Build a Custom Portal with Oracle WebCenter

Istvan Kiss
Principal Curriculum Developer, Oracle WebCenter

Christina Kolotouros
Director, Oracle WebCenter Curriculum Development & Product Management

Ellen Gravina
Senior Principal Curriculum Developer, Oracle WebCenter
Introduction
Oracle WebCenter Suite, an offering in Oracle's Fusion Middleware product line, is designed to help you build more effective applications that take advantage of Service-Oriented Architecture (SOA) to bring complete context to the information worker's daily tasks. WebCenter Suite weaves process, business intelligence, structured and unstructured content, communication, and Web 2.0-style services into the very fabric of the application to create next generation online work environments.

Today we will focus on key new features of the PS3 release, including site navigation, site resource management (such as page templates), and deep content integration with Oracle UCM. This enables you to create and update polished Websites without programming.

You will work on the website for El Piju, a fictitious construction company specializing in natural building. Natural building is all about achieving sustainability through the use of minimally-processed, plentiful, and renewable resources as well as recycled or salvaged materials that produce healthy living environments.

Initially you will use JDeveloper to create a custom WebCenter portal application. El Piju’s Web designers have given you a custom page template for the Website. You will use this new template to create pages and page hierarchy, which will provide navigation for your application. You will also populate some of the pages using content from an Oracle Content Server.

Later you will switch to the running application and further refine the portal application at run time. You will learn that at run time you can create new pages, edit existing pages, upload or create new content and edit existing content stored in the Content Server. You will also see, how security roles and access privileges can be defined, or modified at run time.

Create a WebCenter Portal Application

1. Start the Content Server and related components.
   a. In the Linux desktop, right-click an empty area and select Open Terminal from the pop-up menu.

   ![Open Terminal](image)

   A new terminal window opens.
   b. In the terminal window type `vmctl`, then in the Oracle WebCenter

Hands-On Lab: Build a Custom Portal with Oracle WebCenter p-2
Administration menu choose option 1 (one) to start all the required components:

- Oracle Database and TNS Listener
- WebLogic Administration Server
- Managed WebLogic server instance UCM_server1, that hosts Oracle Content Server.

c. Starting the Content Server will take several minutes. However you don’t have to wait till it fully starts, you can continue with the following steps.

2. Start JDeveloper.

Double click the JDeveloper icon on the desktop.

3. Create a new WebCenter Portal application.
   a. Bring up the new application wizard by selecting **New Application** under the **Application Navigator**.
b. In the Application Name field, enter ElPijuPortal. Note this also updates the Directory. For completing the practices it is important that you give the above – case sensitive – name for your application. Navigate to the bottom and select WebCenter Portal Application.

c. Click Finish, since we do not want to change the application’s default settings.

4. In this screen you see the resulting application, with two projects. We will use the Portal project to create the application, while the PortalWebAssets project can be used for the static resources, like images, HTML, or JavaScript files, etc.
You can close the EIPijuPortal Overview pane.

5. Start JDeveloper’s integrated WebLogic Server.

Select **Start Server Instance (IntegratedWebLogicServer)** from Run menu.

As the server starts, it will display status messages to the Log pane at the bottom of the JDeveloper’s window. The server fully started, when you see the message:

*IntegratedWebLogicServer Started*

However you don’t have to wait for the server, keep on doing the following steps.
6. Change the Portal project’s properties.
   
a. Make sure that the Portal project is selected. In the Application Navigator, click the Project Properties icon.

   ![Application Navigator screenshot](image1.png)

   b. Select **Java EE Application** properties and change the Java EE Application Name to **ElPijuPortal** and the Web context root to **elpiju**.

   ![Project Properties screenshot](image2.png)

   c. Click **OK** to close.

7. Create a Content Repository Connection.
Corporate documents are stored in an Oracle Content Server. Since we would like to display and manage these documents in our portal, we need to create a connection that describes how the Content Server can be accessed.

a. Select New from File menu and choose Content Repository from Connections. Click OK.

b. Create a Content Repository connection with the following attributes:

<table>
<thead>
<tr>
<th>Create connection in</th>
<th>Application Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Name</td>
<td>ElPijuDocuments</td>
</tr>
<tr>
<td>Repository Type</td>
<td>Oracle Content Server</td>
</tr>
<tr>
<td>Set as primary connection</td>
<td>Selected</td>
</tr>
<tr>
<td>RDIC Socket Type</td>
<td>socket</td>
</tr>
<tr>
<td>Server Host Name</td>
<td>localhost</td>
</tr>
<tr>
<td>Content Server Listener</td>
<td>4444</td>
</tr>
<tr>
<td>Authentication</td>
<td>Identity Propagation</td>
</tr>
</tbody>
</table>
In the corporation, Web designers created a new page template to provide a corporate-specific, uniform layout for the portal application. This template was developed using JDeveloper, then exported and packaged into a site resource archive.

