# Contents

## Preface

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>About This Guide</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Audience and Scope</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Related Guides</td>
<td>1</td>
</tr>
</tbody>
</table>

## 1 About This Guide

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

## 2 Customizing Industry Solutions

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Oracle Sales Cloud Industry Solutions: Overview</td>
<td>3</td>
</tr>
<tr>
<td>Customizing Industry Solutions: Explained</td>
<td>3</td>
</tr>
<tr>
<td>Creating a Sandbox for Industry Solution Changes: Explained</td>
<td>4</td>
</tr>
</tbody>
</table>

## 3 Customizing Industry Solution Objects and Pages

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Using Application Composer to Change Industry Solutions: Overview</td>
<td>7</td>
</tr>
<tr>
<td>Customizing Industry Solution Objects: Explained</td>
<td>8</td>
</tr>
<tr>
<td>Customizing Industry Solution Fields: Explained</td>
<td>12</td>
</tr>
<tr>
<td>Actions and Links for Industry Solutions: Explained</td>
<td>14</td>
</tr>
<tr>
<td>Configuring Security for Industry Solutions: Explained</td>
<td>20</td>
</tr>
<tr>
<td>Scripting Against Industry Solution Objects: Explained</td>
<td>24</td>
</tr>
<tr>
<td>Using Runtime Messages: Explained</td>
<td>26</td>
</tr>
<tr>
<td>Customizing Industry Solution Page Layouts: Explained</td>
<td>27</td>
</tr>
</tbody>
</table>

## 4 Customizing Industry Solution Pages Using Other Tools

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>31</td>
</tr>
<tr>
<td>Using Page Composer to Change Industry Solutions: Overview</td>
<td>31</td>
</tr>
<tr>
<td>Making Saved Searches Available to All Users: Procedure</td>
<td>32</td>
</tr>
<tr>
<td>Bulk Text Customizations: Explained</td>
<td>33</td>
</tr>
<tr>
<td>Changing Field Display Labels: Explained</td>
<td>36</td>
</tr>
</tbody>
</table>

## 5 Customizing Analytics

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>39</td>
</tr>
<tr>
<td>Understanding Analytics for Industry Solutions: Overview</td>
<td>39</td>
</tr>
<tr>
<td>Customizing Analytics for Industry Solutions: Explained</td>
<td>39</td>
</tr>
</tbody>
</table>
6  Creating Object Workflows  47
   Automating Processes with Object Workflows: Explained  47

7  Oracle Sales Cloud Web Services  51
   Web Services for Industry Solutions: Overview  51
   Calling Web Services from Groovy Scripts: Explained  52
   Working with Industry Solution Objects Using SOAP Web Services: Worked Example  55

8  Publishing Your Changes  59
   Publishing Your Industry Solution Customizations: Explained  59
Preface

This preface introduces information sources that can help you use the application.

Oracle Applications Help

Use the help icon to access Oracle Applications Help in the application. If you don’t see any help icons on your page, click the Show Help icon in the global header. Not all pages have help icons. You can also access Oracle Applications Help at https://fusionhelp.oracle.com.

Using Applications Help

Watch: This video tutorial shows you how to find help and use help features.

Additional Resources

- Community: Use Oracle Applications Customer Connect to get information from experts at Oracle, the partner community, and other users.
- Guides and Videos: Go to the Oracle Help Center to find guides and videos.
- Training: Take courses on Oracle Cloud from Oracle University.

Documentation Accessibility

For information about Oracle’s commitment to accessibility, see the Oracle Accessibility Program.

Comments and Suggestions

Please give us feedback about Oracle Applications Help and guides! You can send e-mail to: oracle_fusion_applications_help_ww_grp@oracle.com.
1 About This Guide

Audience and Scope

This guide describes how sales administrators and implementors can customize Oracle Sales Cloud industry solutions, such as Oracle Sales Cloud for Consumer Goods. Read this document to understand common customization tasks, regardless of which industry solution is installed.

For more information on how to set up, implement, and use Oracle Sales Cloud industry solutions, refer to your industry solution-specific document on My Oracle Support, or at http://docs.oracle.com/cloud/.

Related Guides

You can refer to the following guides to learn more about customizing Oracle Sales Cloud industry solutions.

<table>
<thead>
<tr>
<th>Guide</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Sales Cloud Getting Started with Oracle Sales Cloud Customizations</td>
<td>Introduces you to user interface elements, user interface types, and simple, common customizations of Oracle Sales Cloud.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Getting Started with Communications Implementation</td>
<td>Describes how to set up your instance of Oracle Sales Cloud for Communications in conjunction with the activities that you perform to integrate Oracle Sales Cloud with Siebel CRM.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Getting Started with Consumer Goods Implementation</td>
<td>Describes how to set up your instance of Oracle Sales Cloud for Consumer Goods in conjunction with the activities that you perform to set up Oracle Sales Cloud.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Getting Started with Financial Services Implementation</td>
<td>Describes how to set up your instance of Oracle Sales Cloud for Financial Services in conjunction with the activities that you perform to set up Oracle Sales Cloud.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Getting Started with High Tech and Manufacturing Implementation</td>
<td>Describes how to set up your instance of Oracle Sales Cloud for High Tech and Manufacturing in conjunction with the activities that you perform to set up Oracle Sales Cloud.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Customizing Sales</td>
<td>Describes how to create and extend objects and customize the user interfaces and navigation menus.</td>
</tr>
<tr>
<td>Oracle Sales Cloud Groovy Scripting Reference</td>
<td>Explains the basics of how to use the Groovy scripting language to enhance Oracle Sales Cloud.</td>
</tr>
<tr>
<td>Oracle Applications Cloud Customizing the Applications for Functional Administrators</td>
<td>Describes the tools and concepts for customizing and extending the applications.</td>
</tr>
</tbody>
</table>
Related Topics

- Oracle Help Center
2 Customizing Industry Solutions

Oracle Sales Cloud Industry Solutions: Overview

Oracle Sales Cloud offers industry solutions that provide industry-specific tasks on top of the base set of capabilities available in our sales applications.

The industry solutions available with Oracle Sales Cloud are:

- Communications
- Consumer Goods
- Financial Services
- High Tech and Manufacturing
- Higher Education

To learn how to use each application, consult your documentation that comes with your industry solution. Use this guide to learn how to use Oracle Sales Cloud's set of standard customization tools to make additional changes that you might require.

Customizing Industry Solutions: Explained

Use Oracle Sales Cloud’s configuration and customization toolset to make changes to your industry solution. An industry solution is a vertical application that’s built on top of Oracle Sales Cloud. The industry solution contains industry-specific business objects and user interface pages, plus all the underlying code to make it work with the base set of Sales Cloud applications. You can make virtually any change to an industry solution’s business objects or pages that you can make to a regular Sales Cloud application. Let’s look at the tools you can use to make your changes.

The toolset includes Application Composer, Page Composer, User Interface Text Editor, and BI Composer.

- Application Composer
  
  Application Composer is a browser-based tool that business analysts and administrators, not just programmers, can use to customize Oracle Sales Cloud. Using this tool, you can make the types of data model changes which previously could only be made by application developers. For example, you can create a new object and related fields and then create new user interface pages to expose that object to users.

  Use Application Composer to make the following changes to an industry solution:
  
  - Customize industry solution objects.
  - Customize industry solution fields.
  - Customize industry solution page layouts.
  - Create your own actions and links.
  - Configure the security policy for an industry solution object.
  - Create your own groovy scripts for an industry solution object.
  - Add new fields to an industry solution custom subject area for reporting purposes.
- View diagnostic messages generated by any script for assistance with troubleshooting.

- **BI Composer**

  The Oracle Business Intelligence (BI) Composer is a tool you can use to build reports. When building reports, you select a report subject area from within BI Composer. A report subject area is a set of entities, attributes, and measures that represent information about the areas of an organization's business. To build reports, use either the predefined report subject areas that are delivered for Oracle Sales Cloud, or create a custom subject area using a wizard available in Application Composer.

  Some custom subject areas are already provided by Oracle for some industry solution objects. Use Application Composer to make changes to industry solution custom subject areas, then use BI Composer to create or update reports.

- **Customize User Interface Text Tool**

  Use the Customize User Interface Text tool to simultaneously customize multiple occurrences of entire words or phrases in the user interface (UI). The Customize User Interface Text Tool can access the vast majority of strings across pages, including pages for your industry solution.

- **Page Composer**

  Page Composer is a page editor that you can use to easily edit the user interface composition at run time. In Oracle Sales Cloud, Page Composer is intended for simple user interface editing functions, such as showing and hiding regions, fields, and tables, changing the order of regions, or changing a dashboard page layout. You can also use Page Composer for adding or removing predefined content from the Resource Library. All changes are done and stored in the UI layer.

  Oracle Sales Cloud transactional pages and dashboards are enabled for run time customization using Page Composer. These pages and regions are delivered already enabled for page editing. Administration and setup pages are not Page Composer-enabled.

  When making changes to an industry solution, you can use Page Composer to make these kinds of changes:

  - Created saved searches for your users.
  - Show or hide fields.
  - Create changes for a specific user role.

---

**Creating a Sandbox for Industry Solution Changes: Explained**

To make customizations to your industry solution, you must always be in an active sandbox. You can either create a sandbox or select an existing sandbox, and then you designate it as an active sandbox. The active sandbox stores your changes and keeps them hidden from your end users while you are working on your customizations. After testing the sandbox, you can publish it and the changes are merged into the mainline metadata where your users can view all application updates.

