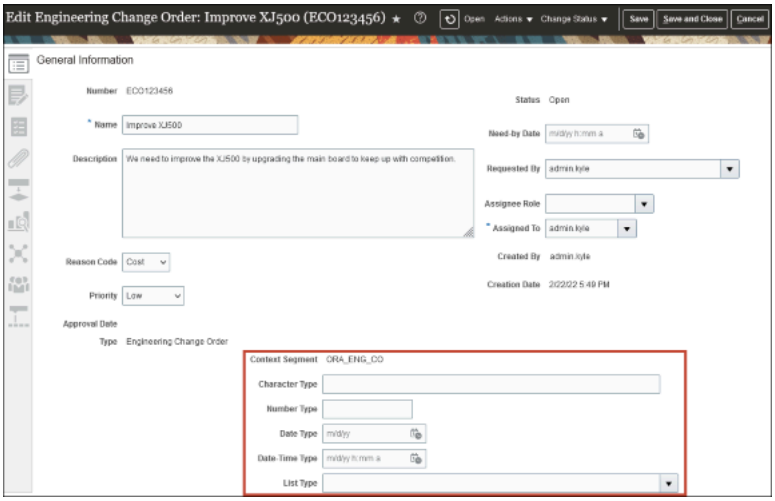
- Number
- Date
- Date Time
- List of Values (Character or Number)

Set these attribute segments as required. You can also set a date range.

Note:

- You must deploy the attribute segments and groups if you want to make them available for use.
- Attributes are assigned to the change order General Information page when they're deployed.
- Use the Sequence property in the setup task to define the display order. All the global attributes are displayed first, followed by the display of contextual attributes.
- In the setup task, you can disable an attribute so that it will no longer display.

This image displays a change object that has admin-created fields.



Create New Page Layouts

Use Application Composer to modify or create new page layouts. You can make changes that some, or all, of your end users can see, depending on the conditions you set.

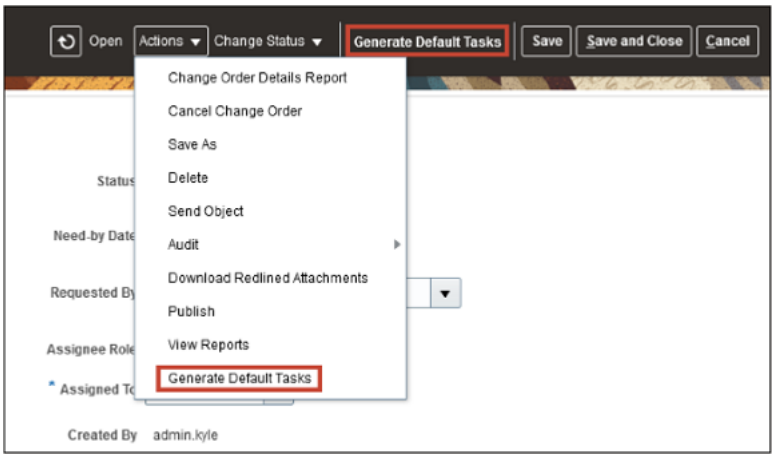
Create Page Layouts

- In the Page Layouts node, you can create multiple page layouts for change orders. The multiple page layouts can be assigned based on conditions such as when a user selects a specific value or when a change order has reached a specific status or if the user has a specific role assigned.

For example, if you created a page layout and entered the following Advanced Expression to that layout: StatusName == "Open", then that layout would only appear when a change order entered the Open status.

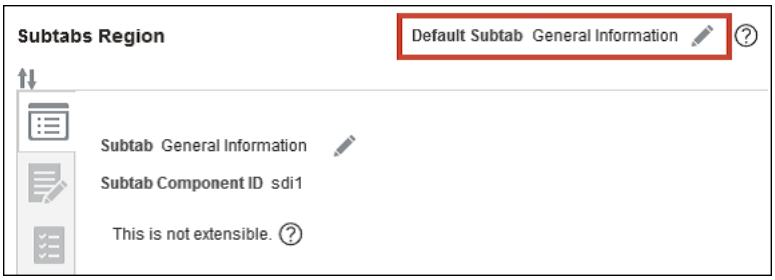
- In addition, you can add action buttons and/or Action menu selections to a page layout that allows the user to run Groovy scripts defined by an administrator in Application Composer.

  This image displays a change order with a configured action button.



