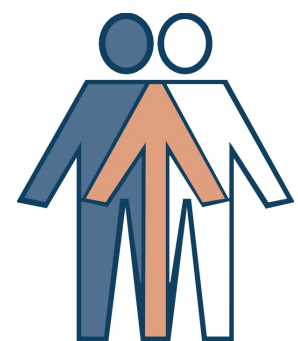


Oracle Forms as Web Component

Step-by-Step How-To Guide

Document:
Version:
Author:
Last updated:

Forms-as-Web-Components-Step-By-Step.docx
1.2
Wilfred van der Deijl
13 June 2007



EUROTRANSPLANT

Change record

This document will probably evolve over time. In its first version it still contained a list of “Future Ideas” at the end of the document. As more people will use this document, revisions will probably appear. Check back regularly at <http://www.oratransplant.nl/oracle-forms-as-web-component/> for an updated version. This document contains the following revisions:

Date	Author	Version	Change reference
14 May 2007	Wilfred van der Deijl	1.0	Creation.
17 May 2007	Wilfred van der Deijl	1.1	Multi-record Form uses current record visual attribute Cleaner URL and web application name Submit button in customers table now navigates to orders Included sample of HTML with < and > Fixed error in path to region definition Removed absolute path to Forms files Add mayscript=true as HTML parameter for Firefox Changes to the Outbound JavaScript API for Firefox Corrected ID to lowercase of case sensitive Firefox JavaScript Fixed error in CommunicatorBean.getProperty() Import java.applet.Applet while developing CommunicatorBean Fixed performance issue of starting initial form twice Make visual integration work with Firefox Handle ON-ERROR and ON-MESSAGE in JavaScript alerts Added comments which texts are available in copy-and-paste.txt
13 Jun 2006	Wilfred van der Deijl	1.2	Update the license conditions, allowing for commercial use with some minor restrictions

Contents

1	INTRODUCTION	1
1.1	License.....	1
1.2	Comments.....	1
2	PREREQUISITES.....	2
3	SETUP DEVELOPER SUITE	3
4	SETUP DATABASE	5
5	CREATE ORDERS FORM	6
6	CREATE J2EE MODEL PROJECT	12
7	CREATE VIEWCONTROLLER PROJECT.....	22
8	INCLUDE THE FORMS APPLET.....	31
9	SETUP INBOUND JAVASCRIPT API	39
10	SETUP OUTBOUND JAVASCRIPT API	48
11	SUSPEND AND REUSE FORMS APPLET	51
12	VISUAL INTEGRATION	60
13	FUTURE IDEAS AND IMPROVEMENTS	64

1 Introduction

This Step-by-Step describes in detail how to setup an ADF Faces web application that integrates an Oracle Form seamlessly. For a full description of the concept, its background and sample files please visit <http://www.oratransplant.nl/oracle-forms-as-web-component/>

This is a very extensive document and it might seem daunting at first. The number of pages is this high because of the number of screenshots. Don't let the size scare you off. Once you get started you will notice you can progress through the pages quite rapidly.

1.1 License

You can use the concept free of charge but some limitations might apply. Please check <http://www.oratransplant.nl/oracle-forms-as-web-component/> for these conditions and to prevent you might be charged for the use of the concept in the future.

1.2 Comments

If you have any comments or questions regarding this How-To guide or the concept itself, please visit at <http://www.oratransplant.nl/oracle-forms-as-web-component/> to leave a comment or contact me directly using the Contact page.

2 Prerequisites

This step-by-step is written with the following prerequisites:

- Oracle Developer Suite 10gR2 (10.1.2.0.2) should be installed. A trial version can be downloaded at <http://www.oracle.com/technology/software/products/ids/>. Optionally patch set 2 (version 10.1.2.2.0) could be applied. This can be downloaded as patch 4960210 from MetaLink at <http://metalink.oracle.com/>. In this step-by-step Developer Suite 10gR2 was installed in an Oracle Home with the name "ds1012_1".
- JDeveloper 10.1.3.2.0 is used in the examples. JDeveloper is free and can be downloaded at <http://www.oracle.com/technology/software/products/jdev/>
- The Java classes and beans that are used in Oracle Forms can just as well be created with older versions of JDeveloper such as version 10.1.2.1.0 which comes with Developer Suite 10gR2. The sample ADF Faces application can only be created with a 10.1.3.x version of JDeveloper since ADF Faces was introduced in JDeveloper 10.1.3.0.
- The samples require an Oracle database. Any version would do but the step-by-step uses the free Express Edition of the Oracle Database version 10.2.0.1. It can be downloaded at <http://www.oracle.com/technology/software/products/database/xe/>
- Sun Java Runtime Environment 1.4.2_14 or later should be installed. It can be downloaded at <http://java.sun.com/j2se/1.4.2/download.html>. Several compatibility issues exist with Oracle Forms and Sun JRE 1.5.x or 1.6.x. So, be sure to use a 1.4.2_x release.

3 Setup Developer Suite

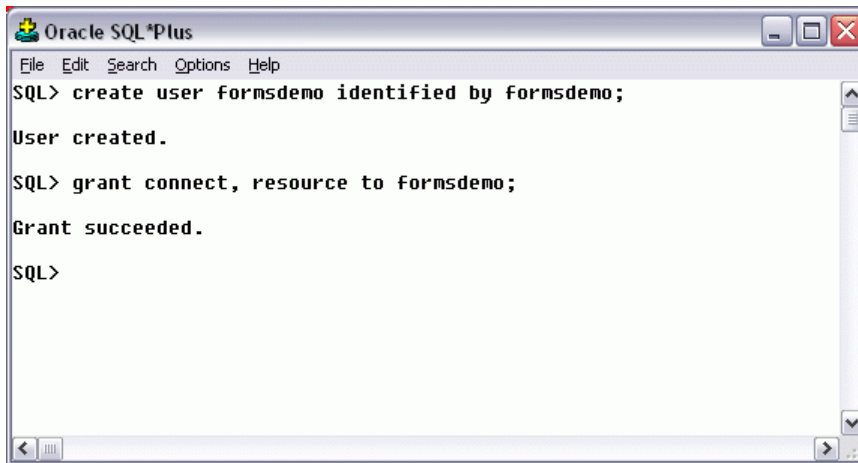
1. Edit the `DEVSUITE_HOME\forms\server\formsweb.cfg` with your favorite text editor. Change the value of the `baseHTMLjinitiator` parameter to `"basejpi.htm"`. This instructs Oracle Forms to use Sun JVM as the client Java Virtual Machine.
2. In the same file change the `jpi_classid` parameter to `"CAFEEFAC-0014-0002-0014-ABCDEFEDCBA"`. This instructs the client to use Sun JVM 1.4.2_14. If you are using a different version of Sun's JVM then change this value accordingly.
3. Change the `jpi_mimetype` to `"application/x-java-applet;jpi-version=1.4.2_14"` to also reflect the use of Sun's JVM 1.4.2_14.
4. Start the registry editor by selecting Start > Run and type `"regedit"`. Then edit `HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_xxxx\FORMS_PATH` (where `KEY_xxxx` represents the name of your Oracle Home). Add `"c:\files"` to the `FORMS_PATH` variable.
5. Start the OC4J (Oracle Containers for Java) that comes with Developer Suite 10g and wait for the initialization to complete. You can start OC4J at Start > All Programs > Oracle Developer Suite – ds1012_1 > Forms Developer > Start OC4J Instance.
6. Add an Oracle TNS Alias to the `TNSNAMES.ORA` file for the Developer Suite Oracle Home. To do this use Start > All Programs > Oracle – ds1012_1 > Configuration and Migration Tools > Net Configuration Assistant or edit the `DEVSUITE_HOME\NETWORK\ADMIN\tnsnames.ora` directly.
7. When using the Net Configuration Assistant, select Local Net Service Name configuration and press "Next". Then Select Add and press "Next". Specify `"XE"` as Service Name and press "Next". Select TCP as network protocol and press "Next". Specify `"localhost"` as the host name, leave the port selection at the standard 1521 and press "Next". Perform a test. This probably fails with an invalid username/password but that is an indication the database connection itself was successful. Press "Next" once more, specify `"XE"` as the Net Service Name and press "Next". Do not create another service and click "Next" until the wizard finishes. Press Finish to close the Net Configuration Assistant.
8. When not using the Net Configuration Assistant edit the `DEVSUITE_HOME\NETWORK\ADMIN\tnsnames.ora` file directly and add:

```
XE =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP) (HOST = localhost) (PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = XE)
    )
  )
```

)

4 Setup database

1. Start the Oracle XE database by selecting Start > All Programs > Oracle Database 10g Express Edition > Start Database
2. Start SQL Plus at Start > All Programs > Oracle – ds1012_1 > Application Development > SQL Plus
3. Login with user SYSTEM, Host String XE and supply the password of the SYSTEM user. You have set this password yourself during the Oracle Database XE installation.
4. Create a new user:



```
Oracle SQL*Plus
File Edit Search Options Help
SQL> create user formsdemo identified by formsdemo;

User created.

SQL> grant connect, resource to formsdemo;

Grant succeeded.

SQL>
```

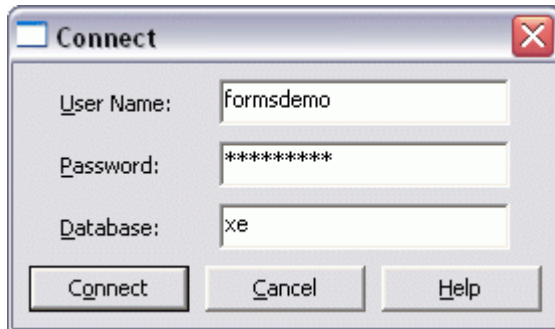
5. Now connect with this new user and run the SQL script shipped with Oracle Developer Suite to install some sample tables. You can safely ignore the last error message about insufficient privileges while creating a view. We're not going to use that view during this step-by-step.



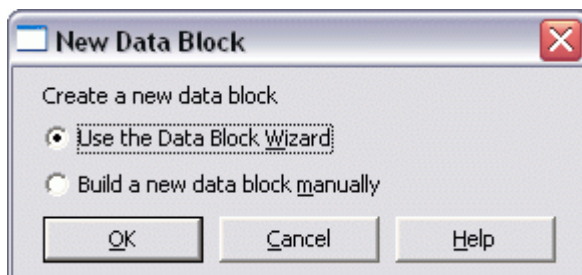
```
Oracle SQL*Plus
File Edit Search Options Help
SQL> connect formsdemo/formsdemo@xe
Connected.
SQL> @C:\oracle\product\10.1.2\ds_1\tools\dbtab\demo1d|
```


5 Create Orders Form

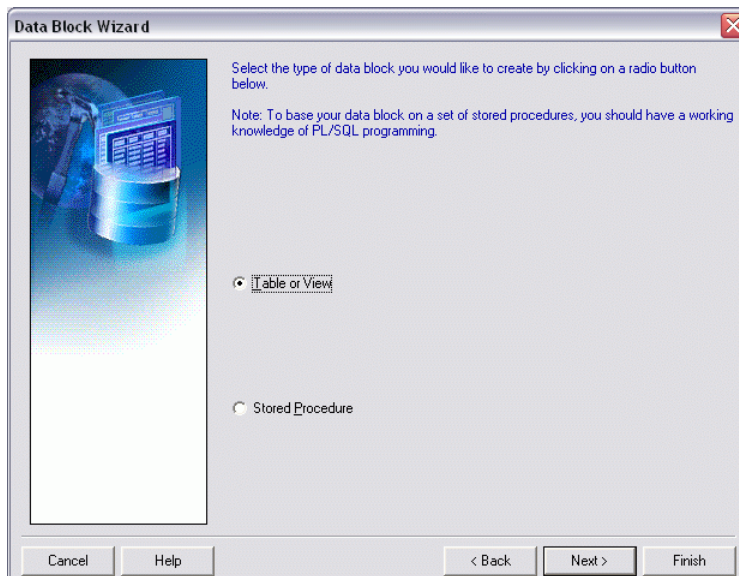
1. Start Forms Builder to create a multi record Form to edit Orders.
2. Choose File > Connect from the menu and connect with username "formsdemo", password "formsdemo" and database "xe":



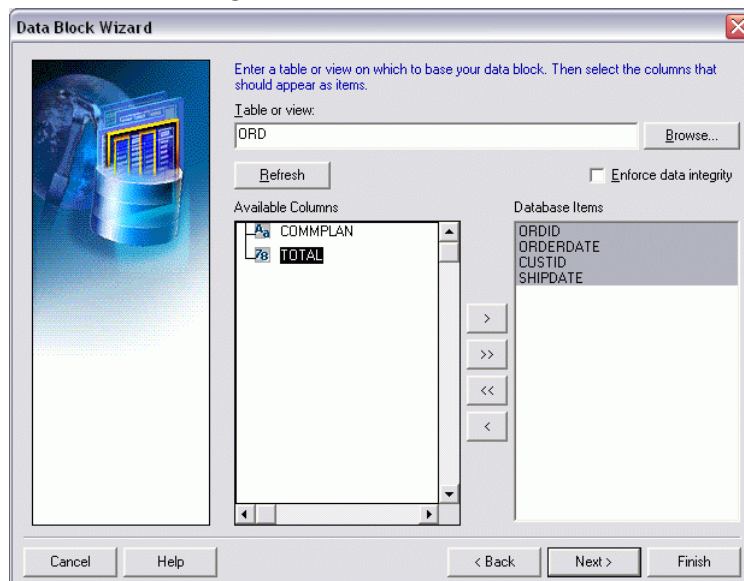
3. Double click the Data Blocks node and select "Use the Data Block Wizard" to create a new block:



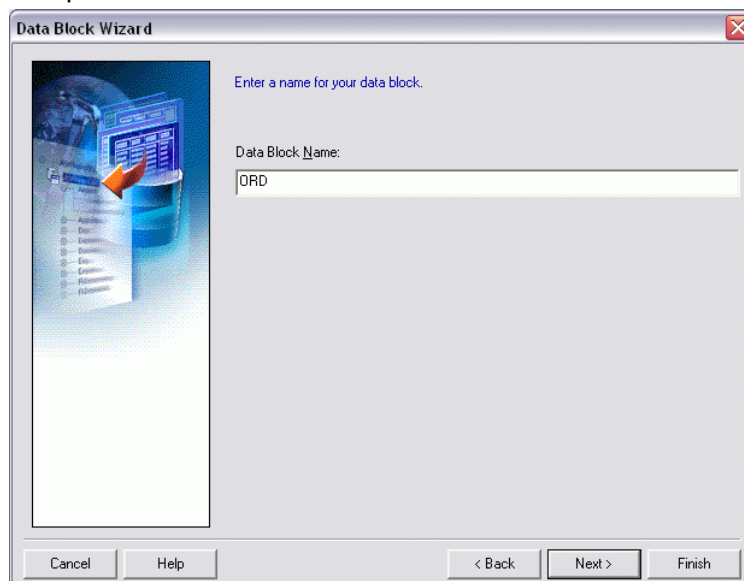
4. Select "Table or View" as the data source:



5. Type "ORD" as the table name and press the Refresh button to retrieve the column names. Select ORDID, ORDERDATE, CUSTID and SHIPDATE. Then press the button with the arrow to the right to select these four columns:



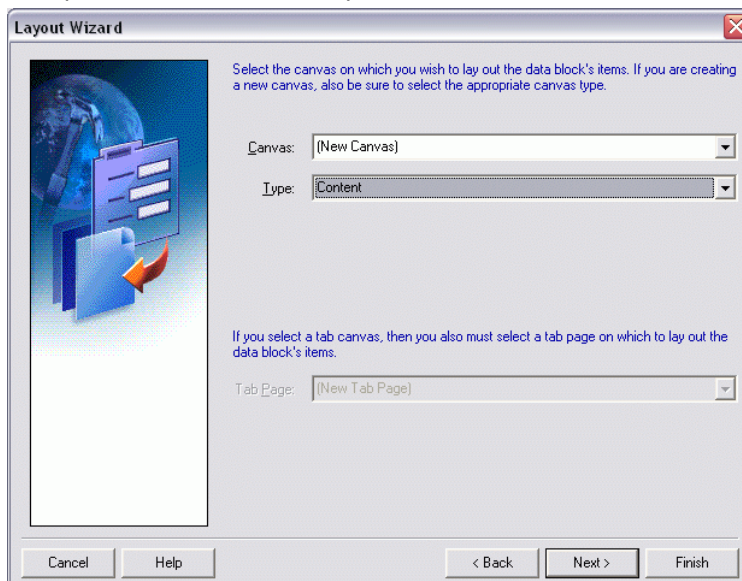
6. Accept the default data block name of ORD:



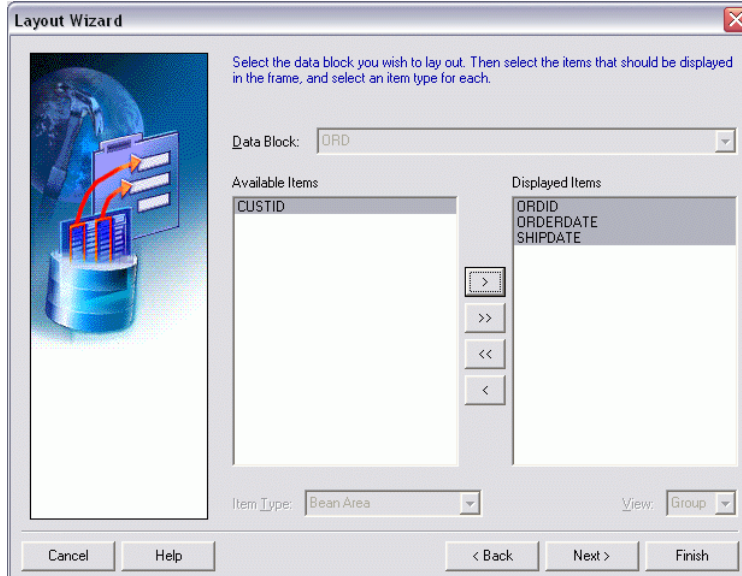
7. Opt to start the Layout Wizard:



8. Accept the defaults of the layout wizard to create a new content canvas:



9. Select the ORDID, ORDERDATE and SHIPDATE columns. Then press the button with the arrow to the right to select these three columns:



Layout Wizard

Select the data block you wish to lay out. Then select the items that should be displayed in the frame, and select an item type for each.

Data Block:

Available Items

CUSTID

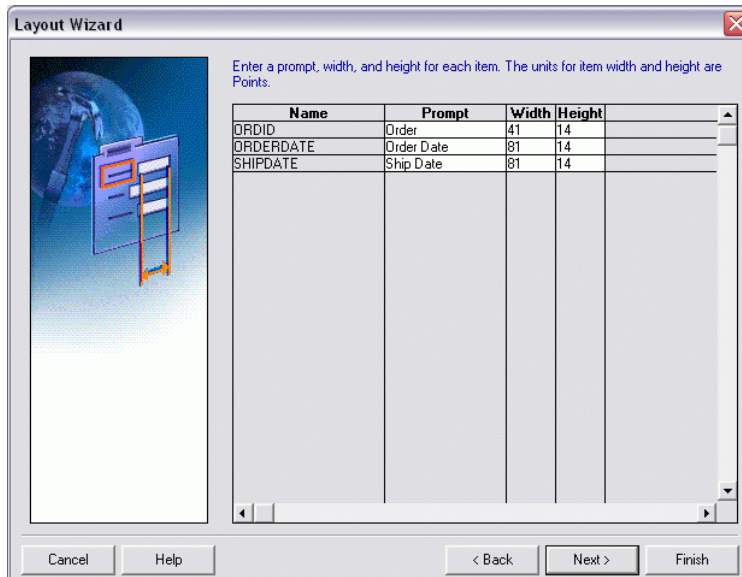
Displayed Items

ORDID
ORDERDATE
SHIPDATE

Item Type: View:

Cancel Help < Back Next > Finish

10. Change the prompts to meaningful values and leave the suggested values for width and height:



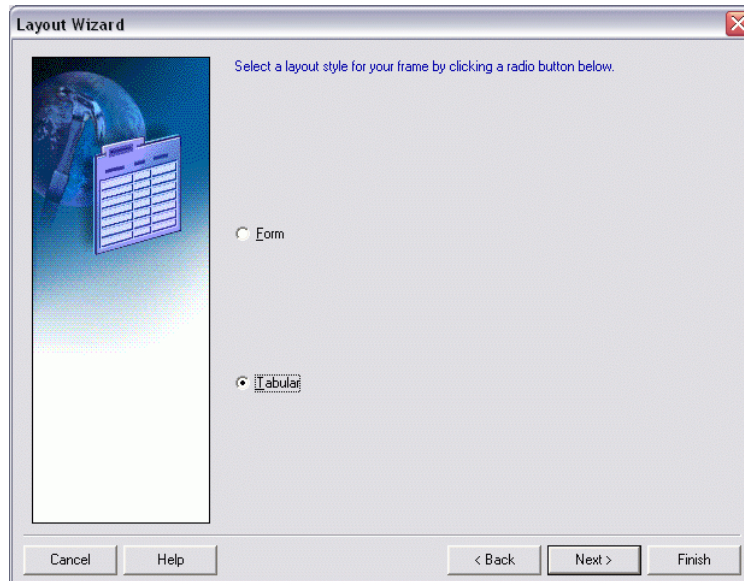
Layout Wizard

Enter a prompt, width, and height for each item. The units for item width and height are Points.

