

Developing an ADF 11g client for Agile PLM

Table of Contents

1	LAB OVERVIEW	3
2	GETTING STARTED	4
	2.1 Starting Oracle JDeveloper 11gR1.....	4
3	CREATE THE ADF CLIENT	5
	3.1 Create the ADF Application.....	5
	3.2 Create the Web Service Data Control.....	8
	3.3 Create the JSF Page.....	11
4	TESTING	19
5	APPENDIX: USING HARDCODED VALUES	21

1 Lab Overview

This lab shows how to develop an ADF 11g based UI for Agile PLM.

Agile 9.3 exposes several key PLM functionalities as web services, which allow easy integration with existing applications. The core web services enable access to Agile Business Objects, Collaboration, Metadata, Attachments, Search, Tables, Product Collaboration (PC) and Engineering Collaboration (EC) services for integration with existing ERP, CRM and SCM applications.

The Web Service Data Control in ADF 11g provides an easy and convenient interface to incorporate web services in an ADF application. The data control abstracts the implementation of a business service (such as a web service), thus providing a consistent mechanism for the ADF pages to access the data.

Additional information:

ADF 11g

<http://www.oracle.com/technology/products/adf/index.html>

Agile and Fusion Middleware Best Practice Center on Oracle Technology Network:

<http://www.oracle.com/technology/tech/fmw4apps/agile>

In this exercise:

1. Create an ADF 11g application that uses the Agile WSDL to create a web service data control.
2. Design a JSF page and integrate the web service data control that was created in step 1.
3. Deploy and test the page on the integrated WebLogic Server.

Software Used:

- Agile PLM 9.3
- Oracle JDeveloper and ADF 11g (available for download from [OTN](#))

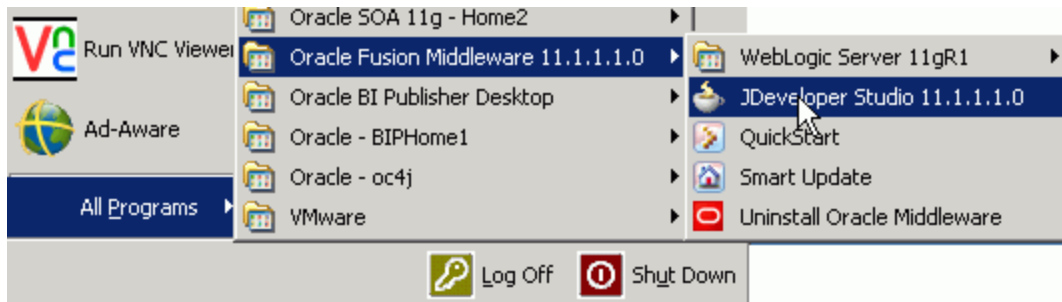
Questions:

