IBM RATIONAL APPLICATION DEVELOPER 6.0 – LAB EXERCISE

Building and Testing a JSF-Portlet Application

What this exercise is about

In this lab, you will build a portlet in IBM Rational Application Developer v6.0 using the JSF-Portlet Framework. After you have finished designing and implementing your portlet, you will test it using the WebSphere Portal v5.0 Test Environment in Rational Application Developer.

Lab Requirements

List of system and software required for the student to complete the lab.

- IBM Rational Application Developer v6.0 with the Portal Tools Additional Feature selected
- WebSphere Portal Server v5.0.2.2 Test Environment
- The lab source files LabFiles60.zip must be extracted to the root directory C:\

What you should be able to do

At the end of this lab you should be able to:

- Build a portlet application using the JSF-Portlet Framework included by the WebSphere Portal Server
- Test a portlet application in the Portal Test Environment

Introduction

In this lab, you will design a portlet that displays basic contact information; you can think of it as an address book. The main page of the View Mode is simply a list of contacts. When the user clicks on the name of a specific contact, the portlet displays detailed information about that contact. The Edit mode works mostly the same way, but clicking on a specific contact allows the user to modify that contact's information.

Exercise Instructions

Some instructions in this lab may be Windows operating-system specific. If you plan on running the lab on an operating-system other than Windows, you will need to execute the appropriate commands, and use appropriate files (.sh vs. .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references, as follows:

Reference Variable	Windows Location	AIX/UNIX Location
<WAS_HOME>	C:\WebSphere\AppServer	/usr/WebSphere/AppServer /opt/WebSphere60/AppServer
<IRAD_HOME>	C:\Program Files\IBM\RSDP\6.0	
<LAB_FILES>	C:\Labfiles60	/tmp/Labfiles60
<LAB_NAME>	IRAD_Portal_JSF	

Windows users please note: When directory locations are passed as parameters to a Java program such as EJBdeploy or wsadmin, it is necessary to replace the backslashes with forward slashes to follow the Java convention. For example, C:\LabFiles60\ would be replaced by C:/LabFiles60/.

Solution instructions are included at the end in case you are unable to complete the lab.

Part 1: Creating a New JSF-Portlet Project

In this part, you will create a new workspace and a JSF-Portlet project.

____ 1. Start Rational Application Developer.

__ a. Select **Start > Programs > IBM Rational > IBM Rational Application Developer V6.0 > Rational Application Developer**.

__ b. A dialog box will be displayed allowing you to select the location where you would like the workspace directory to be stored. Enter **<LAB_FILES>\<LAB_NAME>\workspace** for the location and select **OK**. Application Developer will start with an empty workspace. An empty workspace will leave your existing workspace untouched and help avoid name conflicts between what you may already have in your workspace and what you will be creating in this lab.

NOTE: If the Auto Launch Configuration Change Alert window appears click the **Yes** button to change the auto launch eclipse instance to use when opening IBM Rational Software Development Platform in the future.

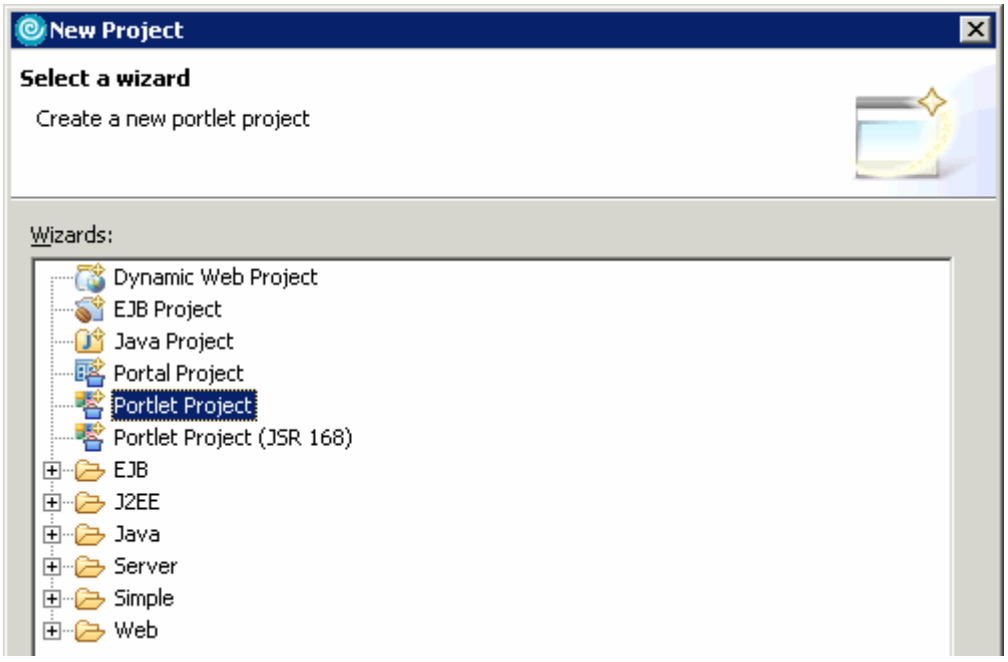
____ 2. When Rational Application Developer v6.0 opens, close the welcome page.

____ 3. Create a new project that will contain a portlet that utilizes the JSF-Portlet Framework.

__ a. Click **File > New > Project…**. The New Project wizard opens.

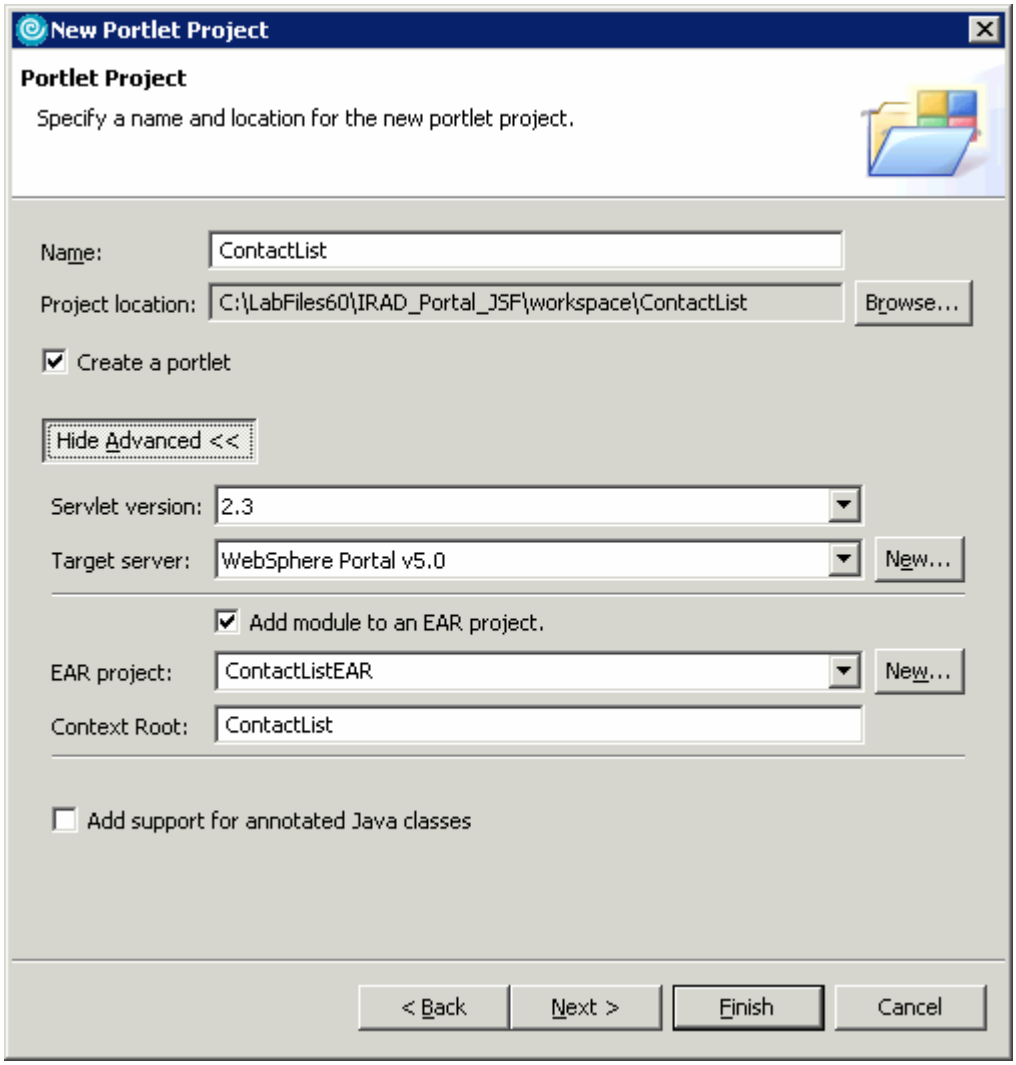
__ b. On the Select panel, select **Portlet Project** and click **Next >**.

NOTE: The JSF-Portlet Framework is also supported using the JSR 168 Portlet API.