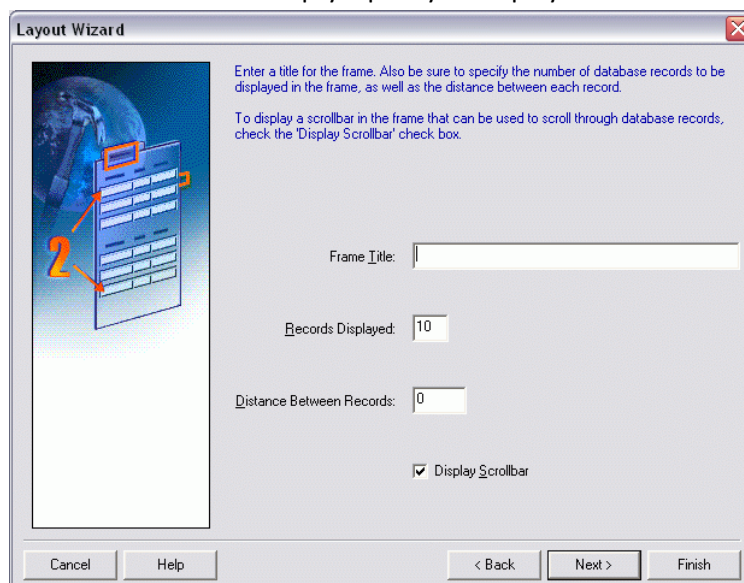
Name	Prompt	Width	Height
ORDID	Order	41	14
ORDERDATE	Order Date	81	14
SHIPDATE	Ship Date	81	14

Cancel Help < Back Next > Finish

11. Select Tabular as the layout style:

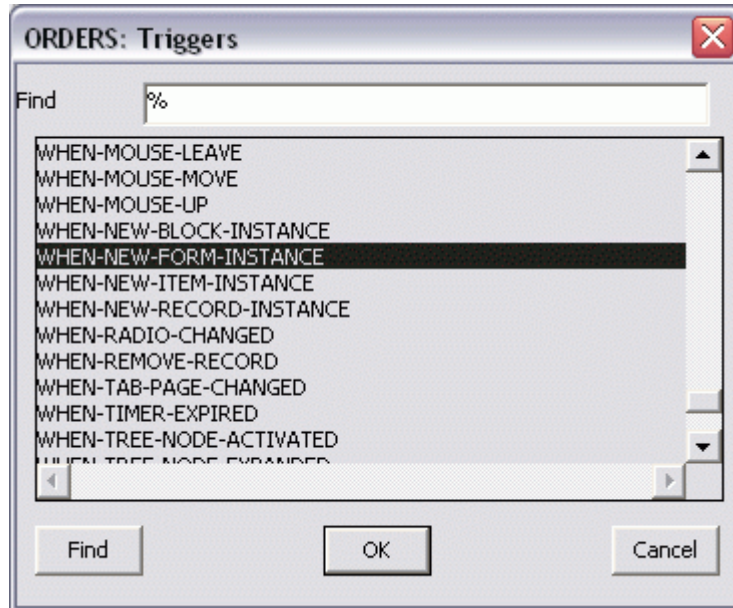


12. Leave the frame title empty. Specify to display 10 records and to display a scroll bar:

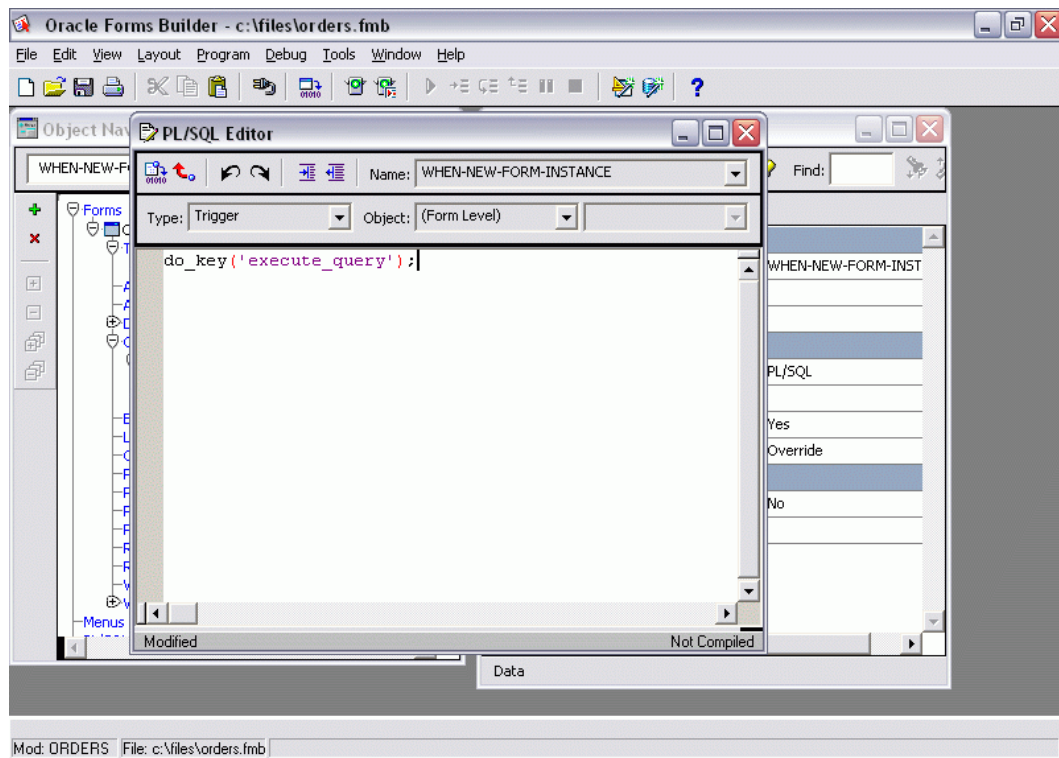


13. Complete the Layout Wizard and save the form by pressing Ctrl-S. Specify "orders" as the name of the form and save it to the directory you also specified in the `FORMS_PATH` during Developer Suite setup. This step-by-step assumes "c:\files"
14. In the Object Navigator, select the Visual Attributes node and press the button with the green plus to create a new visual attribute. Display the property palette if it is not yet displayed and set the Name to "CUR_REC" and the Background Color to "OLAFLight"
15. Go to the properties of the ORD block and set the Current Record Visual Attribute property to "CUR_REC". This ensures that the currently selected record uses a different background color, which clearly indicates to the user which current is selected.

16. In the Object Navigator select the Triggers node and press the button with the green plus to add a Forms level trigger. Select `WHEN-NEW-FORM-INSTANCE` from the list of triggers:



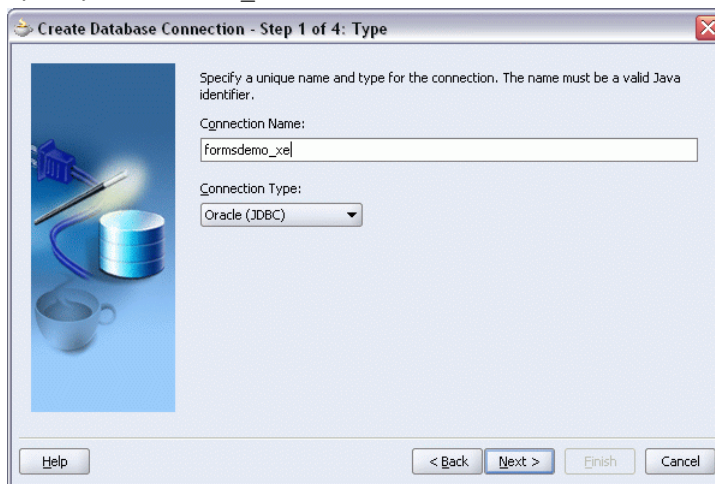
17. Add one line to the trigger to automatically perform an execute-query when the form is started:



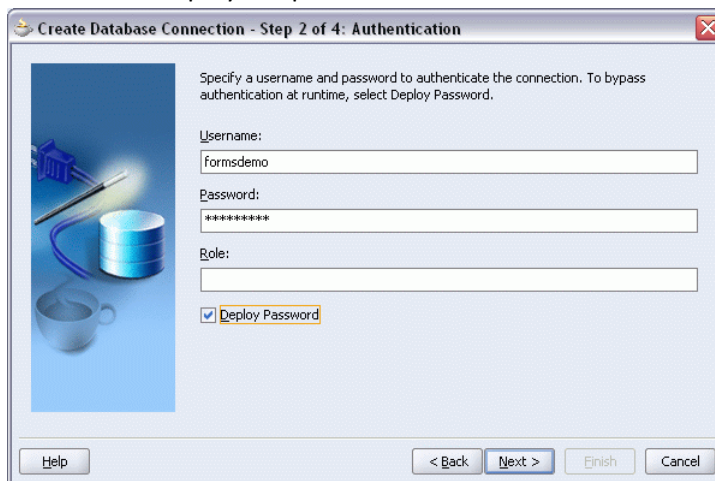
18. Select Program > Run Form or press Ctrl-R to run the Form. Notice how all Orders in the database are queried.

6 Create J2EE Model project

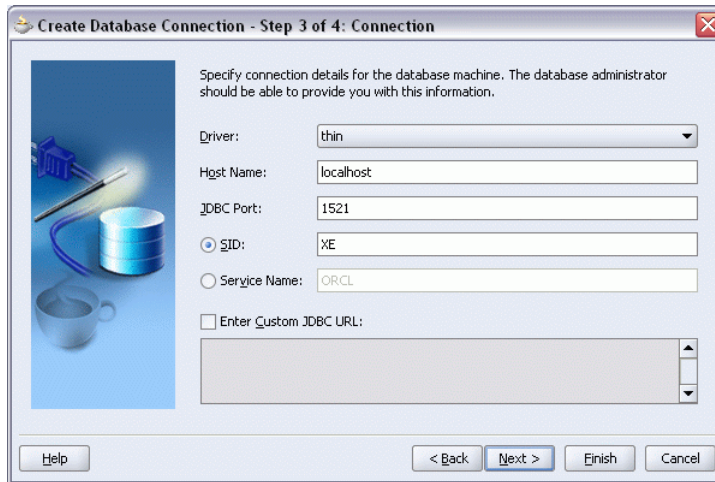
1. Start JDeveloper 10.1.3.2.0
2. Choose View > Connection Navigator from the menu. Right click the Database node and select "New Database Connection"
3. Specify "formsdemo_xe" as connection name:



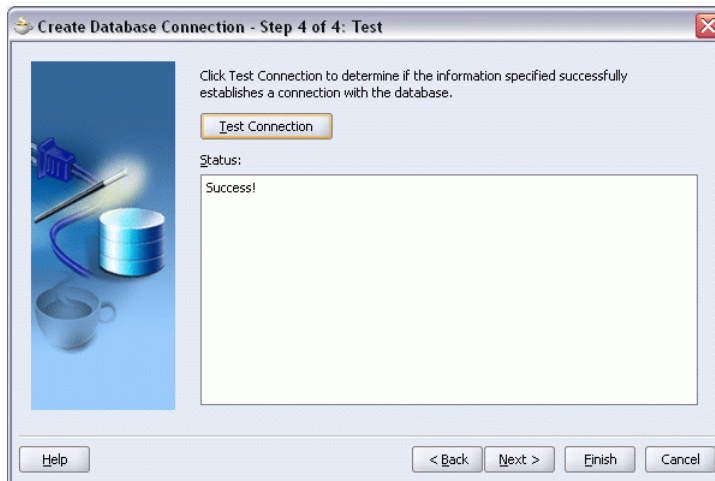
4. Set the username to "formsdemo", the password to "formsdemo" as well and check the checkbox to deploy the password.



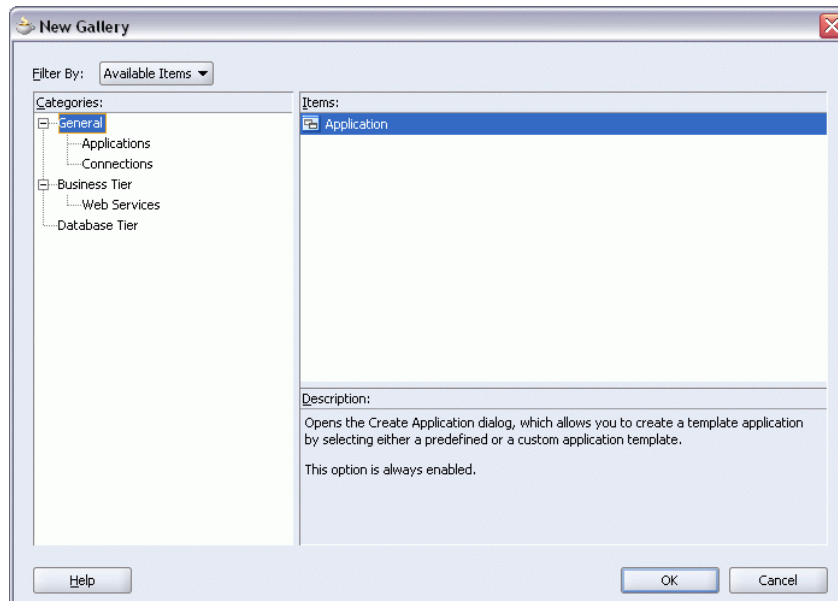
5. Change the default SID of ORCL to XE:



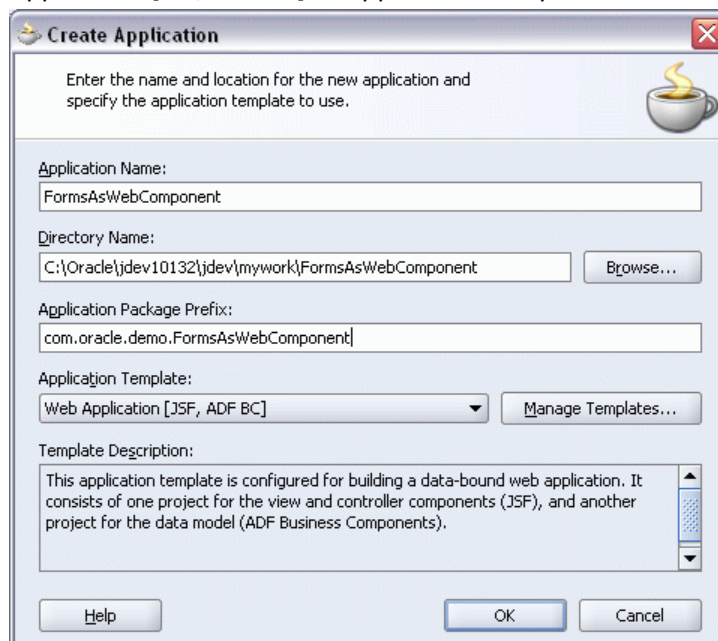
6. On the last page of the wizard press the button to test the connection:



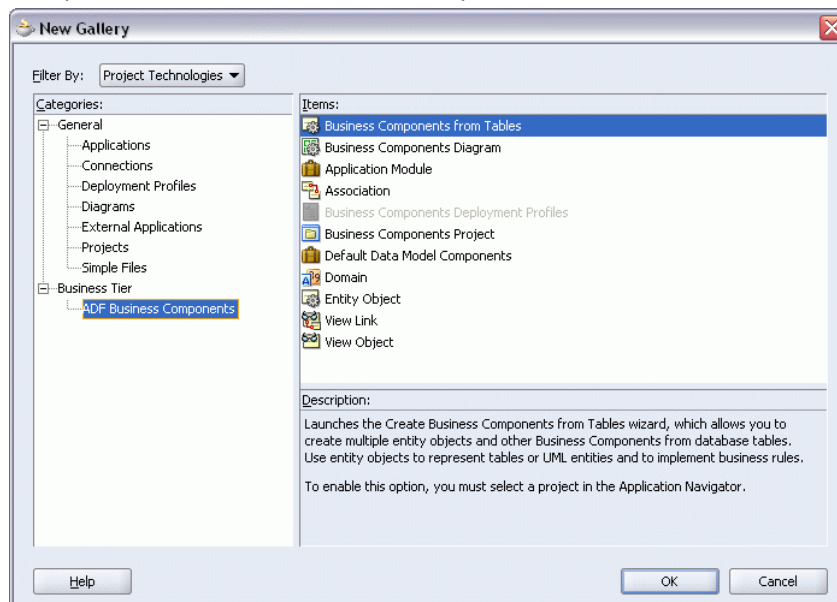
7. Select File > New. In the New Gallery select the General node and then the item "Application":



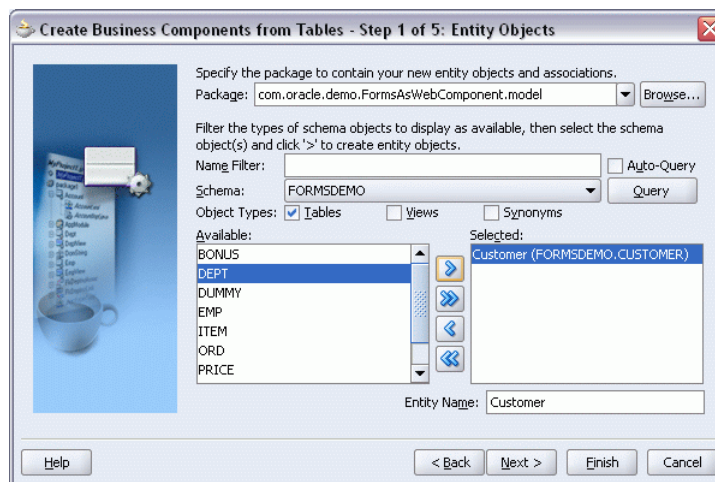
8. Specify "FormsAsWebComponent" as application name, "com.oracle.demo.FormsAsWebComponent" as Application Package Prefix and "Web Application [JSF, ADF BC]" as Application Template:



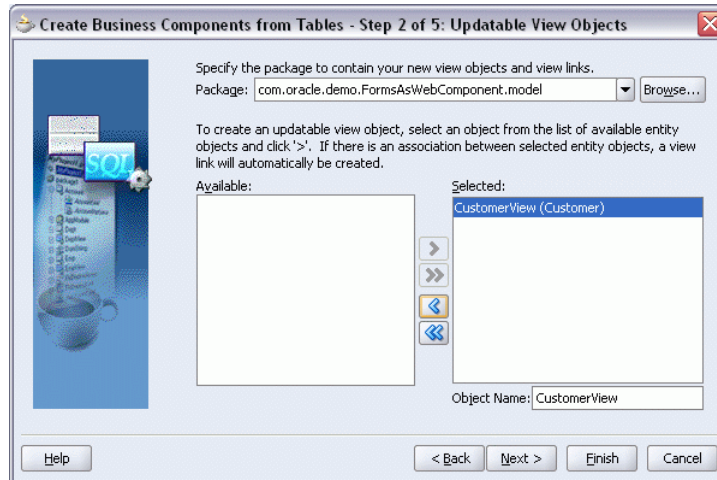
9. A default workspace is created with a Model and a ViewController project. Right click the Model project and select New. In the New Gallery select Business Tier > ADF Business Components and select Business Components from Tables:



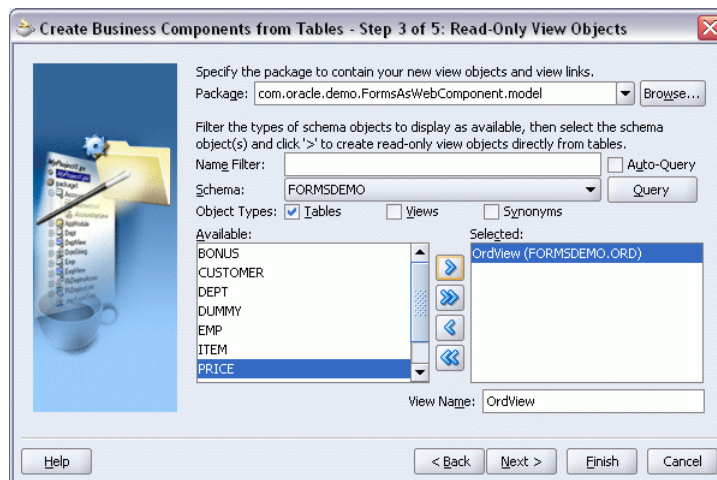
10. Select formsdemo_xe as the connection. Follow the wizard until you reach the page to create Entity Objects. Press the Query button to retrieve a list of tables. Then select the CUSTOMER table from the left hand list and press the button to move it to the right hand list:



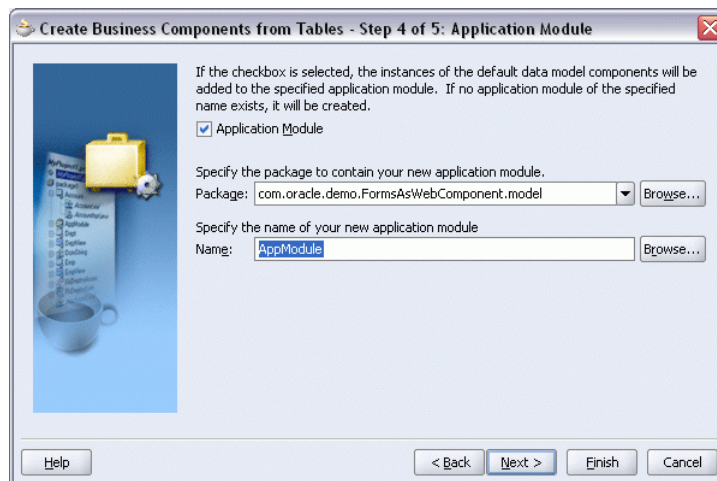
11. On the next page of the wizard move the Customer from the Available (left hand) list to the Selected (right hand) list to create an Updatable ViewObject:



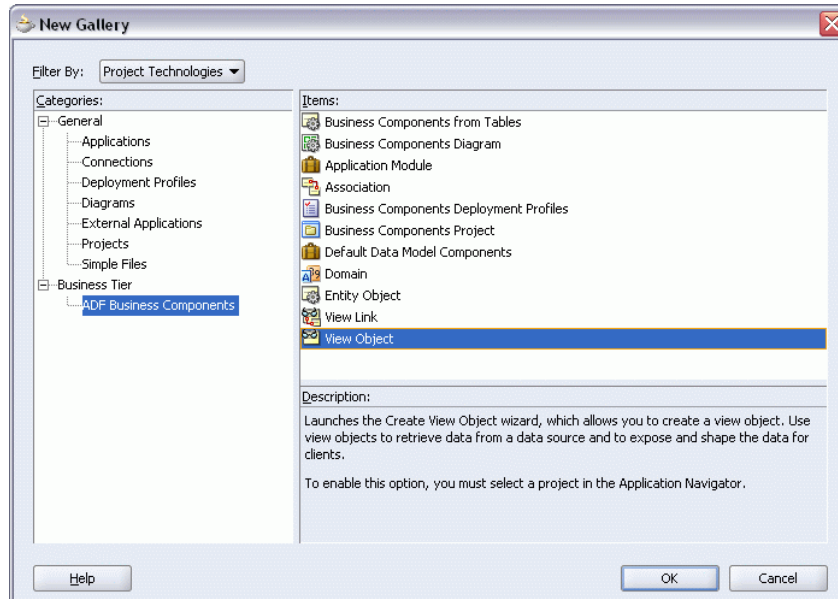
12. The next page of the wizard is to create default read-only view objects. Press the Query button to retrieve the list of database tables. Move the ORD table to the Selected list:



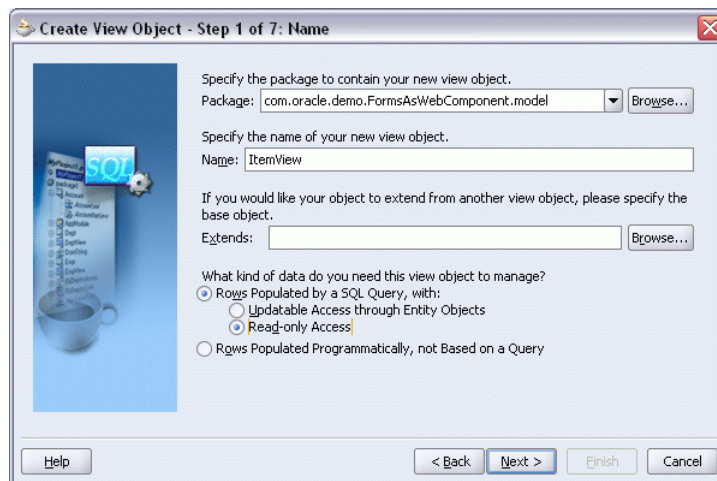
13. Leave all the defaults on the next page of the wizard to create a default Application Module:



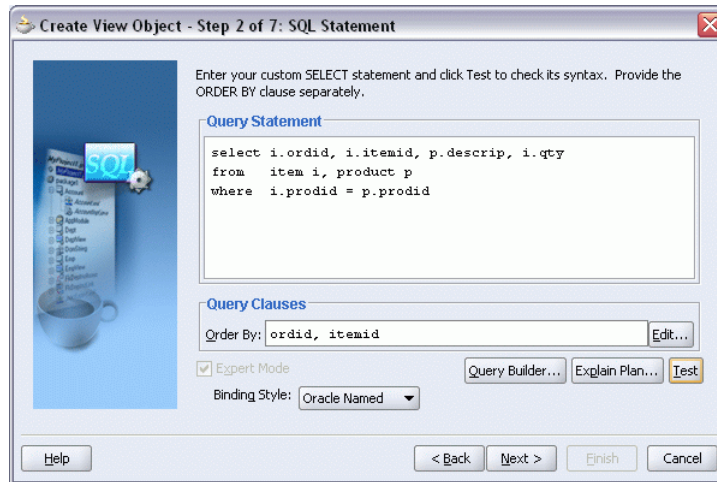
14. Finish the Business Components wizard. Again right click the Model project and select New. From Business Tier > ADF Business Components select View Object:



15. Set the name to "ItemView" and be sure to check "Read-only Access":

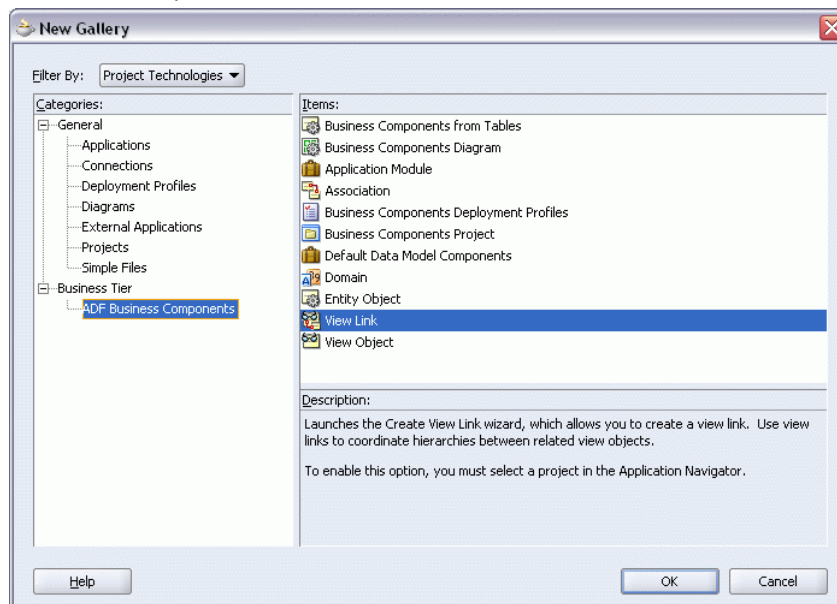


16. Specify the query and order by clause for this View Object:

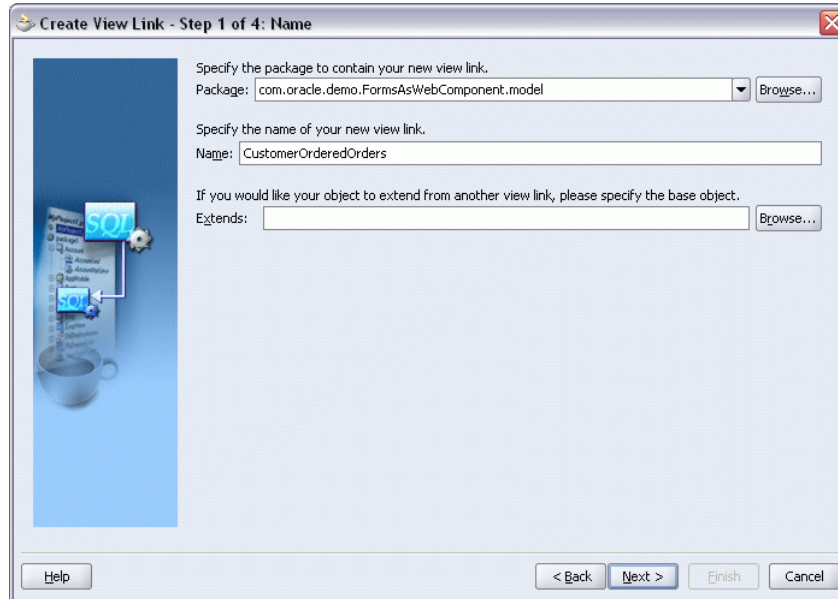


17. Press Finish to complete the View Object wizard.

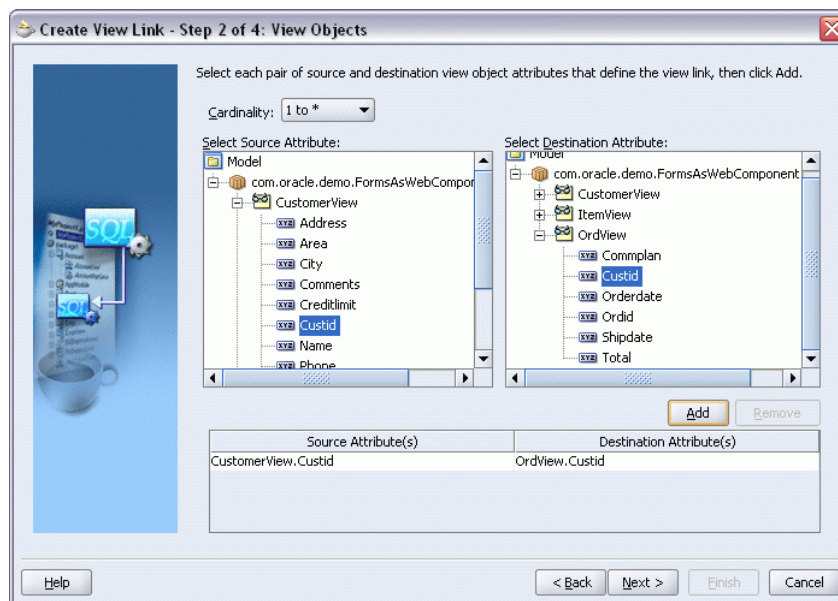
18. Once again right click the Model project and select New. From Business Tier > ADF Business Components select View Link



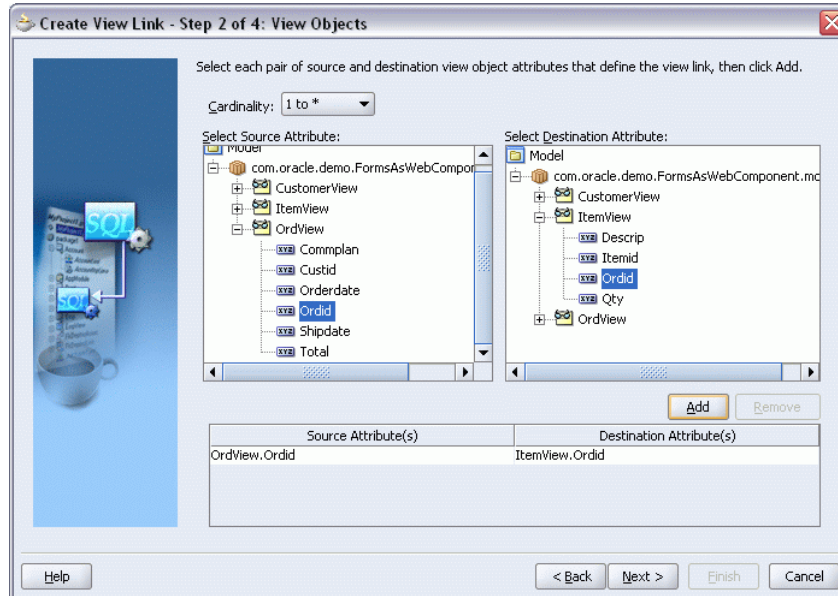
19. Set the View Object name to “CustomerOrderedOrders” and continue the wizard



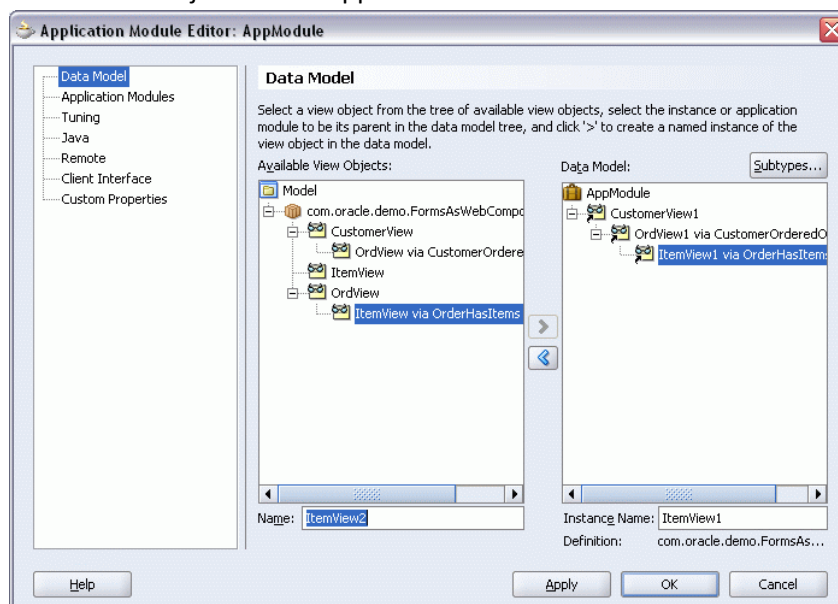
20. Set the Cardinality to “1 to *”. In the Source Attribute tree select CustomerView > Custid. In the Destination Attribute tree select OrdView > Custid. Then press the Add button to add the combination of attributes that define this view link.



21. Finish the rest of the wizard with the defaults to create the View Link. **Then once again right click the Model project and create a new View Link.** This time set the name to "OrderHasItems", set the cardinality to "1 to *" and use OrdView > Ordid and ItemView > Ordid as the attributes for the link



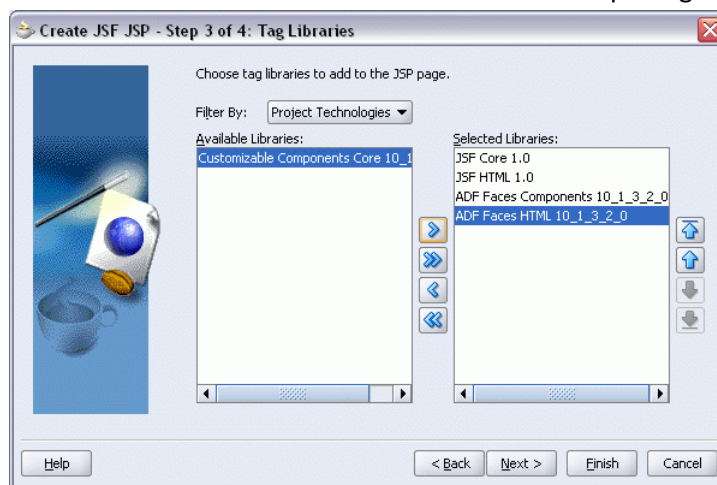
22. In the Application Navigator navigate to Model > Application Sources > com.oracle.demo > FormsAsWebComponent > Model. Right click AppModule and select Edit AppModule. In the Data Model tree select "OrdView1" and press the button with the arrow to the left to remove it from the list. Then select CustomerView > OrdView in the left hand list, select CustomerView1 in the right hand list and press the button with the arrow to the right to add OrdView as a child to CustomerView1. Repeat this action by selecting OrdView > ItemView and add it as child to OrdView1. At the end you should have three nested view objects in the application module.



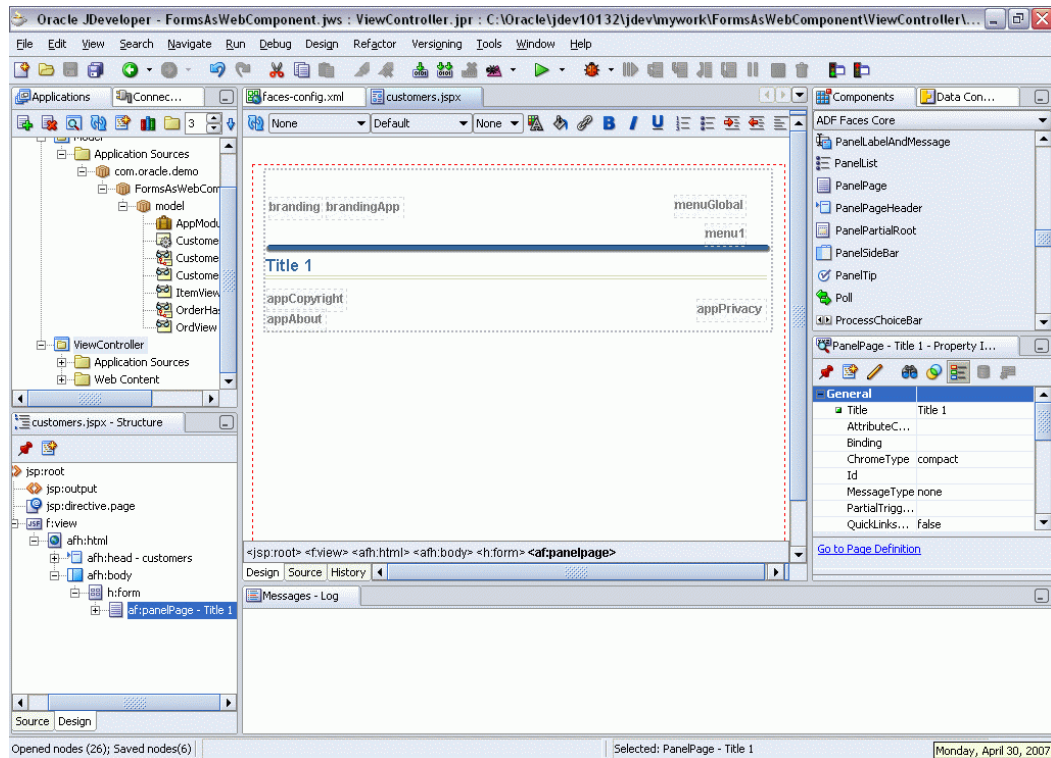
23. This completes the setup of the Model project of your J2EE/ADF application

7 Create ViewController project

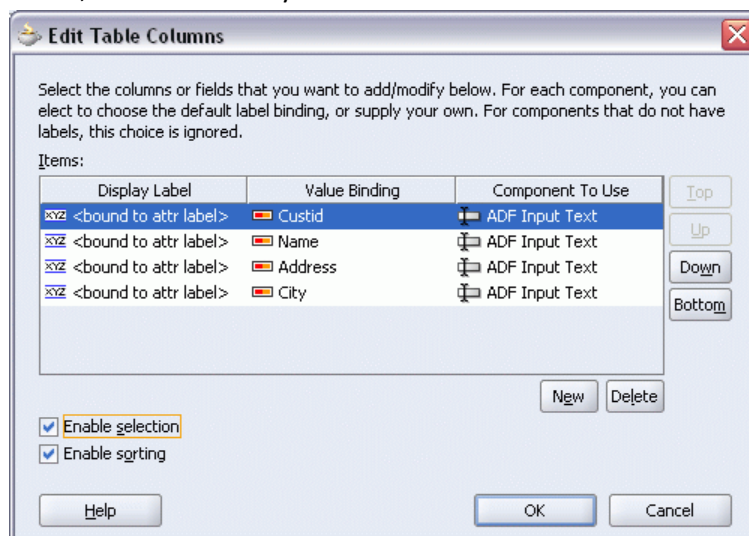
1. In the Applications Navigator of JDeveloper right click the ViewController project and select Project Properties. Go to the J2EE Application node and set the J2EE Web Application Name **and** the J2EE Web Context Root to “FormsAsWebComponent”
2. In the Applications Navigator of JDeveloper right click the ViewController project and select Open JSF Navigation. This opens an empty `faces-config.xml` editor. Drag and drop a JSF Page from the right hand component palette to the editor and change the suggested name of the page to “/customers.jspx”. Then double click the JSF page icon to start the JSF page editor. Complete the wizard until the third page to select the Tag Libraries. Add both ADF Faces libraries before completing the wizard.



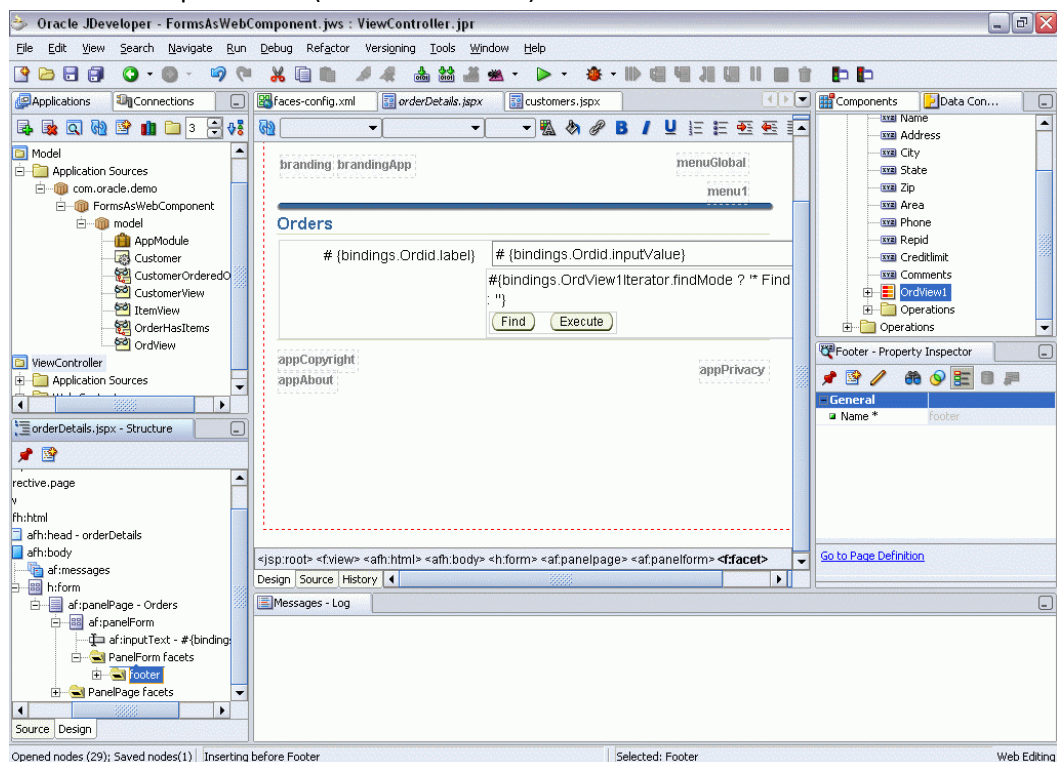
3. The editor for `customers.jspx` starts with a blank page. In the Component Palette select “ADF Faces Core” from the pop list. Then drag and drop PanelPage to the editor



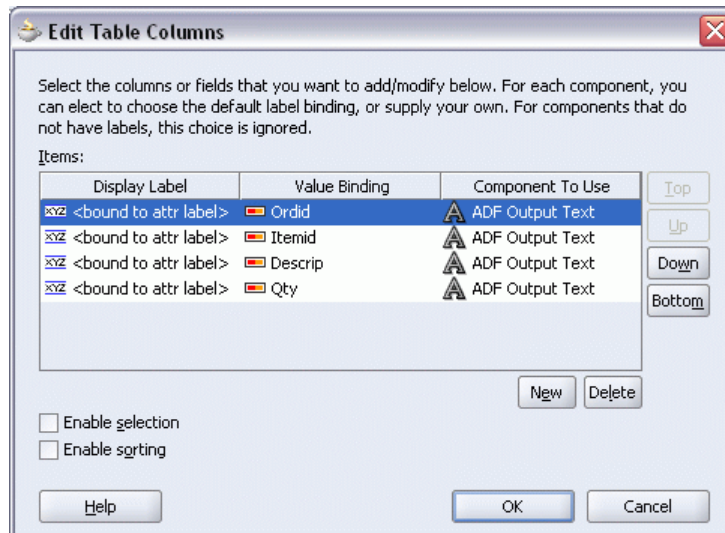
4. In the editor click the “Title 1” title and then change the title to “Customers” in the right hand side Property Palette. Then select the Data Control Palette tab next to the Component Palette or select View > Data Control Palette from the menu. Expand the AppModuleDataControl and drag and drop CustomersView1 to the editor. Select Table > ADF Table as display type. To keep the page simple remove all attributes except Custid, Name, Address and City. Also check the two checkboxes to enable Selection and Sorting.



