

# Accessing data with JSF portlets

Visual development tools make Website creation easy

Skill Level: Intermediate

[Masato Noguchi \(mnoguchi@us.ibm.com\)](mailto:mnoguchi@us.ibm.com)  
Software Engineer  
IBM

[Takeshi Watanabe \(nabet@us.ibm.com\)](mailto:nabet@us.ibm.com)  
Product Manager  
IBM

25 Jul 2004

In this tutorial, you'll learn how to create portlets using the new visual development functions in WebSphere Studio Application Developer V5.1.2 and Portal Toolkit V5.0.2.2. In particular, you'll use: JavaServer Faces, WebSphere Data Object, and Click-to-Action. We'll apply these features to create a sample auction database portal. The programming models and runtime frameworks, along with new easy-to-use tools, make it easier to build applications running on WebSphere Portal. Some of this technology is so new that it is for prototyping purposes only. However, learning about it now will help you prepare so you can implement it when it is available.

## Section 1. Before you start

### About this tutorial

In this tutorial, you'll learn how to create portlets using the new visual development functions in WebSphere Studio Application Developer V5.1.2 and Portal Toolkit V5.0.2.2. In particular, we'll look at the following features:

- **JavaServer™ Faces**, for portlet user interfaces
- **WebSphere Data Object**, for accessing a database
- **Click-to-Action**, for communication between portlets

JavaServer Faces (JSF) is a new standard (JSR 127) that can help you build Web application user interfaces (for more on JSF technology, see [Resources](#)). WebSphere Studio V5.1.2 now supports the final release of the JSF 1.0 specification, and Portal Toolkit V5.0.2.2 enables the creation of JSF application as portlets running on WebSphere Portal.

WebSphere Data Object (WDO) makes it easier to access data objects, such as database tables, from Web applications. WDO is not supported in WebSphere Portal V5.0, but WebSphere Studio V5.1.2 allows you to use WDO with portlets for prototyping purposes only, so you can evaluate how WDO works in harmony with JSF to build portlets quickly. WDO will be replaced by a new standard, called *Service Data Objects* (SDO, JSR 235), in a future release of the WebSphere platform. However, learning about WDO now will help you prepare to implement SDO when it is available.

WebSphere Studio V5.1.2 and Portal Toolkit V5.0.2.2 also enable portlet communication using the WebSphere Portal Click-to-Action feature.

The programming models and runtime frameworks discussed here, along with new easy-to-use tools based on them, can improve the productivity of developers building applications running on WebSphere Portal. You may apply the methodology outlined in this tutorial to your custom portlet development projects.

## Prerequisites

To create portlets in this tutorial, you'll need the following software installed on your computer:

- WebSphere Studio Application Developer V5.1.2
  - Portal Toolkit V5.0.2.2
  - WebSphere Portal V5.0.2.1 (installed as the Portal Test Environment by the Portal Toolkit installation program)Portal Toolkit V5.0.2.2 and WebSphere Portal V5.0.2.1 are packaged in WebSphere Studio 5.1.2.
- [Download DB2 UDB V8.2](#)

## Who should take this tutorial?

This tutorial is for anyone who builds custom-made portlet applications and wants to leverage new programming models and tools to build them quickly.

To get the most out of this tutorial, you should have basic knowledge of:

- Portlet applications running on WebSphere Portal
- Database applications, particularly J2EE™ applications that access DB2
- WebSphere Studio Application Developer and Portal Toolkit

This tutorial does not require advanced Java programming skills.

---

## Section 2. Getting started

### Our sample portlet application: An overview

The features of the sample application that you'll create in this tutorial are as follows:

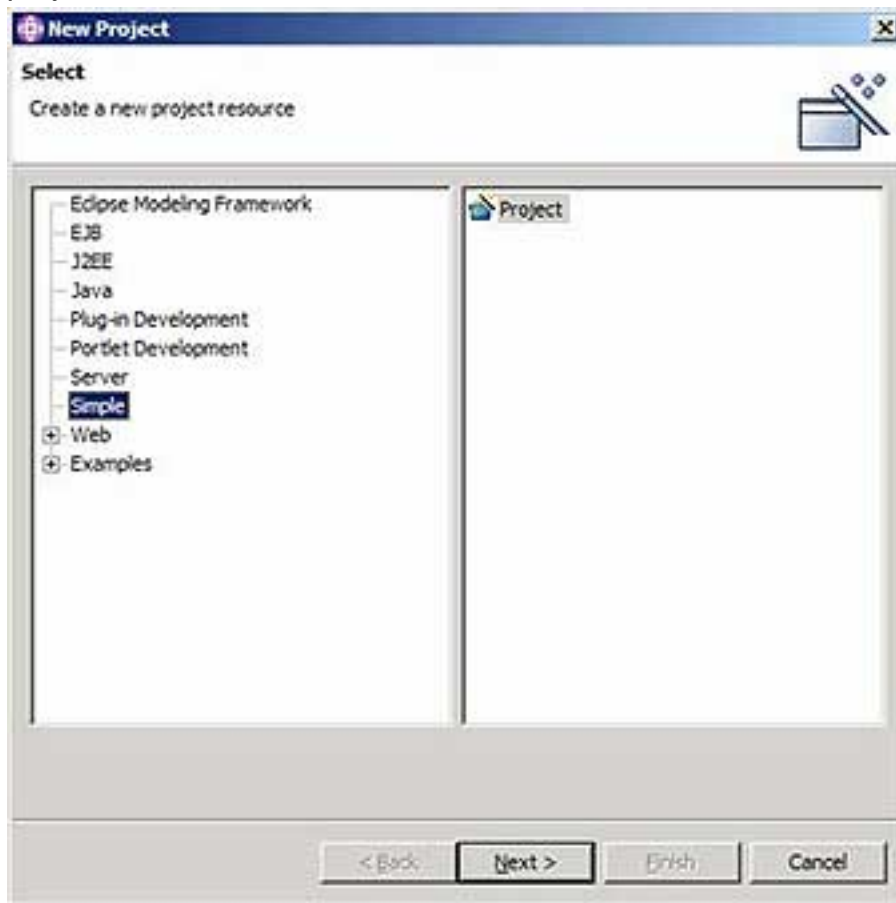
- The sample application is an auction portal site. Two portlets are running inside one portal page; both access a database.
  - The *Online Item* portlet shows items available on the auction site.
  - The *Item Detail* portlet shows detailed information about items available in the auction site. This portlet has another page that can update a record in an auction database.
- The two portlets use JSF controls.
- The portlets use WebSphere Data Object to access the database.
- The portlets use the Click-to-Action feature to process item IDs to get detailed information on each auction item.

### Setting up a database

For this tutorial, you'll use the Auction example database that comes with WebSphere Studio Application Developer V5.1.2. You'll use DB2 V8.2 for the

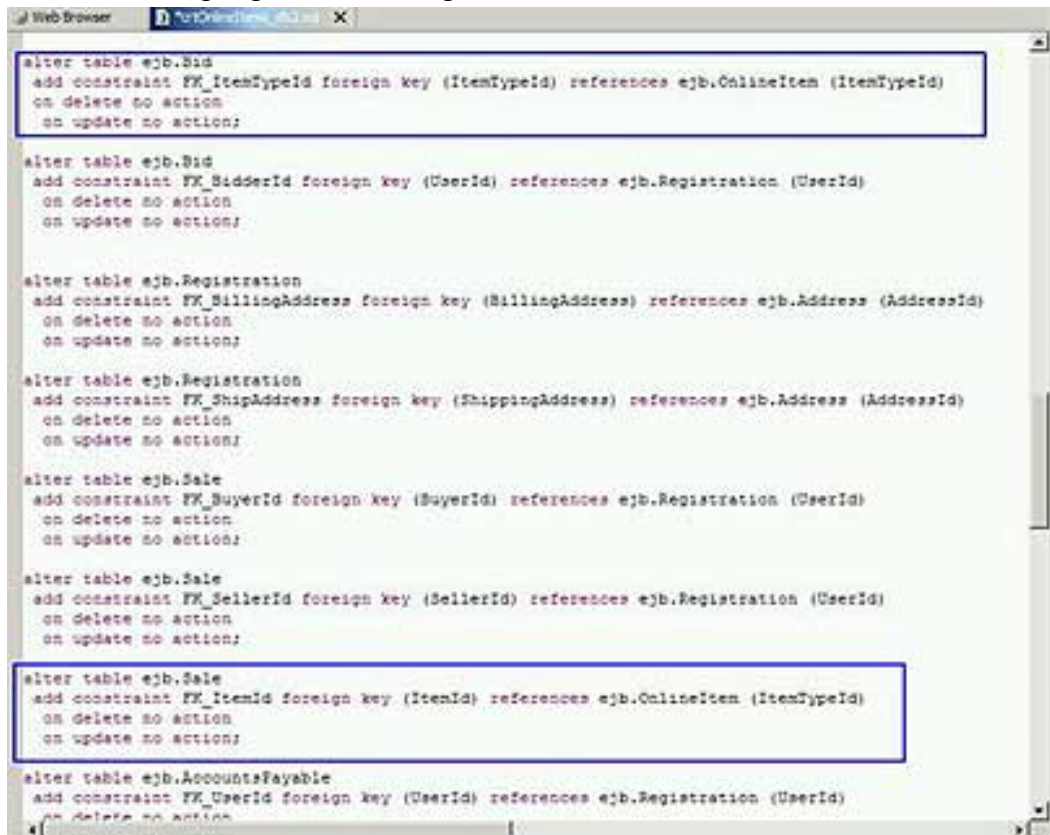
database. To get started:

1. Open a command prompt and enter `db2cmd` to open a DB2 command window.
2. Enter `db2 create db AUCTION` in the command window.
3. Start WebSphere Studio Application Developer 5.1.2.
4. Create a simple project by choosing **File=> New=> Project**. Call your project `Auction_DB`.



5. Switch to the Data perspective.
6. Choose **File=> Import=> File System** and import the SQL file `crtOnlineItems_db2.sql` from `<WebSphere Studio installed folder>\ samples\scenario_parts\auction\v51`. This file comes pre-installed with WebSphere.
7. You'll need to edit `crtOnlineItems_db2.sql` to delete or disable the SQL

statements highlighted in the figure below.



```
alter table ejob.Bid
add constraint FK_ItemTypeId foreign key (ItemTypeId) references ejob.OnlineItem (ItemTypeId)
on delete no action
on update no action;

alter table ejob.Bid
add constraint FK_BidderId foreign key (UserId) references ejob.Registration (UserId)
on delete no action
on update no action;

alter table ejob.Registration
add constraint FK_BillingAddress foreign key (BillingAddress) references ejob.Address (AddressId)
on delete no action
on update no action;

alter table ejob.Registration
add constraint FK_ShipAddress foreign key (ShippingAddress) references ejob.Address (AddressId)
on delete no action
on update no action;

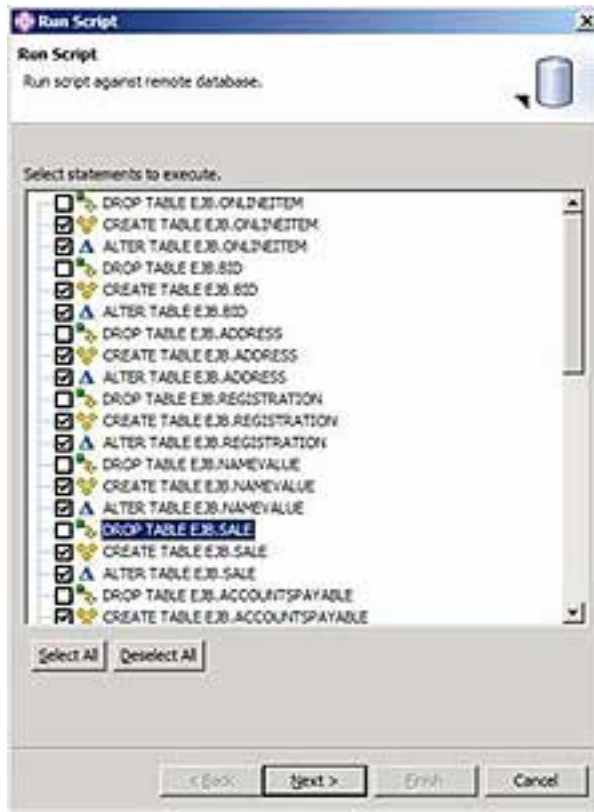
alter table ejob.Sale
add constraint FK_BuyerId foreign key (BuyerId) references ejob.Registration (UserId)
on delete no action
on update no action;

alter table ejob.Sale
add constraint FK_SellerId foreign key (SellerId) references ejob.Registration (UserId)
on delete no action
on update no action;

alter table ejob.Sale
add constraint FK_ItemId foreign key (ItemId) references ejob.OnlineItem (ItemTypeId)
on delete no action
on update no action;

alter table ejob.AccountsPayable
add constraint FK_UserId foreign key (UserId) references ejob.Registration (UserId)
on delete no action
```

8. Select the imported SQL statement and execute **Run on Database Server...** in the menu.
9. In the Run Script dialog, uncheck all the boxes for items beginning with DROP, then click **Next**.



