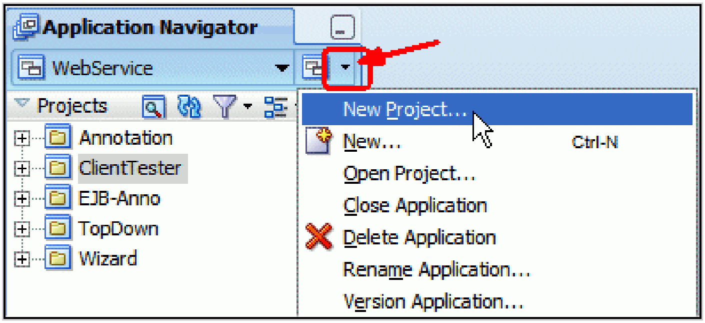
Part 6: Developing an ADF Client Using a Web Service Data Control

Web services allow enterprises to expose business functionality irrespective of the platform or language of the originating application. The business functionality is exposed in such a way that it is abstracted to a message composed of standard XML constructs that can be recognized and used by other applications. Oracle ADF provides an access layer that can be used to access many types of business or data services, include Web Services.

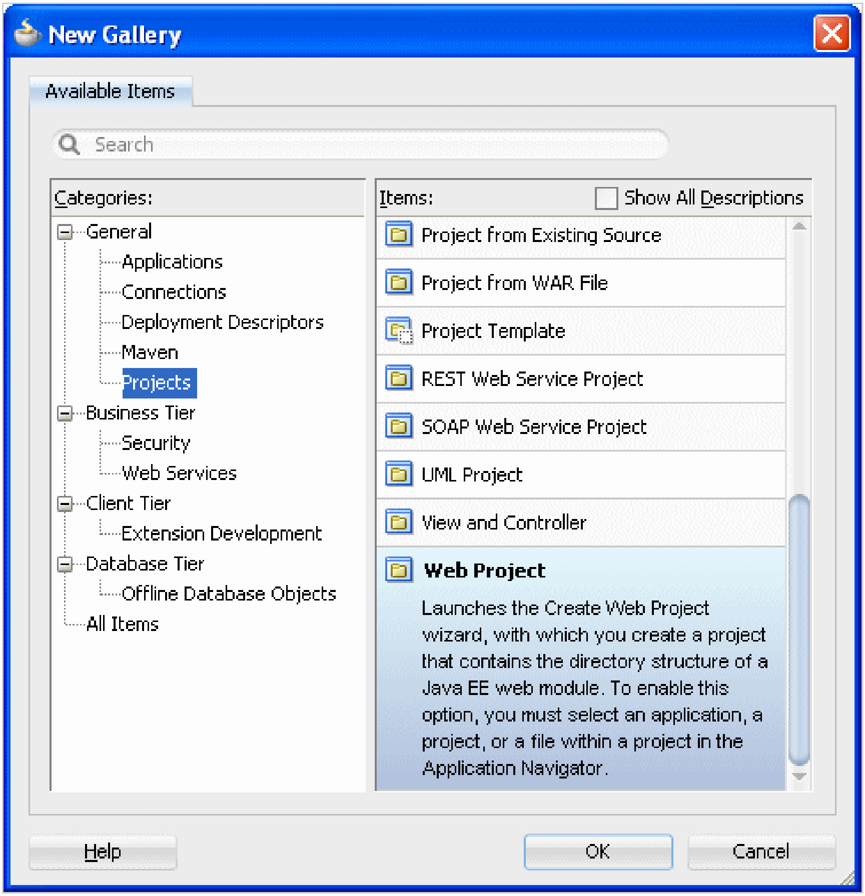
Step 1: Creating the Web Service Data Control

In this section you use the WSDL deployed to the integrated server and create a data control from it. Your application will not contain the original Java code as in the previous scenarios; it creates the data control from the deployed WSDL. Once you create the data control, any ADF client application can use it.

1. In the Application Navigator, click the **down-arrow** to the far right of the Application name and select **New Project** from the menu.

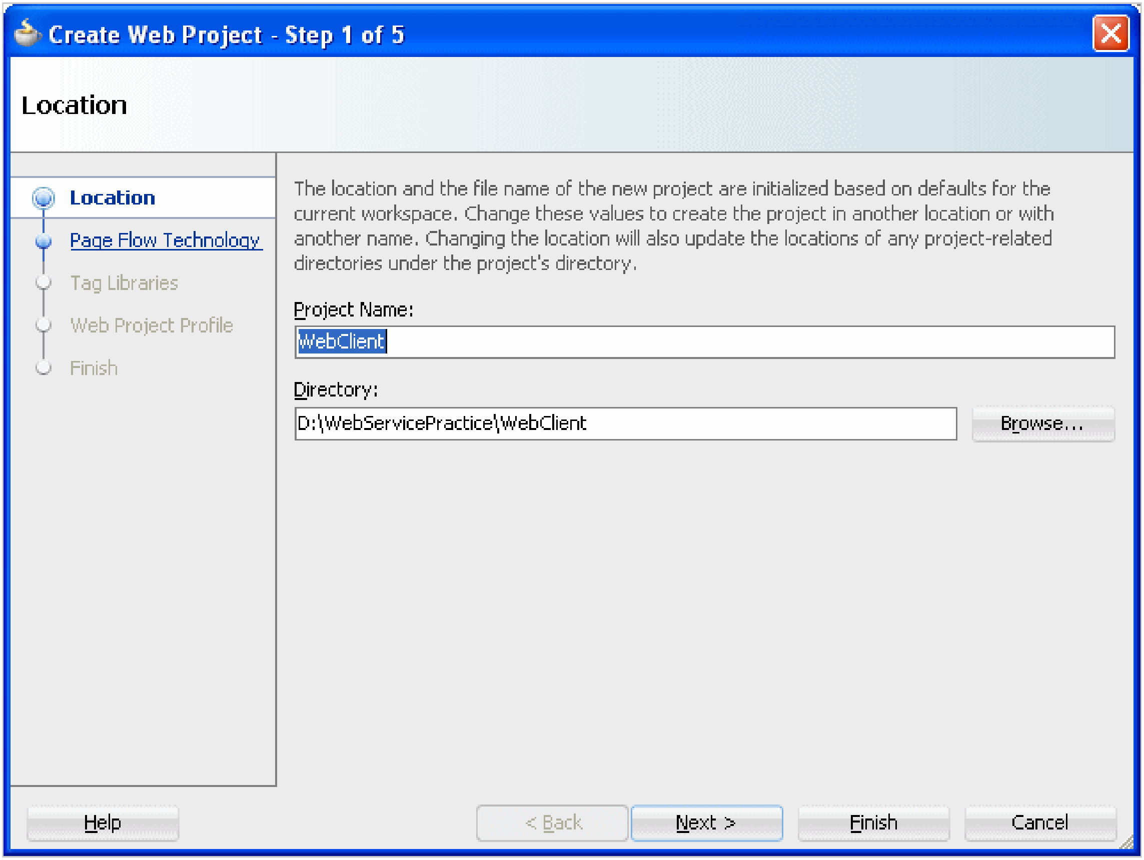


2. In the new Gallery, select **Web Project** and click **OK**.



Launches the Create Web Project wizard, with which you create a project that contains the directory structure of a Java EE web module. To enable this option, you must select an application, a project, or a file within a project in the Application Navigator.