- Within a page layout, you can add new tabs with admin-defined content to fit your business process.
- You can also hide tabs and change the tab order within a page layout, but first you must set the Default Subtab, or you will receive an error. This feature enables you to configure what tabs users see, based on the conditions defined in the page layouts.

  This image displays the Subtabs Region of Application Composer



Create Tabs

Use Application Composer to create extra tabs within a page layout.

Admin-Defined Tabs

Use the tabs to display either a related object, a context link, or mashup content.

Note: You can't create child objects for change orders. The Create Child Object button is no longer available for change order objects in Application Composer.

Create Related Object

To assign a related object, first create a new object in the Custom Objects node. After you create a new object, relate it to the change order object.

Note: You can't create child objects for change orders. The **Create Child Object** button is no longer available for the change order object. However, you can create a one-to-many relationship for your configured object from the Relationships node in Application Composer. Child objects that you've created before Update 23D are available, and you can edit the attributes of such objects, although usage isn't recommended.

Here's how:

1. Select the **Relationships** node in the Common Setup pane to create a new relationship.
2. In the Create Relationship page, set **Change Order** as the Source Object.
3. Set the **Custom Object** as the Target Object.
4. Set the Cardinality to **1: M**.
5. Click **Save and Close**.

After defining the relationship between the new object and the change order object, add the object as a new tab on change orders. Here's how:

1. Navigate to **Standard Objects** > **Change Orders** > **Pages**. Select the page layout that you've created.
2. In Details Layout <page_name>, in the Subtabs Region, click **Add** icon to create a new tab.
3. Select **Related Object** and click **Next**.
4. Select the related object as the Data Object.
5. Click **Save and Close**.

Create Context Link

Here's how you create a context link:

1. Navigate to **Standard Objects** > **Change Orders** > **Pages**. Select the page layout that you've created.
2. In Details Layout <page_name>, in the Subtabs Region, click the **Add** tab icon to create a new tab.
3. Select **Context link** and click **Next**.
4. Select the Data Object.
5. Define the search criteria.
6. Select fields for the summary table.
7. Click **Save and Close**.

Assign Mashup Content

Here's how you assign mashup content:

1. Select the **Mashup Content** node in the Common Setup pane to register a web application.
2. On the Register Web Application page, enter the URL of the page you want to display on your new tab.
3. Navigate to **Standard Objects** > **Change Orders** > **Pages**. Select the page layout that you've created.
4. In Details Layout <page_name>, in the Subtabs Region, click the **Add** tab icon to create a new tab.
5. Select **Mashup Content** and click **Next** to add the new tab to the change order page.

6. Click the button next to your web application.

   The web application page is rendered within your new tab.
7. Click **Insert**.
8. Click **Save and Close**.

Create Actions and Links

You can define an action or link for an object, and use Application Composer's work area configuration pages to add that action or link to a user interface page, such as a list page or details page.

Create Actions and Links

Here's how you add actions and links to a change order:

1. In the Change Order node, click **Actions and Links**.
2. In the Change Order: Actions and Links page, click the **Create** button.
3. In the Change Order: Create Action or Link page, enter the Display Label, and Name.
4. Select **Action** as the type.

   > **Note:** Links aren't supported for change orders.
5. Select Script as the Source.
6. Enter the script and choose when the script has to be run.
   - Actions must be assigned a script from the Object Functions tab found in the Server Scripts node.
   - After creating an action, you must add it to a page layout.

     Here's how you add the action to a page layout:
     i. In the Pages node, open a page layout and click the edit icon next to the Actions menu.
     ii. Add the action either to the Actions menu or as an independent button.

Create Automation with Scripting

You write Groovy scripts using Application Composer's expression builder.

Server Scripts

In the Server Scripts node, you can use Groovy scripting and web service calls to automate actions that fit your business process. Within this node you can create Validations, Triggers, or Object Functions.

Validations

- Use validations to define a condition. If the condition isn't met, ensure that you create an error message that guides the user to the correct action.
- Validations can be either Object Rules or Field Rules.
- An object rule is invoked when the user clicks **Save**. It then checks the condition to determine if the error message should be displayed.

- A field rule is invoked when the value of the selected field changes. It then checks the condition to determine if the error message should be displayed.

Triggers

Triggers are scripts that you can write to complement the default processing logic for a standard or configured object. You create triggers from the Server Scripts node in Application Composer.

Triggers update a change order based on the defined condition.