- Srikant Subramaniam srikant.subramaniam@oracle.com

2 Getting Started

2.1 Starting Oracle JDeveloper 11gR1

1. Select **Start > Oracle Fusion Middleware 11.1.1.1.0 > JDeveloper Studio 11.1.1.1.0**.



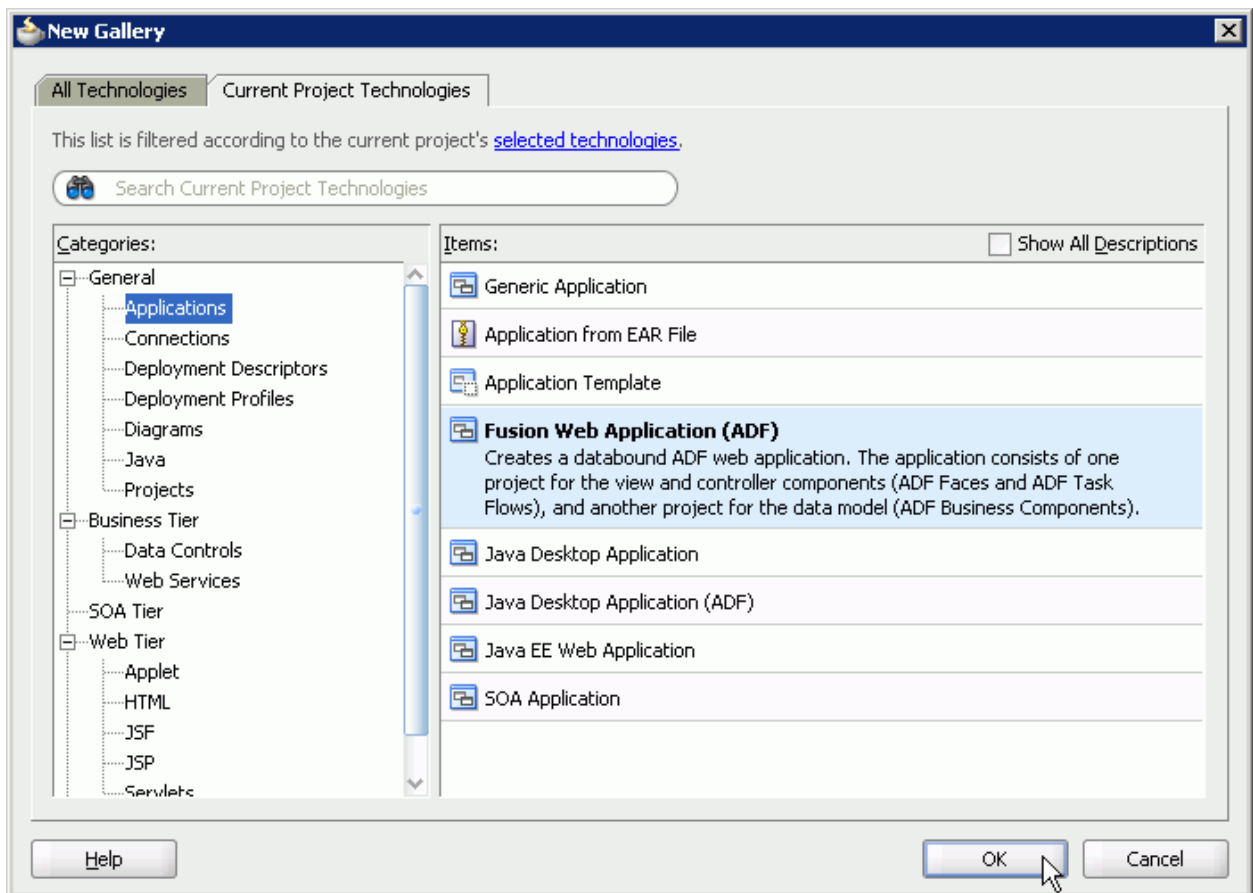
3 Create the ADF Client

3.1 Create the ADF Application

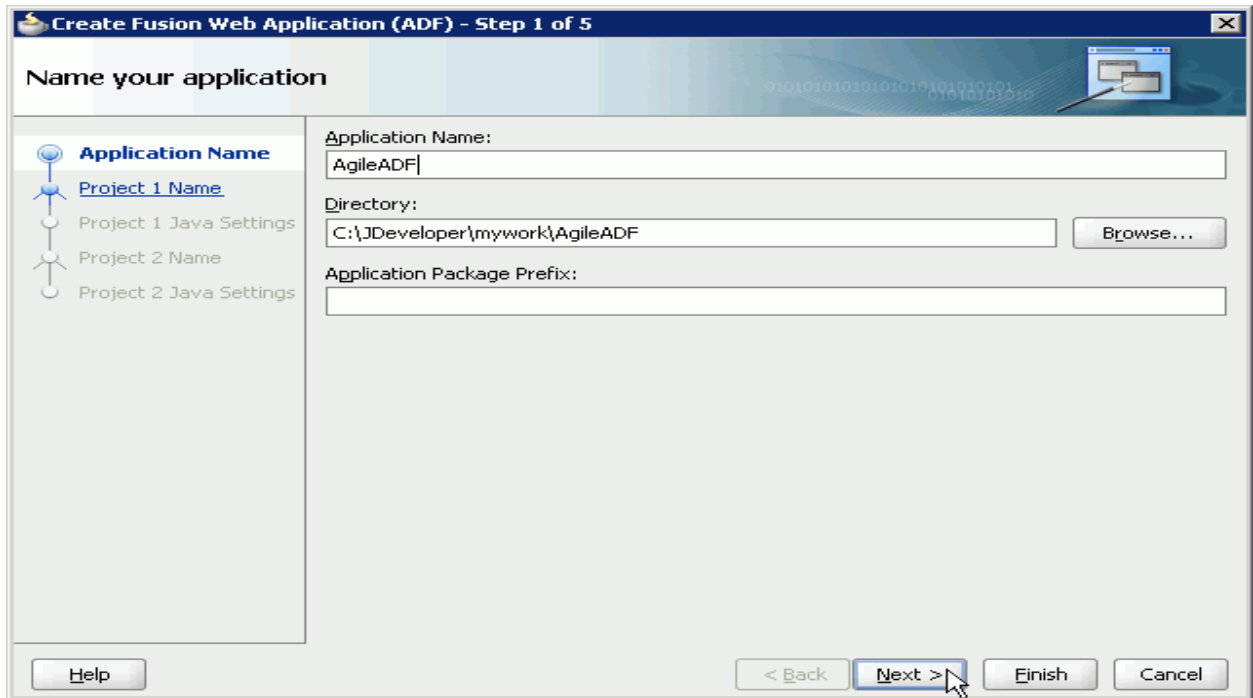
1. Choose **File > New**.