3. In the Create Web Project wizard, name the project **WebClient**. Click **Next** and **Next** again.



Create Web Project - Step 1 of 5

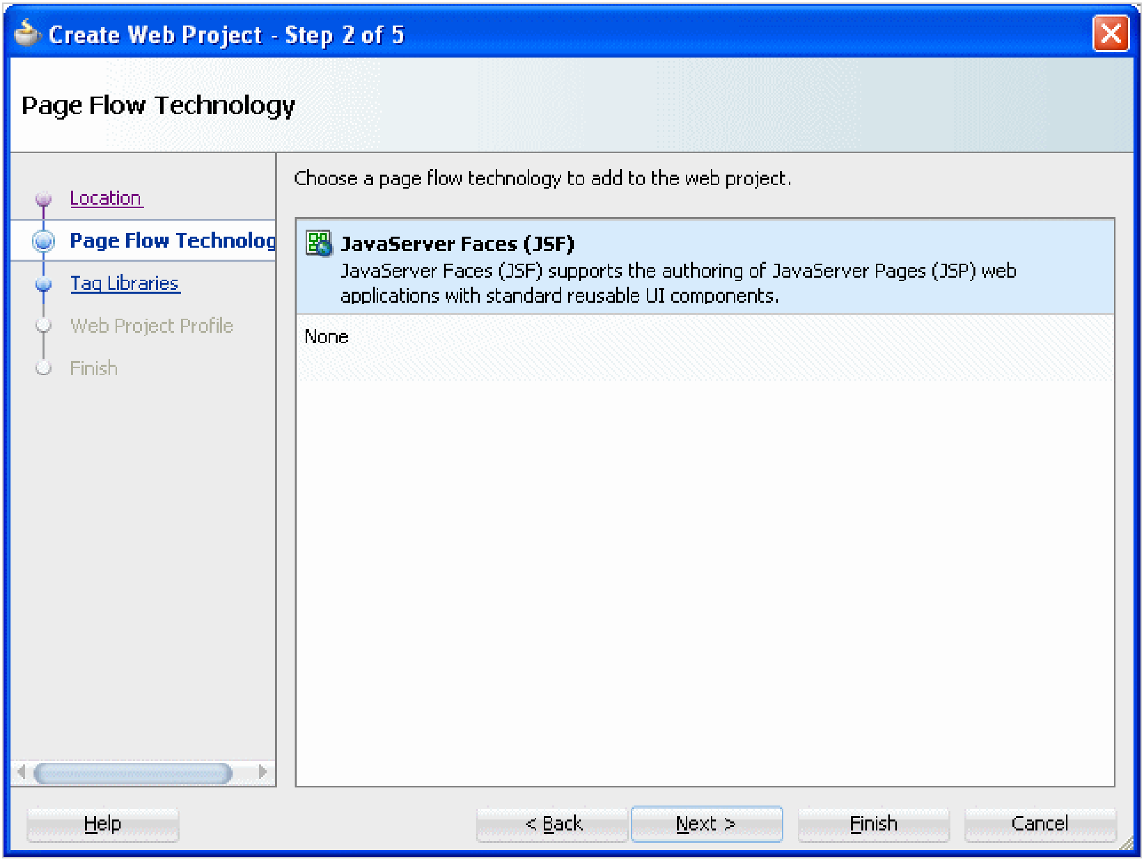
Location

The location and the file name of the new project are initialized based on defaults for the current workspace. Change these values to create the project in another location or with another name. Changing the location will also update the locations of any project-related directories under the project's directory.

Project Name: WebClient

Directory: D:\WebServicePractice\WebClient

4. In the Page Flow Technology step of the wizard, select **JavaServer Faces (JSF)**, and click **Next**.



Create Web Project - Step 2 of 5

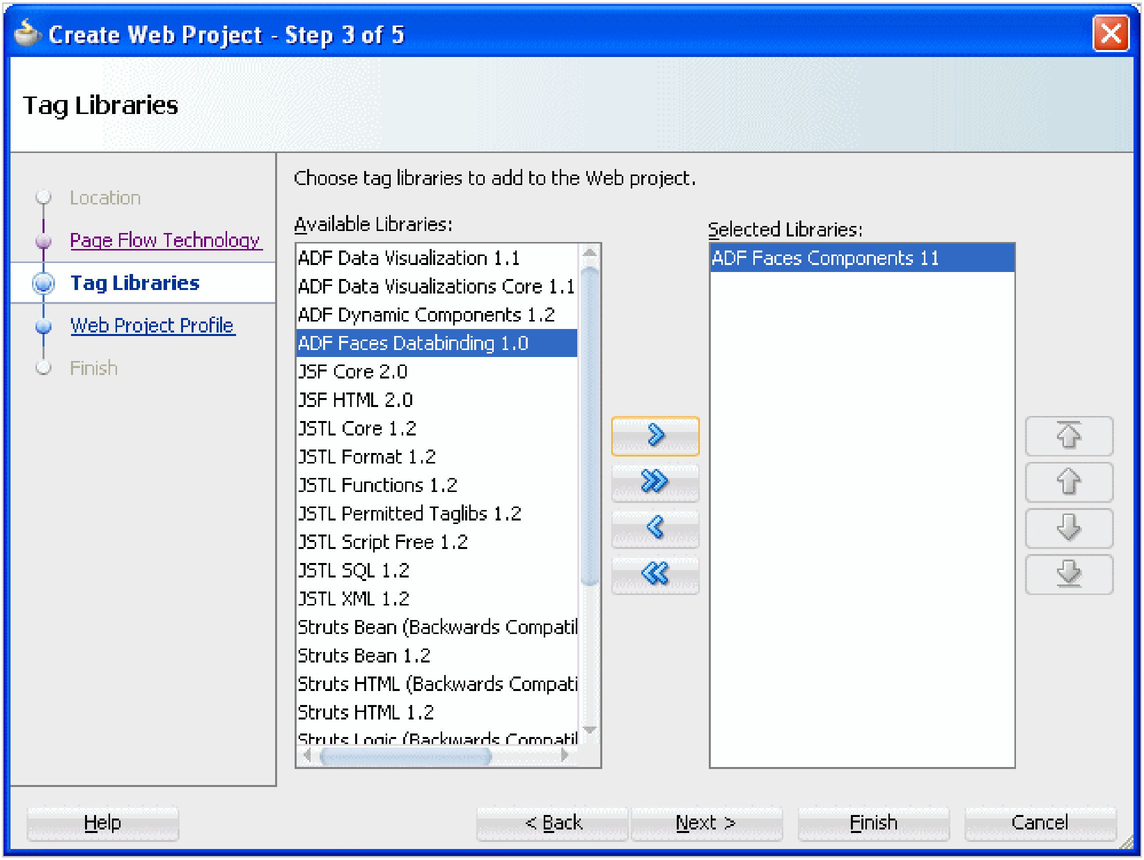
Page Flow Technology

Choose a page flow technology to add to the web project.

JavaServer Faces (JSF)
JavaServer Faces (JSF) supports the authoring of JavaServer Pages (JSP) web applications with standard reusable UI components.

None

5. In the tag libraries, shuttle the **ADF Faces Components 11** from Available Libraries to Selected Libraries


Create Web Project - Step 3 of 5

Tag Libraries

Choose tag libraries to add to the Web project.

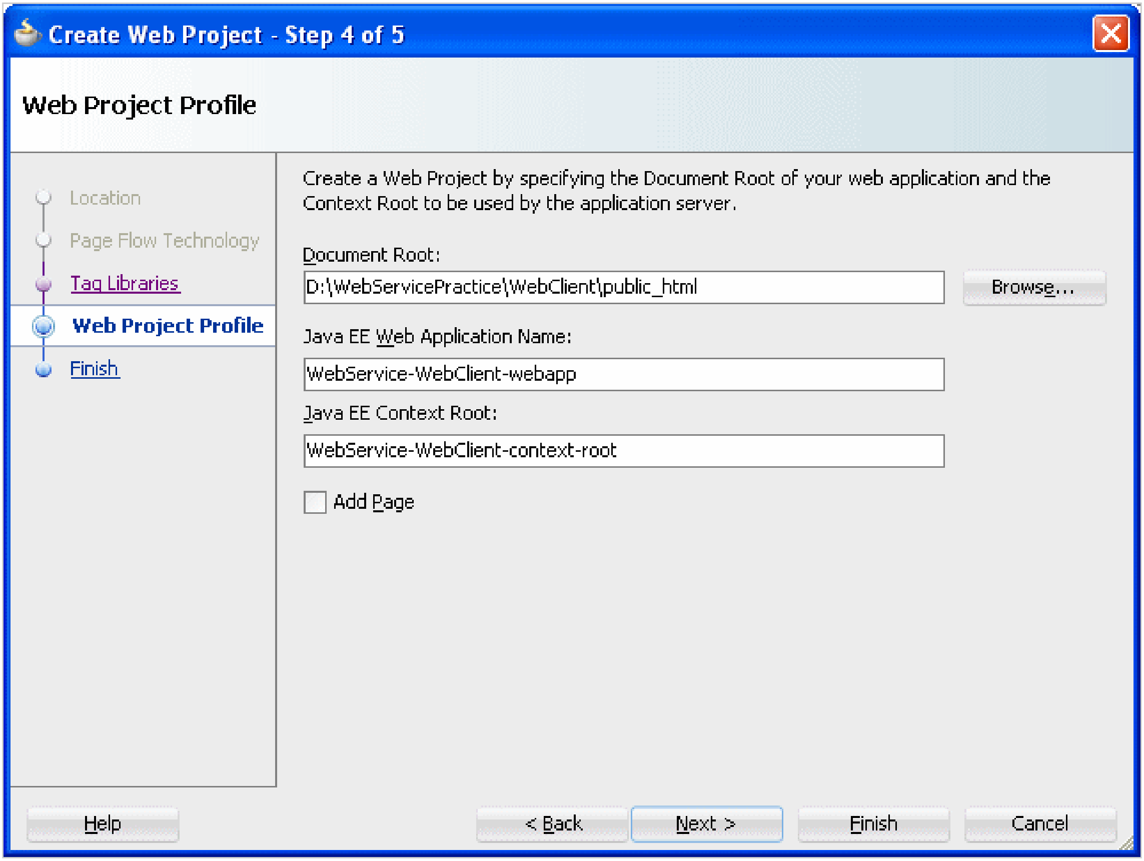
Available Libraries:

ADF Data Visualization 1.1
ADF Data Visualizations Core 1.1
ADF Dynamic Components 1.2
ADF Faces Databinding 1.0
JSF Core 2.0
JSF HTML 2.0
JSTL Core 1.2
JSTL Format 1.2
JSTL Functions 1.2
JSTL Permitted Taglibs 1.2
JSTL Script Free 1.2
JSTL SQL 1.2
JSTL XML 1.2
Struts Bean (Backwards Compatil
Struts Bean 1.2
Struts HTML (Backwards Compati
Struts HTML 1.2
Struts Logic (Backwards Compatil

Selected Libraries:

ADF Faces Components 11

6. Click **Next**.


Create Web Project - Step 4 of 5

Web Project Profile

Create a Web Project by specifying the Document Root of your web application and the Context Root to be used by the application server.