⚠️ **Caution:** Do not import or delete metadata files. These operations modify sandbox contents and could cause issues with sandbox functionality.
To create a sandbox:

1. Click your user name in the global area, and select **Manage Sandboxes** from the **Administration** menu.
2. Click the New icon to create a sandbox.
3. Click **Save and Close**.
4. Select the new sandbox (or an existing one), and click **Set as Active**. The sandbox is designated as the active sandbox.
5. Close the Manage Sandboxes dialog box.
3 Customizing Industry Solution Objects and Pages

Using Application Composer to Change Industry Solutions: Overview

Use Application Composer to customize your industry solution. Application Composer is a browser-based tool that business analysts and administrators, not just programmers, can use to customize Oracle Sales Cloud. Using this tool, you can customize the business objects and other data model components that are provided with your industry solution. For example, add custom fields to an industry solution business object, or create a relationship between an industry solution object and another object. Once you complete your data model changes for your industry solution, you can also use Application Composer to expose those changes on user interface pages.

What Will You See?

Application Composer is organized with an object tree on the left side of the page. The tree contains a section for standard objects and custom objects. Once expanded, each object has nodes for fields, pages, actions and links, security, and other details. You use this tree to make changes to an application in Sales Cloud.

Typically, when using Application Composer for the first time, the custom objects tree in Application Composer is empty. There are no custom objects yet. An environment that has an industry solution installed, however, has special industry solution objects in the custom objects tree already predefined.

There might also be industry solution fields in the custom fields table of standard objects, such as Account or Contact. An industry solution might also provide industry-specific reporting, in which case you might also see one or more custom subject areas in Application Composer, available for customization.
What Can You Change?

Use Application Composer to make the following changes to an industry solution:

- Customize industry solution objects, which appear right alongside standard business objects in Application Composer’s object tree.
  Customization capabilities are the same as standard object customization capabilities. For example, add custom fields to an industry solution object. Or create a relationship between an industry solution object and another object.

- Customize industry solution fields. These fields can appear in either an industry solution object’s field list, or in a standard object’s custom fields list.
  For each field type (text, number, and so on), customization capabilities are the same as standard field customization capabilities.

- Customize industry solution page layouts. Industry solution page layouts are provided for the pages of industry solution objects. They are permanently inactive (deprecated).
  To customize these pages, you must first duplicate them to create new page layouts that you can then change.

- Create your own actions and links, and add them to industry solution page layouts.

  **Note:** Some actions and links are provided by Oracle for some industry solution objects. These actions and links are protected and you cannot configure them further.

- Configure the security policy for an industry solution object through the Security node of an industry solution object.
  To understand whether you need to adjust the security policy, refer to the documentation provided with your industry solution.

- Create your own groovy scripts for an industry solution object.

  **Note:** Some scripts are already provided by Oracle for some industry solution objects. These scripts, such as field and object triggers and validation rules, are protected and you cannot configure them further. Industry solution scripts run before your scripts run, on a per object, per field, and per script type basis. For example, your object After Create triggers will run after industry solution After Create triggers on the same object.

- Add new fields to an industry solution custom subject area, configure security, and create new reports.

- View diagnostic messages generated by any script through the Runtime Messages page. Use these diagnostic messages to assist with debugging your new scripts, as well as to debug the interaction between your scripts and industry solution scripts.

Customizing Industry Solution Objects: Explained

An industry solution is a vertical application that’s built on top of Oracle Sales Cloud. The industry solution contains industry-specific business objects and user interface pages, plus all the underlying code to make it work with the base set of Sales Cloud applications. In Application Composer, industry solution objects appear right alongside standard business objects in Application Composer’s object tree. You can make virtually any change to an industry solution’s business objects or pages.
that you can make to a regular Sales Cloud application. For example, add custom fields to an industry solution object. Or create a relationship between an industry solution object and another object.

Application Composer's Object Tree

Access Application Composer from a Sales Cloud application at run time by using the Navigator menu. The first view of Application Composer is the main Overview page, which is the entry point into all your customization options.

On the main Overview page, the regional pane at left displays the object tree, which lets you browse an application’s existing object configuration in a tree format. The object tree reflects the latest configuration of the application: both standard objects as well as custom objects. When an industry solution is installed, industry-specific business objects appear in the Custom Objects section of the tree.

To view industry solution objects:

1. Select Application Composer from the Navigator menu, under the Tools category.
2. On the main Overview page, select an application from the Application choice list, such as Sales.
3. Expand the Custom Objects node to view and edit industry solution objects.

For example, look at object details such as fields and UI pages where the object is exposed.

**Tip:** At the top of the object tree, you can also click the New icon to create a new custom object.

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.

Industry solution objects always appear on simplified user interface pages only.
Actions and links
   Add actions or links to pages.
Server scripts
   Write application logic that controls the behavior of an object’s records.
Security
   Implement functional and data-level security for an object and its records.

### Using the Object Overview Page
The Object Overview page provides a high-level overview of an object. The Object Overview page displays the primary attributes for an object, plus a list of child objects and related objects, if any.

To access an industry solution object’s Overview page:

1. Select an application from the Application choice list.
2. Select an industry solution object in the Custom Objects tree.

From the object’s Overview page, you can:

- Edit the object’s primary attributes, described in the previous section.
- View the parent child relationships that were created for this object.
  You can also create new child objects from this page, which implicitly creates a new parent child relationship.
- View the non-parent child relationships that were created for this object.

### Editing an Object’s Attributes
After an object has been created, you can edit its attributes from its Object Overview page.
To edit an object’s attributes:

1. Select an application on the main Overview page.
2. Select **Standard Objects** or **Custom Objects** in either the object tree or local area of the main Overview page.

   ✍ **Note:** Industry solution objects always appear in the Custom Objects section of the object tree.

3. From the resulting summary table, select an object and then select the Edit icon to navigate to its Object Overview page.
4. On the Object Overview page, click **Edit**:

   ![Edit Object: Promotion](image)

   - Change the object’s display label, plural label, and description at any time.
   - You cannot change the Object Name and API Name after the object has been created.

   A custom object’s API name is automatically derived using the logical name followed by _c. You use the API name in Groovy expressions that you build with the expression builder, when writing business logic for the object.

   💡 **Tip:** You can always determine if an object belongs to the industry solution because its object name and API name are prefixed with __ORA (double underscore followed by ORA), then a two letter acronym representing the industry solution, followed by another double underscore (__). For example: __ORACO__. The two letter acronyms for Oracle’s set of industry solutions include:
   
   - __ORATC__ for Communications
   - __ORACO__ for Consumer Goods
   - __ORAFLS__ for Financial Services
   - __ORAHHT__ for High Tech and Manufacturing
Viewing Child and Related Objects

The Overview page displays a list of child objects and related objects, if any, that have been created for an object. You can also create new child objects from this page.

- A child object is an object with a cascade delete relationship to a parent object. This means that if you delete the parent object, then all its children are automatically deleted. A child object does not exist outside the context of the parent object, and does not have its own work area. You cannot change a child object to a parent object after the child object has been created.

- Related objects can exist independently of each other, even if one object is deleted. Related objects are connected in a foreign key-based relationship between two top-level objects, not as parent and child. These types of relationships include reference relationships and dynamic choice list relationships.

Related objects can have either a one-to-many or a many-to-one relationship with the current object. Note that an object can be related to itself to model a hierarchy of the object. In this case, the object itself is displayed on its Object Overview page as a related object. For example, the Department and Sub-department objects would be displayed in this way.

**Note:** You do not create these types of relationships from this page. Instead, manage relationships from the Relationships page, which you can access from Application Composer’s main Overview page. Or, create a dynamic choice list relationship by creating a dynamic choice list field for an object, which derives its choice list values from another object.

To create a child object for a standard or custom object:

1. Navigate to an object’s Object Overview page.
2. Click the **Create Child Object** button. Creating a child object is the same as creating a custom object, except:
   - The current object is automatically displayed as the parent object.
   - Specify the **Child Collection Name** field to specify the internal name for this set of child object records, which can be used later when writing Groovy scripts.

Deleting a Custom Object

Application Composer does not support the deletion of industry solution objects.

Customizing Industry Solution Fields: Explained

Industry solution fields are custom fields that can appear in two places: on industry solution objects and on standard objects like Account or Contact. Use Application Composer to configure fields on these objects. You can hide these fields, or create new fields. The fields that you create are custom fields. When creating a custom field, Application Composer provides a set of field types that you can choose from. For example, you can create a check box field, a long text field, or a formula field. After you create a custom field, you have to then manually display the field on the desired user interface page or pages.
Industry Solution Object Fields

Industry solutions have their own business objects, and each object has its own set of standard and custom fields. If you need to capture new pieces of data, you can create custom fields on these industry solution objects. Creating custom fields is described below.

Standard Objects with Industry Solution Fields

Your industry solution might also come with industry solution fields already predefined against standard objects, such as Account or Contact. They appear in the standard object’s custom fields list. Use Application Composer to adjust their position in page layouts and remove them from the layout.

Viewing an Object’s Fields

To review the standard and custom fields for an object, and to create custom fields, navigate to the object’s Fields page in Application Composer.

1. Navigate to Application Composer from the Tools category in the Navigator.
2. Select an application from the Application choice list.
3. Expand the object and select the Fields node.

Click the Standard Fields or Custom Fields tabs to view the standard or custom fields available for the object.

- On the Standard Fields tab:
  - Review the list of standard fields that are delivered for an object, and optionally modify the display label and help text for a field.
  - The list of standard fields includes all the fields that are delivered by Oracle for an object, as well as system fields, which could include:
    - CreatedBy
    - CreationDate
    - Id
    - LastUpdateDate
    - LastUpdatedBy
    - RecordName

  **Note:** The custom objects that you create also contain these same system fields, among others.

- On the Custom Fields tab:
  - Review the list of custom fields that you created specifically for your implementation, and create new custom fields.

  **Tip:** An industry solution might also come with industry solution fields already predefined against standard objects. If such fields exist on standard objects, then they appear here as custom fields.
Adding Fields to Objects

To create a custom field:

1. Confirm that you’re in a sandbox session, before making any changes to your industry solution.
2. In Application Composer, select the object that you want to make changes to, then select the object’s Fields node.
3. On the Custom Fields tab, click **New**.