10. In the Database Connection dialog, set parameters for the database (using DB2) as follows:
- Database: AUCTION
  - User ID: Enter the administration user ID you chose when you set up DB2
  - Password: Enter the password for the administration user ID you chose when you set up DB2
  - Database vendor type: DB2 Universal Database V8.1
  - JDBC driver: IBM DB2 APP DRIVER
- Then click **Finish**.



11. In the DB Servers view, select **Refresh** from the menu.

If the database was created successfully, the following tables should be displayed.



---

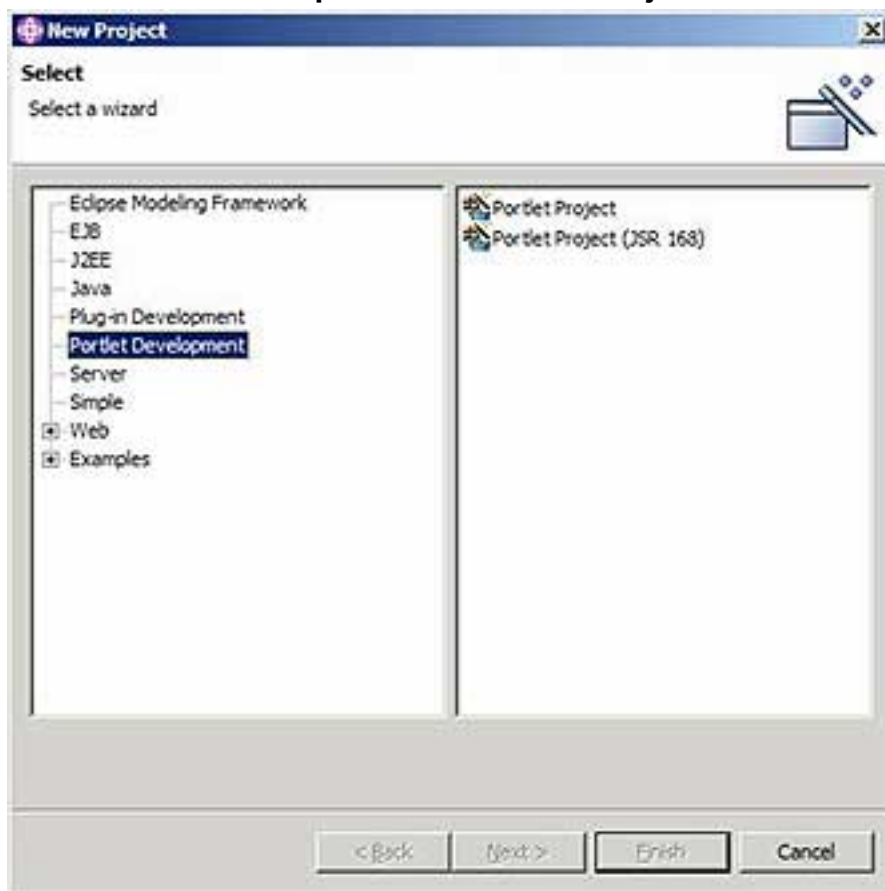
## Section 3. Creating the Online Item portlet

### Creating the portlet project

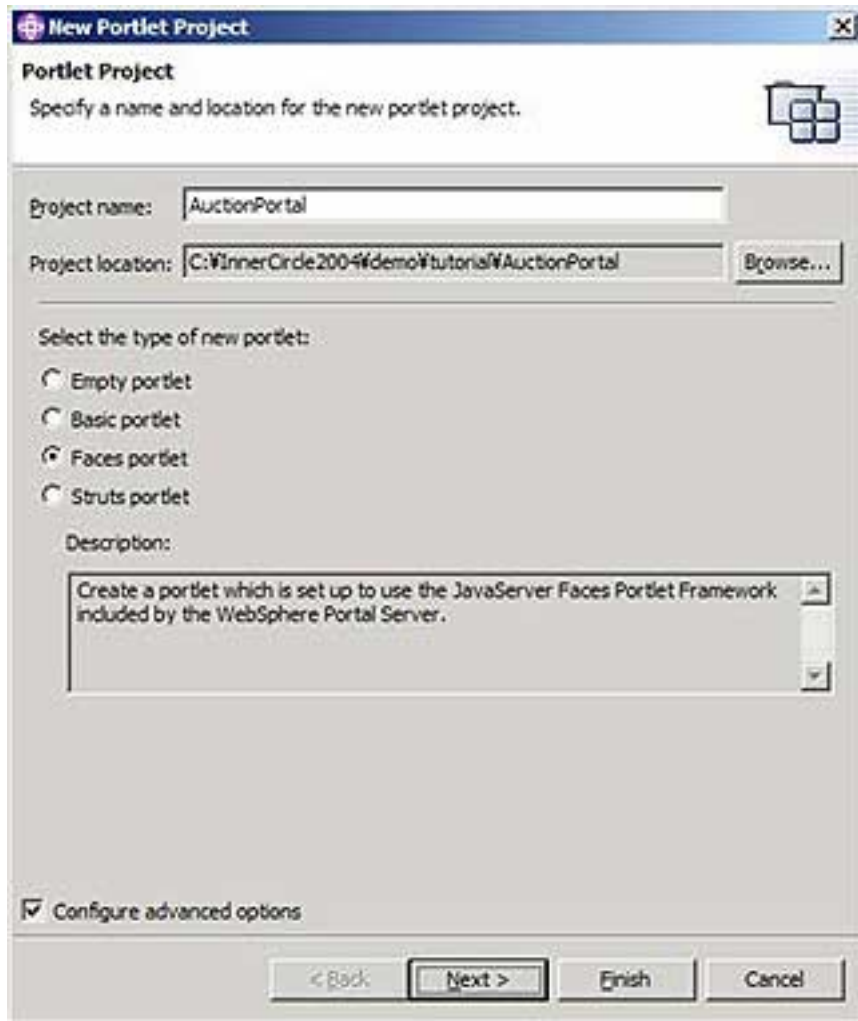
In this section, you'll create a new JSF portlet. The Online Item portlet will access items available in the Auction database that you created in the previous section.

To access the database information, you'll use WDO (WebSphere Data Object).

1. Select **New=> Project** to execute the New Project wizard. In the wizard, select **Portlet Development** and **Portlet Project**.



2. In the Portlet Project dialog, enter `AuctionPortal` as the project name and select **Faces portlet** to create a JSF portlet application. Then select **Configure advanced options** and click **Next**.



3. Click **Next** to skip the J2EE Settings and Features dialogs and go to **Portlet Settings**. Use `AuctionPortal List` as the value for **Application name**, **Portlet name** and **Portlet title**. Click **Next**.

**New Portlet**

**Portlet Settings**  
Define the general settings of the portlet.

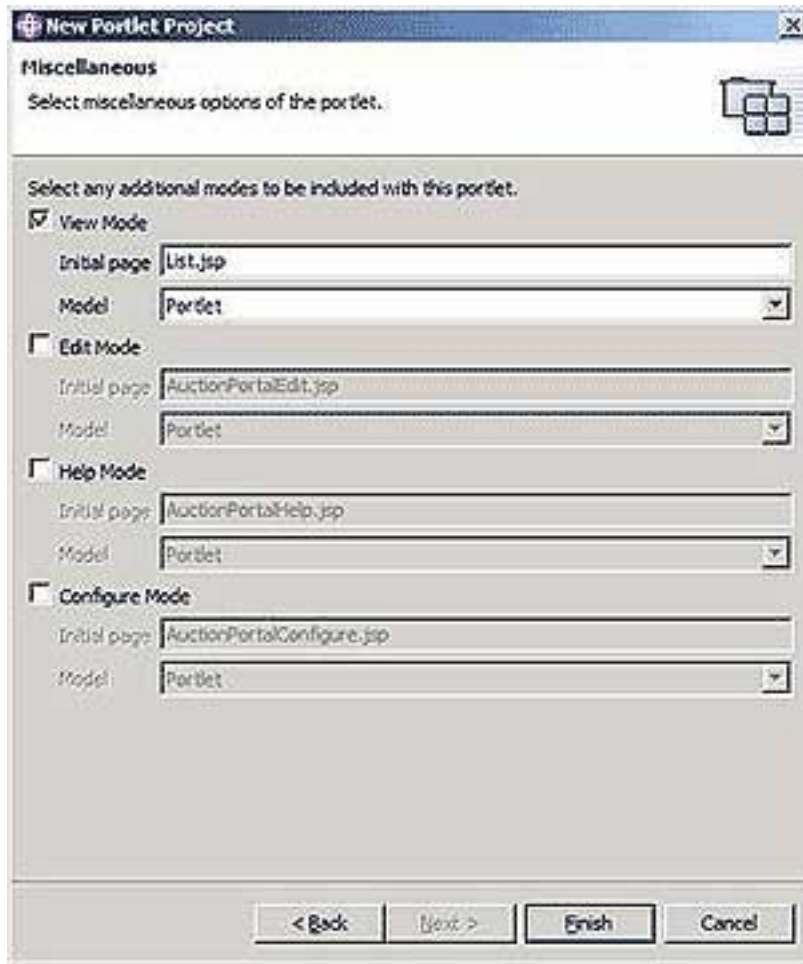
**General**  
Application name: AuctionPortal List  
Portlet name: AuctionPortal List

**Internationalization**  
Default locale: en English  
Portlet title: AuctionPortal List

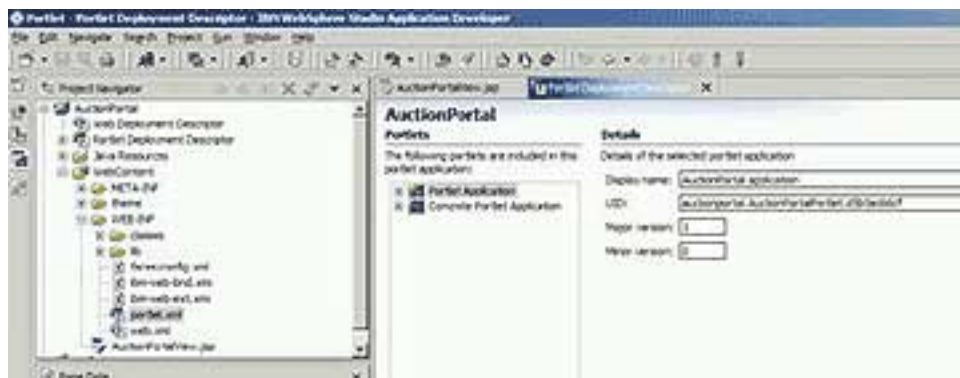
**Code generation options**  
 Change code generation options  
Package prefix: auctionportal  
Class prefix: AuctionPortalPortlet

< Back   Next >   Finish   Cancel

4. In the Miscellaneous dialog, change **Initial page** to `List.jsp` and then click **Finish**.



The Portlet perspective should now be open as shown below. For now, close the JSP file and Portlet Deployment Descriptor file.



## Creating a data table using WDO

You'll create a portlet JSP file that displays bid status information by accessing the

## Auction database using WDO.

1. Use Page Designer to open List.jsp. Delete the text `Place contents here`. Enter `Online Item`, and change its text style to **Heading 3** in the **Attributes** view. Then press Enter.



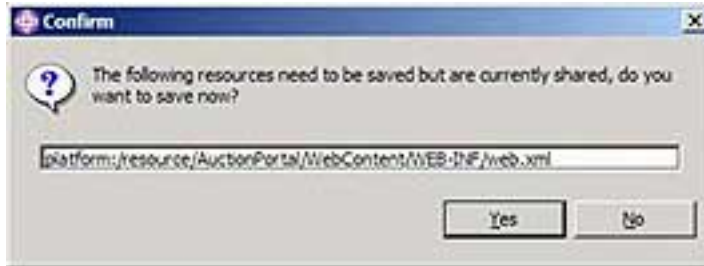
2. Drag a **Relational Record List** element from the Palette view and drop it on List.jsp under **Online Item**.



3. At this point, you will see a warning dialog about using WDO in WebSphere Portal 5.0 environment. This configuration is not supported by WebSphere Portal 5.0, but it is enabled for your prototyping. Click **Yes** to continue.



4. portlet.xml will now be open, and the following dialog will be displayed. Click **Yes** to continue.



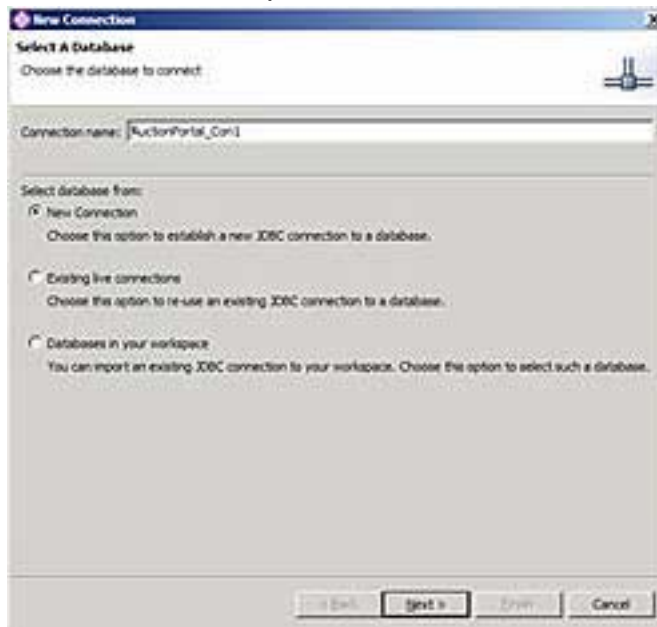
5. In the Relational Record List dialog, enter `items` for a reference name for the record list and then click **Next**.