Document Root: D:\WebServicePractice\WebClient\public_html

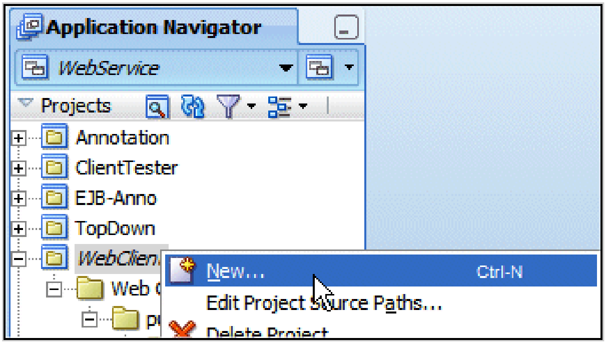
Java EE Web Application Name: WebService-WebClient-webapp

Java EE Context Root: WebService-WebClient-context-root

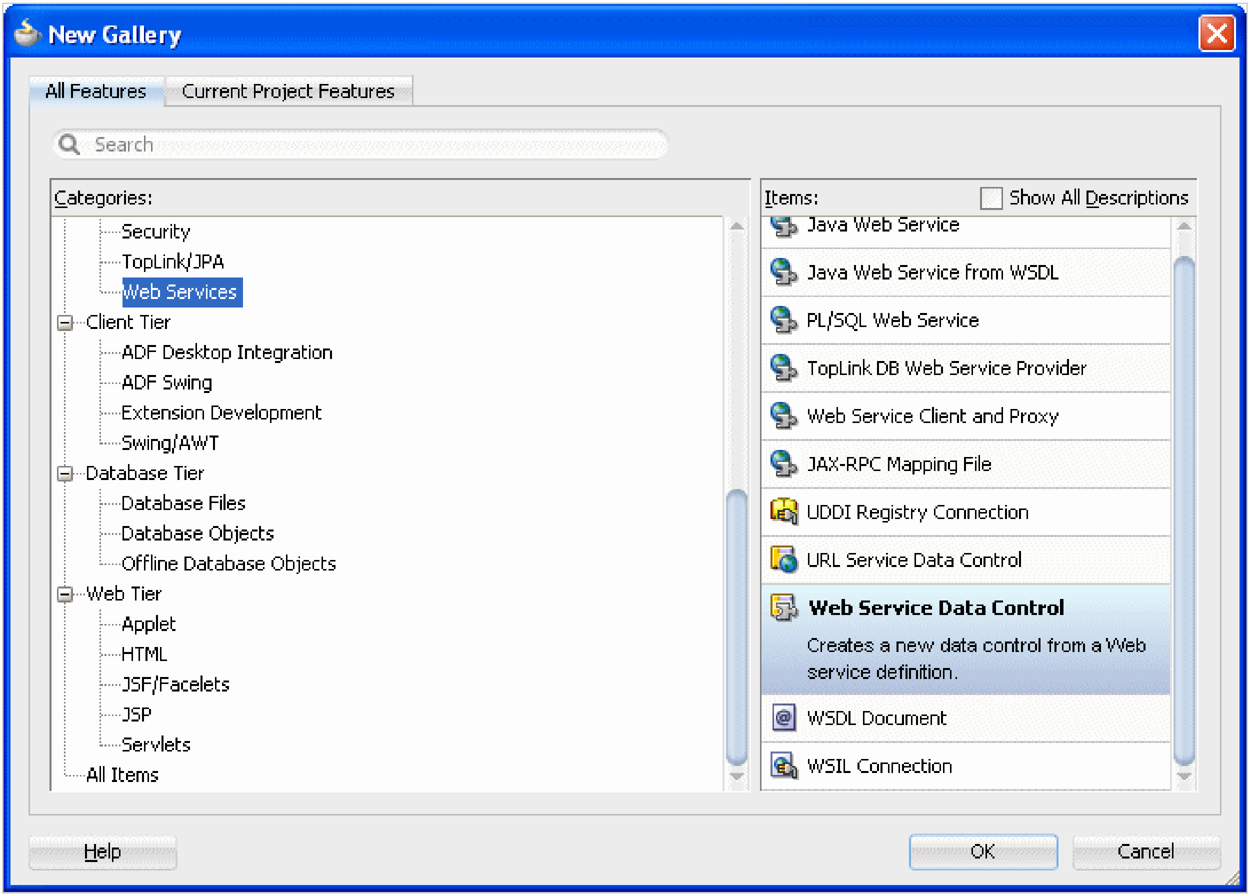
□ Add Page

7. Click **Next** and **Finish**.

8. In the Application Navigator, right-click the **WebClient** node and select **New** from the context menu.

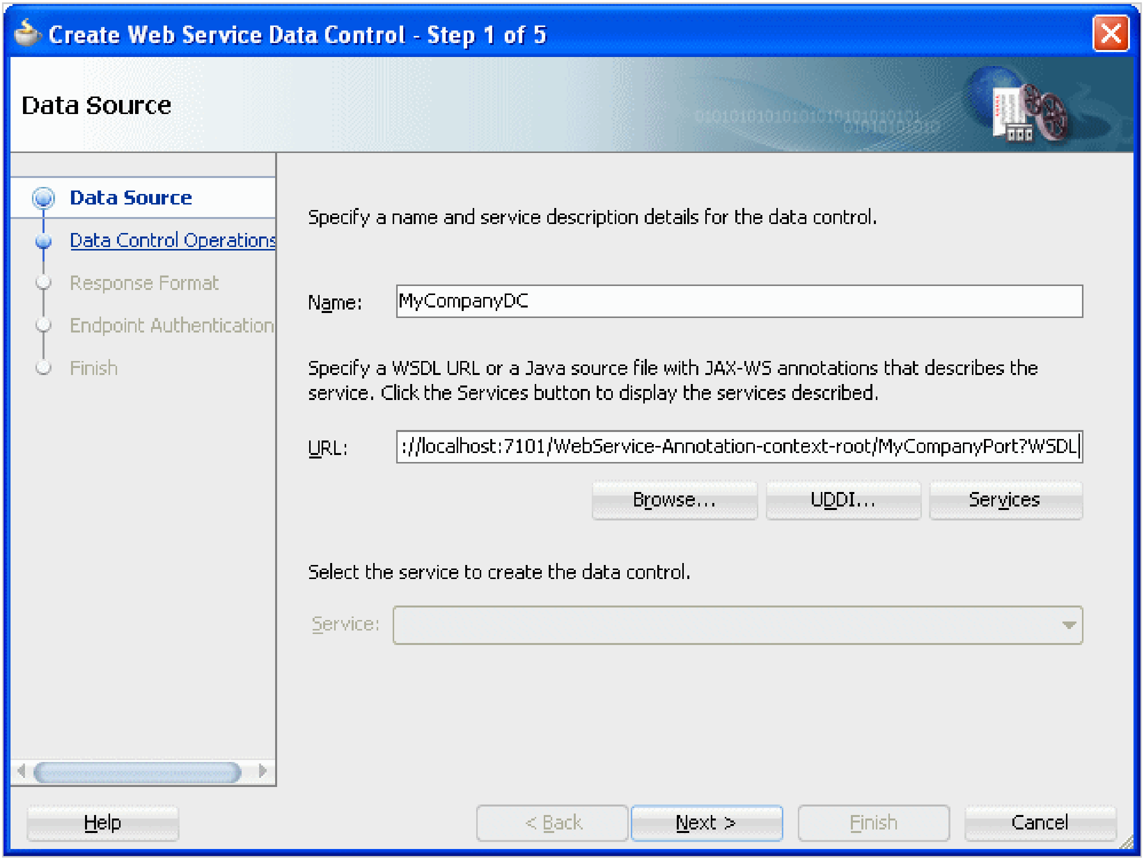


9. In the New Gallery, in the **Business Tier** select the **Web Services** node and choose the **Web Service Data Control** in the Items list and click **OK**.