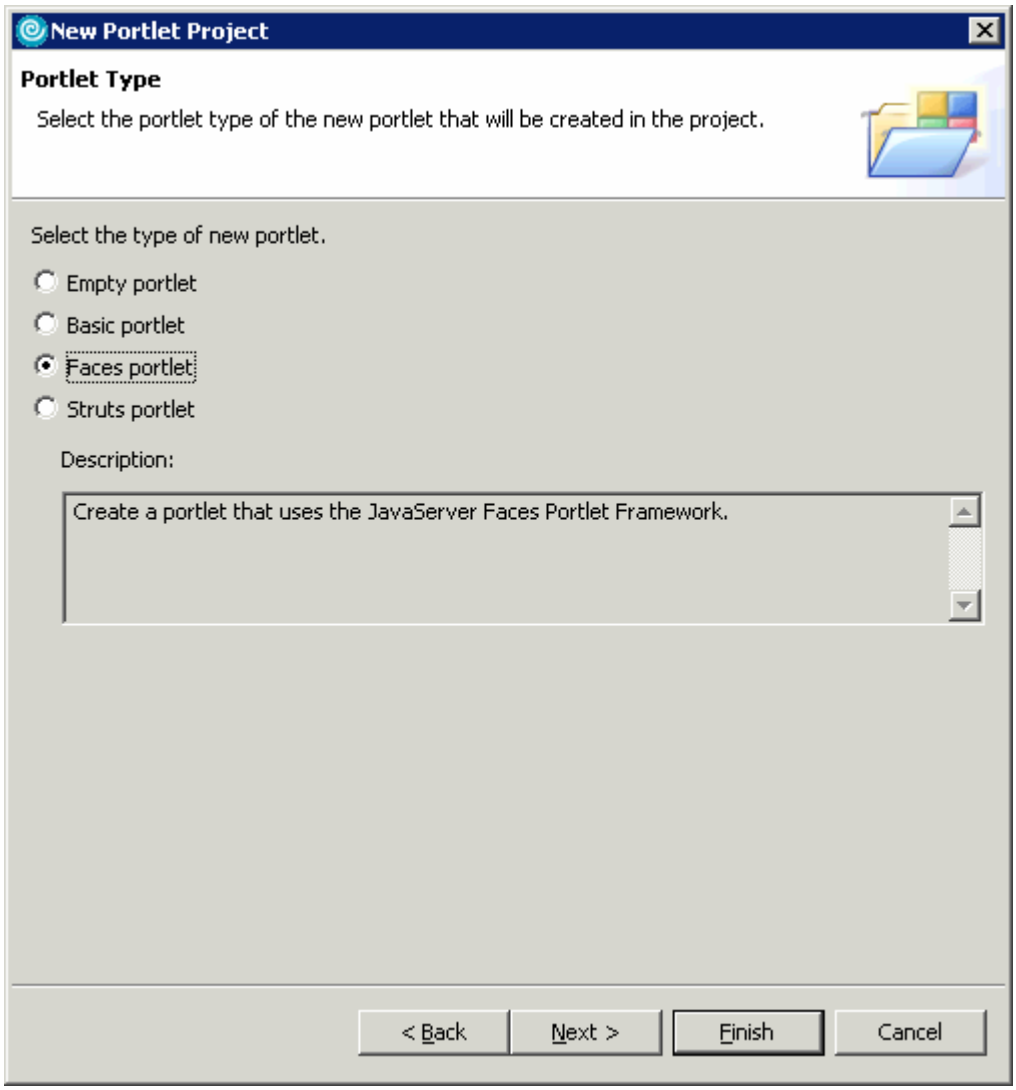


__ c. In the Confirm Enablement window, read the message and click **OK**. Creating a Portlet project requires that advanced Web Development capabilities be enabled within Application Developer.

__ d. On the Portlet Project panel, enter **ContactList** in the Name field. Click **Show Advanced >>** to show the advanced properties of the portlet project wizard. Verify that the Target server field is set to **WebSphere Portal v5.0**. Click **Next >**.



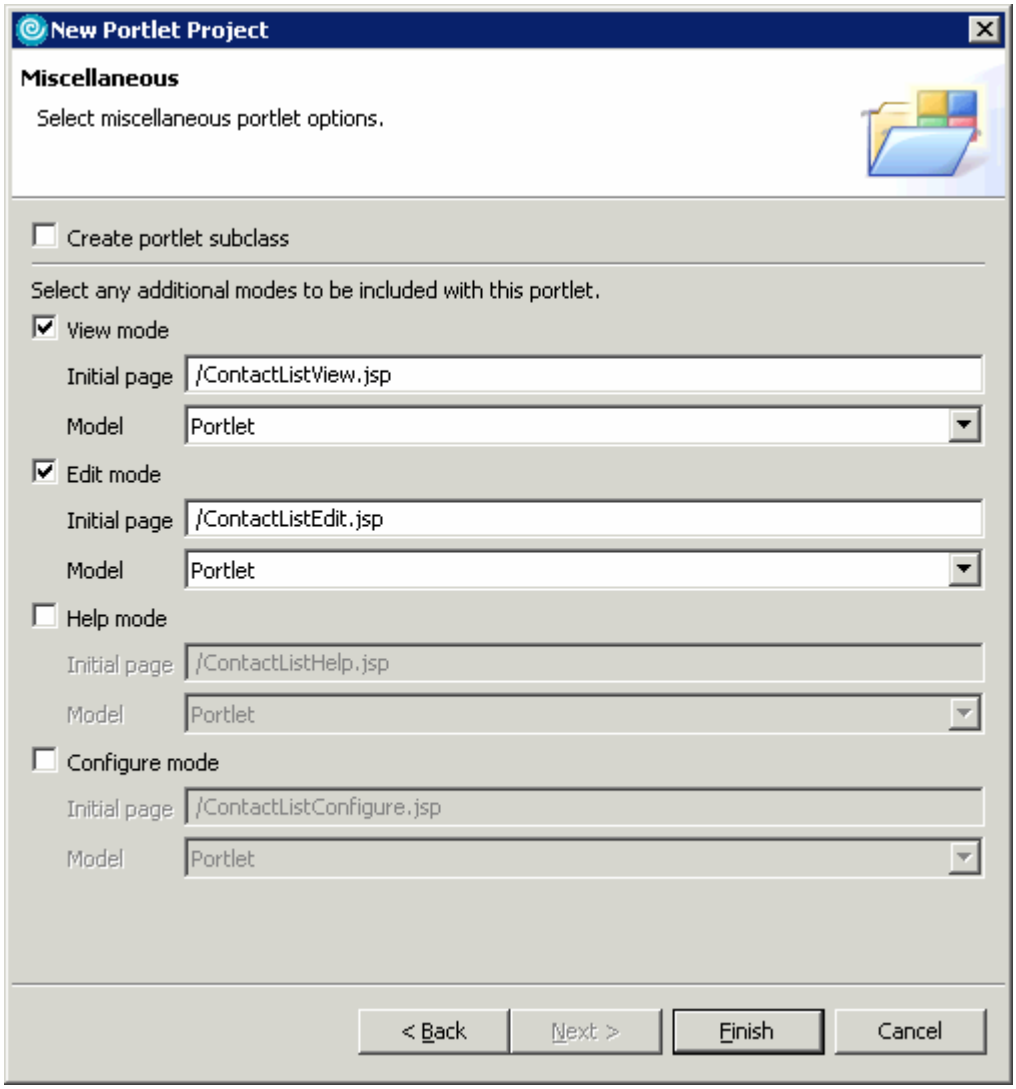
__ e. In the Portlet Type panel, select **Faces portlet** (see the screen capture below) and click **Next >**.



__ f. On the Features Page panel, observe the defaults and click **Next >**.

__ g. On the Portlet Settings page, observe the defaults and click **Next >**.

__ h. On the Miscellaneous page, check the box next to **Edit Mode** to add an edit mode to the new portlet. Leave the default values for the initial JSP page's name and the model in which the new JSP page should follow. See the screen capture below to verify that your settings are correct. Click **Finish**.



__ i. In the Confirm Perspective Switch window, read the message and click **Yes**. Portlet projects are associated with the Web Perspective within Application Developer.

__ j. The portlet project will be created and the ContactListView.jsp (the initial page for the portlet's view mode) file will be opened in the JSP editor.

__ k. **Close** the ContactListView.jsp editor.

____ 4. Observe the default settings of the new Portlet project.

__ a. In the Project Explorer view, expand **Dynamic Web Projects > ContactList > WebContent > WEB-INF**. Note that the JSF configuration file (**faces-config.xml**) has already been created.

The faces-config.xml will contain the configuration information for all modes of the created portlet project.

__ b. Double click on **faces-config.xml** to open it in an XML File Editor.

__ c. In the XML Editor, find the **<factory…> tag** under the <**faces-config…> tag**. Notice that the JSF context factory is using the JSF-Portlet implementation for our project.

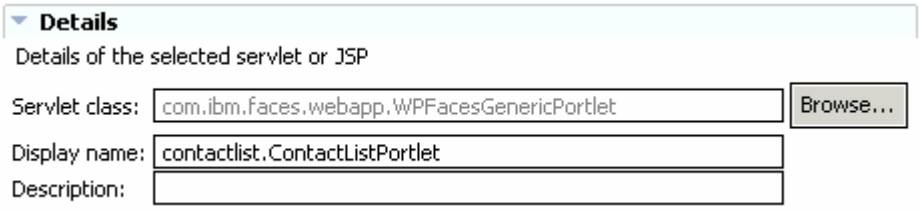
```
<faces-config>
    <factory>
        <faces-context-factory>
        com.ibm.faces.context.WPPortletFacesContextFactoryImpl</faces-context-factory>
    </factory>
    <managed-bean>
        <managed-bean-name>pc_ContactListView</managed-bean-name>
        <managed-bean-class>pagecode.ContactListView</managed-bean-class>
        <managed-bean-scope>request</managed-bean-scope>
    </managed-bean>
```

__ d. **Close** the XML File Editor.

__ e. In the Project Explorer view, double click on **web.xml** to open it in the Web Deployment Descriptor editor.

__ f. Select the **Servlets** tab of the editor.

__ g. Under the **Servlets and JSPs** section, select **contactlist.ContactListPortlet**. Under the Details section, note that the Servlet class is com.ibm.faces.webapp.WPFacesGenericPortlet.