6. In the Record List Properties dialog, click **New** to create a new database connection.



7. Click **Next** to accept the defaults in the Select a Database dialog.

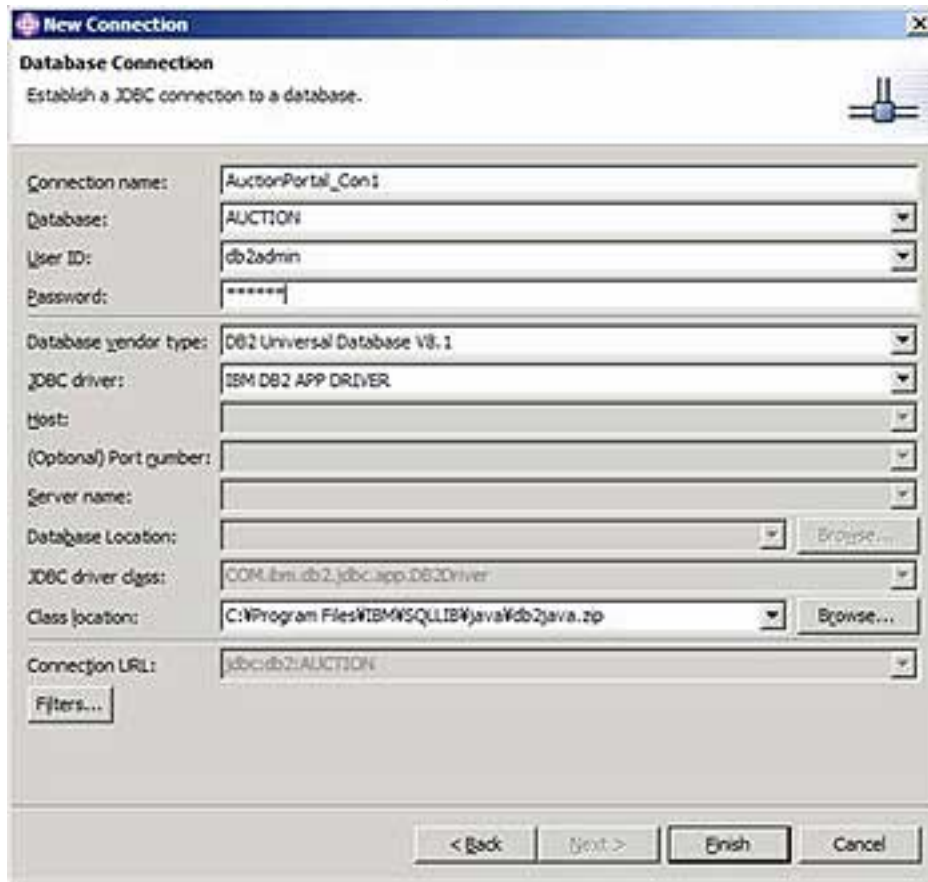


8. In the Database Connection dialog, enter the following parameters for connecting to the Auction database:
  - Database: AUCTION
  - User ID: The ID for the administrator user you chose when you set up DB2
  - Password: The password for the ID for the administrator user you

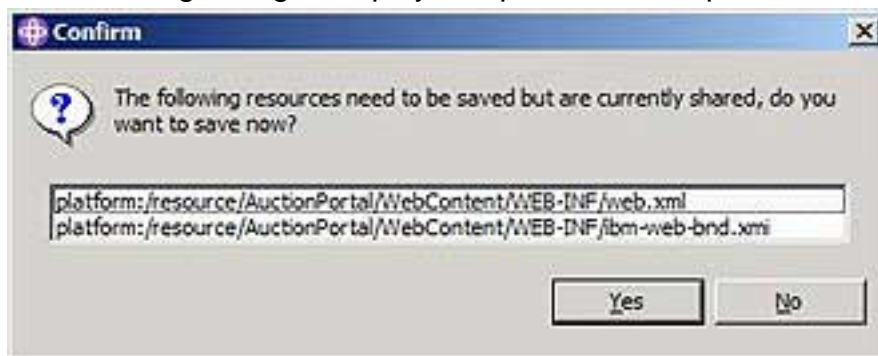
chose when you set up DB2

- Database vendor type: DB2 Universal Database V8.1
- JDBC driver: IBM DB2 APP DRIVER

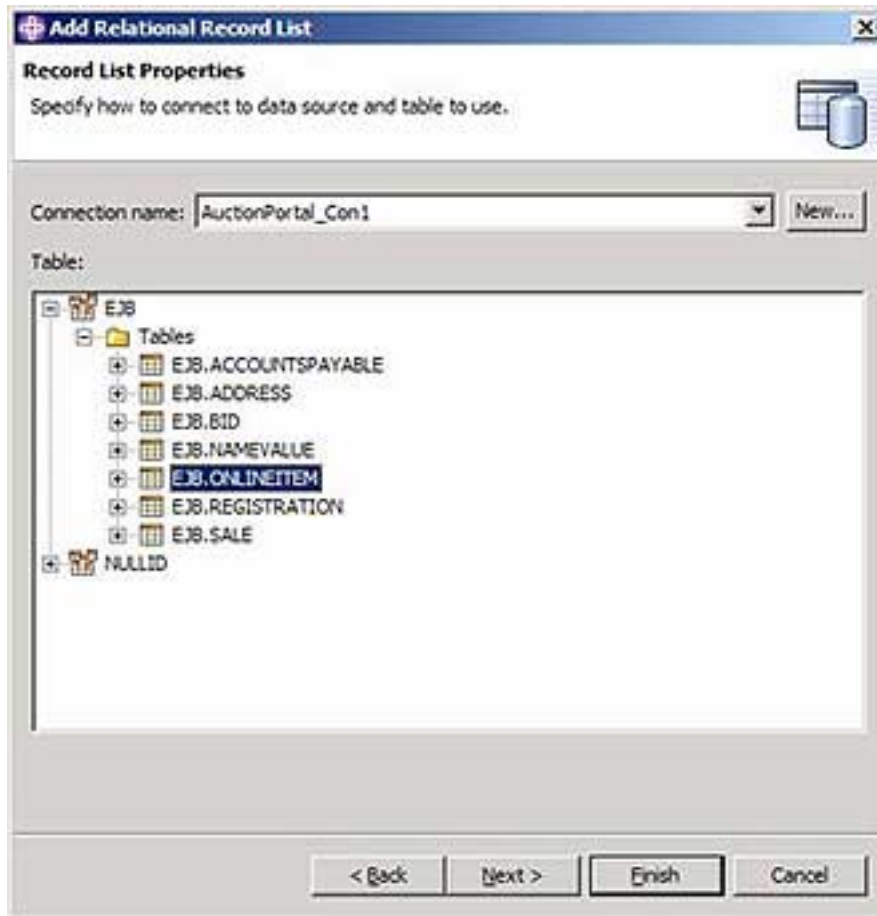
Once you've made these changes, click **Finish**.



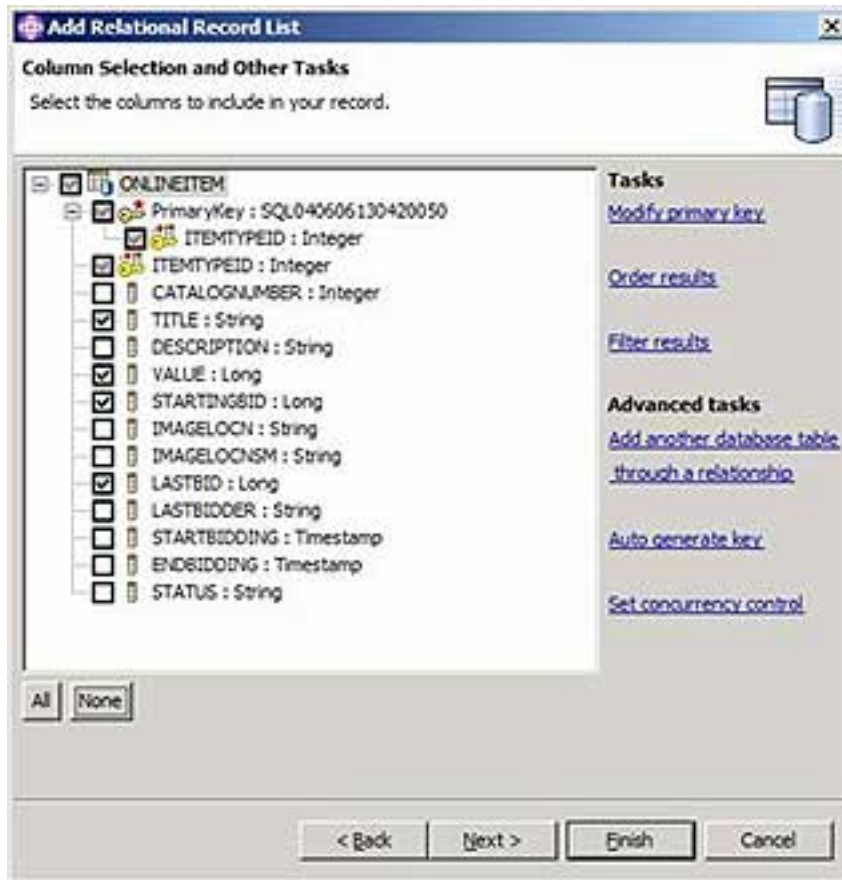
9. The following dialog is displayed if portlet.xml is open. Click **Yes**.



10. In the Record List Properties dialog, select the EJB.ONLINEITEM table and click **Next**.

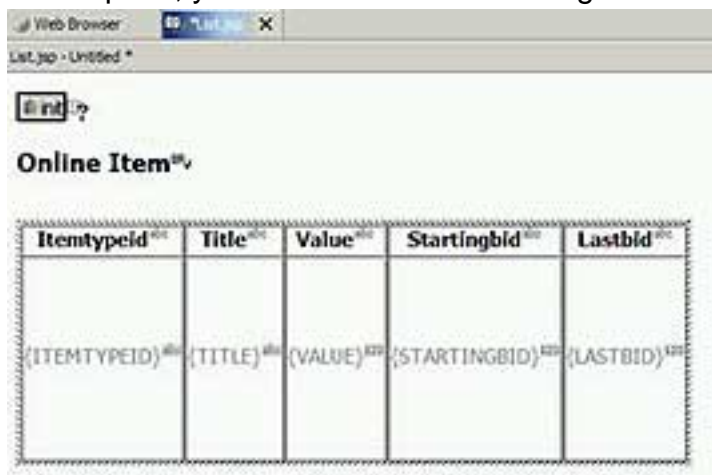


11. In the Column Selection and Other Tasks dialog, select items from the EJB.ONLINEITEM table as shown in the following figure.

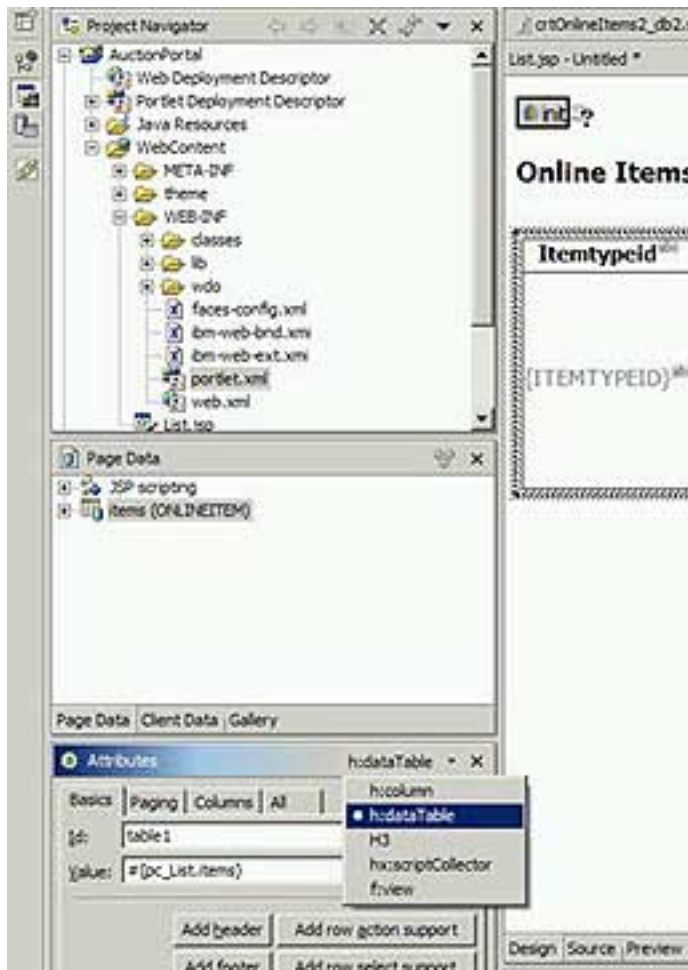


Then click **Finish**.