10. In the first step of the Create Web Service Data Control wizard, name the data control **MyCompanyDC**, and then paste the WSDL URL that you copied earlier:
http://localhost:7101/WebService-Annotation-context-root/MyCompanyPort

As before, append **?WSDL** to the URL. When you press [Tab], the web service for the data control is populated.



Specify a name and service description details for the data control.

Name: MyCompanyDC

Specify a WSDL URL or a Java source file with JAX-WS annotations that describes the service. Click the Services button to display the services described.

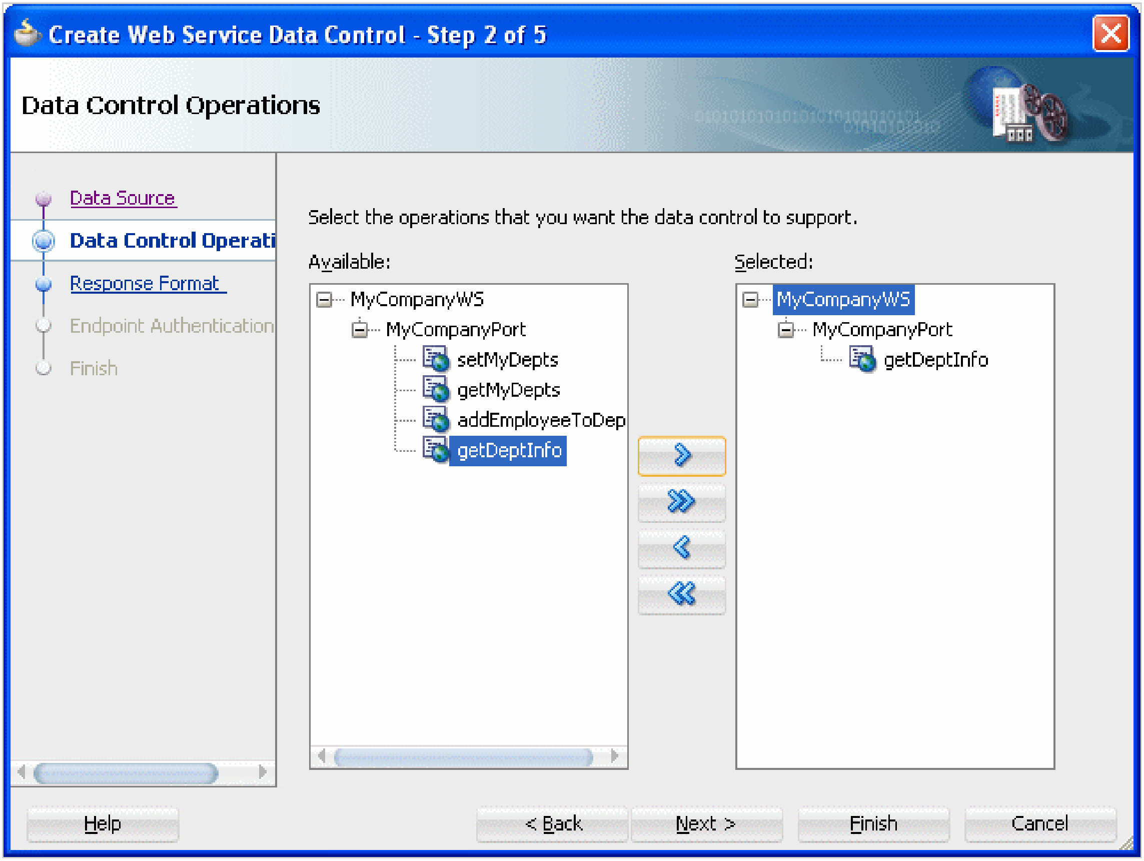
URL: ://localhost:7101/WebService-Annotation-context-root/MyCompanyPort?WSDL

Select the service to create the data control.

Service:

11. Click **Next**.

12. In the Data Control Operations page, shuttle the **getDeptInfo** method to the Selected side.



Select the operations that you want the data control to support.

Available:

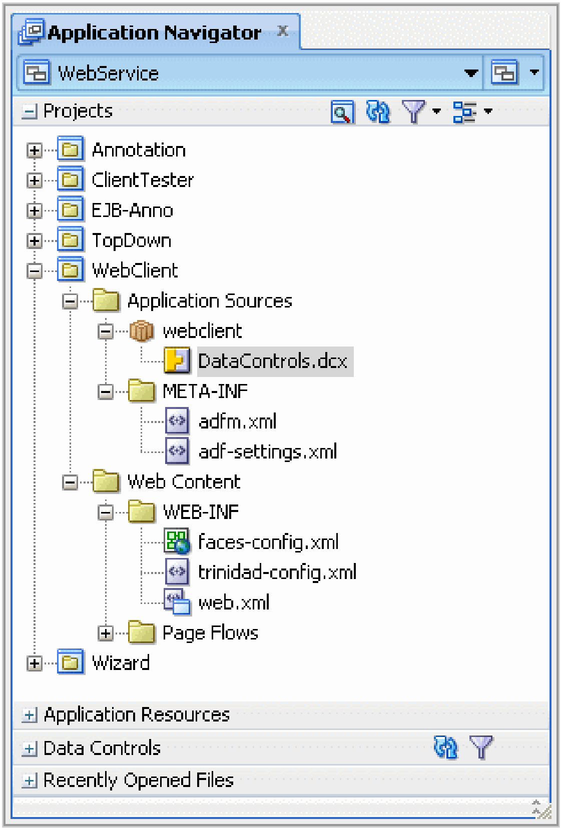
```
MyCompanyWS
  MyCompanyPort
    setMyDepts
    getMyDepts
    addEmployeeToDep
    getDeptInfo
```

Selected:

```
MyCompanyWS
  MyCompanyPort
    getDeptInfo
```

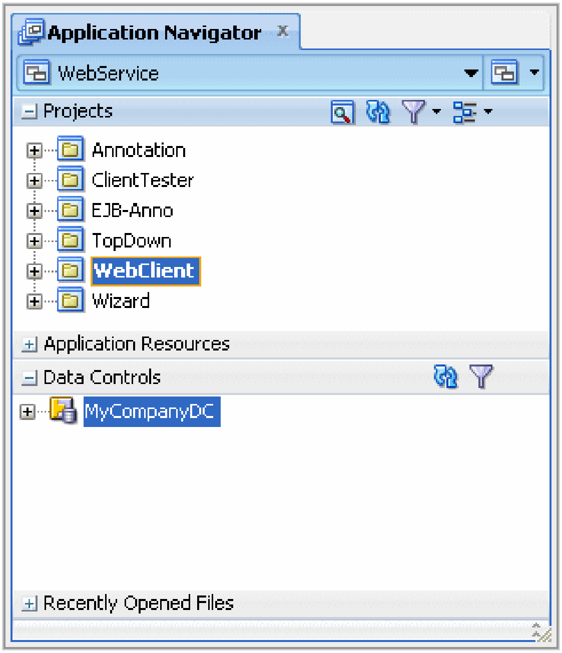
13. Click **Next**, then **Finish**.

The Application Navigator should look something like:



14. Click **Save All** to save your work.

15. Expand the **Data Controls** accordion in the Application Navigator to expose the **MyCompanyDC** data control that has just been created.

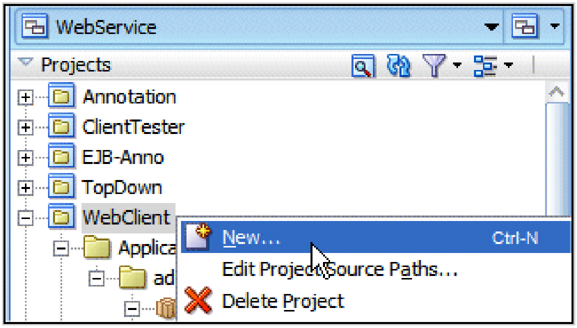


At this point you are ready to create the JSF page and use the data controls.