__ h. **Close** the Web Deployment Descriptor editor.

Part 2: Import Pre-created Beans and JSP Pages

In this part, you will import parts of the Contact List application that have already been created. The parts of the application that you will be importing are the beans and the ContactListViewDetails.jsp and ContactListEdit.jsp pages.

____ 1. Import the pre-created beans and JSP pages into the Contact List application.

__ a. Click **File > Import…**. The Import wizard will open up.

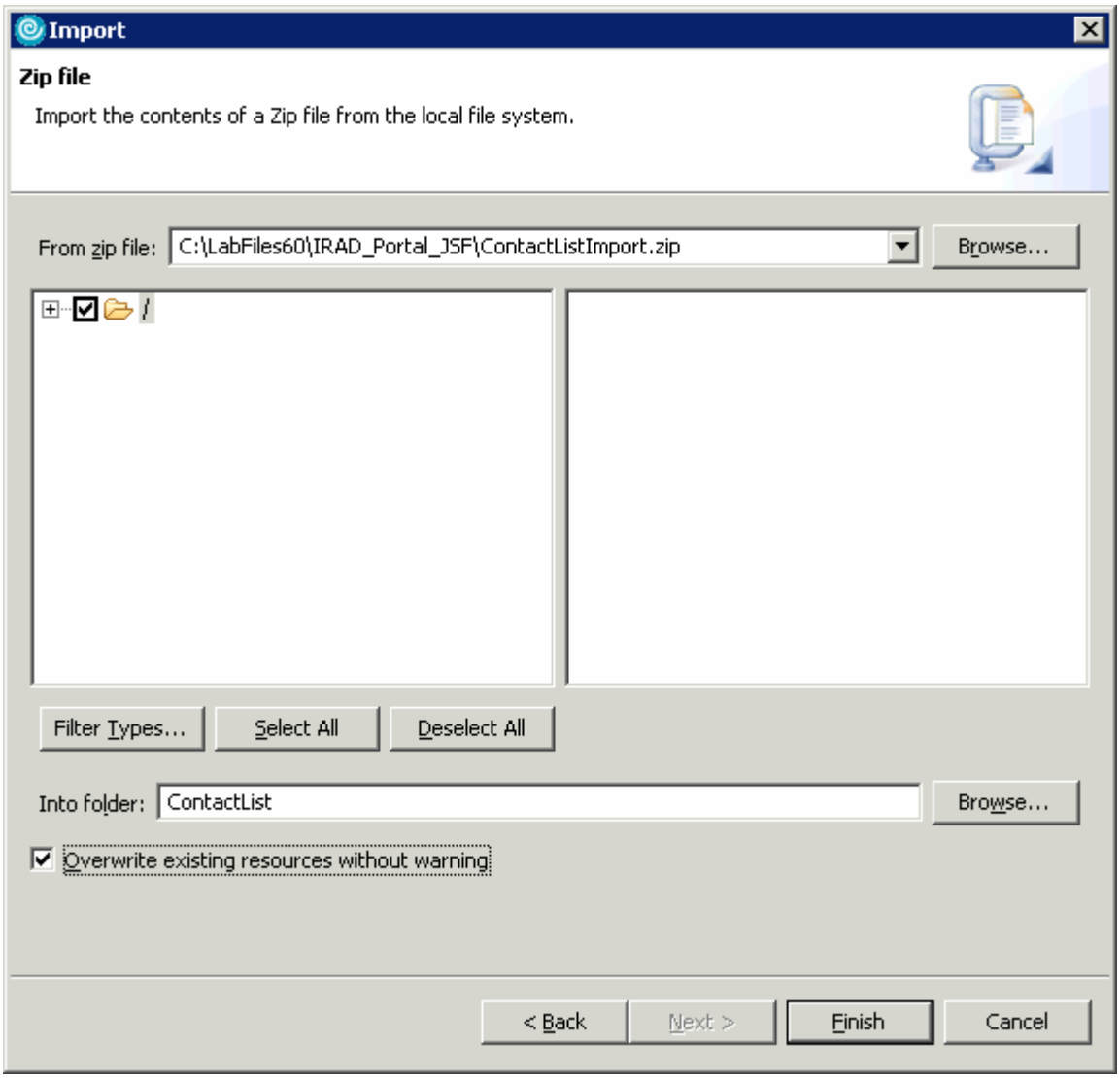
__ b. In the Select panel, select **Zip file** and click **Next >**.

__ c. In the Zip file panel, fill in the following values (see the screen capture below for more details) and click **Finish**.

1) From zip file: **<LAB_FILES>\<LAB_NAME>\ContactListImport.zip**

2) Into folder: **ContactList**

3) Place a check mark in the box next to **Overwrite existing resources without warning**



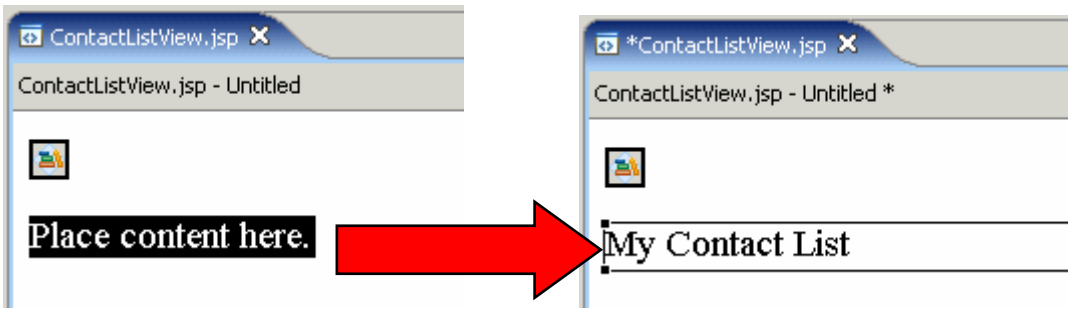
Part 3: Implement the ContactListView.jsp Page

In this part, you will implement the initial page of the View mode of the JSF-Portlet application. The page will allow you to view all entries in the address book in a tabular format. In this part, you will see that creating a JSF application using the JSF-Portlet framework requires no special implementation on your part.

____ 1. Add a header to the ContactListView.jsp page.

__ a. In the Project Explorer view, expand **Dynamic Web Projects > ContactList > WebContent** and double-click on **ContactListView.jsp** to open the JSP editor.

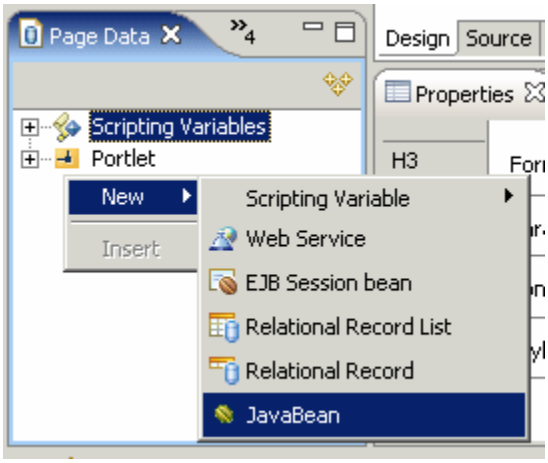
__ b. In the ContactListView.jsp JSP editor, highlight the “Place content here.” sentence and type “My Contact List”.



__ c. Highlight “My Contact List”, right-click on it and select **Insert > Paragraph > Heading 3**.

____ 2. Add a ContactList JavaBean to the page.

__ a. Add the ContactList JavaBean to the page. With the ContactListView.jsp tab selected, right-click in the Page Data view and select **New > JavaBean**. The Add JavaBean wizard will open.