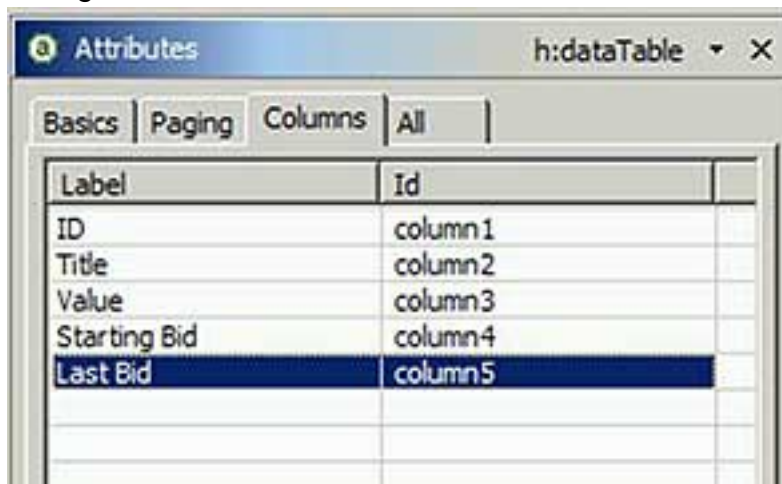
- At this point, you should see the following data table in List.jsp:



- Now you need to change the title of each column. Click on the data table and select **h:dataTable** in the Attributes view.



14. Select the Columns tab and edit the label for each column as shown in the figure below.

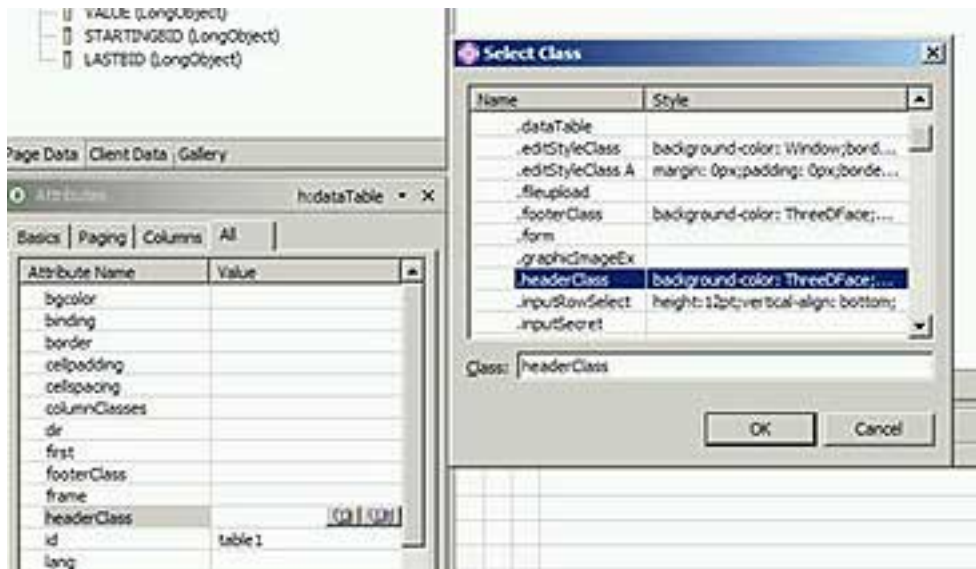


15. Now you need to change the style of the data table to specify a header

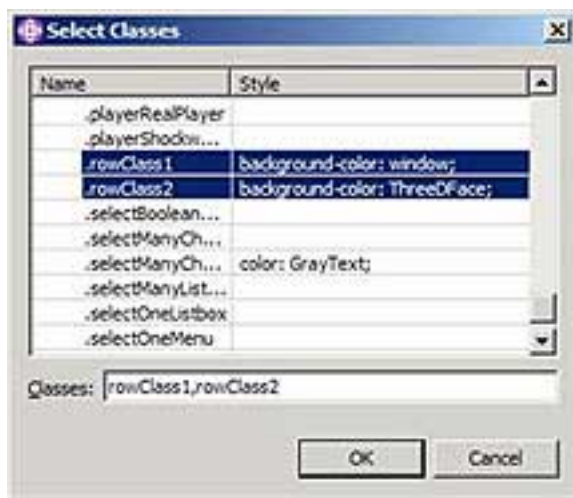
style. Click on the All tab and select **headerClass**, then click the Select Class button, which looks like this:



This will show the Select Classes dialog. In this dialog, choose **.headerClass** and click **OK**.



16. Now you need to change the style of the rows. To improve readability, you'll use two alternating colors as the background for the rows. Select **rowClasses** and click the Select Class button to show the Select Classes dialog again. In this dialog, choose **.rowClass1** and **.rowClass2** and click **OK**.



17. Now the data table should look like this:

Online Item

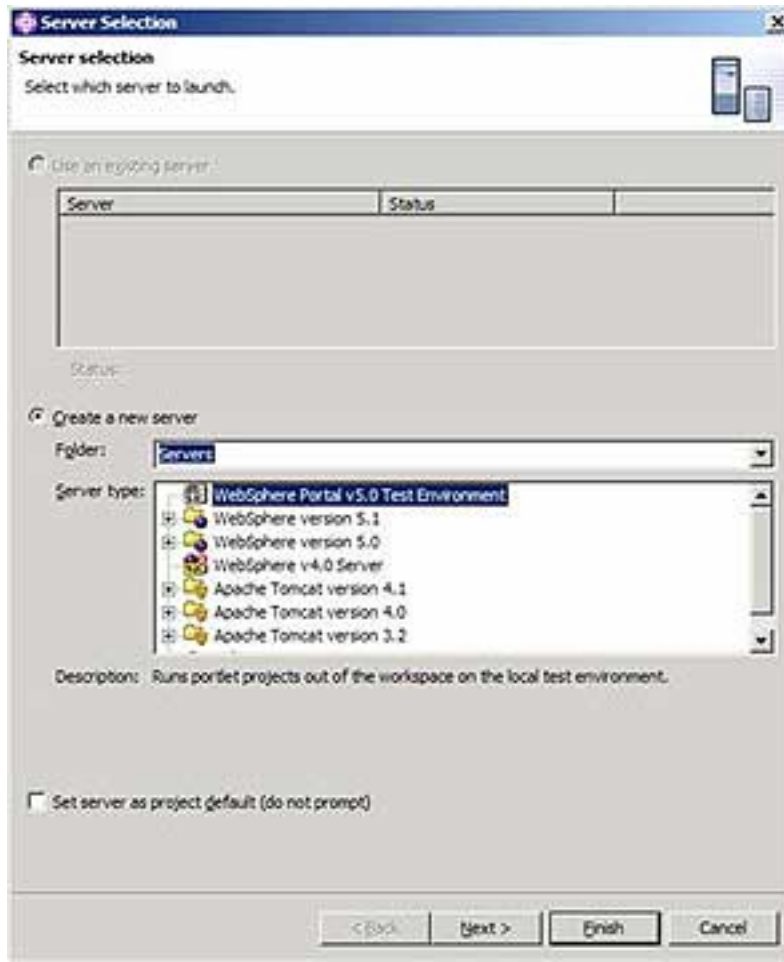
ID	Title	Value	Starting Bid	Last Bid
{ITEMTYPEID}	{TITLE}	{VALUE}	{STARTINGBID}	{LASTBID}

Save List.jsp.

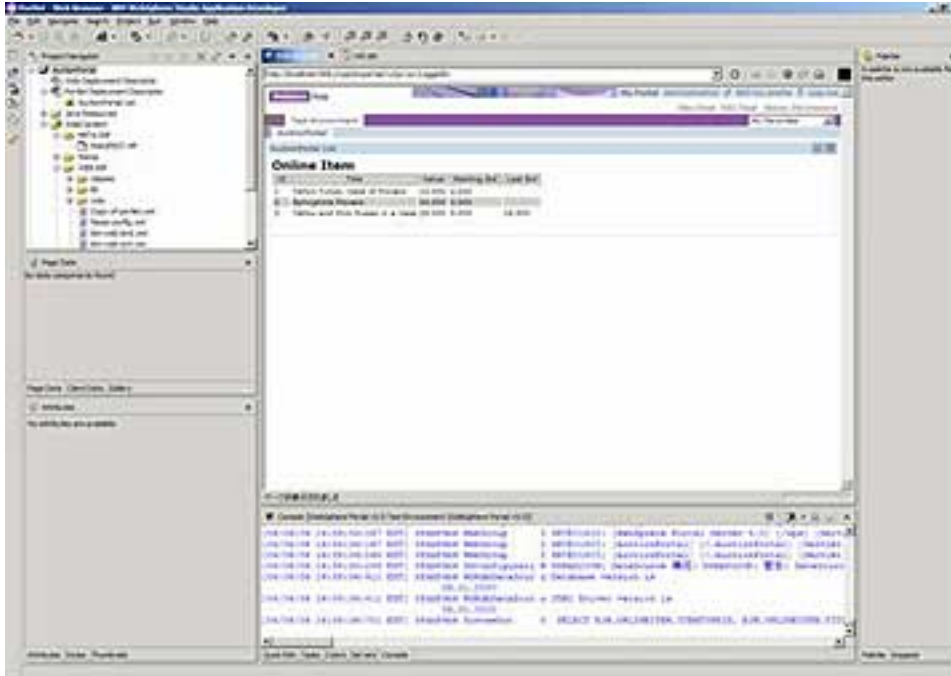
- Now you need to test the created portlet in the WebSphere Portal Test Environment. Right-click **AuctionPortal** in the Project Navigator view and select **Run on Server**.



- At this point, the Server Selection wizard will launch automatically. Select **WebSphere Portal v5.0 Test Environment** and click **Finish**.



WebSphere Application Studio and WebSphere Portal will now start, and the portlet you created will be initialized, as shown in the figure below.



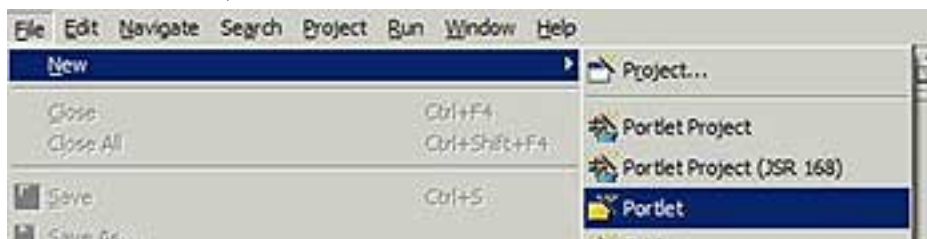
Once you confirm that the portlet is running, stop the server in the Server view. If you don't stop it now, you'll need to reboot it later.

## Section 4. Creating the Item Detail portlet

### Adding the Item Detail portlet to the AuctionPortal project

In this section, you'll create a JSF portlet that shows detailed information for a selected item that is up for auction on the site.

1. From the menu, choose **File=> New=> Portlet**.



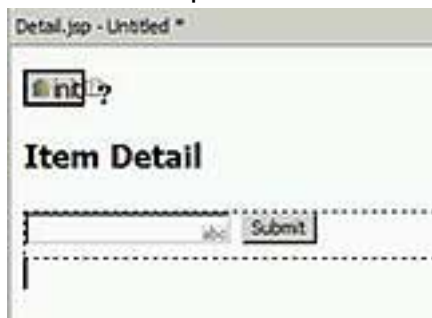
2. On the Portlet dialog, enter **AuctionPortal** as the value for **Project**, click the **Faces** portlet radio button, and select the **Configure advanced**

**options** checkbox, then click **Next**.



3. Click **Next** to skip the Features page and go to **Portlet Settings**. Change **Application name**, **Portlet name**, and **Portlet title** to AuctionPortal Detail. Then click **Next**.

4. On the next screen, change Initial page to `Detail.jsp` and click **Finish**.
5. You'll now need to edit `Detail.jsp`. In the Attributes view, enter the text `Item Detail` and change its text style to **Heading 3**. Drag the **Input Faces** control from the Palette view and drop it under the text you just entered, and drag **Command-Button** control from the Palette and drop it next to the input field.



## Accessing detailed item information usingWDO

Now you'll create a table that shows detailed information about a selected item in the Auction database.

1. Select **Relational Record** in the Palette view and drag and drop it into Detail.jsp under the input field.



2. In the Relational Record dialog, enter `detail` as the reference name for the record, select the radio button for **Display an existing record (read-only)**, and click **Next**.

**Add Relational Record**

**Relational Record**  
Enter a reference name for record.

Name:   
Create a name to refer to this record within the page.

Reuse metadata definition from an existing record or record list

Input file:

You can automatically add data controls to your page to work with this record (configure details on last page of this wizard).