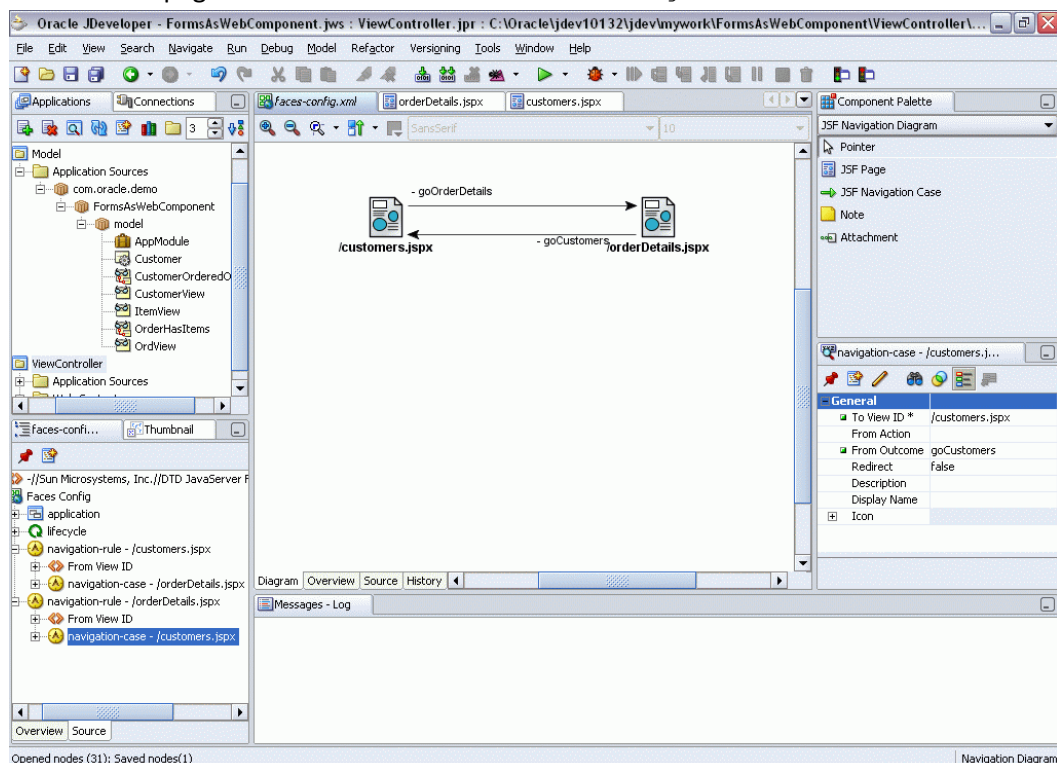
5. Return to the faces-config.xml editor and drag another JSF Page from the component palette. Name it “/orderDetails.jspx”. Double click the new page icon to open that editor. Complete the wizard accepting all defaults. Again drag and drop a PanelPage from the Component Palette to the editor. Change the title “Title 1” to “Orders”.
6. In the Data Control Palette expand the AppModuleDataControl and CustomersView1. Drag and drop the OrdView1 to the orderDetails page and paste it as Forms > ADF Search Form. This creates a search Form which we later hide and fill in from within Oracle Forms to pass back the selected order to the ADF web application. For now expand af:panelForm node in the Structure Pane and remove all inputText and inputDate elements accept the first (used for Order ID).



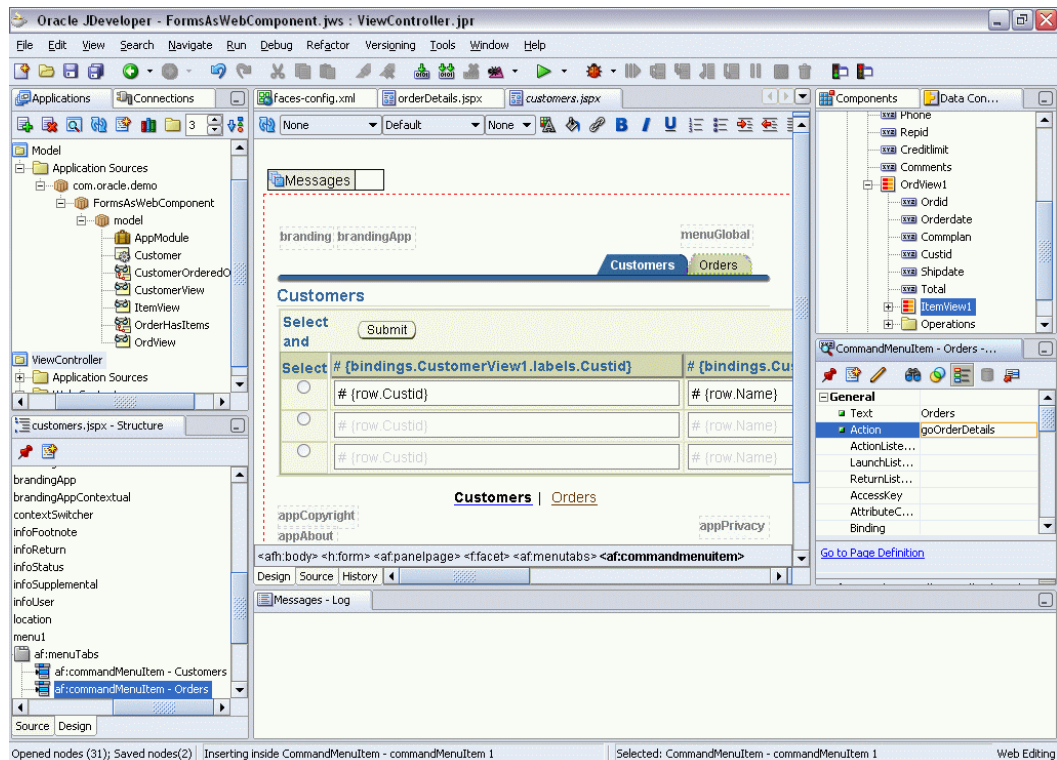
- Expand the OrdView1 in the Data Control Palette and drag and drop the ItemView1 below the search Form in the ADF Faces page. Drop it outside the box surrounding the search form, more between the appCopyright and appPrivacy. Select Tables > ADF Read-only Table as type. Leave the default of displaying all four columns without selection and sorting.



- Go to the faces-config.xml editor. Click JSF Navigation Case from the component palette. Then first click customers.jspx then orderDetails.jspx. Click the arrow that was drawn and go to the Property Palette. Change the From Outcome from "success" to "goOrderDetails". Repeat the process to add a JSF Navigation Case from the orderDetails page back to customers. Name that case "goCustomer".

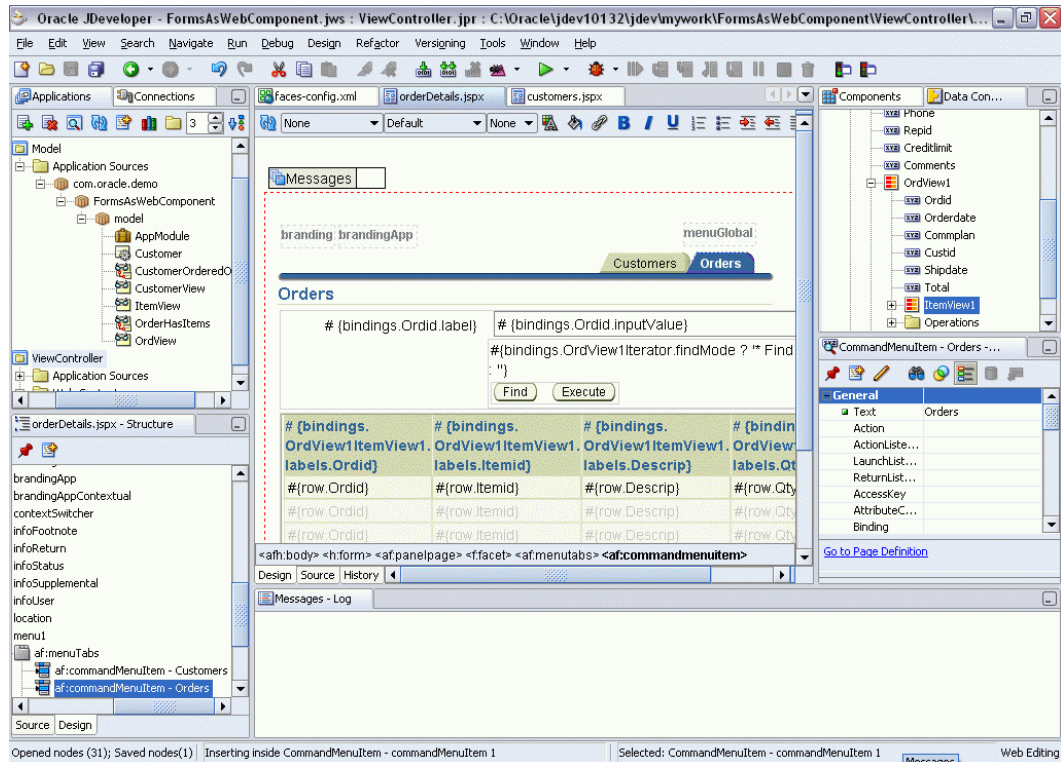


9. Go to the JSF editor for the Customers page, click the menu1 placeholder at the right hand top of the page. Then in the structure pane at the left hand bottom right click the menu1 node and select Insert inside menu1 > MenuTabs. Then right click the new af:menuTabs node and select Insert inside af:menuTabs > CommandMenuItem. In the property palette set the Text to "Customers" and the Selected property to true to render this tab as being selected. Then right click the new af:commandMenuItem node in the Structure Pane and select Insert after af:commandMenuItem > CommandMenuItem. For this commandMenuItem set the Text property to "Orders" and the Action property to "goOrderDetails". Leave the Selected property false.



10. Still in the Customer page, click the submit button in the header of the ADF Faces table. Then go to the Property Inspector and set the Text property to "Show Orders". Set the Action property to "goOrderDetails"

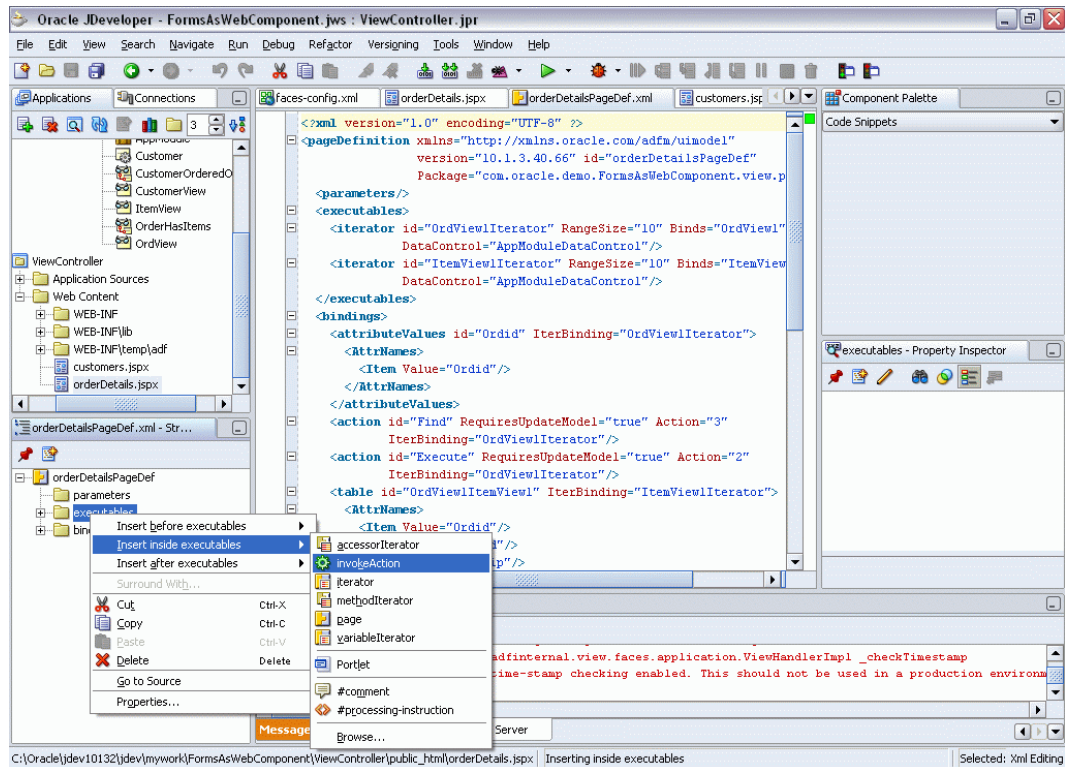
11. Go to the JSF editor for the orderDetails page. Add the same menuTabs and commandMenuItems. This time set the Action of the “Customers” menuitem, leave the Selected property false. For the “Orders” menuitem, leave the Action property empty but set the Selected property to true.



The screenshot shows the Oracle JDeveloper IDE with the JSF editor for the 'orderDetails' page. The main window displays the 'Orders' section with a table of data. The table has columns for OrdId, ItemId, Description, and Quantity. The 'Customers' menu item is selected in the 'af:commandMenuItem' component. The 'af:menuTabs' component is also visible in the design view.

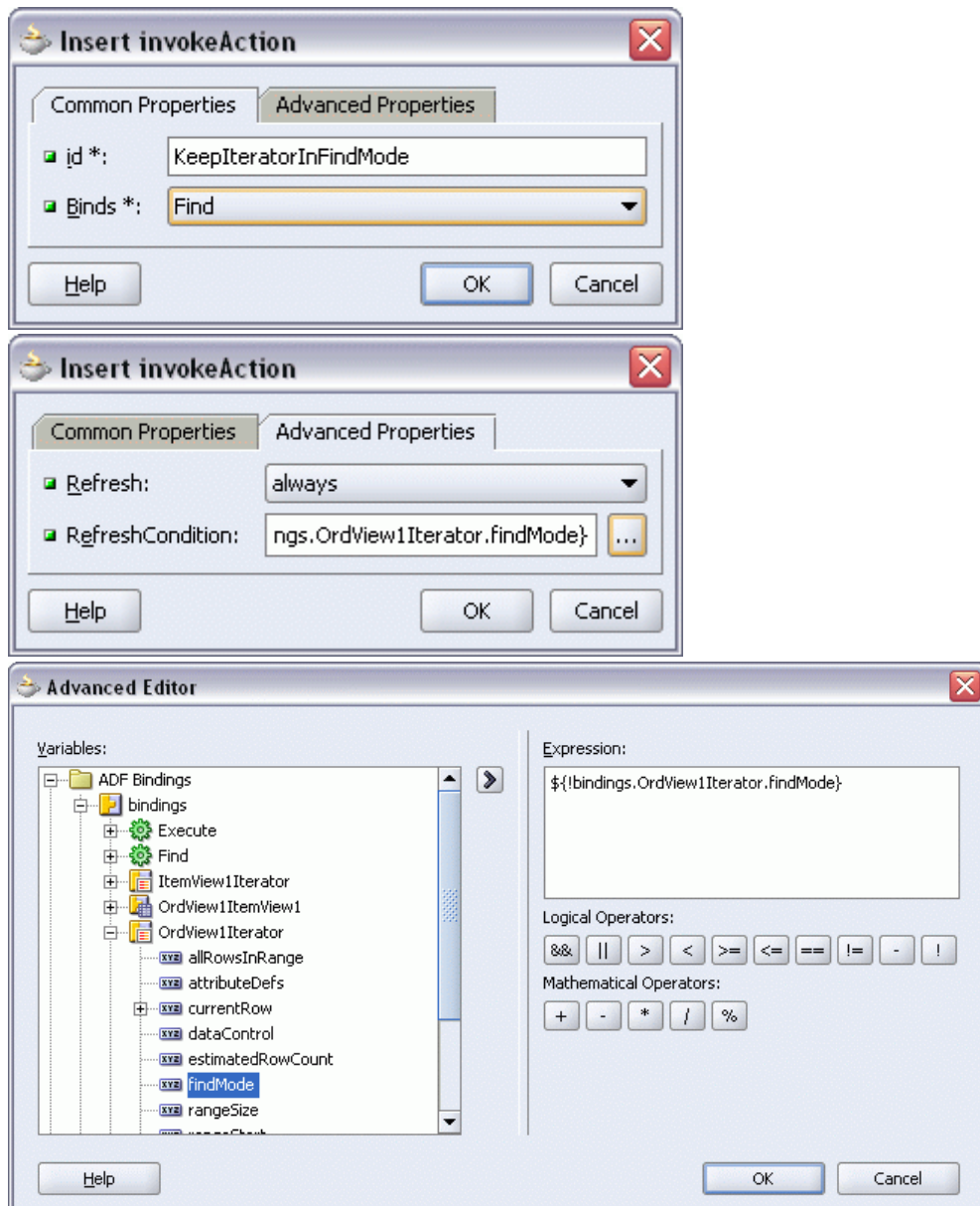
OrdId	ItemId	Description	Quantity
1	1	Item 1	1
2	2	Item 2	2
3	3	Item 3	3
4	4	Item 4	4
5	5	Item 5	5

12. We still need to make sure the Search Form in the orderDetails page stays in Query Mode at all times. For this, go the the JSF editor for the orderDetails page. Right click in the editor and select Go to Page Definition. This opens the editor for the orderDetailsPageDef.xml file. In the structure pane right click the Executables folder and select Insert inside executables > invokeAction.

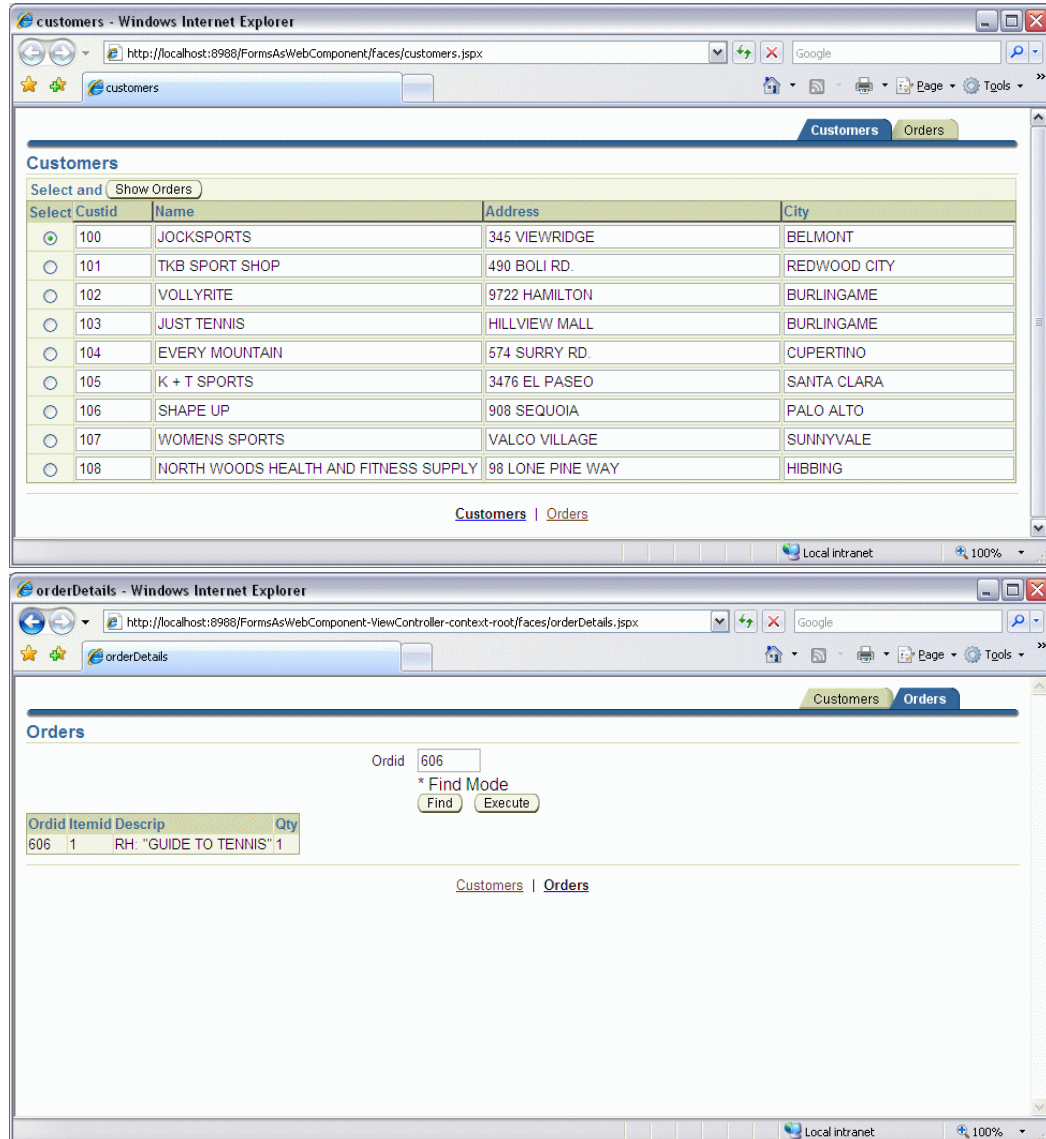


13. In the first tab of the Insert invokeAction dialog, set the ID of the invokeAction to "KeepIteratorInFindMode" and bind it to the Find action. Then go to the second tab of the dialog and set the Refresh property to Always. For the RefreshCondition press the details button to open an editor. Expand the tree to ADF Bindings > bindings > OrdView1Iterator > findMode. Press the arrow button to add it to the EL expression. Then add an exclamation mark before the bindings keyword to negate the expression. You end up with the following expression:

```
${!bindings.OrdView1Iterator.findMode}
```



14. You can now run the application. Go to the faces-config.xml, right click the customers page and select Run. The application will start with a table of customers on the initial page. Click on the Orders tab to go to the orders page. Enter 606 as the Order ID and press the Execute button to retrieve the details of this order. You can also try to query order 609, 620 and 621 for this first customer.