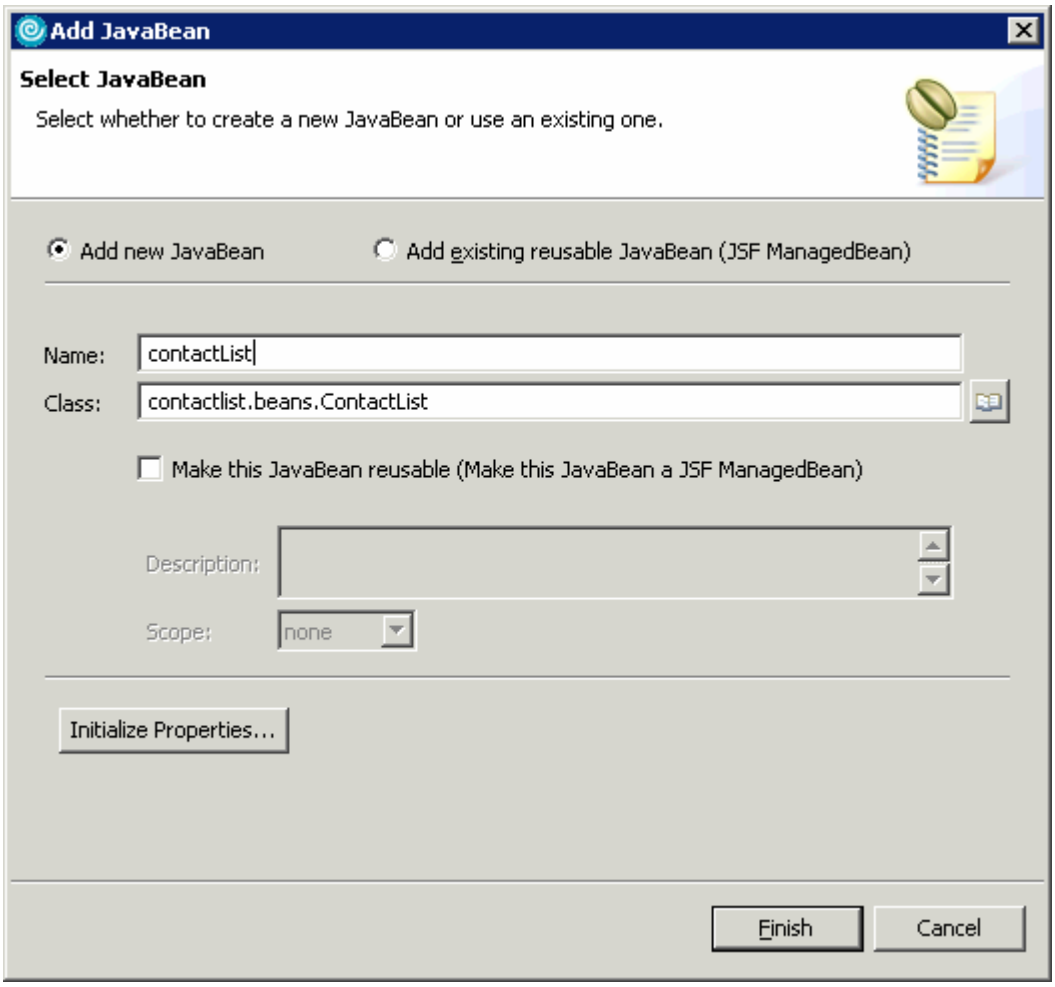


__ b. In the Select JavaBean panel, fill in the following information (see the screen capture below for more details) and click **Finish**.

1) Select **Add new JavaBean**

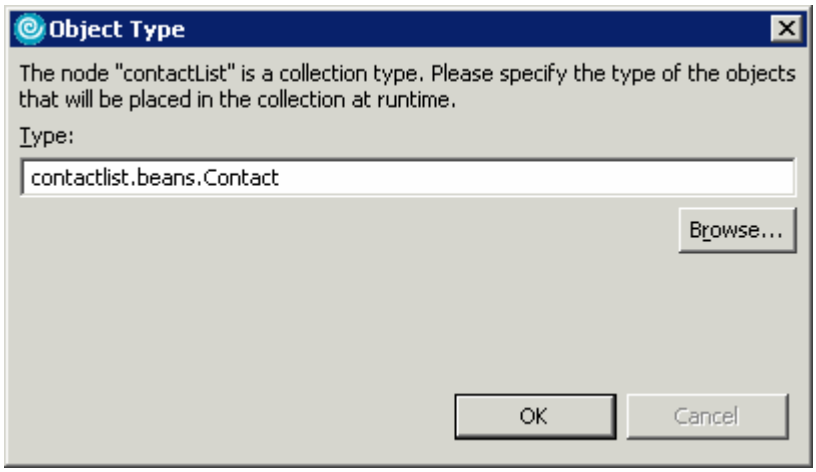
2) Name: **contactList**

3) Class: **contactlist.beans.ContactList**



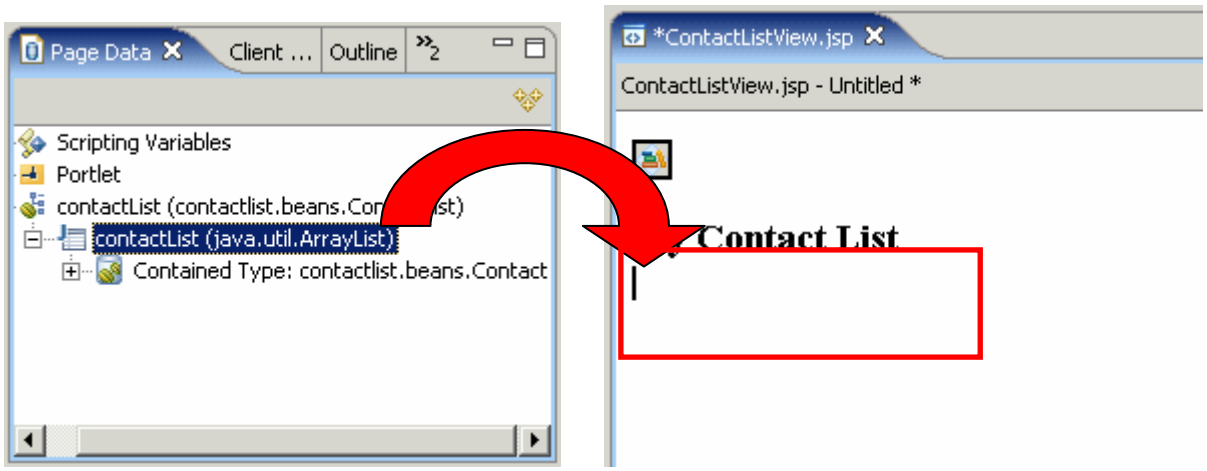
__ c. Add the Contact bean as the contained type of the Contact List bean. In the Page Data view, expand **contactList (contactlist.beans.ContactList) > contactList (java.util.ArrayList)**. Right-click on **Contained Type: Not specified** and choose **Change contained type**. In the Object Type window, fill in **contactlist.beans.Contact** and click **OK**.

NOTE: If the Object Type dialog box does not appear, right click on **Contained Type: Not specified** and choose **Change contained type.**



____ 3. Add a table to the page to display the contents of the contactList bean.

__ a. In the Page Data view, expand **contactList (contactlist.beans.ContactList)** and select **contactList (java.util.ArrayList)**. **Drag-and-drop the selected item from the Page Data view to the line below "My Contact List"** in the JSP editor. See the screen capture below for an illustration.



__ b. The Insert JavaBean wizard that comes up allows you to change the columns that will be displayed in the table. In the Fields to display field, select the following columns and arrange them into the order below (see the screen capture below for more details). Click **Finish** when you are done.