   Application Composer provides a set of field types that you can choose from when creating new fields:
   - Check box
   - Currency
   - Date
   - Datetime
   - Dynamic choice list
   - Fixed choice list
   - Formula
   - Long text
   - Number
   - Percentage
   - Record Type
   - Text

4. Select the type of field you want to create, and then specify the required field attributes to create the custom field.
5. After you create custom fields, you must expose those fields on the right user interface pages, before your end users can see them. Highlight the object’s Pages node to add custom fields to one or more page layouts for the object. See: Customizing Industry Solution Page Layouts: Explained.

Deleting Fields

You can’t delete either standard or custom fields from objects. If you no longer need a field, then optionally enter a note in the field description that the field is no longer used.

Actions and Links for Industry Solutions: Explained

In Oracle Sales Cloud, you can add actions, such as scripts, and buttons to detail pages, list pages, and so on. You can also create special fields, rendered as links, that are displayed with other fields throughout the application. Some industry solutions provide actions and links for the simplified UI pages of some objects. You can’t change these special actions and links, nor will they display in lists of actions and links. However, you can adjust their position in page layouts and remove them from layouts. Also, you can add your own actions and links to your industry solution.
What are Actions and Links?

An action can be based on a script (a Groovy method that is defined on the object) or a URL. After you create an action, you can expose the action as a button or an option on the Actions menu. After you create a link, you can select the link as a field for display at run time.

A button can perform an action or navigate the user to another page in the run time application, or to another Web site. For example, you might want to provide a static link from an overview page to a corporate Web site. Or, you might want to include a button on a summary table, which users can click at run time to create a new type of record from a selected row, such as escalating an existing “trouble ticket” to a more severe “case” that can be managed separately.

Note: Do not create custom buttons to populate the mandatory or required fields on the UI. Your end users must enter the values in the mandatory fields manually.

When configuring the pages for a standard or custom object, you can add custom actions or links to a page-level or task-level Actions menu or as a toolbar button. You can also manage the Actions menu by hiding or showing menu items, rearranging the action groupings or display sequence, and managing the toolbar by hiding or showing icons and buttons. You can also configure the Actions menu and buttons in the Create and Edit subtabs.

Tip: To support functions that don’t need to be displayed prominently on the page, add actions as options on the Actions menu. To support key functions that are frequently executed by your users, add actions as buttons. When displaying actions as buttons, be sure to test your page at run time (in all supported languages) to confirm that the presentation of buttons is as expected. Button display could be unexpected due to the available space on the page at run time, the number of buttons on the page, and button width (which depends on label length).

Note: If you create a custom button on a table that appears on a simplified UI page, and that table has no rows, then the button is automatically disabled.

Adding Actions or Links

You add actions or links in two steps:

1. Define an action or link for an object.
2. Use Application Composer to add that action or link to a simplified page.
The following figure shows a button and a link added to the Sales Opportunities Overview page.

### Defining Actions or Links

To define an action or link for an object:

1. Select an application from the Application choice list in Application Composer.
2. Select a standard or custom object in the object tree.
3. Select the **Actions and Links** node.

To create a new script or URL:

1. In the Create Action or Link page, enter a descriptive name in the Display Label field.
2. For Type, select **Action** and, for Source, select **Script** or **URL**.
3. In the Script region click the **New** icon.

The following figure shows the Create Action or Link page showing a static URL enclosed in double quotation marks.
If the source is a URL, you can enter a static URL, enclosed in double quotation marks. Or, you can define the URL by using the expression builder, which provides access to this object's fields to assist you in constructing the URL. If this object has a parent or relationship with a source object, then optionally change the context to access another object's fields for URL definition.

Any new functions that you create will be added to the Method Name choice list. If functions were already created for the object, then you can select one of them from the Method Name choice list. Object functions that are created elsewhere through other flows, such as server scripts, can also be used here.

To switch the context to the object's parent or related source object, for access to the object's fields for the URL definition, check the Select alternative context check box.

The following figure shows a script in the URL Definition window.

If the source is a script, you can either select a predefined object function from the Method Name choice list, or create a new object function using the expression builder.

**Exposing Actions or Links on Pages**

After you save actions or links, you can expose them on UI pages by configuring Application Composer options available in the Edit Summary Table page in the Pages node of an object.
The following figure shows a selected link and fields in the Edit Summary Table page in the Pages node for the Opportunity object.

When choosing to display a link, you select it just as you select to display standard or custom fields. This is because, at run time, the UI displays the URL link as if it is a field in a table. Actions can be configured in potentially two places in the UI, on the toolbar as a button and in the Actions menu for a table. Depending on how you configure actions and links, in the run time summary table, you could see both the buttons and actions, or one, or none.
The figure above shows the Configure Summary Table: Actions region, with options checked for the Show Create, Show Edit, and Show Delete options on the Action menu. It also shows a custom button and a custom action.

After you define an action, you can then expose it as a button or an Actions menu option in a variety of locations:

- On simplified pages
- Summary table on the overview page
- Default summary on the details page
- Summary table on a details page’s subtab
- Revenue table on the details page for the opportunity object

The following figure shows an overview page with exposed Create, Edit, and Delete options and a custom Ask_Assistance option on the Actions menu. It also shows the custom toolbar button Ask_Assistance, and a custom table column.
After you define a link for an object, you can add that link to a variety of locations in that object’s work area. You can add a link wherever you can add a field. Possible locations include, but are not limited to:

- As a column in the summary table on the overview page
- Default summary on the details page
- As a column in the summary table on a details page’s subtab
- In the detail form under the summary table on a details page’s subtab
- As a column in the revenue table on the details page for the opportunity object

Configuring Security for Industry Solutions: Explained

You can configure the security policy for an industry solution object through the object’s Security node. You should review the documentation provided with the installed industry solution to understand whether you need to adjust the security policy.

After you create custom objects and fields, you then expose them on desktop pages for your users. Your next step is to control which users can access that object’s data at run time. By default, the object and its records are visible and editable only to a default duty role specified by the application. Grant additional access manually using Application Composer's security policy configuration pages. A security policy specifies which users are authorized to access an object’s data, and what type of access they have. For example, does a user have view only access, or can the user create and update an object’s record, as well? Define security policies for an object by authorizing the roles whose users can access that object’s data. Or, define security policies for a role by specifying access levels across multiple custom objects.

Your industry solution documentation lists how to configure the security policy for each industry solution object.

Review these aspects of the custom object security process in Application Composer before you begin to define your security policies:

- Securing objects
- Securing roles
- Enabling function security and data security
- Application Composer and the Oracle Authorization Policy Manager (APM)
Securing Objects

The object-centric Define Policies page displays a list of the enterprise-level duty roles which map to an Oracle Sales Cloud job role. Use this page to manage access to either a top-level or child custom object by specifying a security policy for one or more duty roles. When you do this, users with the corresponding roles can access the custom object and its data, depending on the security policies you define.

To access the object-centric Define Policies page:

1. Select an application on the main Overview page.
2. Select a custom object in the object tree.
3. Select the Security node.

Or, from the role-centric Define Policies page, select a custom object.

From the object-centric Define Policies page, you can:

- Enable function security for a role.
- Enable data security for a role.

Securing Roles

The Role Security page displays a list of the enterprise-level duty roles, which map to a Sales Cloud job role. Select a role and click the Define Policies button to navigate to the role-centric Define Policies page, which displays a list of the custom objects for your Sales Cloud implementation. Use this page to manage access for users with the corresponding role by specifying a security policy for one or more top-level or child custom objects. When you do this, users with the corresponding role can access the custom objects and related data, depending on the security policies you define.

To access the role-centric Define Policies page:

1. Select an application on the main Overview page.
2. Select the Role Security node from the Common Setup pane.
Or, select the **Role Security** hyperlink in the local area of the main Overview page.

Or, from the object-centric Define Policies page, select a role.
3. Click the **Define Policies** button.

### Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>View</th>
<th>Update</th>
<th>Create</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>MGTerr</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Auto110</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>B18BATT</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Solution</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>JTestEn</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Auto110</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>BATB18</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Activity</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Tw_cust</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

From the role-centric Define Policies page, you can:

- Enable function security for a custom object.
- Enable data security for a custom object.
- View related roles, if any.

If a related role is displayed next to an object, then the selected role is inheriting its access to that object from the related role. You can drill down into the related role to view its security policies.

### Enabling Function Security and Data Security

A security policy specifies the type of access to an object and its records that users with the corresponding roles have. Access includes both function security as well as data security. Security settings are the same whether you are defining a security policy for an object or a role.

On the Define Policies page, the first four columns in the table manage function security, which applies to the object as a whole:

- **Create**
  
  Users with the corresponding role can create a record of the object.

- **View**
  
  Users with the corresponding role can view the object’s work area pages.

- **Update**
  
  Users with the corresponding role can update a record of the object.

- **Delete**
  
  Users with the corresponding role can delete a record of the object.
Users with the corresponding role can delete a record of the object.

The next two columns in the table manage data security.

- **View All**
  Users with the corresponding role can view the object’s records.

- **Update All**
  Users with the corresponding role can update the object’s records.

**Tip:** When clicking **View All** or **Update All**, the corresponding **View** and **Update** function security check boxes are automatically selected. Wait for the page to refresh to confirm all your selections.

**Note:** To let users view or update records at run time, you must enable both function security as well as data security for an object. To let users create records, you only have to enable function security.

**Application Composer and the Oracle Authorization Policy Manager (APM)**

Oracle Authorization Policy Manager (APM) manages the security policies that control access based on roles. However, you define the security policies for custom objects in Application Composer’s object-centric and role-centric Define Policies pages. This is outside APM.

Since you define the security policies outside APM, you cannot later modify the security policies within APM. Instead, modify all security policies for custom objects using only Application Composer.