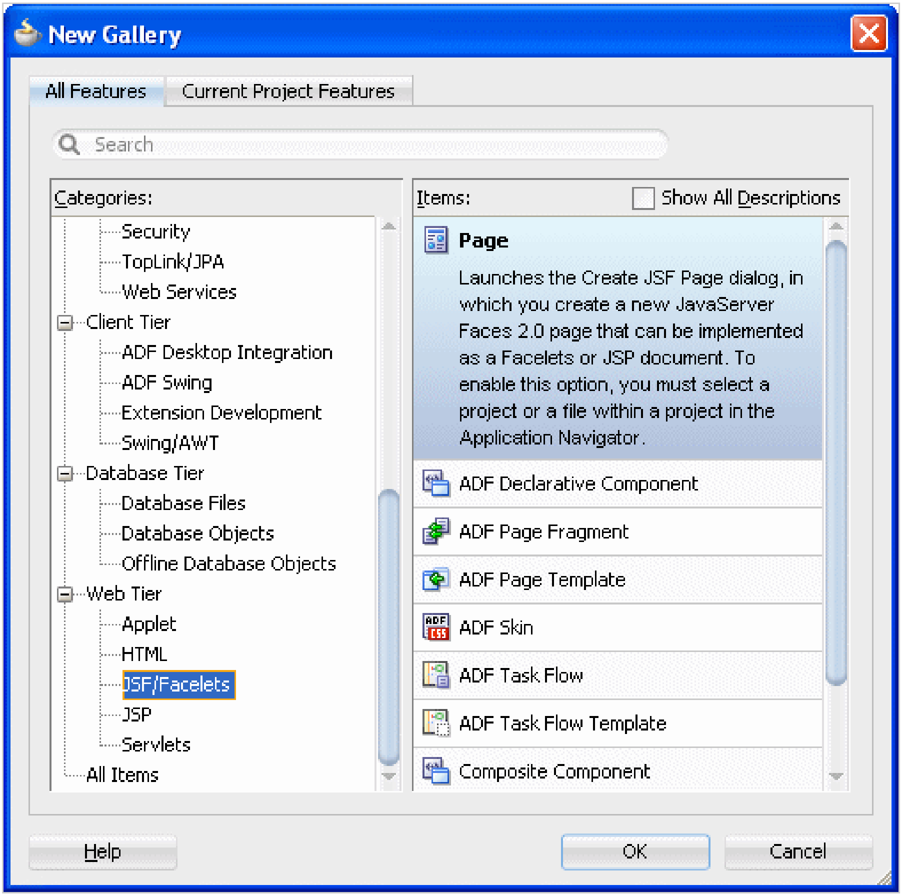
Step 2: Creating a JSF Page to Consume the Web Service Data Control

In this section you update the client class to invoke the web service proxy and return the result to the message window.

1. Invoke the **New Gallery** from the **WebClient** project.

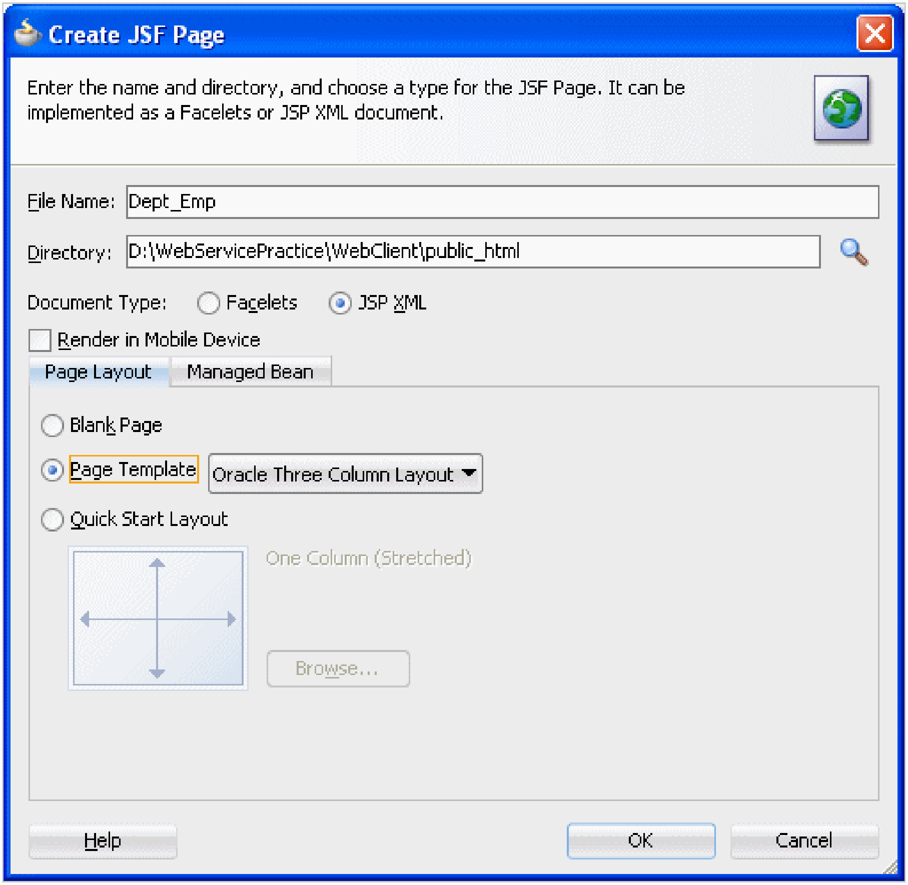


2. Expand the **Web Tier** node and select the **JSF / Facelets** sub node. In the Items column, select **Page** and click **OK**.



Launches the Create JSF Page dialog, in which you create a new JavaServer Faces 2.0 page that can be implemented as a Facelets or JSP document. To enable this option, you must select a project or a file within a project in the Application Navigator.

3. Name the page **Dept_Emp**, select the **Create as JSP XML** radio button, and checking the **Page Template**, select the **Oracle Three Column Layout** and click **OK.**



Create JSF Page

Enter the name and directory, and choose a type for the JSF Page. It can be implemented as a Facelets or JSP XML document.