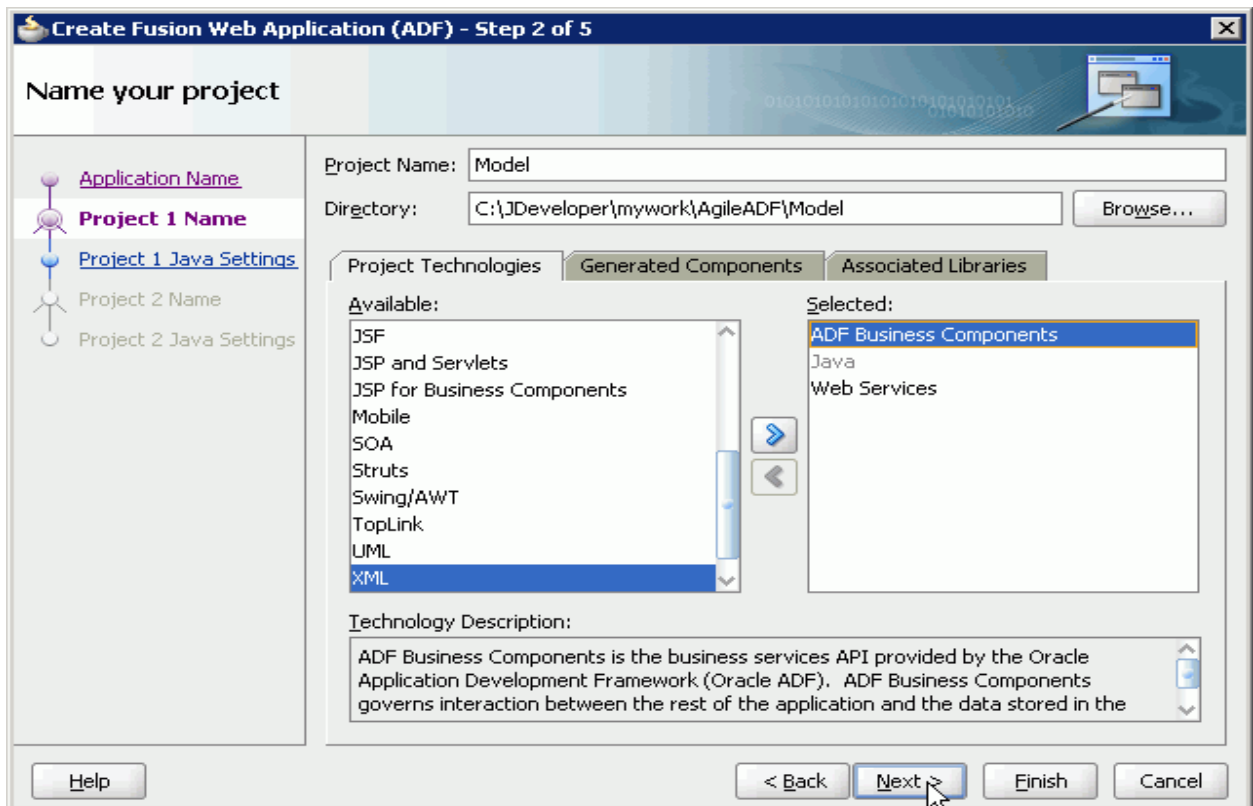
2. In Categories, select **Applications** and select **Fusion Web Application (ADF)** in Items.
3. Click **OK**.



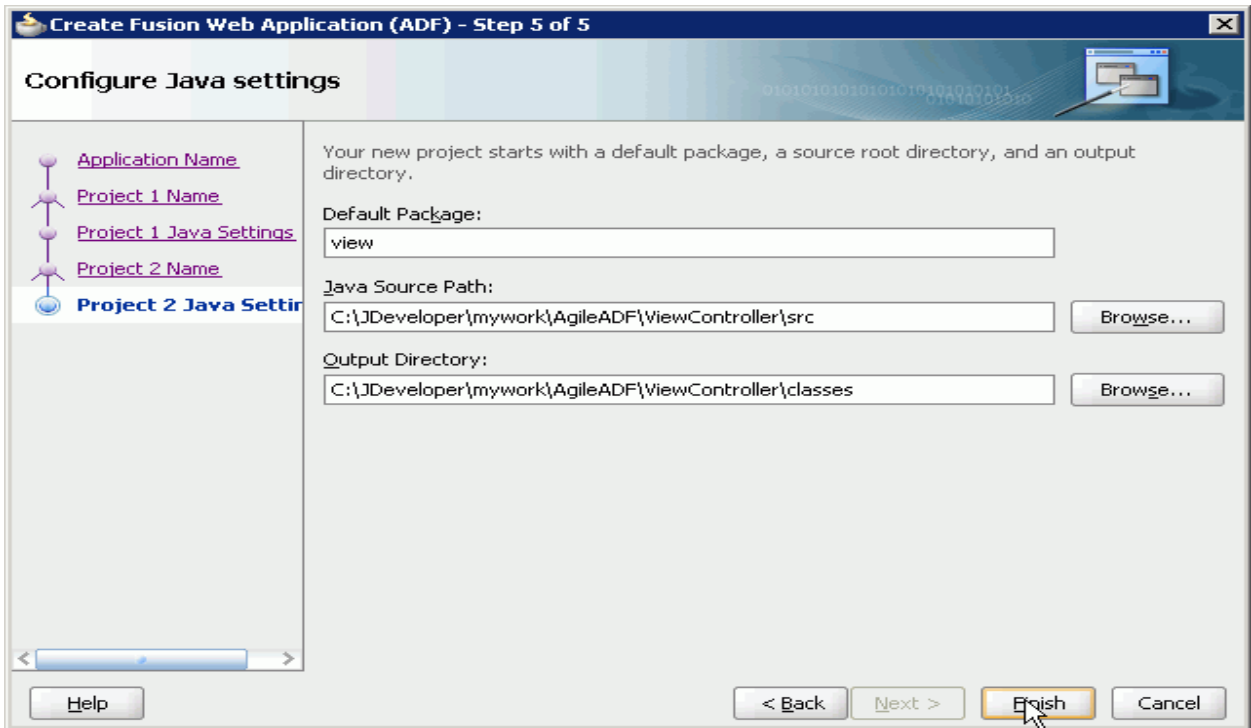
- In Application Name field, type **AgileADF** and click **Next**.