**Scripting Against Industry Solution Objects: Explained**

Some scripts are already provided by Oracle for some industry solution objects. These scripts, such as field and object triggers and validation rules, are protected and you cannot configure them further. Industry solution scripts run before your scripts run, on a per object, per field, and per script type basis. For example, your object After Create triggers will run after industry solution After Create triggers on the same object. This topic provides an overview of the type of scripting you can do in Application Composer. For a complete understanding of the scripts provided in your industry solution, refer to your industry solution’s documentation.

Groovy is a standard, dynamic scripting language for the Java platform for which Application composer provides deep support. This topic provides an overview of where you can use groovy in your application and gives some samples of one or more lines of Groovy code.

For more information on Groovy scripting, see Groovy Scripting Reference For Application Composer at http://www.oracle.com/pls/topic/lookup?ctx=cloud132&id=CGSAC.

**Note:** Read “Supported Classes and Methods for Use in Groovy Scripts”, which documents the only classes and methods you may use in your Groovy scripts. Using any other class or method will raise a security violation error when you migrate your code to later Oracle Sales Cloud maintenance releases. Therefore, we strongly suggest that the Groovy code you write uses only the classes and methods shown there to avoid the unpleasant and possibly time-consuming task of having to rewrite your code in the future.
Terminology Explained

The term script is used to describe one or more lines of Groovy code that the Oracle ADF framework executes at run time. Often a very-short script is all that is required.

For example, to validate that a CommissionPercentage field’s value does not exceed 40%, you might use a one-line script like:

\[
\text{return \ CommissionPercentage < 0.40}
\]

In fact, this one-liner can be conveniently shortened by dropping the return keyword since the return keyword is always implied on the last line of a script:

\[
\text{CommissionPercentage < 0.40}
\]

For slightly more complicated logic, your script might require some conditional handling. For example, suppose the maximum commission percentage is 40% if the salesperson’s job grade is less than or equal to 3, but 60% if the job grade is higher. Your script would grow a little to look like this:

\[
\begin{align*}
\text{if (JobGrade <= 3) \{} \\
\quad \text{return \ CommissionPercentage < 0.40} \\
\text{\}} \quad \text{else \{} \\
\quad \text{return \ CommissionPercentage < 0.60} \\
\text{\}}
\end{align*}
\]

Scripts that you’ll write for other purposes like complex validation rules or reusable functions may span multiple pages, depending on your needs.

When a context requiring a Groovy script will typically use a short (often, one-line) script, we emphasize that fact by calling it an expression, however technically the terms script and expression are interchangeable. Anywhere you can provide a one-line expression is also a valid context for providing a multi-line script if the need arises. Whether you provide a short expression or a multi-line script, the syntax and features at your disposal are the same. You need only pay attention that your code returns a value of the appropriate type for the context in which you use it.

Refer to the Groovy Scripting Reference For Application Composer for a discussion of all the return types.

Where You will Use Groovy in Your Application

There are a number of different contexts where you will use Groovy scripts as you customize existing objects or create new custom ones.

You will write shorter scripts to provide an expression to:

- calculate a custom formula field’s value
- calculate a custom field’s default value
- make a custom field conditionally updatable, or
- make a custom field conditionally required
- define the condition for executing an object workflow

You will generally write somewhat longer scripts to define:

- a field-level validation rule
- an object-level validation rule
- a trigger to complement default processing
• utility code in a global function, or
• reusable behavior in an object function

If you anticipate calling the same code from multiple different contexts, any of your scripts can call the reusable code you write in either global functions or object functions. As their name implies, global functions can be called from scripts in any object or from other global functions. Object functions can be called by any scripts in the same object, or even triggered by a button in the user interface.

After exploring the Groovy basic techniques needed to understand the examples documented in Groovy Scripting Reference For Application Composer at http://www.oracle.com/pls/topic/lookup?ctx=cloud132&id=CGSAC, see "Groovy Scripting: Examples" for a concrete example of each of these usages. Also see "Groovy Tips and Techniques" in Groovy Scripting Reference For Application Composer at http://www.oracle.com/pls/topic/lookup?ctx=cloud132&id=CGSAC151 for getting the most out of Groovy in your application.

Using Runtime Messages: Explained

You can view diagnostic messages generated by any script through the Runtime Messages page. Use these diagnostic messages to assist with debugging your scripts and the interaction between your scripts and industry solution scripts.

Use the Runtime Messages page, also known as the diagnostic dashboard, to view the diagnostic messages your scripts have written to the log. On the Runtime Messages page, click the Get Latest Log Messages button to retrieve the latest Groovy script messages from the log file.

Runtime Messages

Runtime messages are diagnostic messages that you add to your script. They’re useful for debugging your scripts if an error occurs.

To access the Runtime Messages page in Application Composer, go to the Common Tasks pane on the bottom left side, then click Run Time Messages. You must be in an active sandbox to perform this action.

Runtime messages are user-specific. Only you can see the messages that you create and add to the system.

Using a Script to Write Messages to the Log

To write messages to the diagnostic log, use the print or println function. The former writes its value without any newline character, while the latter writes its value along with a newline. For example:

```
// Write a diagnostic message to the log. Notice how
// convenient string substitution expressions are
println("Status = \${Status_c}")
```

Finding Messages

To find your messages on the Runtime Messages page:

1. In Application Composer, select the web application from the Application list of values.
2. Go to the Common Tasks pane on the bottom left side, then click Runtime Messages.
3. Click the Get Latest Log Messages button to retrieve the latest Groovy script messages from the log file.
Your println Groovy scripts are written to different applications depending on the UI from which they’re invoked (in other words, from where your scripts are triggered).

- If your script is triggered from the simplified UI, then the messages are written to Common Setup > Runtime Messages under the application that displays in the URL when you click the global Home icon: http://<host>:<port>/<application>/faces/CrmFusionHome.

  For example, if the application in the URL is Customer, then navigate to Common Setup > Runtime Messages under the Customer Center application to find your runtime messages.

- If your script is triggered from the desktop UI, then the messages are displayed on the same Runtime Messages page, but under the actual web application from where your script was triggered.

For example, if a println Groovy script is tied to a Create Quote button which is displayed on a simplified UI Opportunity page, then the message will be printed in the Customer Center application’s runtime messages. If the same action is displayed on the desktop UI Opportunity page, then the associated message will be printed in the Sales application’s runtime messages, since your script was triggered from the Sales application.

To sort in reverse chronological order so you can see your most recent messages first, click the down-pointing arrow in the Time stamp column header.

Customizing Industry Solution Page Layouts: Explained

The industry solutions available for Oracle Sales Cloud are available to end users on simplified user interface (UI) pages only. Pages of other types, such as mobile or desktop pages, are not available for industry solutions. Industry solution page layouts are permanently deprecated (inactive). As part of your industry solution setup activities, you must make copies of these special page layouts, then indicate who can see them and add some other attributes, before rolling out the industry solution to your end users. This is referred to as "activating" a layout in your industry solution documentation. Refer to this documentation to review your page layout setup tasks specific to your industry solution.

What's a Page Layout?

A page layout is a design of a page which you create, and then attach conditions to. Using conditions, you can present the same Sales Cloud page differently to various users. For example, a sales manager can see one version of the page, while the sales representative sees another version. Page layouts are available only for simplified pages.

Working With Industry Solution Page Layouts

Industry solutions are delivered with standard layouts, which are automatically inactive. You cannot edit the existing standard layouts. Instead, you must duplicate them first, and then edit the new page layout.

You can:

- Duplicate industry solution page layouts

  To create a new page layout, duplicate an existing layout and then make your edits. Instructions specific to your industry solution are provided in your industry solution documentation.

- Edit page layouts

  Edit your new page layouts that you created for the pages of industry solution objects.

- Inactivate, or deprecate, page layouts
You cannot delete page layouts, but you can inactivate them by deselecting the Active check box for a page layout on the Simplified Pages tab.

**Editing Page Layouts**

When you edit a page layout, you can interact with items inside regions, or interact with the regions themselves. You can:

- Add fields, actions, links, buttons, and subtabs.
- Hide and show, reorder, and relabel regions, including subtabs.

To edit a page layout:

1. In Application Composer, navigate to the object that you want to create page layouts for.
2. Expand the object in the object tree, and select the Pages node.
3. On the Simplified Pages tab, select a page type, such as the creation page or details page. Then, in that page type’s region, click the **Duplicate Layout** icon to duplicate and edit an existing layout.

4. Next, design the page layout. Depending on how the page is designed, you can add fields, actions, links, and buttons.

   You can also hide and show, reorder, and relabel regions.
If the page layout is for a details page, then you can also add and reorder subtabs.

Assigning Conditions to Page Layouts

After you edit the page layout, you can assign one or more conditions that control when the layout is displayed. You assign conditions to page layouts when viewing them on the Simplified Pages tab.

**Note:** The layouts you create are displayed in a table, and the order of layouts in each table is significant. At runtime, Application Composer evaluates the condition or conditions specified in each layout, starting with the first layout listed in the table. The first layout that matches all Type, Role, and Expression conditions is selected for display at run time. The standard layout is always the last layout in the table, and it cannot be removed.

- **Type**
  
  a. Select the page layout you want to add conditions to.
  
  b. Select a record type field value, if a record type field has been created for the object. At run time, if the value is selected, then this page layout displays.
The default value for this condition is ANY, so if you do not specify a Type condition for a layout, then Application Composer views this condition as satisfied when evaluating a layout for display at run time.

- **Role**
  a. Select the page layout you want to add conditions to.
  b. Select the role who can see this page layout. For example, perhaps only the sales representative can see this page layout at run time.

  The default value for this condition is ANY, so if you do not specify a Role condition for a layout, then Application Composer views this condition as satisfied when evaluating a layout for display at run time.