8. Import the template.
   a. In JDeveloper’s Application Navigator right-click on Portal project and select **Import Portal Resource** from the pop-up menu.
b. Select `/home/oracle/HOL/ElPijuTemplate.ear` and click **OK**.

c. When the import is finished, you can find the template file in the Application Navigator, under the **Portal > Web Content > oracle > webcenter > portalapp > pagetemplates** node.
d. You have to register the imported template with the application. You do this by editing the `pagetemplate-metadata.xml` file. Using Application Navigator to select the file.

Double-click the file name or select open after right-clicking the name. In this file there are already two templates registered. Add a third element to the `pageTemplateDefs` element:

```xml
<pagetemplate-jsp-ui-def>
  /oracle/webcenter/portalapp/pagetemplates/ElPijuTemplate.jspx
</pagetemplate-jsp-ui-def>
```
9. Set the newly uploaded template as the default template for the already existing pages.
   a. In the Application Navigator pane, locate **Application Resources > Descriptors > ADF META-INF > adf-config.xml** and open it by double-clicking on the name.

   ![Application Navigator Screen](image)

   b. If the file is opened in Overview mode, switch to the source mode by clicking on the **Source** tab at the bottom of the viewer pane.

   ![Source Tab](image)

   c. In the search bar at the top of the viewer pane, under **adf-config.xml**, type **_globe** to find the current default page template setting in this file. You should see the following preference setting:

   ```xml
   <portal:preference
       id="oracle.webcenter.portalapp.pagetemplate.pageTemplate"
       desc="Default Page Template"
       value="/oracle/webcenter/portalapp/pagetemplates/pageTemplate_globe.jspx"
       resourceType="Template" display="true"/>
   ```

   Change the highlighted value to: **ElPijuTemplate.jspx**

   d. You should now see the following:
Create and Populate Pages

   a. In the Application Navigator expand the node **Portal > Web Content > oracle > webcenter > portalapp > pages** node. Here you can find the portal application’s initial pages: error.jspx, home.jspx, and login.jspx.

   ![Application Navigator](image1.png)

   b. Right click on **pages** and select **New** from the pop-up menu.

   ![Application Navigator](image2.png)

   c. In the New Gallery, select **Web Tier > JSF** and choose **JSF Page**.
d. In the Create JSF Page dialog, make sure that the **Create as XML Document** checkbox is selected. Set the file name to `news.jspx`, select the **Page Template** radio button, and make sure that **ElPiju Default Page Template** is selected in the drop-down list. Click **OK**.
e. After the page is created, it opens in the editor pane. You can close this pane, we will not edit the news page.

f. Similarly, create another page called `press_releases.jspx`. Keep the editor pane open, and switch to the **Source** view mode, if necessary.

11. Define the page hierarchy.
   All the pages of the portal application are stored in the same folder: `/oracle/webcenter/portalapp/pages`. However, you can define a hierarchy of these pages. In our portal, the page hierarchy will drive the navigation, i.e., define how users can access the pages.
   a. Open `pages.xml` under the `pagehierarcy` node in Overview mode. In the initial page hierarchy, only the Home page is defined.
   b. Select `news.jspx` in the Application Navigator. Drag the file and drop it on top of the Root node in the hierarchy.
c. Similarly, drop **press_releases.jspx** on top of news page, making it a sub page of the news page. Then select each page in the hierarchy and change the titles of the pages to **News** and **Press Releases**, respectively. This is the resulting page hierarchy:

12. Add Content to the Press Releases page.

This is how the Press Releases page will look in the finished portal. You can see the corporate page template, with header and footer area and left-side navigation. This will be identical in all of the pages. The page-specific content is in the red box, as indicated in the screenshot. You will add the content to the page: on the left side you will add an accordion, displaying documents from the Content Server, on the right, you will add some static images, also stored in the Content Server. In order to this, you need to add content to the page.
a. Make sure that `press_releases.jspx` is opened in the Editor pane. We recommend that you work in Source view mode.

b. In the Component Palette, scroll to select the **Oracle Composer** group.

c. Drag the **Page Customizable** component and drop it on the `<f:facet name="content"/>` tag.
This is the resulting code fragment.

```xml
<af:document id="d1">
  <af:pageTemplate value="/oracle/webcenter/portalapp/pagetemplate.html" id=""/>
  <af:facet name="content">
  <pe:pageCustomizable id="pageCustomizable1">
    <cust:panelCustomizable id="panelCustomizable1" layout="scroll" />
    <f:facet name="editor">
      <pe:pageEditorPanel id="pep1"/>
    </f:facet>
  </pe:pageCustomizable>
  </af:facet>
</af:document>
</jsp:view>
</jsp:root>
```

Delete the highlighted `panelCustomizable` element; we will not use it.