The screenshot shows two browser windows. The top window, titled 'customers - Windows Internet Explorer', displays the 'Customers' page. It features a table with columns: Select, Custid, Name, Address, and City. The table lists 8 customers. Below the table are links for 'Customers' and 'Orders'. The bottom window, titled 'orderDetails - Windows Internet Explorer', displays the 'Orders' page. It has a text input for 'Ordid' with the value '606'. Below the input are buttons for 'Find Mode', 'Find', and 'Execute'. A table below shows the order details for Ordid 606.

Select	Custid	Name	Address	City
<input checked="" type="radio"/>	100	JOCKSPORTS	345 VIEWRIDGE	BELMONT
<input type="radio"/>	101	TKB SPORT SHOP	490 BOLI RD.	REDWOOD CITY
<input type="radio"/>	102	VOLLYRITE	9722 HAMILTON	BURLINGAME
<input type="radio"/>	103	JUST TENNIS	HILLVIEW MALL	BURLINGAME
<input type="radio"/>	104	EVERY MOUNTAIN	574 SURRY RD.	CUPERTINO
<input type="radio"/>	105	K + T SPORTS	3476 EL PASEO	SANTA CLARA
<input type="radio"/>	106	SHAPE UP	908 SEQUOIA	PALO ALTO
<input type="radio"/>	107	WOMENS SPORTS	VALCO VILLAGE	SUNNYVALE
<input type="radio"/>	108	NORTH WOODS HEALTH AND FITNESS SUPPLY	98 LONE PINE WAY	HIBBING

Ordid	Itemid	Descrip	Qty
606	1	RH. "GUIDE TO TENNIS"	1

8 Include the Forms applet

1. We're now going to include the Oracle Forms applet in the JSF page. For this you need the HTML to activate the applet. You could run the orders Form and use the HTML presented to the browser as starting point. This will be described in the next few steps. Alternatively you can skip these steps and copy and paste the HTML from this paper. Run the orders Form from within Forms Builder. Once the Form is displayed, right click in a blank area of the web page and select View Source...

2. The HTML source code of the page will be opened in a text editor. Copy all HTML between the `<body>` and `</body>` tags to a new file. Next you will have to make some changes to this HTML. Since the HTML will no longer be served from the Forms server, but from the JSF server, you need to change all the relative URLs to absolute URLs. Look at the address bar in your web browser running Oracle Forms to get the base URL of the Forms server. These samples use `http://formscomp:8889/` as the base URL. Then make the following changes to the HTML code in your editor:

- Change `<SCRIPT LANGUAGE="JavaScript" SRC="java/forms_ie.js">` to `<SCRIPT LANGUAGE="JavaScript" SRC="http://formscomp:8889/forms/java/forms_ie.js">` so that the SRC attribute has the absolute URL to the JavaScript file
- Change `<PARAM NAME="CODEBASE" VALUE="/forms/java">` to `<PARAM NAME="CODEBASE" VALUE="http://formscomp:8889/forms/java">`
- Change `<PARAM NAME="serverURL" VALUE="/forms/lervlet;jsessionid=****">` to `<PARAM NAME="serverURL" VALUE="http://formscomp:8889/forms/lervlet">` using an absolute URL and removing the `jsessionid` since that session ID is dynamic
- For the `<EMBED>` tag change the `java_codebase` attribute from `"/forms/java"` to `"http://formscomp:8889/forms/java"`
- For the `<EMBED>` tag change the `serverURL` attribute from `"/forms/lervlet;jsessionid=c0a84f8422b9aa65d9e2a3c746edbd69b70a1fde6261"` to `"http://formscomp:8889/forms/lervlet"` using an absolute URL and removing the `jsessionid` since that session ID is dynamic
- Change `"c:\files\orders.fmx"` to `"orders.fmx"` **twice** since we added `"c:\files"` to the `FORMS_PATH` in the setup phase.

3. You should end up with something like:

```
<COMMENT id="forms_plugin_info"
  plug_ver="clsid:CAFEEFAC-0014-0002-0014-ABCDEFEDCBA"
  appheight="600"
  appwidth="750"
```

```

    appcodebase="http://java.sun.com/products/plugin/autodl/jinstall-
1_4_2-windows-i586.cab#Version=1,4,2,06">
</COMMENT>
<!-- Forms applet definition (start) -->
<NOSCRIPT>
<OBJECT classid="clsid:CAFEEFAC-0014-0002-0014-ABCDEFFEDCBA"
        codebase="http://java.sun.com/products/plugin/autodl/jinstall-
1_4_2-windows-i586.cab#Version=1,4,2,06"
        WIDTH="750"
        HEIGHT="600"
        HSPACE="0"
        VSPACE="0">
</NOSCRIPT>
<SCRIPT LANGUAGE="JavaScript"
SRC="http://formscomp:8889/forms/java/forms_ie.js"></SCRIPT>
<PARAM NAME="TYPE"          VALUE="application/x-java-applet;jpi-
version=1.4.2_14">
<PARAM NAME="CODEBASE"     VALUE="http://formscomp:8889/forms/java">
<PARAM NAME="CODE"         VALUE="oracle.forms.engine.Main" >
<PARAM NAME="ARCHIVE"      VALUE="frmall.jar" >
<PARAM NAME="serverURL"    VALUE="http://formscomp:8889/forms/lervlet">
<PARAM NAME="networkRetries" VALUE="0">
<PARAM NAME="serverArgs"
        VALUE="escapeParams=true module=orders.fmx userid= sso_userid=%20
sso_formsid=%25OID_FORMSID%25 sso_subDN= sso_usrDN= debug=no host= port=
buffer_records=no debug_messages=no array=YES obr=YES query_only=no
quiet=yes render=no record= tracegroup= log= term=">
<PARAM NAME="separateFrame" VALUE="false">
<PARAM NAME="splashScreen"  VALUE="">
<PARAM NAME="background"    VALUE="">
<PARAM NAME="lookAndFeel"   VALUE="Oracle">
<PARAM NAME="colorScheme"   VALUE="teal">
<PARAM NAME="serverApp"     VALUE="default">
<PARAM NAME="logo"          VALUE="">
<PARAM NAME="imageBase"     VALUE="DocumentBase">
<PARAM NAME="formsMessageListener" VALUE="">
<PARAM NAME="recordFileName" VALUE="">
<PARAM NAME="EndUserMonitoringEnabled" VALUE="">
<PARAM NAME="EndUserMonitoringURL" VALUE="">
<PARAM NAME="heartBeat"     VALUE="">
<PARAM NAME="allowAlertClipboard" VALUE="">
<PARAM NAME="disableValidateClipboard" VALUE="">
<COMMENT>
<EMBED SRC=""
PLUGINSPPAGE="http://java.sun.com/products/archive/j2se/1.4.2_06/index.html
"
        TYPE="application/x-java-applet;jpi-version=1.4.2_14"
        java_codebase="http://formscomp:8889/forms/java"
        java_code="oracle.forms.engine.Main"
        java_archive="frmall.jar"
        WIDTH="750"
        HEIGHT="600"
        HSPACE="0"
        VSPACE="0"
        serverURL="http://formscomp:8889/forms/lervlet"
        networkRetries="0"

```

```

serverArgs="escapeParams=true module=orders.fmx userid=
sso_userid=%20 sso_formsid=%25OID_FORMSID%25 sso_subDN= sso_usrDN=
debug=no host= port= buffer_records=no debug_messages=no array=YES obr=YES
query_only=no quiet=yes render=no record= tracegroup= log= term="
    separateFrame="false"
    splashScreen=""
    background=""
    lookAndFeel="Oracle"
    colorScheme="teal"
    serverApp="default"
    logo=""
    imageBase="DocumentBase"
    recordFileName=""
    EndUserMonitoringEnabled=""
    EndUserMonitoringURL=""
    heartBeat=""
    disableValidateClipboard=""
>
<NOEMBED>
</COMMENT>
</NOEMBED></EMBED>
</OBJECT>
<!-- Forms applet definition (end) -->

```

4. Use you text editor to replace all "<" characters with "<" and all ">" characters with ">". [If you're using the vi editor: :%s/</\</g] This should result in something like:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```

&lt;!-- Forms applet definition (start) --&gt;
&lt;NOSCRIPT&gt;
&lt;OBJECT classid="clsid:CAFEEFAC-0014-0002-0014-ABCDEFFEDCBA"
    codebase="http://java.sun.com/products/plugin/autodl/jinstall-
1_4_2-windows-i586.cab#Version=1,4,2,06"
    WIDTH="750"
    HEIGHT="600"
    HSPACE="0"
    VSPACE="0"&gt;
&lt;/NOSCRIPT&gt;
&lt;SCRIPT LANGUAGE="JavaScript"
SRC="http://formscomp:8889/forms/java/forms_ie.js"&gt;&lt;/SCRIPT&gt;
&lt;PARAM NAME="TYPE"        VALUE="application/x-java-applet;jpi-
version=1.4.2_14"&gt;
&lt;PARAM NAME="CODEBASE"    VALUE="http://formscomp:8889/forms/java"&gt;
&lt;PARAM NAME="CODE"        VALUE="oracle.forms.engine.Main" &gt;
&lt;PARAM NAME="ARCHIVE"     VALUE="frmall.jar" &gt;
&lt;PARAM NAME="serverURL"
VALUE="http://formscomp:8889/forms/lervlet"&gt;
&lt;PARAM NAME="networkRetries" VALUE="0"&gt;
&lt;PARAM NAME="serverArgs"
    VALUE="escapeParams=true module=orders.fmx userid= sso_userid=%20
sso_formsid=%25OID_FORMSID%25 sso_subDN= sso_usrDN= debug=no host= port=

```

```

buffer_records=no debug_messages=no array=YES obr=YES query_only=no
quiet=yes render=no record= tracegroup= log= term="&gt;
<lt;PARAM NAME="separateFrame" VALUE="false"&gt;
<lt;PARAM NAME="splashScreen" VALUE=""&gt;
<lt;PARAM NAME="background" VALUE=""&gt;
<lt;PARAM NAME="lookAndFeel" VALUE="Oracle"&gt;
<lt;PARAM NAME="colorScheme" VALUE="teal"&gt;
<lt;PARAM NAME="serverApp" VALUE="default"&gt;
<lt;PARAM NAME="logo" VALUE=""&gt;
<lt;PARAM NAME="imageBase" VALUE="DocumentBase"&gt;
<lt;PARAM NAME="formsMessageListener" VALUE=""&gt;
<lt;PARAM NAME="recordFileName" VALUE=""&gt;
<lt;PARAM NAME="EndUserMonitoringEnabled" VALUE=""&gt;
<lt;PARAM NAME="EndUserMonitoringURL" VALUE=""&gt;
<lt;PARAM NAME="heartBeat" VALUE=""&gt;
<lt;PARAM NAME="allowAlertClipboard" VALUE=""&gt;
<lt;PARAM NAME="disableValidateClipboard" VALUE=""&gt;
<lt;COMMENT&gt;
<lt;EMBED SRC=""
PLUGINS PAGE="http://java.sun.com/products/archive/j2se/1.4.2_06/index.html
"
    TYPE="application/x-java-applet;jpi-version=1.4.2_14"
    java_codebase="http://formscomp:8889/forms/java"
    java_code="oracle.forms.engine.Main"
    java_archive="frmall.jar"
    WIDTH="750"
    HEIGHT="600"
    HSPACE="0"
    VSPACE="0"
    serverURL="http://formscomp:8889/forms/lervlet"
    networkRetries="0"
    serverArgs="escapeParams=true module=orders.fmx userid=
sso_userid=%20 sso_formsid=%25OID_FORMSID%25 sso_subDN= sso_usrDN=
debug=no host= port= buffer_records=no debug_messages=no array=YES obr=YES
query_only=no quiet=yes render=no record= tracegroup= log= term="
    separateFrame="false"
    splashScreen=""
    background=""
    lookAndFeel="Oracle"
    colorScheme="teal"
    serverApp="default"
    logo=""
    imageBase="DocumentBase"
    recordFileName=""
    EndUserMonitoringEnabled=""
    EndUserMonitoringURL=""
    heartBeat=""
    disableValidateClipboard=""
&gt;
<lt;NOEMBED&gt;
<lt;/COMMENT&gt;
<lt;/NOEMBED&gt;<lt;/EMBED&gt;
<lt;/OBJECT&gt;
<lt;!-- Forms applet definition (end) --&gt;

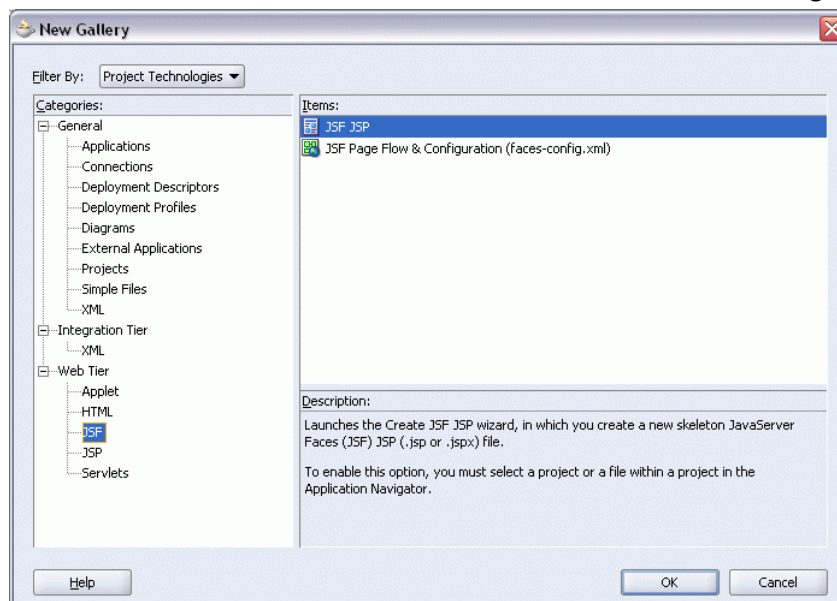
```

5. If you use a Forms version prior to 10.1.2.2.0 the forms_ie.js JavaScript file used in the above HTML is not included in your installation. In that case, create file called `DEVSUITE_HOME\forms\java\forms_ie.js` with the following content:

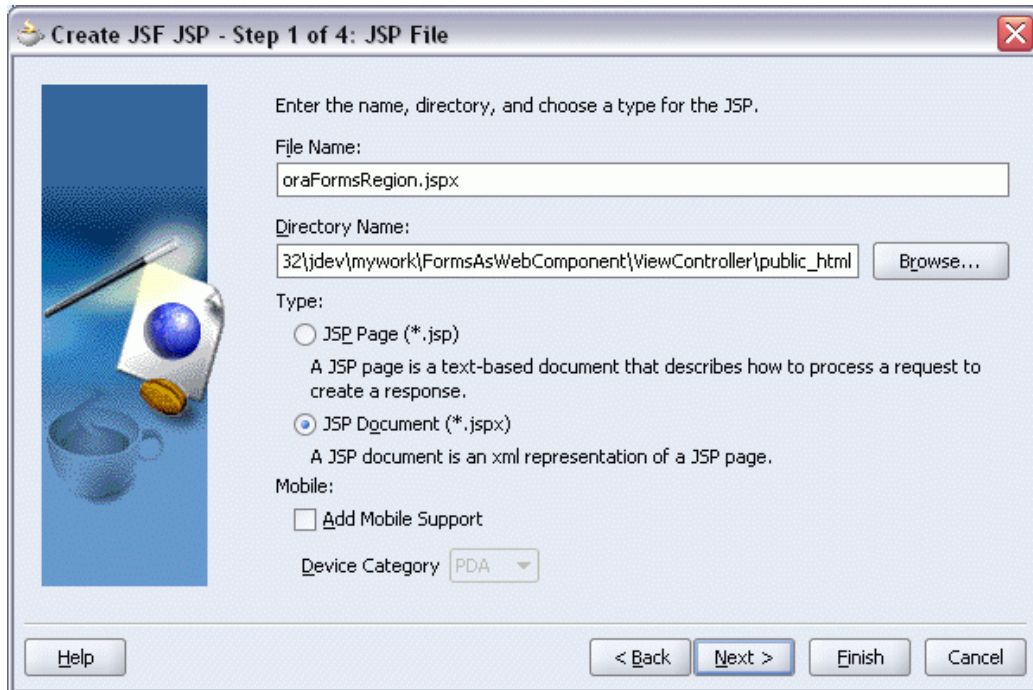
This text is also available in copy-and-paste.txt in the sample files for your convenience

```
var plugin_info = document.getElementById("forms_plugin_info");
var jversion = plugin_info.getAttribute("plug_ver");
var jcodebase = plugin_info.getAttribute("appcodebase");
var jwidth = plugin_info.getAttribute("appwidth");
var jheight = plugin_info.getAttribute("appheight");
document.write('<object classid="' + jversion + '"\n');
document.write('codebase="' + jcodebase + '"\n');
document.write('WIDTH="' + jwidth + '"\n');
document.write('HEIGHT="' + jheight + '"\n');
document.write('HSPACE="0"\n');
document.write('VSPACE="0">\n');
```

6. In JDeveloper go to the Applications Navigator, expand ViewController and right click the Web Content folder. Choose New and from the Web Tier > JSF category, select JSF JSP.



7. Use `oraFormsRegion.jspx` as file name and accept the rest of the defaults.



8. When the editor for `oraFormsRegion.jspx` opens switch to the source view by using the tabs at the bottom of the editor window. Select all source code and replace it with:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
<?xml version='1.0' encoding='windows-1252'?>
<af:regionDef var="regionParams"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:af="http://xmlns.oracle.com/adf/faces">
  <f:verbatim></f:verbatim>
</af:regionDef>
```

9. Copy the HTML to include the Forms applet and paste it between `<f:verbatim>` and `</f:verbatim>`:

This text is also available in copy-and-paste.txt in the sample files for your convenience

10. Expand ViewController Web Content > META-INF in JDeveloper and double click `region-metadata.xml` to edit it. Switch to source view.

11. Paste the following code between `<faces-config>` and `</faces-config>`

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
<component>
  <component-type>
    com.oracle.demo.FormsAsWebComponent.regions.oracleForm
  </component-type>
  <component-class>
    oracle.adf.view.faces.component.UIXRegion
  </component-class>
  <component-extension>
    <region-jsp-ui-def>/oraFormsRegion.jsp</region-jsp-ui-def>
  </component-extension>
  <attribute>
    <attribute-name>formModuleName</attribute-name>
    <attribute-class>java.lang.String</attribute-class>
    <attribute-extension>
      <required>true</required>
    </attribute-extension>
  </attribute>
</component>
```

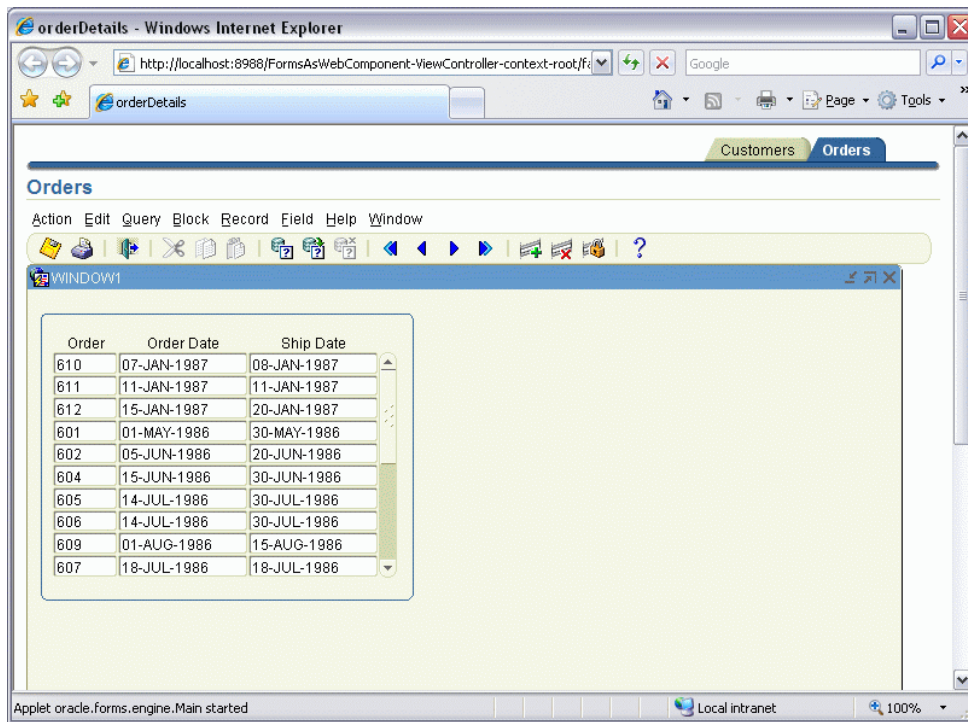
12. Open the `orderDetails.jspx` page in JDeveloper. Also switch to the source view. Then find the `<af:panelForm>` tag for the search form. This is just after the `<f:facet>` tag for the `appAbout` region. Before this `<af:panelForm>`, but after `<f:facet>` type the following:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
<af:region id="oraFormRegion"
  regionType="com.oracle.demo.FormsAsWebComponent.regions.oracleForm">
  <f:attribute name="formModuleName" value="orders.fmx" />
</af:region>
```

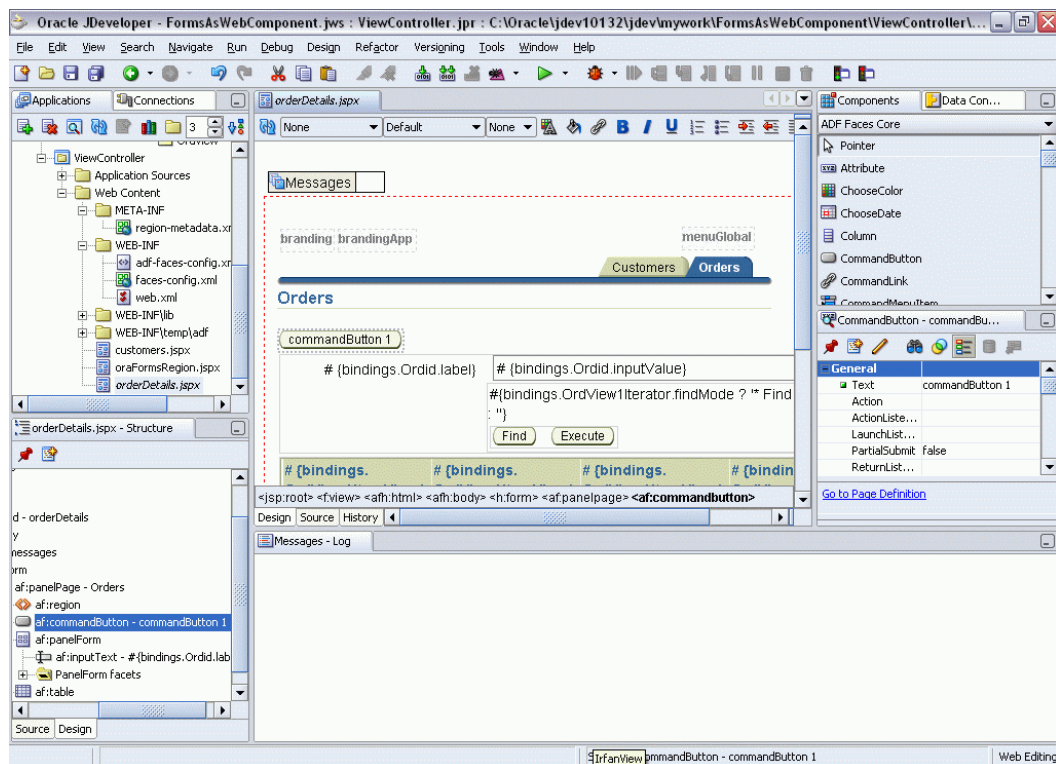
13. If you run the `orderDetails.jspx` page now you can see that the Forms applet is started, but it prompts for a username and password to logon to the database. To prevent this edit the `oraFormsRegion.jspx` page and replace `"userid="` with `"userid=formsdemo/formsdemo@xe"` **twice**. This hard codes the connection details to the database and makes these details visible to the end user in the HTML source. An alternative would be to use Oracle Single Sign On in combination with Oracle Forms. In that setup the database connection would be made with the credentials of the actual user.
14. We need to clean up some additional details in `orderDetails.jspx`:
- Set the value of `splashScreen` to `"no"` **twice**
 - Set the value of `background` to `"no"` **twice**

- Set the value of `colorScheme` to “blaf” **twice**. This is an undocumented color scheme that uses Oracle’s BLAF guideline colors.
 - Set the value of `logo` to “no” **twice**
15. If you now run `orderDetails.jspx` the page will include the Oracle Form running in an applet. It still queries all of the order records and is not visually integrated into the application. That’s all covered in the next sections.