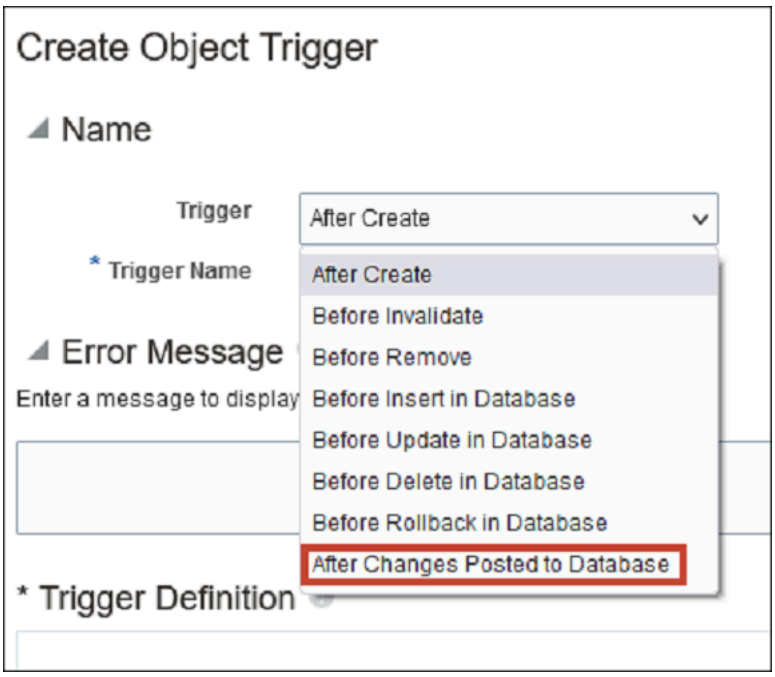
Triggers are of two types:

- Object Triggers
- Field Trigger

Object Triggers

An Object Trigger is invoked when you click **Save** or **Save and Close**. The only supported object trigger for change orders is **After Changes Posted to Database**.

This image displays an object trigger



Field Triggers

A field trigger is invoked when the value of the selected field changes.

Example

Let's say you want to automatically set the Priority of the change order to Urgent when the Reason Code is set to Error. Create a trigger that runs the following Groovy script:

```
Copyif (ReasonCode == "ERROR"){ setAttribute("PriorityCode", 'URGENT') }
```

Note: The internal names of Reason Code and Priority Code are used in the code. The values associated with ReasonCode and PriorityCode are the Lookup Code for each value. You can find these values in the Manage Change Reasons and Manage Change Priorities setup tasks in Setup and Maintenance task.

Note:

- Avoid parallel updates to change header attributes by triggers and global functions. For example, if your object-level or field-level triggers attempt to update header attributes while web service calls (invoked through global functions as part of entry-exit criteria) also attempt to update these attributes, modifications made by the triggers may not get saved because the web service calls would have bumped up the object version.
- The content in this topic for Triggers is also applicable for these objects: quality issues, and actions, ideas, concepts, requirements, and proposals.

.

Object Functions

Object functions are used by Actions as discussed in the Actions and Links section.

Note: Only change order actions and Application Composer triggers that are described in these sections are supported.

Work with Descriptive Flexfields and Scripting

Groovy scripting provides access to global descriptive flexfields and contextual descriptive flexfields. Global descriptive flexfields display on all change types while contextual descriptive flexfields only display on the change type to which they're assigned.

Work with Global Descriptive Flexfields

To work with global descriptive flexfields, use the ChangeObjectDFF accessor to access the collection of descriptive flexfields. In the descriptive flexfield collection, use the internal name of the attribute to get it and set it.

In the following example, the value of the global descriptive flexfield Implementation Reason is copied to the global descriptive flexfield PCN.

Let's look at the format for Object Functions and Field Triggers:

```
def changeDFF = ChangeObjectDFF
def reason = changeDFF.implementationReason
changeDFF.setAttribute('pcn', reason)
```

Let's look at the format for Object Triggers:

```
def changeDFF = ChangeObjectDFF
def reason = changeDFF.implementationReason
if (reason != changeDFF.getAttribute('pcn'))
 {
 changeDFF.setAttribute('pcn', reason)
```

```
}
```

Work with Contextual Descriptive Flexfields

Use the ChangeObjectDFF accessor to access the descriptive flexfield collection like global descriptive flexfields. However, with contextual descriptive flexfields, assign the API name (all lower case) to a variable, and use the variable to get or set the attribute.