- **Advanced Expression**
  a. Select the page layout you want to add conditions to.
  b. Click the Calculator icon.
  c. Enter a Groovy expression that controls when this page layout is displayed.
4 Customizing Industry Solution Pages Using Other Tools

Using Page Composer to Change Industry Solutions: Overview

Use Page Composer to customize the user interface pages that are provided with your industry solution. Page Composer is a web-based tool you can use to modify user interface (UI) pages and components for all products designated for use with Page Composer. In Oracle Sales Cloud, Page Composer is intended for simple user interface editing functions. For example, use Page Composer to change the label for a field on a limited area. Or use Page Composer to hide or show fields on a page for a specific role, such as only a sales manager. You can also use Page Composer to create saved searches which can be used by all your end users to query and display a subset of records on a page.

What Will You See?

When you interact with Page Composer, you can do so in two different modes: Source View and Design View. For the most part, when making changes to Oracle Sales Cloud pages using Page Composer, you can use only Design View. Even with Design View, Page Composer provides two different modes of Design View.

- The first mode, Design View: Standard mode, is selected by default in all Sales Cloud pages when opening a page with Page Composer with the Design button selected.
- The second mode, Design View: Direct Selection mode, is activated when you click the Select tab for the UI page you want to customize.

With the Design View: Direct Selection mode, you can select and edit UI elements such as form fields and table columns. In Direct Selection mode, selectable UI components become apparent when you move your cursor over the UI component. Selectable UI components are highlighted and can be edited.

In Sales Cloud, Direct Selection mode is available when you customize pages, but not when you personalize a dashboard page.

What Can You Change?

Use Page Composer to make the following changes to an industry solution:

- Customize industry solution field labels.

  It’s a best practice to use Page Composer when your goal is to limit a field label change to the smallest scope possible. If your goal is to make large-scale changes, such as across multiple pages, then use the Customize User Interface Text tool.

- Customize the look and feel of industry solution pages. For example, show or hide fields.
You can also indicate if you want your page changes to be visible only for a specific role, like a Sales Manager or a Sales VP.

- Configure dashboard KPIs.

Some industry solutions are delivered with dashboards that are populated with key performance indicators (KPIs) for more accurate forecasting and management of resources. In some cases, administrators must use Page Composer to add industry-specific KPIs to these dashboards. These KPIs are also referred to as reports or infolets.

For example, when setting up the Communications industry solution, administrators will use Page Composer to add communications-specific reports to the dashboard landing page. These steps are outlined in the documentation that's provided with the Communications industry solution.

- Create saved searches.

You can save search criteria and search settings for reuse to speed up future searches. You can create the saved searches (also called saved lists) for your own use, or create them while in Page Composer to make the saved searches available to other users in Oracle Sales Cloud.

You can also delete, rename, or change the search options for a saved search.

For example, when setting up the Consumer Goods industry solution, administrators will use Page Composer to create a saved search for sales managers. This search will let sales managers view specific information about accounts owned by their reporting sales reps. These steps are outlined in the documentation that’s provided with the Consumer Goods industry solution.

### Making Saved Searches Available to All Users: Procedure

Use Page Composer to customize saved searches, and make them available for all users. Create and edit saved searches using Page Composer at the site layer. Users can run these saved searches again later to use the same search criteria and other settings. You must create or edit saved searches only at the site layer to make them available for all users.

### Customizing Saved Searches for All Users

Follow these steps:

1. Activate a sandbox.
2. On the search page that has a Save button, click your user name in the global area, and select Customize <Page Name> Pages under Administration.
3. If prompted to select a customization layer, select the site layer to open the search page in customization mode.
4. From the View menu, select Design.
5. Create and edit saved searches.

**Note:** The steps for creating and editing saved searches are the same regardless of whether you're working on saved searches for yourself or for all users.

6. Save your changes and close Page Composer.
7. After testing your changes, publish the sandbox to make your changes available to all users.
Bulk Text Customizations: Explained

Use the User Interface Text tool to simultaneously customize multiple occurrences of entire words or phrases in the user interface (UI).

You can use this tool to do the following activities for bulk text customization:

- Sandbox integration
- Case-sensitive and whole word searches
- Singular and plural text replacement
- Contextual previews

🔍 Note: You must activate a sandbox to use the User Interface Text tool.

To use the User Interface Text tool, from the Navigator menu, select **Tools - User Interface Text**. Then, click **Search and Replace** to search and replace texts in bulk. The User Interface Text tool searches text on pages and in messages in the message dictionary. The search includes user assistance only if the user assistance text is in the message dictionary. The customization functionality for this tool doesn't extend to text in service oriented architecture (SOA) processes.

In the User Interface Text tool, you can:

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

Search and Replace

After you activate a sandbox and click **Search and Replace**, enter the search text and the replacement text. You can enter the singular and plural forms of whole words or phrases. You can also use the following check boxes:

- Match Case - To perform case-sensitive searches.
- Match Complete Word or Phrase - To search for an exact match of your search text value.

🔍 Note: You can't 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>Flex</td>
<td>flex</td>
<td>Yes</td>
<td>The application searches for any occurrence of your search string</td>
</tr>
<tr>
<td>Search Text</td>
<td>Expected Match</td>
<td>Match?</td>
<td>Reason</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>flex</td>
<td>flexfields</td>
<td>No</td>
<td>The application treats your search text value as a whole word. The text flex isn’t the same as the text flexfields.</td>
</tr>
<tr>
<td>^Flex$</td>
<td>flexfields</td>
<td>No</td>
<td>Use ^string$ in the search field to say this string must match the complete field.</td>
</tr>
<tr>
<td>^Flex$</td>
<td>flex</td>
<td>Yes</td>
<td>Use ^string$ in the search field to say this phrase must appear at the beginning of a field.</td>
</tr>
<tr>
<td>flex credits</td>
<td>Flex Credits Configuration</td>
<td>Yes</td>
<td>The application searches for the exact spelling and sequence of words without regard to their position in the strings it searches.</td>
</tr>
</tbody>
</table>

Unless you select **Match Case**, all matches are considered exact.
### Include one or more of the following match categories in your search:

- User Interface Text
- Oracle Transactional Business Intelligence Repository Definition Text
- Enterprise Scheduler Text
- Global Menu Label Text
- Multipart Validation Message

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

### Preview and Adjust

The preview sorts the search results and presents the matches on tabs based on match categories. 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.

**Caution:** Some tab names on the Preview Text Changes page are followed by an asterisk (*) (for example, the Messages tab). For these tabs, once you save the preview text changes, these text changes are applied to the mainline metadata, that is, outside the sandbox. So you can’t undo your changes after you preview and save them even though you are currently in a sandbox. Also, on destroying the sandbox, these changes remain as is.

Each row on all tabs includes:

- A view of the existing text and the immediately surrounding text for context. You can’t 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.

Each row on the Messages tab includes an indicator. This indicator identifies when the search text appears in a message subcomponent, not necessarily in the message body displayed in the row. You can expand the row and view subordinate rows that display the message subcomponent containing 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.

In the Global Menu Label Text tab, you can’t update a secure JSON Web Token (JWT).

If you have multiple languages in your application and you want to make similar text changes in those languages:

1. Change your language preference
2. Search and replace text
3. Validate your changes

Save and Publish

After you review and adjust the matches:

1. Save your customizations.
2. Thoroughly test the run time pages to make sure that every occurrence of the text is replaced, as you wanted.
3. Publish the sandbox.

Note the following points:

- Do not publish a sandbox before you visually inspect and validate all pages and messages that contain text that you customized.
- Users can view:
  - Message, business intelligence, and enterprise scheduler text customizations when you save them, even if you don’t publish the sandbox.
  - Page text customizations when you publish the sandbox.

Related Topics

- Selecting a Text Customization Tool: Points to Consider
- Using Sandboxes: Explained
- Setting Up Sandboxes: Procedure

Changing Field Display Labels: Explained

You can change the display labels (strings) that appear in Oracle Sales Cloud using one of three tools: Customize User Interface Text Tool, Application Composer, or Page Composer. These tools enable the administration of strings. Your tool selection depends on the scope of the changes that you want to make. For large-scale changes, use the Customize User Interface Text Tool. For changes on a single page, use Page Composer. For changes to all labels for a custom object, use Application Composer. Read this topic to learn more about best practices for changing display labels.

Display Labels

A single field label, such as Products, can occur throughout Oracle Sales Cloud on a variety of pages. However, just because the same field label appears multiple times doesn't mean that it's the same field across all pages. In fact, it's probably not.
Note that there's a difference between a field label instance and a field label value. The value is what is seen on a page. The instance is the actual field location where the label value is retrieved from. Typically, multiple display label instances may exist that have the same value. This is especially true for common labels such as Account and Customer.

For example, let's say you want to change Products to Our Products, but the field label Products appears throughout Oracle Sales Cloud 500 times. You don't have to change the Products display label 500 times. However, you'll probably need to change it more than once. Again, this is due to the underlying architecture of Oracle Sales Cloud where multiple display label instances can (and do) exist with the same value.

Display Labels and Multiple Languages
Oracle Sales Cloud is fully globalized and supports multiple languages. English is installed by default and additional languages can be requested. When additional languages are installed, any user can change the current session locale (language) of their login session through the Preferences menu. However, string administration tools can change display labels to any language, regardless of the session locale. For example, the active session locale can be French, but you can use any string administration tool to enter Korean display label values. As a result, Korean display label values will be seen, even if the session locale is French. Keep this in mind when making changes to field display labels.

Changing Display Labels Across Multiple Pages
The Customize User Interface Text Tool, sometimes referred to as the String Editor, administers the vast majority of display labels across Oracle Sales Cloud. It is designed for bulk display label search and replace. This tool supports all installed languages and can be used to create and modify locale-specific label values.