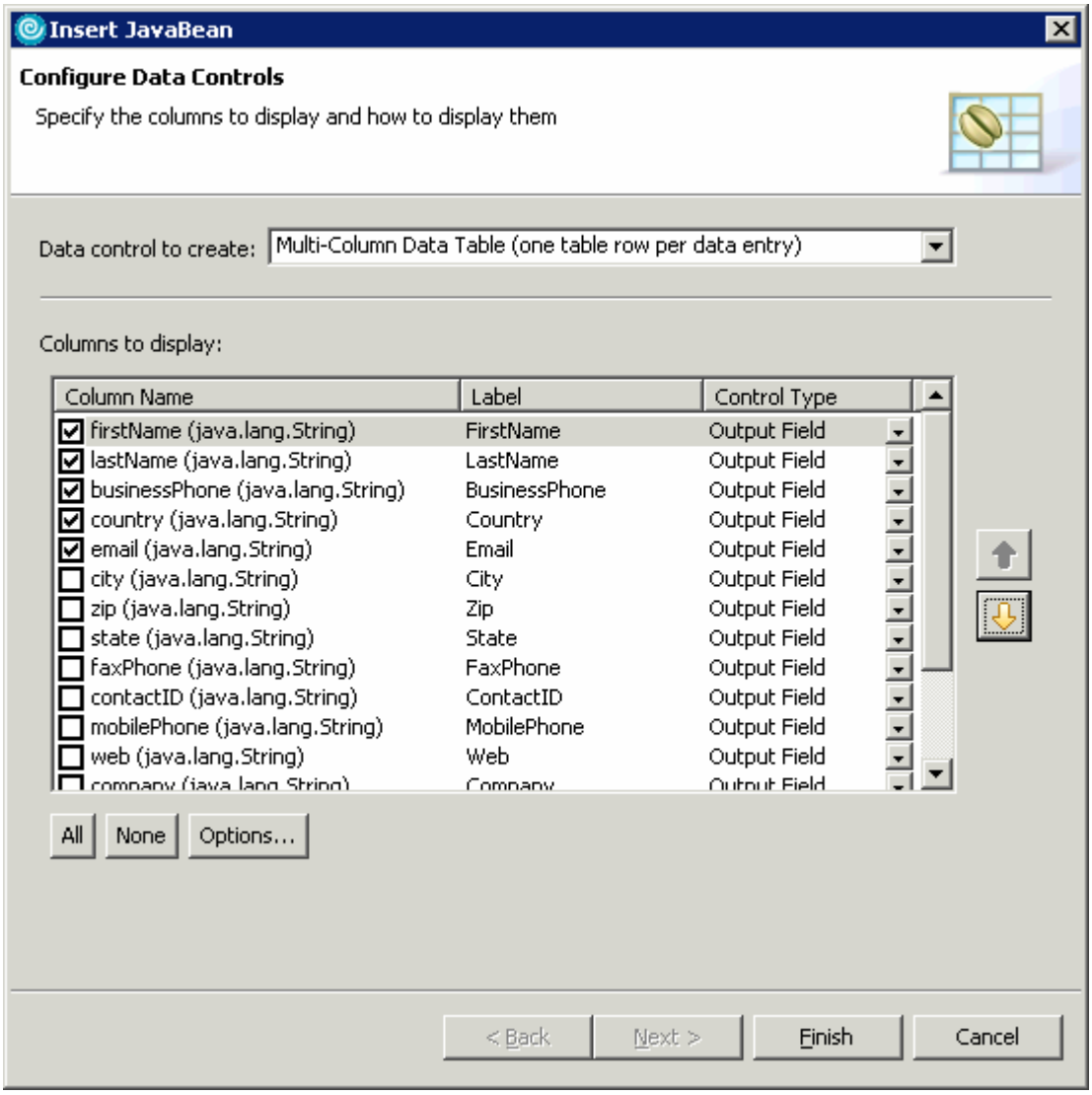
1) First Name

2) Last Name

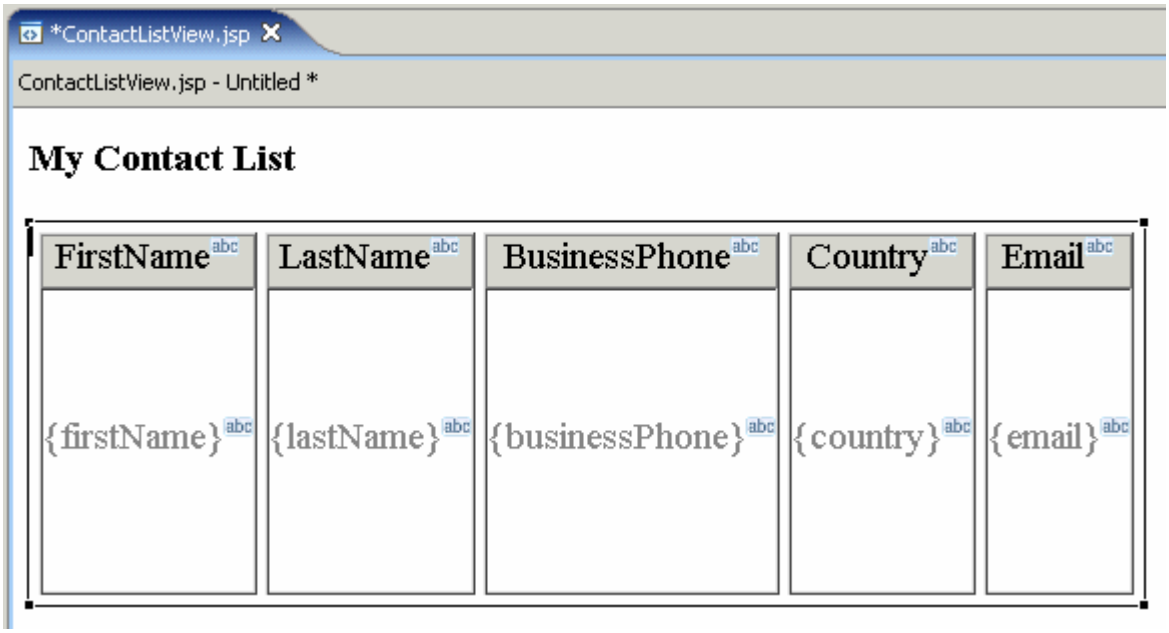
3) Business Phone

4) Country

5) Email

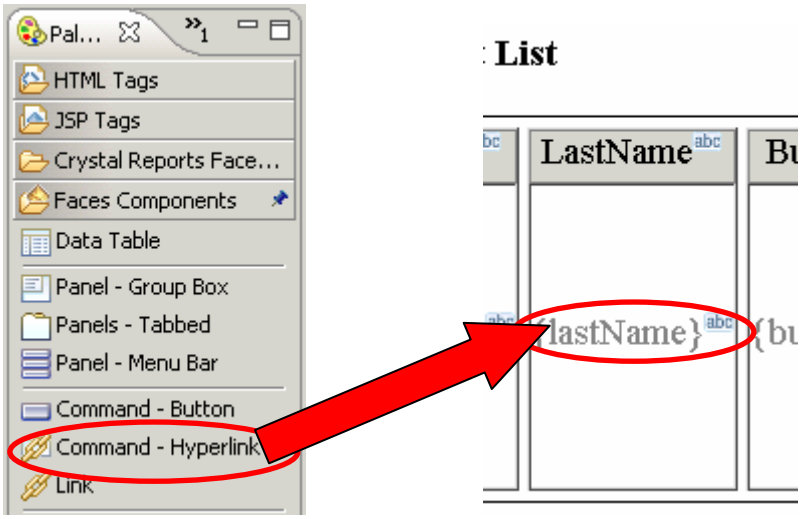


__ c. The end result of the above steps should look like the screen capture below.

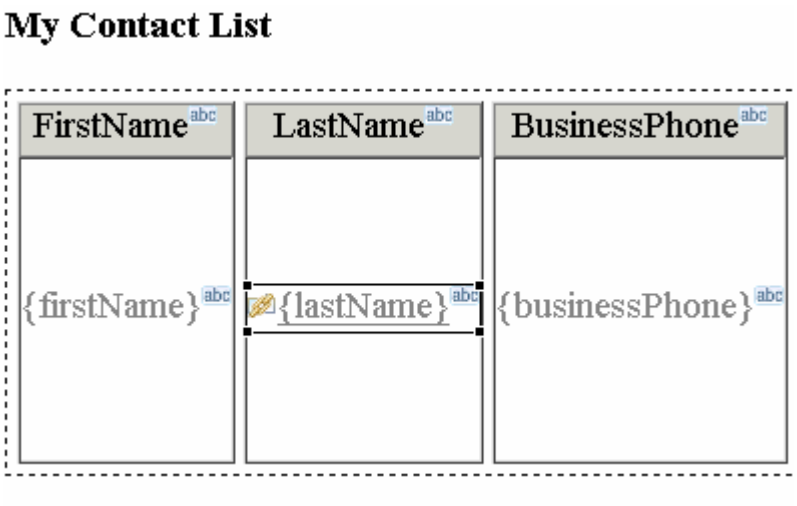


____ 4. Add a Command-Hyperlink from the contact's last name to the ContactListViewDetails.jsp page.

__ a. In the Palette view, ensure that the **Faces Component** drawer is open. **Drag-and-drop the Command – Hyperlink onto the {lastName}** value in the table. See the screen capture below for an illustration.



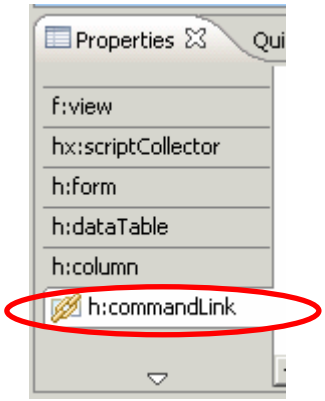
__ b. The end result of the above steps should look like the screen capture below.



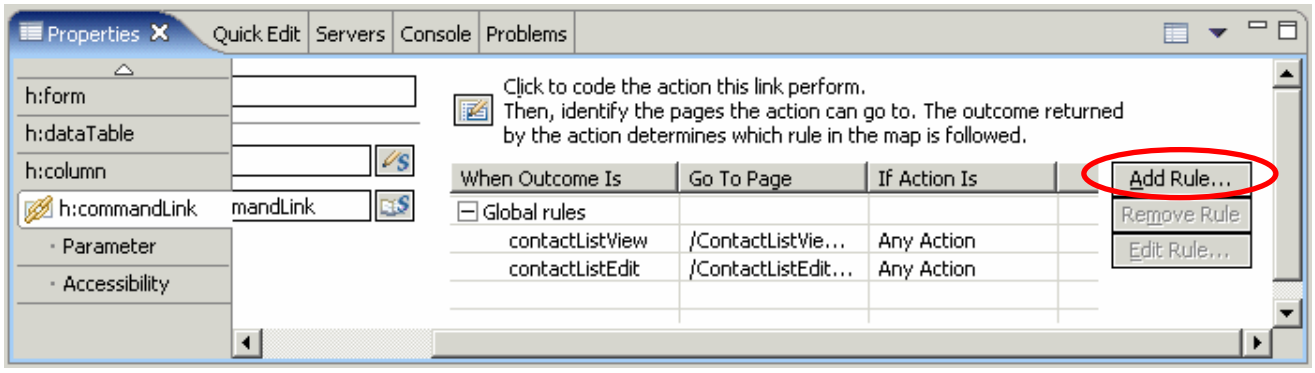
____ 5. Configure the command hyperlink's attributes.