File Name: Dept_Emp

Directory: D:\WebServicePractice\WebClient\public_html

Document Type: ○ Facelets ● JSP XML

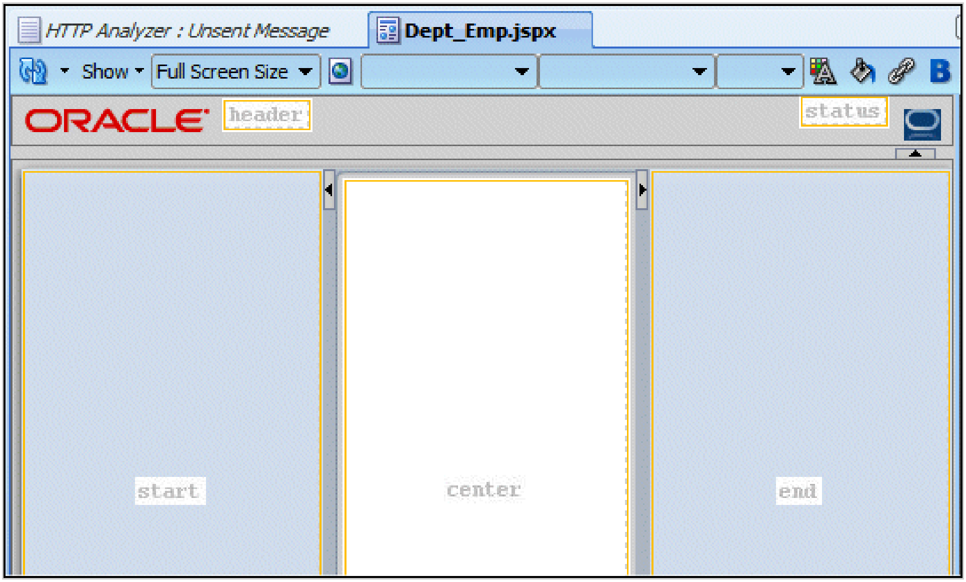
□ Render in Mobile Device

○ Blank Page

● Page Template: Oracle Three Column Layout

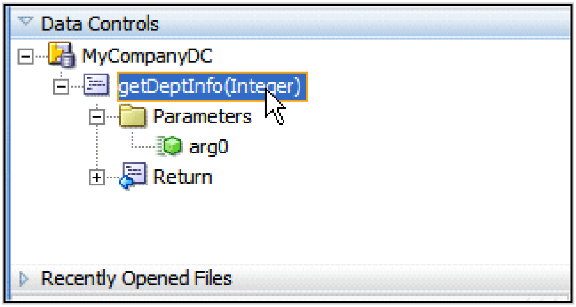
○ Quick Start Layout

The page will open in the page editor

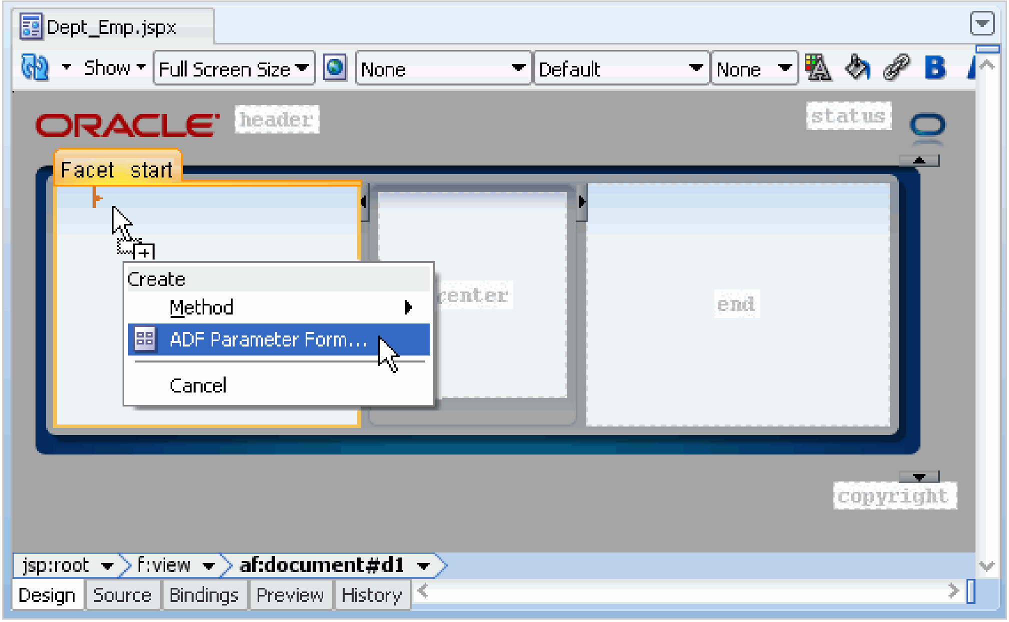


You now add the data controls for the method. The page should accept a parameter, execute the call to the web service, and return the results in a Master Detail layout.

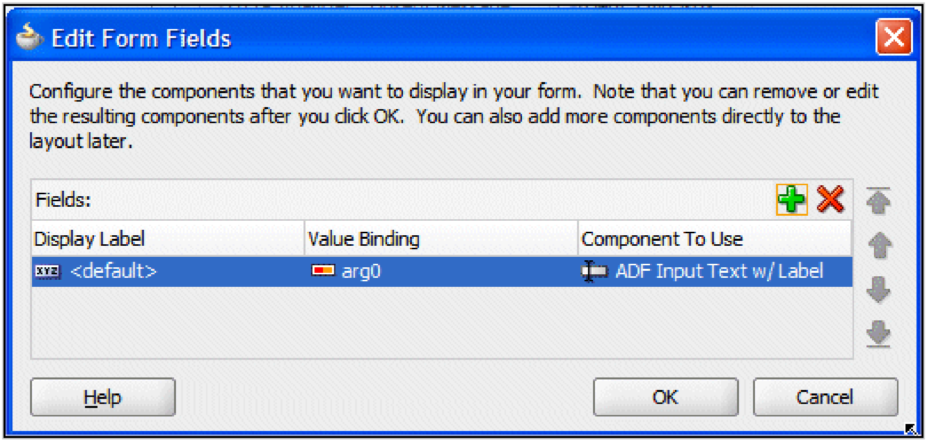
4. In the Data Control panel, expand the **MyCompanyDC** data control and select the **getDeptInfo(Integer)** method.



5. Drag the **getDeptInfo(Integer)** method onto the **start** facet of the page. When you drop it, a menu is displayed. Select **ADF Parameter Form**.



6. In the Edit Form Fields window click **OK** to accept the default values.


Edit Form Fields

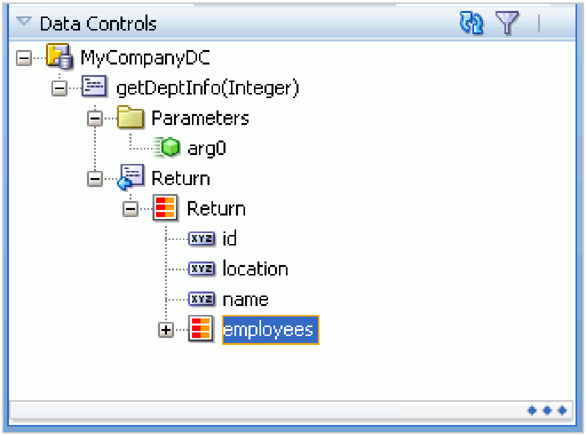
Configure the components that you want to display in your form. Note that you can remove or edit the resulting components after you click OK. You can also add more components directly to the layout later.

Fields:

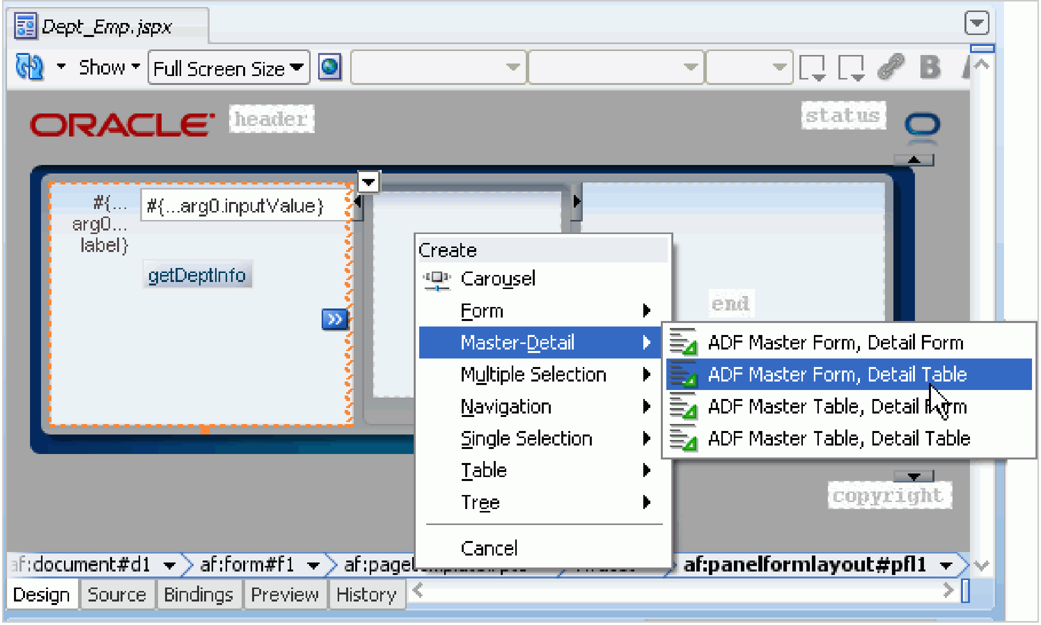
Display Label	Value Binding	Component To Use
<default>	arg0	ADF Input Text w/ Label