In the following example, the value of the global descriptive flexfield Implementation Reason is copied to the contextual descriptive flexfield Justification.

Let's look at the format for Object Functions and Field Triggers:

```
def apiName = 'justification'
def changeDFF = ChangeObjectDFF
def reason = changeDFF.implementationReason
changeDFF.setAttribute(apiName, reason)
```

Let's look at the format for Object Triggers:

```
def apiName = 'justification'
def changeDFF = ChangeObjectDFF
def reason = changeDFF.implementationReason
if(reason != changeDFF.getAttribute(apiName))
 {
 changeDFF.setAttribute(apiName, reason)
 }
```

Work with Global Functions

Global functions enable you to:

- Create reusable code that you can reference in other scripts.
- Invoke a script from a product rule using InvokeGlobalFunction.

Since change orders are available in Application Composer, you can create triggers and validations for conditionally invoking scripts in most cases. However, invoking a script based on status change can only be accomplished through a global function being invoked by a product rule triggered as an entry or exit criteria.

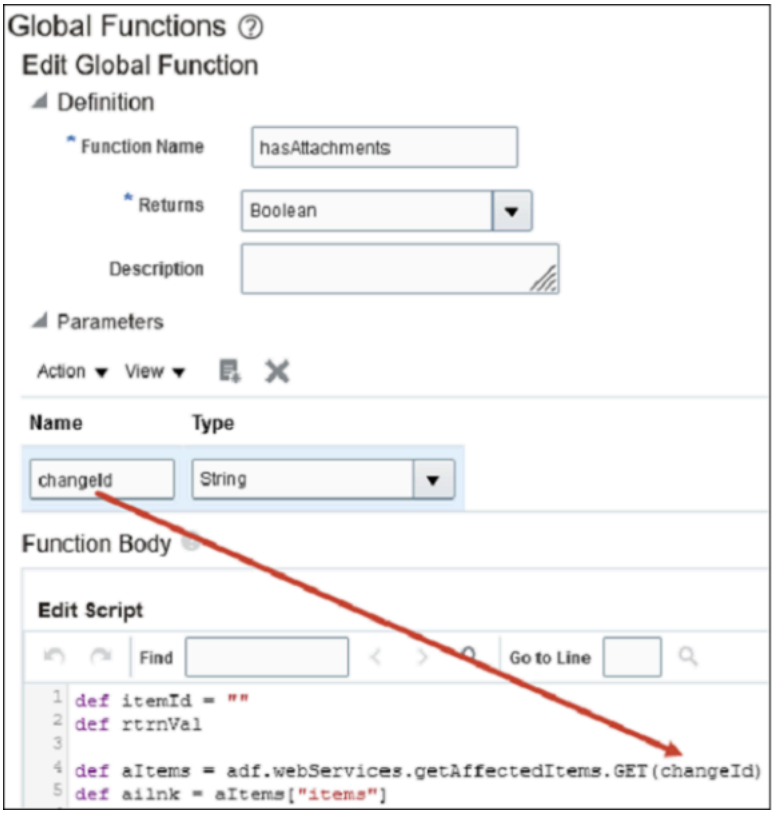
Items aren't available in Application Composer. Therefore, you can only apply scripts to items as global functions that are invoked through a product rule.

Since global functions have no current object context, you must consider the following points while working with global functions:

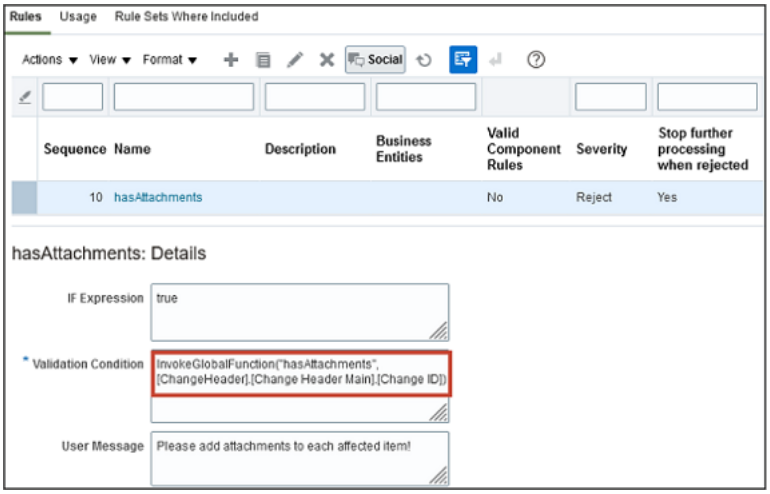
- You won't be able to use Groovy functions to perform operations in your script.
- Global functions call REST web services to accomplish what’s needed.
- To write global functions that work on a particular object, and call them from an object function, refer to Passing the Current Object to a Global Function.