__ a. Switch to the Properties view by selecting **Window > Show View > Properties**.

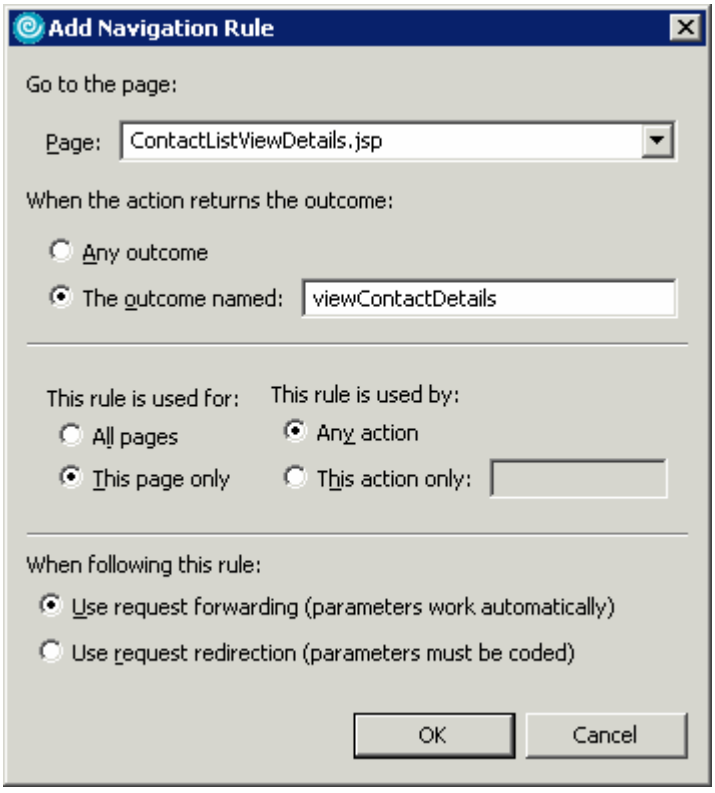
__ b. In the Properties view, verify that the **h:commandlink** tab is selected (see the screen capture below).



__ c. Add a navigation rule to be used within the page to direct it to the ContactListViewDetails.jsp page. In the Properties view, click **Add Rule…** (see the screen capture below).

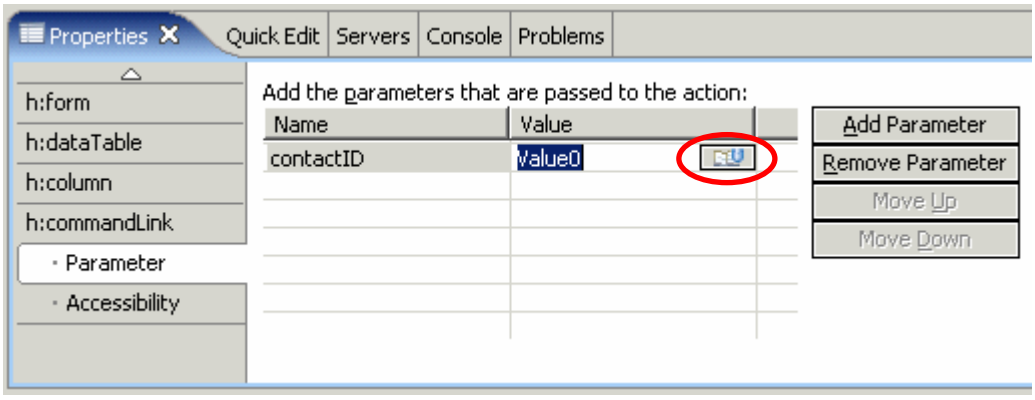


__ d. In the Add Navigation Rule window, for the Page field fill in **ContactListViewDetails.jsp**. Select '**The outcome named**' and enter **viewContactDetails** into the empty field (see the screen capture for more details). Click **OK**.

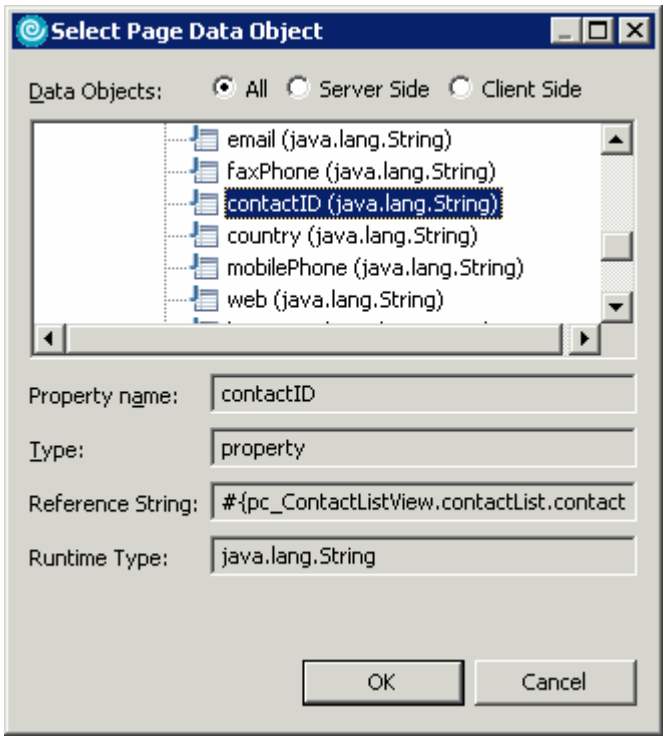


__ e. Add parameters to the URL that will be passed to the ContactListViewDetails.jsp page. In the Properties view, select the **Parameter** tab. In the Parameter tab, click **Add Parameter**.

__ f. In the table, change **Name0** to **contactID**. Click the **Select Page Data Object button** in the next column (see the screen capture below for more details).



__ g. In the Select Page Data Object window, expand **contactList (contactlist.beans.ContactList) > contactList (java.util.ArrayList) > Contained Type: contactlist.beans.Contact** and select **contactID (java.lang.String)** (see the screen capture below for more details). Click **OK**.

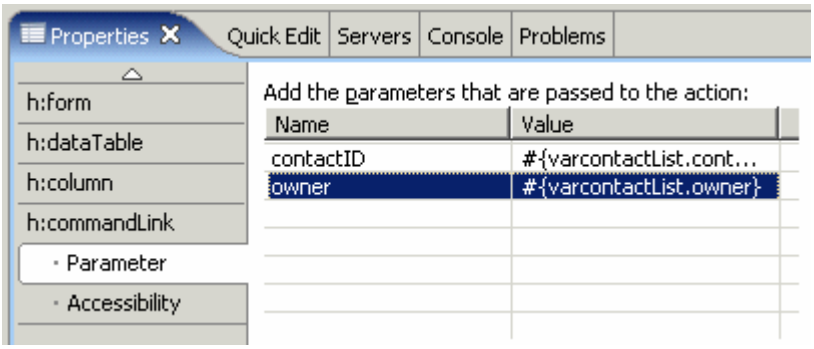


__ h. Add a second parameter to pass. In the **Parameter** tab, click **Add Parameter**.

__ i. In the table, change **Name1** to **owner**. Click the **Select Page Data Object button** in the next column.

__ j. In the Select Page Data Object window, expand **contactList (contactlist.beans.ContactList) > contactList (java.util.ArrayList) > Contained Type: contactlist.beans.Contact** and select **owner (java.lang.String)**. Click **OK**.

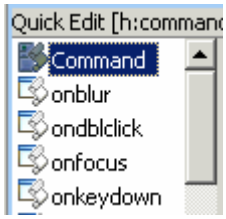
__ k. The end result of the above steps should look like the screen capture below.



____ 6. Add code to the Command Hyperlink so that it makes use of the navigation rule defined above.

__ a. Switch to the Quick Edit view by selecting **Window > Show View > Quick Edit**.

__ b. In the Quick Edit view, ensure that **Command** is selected in the left column.



__ c. Left-click in the right column to add a command to the hyperlink.

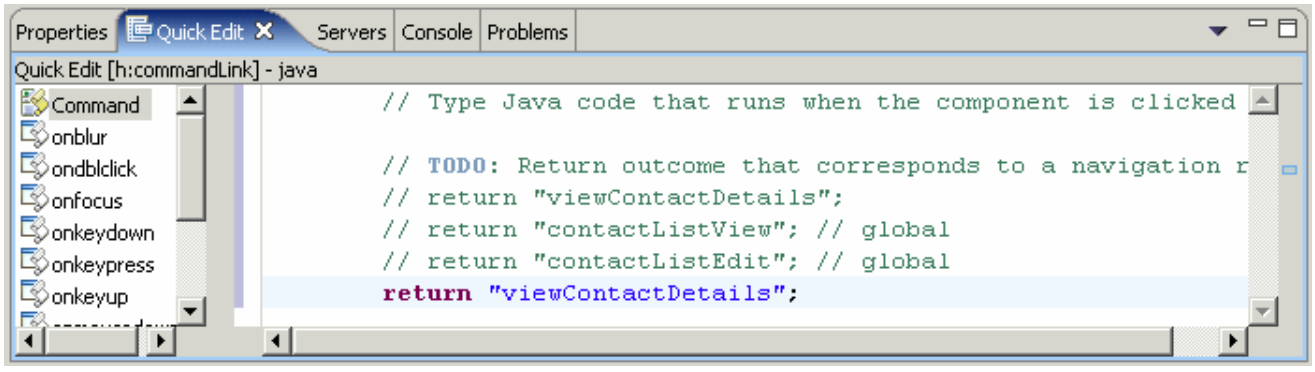
__ d. In the right column, change the following line of code:

```
return "";
```

To

```
return "viewContactDetails";
```

__ e. The end result of the above steps should look like the screen capture below.



__ f. Save and close ContactListView.jsp.

____ 7. Add code to the contactList getter and setter methods for the ContactListView.jsp page.

__ a. In the Project Explorer view, expand **Dynamic Web Projects > ContactList > Java Resources > JavaSource > pagecode** and double-click on **ContactListView.java** to open it.