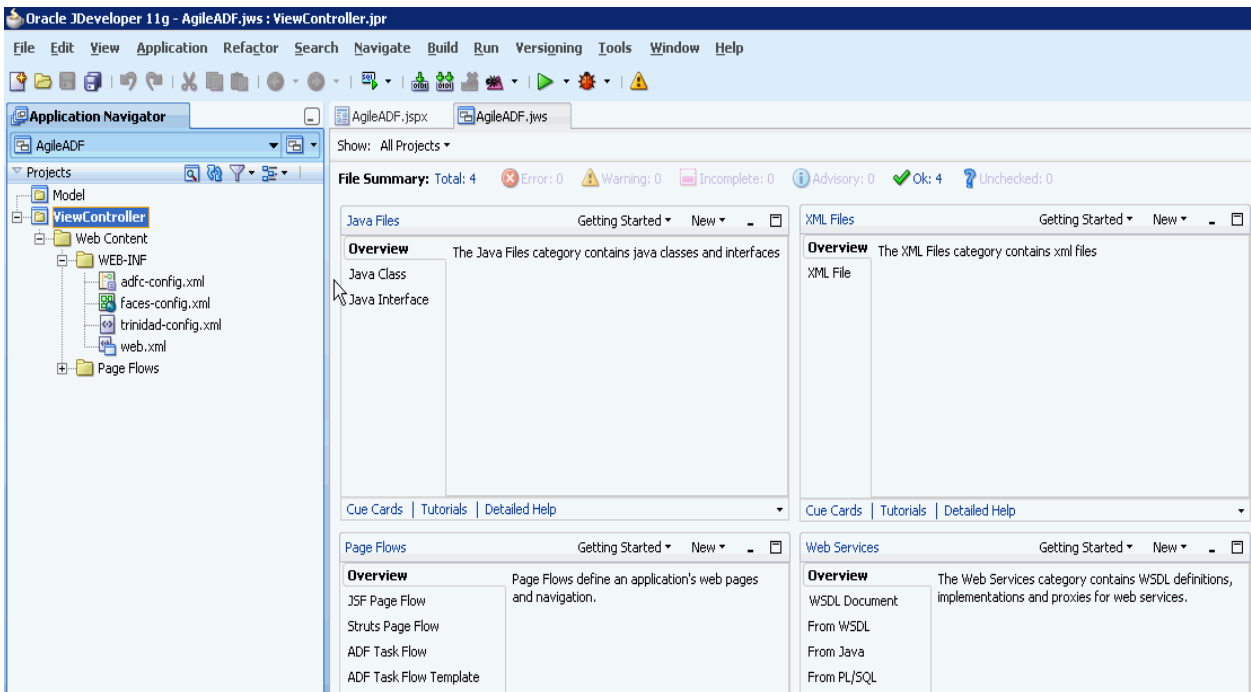
- In Project Technologies > Available, select **Web Services** and move it to the **Selected** box. Ensure your screen looks as follows and click **Next**.



6. Accept default settings and click **Next** to advance through Steps 3 and 4.
7. Click **Finish**.

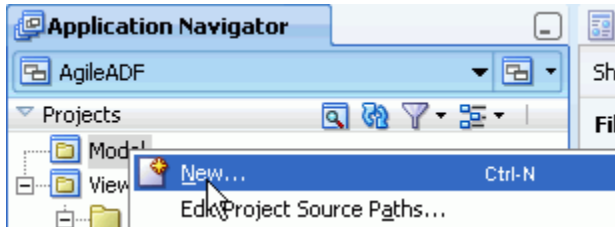


The **AgileADF** application is created with two Projects (Model and ViewController).