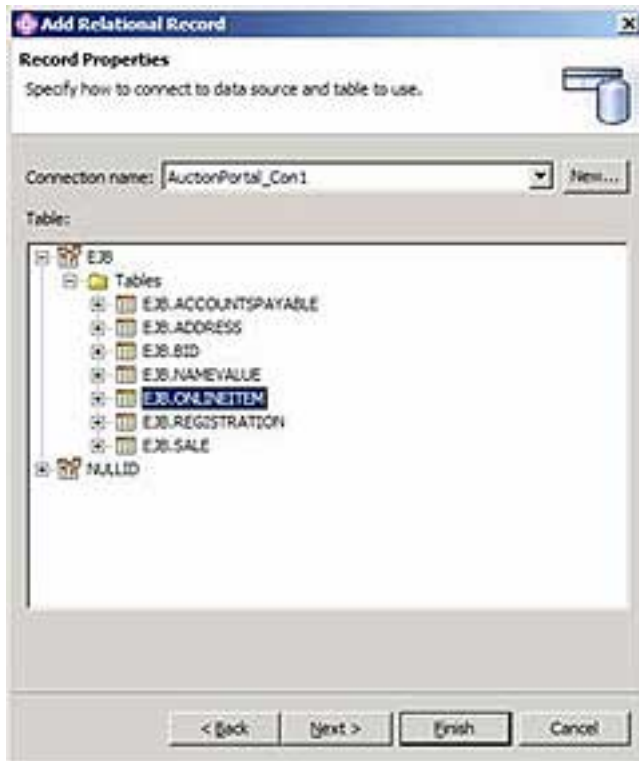
Add input/output controls to display the record on the web page

Create controls for:

- Displaying an existing record (read-only)
- Updating an existing record
- Creating a new record

< Back   **Next >**   Finish   Cancel

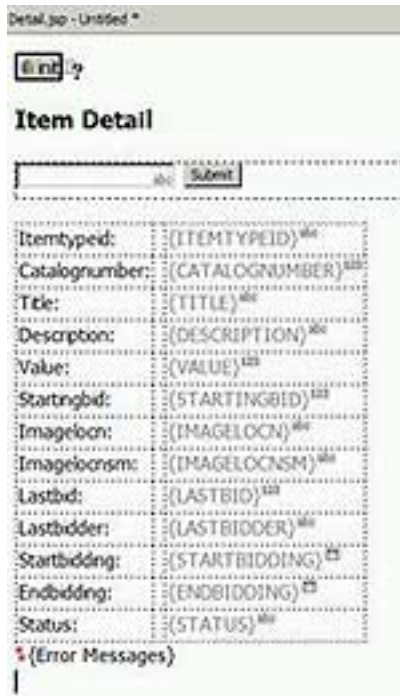
3. In the Record Properties dialog, select **EJB.ONLINEITEM** and click **Next**.



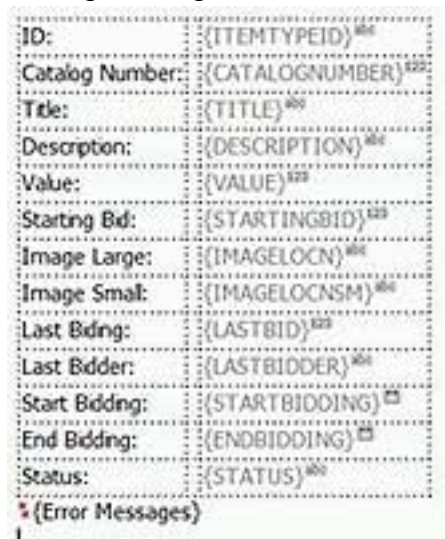
4. In the Column Selection and Other Tasks dialog, confirm that all columns are selected and click **Finish**.



- At this point, you will see the following table in Detail.jsp.

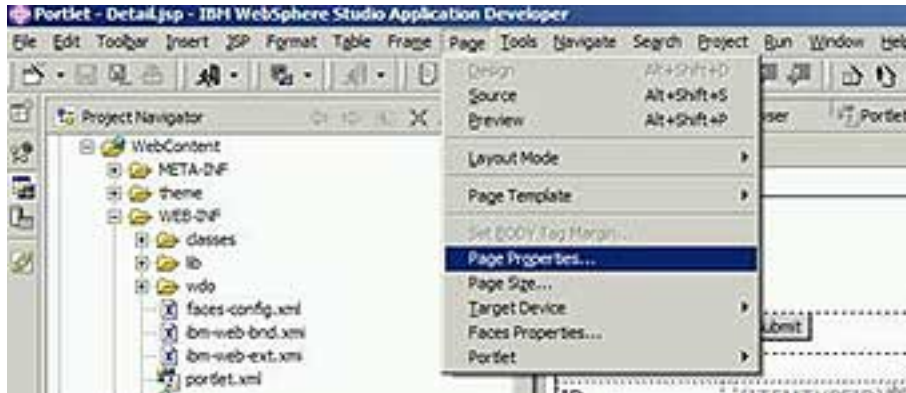


In Page Designer, edit all the row labels as shown in the following figure.

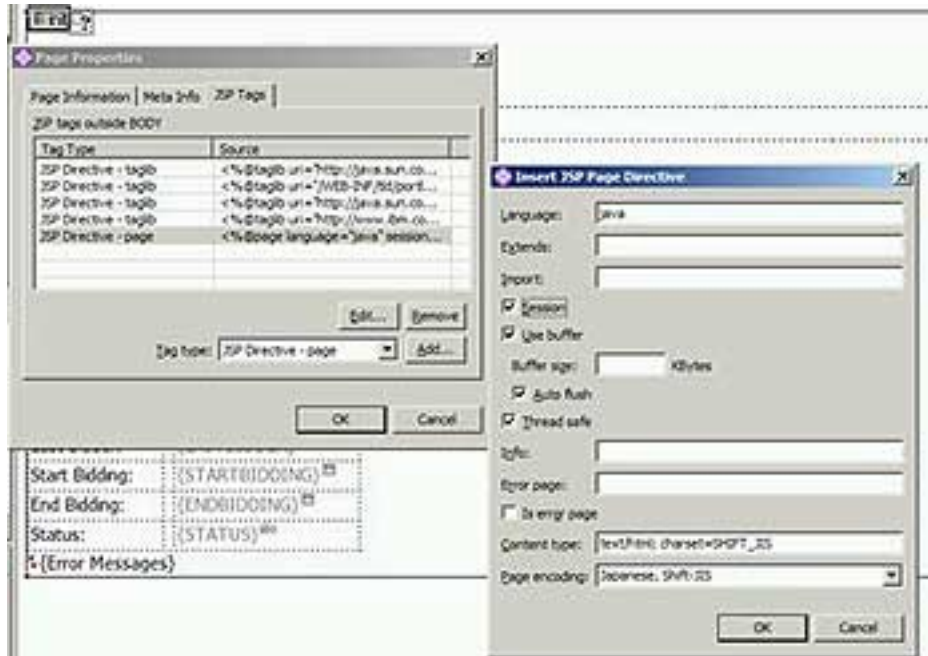


Now you need to create a new variable to bind user input data. This variable will be used by a database query to find detailed information about the item corresponding to the ID being entered in the input field. You'll use this variable in the session scope of this application. To use this variable in a session, you'll first need to enable sessions for Detail.jsp.

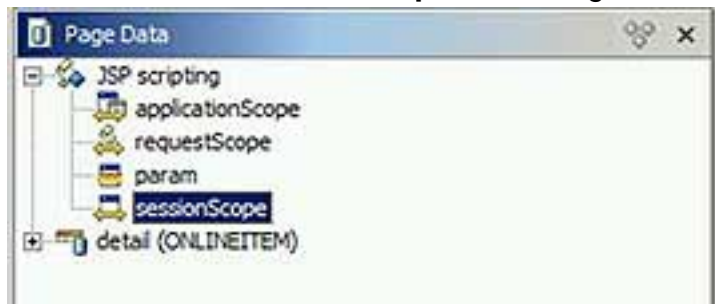
1. Click **Detail.jsp** and go to the **Page=> Page Properties** menu.



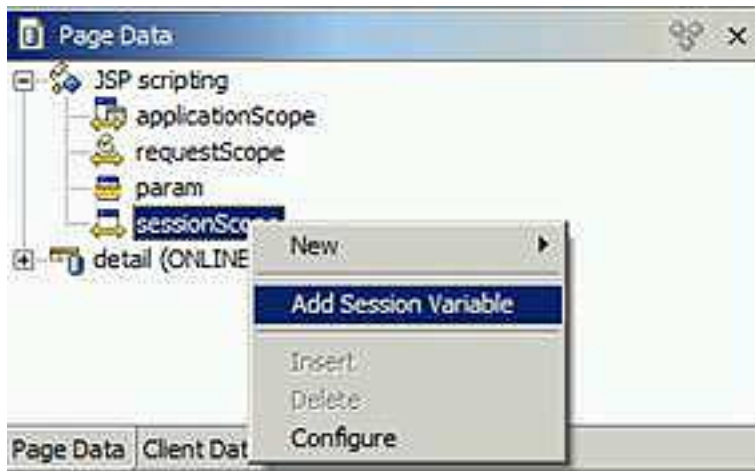
2. Click **JSP Tags** and select **JSP Directive - page**, then click the Edit button. Select **Session** on the Insert JSP Page Directives dialog, then click **OK**.



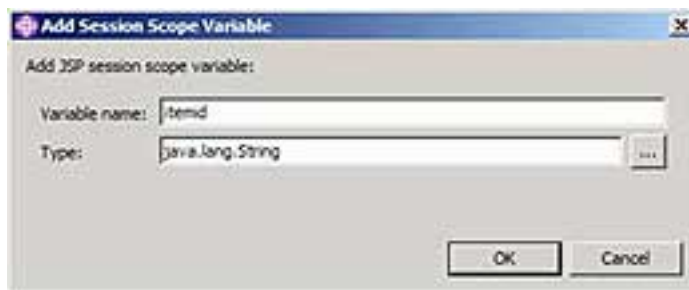
You should see **sessionScope** in the Page Data view.



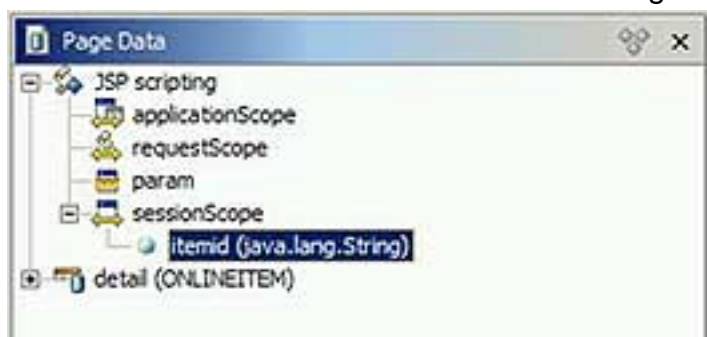
3. Select **sessionScope**, then right-click and select **Add Session Variable** in the menu.



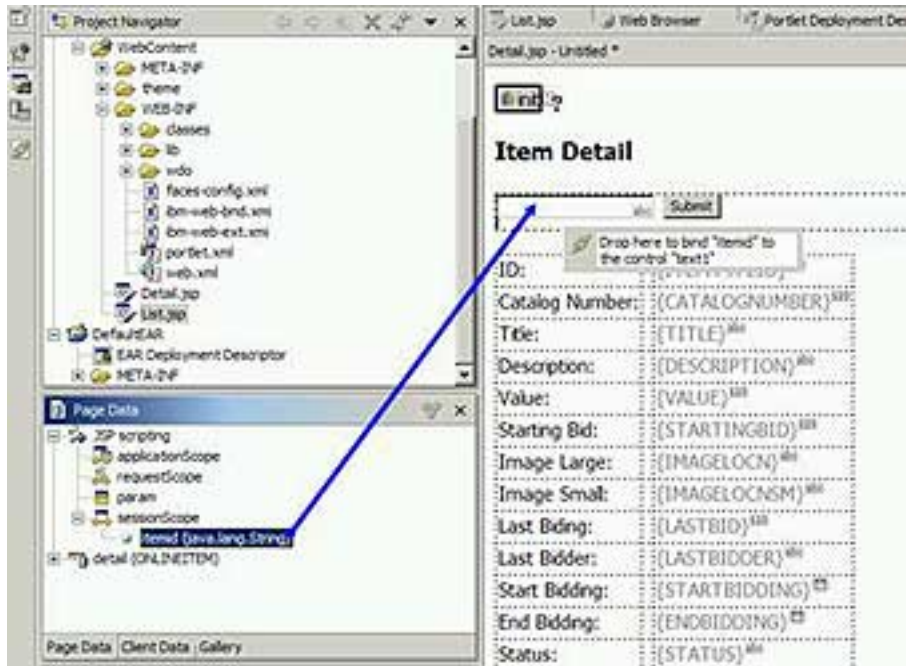
4. Enter `itemid` for Variable Name and `java.lang.String` for Type, then click **OK**.