9 Setup Inbound JavaScript API

1. We're going to setup the Inbound JavaScript API by adding a JSF button to the orderDetails.jspx page and have it perform a Commit action in the running Form. First open the orderDetails.jspx page in the Design View. Then drag a CommandButton component from the ADF Faces Core library in the Component Palette. Drop it just behind the Orders title in the visual editor.



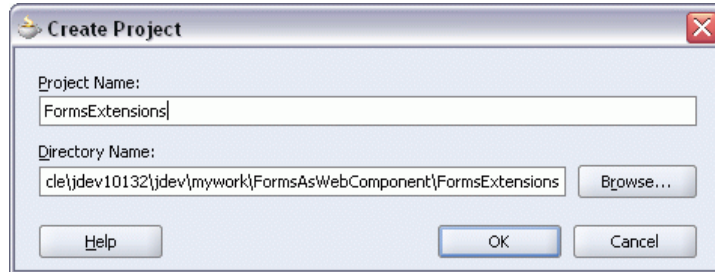
2. Change the Text property to "Save Form" and set the Onclick property in the JavaScript Events group to `document.formsapplet.raiseEvent('do_key','commit_form');`. Set the PartialSubmit property to true. This ensures not the entire page (including the Java applet) is reloaded when the button is pressed. Otherwise the Forms java applet would be restarted and reset each time the button is pressed.
3. The JavaScript refers to an object with the id "formsapplet". We need to make sure the Forms applet does use this ID. Open the oraFormsRegion.jspx and add `ID="formsapplet"` to the OBJECT tag. Also add `ID="formsapplet"` to the EMBED tag further down in the page.
4. Edit `DEVSUITE_HOME\forms\java\forms_ie.js` and add

```
document.write('id="formsapplet"\n');
```

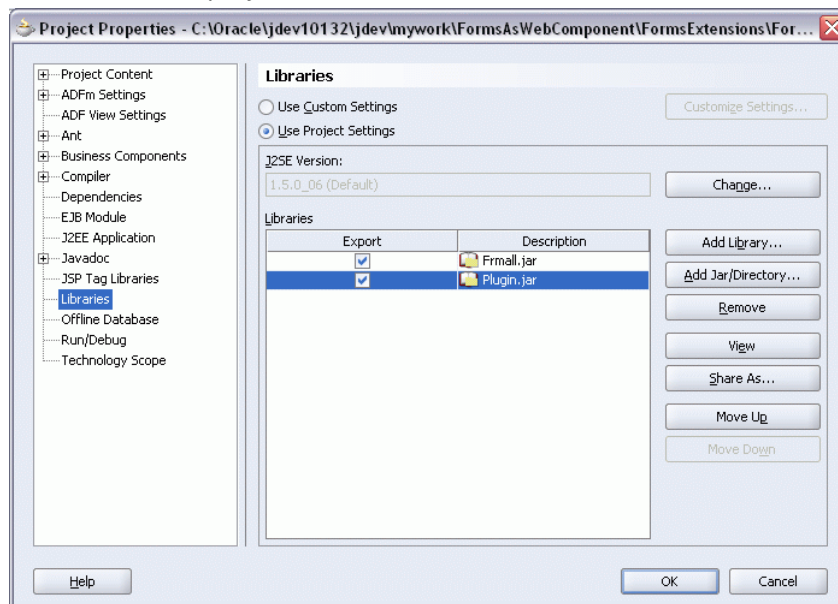
before the line

```
document.write('codebase="' + jcodebase + '"\n');
```

5. We now need to subclass the `oracle.forms.engine.Main` class to add a public method `raiseEvent`. In JDeveloper right click the FormsAsWebComponents workspace in the Applications Navigator. Select New Project. Choose Empty Project from the New Gallery. Name the project “FormsExtensions”



6. Double click the FormsExtensions project to open the Project Properties editor. Go to the Libraries node and press the Add Jar/Directory button. Browse to `DEVSUITE_HOME\forms\java\frmall.jar`. Press the button again and also add `DEVSUITE_HOME\jdk\jre\lib\plugin.jar`. This adds these two JAR files to the ClassPath of the project.



- Right click the FormsExtensions project and select New. From the New Gallery select General > Simple Files > Java Class to create a new Java class. Name the class "CommunicatorBean", set the package to "com.oracle.demo.ExtendedForms". Have the class extend "oracle.forms.ui.VBean". Leave the checkboxes as they are (Public and Generate Default Constructor checked).



- Switch the Java editor to Source view and add the following between the curly brackets of the CommunicatorBean class:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
static IHandler mHandler;
protected static final ID EVENT_MSG_TO_FORMS =
    ID.registerProperty("MessageToForms");
protected static final ID PROP_EVENT =
    ID.registerProperty("Event");
protected static final ID PROP_PAYLOAD =
    ID.registerProperty("Payload");

public void init(IHandler handler) {
    super.init(handler);
    mHandler = handler;
}

public void sendMessageToForms(String event, String payload) {
    try {
        mHandler.setProperty(PROP_EVENT, event);
        mHandler.setProperty(PROP_PAYLOAD, payload);
        // trigger WHEN-CUSTOM-ITEM-EVENT trigger in Forms
        CustomEvent ce = new CustomEvent(mHandler, EVENT_MSG_TO_FORMS);
        dispatchCustomEvent(ce);
    } catch (FException e) {
        e.printStackTrace();
    }
}
```

9. Press Alt-Enter repeatedly to import `oracle.forms.handler.IHandler`, `oracle.forms.properties.ID` and `oracle.forms.ui.CustomEvent`
10. Right click the FormsExtensions project and select New. From the New Gallery select General > Simple Files > Java Class to create a new Java class. Name the class "MainEngine", set the package to "com.oracle.demo.ExtendedForms". Have the class extend "oracle.forms.engine.Main". Leave the checkboxes as they are (Public and Generate Default Constructor checked).



11. Switch the Java editor to Source view. Add the following between the curly brackets of the MainEngine class:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
private CommunicatorBean findFirstCommunicator() {
    CommunicatorBean communicator = null;
    DesktopContainer desktop = getDesktop();
    if (null != desktop) {
        // loop al windows in the order they are stacked
        for (int iWindow = 0;
            (iWindow < desktop.getWindowCount()) && (null ==
                communicator);
            iWindow++) {
            // see if this window contains a Communicator
            LWWindow window = desktop.getWindowAtStackPosition(iWindow);
            communicator = findFirstCommunicatorInContainer(window);
        }
    }
    return communicator;
}

private CommunicatorBean findFirstCommunicatorInContainer(
    Component component) {
    if (component instanceof CommunicatorBean) {
        // the component being evaluated is a communicator
    }
}
```

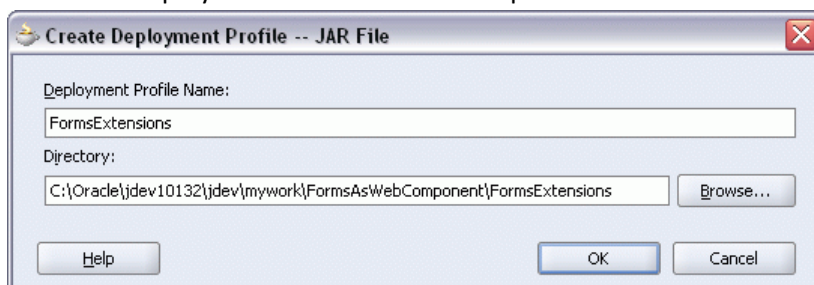
```

        // return it, so the recursion stops and the communicator is
        // returned
        return (CommunicatorBean)component;
    } else if (component instanceof Container) {
        // the component being evaluated is a container, evaluate its
        // children
        Component[] children = ((Container)component).getComponents();
        CommunicatorBean communicator = null;
        for (int iChild = 0;
            (iChild < children.length) && (null == communicator);
            iChild++) {
            communicator =
                findFirstCommunicatorInContainer(children[iChild]);
        }
        // we get here when all of the children have been evaluated and
        // none of them returned a Communicator or if a child did actually
        // return a Communicator
        return communicator;
    } else {
        // the component being evaluated is not a container, nor a
        // communicator.
        return null;
    }
}

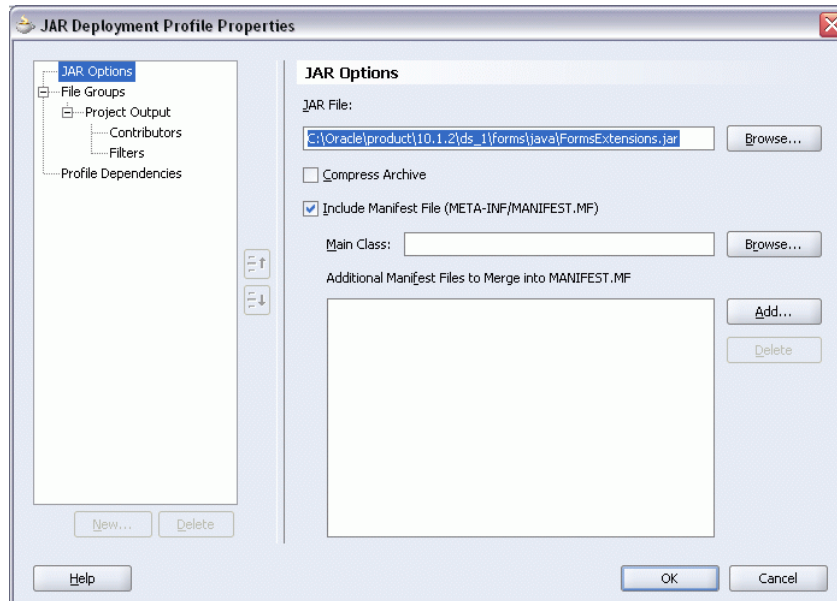
public void raiseEvent(String event, String payload) {
    CommunicatorBean communicator = findFirstCommunicator();
    if (null != communicator) {
        communicator.sendMessageToForms(event, payload);
    }
}
}

```

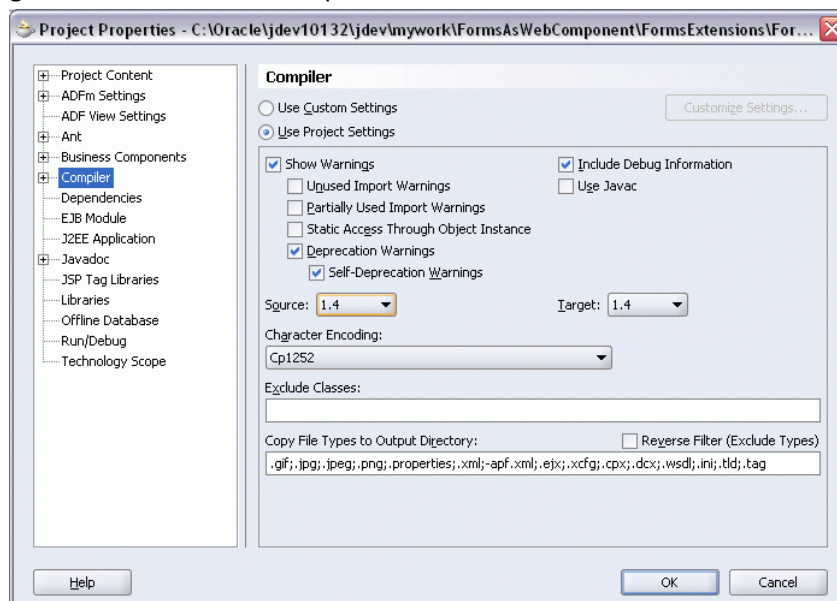
12. Press Alt-Enter repeatedly to import `java.awt.Component`, `java.awt.Container`, `oracle.ewt.lwAWT.lwWindow.DesktopContainer` and `oracle.ewt.lwAWT.lwWindow.LWWindow`
13. Right click the `FormsExtensions` project once more and select New. Select JAR file under General > Deployment Profiles. Name the profile "FormsExtensions"



14. Change the JAR file location to `DEVSUITE_HOME\forms\java\FormsExtensions.jar`



15. Double click the FormsExtensions project node to open the Project Properties editor. Go to the Compiler node and change the Source and Target property to 1.4 to make sure the generated classes are compatible with Java 1.4 which is used on the client.



16. Expand the Resources node in the Applications Navigator, right click the deployment profile and select Deploy to JAR file to create the JAR file. This creates the JAR file and puts it in the Oracle Forms Java directory.
17. Go back to the oraFormsRegion.jspx and change the Archive parameter from "frmall.jar" to "FormsExtensions.jar,frmall.jar" **twice**. This ensures the JAR file is loaded by the client when starting Oracle Forms.

18. Also change "oracle.forms.engine.Main" to
"com.oracle.demo.ExtendedForms.MainEngine" **twice**.

19. To allow JavaScript to communicate to the Java Applet add

```
<PARAM NAME="mayscript" VALUE="true">
```

between all the other parameters and further down the document add

```
mayscript="true"
```

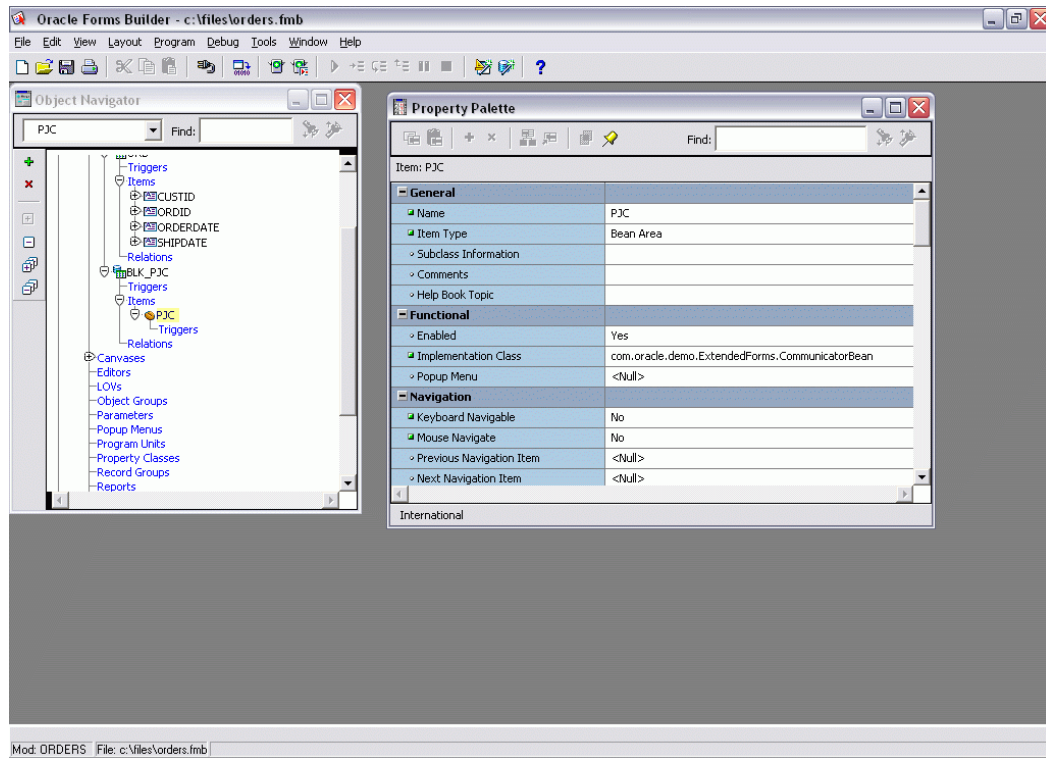
to the attributes of the EMBED tag.

20. Go to Forms builder to edit the orders.fmb Form. Select the Data Blocks node and press the toolbar button with the green plus to create a new data block. Opt to create the new block manually.



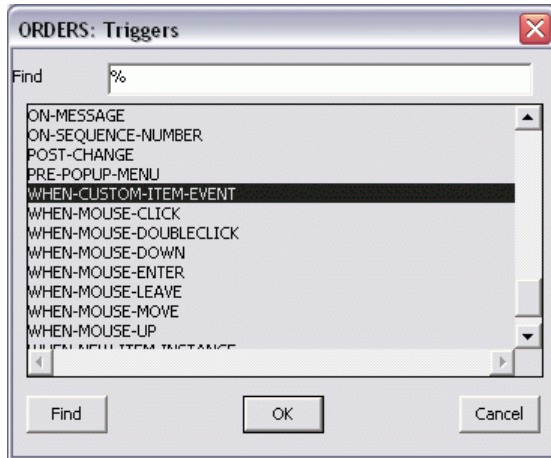
21. Set the Name property of the block to "BLK_PJC" and the Database Block property to No. Drag the BLK_PJC block below the ORD block so it is the last block in the Form.

22. Select the Items node under the BLK_PJC bean and press the green plus button again to create a new item. Set the following properties:



- Name: PJC
- Item Type: Bean Area (this is the very first option in the pop list)
- Implementation Class: `com.oracle.demo.ExtendedForms.CommunicatorBean`
- Keyboard Navigable: No
- Mouse Navigable: No
- Canvas: CANVAS4 (or whatever name the one canvas has)
- Width: 1
- Height: 1
- Bevel: None

23. Double click the Triggers node under the PJC item to create a new trigger. Select **WHEN-CUSTOM-ITEM-EVENT** as trigger type.



24. Add the following code to the trigger:

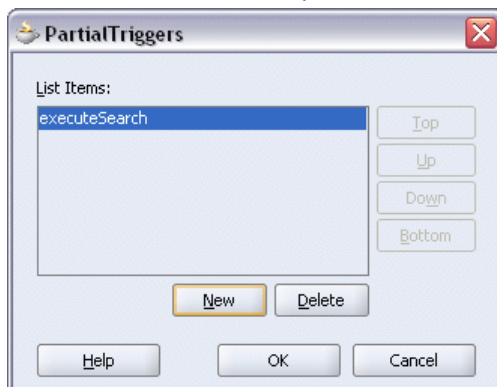
This text is also available in copy-and-paste.txt in the sample files for your convenience

```
declare
  BeanEventDetails ParamList;
  ParamType        number := text_parameter;
  Event             varchar2(1000);
  Payload           varchar2(1000);
begin
  BeanEventDetails :=
    get_parameter_list(:system.custom_item_event_parameters);
  get_parameter_attr(BeanEventDetails, 'Event', ParamType, Event);
  get_parameter_attr(BeanEventDetails, 'Payload', ParamType, Payload);
  if event='do_key' then
    do_key(payload);
  end if;
end;
```

25. Press Ctrl-S and Ctrl-T so save and generate the form.
26. Run the orderDetails page and see how you can change data in the Form and then press the JSF button to perform a commit in the Oracle Form.