Note: You cannot modify list of value fields (also known as choice lists) using the Customize User Interface Text Tool. Instead, use Setup and Maintenance. In some cases, you can also use Application Composer to modify these special types of fields.

Let's look at changing the display label Products to Our Products wherever it appears throughout Oracle Sales Cloud.

To make a large-scale change of a display label value across multiple pages:

1. First, log in as an application implementation consultant and confirm that you are working in an active sandbox.
2. Using the Navigator, select User Interface Text from the Tools category.
3. Click Search and Replace.
4. In the Search For and Replace With boxes, enter the word or phrase that you want to change and the corresponding replacement.

Tip: For optimal results, enter search criteria that produces the smallest possible number of search results. This makes it simpler to preview all possible matches in the next step. To do this, instead of entering Products, enter ^Products$. Behind the scenes, the search engine finds label values where Products is the entire display label value.

5. Click Match Case.
6. Click User Interface Text and Global Menu Label Text to search only the display label category that contains the vast majority of UI display labels.

All other check boxes can be deselected.
7. Click Preview Changes to review, modify, and exclude individual occurrences before you save your changes.
Changing Display Labels on a Page

Let’s look at changing the display label Customer to Prospect, but only on a single page that sales reps use to manage their prospects. In this case, you’ll use Page Composer.

In contrast to the Customize User Interface Text tool, Page Composer accesses only one label instance at a time and only through a rendered desktop or simplified UI page. Even though its use is limited to these UI channels, using Page Composer to change a label might result in changes to other UI channels, such as Mobile, because a display label instance can be the source of display label values on any UI channel.

Also, Page Composer might not have access to all the display labels on a given page. In contrast, the Customize User Interface Text tool has access to the greatest number of display labels on any page.

Changing Custom Object Display Labels

Let’s look at changing all the display labels for a custom object, Trouble Ticket. In this case, you’ll use Application Composer.

Use Application Composer to change a display label value for custom object and custom field display labels. The change is displayed everywhere the original label value was displayed, across all UI channels.

Note, however, that if you use Application Composer to change display labels for standard object and standard object standard fields, the change is not displayed everywhere. This is because Application Composer changes only one instance of a label, while there are typically multiple instances of a label in use for every standard object and standard object standard field. In this case, Application Composer and Page Composer are similar tools: both tools change one and only one instance of a display label. Again, only the Customize User Interface Text tool has the potential to change every instance of a display label.

Application Composer is an English only tool. It reads and writes only to English label data sources. Application Composer reads and writes in English display label data sources only, even if a non-English locale is the active session locale.

💡 Tip: To add non-English display label values for custom objects, custom fields, standard objects, and standard fields, use the Customize User Interface Text tool.
5 Customizing Analytics

Understanding Analytics for Industry Solutions: Overview

Some industry solutions are delivered with dashboards that are populated with key performance indicators (KPIs) for more accurate forecasting and management of resources. Refer to the documentation that came with your industry solution to understand the analytics capabilities that are provided for you. In many cases, you must use one of the customization tools provided by Oracle Sales Cloud to properly set up the analytics capabilities for each industry solution.

What's Available?

Refer to the documentation that came with your industry solution to understand the analytics capabilities that are provided for you.

For example:

- **Communications**
  
  Sales executives can review key numbers in the dashboard for the quota achieved. The KPIs represent current quarter opportunities and projected revenue. The KPIs also represent a breakdown of those figures across contract renewals and new deals, historical and current renewal rates, bundle deal targets and stalled deals.

- **Consumer Goods**
  
  Sales managers and field sales representatives can stay up to date on KPIs through a dashboard. Users can use the infolets on the dashboard to quickly view important metrics related to appointments, accounts, promotions, tasks, and sales performance. Users can then click an infolet to drill down to more details to view related transactional data.

- **Financials**
  
  Retail bank managers can view opportunity reports for home loan applications, and gain insights using KPIs. Users can use the infolets on the dashboard to quickly view important metrics related to loan applications, opportunities, and performance. Users can then click an infolet to drill down to more details to view related transactional data.

Displaying Industry-Specific Analytics

You can customize the dashboards that come with your industry solution by adding prebuilt reports that are specific to your industry solution. To do this, administrators must use Page Composer to add industry-specific KPIs to these dashboards. These KPIs are also referred to as infolets or reports.

For example, when setting up the Communications industry solution, administrators will use Page Composer to add communications-specific infolets to the dashboard. These steps are outlined in the documentation that's provided with the Communications industry solution.

Administrators setting up the Financials Services industry solution will also use Page Composer to add financial services-specific infolets to the dashboard, as well. Refer to the Financials Services industry solution documentation for details.
Customizing Analytics for Industry Solutions: Explained

Some industry solutions are delivered with dashboards that are populated with key performance indicators (KPIs) for more accurate forecasting and management of resources. These KPIs are also known as infolets, which are actually prebuilt reports that come with your industry solution. To ensure industry solution dashboards reflect the right KPIs, administrators use Page Composer to manually hide horizontal reports, and display industry-specific reports. But what if you want to customize these reports by adding custom fields, for example? You can do this using Oracle Sales Cloud’s customization tools. Use Application Composer to modify or create custom subject areas (the foundation that reports are built on top of), and use BI Publisher to modify or create new reports.

What are Infolets (or Reports)?

Infolets are regions on a dashboard that let users quickly view important metrics related to their specific areas of interest. Infolets are basically reports that are built using Oracle Business Intelligence Enterprise Edition (EE). Users can view important details in the infolet, and then click that infolet to drill down to view related transactional data.

As an administrator, you can:

- Use BI Composer to update existing infolets to capture custom data and also to create new reports.
- Use Application Composer to create entirely new custom subject areas (the foundation that reports are built on top of), then use BI Composer to create new reports built using that subject area. Finally, use Page Composer to display your new reports.
- Use Page Composer to hide and show infolets on a dashboard.

What is BI Composer?

The Oracle Business Intelligence (BI) Composer is a tool you can use to build reports. When building reports, you select a report subject area from within BI Composer. A report subject area is a set of entities, attributes, and measures that represent information about the areas of an organization’s business. To build reports, use either the predefined report subject areas that are delivered for Oracle Sales Cloud, or create a custom subject area using a wizard available in Application Composer. Some custom subject areas are already provided by Oracle for some industry solution objects.

Adding Custom Fields to Reports

You can add new custom fields to an industry solution if you need to capture data that’s not already captured. When you add a custom field to an industry solution object, that object’s existing custom subject area (the foundation that reports are built on top of) is automatically updated to include your custom field. To include your custom field on existing reports, however, you’ll need to use the Customize option in BI Composer to make that change. Prebuilt reports (infolets) aren’t automatically updated to include your new custom field.

The Customize option automatically creates a custom copy of a predefined report and stores it in the Shared Folders - Custom within the Oracle BI Presentation Catalog. The report definition, directory structure, and original report permissions are copied.

This custom copy is linked internally to the original report. You can edit the custom copy of the report, leaving the original report intact. When users open or schedule the original report, they are actually using the custom version.
Aside from conveniently copying a predefined report to the Custom directory, the Customize option:

- Makes it unnecessary to update processes or applications that invoke the report. For example, if the original report is set up to run as a scheduled process, then don’t change the setup. When users submit the same scheduled process, the custom report runs instead of the original.
- Automatically copies the security settings of the original report.
- Removes the risk of patches overwriting your edits. If a patch 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 structure of the data model might affect your custom report.

To update an existing industry solution report to incorporate your new custom field:

1. Republish the industry solution’s custom subject area to incorporate the custom field.
3. In the Folders pane, select the predefined report.
4. Select **Customize** from the More menu for the report.
5. The copied report in the Custom directory opens, so proceed to edit this report.

To edit the custom report again later, you don’t need to be in the BI Server. Just go to the Oracle BI Presentation Catalog and either:

1. Select the **Customize** or **Edit** option for the original report.
2. Find your custom report in the Custom directory and select **Edit**.

**Tip:** Industry solution analytics are located in the `/shared/Extension/[Industry Solution]` directory. Before modifying the analytics in this location, you should make copies of the analytics to be modified and place them in another location. Or, use the Customize option, which automatically creates a copy of the desired report into the Custom directory. Avoid saving modified or new content in or below the `/shared/Extension/` directory because content in this location can be periodically refreshed by an update and your changes can be lost. Existing references to the original content should be updated to reflect the new content location as necessary.

Review the following section to learn how to use BI Composer to create reports.

**Creating Reports Using Existing Subject Areas**

To create a report using BI Composer, use either the predefined report subject areas that are delivered for Oracle Sales Cloud or the predefined custom subject areas delivered for your industry solution.

To create a report in BI Composer, you’ll first access a report subject area (either a standard or custom subject area) which is the foundation for your report. Then, you can create the report.

To create a report in BI Composer:

1. From the Navigator menu, select **Tools - Reports and Analytics**.
2. In the Contents pane, click **Create**.
3. Select the published custom subject area and start creating your report.
This figure shows a sample custom subject area. Note the custom: prefix.

4. Select the fields and layout for the report. Optionally select filters to filter report data.
5. Preview your report to ensure you’re happy with it.
6. Save your report.

To watch a video that walks you through this example, see Fusion Application CRM Analytics Introduction (1474869.1) on My Oracle Support at https://support.oracle.com.

**Tip:** You can also create an entirely new custom subject area using Application Composer if required. Refer to the next section for details on creating custom subject areas.

Creating New Custom Subject Areas
You create custom subject areas using a wizard in Application Composer. The section below describes the overall process to create a custom subject area. For more specific details, review the Oracle Sales Cloud Customizing Sales Guide.

To create a custom subject area:
1. Navigate to Application Composer.
2. Select an application from the **Application** list.
3. Click **Custom Subject Areas** on the Overview page of the Application Composer.
4. Select **Actions - Create**.
   Create Custom Subject Area: Define Custom Subject Area page opens.
This figure shows the first page in the guided process for creating custom subject areas.