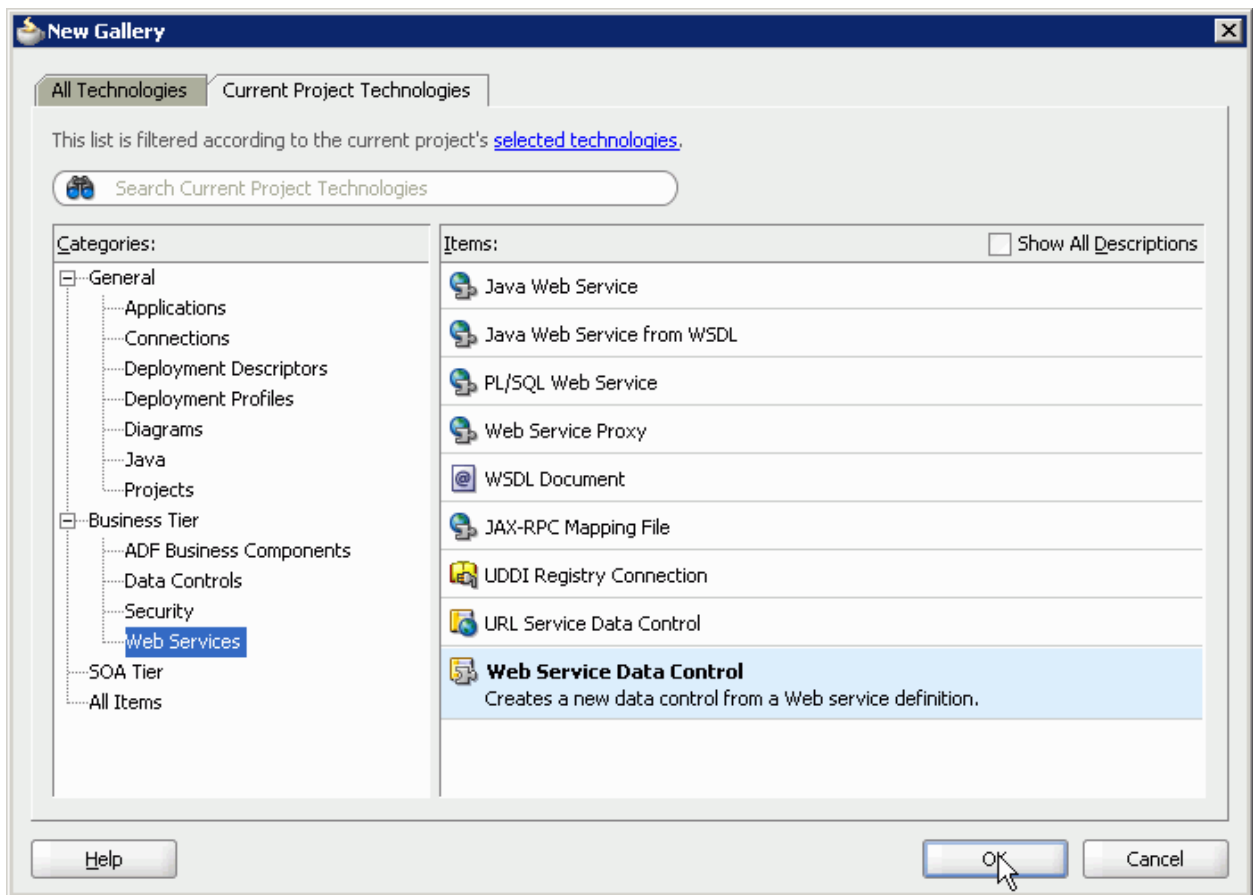


3.2 Create the Web Service Data Control

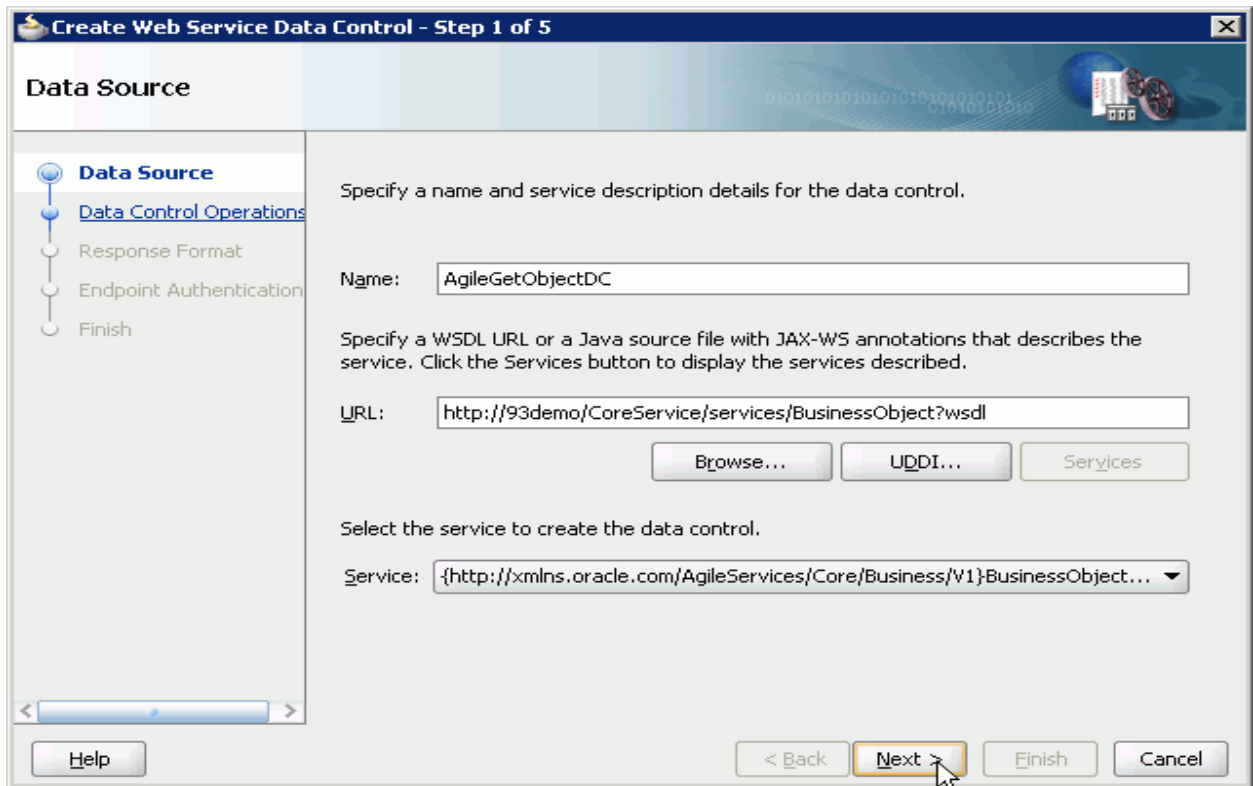
1. In the Application Navigator, right click on the Model project and select **New**.