10 Setup Outbound JavaScript API

1. The next thing we're going to setup is the Outbound JavaScript API. We will use this to query the detail JSF table with the order lines of the order selected in Oracle Forms. First go to JDeveloper and edit the orderDetails.jspx page.
2. Select the inputText item used for entering the Order ID in the search Form. Then go to the property Palette and set the Id property to "ordid". This gives the object and ID which we can reference from JavaScript.
3. Select the commandButton to execute the search (labeled Execute, **not** Find). Set the Id property to "executeSearch". Set the PartialSubmit property to true to ensure not the entire page is refreshed when this button is pressed, but only the objects that have been marked to refresh.
4. Click on the JSF table to show the order lines. Then in the structure pane navigate to the parent af:table node. Then go to the property palette and go to the PartialTriggers property. Press the button to go to the editor for PartialTriggers. Press the New button to add a new PartialTrigger. Click on the new PartialTrigger and select executeSearch from the poplist. This ensures the JSF table with the order details is refreshed when the executeSearch button is pressed.



5. In the structure pane navigate to the parents of the current node until you find the h:form node that defines the HTML form. Select this node and go to the property palette to set the Id to "frm". This will be used as prefix for all the HTML ID's of the elements contained in this form.
6. Once again click on the inputText item to input the Order ID in the search form. In the structure pane select the parent af:panelForm node. Then in the property palette set the InlineStyle property to "display:none;". This adds some CSS to the element to hide it from the browser. Do not use the rendered=false property since that completely removes the HTML from the client.
7. Now go to the CommunicatorBean.java and add the following code at the top of the class:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
protected static final ID PROP_EVAL_EXPR =  
    ID.registerProperty("EvalExpression");  
protected static final ID PROP_EVAL_RESULT =  
    ID.registerProperty("EvalResult");  
private String mExprResult;
```

8. Add the following code to the end of the class:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
public boolean setProperty(ID property, Object value) {  
    if (PROP_EVAL_EXPR == property) {  
        JSONObject appletWindow = JSONObject.getWindow(mHandler.getApplet());  
        Object evalResult = appletWindow.eval(value.toString());  
        setProperty(PROP_EVAL_RESULT, evalResult);  
        return true;  
    } else if (PROP_EVAL_RESULT == property) {  
        mExprResult = (value == null ? null : value.toString());  
        return true;  
    } else {  
        return super.setProperty(property, value);  
    }  
}  
  
public Object getProperty(ID property) {  
    if (PROP_EVAL_RESULT == property) {  
        return mExprResult;  
    } else {  
        return super.getProperty(property);  
    }  
}
```

9. Press Alt-Enter to import the netscape.javascript.JSObject class
10. Right click FormsExtensions.deploy in JDeveloper and select Deploy to JAR file
11. Now go to Forms builder and add a WHEN-NEW-RECORD-INSTANCE trigger to the ORD block. Use the following PL/SQL code:

This text is also available in copy-and-paste.txt in the sample files for your convenience

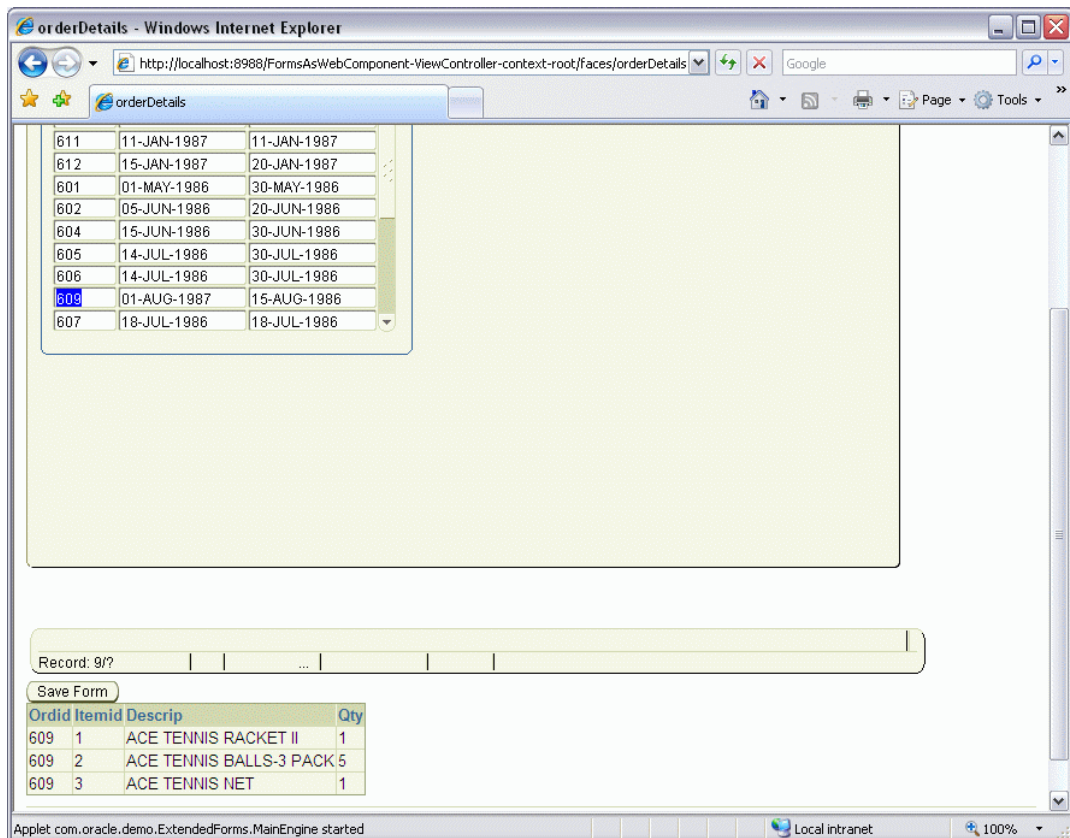
```
declare  
    onClick varchar2(2000);  
begin  
    -- set the Order ID  
    set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
```



```

'document.getElementById('frm:ordid').value=' || :ord.ordid);
-- get the onClick event of the search submit button
set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
'document.getElementById('frm:executeSearch').onclick');
onClick := get_custom_property('BLK_PJC.PJC', 1, 'EvalResult');
-- only retain onClick code between { and }
onClick := substr(onClick, instr(onClick, '{')+1);
onClick := substr(onClick, 1, instr(onClick, '}', -1)-1);
onClick := substr(onClick, 1, instr(onClick, 'return false;', -1)-1);
onClick := trim(onClick);
onClick := rtrim(ltrim(onClick, chr(10)), chr(10));
onClick := trim(onClick);
-- perform the onClick event
set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression', onClick);
-- synchronize to actually set the property
-- otherwise Forms runtime waits for the next roundtrip to the client
synchronize;
end;
```

12. Compile the form using Ctrl-S and Ctrl-T.
13. Now run the orderDetails.jspx page from JDeveloper and notice how the JSF table with order details is queried when scrolling through the orders in Oracle Forms. Some orders have no details, which might seem like the synchronization does not work. Check orders 606 and 609 to see some details.



Order ID	Date	Description
611	11-JAN-1987	11-JAN-1987
612	15-JAN-1987	20-JAN-1987
601	01-MAY-1986	30-MAY-1986
602	05-JUN-1986	20-JUN-1986
604	15-JUN-1986	30-JUN-1986
605	14-JUL-1986	30-JUL-1986
606	14-JUL-1986	30-JUL-1986
609	01-AUG-1987	15-AUG-1986
607	18-JUL-1986	18-JUL-1986

Record:	9/?		
Save Form			
Orderid	Itemid	Descrip	Qty
609	1	ACE TENNIS RACKET II	1
609	2	ACE TENNIS BALLS-3 PACK	5
609	3	ACE TENNIS NET	1

11 Suspend and Reuse Forms Applet

1. If you run the application and navigate back and forth between the Customers and Orders tab you can notice that the Forms applet is started from scratch each time you navigate to the Orders tab. This can be prevented by using a feature called Legacy LifeCycle Cache. In JDeveloper open oraFormsRegion.jsx. Add

```
<PARAM NAME="legacy_lifecycle" VALUE="true">
```

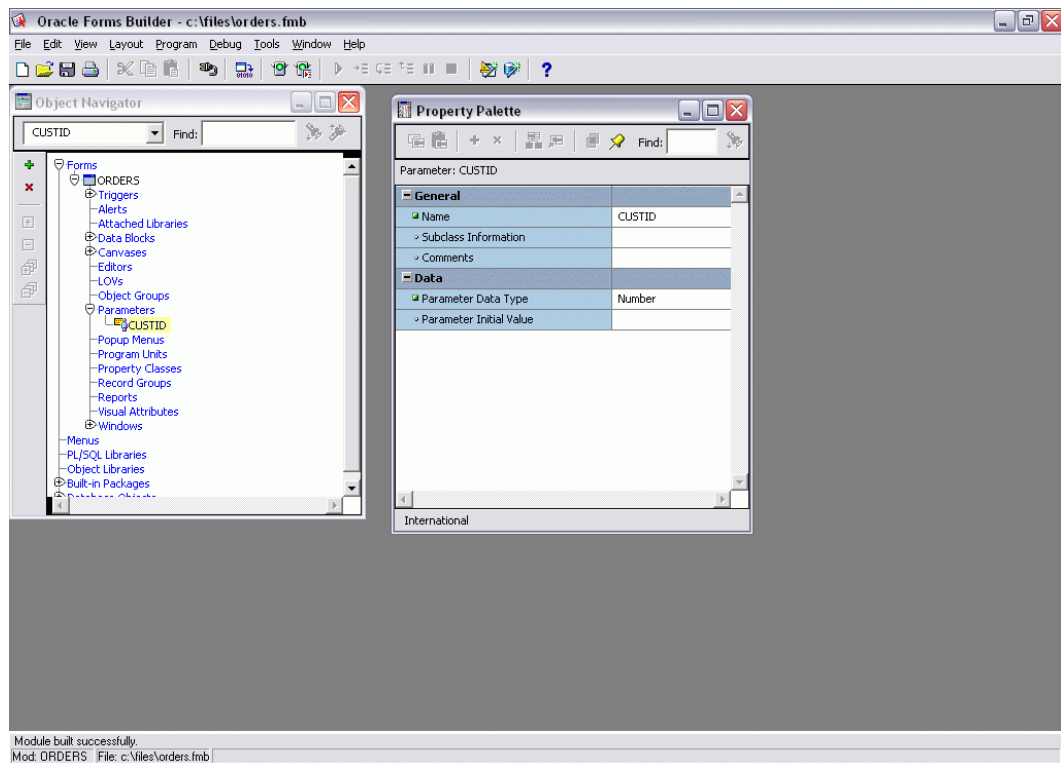
between all the other parameters and further down the document add

```
legacy_lifecycle="true"
```

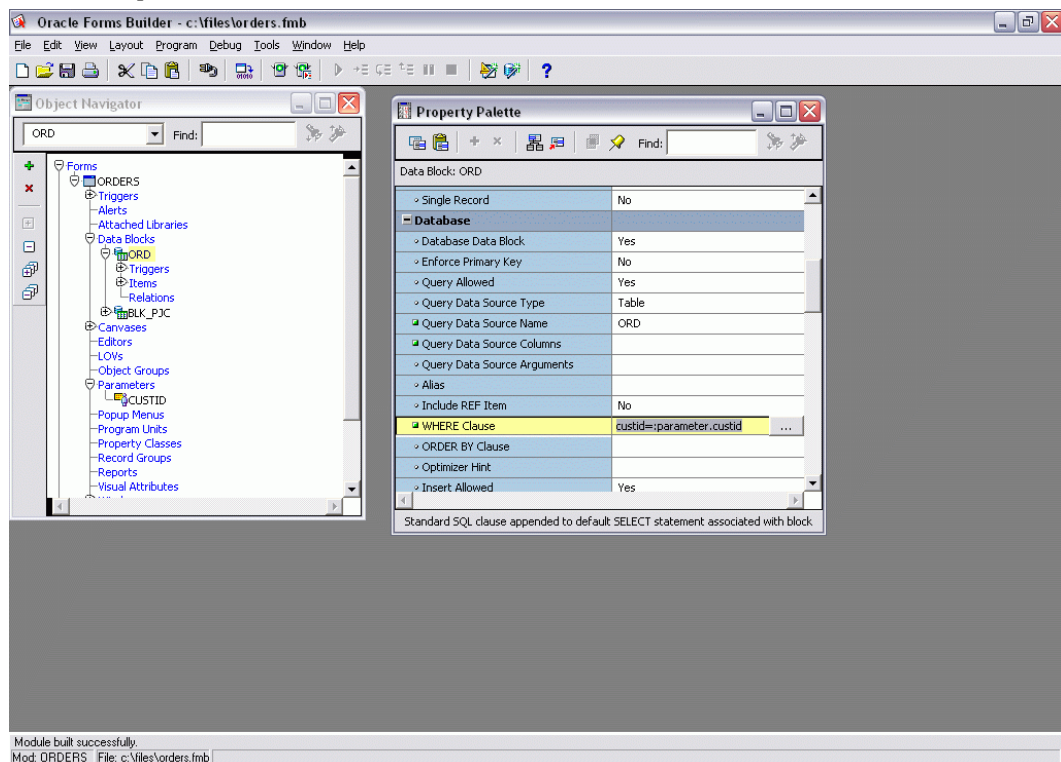
to the attributes of the EMBED tag.

2. Run the application again and notice how the applet is retained when navigating away from the Orders tab and back to the Orders tab.
3. The next thing we need to accomplish is that the Oracle Form queries only the orders for the selected client on the Customers tab. We first need to make sure the ID of the current customer is included in the orderDetails page, so the Oracle Form can get to it using the Outbound JavaScript API. Open the orderDetails.jsx page in the JDeveloper editor and make sure you're in the visual editor. Then go to the Data Control Palette and drag AppModuleDataControl > CustomerView1 > Custid to the orderDetails page, just behind the Orders title. Drop it as a Texts > ADF Input Text. Select the new af:inputText node in the Structure Pane. Right click the node and select Convert. Convert the item to an inputHidden. Finally set the ID in the Property Palette to "customerID".

- Now open the orders.fmb form in Forms Builder. Double click the Parameters node to create a new Parameter. Press F4 to open the Property Palette. Set the name to "CUSTID" and the Data Type to Number.



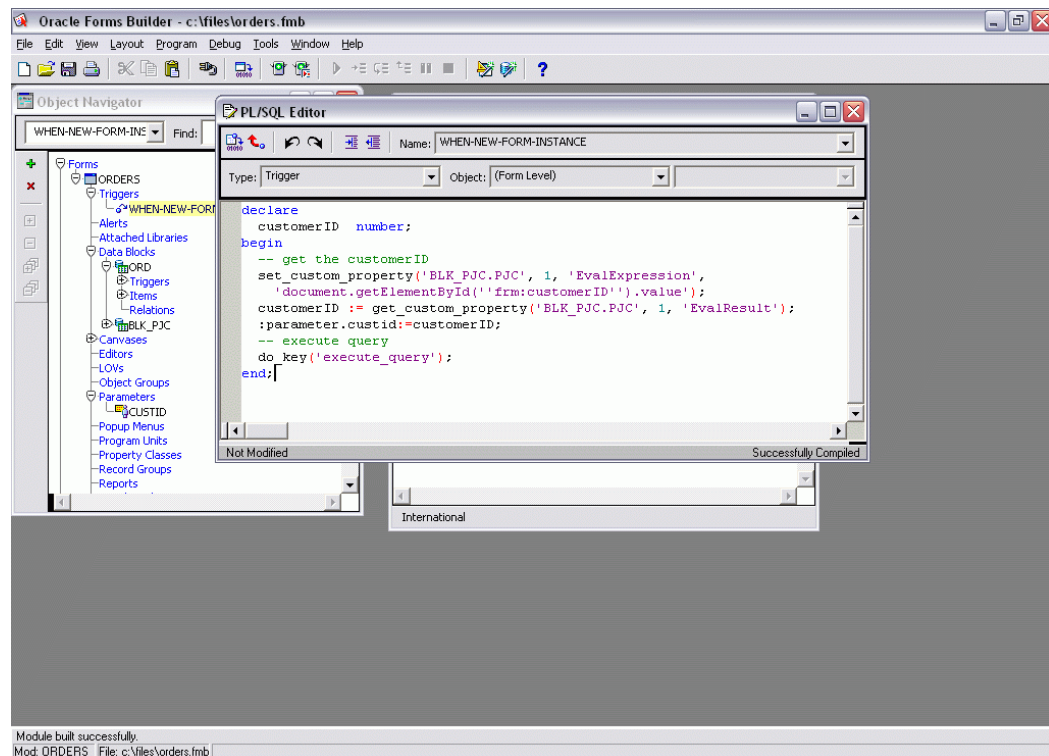
- Next go to the properties of the ORD block and set the Where Clause property to "custid=:parameter.custid"



- Now go to the WHEN-NEW-FORM-INSTANCE trigger and replace the single line to execute the query with the following code:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
declare
    customerID    number;
begin
    -- get the customerID
    set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
        'document.getElementById(''frm:customerID'').value');
    customerID := get_custom_property('BLK_PJC.PJC', 1, 'EvalResult');
    :parameter.custid:=customerID;
    -- execute query
    do_key('execute_query');
end;
```



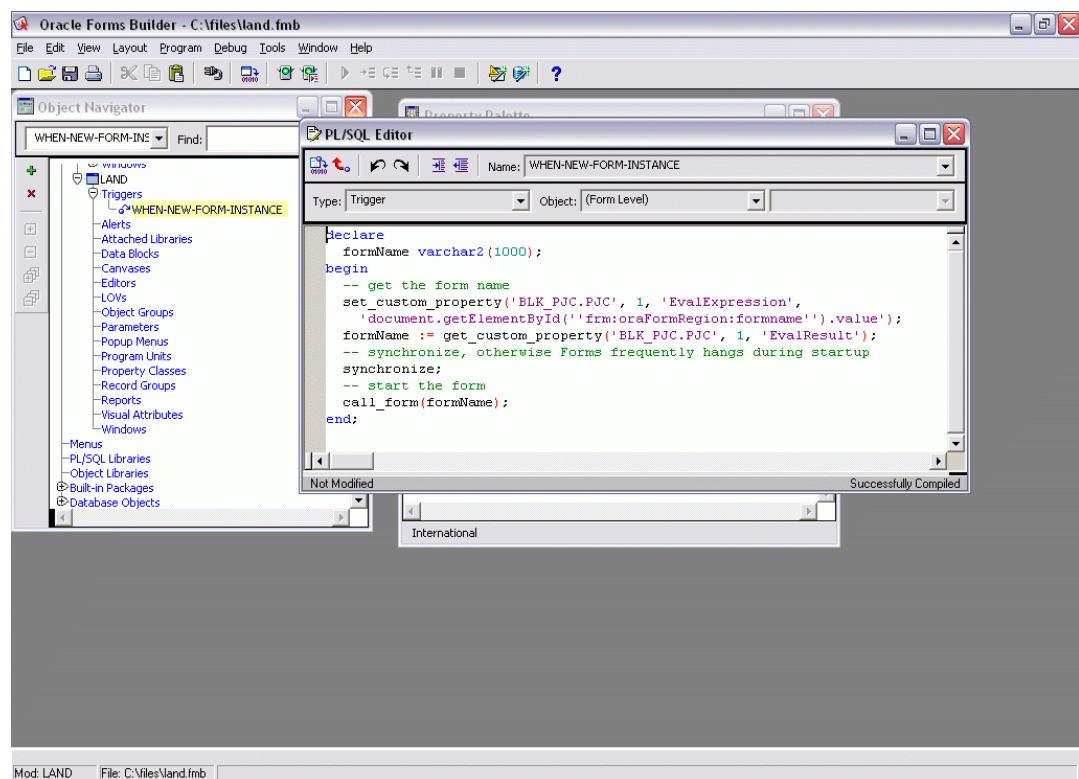
- Save and generate the Form using Ctrl-S and Ctrl-T. Now run the application and notice only the orders for the first customer are shown in the Form. Unfortunately moving to the Customers tab, selecting another customer and returning to the Orders tab does not query the orders for this other customer. That's because the Form is not queried again when the suspended applet is restored from the LifeCycle Cache. We will come to this later.

8. First we have to solve another problem. The name of the Form module (orders.fmx) is currently hard coded in the oraFormsRegion.jsx. We need to be able to reuse this region definition in other JSF pages that include other Forms. Also, we need to move the Form module name out of the <OBJECT> and <EMBED> tags. If we do not do that, the content of these tags would be different for each form and that would prevent Forms from using the Legacy LifeCycle Cache. Create a new Form in Forms Builder and save it (Ctrl-S) as c:\files\land.fmb.
9. In the land.fmb Form, create a WHEN-NEW-FORM-INSTANCE trigger with the following code:

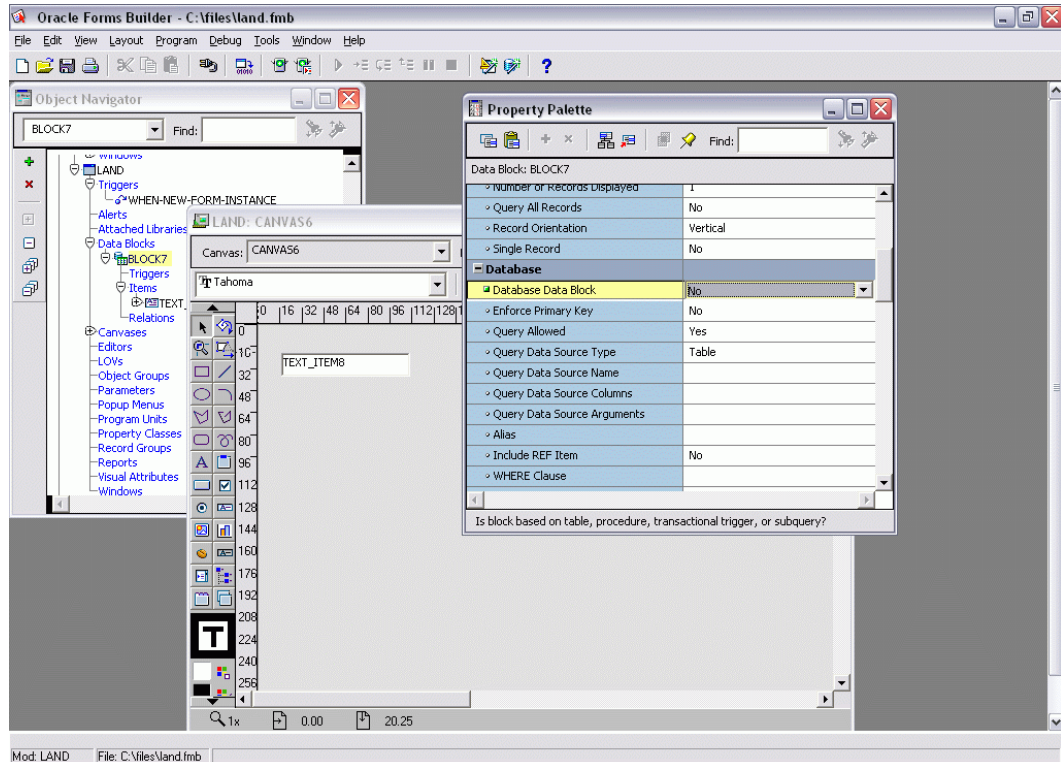
This text is also available in copy-and-paste.txt in the sample files for your convenience