d. Drop the Layout Customizable component on top of the `<pe:pageCustomizable>` tag.

e. To select the `<pe:layoutCustomizable ...>` tag, click somewhere in the tag name. The Property Inspector pane will display the attributes of the Layout Customizable tag. Change two properties: set Type to `twoColumnNarrowRight` and ShowIcon to `false`.
This is the resulting page fragment:

```xml
<pe:pageCustomizable id="pageCustomizable1">
  <pe:layoutCustomizable id="layoutCustomizable1" type="twoColumnNarrowRight" showIcon="false">
    <cust:panelCustomizable id="panelCustomizable1" />
    <f:facet name="contentA">
      <cust:panelCustomizable id="panelCustomizable2" />
    </f:facet>
    <f:facet name="contentB">
      <cust:panelCustomizable id="panelCustomizable3" />
    </f:facet>
  </pe:layoutCustomizable>
</pe:pageCustomizable>
```

Note that by adding a Layout Customizable element, you also got three <cust:panelCustomizable ...> elements, which would hold the actual content. Since we’ve decided to use a two-column layout, you will use only the first two Panel Customizable elements.

f. Select the first Panel Customizable element and use the Properties Inspector pane to set the width property to 600 px. You can either type the width setting to the **InLineStyle** attribute as `width:600.0px;` or use the Box’s **Width** attribute.
g. Similarly, select the second Panel Customizable element – the one in the facet called contentA. Change its width to 194 pixels.

h. In JDeveloper’s Application Navigator pane, expand the Application Resources > Connections > Content Repository > ElPijuDocuments node. When prompted for Connections Credentials, use weblogic for the username and welcome1 for the password.

Further expand the node Contribution Folders > El Piju > Site > right-side-pics > PressReleases.
Drag the first image from this folder and drop it on the second Panel Customizable element. When you release the mouse, JDeveloper will ask you how to insert the image to the page. Choose Create > ADF Image.

This is the resulting code fragment.

```xml
<f:facet name="contentA">
  <cust:panelCustomizable id="panelCustomizable3" inlineStyle="width:194.0px;">
    <af:image
```
Add the other two images from the folder to the same Panel Customizable component.

i. Expand the node **Contribution Folders > El Piju > Site > News**. Drag the **PressReleases** folder and drop it on the first Panel Customizable. Choose **Documents – Content Presenter** from the pop-up menu.

In the Edit task Flow Binding dialog, enter in the templateView field **oracle.webcenter.content.templates.default.list.accordion**.

This is the resulting code fragment. Note that the task flow’s parameters are not
visible in the source; they are stored separately, in the page definition file.

```xml
<cust:panelCustomizable id="panelCustomizable2"
  inlineStyle="width:600.0px;">
  <af:region
    value="#{bindings.doclibcontentpresenter1.regionModel}" id="r1"/>
</cust:panelCustomizable>
```

j. Save all pending changes. Click the Save All icon in the tool bar.

13. Add pre-created pages.

When developing a custom portal application, you will be creating many more pages and you have to populate them. This can be done at design time using JDeveloper, as you have seen it in the previous steps, or at run time, as you will do later in this practice.

In order to save time during the practice, we have created several pages and packaged them in a ZIP archive. Next you will unzip the archive over the folder where JDeveloper keeps the application’s source.

a. Minimize JDeveloper window if needed, but do not close it.

b. Select the previously used terminal window, or open a new one as it is described in step #1/a.

c. Type the following command in the terminal:

```
unzip -d $JDEV_USER_DIR/mywork/home/oracle/HOL/Pages.zip
```

The program will warn you that it is going to replace existing files. Answer All to replace all files without further warnings.

d. Return to the JDeveloper window. If there is a warning that files were externally modified, click OK to accept it.

e. In the Application Navigator pane, select the Portal project. Use the Refresh icon or choose Refresh from the View menu. Similarly refresh the PortalWebAssets project.

f. Open the page hierarchy description file: Portal > Web Content > oracle > webcenter > portalapp > pagehierarchy > pages.xml. As you expand the nodes
observe that several new pages have been added to the project and placed into the page hierarchy.

**Page Hierarchy**

Drag ADF pages and drop them in the page hierarchy

---

### Run the Portal Application and Explore the Pages

14. Run the portal application.
   
   a. Select Portal under Projects and click the Run icon.

   ![Application Navigator](image)

   **Oracle JDeveloper 11g Release 1 - E**
   
   **Application Navigator**
   
   **EPIjuPortal Overview**

   **Java Files**
   
   XML Files
   
   Offline Databases
   
   Page Flows
   
   Web Pages

   ![Run Icon](image)

   b. JDeveloper will first compile the project. If it is successful, it will start its integrated WebLogic Server, which is used to test Web applications. After the server is started, JDeveloper will deploy the current project as a Web application.

   c. As a last step, JDeveloper will start a browser and point it to the URL of the currently deployed application, in our case:

   ```
   http://127.0.0.1:7101/elpiju
   ```
d. The browser will be redirected to the portal’s home page.