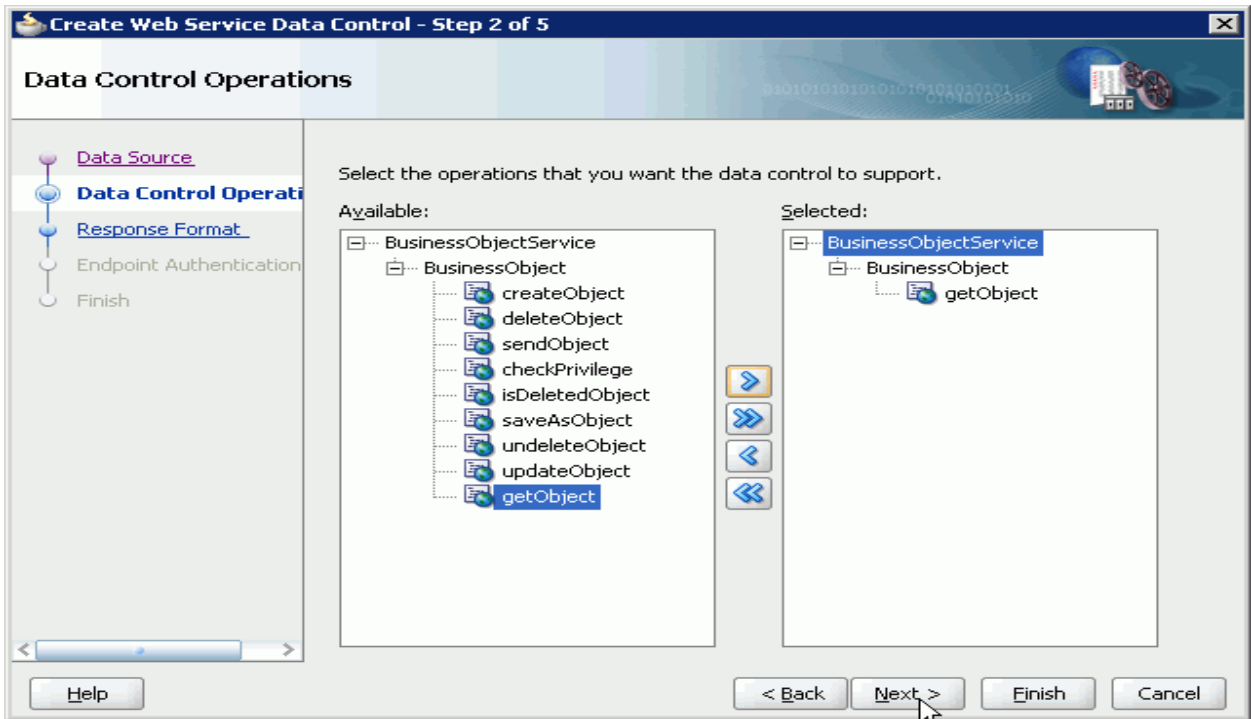
2. In New Gallery, select **Business Tier > Web Services > Web Service Data Control** and then click **OK**.



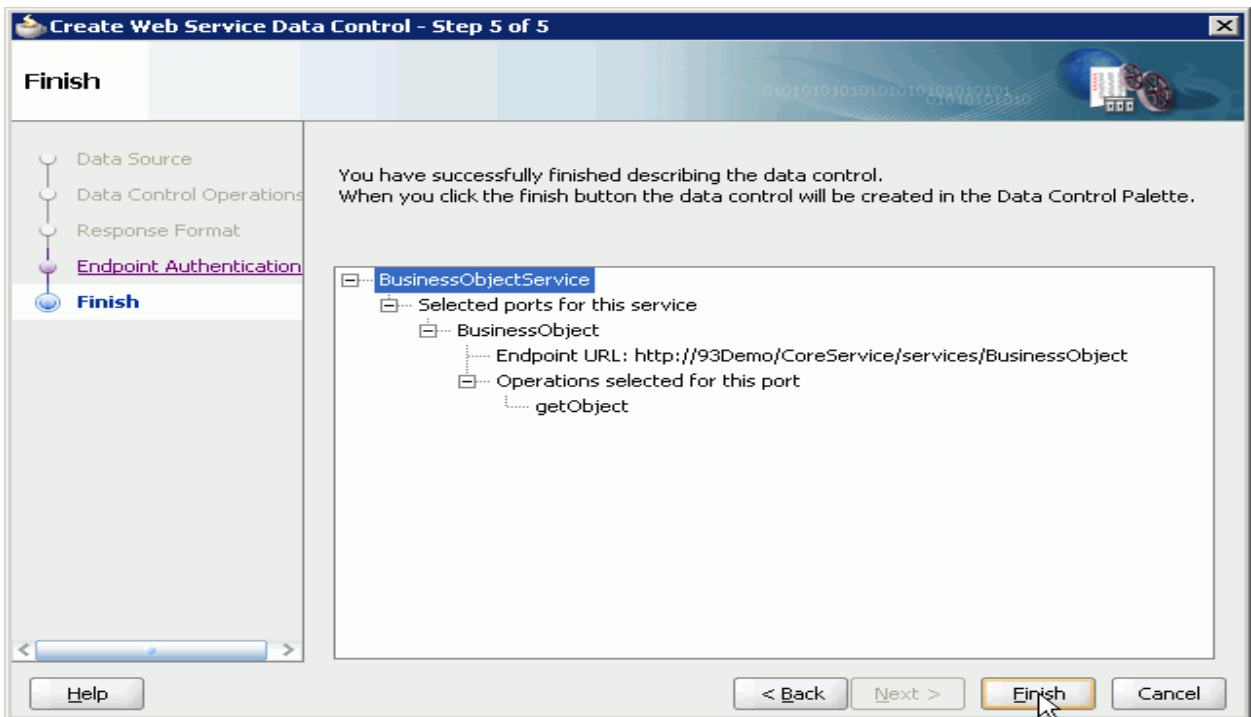
3. In Create Web Service Data Control wizard, select **Data Source** and do as follows:
 - a. In **Name** field, type **AgileGetObjectDC**.
 - b. In **URL** field, type the URL for your Agile web services WSDL.
This example uses: <http://93demo/CoreService/services/BusinessObject?wsdl>
 - c. Click **Next**.