7. Click **OK**.

8. In the Data Controls panel, expand **MyCompanyDC | getDeptInfo(Integer) | Return | Return** and select **employees**.

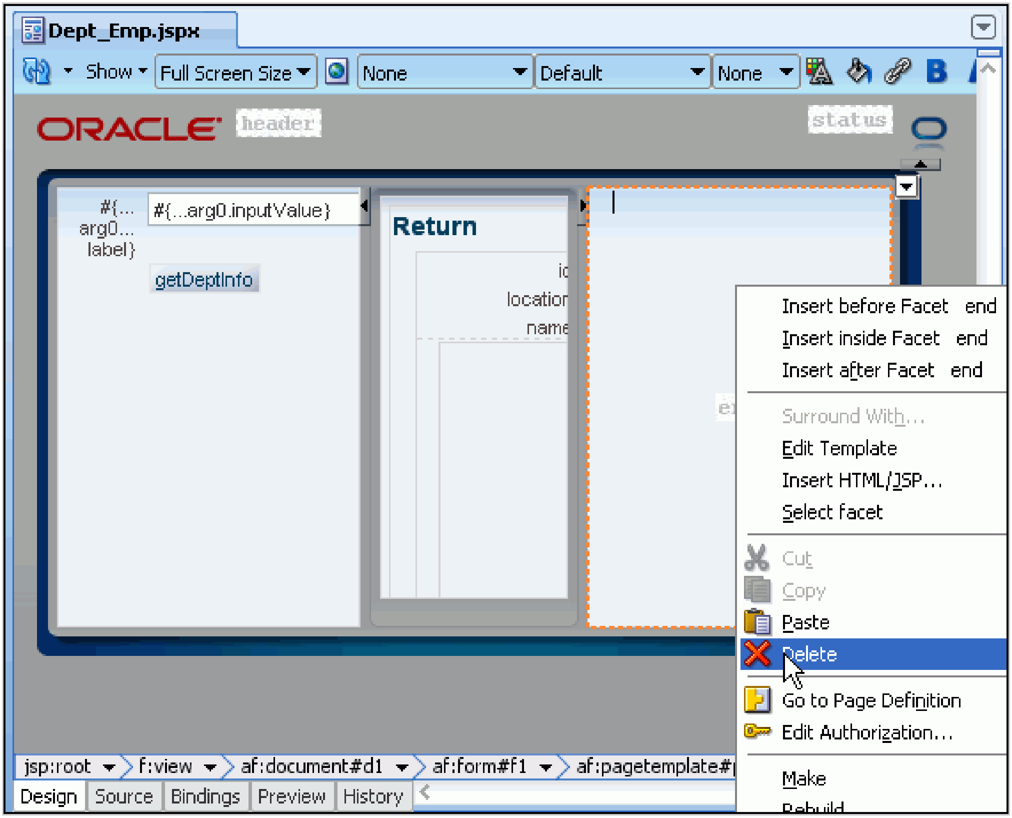


9. Drag **employees** onto the **center** facet. In the dynamic menu select **Master-Detail | ADF Master Form - Detail Table**.

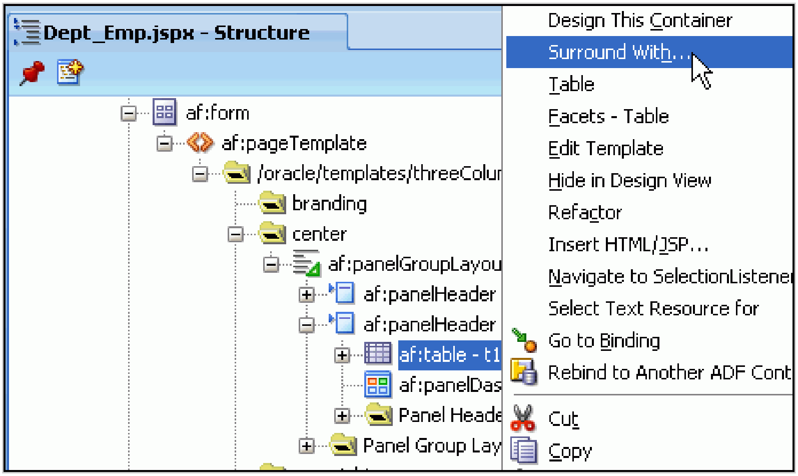


10. Click **Save All** to save your work.

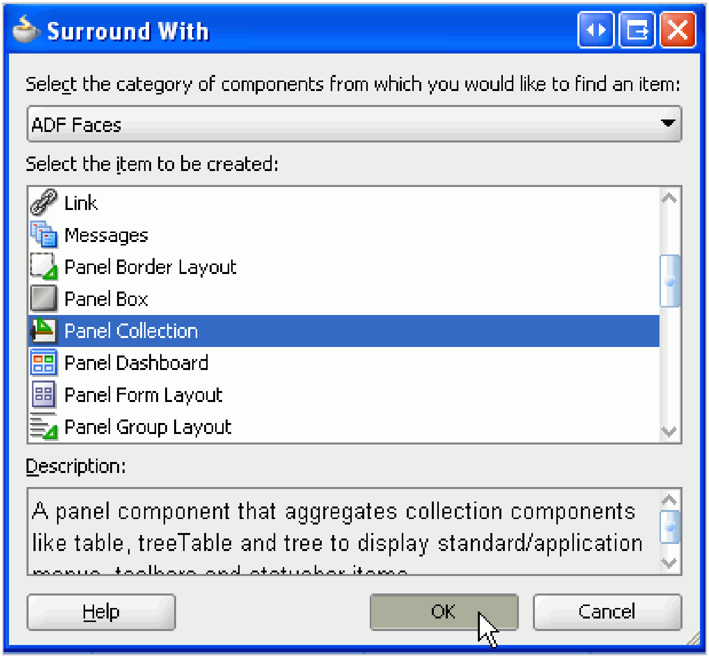
11. Right-click within the **end** facet and select **Delete** from the context menu.



12. In the Structure window for the page, expand the **center** facet. Under the center facet, expand the **af:panelGroupLayout - vertical** and the **af:panelHeader - employees** nodes.
Right-click the **af:table - t1** component and select **Surround With** from the context menu.



13. In the Surround With window select the **Panel Collection** and click **OK**.



Surround With

Select the category of components from which you would like to find an item:

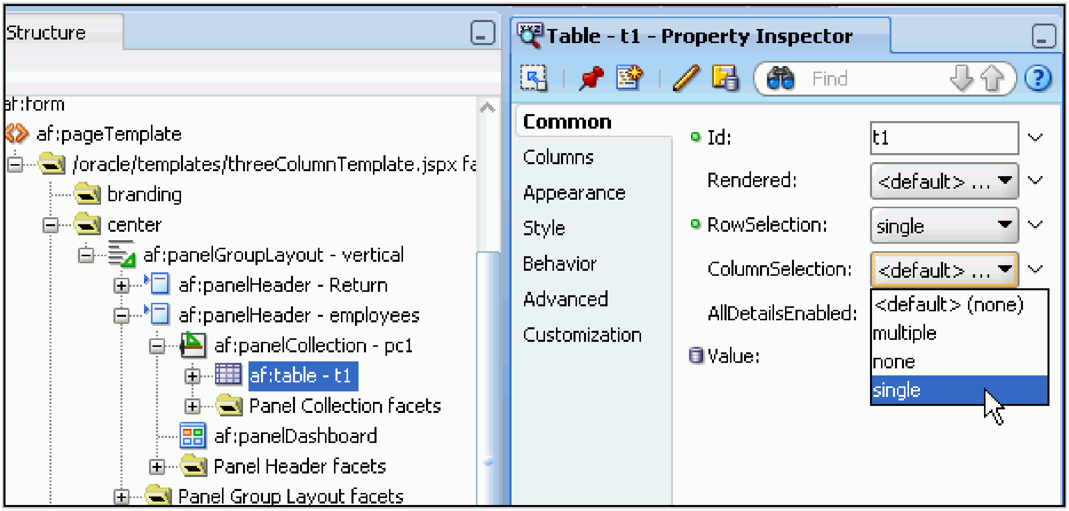
ADF Faces

Select the item to be created:

Description:

A panel component that aggregates collection components like table, treeTable and tree to display standard/application menus, toolbars and statusbar items

14. Expand the **panelCollection** and select the **af:table - t1** component. In the Property Inspector, set the **Column Selection** property to **single**.



Id: t1

Rendered: <default> ...

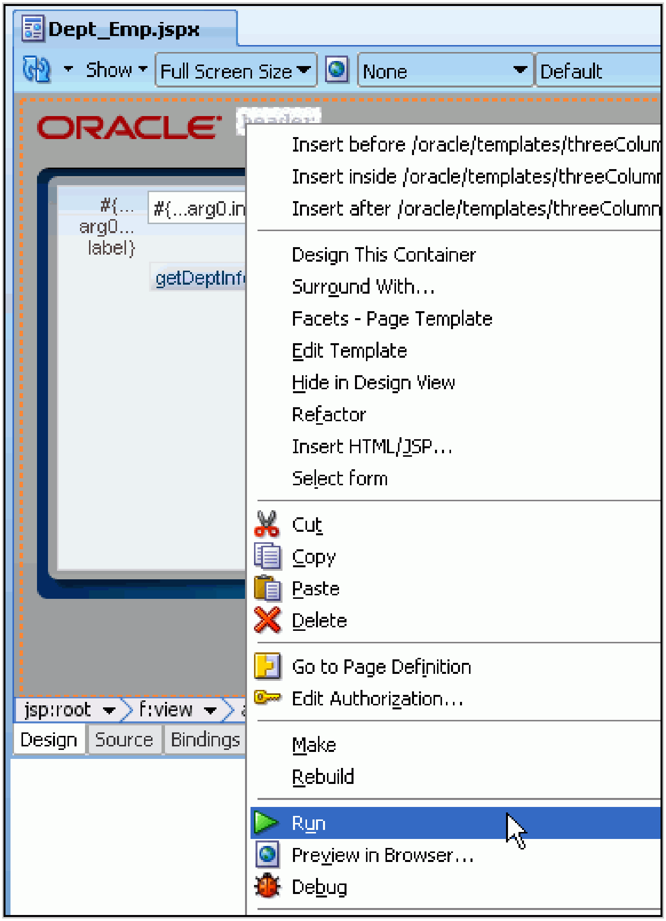
RowSelection: single

ColumnSelection: <default> ...