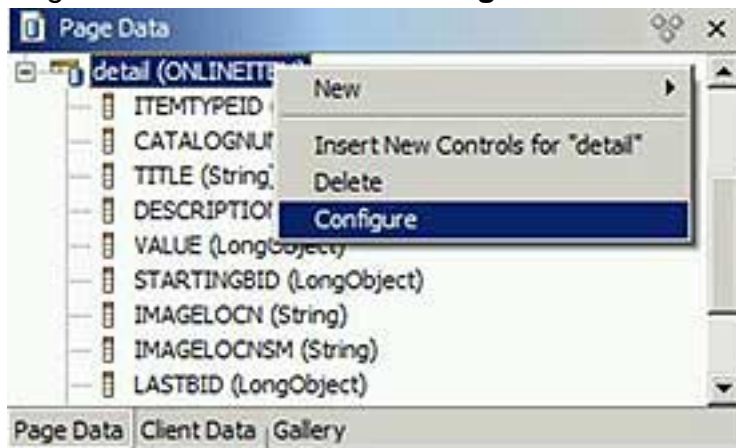
Your variable should now be shown in the Page Data view.



5. Now you need to bind `itemid` to the input text field. Drag and drop `itemid` from the Page Data view onto the input text field in Page Designer.



- Now itemid must be passed to a database query. Right-click **detail** in the Page Data view and select **Configure**.



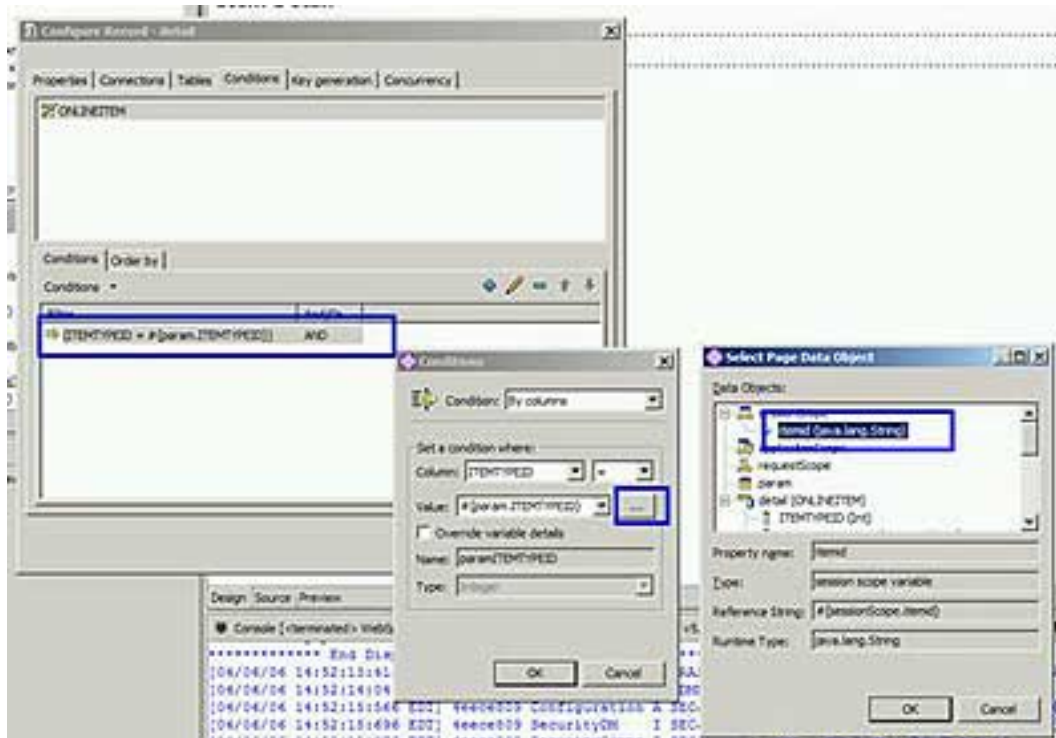
- Click the **Conditions** tab and select **(ITEMYPEID = ...)**, then click the Edit button, which looks like this:



- Click the button for selecting page data objects, which looks like this:



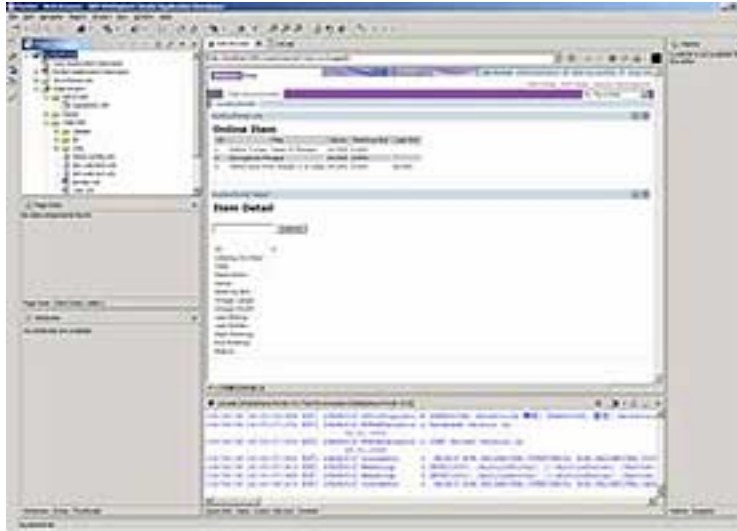
You'll see this button next to the Value input field. Select **itemid** and click **OK** on the Select Page Data Object and Condition dialogs. Then, click **Close** on Configure Condition - Detail dialog.



9. Save Detail.jsp.

Now you're ready to run the Item Detail portlet in the WebSphere Portal Test Environment.

1. First, you need to start the environment; if it's already running, you'll need to restart it. Go to the Servers view and right-click **WebSphere Portal V5.0 Test Environment**, then select **Start** or **Restart** as appropriate in the context menu.
2. Next, right-click **Auction Portal** in the Project Navigator view and select **Run on Server**. Two portlets will appear as illustrated in the following figure.



- Enter a number between 1 and 3 in the input text field. Detailed item information will be displayed.



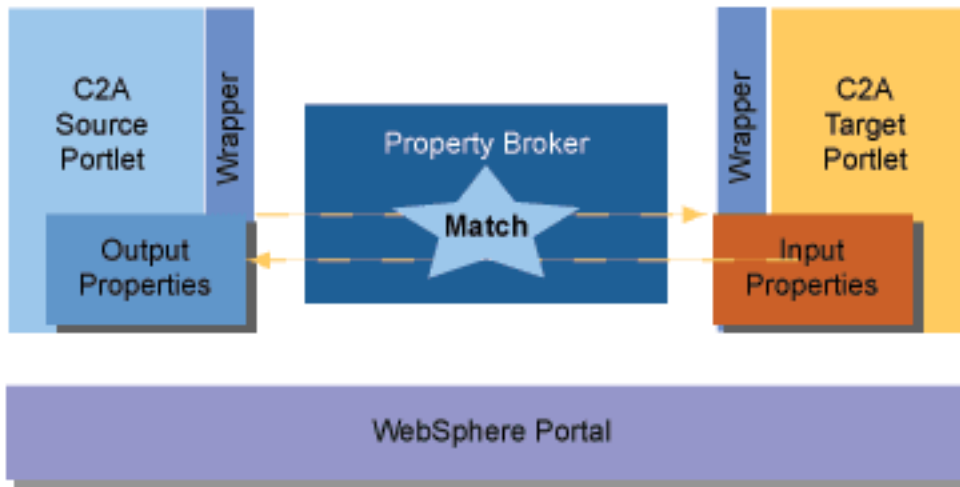

---

## Section 5. Enabling Click-to-Action

### Introduction

In this section, you'll enable communication between your two portlets using

WebSphere's Click-to-Action feature. Click-to-Action (C2A) is used to create cooperative portlets. It enables end users to transfer data from one portlet (*source portlet*) to one or more other portlets (*target portlets*) through pop-up menus displayed next to the data to be transferred.

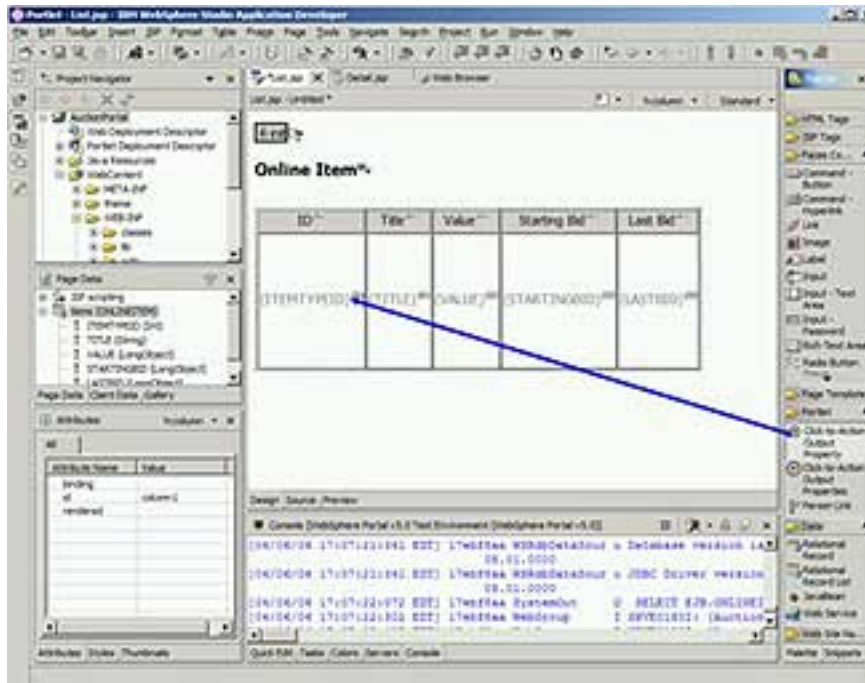


See [Resources](#) for more information on C2A.

## Enabling Click-to-Action, with the Online Item portlet as a source portlet

Now you'll enable Click-to-Action functionality between the Online Item portlet and Item Detail portlet. The user will be able to display detailed information about an item shown in the Online Item portlet using Click-to-Action.

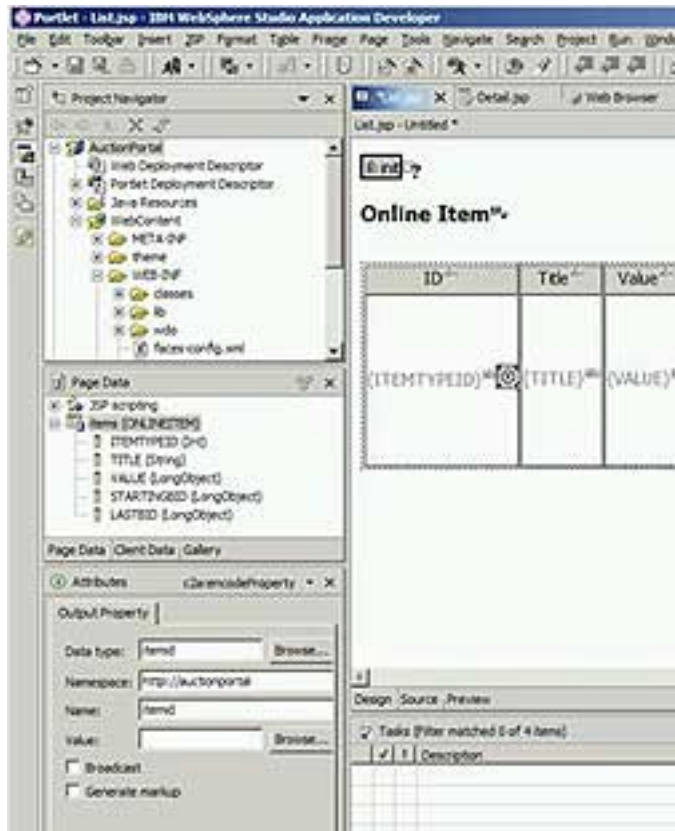
1. Open List.jsp in Page Designer.
2. Drag the **Click-to-Action Output Property** control from the Portlet drawer in the Palette view and drop it next to **{ITEMTYPEID}** in the data table.