- To pass the current object context when invoking a global function from a product rule, create a parameter in the global function that receives the object ID from the rule.

  This image displays the Global Functions in Application Composer.



This image displays the validation condition that shows the global function



Groovy Script Examples

Application Composer supports Groovy scripting, which is a standard, dynamic scripting language for the Java platform. This section provides an example of how to use the setAttribute() function and the getAttribute() function with one or more lines of Groovy code.

Set an Attribute Value

To set the value of an attribute, use the setAttribute() function. The following example sets the Priority field to "High" on a change order using the PriorityCode attribute.

```
setAttribute("PriorityCode", "HIGH")
```

To find the code names for the priority values, in the Setup and Maintenance work area, go to the following:

- Offering: Product Management
- Functional Area: Change Orders
- Task: Manage Change Priorities

Get the Lookup Code for the value that you want to assign.

Get an Attribute Value

To get the value of an attribute, either use the getAttribute() function or just use the API name of the attribute.

Each of these scripts does the same thing, they return the value in the Priority field on a change order to Runtime Messages:

```
def getPriority = getAttribute("Priority") println(getPriority)

def priority = Priority println(priority)

println(Priority)
```

The command println() prints the value in the parenthesis to the **Runtime Messages** log in Application Composer in the Common Setup pane. To troubleshoot a script, it's very important to enable **Runtime Messages** in Application Composer.

Let's see how to enable **Runtime Messages**:

1. In the Common Setup pane, select **Runtime Messages**.
2. In the Application Script Log page, select the check box **Enable My Application Script Logging**.

Related Topics

- Global Functions
- Defining Utility Code in a Global Function
- Item Rule Object Functions
- Passing the Current Object to a Global Function

Work with Collections

Subobjects such as Tasks and Attachments are organized into collections. To work with attributes in a collection, use the accessor and iterate through the records or rows in the collection.

Let's look at how to work with collections.

- Get task attributes
- Set task attributes
- Add a task to change order

Get Task Attributes

Tasks are stored as a collection and therefore to access them you must iterate through them. In this case, the accessor is Change Task.

```
def changeTasks = ChangeTask
changeTasks.reset()
while(changeTasks.hasNext()){
def eachTask = changeTasks.next()
println("ActionCode: "+eachTask.ActionCode)
println("Assigned By: "+eachTask.AssignedByText)
println("Assigned On: "+eachTask.AssignedDate)
}
```

Set Task Attributes

You can update some task attributes by iterating through them, using an "if" statement to access the desired task, and then using set Attribute to set a value.

Let’s look at the code to set task attributes:

```
def changeTasks = ChangeTask
changeTasks.reset()
while(changeTasks.hasNext()){
def eachTask = changeTasks.next()
if(eachTask.Name == "Update tasks"){
eachTask.setAttribute("Description",'This is another new description')
println("Description: "+eachTask.Description)
}
}
```

Add a Task to Change Order

Let's look at the code to add a task to a change order:

```
def changeTasks = ChangeTask
def newTask = changeTasks.createRow()
newTask.setAttribute("Name", "Update the design doc")
newTask.setAttribute("RequiredFlag", "N")
newTask.setAttribute("CompleteBeforeStatusCode", 6)
newTask.setAttribute("Description", "Update the design doc to fit the proposed changes.")
newTask.setAttribute("AssignedTo", "<username>")
changeTasks.insertRow(newTask)
```

3 Extend Quality Management

Quality Management

Quality Management is performed using four different object types. These include problem reports, corrective actions, quality issues, and quality actions.

Note: To configure problem reports and corrective actions, use the information provided in change orders.

What You Can Configure for Quality Management

Here are the configurations that are applicable for quality management objects.

- Additional Attribute Types, which include:
  - Dynamic Choice List
  - Fixed Choice List
- Check box
- Text
- Number
- Currency
- Percentage
- Datetime
- Time
- Long Text
- Formula
- Record Type
- Page Layout
- Subtabs that include context links and relationships
- Show or Hide Tabs
- Buttons or Actions
- URL Tabs
- First-level Objects
- Child Objects

Configure Quality Issues and Quality Actions

You can configure fields and page layouts for quality management objects in Application Composer.