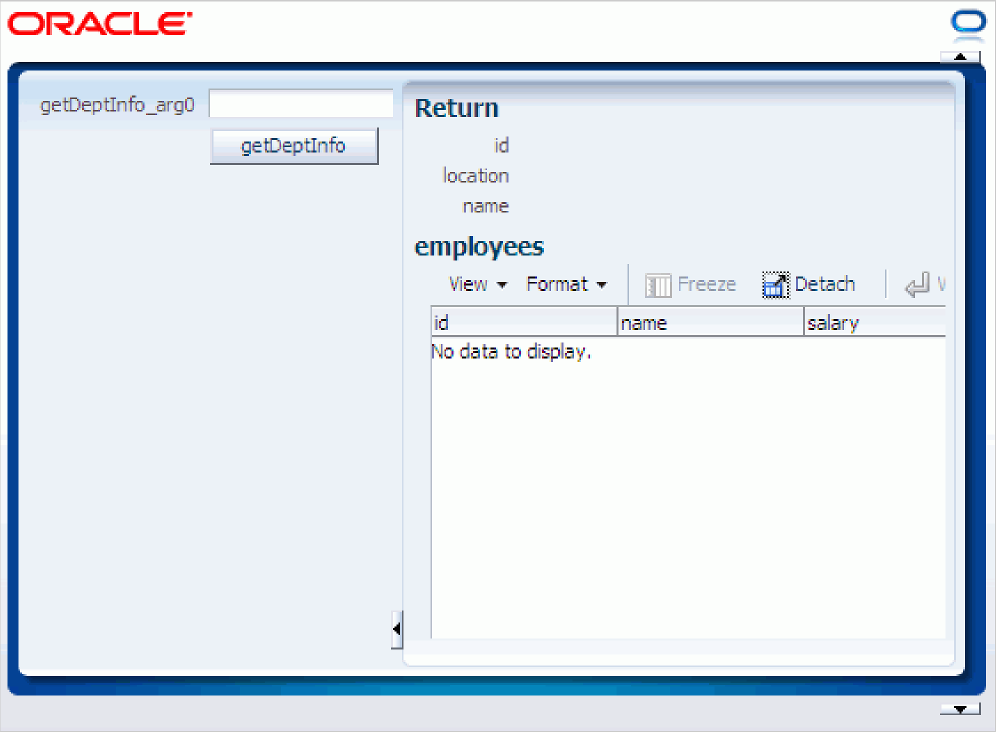
AllDetailsEnabled:

Value:

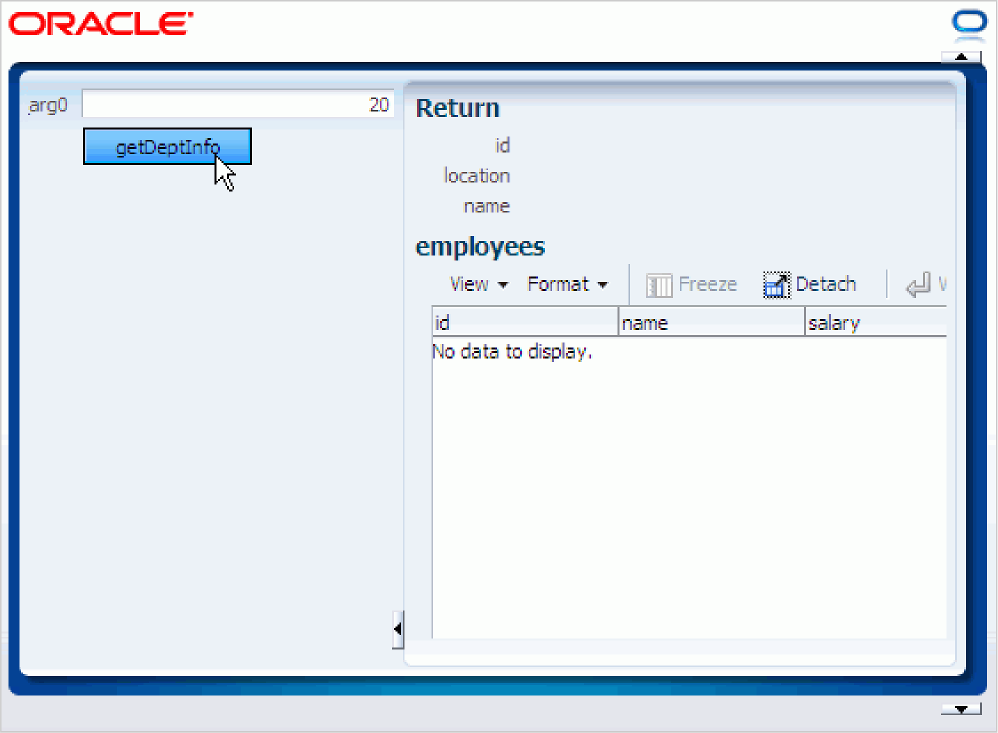
15. Right-click within the page and select **Run** from the context menu.



16. The page loads into your default browser.



17. In the parameter field enter a department value and click the **getDeptInfo** button.



arg0: 20

Return

id

location

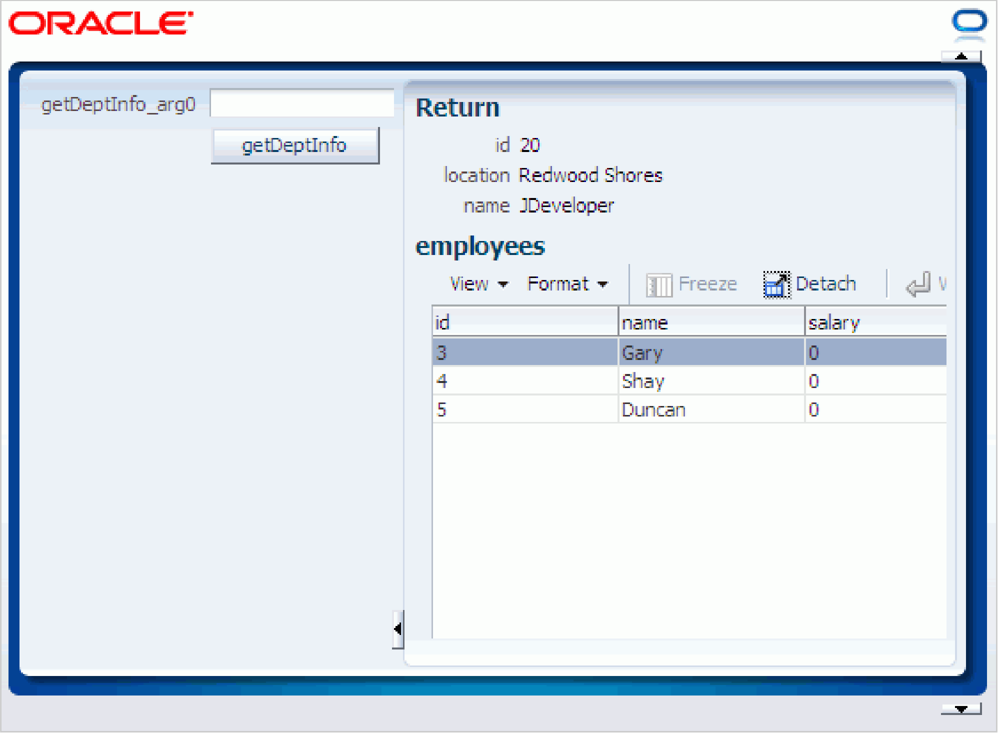
name

employees

id	name	salary

No data to display.

18. The related employees are retrieved and displayed



getDeptInfo_arg0:

Return

id 20

location Redwood Shores

name JDeveloper

employees

id	name	salary
3	Gary	0
4	Shay	0
5	Duncan	0

Summary:

In this tutorial you have seen how to create Web Services from two perspectives; Top-down and Bottom-up. In the Top-down approach you created Web Services from scratch based on what you wanted to do. In the Bottom-up sections, you created Web Services based on existing services. You learned how to:

- Create an annotated POJO Web Service
- Create a POJO and use JDeveloper to declaratively annotate the Web Service
- Build a Web Service using an existing WSDL document
- Build a annotated Web Service based an Enterprise Java Bean
- Create a proxy class for an existing Web Service so you can consume it for a Java program

- Create an ADF Data Control based on an existing Web Service

To learn more about creating enterprise applications using ADF with JDeveloper, refer to:

- Fusion Developer's Guide for Oracle ADF
- Web User Interface Developer's Guide for Oracle ADF
- User's Guide for Oracle JDeveloper