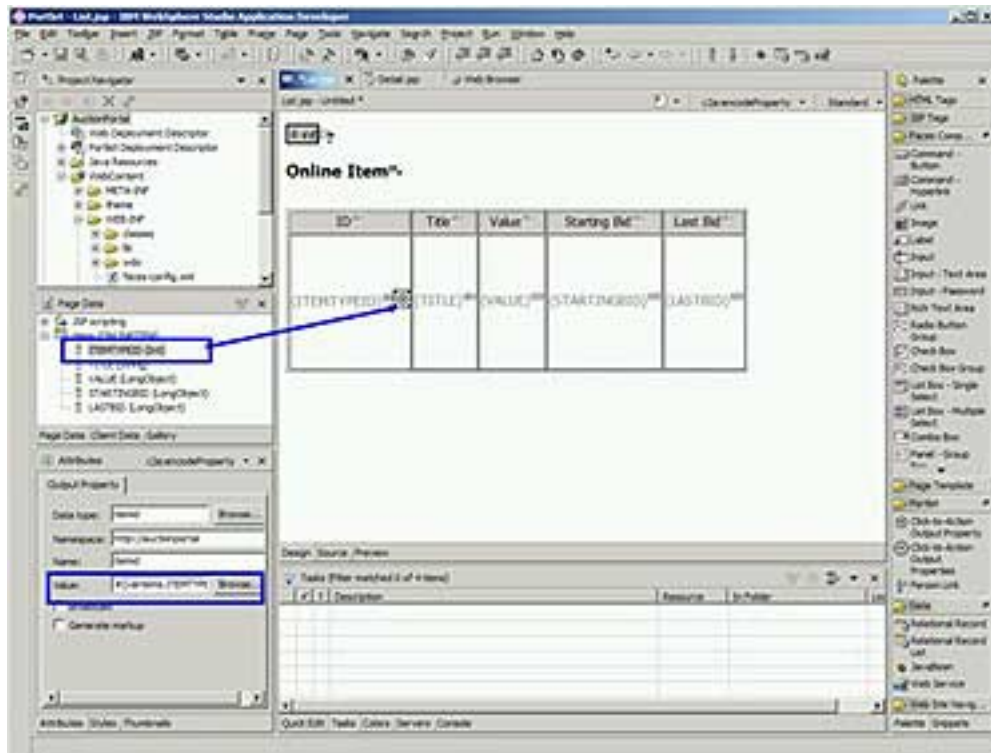
3. In the Insert Click-to-Action Output Property dialog, enter `itemid` for the data type and select **AuctionPortal List** from the Portlet drop-down menu. A WSDL file will be created by this operation; name it `List`.



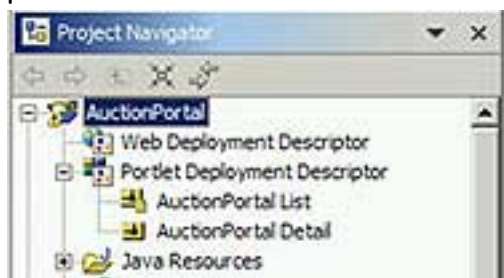
A down-arrow button should now be displayed next to `{ITEMTYPEID}`.



- Now you need to specify an item ID that will be passed to the target portlet to show its detailed information. Drag **ITEMTYPEID** from the Page Data view and drop it onto the Click-to-Action button in Page Designer. This action will set the Value in the Attributes view to c2a:encodeProperty, as shown in the figure below.



At this point, if you look into the Project Navigator, you will see a source portlet icon on the AuctionPortal List portlet.

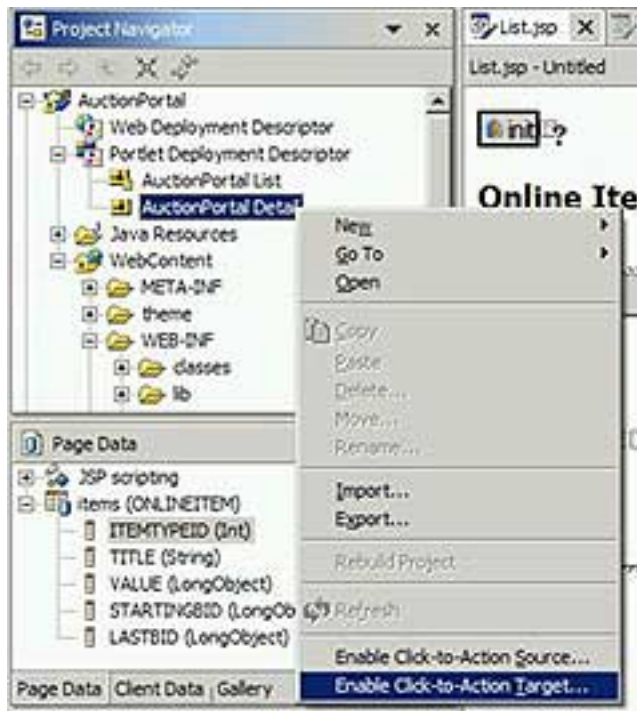


5. Save and close List.jsp.

## Enabling Click-to-Action in the Item Detail portlet as a target portlet

Now you'll enable Click-to-Action in the Item Detail portlet as a target portlet, to process an item ID passed from the Online Item portlet to show detailed item information.

1. In the Project Navigator, right-click **AuctionPortal Detail** and select **Enable Click-to-Action Target**.



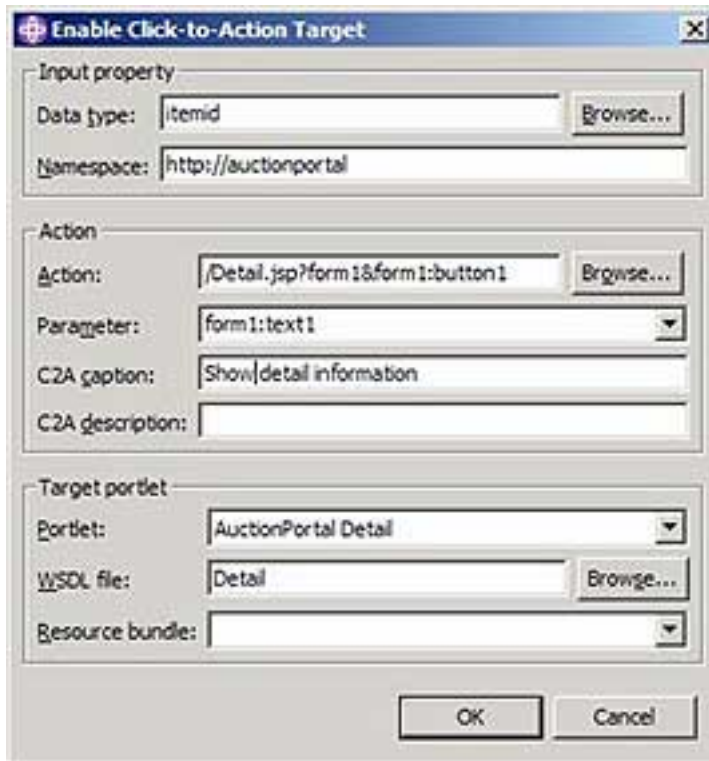
2. In the Enable Click-to-Action Target dialog, set parameters as follows:
  - For Data Type, click the Browse button and select **List.wsdl**. (You can find this file under AuctionPortal\WebContent project.) itemid will then appears in the Data Type field.



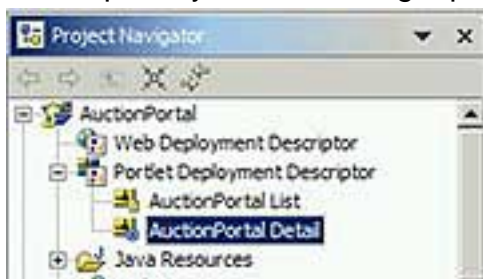
- Click the Browse button next to the Action field and select **button1** under **/Detail.jsp=> form1**. Then click **OK**.



- Enter Show detail information... as the **C2A caption**. This text will be shown in the context menu when the end user clicks on a portlet's Click-to-Action button.
- Enter detail as the WSDL file. Then Click **OK**.



At this point, you'll see a target portlet icon on the AuctionPortal detail.



## Testing Click-to-Action functionality

Now you need to see if the Click-to-Action functionality works. Restart the WebSphere Portal Test Environment. You'll see the Click-to-Action icon, illustrated in the following figure, next to the names of items in the Online Item portlet.



Click this icon and select **Show detail information....** You will see the detailed information for the selected item in the target portlet, as shown below.



## Showing detailed information by wired actions

When using Click-to-Action as you did in the previously, each time the user sends an item ID from a source portlet to a target portlet, a context menu is shown and a menu item must be selected. You can also send data merely by clicking on a Click-to-Action icon; this is called a *wired action*. This can be specified on running portlet applications, as you'll see on this panel.

1. Press the Ctrl key and click the Click-to-Action icon, which looks like this:



Select **Show detail....** You should see the following dialog:



2. Click **Yes**. The Click-to-Action icons will be changed into the following wired action icon:



This icon indicates that a wired action is set.



3. Click the wired action icon. Detailed information about the selected item ID will be shown in the target portlet automatically.
4. To disable this wired action, press the Ctrl key and click the wired action icon. Click **Yes**.



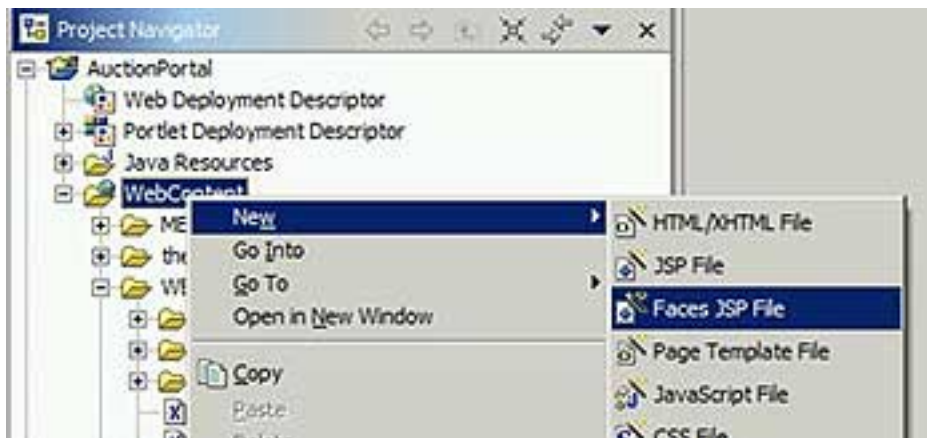
All of the wired action icons will be transformed back into the original Click-to-Action icons.

## Section 6. Creating a page in the Item Detail portlet

### Creating a new page for updating a record

Now, you'll create a JSP page in the Item Detail portlet with which the user can update a database record.

1. In the Project Navigator view, right-click **WebContent** and select **Faces JSP File**.



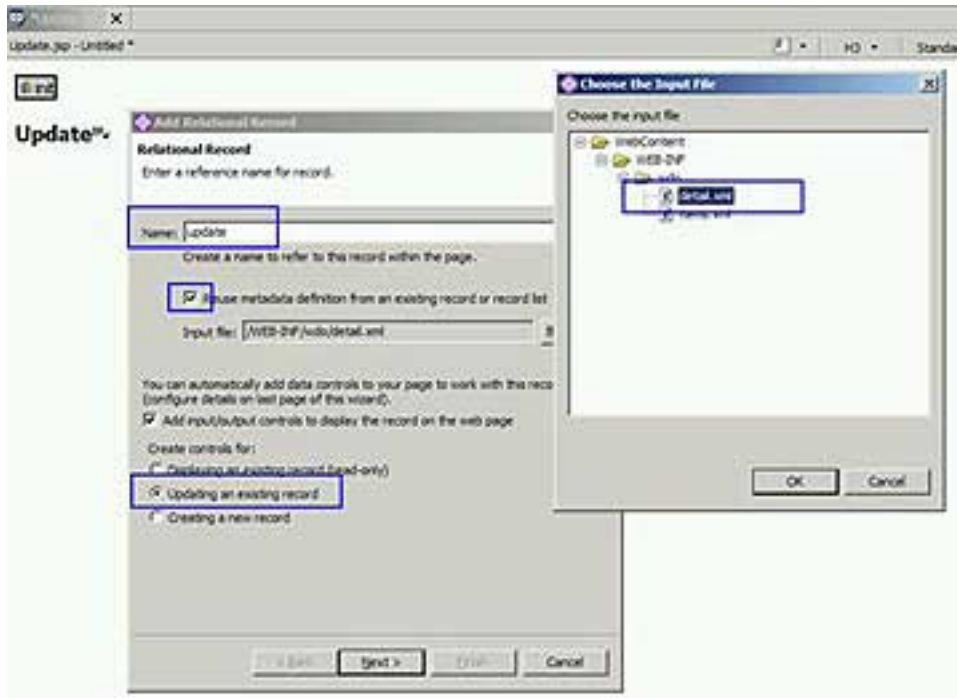
2. In the Faces JSF File dialog, enter `Update` as the **File Name** and click **Finish**.



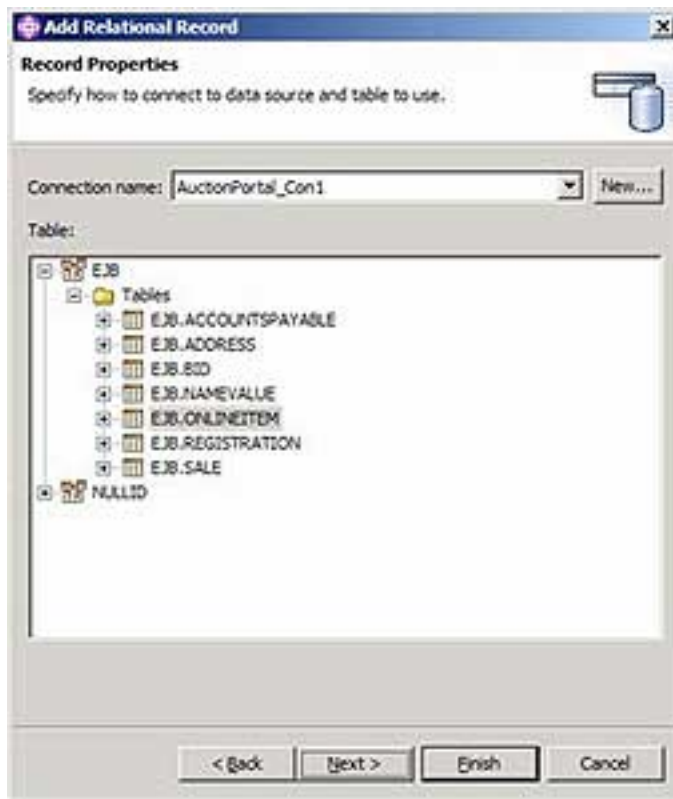
3. Enter `Update` in `Update.jsp` and change its text style to **Heading 3** in the Attributes view, then click **Enter**.