Fields

You can create and modify all extra fields for quality issues and actions in Application Composer. The fields aren't visible to users until they're added to a page layout. Fields can be used through Groovy scripting and REST web services without adding it to a page layout. You can create the following type of fields for both issues and actions:

- Text: allows users to enter 254 characters of simple text.

  > **Note:** The 1500-character limit applies if the characters are single byte. If the characters are multibyte, such as Japanese or Chinese, then the maximum character limit is 1500 characters divided by the number of bytes per multibyte character. For example, if characters are 2 bytes, then the name is limited to a maximum of 750 characters. If a mix of characters is used, then the maximum sum of character bytes that's supported is 1500.

- Number: allows users to enter a number, including decimals.
- Date: allows users to select a date.
- Long Text: allows users to enter 32,000 characters.

  > **Note:** It's recommended that you don't exceed five long text fields on an object for performance reasons. Long text isn't supported by Audit History.

- Check box: provides a check box to select or deselect (true/false).
- Percentage: a numeric field that displays as a percentage on the page. The percentage field will take the entered value and store it as a percentage of 100. For example, if you enter 95, the value is stored as .95 although it displays 95%.
- Datetime: allows users to select a date and time.
- Record Type: invokes page layout to change based on value selection.
- Choice List (Fixed): allows the administrator to create a list of values from which users can select.
- Choice List (Dynamic): allows users to select from a list of other objects such as ideas or requirements.
- Formula: allows the administrator to prepare a calculation based on a formula and data from other fields.

Page Layouts

Here are some of the points that you must consider for page layouts:

- Fields for quality issues and quality actions are made available on the user interface through page layouts. Use the Pages node to create multiple page layouts that can apply to specific scenarios based on criteria that you define.
- The Standard page layout provides a basic display but can't be modified.
- Page layouts are displayed based on two factors: order and criteria.
  - The order of display of page layouts is based on how they're organized in the application.
  - You can change the order of the layouts to suit your purposes.
  - If you've defined the criteria for display of the page layout, based on the current user session, the page layouts are scanned from the beginning and the page layout is displayed based on the criteria.
  - Sometimes, multiple criteria may apply for the display of page layouts. In such cases, the page layout display is based on the defined criteria, and the order in which the page layout is organized. Usually, the first page layout that matches at least one criterion is displayed.
  - Page layout criteria come in the form of: Type, Role, and Advanced Expression.

- Type is determined by a Record Type field added to the page layout. You can set the page layout display based on a particular value in a Record Type field.
- Role is the assignment of one or more roles to the display of a page layout. If the user with a particular role is viewing the object, the page layout corresponding to that role is displayed.
- Advanced Expression provides you with the ability to create a Groovy script that will determine the page layout display. For example, in the Advanced Expression, if the Severity is set to High then that corresponding page layout is displayed.

  Example: Severity == "ORA_HIGH"

> **Note:** It's a best practice to create a copy of the standard page and leave it as a default layout. All page layouts created using Application Composer must be duplicates of the standard template not a copy of another configured page layout.

Custom Objects

New objects you create and configure using Application Composer are located in the Custom Objects node. You can create top-level objects (objects without a parent) or child objects (objects created in the context of a parent).

> **Note:** If you create a subtab using Application Composer, and expose the configured objects:
> - The Save As functionality won't save these objects.
> - The configured child objects aren't supported.

Important Considerations

Let's look at some of the considerations applicable to quality management objects:

- You can’t use triggers on status attributes. But you can use entry and exit criteria to trigger an action based on change in the status, using global functions.
- You can use entry or exit criteria for validation use cases only. Assignment use cases aren't supported. Assignment use cases include any operation that results in adding or modifying the attribute values, or the content in any of the tabs.
- You can't use the old value, new value functions for tracking the status.
- In case you are using a workaround for storing the status code of an attribute, and using triggers based on changes to this attribute, you can do so between two open statuses only.

  It's not supported for the following status changes:

  - Open to Interim
  - Interim to Interim
  - Interim to Approval
  - Approval to Complete

Related Topics

- Watch Video: Use a Trigger to Preset Questions in the Description

4 Extend Innovation Management

Innovation Management

Innovation Management helps you capture ideas from any source for new products, services, markets, or customer experiences. Each proposal is evaluated for value, cost, and constraints.