Here are the steps in the train stops that you can use for configuring your custom subject area:

1. **Define Custom Subject Area**

   In this step, you provide the name for your subject area and select the primary object that is the basis for the reports you create later using the custom subject area. Subject areas usually have names or labels that correspond to the type of information they contain, such as service requests and orders. Display labels have the Custom: prefix added automatically.

2. **Select Child Objects**

   In this step, you select the child objects whose data you want to use in your reports. You can add child objects only if the primary object has child objects. Otherwise, the add icon is disabled. When you select child objects for your custom subject area, consider the following points:

   - For a one-to-many primary-child relationship, you can add multiple child objects if required. The parent-child-grandchild-grand-grand child hierarchy supports adding only up to three levels of child objects, for example, parent-child1-child1.1-child1.1.1.
   - For a many-to-one primary-related relationship, you can add as many related objects as you want.

3. **Configure Fields**

   In this step, you select the fields that you want to display on your reports. You typically add at least one field from each of the objects that you have selected for your custom subject area.

   Select the desired measures to generate for number, date, or currency fields from all the available objects so that the subject area includes only those measures that you want to analyze. Also, define at least one measure.

   In the **Select Aggregations** column, select an option from the list of predefined formulas that you can apply to the Measure field. When you select the formula, the application applies the selected formulas to the selected field and measures.

   You can change the display labels of the fields that you select in this step. Additionally, you can use the Select Fields dialog to remove fields that belong to the primary object, or add fields from the related objects. The Select Fields dialog appears when you click Select Fields when configuring fields for your custom subject area.
This figure shows the Select Fields dialog.

After you publish your custom subject area, the fields you have selected for your subject area are automatically added to their owning object’s directory. If you have also defined measures, those fields are automatically added to the Facts directory.

This figure shows the fact directory and the object directory whose fields are selected.

4. Configure Implicit Fact
Use the mostly used measure as the Implicit Fact.

5. Configure Date Leveling

If required, select the Date columns for date leveling.

6. Configure Security

Select the required security level for the Everyone Role Name, which is added by default or add additional Role Names by clicking in the + icon and define the security level for each one of them.

**Note:** The security definition here only control who can access the custom subject area definition to create reports. It doesn’t control data visibility which is automatically controlled based on the user running the reports.

7. Review and Submit

Review the custom subject area configuration for all added objects, attributes, and measures, and if satisfied, click Submit. If changes are required click on Back to navigate back to the required screen.

After you submit, the custom subject area configuration is prepared for publishing. You can create and submit a custom subject area either immediately or save and close the custom subject area at any point and submit it later. You must first submit a custom subject area for publishing before you can select it from within Oracle BI Composer. After you save or submit a custom subject area, you cannot modify its primary object.

To access the published custom subject area in BI:

- From the Navigator menu, select **Tools - Reports and Analytics**.
- In the Contents pane, click **Create**.
Select the published custom subject area and start creating your report.

This figure shows a sample custom subject area. Note the custom: prefix.

Displaying Reports on Dashboards
You can display new reports on dashboards using Page Composer. This process is described in the documentation that came with your industry solution.
Creating Object Workflows

Automating Processes with Object Workflows: Explained

A business process typically has some kind of action that you want to occur in an application, based on something happening elsewhere in the application. To automate this type of process, create an object workflow in Application Composer. When you create an object workflow, you indicate which business object you're impacting, like Opportunity or one of your industry solution business objects. You also indicate the actions that you want to happen in the application, plus the conditions that trigger these actions. Conditions are events. Once those events occur, your configured actions are immediately implemented unless you have defined a time rule (execution schedule) for one or more actions.

An Industry Solution Example

For example, when setting up the Consumer Goods industry solution, administrators will create an object workflow to set the status of an order to **On hold** when the order has been submitted and is pending payment for more than 60 days.

These steps are outlined in the documentation that's provided with the Consumer Goods industry solution.

Creating an Object Workflow

To create an object workflow, use Application Composer.

1. Navigate to Application Composer.
2. Select the application from the Application choice list.
3. In the Common Tasks region, click **Object Workflows**.
4. Select **Actions - Create**.

Configure your object workflow. For example:

   a. Select the object that you want to create the object workflow for.
   b. Indicate the conditions that trigger the workflow.
   c. Indicate which actions should be executed when the conditions are met. For example:

      - **Update a field.**
        Use this action to specify the new values that you want to apply to the existing fields. For example, create a workflow that changes the Priority from Medium to High when the win probability exceeds 60 percent. For more information on field updates, see Object Workflows and Field Updates: How They Work Together topic.
      - **Send an e-mail.**
        Use this action to send automatic e-mail alerts. For example, create a workflow that alerts the sales team that the budget availability date has changed.
      - **Create a task.**
        Use this action to create and assign tasks. For example, assign a follow-up task to the owner of an opportunity when the status is still open.
      - **Send an outbound message.**
Use this action to send an outbound message to a Web service at a specified endpoint URL. For example, send account details from one system to another.


Use this action to trigger approval flows. For example, when the Deal Amount exceeds a threshold level, the relevant sales manager receives an approval notification.

Optionally indicate the execution schedule for each event action. The schedule governs when an event action should happen. Otherwise, the event actions are run immediately. You can set a time rule for an event action based on whether that event action must occur after or before the triggering of a workflow or the occurrence of a date, and also specify the time duration in **Hours**, **Days**, or **Weeks**. If you schedule an event action for a time in the past, the event action is executed immediately after it is triggered.

**Tip:** When multiple event actions are configured, the Field Updates happen first before any other event action is triggered. For event actions other than Field Updates, there is no particular sequence and the event actions are triggered based on whether or how you configure the execution schedule.

### Commonly Used Terms

Familiarize yourself with the following terms before you start working with object workflows:

- **Business Object**: A business object can be either a standard object delivered with Oracle Sales Cloud or a custom object that you create based on your business need.

  The business object that you use for configuring the object workflow can be either a parent object or a child object. All attributes available to you for selection when configuring a workflow belong to the object that you select for creating a workflow. This enforces that only the data relevant to the object in context is available for selection. For example, when you define a trigger condition, an expression editor lists only those fields that are relevant to the object that you have selected.

- **Active**: Indicates whether the workflow is in use. Only those object workflows which are in Active status can trigger event actions.

- **Event Point**: An event point is associated with an object and is an instance when an event occurs. Event points are of the following types:
  - When a record is created.
  - When a record is updated.

- **Event Condition**: Event point and event condition together serve as a trigger for object workflows. Event condition is an expression that supports logical, math operations, or field-value lookups. By defining an expression, you can prevent the object workflow to trigger each time a record is created or updated. See Expression Builder in this topic for an overview of how expressions are defined.

- **Event Action**: An event action determines what action is expected from an object workflow when the conditions set for an object are met.

  Event actions are of the following types:
  - Field Updates
  - E-Mail Notification
---

- Task Creation
- Outbound Message
- Business Process Flow

---
7 Oracle Sales Cloud Web Services

Web Services for Industry Solutions: Overview

You can integrate your Oracle Sales Cloud industry solution with external applications using REST or SOAP Web services. For the Web services that you call in your Groovy scripts, you'll need to first create a reference to those services in Application Composer which you can then use in your scripts.

SOAP Web Services

Industry solution objects have SOAP Web service interfaces. For SOAP Web service operations, all industry solution objects are accessed through one of six Oracle Sales Cloud custom business object Web services.

- Custom Common Business Object service
- Sales Custom Business Object service
- Customer Custom Business Object service
- Marketing Custom Business Object service
- Sales Lead Custom Business Object service
- Sales Territory Custom Business Object service

SOAP Web service interactions with industry solution objects are typically made through these services:

- Custom Common Business Object service
- Sales Custom Business Object service

Access top-level industry solution objects (industry solution objects visible in the Application Composer custom objects tree) and their child objects through custom object Web service interfaces. Other industry solution objects can be children of standard objects; access these child objects through the parent standard object Web service interface. Access industry solution fields of standard objects as custom fields of the standard object.

For more information, see:

- "Introduction to Using Web Services" in Oracle Cloud Getting Started with Web Services
- "Working with Custom Object Web Services" in Oracle Sales Cloud Use Cases for Implementing Applications for Oracle Sales Cloud

REST Web Services

Oracle Sales Cloud provides full REST support for all industry solution objects and attributes.

Registering Oracle Sales Cloud Web Services

In the Groovy scripts that you use in Application Composer, you can include calls to both internal and external Web services. For each Web service that you call in your scripts, you must set up a Web service reference that specifies the Web Services Description Language (WSDL) file location and the security scheme, if any, used to access the Web service. When setting up your industry solution, you might have to create such a Web service reference for some internal Web services. This is also known as “registering” a Web service. Once you register a Web service, you can include a call to that service in your scripts.

For example:

- **Consumer Goods**
  - Register the Item Web service using Application Composer, so that Promotion Group product details can be retrieved from Supply Chain Management Cloud.

- **Financials Services**
  - Register the Item Web service using Application Composer, so that item details can be updated in Supply Chain Management Cloud.
  - Register the Web service to use the Sales Coach attachment for each action item.

- **High Tech and Manufacturing**
  - Register the following Web services for Deal Registration to enable partners to convert a lead to a deal:
    - DealService
    - SalesLeadService
    - SalesCustomObjectService

For details about how to use Application Composer to register these internal Web services, refer to the documentation that came with your industry solution.

Calling Web Services from Groovy Scripts: Explained

You can call Web services from your Groovy scripts in Application Composer. You might call a Web service for access to internal or external data, or for example, to perform a calculation on your data.