- In **Data Control Operations** page, move the **getObject** web service method from the **Available** column to the **Selected** column and then click **Next**.



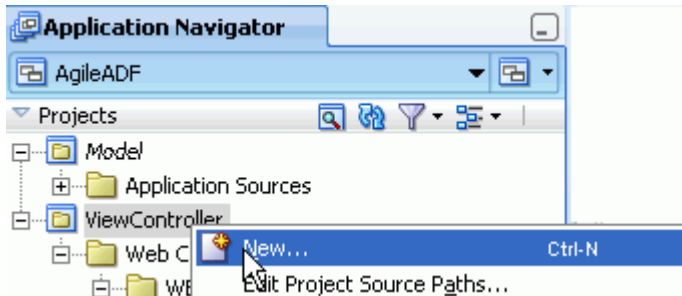
- Accept default settings and click **Next** to advance through Steps 3 and 4. Click **Finish**.



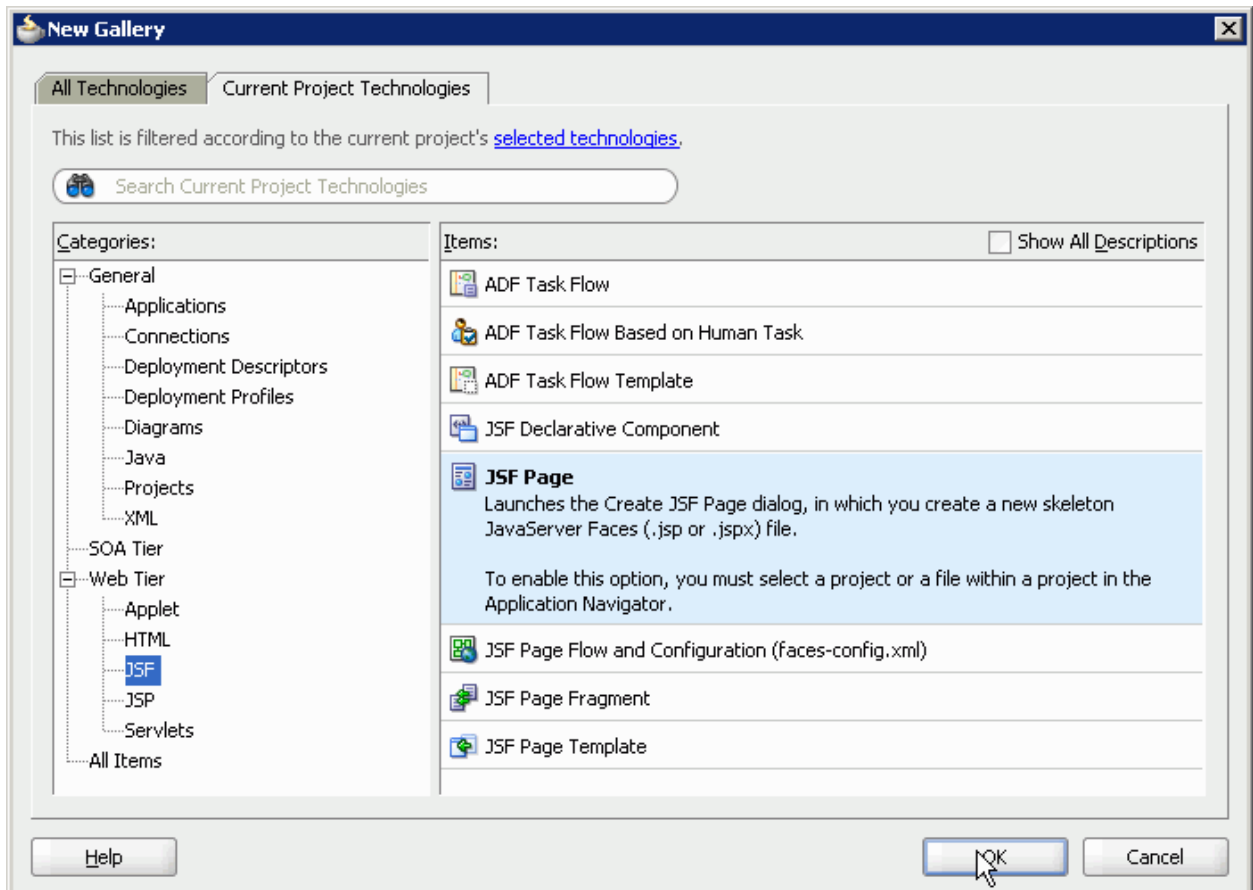
- File > Save All** to save your work.