__ b. Open the **<LAB_FILES>\<LAB_NAME>\snippets\snippet1.txt** file and use the contents of the file to replace the getContactList() method in ContactListView.java.

NOTE: The copied code will first check to see if the contact list is stored within the session. If the contact list it will retrieve it, it doesn't exist the contact list will be created and added to the session.

__ c. Open the **<LAB_FILES>\<LAB_NAME>\snippets\snippet2.txt** file and use the contents of the file to replace the setContactList() method in ContactListView.java.

NOTE: The copied code will store the contact list in the session.

__ d. Save and close ContactListView.java.

Part 4: Test the JSF-Portlet Application on the Portal Test Environment

In this part, you will use the Portal Test Environment to run and test the JSF-Portlet application that you created above.

____ 1. Run the JSF-Portlet Application on a new server configuration.

__ a. In the Project Explorer view, expand **Dynamic Web Projects**. Right-click on **ContactList** and select **Run > Run on Server…**.

__ b. In the Server Selection window, select '**Manually define a server**'. In the Select the server type box, select **WebSphere Portal v5.0 Test Environment**. Click **Next >**.

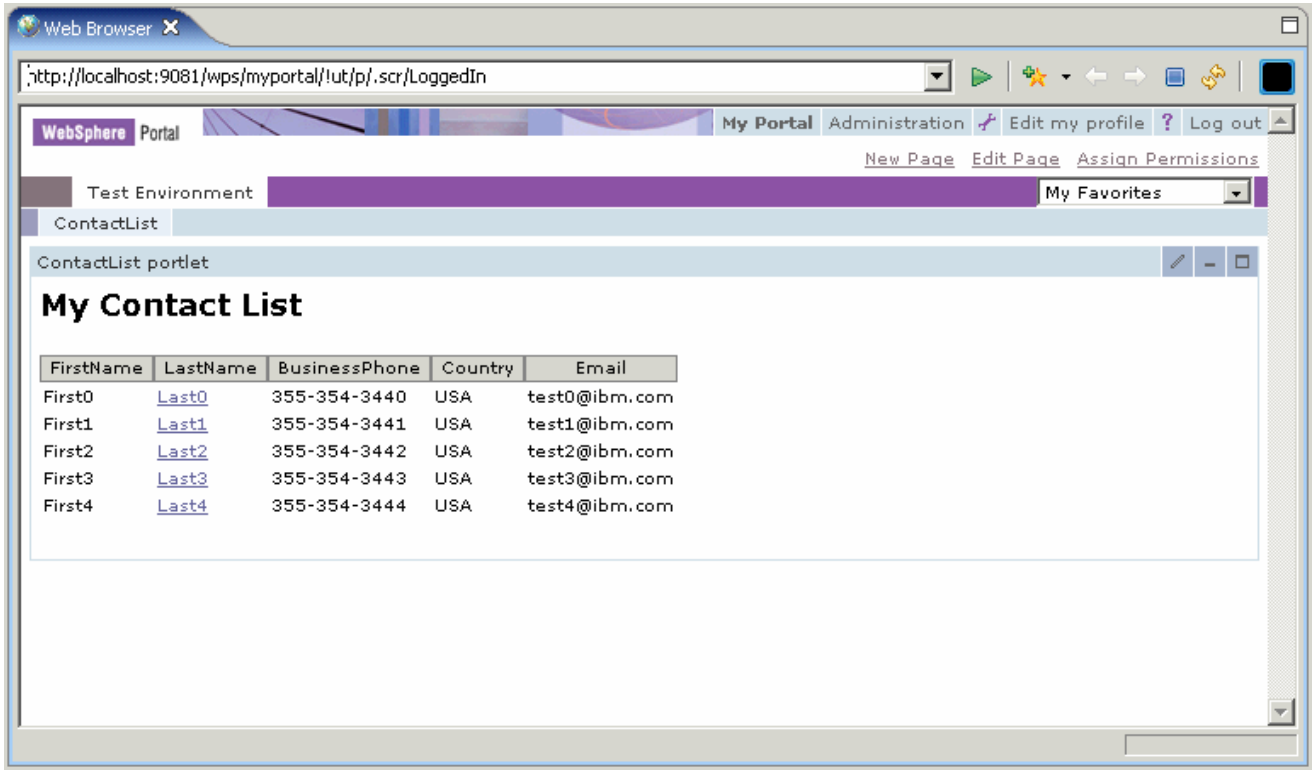
__ c. In the WebSphere Server Configuration Settings panel, leave the default values and click **Finish**.

__ d. A new server and server configuration will be automatically generated, and the Portlet Project will be added to this server. The server will then start, and a web browser will open to display your portlet.

NOTE: Depending on the size of the system on which you are working, the starting of the Portal server may take up to ~5 minutes.

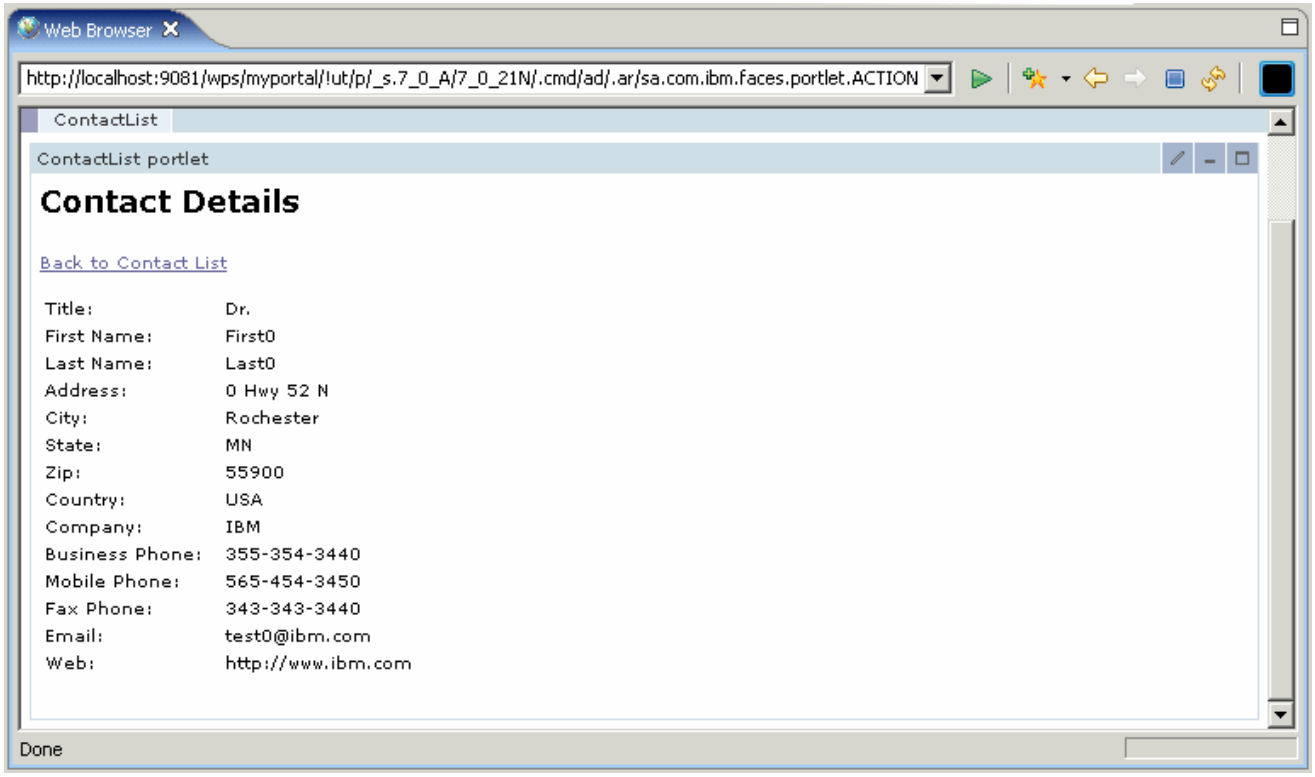
____ 2. Explore the View mode of your portlet.

__ a. When the ContactList portlet comes up, the View mode of the portlet is displayed by default. The page that is initially displayed is the ContactListView.jsp as defined by the Portlet Deployment Descriptor.



__ b. Click on any name in the table to view the contact's detailed information.

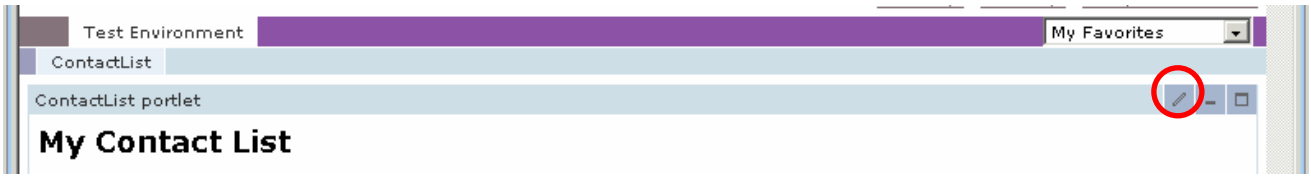
NOTE: Notice that the view mode of the application does not allow you to edit the contact's information.