Innovation Management documents, prioritizes, and agrees on requirements that can be leveraged in developing innovation concepts. Reuse existing items, trace requirements through design, and validate that each has been met to reduce new product introduction risks.

Use Application Composer to extend ideas, concepts, proposals, and requirements. Use Application Composer to develop attributes for each requested idea.

You can use Application Composer to extend Innovation Management objects in the following ways:

- Add standard and configured attributes to the pages.
- Leverage logical expressions to control the visibility of the configured page layouts.
- Use Groovy scripting to validate either a field or an object, automatically execute an action when a specific event is triggered and write object functions that can be reused in multiple contexts.
- Further, use Groovy scripting to update attributes between standard and configured objects; between two configured objects; and between two standard objects.
- Use Groovy scripting to validate through either a field or an object rule, and automatically execute an action when a specific event is triggered. These Server Scripts can be defined for both standard and configured objects.
- Provide separate Create, View, and Manage access with granular privileges.

What You Can Configure for Innovation Management Objects

Here are the configurations that are applicable for Innovation Management objects.

- Additional Attribute Types, which include:
  - Dynamic Choice List: A dynamic choice list is a field that contains a list of values, which are populated from the actual data of another object.

    > **Note:** From a cross-application perspective, you can create a dynamic choice list field for an object in one web application that's populated by the records from a supported object in another web application.

  - Fixed Choice List: A fixed choice list is a field that contains a list of static values that are populated from the lookup types. At run time, you can select one or more values from this field, depending on the field's definition.

    You can create cascading lists of values that are narrowed down based on user selection. For example, you can create a list of countries, states, or counties, with each list displaying values based on the value selected in the parent field. If the state selected is Oregon, the list of counties displayed can be limited to the counties within that state.

- Check box: A check box field is a field where, in the runtime application you can select a check box, indicating a true or false attribute of a record. The available operators are Equal to, Greater than, Greater than or equal to, Is blank, Is not blank, Less than, Less than or equal to, and Not equal to.
- Text: A text field is a field where, in the runtime application, you can enter a combination of letters, numbers, or symbols.
- Number: A number field is a field where you can enter a number during runtime.
- Currency: A currency field is a field where you can enter a currency amount.
- Percentage: A percentage field is a field where you can enter a percentage. The Application Composer automatically adds the percent sign to the number.
- Datetime: A datetime field is a field where you can enter a date, or select a date from a calendar, and enter a time of day. You can show the date or time, or both.
- Date: A date field is a field where you can enter a date, or select a date from a calendar. This type of field has no time component.
- Long Text: A long text field is a field where, in the runtime application, you can enter a combination of letters, numbers, or symbols. This field type supports 32,000 characters. Long text isn't supported by Audit History.

  > **Note:** Don't use more than five long text fields in an object for performance reasons.

- Formula: A formula field is a field that's calculated in the runtime application using the Groovy-based expression included in the formula field's definition.
- Record Type: A record type field contains a list of static values that are populated from lookup types. Associating a choice list value with a role means that only users with that role can see the choice list value. Associating a choice list value with a page layout means that after that value is selected at runtime, the associated page layout is displayed.
- Page Layout: Modify the pages on which an object appears.
- Subtabs that include context links and relationships: Add subtabs to the standard or configured object's details page to display details that are related to the current object but derived from another object, or from another source. You can specify the source of subtab data.
- Buttons or Actions: Add buttons or links to desktop pages.
- Related Objects: Related objects are connected in a foreign key-based relationship between two objects.
- Custom Objects: Custom Objects are objects that are created using Application Composer. You can create either top-level objects (objects without a parent) or child objects (objects created in the context of a parent).

  > **Note:** If you create a subtab using Application Composer, and expose the configured objects:
  >
  > - The Save As functionality won't save these objects.
  > - The configured child objects aren't supported.

Related Topics

- Triggers in Create Automation with Scripting
- Best Practices for Long Text Fields
- Watch Video: Create Cascading Select Lists

Configure Ideas

You can extend ideas by adding new fields or attributes, and creating page layouts.

Add actions and links to the already existing page layouts, or add new tabs to the existing page layouts to extend the Idea object. Another configuration is adding new fields to an existing object (standard objects) or create entirely new objects (custom objects).

Standard Objects

Standard objects come with tabs, and each of the standard objects lets you create and add multiple instances of the child object to one instance of the parent. You can add the child object as a tab to the parent object.