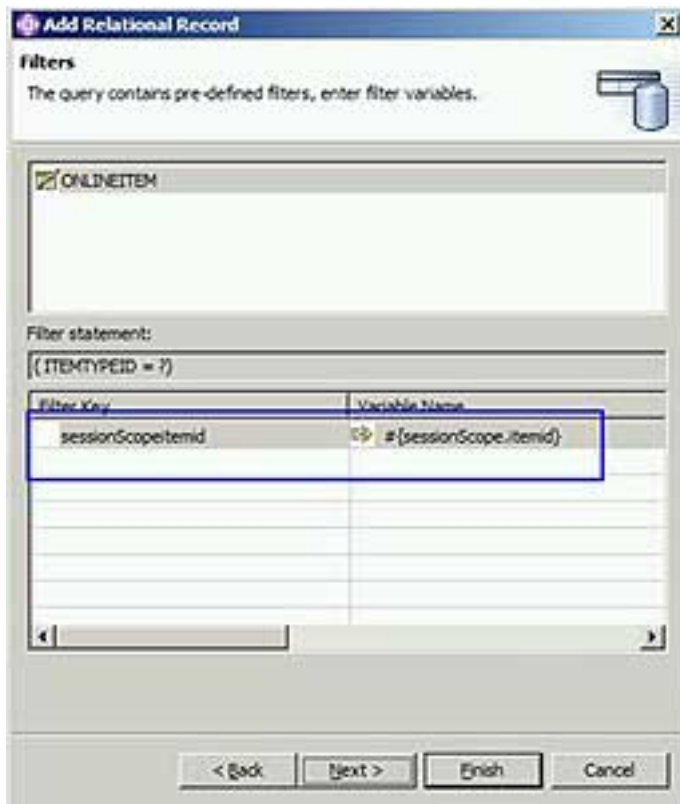
4. Drag a **Relational Record** item from the Palette view and drop it into `Update.jsp`. Enter `update` as its Name, select the **Reuse metadata definition from an existing record or record list** radio button, and choose `detail.xml` as the input file. Select the **Updating an existing record** radio button and click **Next**.



5. In the Record Properties dialog, confirm that EJB.ONLINEITEM is selected and click **Next**.



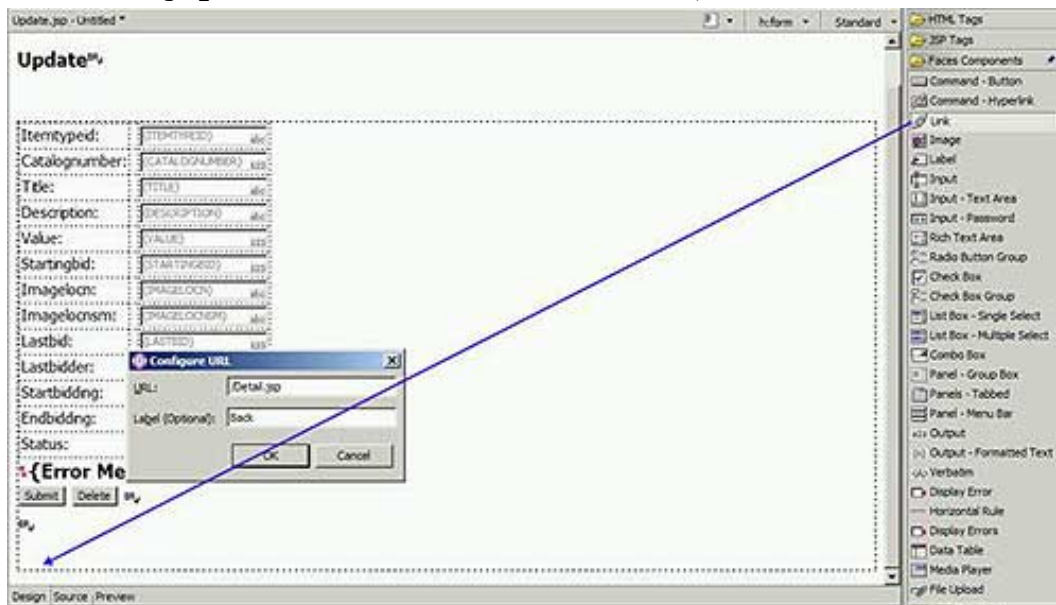
6. In the Filters dialog, confirm that itemid is displayed in sessionScope and click **Finish**.



You should see the following table, which contains input text fields to update records.



- Now you need to create a link to go back to Detail.jsp. Drag a **Link** component from the Palette view and drop it into Update.jsp. Enter /Detail.jsp as its URL and Back as its Label, then click **OK**.



You should see the link as shown below.



8. Save and close Update.jsp.

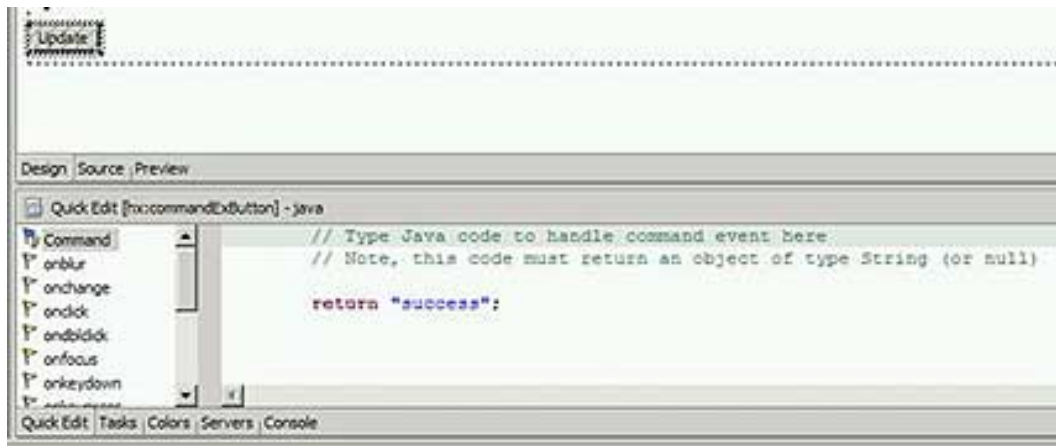
## Modifying the Item Detail portlet to add a navigation button

Now you'll modify the Item Detail portlet to add a navigation button. The user will click this button to jump to the Update page you just created and update a record.

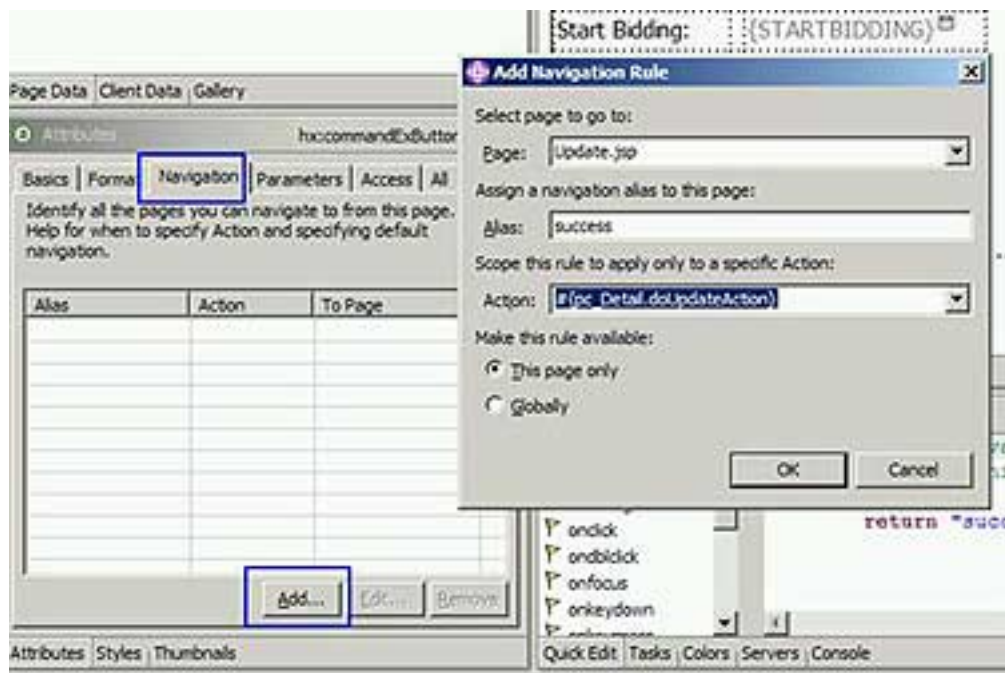
1. Open Detail.jsp by double-clicking on it in the Project Navigator.
2. Drag a **Command - Button** element from the Palette view and drop it into Detail.jsp. In its Attributes view, enter `update` as its ID in the Basic tab, and `Update` as its Label in the Format tab.



3. Click the Update button and select **Command** in the Quick Edit view. Enter `success` as the return statement.



- Click the Navigation tab in the Attributes view for the Update button, then click the Add button. Enter `update.jsp` for Page and `success` for Alias, and choose `#{pc_Detail.doUpdateAction}` from the drop-down list for Action. Then click **OK**.



- Save and close Detail.jsp.

## Testing the Item Detail portlet

Now you're ready to test the page by updating a database record in the WebSphere Portal Test Environment.

1. Right-click **AuctionPortal** in the Project Navigator and select **Run on Server**. Choose an existing server and click **Finish**.
2. When the Item Detail portlet is displayed, enter 1 and click **Submit**. The detail information will be displayed.



3. Click the Update button. You should see the Update page. Enter some new data in the input fields and click the Submit button to update a selected record. You could also delete it by clicking Delete button.
4. Once you've finished working on this page, click **Back** to go back to the Item Detail page.



## Section 7. Summary

In this tutorial, you learned how to perform the following visual development tasks using WebSphere Studio V5.1.2 and Portal Toolkit V5.0.2.2:

- Creating JavaServer Faces portlets
- Using WebSphere Data Object with JSF to access a database
- Enabling Click-to-Action for communication between JSF portlets
- Defining page navigation in a JSF portlet

The visual tools introduced in this tutorial can help you build various types of portlet applications quickly without deep skills in Java programming. The underlying programming models allow you to add advanced application logic (Java code) to JSF portlets built using visual tools. Other tools in WebSphere Studio, such as the Java Editor, team support, and so on can be used seamlessly along with the visual tools.

# Resources

## Learn

- Check out Sun's [JavaServer Faces site](#).
- Learn more at [JSF Central](#).
- Read the IBM Redbook "[WebSphere Studio V5.1.2 JavaServer Faces and Service Data Objects](#)" for a more comprehensive look at building JSF applications with WebSphere Studio 5.1.2 and Service Data Objects.
- There are a number of books out on JavaServer Faces:
  - [Core JavaServer Faces](#) , David M. Geary and Cay S. Horstmann (Prentice Hall, 2004)
  - [JavaServer Faces in Action](#) , Kito D. Mann (Manning Publications Company, 2004)
  - [JavaServer Faces Kick Start](#) , James Turner, Craig McClanahan, and Kunal Mittal (Sams, 2004)
  - [Mastering JavaServer Faces](#), Bill Dudney, Jonathan Lehr, Bill Willis, and Mattingly LeRoy (John Wiley & Sons, 2004)
  - [Pro JavaServer Faces: Building J2EE Applications with JSF](#) , Kim Topley (Springer-Verlag New York, 2004)
  - [JavaServer Faces](#) , Hans Bergsten (O'Reilly & Associates, 2004)
  - [JavaServer Faces Programming](#) , Budi Kurniawan (McGraw-Hill Osborne, 2003)
- Dive in to the [WebSphere Portal product documentation](#).
- Check out the IBM Redbook "[IBM WebSphere Portal V5: A guide for portlet application development](#)."
- Read "[Using cooperative portlets in WebSphere Portal V5](#)," Amber Roy-Chowdhury (*developerWorks*, October 2003)
- Stay current with [developerWorks technical events and Webcasts](#).

## Get products and technologies

- [Download DB2 UDB V8](#)
- Build your next development project with [IBM trial software](#), available for download directly from developerWorks.

## Discuss

- [Participate in the discussion forum for this content.](#)

## About the authors

Masato Noguchi

Masato Noguchi is an advisory software engineer for IBM Rational -- WebSphere Portal Tools.

---

Takeshi Watanabe

Takeshi Watanabe is a product manager for Rational Desktop products at IBM Rational.