__ c. Click the **Back to Contact List** link at the top of the portlet to return to the list of contacts.

____ 3. Explore the Edit mode of the Contact List portlet.

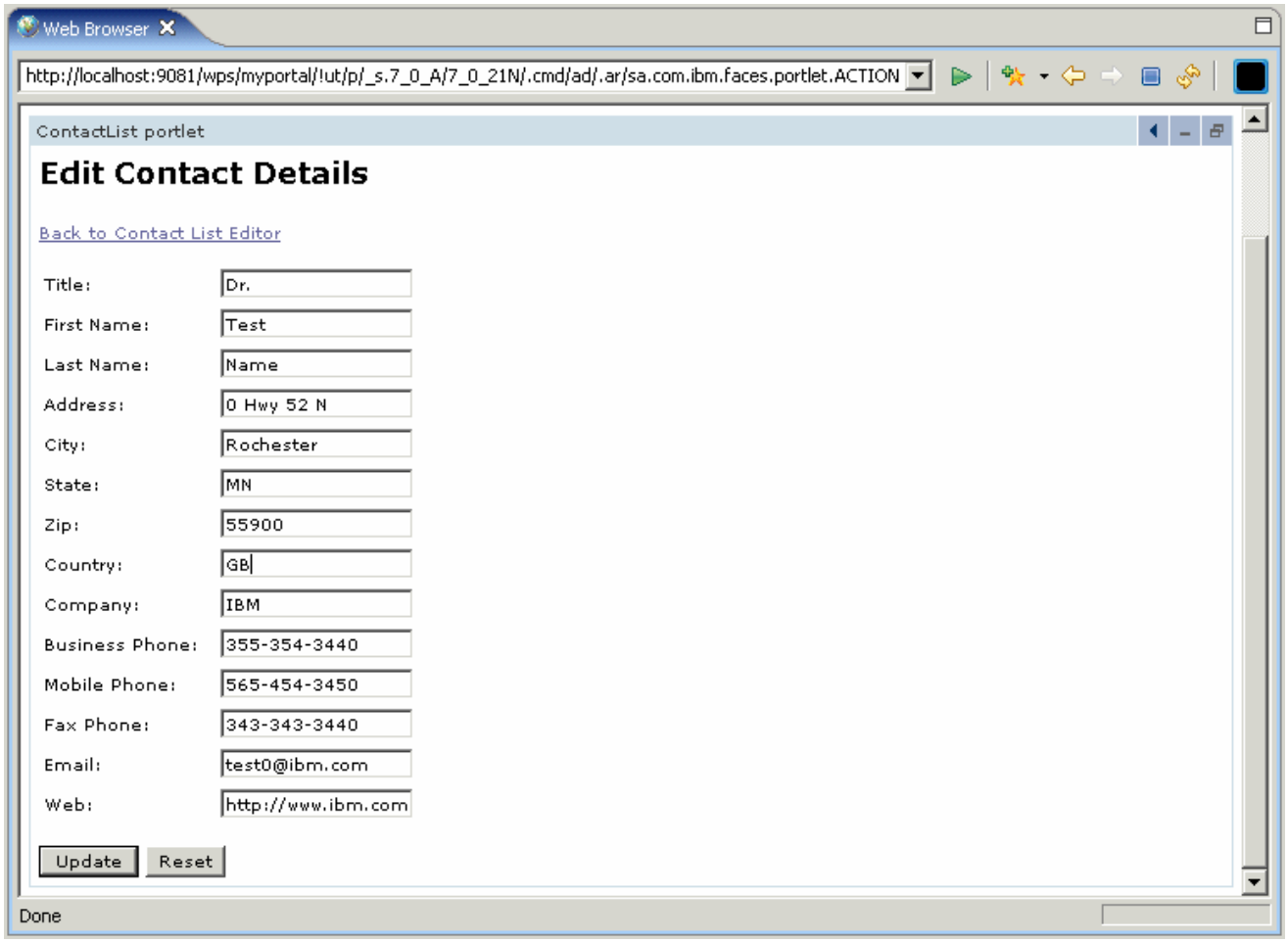
__ a. Click on the **Edit button** (circled in the screen capture below) icon in the upper right corner of the portlet.



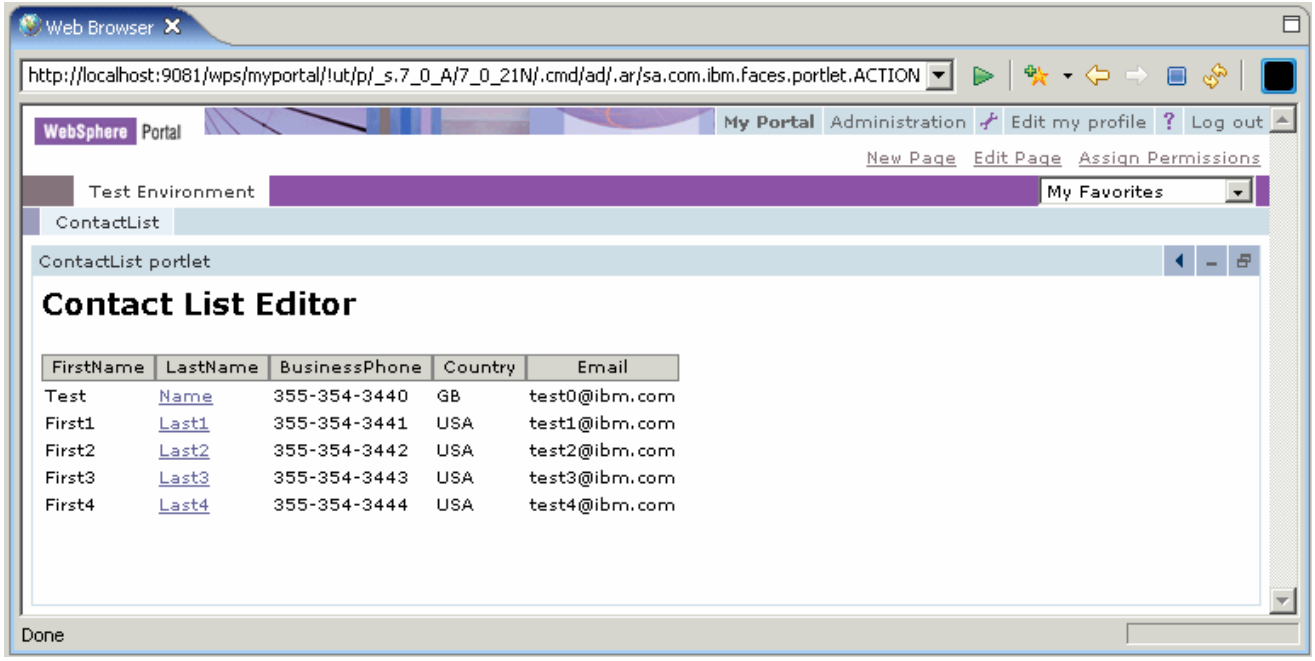
__ b. The initial page of the Edit mode is displayed. The page that is initially shown in edit mode is the ContactListEdit.jsp as defined by the Portlet Deployment Descriptor.

__ c. Click on any name in the table to view the contact's detailed information.

__ d. Edit the information for the contact, and click the **Update** button at the bottom of the form.



__ e. The changes that you previously made should now show up in the table.



__ f. Continue to explore the application until you are satisfied with what you have seen.

____ 4. Stop the test server.

__ a. Switch to the Servers view by selecting **Window > Show View > Servers**.

__ b. Right-click on **WebSphere Portal v5.0 Test Environment @ localhost** and select **Stop**.

__ c. Watch the Console view and wait for the server to come to a complete stop.

____ 5. Exit Application Developer. Select **File > Exit**.

What you did in this exercise

In this exercise, you used Rational Application Developer v6.0 to build a JSF-Portlet application.

Solution Instructions

____ 1. Start Rational Application Developer.

__ a. Select **Start > Programs > IBM Rational > IBM Rational Application Developer V6.0 > Rational Application Developer**.

__ b. A dialog box will be displayed allowing you to select the location where you would like the workspace directory to be stored. Enter **<LAB_FILES>\<LAB_NAME>\solution\workspace** for the location and select **OK**. Application Developer will start with an empty workspace. An empty workspace will leave your existing workspace untouched and help avoid name conflicts between what you may already have in your workspace and what you will be creating in this lab.

____ 2. Import an EAR that contains the completed portlet.

__ a. Select **File > Import…**. The Import wizard will open up.

__ b. In the Select panel, select **EAR file** and click **Next >**.

__ c. In the Enterprise Application Import panel, for the EAR file field enter **<LAB_FILES>\<LAB_NAME>\solution\ContactListEAR.ear**. Change the Target server field to **WebSphere Portal v5.0**. Click **Finish**.

____ 3. Test the portlet application.

__ a. Switch to the Web Perspective. Select **Window > Open Perspective > Web**.

__ b. Follow the steps in Part 4 to test the completed portlet application.

Trademarks and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	iSeries	OS/400	Informix	WebSphere
IBM(logo)	pSeries	AIX	Cloudscape	MQSeries
e(logo)business	xSeries	CICS	DB2 Universal Database	DB2
Tivoli	zSeries	OS/390	IMS	Lotus

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. Microsoft, Windows, Windows NT, and

the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both. Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are

trademarks of Intel Corporation in the United States, other countries, or both. UNIX is a registered trademark of The Open Group in the United States and other countries. Linux is a registered trademark of Linus Torvalds. Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.