```

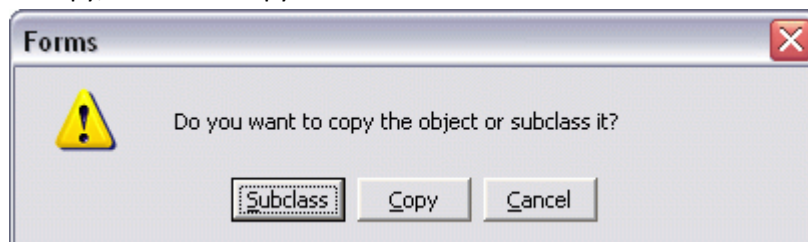
declare
    formName varchar2(1000);
begin
    -- get the form name
    set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
        'document.getElementById(''frm:oraFormRegion:formname'').value');
    formName := get_custom_property('BLK_PJC.PJC', 1, 'EvalResult');
    -- start the form
    call_form(formName);
end;
  
```



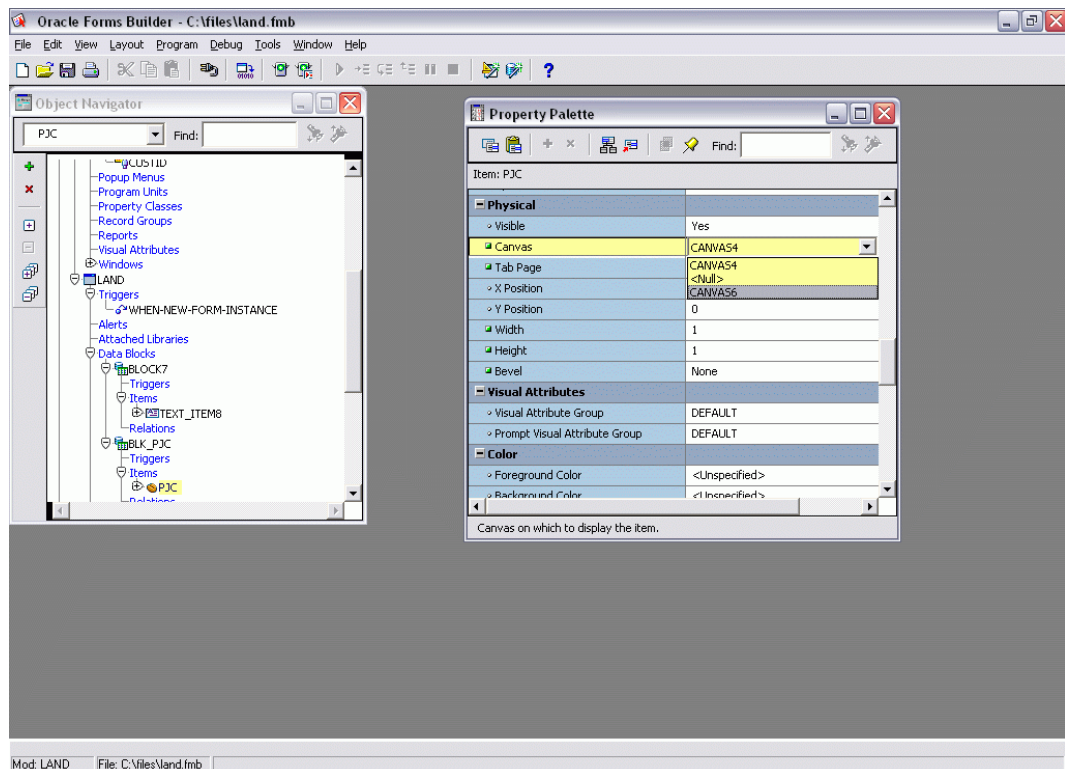
10. Press F2 to open the Layout editor and click on the Textitem button in the palette, then click in the Layout editor. This will create a dummy text item. Each Form must have at least one item that can be navigable for the user. For the created block set the Database Block property to No.



11. In the Object Navigator of Forms Builder drag-and-drop the BLK_PJC from the orders.fmb Form to the land.fmb form below the existing block in land.fmb. When asked to subclass or copy, choose to copy.



12. For the PJC item in the BLK_PJC block change the Canvas property to the only existing canvas in the land.fmb form.



13. Save and generate the form by pressing Ctrl-S and Ctrl-T.
14. In JDeveloper edit oraFormsRegion.jspx and add the following snippet before the `<f:verbatim>` tag:

```
<af:inputHidden value="#{regionParams.formModuleName}" id="formname" />
```

15. In the same file replace "orders.fmx" with "land.fmx" **twice**
16. If you now run orderDetails.jspx again it will still display the same Orders Form, but this time it will have started the Land form first. The Land form inspected which Form is required and then called that form using the `CALL_FORM` built in. This concept allows you to reuse the oraFormsRegion.jspx on many other JSF pages with completely different Forms but still reusing the same Forms applet from the LifeCycle Cache.
17. A remaining problem is that the Orders Form is queried with the orders of the first customer. When selecting another customer on the Customers page and returning to the orderDetails page the orders of the initial customer are still displayed since the suspended Forms applet is restored from the Applet LifeCycle Cache. We need to add handling to the reactivation of the Forms Applet.
18. In JDeveloper edit `MainEngine.java` and add the following to the end of the class definition. **Please leave the three lines to print the origin of the concept.**

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
public void show() {
    super.show();
    System.out.println("Concept of using Oracle Forms as Web Component");
    System.out.println("courtesy of Wilfred van der Deijl, OraTransplant");
    System.out.println("http://www.oratransplant.nl/oracle-forms-as-web-
component/");
    raiseEvent("WHEN-APPLET-ACTIVATED", "");
}
```

19. Edit CommunicatorBean.java and add to the variable declaration at the top:

```
protected static final ID PROP_APPLET_ACTIVE =
    ID.registerProperty("AppletActive");
```

20. Replace the getProperty() method in CommunicatorBean.java with the following code:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
public Object getProperty(ID property) {
    if (PROP_EVAL_RESULT == property) {
        return mExprResult;
    } else if (PROP_APPLET_ACTIVE == property) {
        Applet applet = mHandler.getApplet();
        return (applet == null ? "FALSE" :
            (applet.isShowing() ? "TRUE" : "FALSE"));
    } else {
        return super.getProperty(property);
    }
}
```

21. Press Alt-Enter to import `java.applet.Applet`
22. Right click the FormsExtensions.deploy and select Deploy to JAR file to create a new FormsExtensions.jar file.
23. Go to Forms Builder and edit the WHEN-CUSTOM-ITEM-EVENT trigger of BLK_PJC.PJC in the Orders.fmb form. Replace the code with:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
declare
    BeanEventDetails ParamList;
    ParamType        number := text_parameter;
    Event             varchar2(1000);
    Payload           varchar2(1000);
```

```
begin
  BeanEventDetails :=
    get_parameter_list(:system.custom_item_event_parameters);
  get_parameter_attr(BeanEventDetails, 'Event', ParamType, Event);
  get_parameter_attr(BeanEventDetails, 'Payload', ParamType, Payload);
  if event='do_key' then
    do_key(payload);
  elsif event='WHEN-APPLET-ACTIVATED' then
    exit_form(no_validate);
  end if;
end;
```

24. Press Ctrl-S and Ctrl-T to save and generate the Form.

25. Go to Forms Builder and edit the WHEN-CUSTOM-ITEM-EVENT trigger of BLK_PJC.PJC in the Land.fmb form. We're going to move a lot of the code from the WHEN-NEW-FORM-INSTANCE trigger to this trigger. Replace the code with:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
declare
  BeanEventDetails ParamList;
  ParamType        number := text_parameter;
  Event            varchar2(1000);
  Payload          varchar2(1000);
  formName         varchar2(1000);
  appletActive     varchar2(10);
begin
  BeanEventDetails :=
    get_parameter_list(:system.custom_item_event_parameters);
  get_parameter_attr(BeanEventDetails, 'Event', ParamType, Event);
  get_parameter_attr(BeanEventDetails, 'Payload', ParamType, Payload);
  if event='do_key' then
    do_key(payload);
  elsif event='WHEN-APPLET-ACTIVATED' then
    while true loop
      -- get the form name
      set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
        'document.getElementById(''frm:oraFormRegion:formname'').value');
      formName := get_custom_property('BLK_PJC.PJC', 1, 'EvalResult');
      -- start the form
      call_form(formName);
      -- if we get here the called form is closed
      appletActive := get_custom_property('BLK_PJC.PJC', 1,
        'AppletActive');
      if (appletActive='FALSE') then
        -- exit the loop if the user is closing the browser
        exit;
      end if;
    end loop;
  end if;
end;
```

This code has added an infinite loop in the Land.fmb form. This infinite loop is started on applet activation. The loop checks which Form needs to be started and starts that Form. If the Applet is reactivated from the Applet LifeCycle Cache while in a “real” form, that form performs an `exit_form`. This returns focus to the infinite loop in the landing form. This loop immediately checks which Form to start next and it will start that Form. The only way to abandon the infinite loop is due to the user closing the browser.

You might be inclined to put this code in the `WHEN-NEW-FORM-INSTANCE` trigger of the landing form and not in the `WHEN-CUSTOM-ITEM-EVENT` trigger. Doing so would lead to problems. On initial startup, the landing form would start. In the `WHEN-NEW-FORM-INSTANCE` it would start the “real” form. Once that startup is completed the focus is returned to the user. This triggers the `MainEngine.show()` method. That fires the `WHEN-APPLET-ACTIVATED` event. That would immediately exit the “real” form, returning focus to the landing form. That form would be in an infinite loop and would start the “real” form again. Once that is completed the focus is finally returned to the user. The end result is the same to the end user, but performance is worse since the “real” form is started twice on initial startup.

26. Remove the `WHEN-NEW-FORM-INSTANCE` trigger from the landing form, since that functionality is now covered by the `WHEN-CUSTOM-ITEM-EVENT` trigger.
27. Press Ctrl-S and Ctrl-T to save and generate the Form.
28. Run the application. On the Customers tab select another customer and press the “Show Orders” button to effectuate the selection. Then navigate to the Orders tab to see the orders for that customer. You can switch back and forth between the tabs and the Form will display the orders of the selected customer each time without restarting the Forms applet.

12 Visual Integration

1. The last and final step is to visually integrate the Form in the web page. You could also decide to stop here and keep the feature rich user interface of Oracle Forms. If you want to integrate it completely and go for the intuitive web user experience, then the key is to remove the menu, tool bar and status bar. This can be done by surrounding the Forms applet with a `<div>` tag and manipulate the CSS properties of that tag at runtime. To make this work in Mozilla Firefox we need to use two `<div>` tags. Start by editing `oraFormsRegion.jspx` in JDeveloper. Just after the `<f:verbatim>` tag add the following:

```
<div id="frmouterdiv" style="overflow:hidden; border-style:none;" >
<div id="frminnerdiv" style="overflow:scroll; border-style:none;" >
```

This adds two `<div>` tags which is necessary to make this work with Mozilla Firefox. A single `<div>` with `overflow:hidden` does not work if the `<div>` contains a Java Applet. Using `overflow:scroll` does crop the applet but uses scrollbars. As workaround we first crop the applet with a `<div>` using scrollbars and then crop that `<div>` again with another `<div>` that uses `overflow:hidden`.

2. Just before the closing `</f:verbatim>` tag add the following to close the two `<div>` tags:

```
</div></div>
```

3. After the closing `</f:verbatim>` add the following:

This text is also available in copy-and-paste.txt in the sample files for your convenience

```
<afh:script text="
/* calc size of inner div with extra room for scrollbars */
var innerWidth=#{regionParams.appletWidth}+17;
var innerHeight=#{regionParams.appletHeight}+17;
/* negate shift values to get margins */
var marginLeft=#{regionParams.appletShiftLeft}* -1;
var marginTop=#{regionParams.appletShiftUp}* -1;
/* set CSS properties */
document.getElementById('frmouterdiv').style.width=
  #{regionParams.appletWidth};
document.getElementById('frmouterdiv').style.height=
  #{regionParams.appletHeight};
document.getElementById('frminnerdiv').style.width=innerWidth;
document.getElementById('frminnerdiv').style.height=innerHeight;
document.getElementById('formsapplet').style.marginLeft=marginLeft;
document.getElementById('formsapplet').style.marginTop=marginTop;
"/>
```

This JavaScript takes a number of inputs on the applet size and shift. It calculates the actual values for the DIV and Margins in JavaScript and sets the CSS properties by DOM(Domain Object Model) manipulation. This ensures that the original definition of the

<OBJECT> and <EMBED> tags is left as is, so `legacy_lifecycle` still works.

Only use `/*` and `*/` to mark comments. Do not use `//`, since all lines will be appended to a single long line. Using `//` would comment out the remainder of the JavaScript.

4. Go to the top of `oraFormsRegion.jsx` and add the `xmlns:afh` namespace we just used for the `<afh:script>` tag:

```
<af:regionDef var="regionParams"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:afh="http://xmlns.oracle.com/adf/faces/html"
  xmlns:af="http://xmlns.oracle.com/adf/faces">
```

5. The few new parameters used in `oraFormsRegion.jsx` have to be defined in `META-INF/region-metadata.xml`: After the existing attribute definition for `formModuleName` add the following:

This text is also available in `copy-and-paste.txt` in the sample files for your convenience

```
<attribute>
  <attribute-name>appletWidth</attribute-name>
  <attribute-class>java.lang.Integer</attribute-class>
  <attribute-extension><required>true</required></attribute-extension>
</attribute>
<attribute>
  <attribute-name>appletHeight</attribute-name>
  <attribute-class>java.lang.Integer</attribute-class>
  <attribute-extension><required>true</required></attribute-extension>
</attribute>
<attribute>
  <attribute-name>appletShiftLeft</attribute-name>
  <attribute-class>java.lang.Integer</attribute-class>
  <attribute-extension><required>true</required></attribute-extension>
</attribute>
<attribute>
  <attribute-name>appletShiftUp</attribute-name>
  <attribute-class>java.lang.Integer</attribute-class>
  <attribute-extension><required>true</required></attribute-extension>
</attribute>
```

6. Next edit `orderDetails.jsx` to add values for these new parameters. Find the `<af:region>` tag and add the following attributes as children:

This text is also available in `copy-and-paste.txt` in the sample files for your convenience

```
<f:attribute name="formModuleName" value="orders.fmx"/>
<f:attribute name="appletWidth" value="300"/>
<f:attribute name="appletHeight" value="230"/>
<f:attribute name="appletShiftLeft" value="15"/>
```



```
<f:attribute name="appletShiftUp" value="94"/>
```

7. In Forms Builder set the `Background Color` property of the main Canvas of the Orders and Land form to “white” to let them blend in with the web page which also uses a white background.
8. In the Order Form add an `ON-ERROR` trigger (normally you would put this in a shared library):

This text is also available in `copy-and-paste.txt` in the sample files for your convenience

```
declare
  errorType varchar2(3) := error_type;
  errorCode number      := error_code;
  errorText varchar2(80) := error_text;
  alertMsg  varchar2(2000);
begin
  alertMsg := errorType || '-' || errorCode || ': ' || errorText;
  set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
    'alert('' || alertMsg || '')');
  raise form_trigger_failure;
end;
```

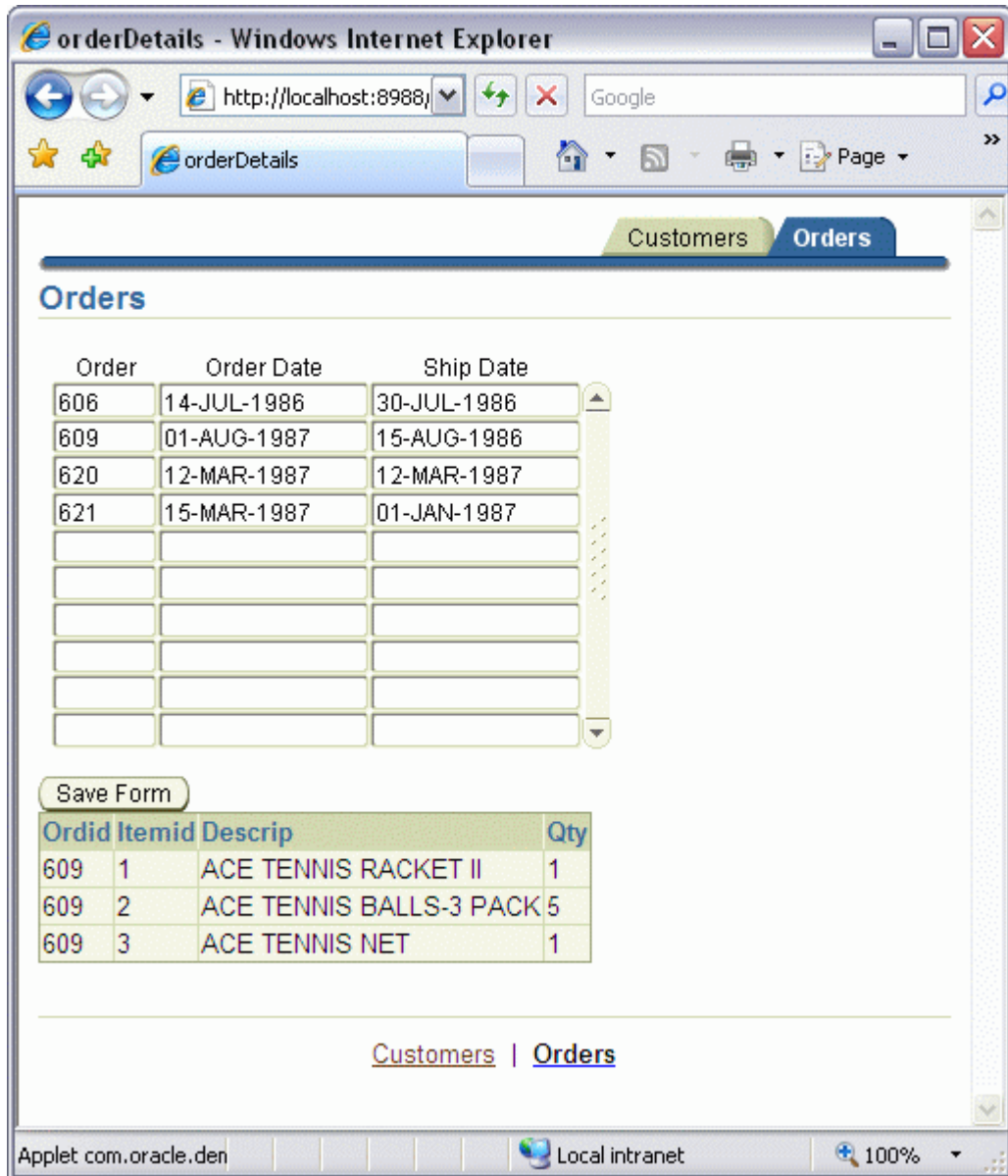
9. Also in the Order Form add an `ON-MESSAGE` trigger:

This text is also available in `copy-and-paste.txt` in the sample files for your convenience

```
declare
  errorType varchar2(3) := message_type;
  errorCode number      := message_code;
  errorText varchar2(80) := message_text;
  alertMsg  varchar2(2000);
begin
  alertMsg := errorType || '-' || errorCode || ': ' || errorText;
  set_custom_property('BLK_PJC.PJC', 1, 'EvalExpression',
    'alert('' || alertMsg || '')');
end;
```

10. For each form, press Ctrl-S and Ctrl-T to save and generate.

11. Run the application and see how the Form is now seamlessly integrated in the web page. Press the “Save Form” button to see the error and message triggers at work. They should raise a JavaScript alert instead of the normal Forms message or alert.



orderDetails - Windows Internet Explorer

http://localhost:8988/

Google

orderDetails

Customers Orders

Orders

Order	Order Date	Ship Date
606	14-JUL-1986	30-JUL-1986
609	01-AUG-1987	15-AUG-1986
620	12-MAR-1987	12-MAR-1987
621	15-MAR-1987	01-JAN-1987

Save Form

Ordid	Itemid	Descrip	Qty
609	1	ACE TENNIS RACKET II	1
609	2	ACE TENNIS BALLS-3 PACK	5
609	3	ACE TENNIS NET	1

Customers | Orders

Applet com.oracle.den Local intranet 100%

13 Future Ideas and Improvements

The concept behind this step-by-step is still evolving and it is a lot of work to keep the step-by-step guide 100% up-to-date. Here is a list of things that are not yet incorporated in this step-by-step guide:

1. The detail JSF Table showing the order lines also shows the Order ID and the Item ID which are redundant. The other two columns could have better titles.
2. Show how to setup a custom JSF component to include the HTML for the Forms applet
3. Show how to create a WebCenter Portlet to embed Oracle Forms.
4. Find a way to use a fixed userid/password without exposing it to the client
5. Both the <EMBED> and <OBJECT> tag use the id "formsapplet". An ID should be unique and this gives problems in Firefox. Currently you will have to change the ID of the <OBJECT> to make the samples work with Firefox.
6. How to handle multi-form applications, like Designer/Headstart generated LOV Forms
7. See if and how it works with multiple browsers open (both same and separate sessions). Doesn't this cause problems with the legacy lifecycle cache?
8. Do not use "com.oracle.demo", but "com.example" instead.