Custom Objects

Custom Objects are objects that you create using the Application Composer. You can create either top-level objects (objects without a parent) or child objects (objects created in the context of a parent).

Let's see some of the attributes of a child object:

- Display Label - node name in Application Composer.
- Plural Label - page title for the object's work area and the icon name in the Navigator.
- Record Name Label - name of the field where you enter the record name.
- Record Name Data Type - This can either be user-entered text or an autogenerated sequence to identify a record.
  - Use "Text" for users to enter their own names for each record.
  - Use "Automatically Generated Sequence" to automatically create the name for each record.
- Object Name - internal name of the object.
- Child Collection Name - This is automatically created and used to assign a generic function across the collection or list.
- Security - define who can use the child object. In the Security node, set up which roles have the privilege to use the object.

> **Note:** If you create a subtab using Application Composer, and expose Custom Objects or configured child objects:
>
> - The Save As functionality won't save the Custom Objects.
> - The configured child objects aren't supported.

5 FAQs

What job role must I have to create my own objects in Application Composer?

Users with any one of the following three job roles can create custom objects and use all other Application Composer functions:

- Customer Relationship Management Application Administrator.
- Application Implementation Consultant.
- Master Data Management Application Administrator.

In Oracle CX Sales, provision the user with the Customer Relationship Management Application Administrator job role (for performing the configurations) and the Custom Objects Administration job role and Sales Administrator job role (for testing the configurations).

What's the difference between fixed choice lists and dynamic choice lists?

A fixed choice list and a dynamic choice list are similar in that the ultimate goal of both types of choice lists is to generate a field with a list of values.

For a fixed choice list, the field's specific list values are populated from a lookup type that you select when you define the field. The list displays in a single column and doesn't change.

A dynamic choice list, meanwhile, is populated from an existing object's actual data, which you can add filters to. Based on how you define the field, the list is dynamically populated at runtime and its values can change depending on the user's context. In addition, you can add more columns to the dynamic choice list field to assist your users in making a selection at runtime.

What Application Composer tasks are available only within a sandbox?

Most Application Composer tasks require you to be in a sandbox. For example, these menu items are available to you only if you're in an active sandbox session.

- Objects
    - Custom Objects

    - Standard Objects
- Common Setup
    - Relationships
    - Role Security
    - Object Workflows
    - Global Functions
    - Run Time Messages
    - Mobile Application Setup
    - Outlook Setup
    - Personalization
    - Web Services
    - Metadata Manager

These menu items are the exceptions. They're available only in a sandbox-free session.

- Custom Subject Areas
- E-Mail Templates
- Import and Export
- Business Processes

Can two objects have the same record number?

Yes, two objects can have the same record number. The record number is unique only within a configured object.

How frequently can I publish a sandbox?

Integration sandboxes are typically published once a week. Publishing integration sandboxes less frequently than once a week isn't recommended.

When you publish an integration sandbox, all private sandboxes are invalid because the label in the mainline metadata application has changed. If you made changes to private sandboxes that you want to retain, then document those changes and then delete all the private sandboxes.

When do I publish a sandbox?

You can publish a sandbox after you've tested and verified that the application changes done in that sandbox are ready to be moved to the mainline metadata.

You must test the following configurations outside a sandbox:

- Import/Export
- Web services
- Custom subject area creation
- Object workflow
- E-mail templates

Can I delete a sandbox?

Yes. You can delete sandboxes. However, you can delete only those that aren't published.

Before you delete a sandbox, you must first confirm that the sandbox isn't active.

CAUTION: Deletion of partial content of a sandbox is risky. It's recommended that you don't use this option.

After you've tested your application changes, you must move those changes to the integration sandbox. Publish your integration sandbox and then delete all the test-only sandboxes. You can then create and work in new sandboxes, including a new integration sandbox.

Can I delete unused configured attributes?

Yes, you can delete any unwanted or unused configured attributes in a sandbox. You can only delete the configured attributes that have never been published.

Here's how you delete the configured attribute:

1. Navigate to Application Composer work area.

   Note: Ensure that you're in a sandbox.

2. Select the **ERP and SCM Cloud** option from the Application list.
3. Select the **Quality** check box in the Object tags.
4. Expand **Standard Objects** > **Quality Issue** > **Fields**.
5. Select the configured attribute and click the **Delete** button.

   In case the attribute is still in use, a message appears restricting you from deleting it. If the attribute is in use, review and remove all usages before you try again.

   The unused configured attribute is deleted.