15. Explore the page navigation.
   a. Note that the header of the page contains a navigation component, which displays the root-level pages from the page hierarchy. Home does not have any subpages; it can be accessed directly. When a page has subpages, its name is displayed with a down-arrow icon. Clicking the name will show all the subpages.

   b. Open the Services > Construction page.
Hands-On Lab: Build a Custom Portal with Oracle WebCenter p-25

Each of the subpages is similar to the Home page. The page header, including the main navigation bar, and the page footer is the same. But in addition to these components, each subpage has a banner image in which the name of the root page is displayed. There is also a navigation component on the left side of the page, which displays the root page name, and all of its subpages, as links. Therefore, from a subpage, all of the sibling pages can be accessed with a single click.

16. Return to the Home page. You can click either the **Home** link in the top navigation bar or the corporate logo in the top left corner of the header.

### Edit a Page

17. Click **Login** and log in as user **weblogic** with password **weblogic1**.

Note that weblogic is a special user, created out of the box for your portal application. As you will see, weblogic has administrator privileges throughout the application. This is how the Home page is displayed for weblogic.
Hands-On Lab: Build a Custom Portal with Oracle WebCenter p-26
c. Click the **Edit Properties** icon. This will display the Box component’s properties. Change the text for the two tabs to **Wheat Paste** and **Earthbag vs. Adobe**, respectively. Click **Apply** and observe the changes before clicking **OK**.

![Component Properties: Box](image)

19. Add Content Presenter to a tab.

![Content Presenter](image)

a. Make sure that the tab with the title **Wheat Paste** is selected. Click **Add Content**. This will open the Business Dictionary, where you can find all the components that can be added to your page. The available components are arranged in folders.
b. Open the **Content Management** folder and click **Add** in the row of the Content Presenter task flow. **Close** the Business Dictionary.
20. Select the content and a template.
   a. Click the first **Edit** icon. In the Content Presenter Configuration Window, select the **Content** tab and click the **Browse** button.
b. In the folder hierarchy, drill down to the Articles folder, select the document about wheat paste, and click Select.

c. There is no need to change the Content Presenter template; we will use the default viewer. Save the changes in the configuration dialog.
21. If you have less than 10 minutes left, move on to the next step. Otherwise repeat steps 19 and 20 for the other tab, adding the document that compares earthbag and adobe construction methods.

22. Click Close at the top right corner of the page. This is how the resulting page looks.

### Edit Content

In this part of the practice, you will learn two ways of editing content stored in Oracle Content Server: direct, in-context editing and uploading documents using Document Manager.

23. Edit the document comparing earthbag and adobe construction methods.
   
   a. Make sure you are viewing the **Articles** page and select the **Wheat Paste** tab.
   b. Activate the in-context editing mode by using **CTRL-SHIFT-C**. You will see a dashed line around the document with Edit and Refresh icons in the top right corner.
Making Wheat Paste

Wheat paste is often used in earthen plasters, floors, finishes, and aisles that need a little extra ‘glue.’ It adds stickiness and durability. It can keep earthen plaster surfaces from dusting off. We’ve also used it for sticking bike posters to the L.A. landscape ;) It’s super easy to make!

Recipe:
2 cups of flour + 2 quarts of water or 1 gallon of water + 4 cups of flour

Add the flour to a little cold water while whisking it in. Add just enough to make a soup.

Add water and flour concoction to boiling water and turn to low heat or off. Stir until the mix starts to turn clear.

You can send it through a sieve to remove lumps for fine finishes.

Store the wheat paste in a clean jar with a tight-fitting lid. The paste will keep for a few weeks if refrigerated.

Michael Blaha

c. Click the View and manage this document (a folder and a document) icon. The document displays in a pop-up Document Manager window. Click Edit.
   a. Open the News > Press Releases page. You should see 3 documents in the accordion display.
b. Open the News > News Manager page. This contains a Document Manager task flow, which points to the News folder in the Content Server.

c. Drill down to the PressReleases folder. Click Upload and upload the Cob Workshop file (201100919 Cob Workshop.html) from /home/oracle/HOL folder. The folder on the Content Server should now contain four documents.
d. Return to the News > Press Releases page. View the newly uploaded document in the accordion display.

25. Congratulations, you have finished! It is a good idea to shutdown your services before exiting your VM. Follow steps 1a and 1b from the beginning of this document, but chose option 0 (zero) this time.

26. If you would like to use this VM to do this exercise again, you should delete the ElPijuPortal application that you created. In your terminal session do the following:

   a. > cd $JDEV_UDER_DIR/mywork
   b. > rm -rf ElPijuPortal