Calling Web service methods from your scripts involves two high-level steps:

1. Creating a reference to the Web service. This includes registering the Web service with a variable name that you use in your Groovy code.
2. Writing Groovy code in Expression Builder that calls the Web service. For each call, the code must prepare the inbound arguments to the Web service, call a Web service method, and then process the return value from the Web service.
Creating a Web Service Reference

To register a Web service for use in your scripts, you first select **Web Services** in the Common Setup pane in Application Composer. You then associate a Web service variable name with a URL that provides the location of the Web Service Description Language (WSDL) resource that represents the service you want to call.

For example, you might register a Web service variable name of EmployeeService for a Web service that your application needs to invoke for working with employee data from another system. The URL for this Web service's WSDL might be:

```
http://example.com:8099/Services/EmployeeService?WSDL
```

Of course, the server name, the port number, and path name for your actual service will be different. If the port number is omitted, then it is assumed that the service is listening on the default HTTP port number 80.

Read "Creating Web Service References for Groovy Scripts: Explained" for more information about creating Web service references.

Writing Groovy Code to Call a Web Service

When you call a Web service from a Groovy script, the code must prepare the arguments to the Web service before calling a Web service method, and then process the data returned from the Web service. If your code passes structured data to and from a Web service, read "Using Groovy Maps and Lists with Web Services" below.

You insert the code for the call to the Web service from the **Web Services** tab in Expression Builder. As shown in the figure, the **Web Services** list displays the set of registered Web service variable names and the **Functions** list displays the available methods for a given Web service.

To insert a call to a Web service in a Groovy script.

1. Select the **Web Services** tab in Expression Builder.
2. Select a variable name from the **Web Services** list.
3. Select a method from the **Functions** list.

   The code that will be inserted is shown under **Function Signature**.

4. Click the **Insert** button to insert the code to invoke the Web service method.

As you can see in the figure, a Web service call from a Groovy script has the following syntax:

```
adf.webServices.YourServiceVariableName.MethodName(args)
```

The information under function signature includes the parameter types and also the return type to indicate the type of variable the result of the call should be assigned to. The possible return types are as follows:

<table>
<thead>
<tr>
<th>Return Value</th>
<th>Return Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void</td>
<td>Void</td>
</tr>
<tr>
<td>Scalar values (integer, string and so on)</td>
<td>The actual Java return type</td>
</tr>
<tr>
<td>Object</td>
<td>Map</td>
</tr>
<tr>
<td>Collection</td>
<td>List</td>
</tr>
</tbody>
</table>
For examples of Groovy code that calls a Web service, read "Web Service Calls in Groovy Scripts: Examples".

Using Groovy Maps and Lists with Web Services

When passing and receiving structured data to and from a Web service, a Groovy map represents an object and its properties. For example, an Employee object with properties named Empno, Ename, Sal, and Hiredate would be represented by a map object having four key-value pairs, where the names of the properties are the keys.

You can create an empty Map object using the syntax:

```groovy
def newEmp = [:]
```

Then, you can add properties to the map using the explicit `put()` method like this:

```groovy
newEmp.put("Empno", 1234)
newEmp.put("Ename", "Sean")
newEmp.put("Sal", 9876)
newEmp.put("Hiredate", date(2013,8,11))
```

Alternatively, and more conveniently, you can assign and update map key-value pairs using a simpler direct assignment notation like this:

```groovy
newEmp.Empno = 1234
newEmp.Ename = "Sean"
newEmp.Sal = 9876
newEmp.Hiredate = date(2013,8,11)
```

Finally, you can also create a new map and assign some or all of its properties in a single operation using the constructor syntax:

```groovy
def newEmp = [Empno : 1234,
             Ename : "Sean",
             Sal : 9876,
             Hiredate : date(2013,8,11)]
```

To create a collection of objects you use the Groovy List object. You can create one object at a time and then create an empty list, and call the list’s `add()` method to add both objects to the list:

```groovy
def dependent1 = [Name : "Dave",
                  BirthYear : 1996]
```
def dependent2 = [Name : "Jenna", BirthYear : 1999]
def listOfDependents = []
listOfDependents.add(dependent1)
listOfDependents.add(dependent2)

To save a few steps, the last three lines in the preceding example can be done in a single line by constructing a new list with the two desired elements in one line like this:

def listOfDependents = [dependent1, dependent2]

You can also create the list of maps in a single operation using a combination of list constructor syntax and map constructor syntax:

def listOfDependents = [[Name : "Dave", BirthYear : 1996], [Name : "Jenna", BirthYear : 1999]]

If the employee object in the previous codes examples has a property named Dependents that is a list of objects representing dependent children, you can assign the property using the same syntax as shown above (using a list of maps as the value assigned):

def newEmp.Dependents = [[Name : "Dave", BirthYear : 1996], [Name : "Jenna", BirthYear : 1999]]

Lastly, note that you can also construct a new employee with nested dependents all in a single statement by further nesting the constructor syntax:


For more information on maps and lists, see the section called Working with Maps in Oracle Sales Cloud Groovy Scripting Reference at http://www.oracle.com/pls/topic/lookup?ctx=cloud132&id=CGSAC130.

Related Topics
- Web Service References for Groovy Scripts: Explained
- Web Service Calls in Groovy Scripts: Examples

Working with Industry Solution Objects Using SOAP Web Services: Worked Example

Industry solution objects have SOAP Web service interfaces. In this example, you examine the Web Services Description Language (WSDL) file for the Sales Custom Business Object inbound Web service of your Oracle Sales Cloud application. This is the Web service an external application would use to query for, insert, update, or delete object instances.
You use the WSDL file to find and examine the XML Schema Document (XSD) file for an industry solution business object. This file lists all of the fields from the object that are available to the Web service. You examine the standard fields, and then add a custom field. Any fields you add to the object are dynamically added to the XSD file. You then retrieve the latest version of the XSD file to see the new field you added to the object.

**Working with Industry Solution Objects Using SOAP Web Services**

In this example, you will confirm that Web services are almost immediately updated when you create custom fields on industry solution objects, without requiring any work on your part. This makes it easy for inbound SOAP Web services to immediately leverage your customizations. All you have to do is provide the `<object>.xsd` file to your integrators or direct them to the custom business object service WSDL location.

To confirm Web services are updated:

1. Examine the URL of your application in your browser’s address bar to determine your machine name (for example, fap0092-crm.oracleads.com)
   
   Note that normally you would find this information by going to the Oracle Enterprise Repository, but we have provided the URL for your convenience.
3. Search for XSD on the page. Your search should find an entry similar to this:
   
   ```xml
   <schema>
   <import namespace="http://xmlns.oracle.com/apps/sales/custExtn/extnService/types/"
   </schema>
   ```

4. Copy the URL in the schemaLocation attribute and paste it into another browser window or tab. The schema definition loads in the browser and reveals the schema definition URLs for industry solution objects in addition to other schemas. Industry solution object names start with a double underscore __ and are easy to recognize.
5. Select the Sales Coach Setup object schema URL from the schemaLocation attribute and paste it into the browser window or tab. The Sales Coach Setup schema definition loads and displays the object fields available through the Web service. Scroll to the bottom of the schema and notice the fields that are available. Notice that a field named **My Text Field** is not in the list of available fields.

   ✨ **Note:** Do not close the XSD file. You will be looking at it again later.

6. Login to your Oracle Sales Cloud application.
7. Navigate to and launch Application Composer.
8. Select the Sales application from the Application List.
9. Expand the Custom Objects tree, then the Sales Coach Setup node.
10. Drill into the Fields node.
11. Create a new custom field My Text Field and view the schema again to verify that the schema has been dynamically updated. Schemas are updated when a sandbox is published or when a field is created outside a sandbox. Continuing this activity involves creating a field outside a sandbox. This is NOT recommended practice, but is necessary for this activity. If you continue, create a custom field. Once a field is created it cannot be deleted, but it will not appear in any pages unless you explicitly add it to a page.
12. Verify that you are not in a sandbox. To verify that you are not in a sandbox, check the global area on the page. If you were working in a sandbox, you would see a link to the sandbox details at the top of the page to the left of the Home page icon.
13. Next, create a new field named My Text Field. Click the **Custom** tab.
14. Click the **Create a custom field** button.
15. Click the **Text** option.
16. Click the **OK** button.
17. Enter the **Display Label** field value My Text Field.
18. Click the **Save and Close** button.
19. Next, confirm that your field appears on the list of custom fields for the Opportunity object. Go back to the XSD file and reload the page.
20. Use Find to verify that your custom field MyTextField_c now appears in the XSD file.
21. You have successfully confirmed that Web services are almost immediately updated when you create custom fields, without requiring any work on your part. This makes it easy for inbound Web services to immediately leverage your customizations.
8 Publishing Your Changes

Publishing Your Industry Solution Customizations: Explained

After completing the changes to your industry solution and testing your sandbox, publish the sandbox to make your changes available to your end users.

Prerequisites

Before publishing your sandbox, test your changes. Be careful when working with multiple sandboxes since changes in one sandbox could overwrite changes from another sandbox. When working with multiple sandboxes, always designate one sandbox as the integration sandbox that holds all changes from multiple authors and only publish from this one.

Publishing Sandboxes

To publish a sandbox:

1. Click your user name in the global area, and select Manage Sandboxes from the Administration menu.
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 metadata.
4. Close the Manage Sandboxes dialog box.
**Glossary**

**customization**
A change to the predefined artifacts of the application. Customizations impact multiple users.

**customization layer**
A level that represents the scope of users that a customization impacts. For example, all users or only those who meet specific criteria.

**data model**
The metadata that determines where data for a report comes from and how that data is retrieved.

**global area**
The region at the very top of the user interface that remains the same no matter which page you're on.

**mainline metadata**
The primary branch of metadata that a sandbox is published to. Once published, changes made in the sandbox become available to all users.

**sandbox**
A testing environment that isolates untested code changes from the mainline environment so that these changes don't affect the mainline metadata or other sandboxes.

**site layer**
Customizations made in this layer affect all users.