3.3 Create the JSF Page

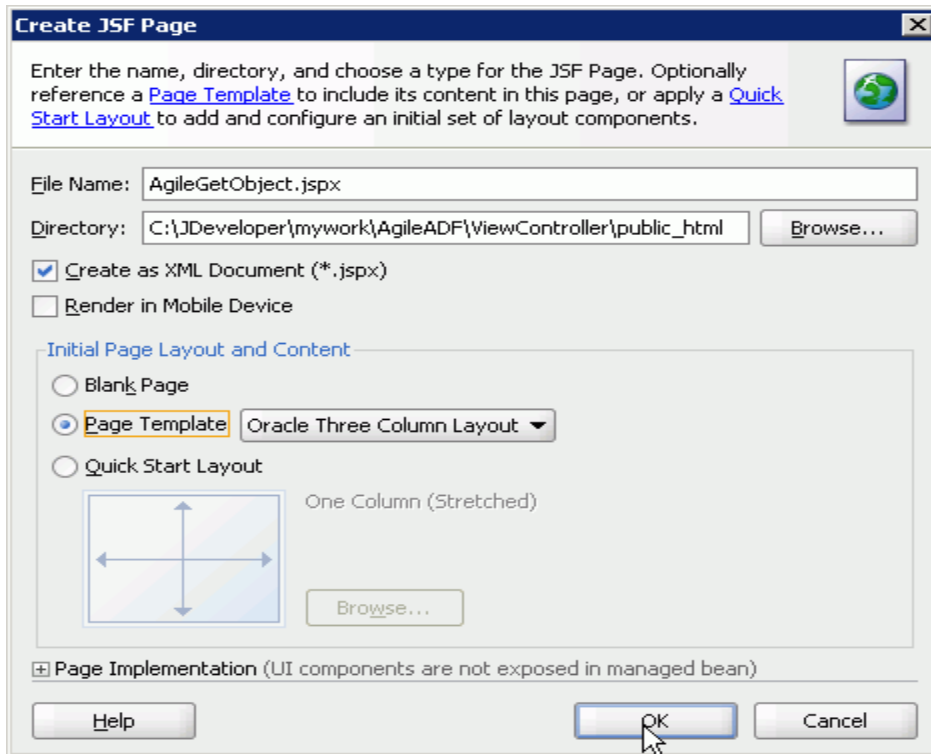
1. In the Application Navigator, right click on the **ViewController** project and then select **New**.



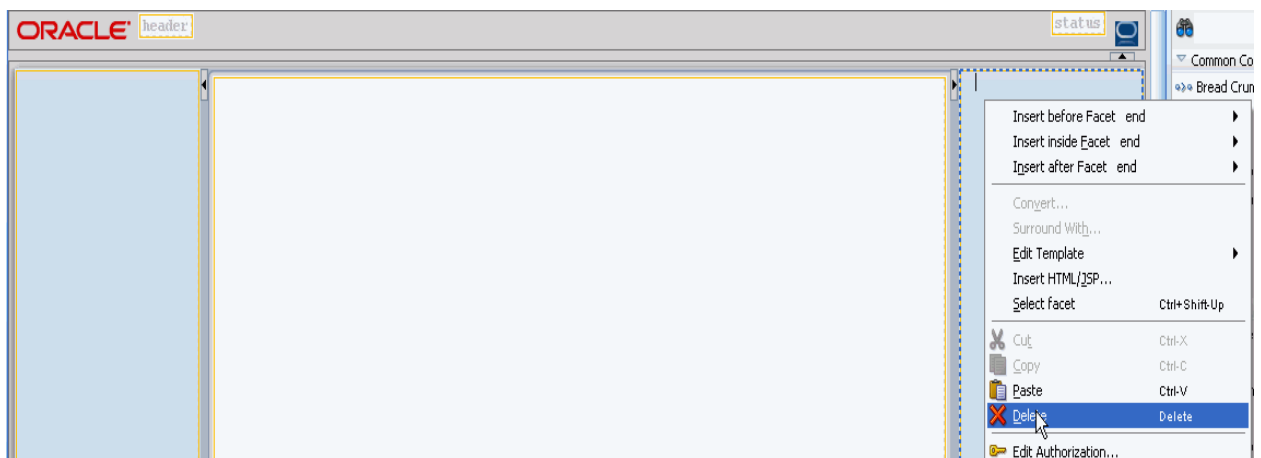
2. In Categories column, select **Web Tier > JSF**.
3. In Items column, select **JSF Page** and then click **OK**.



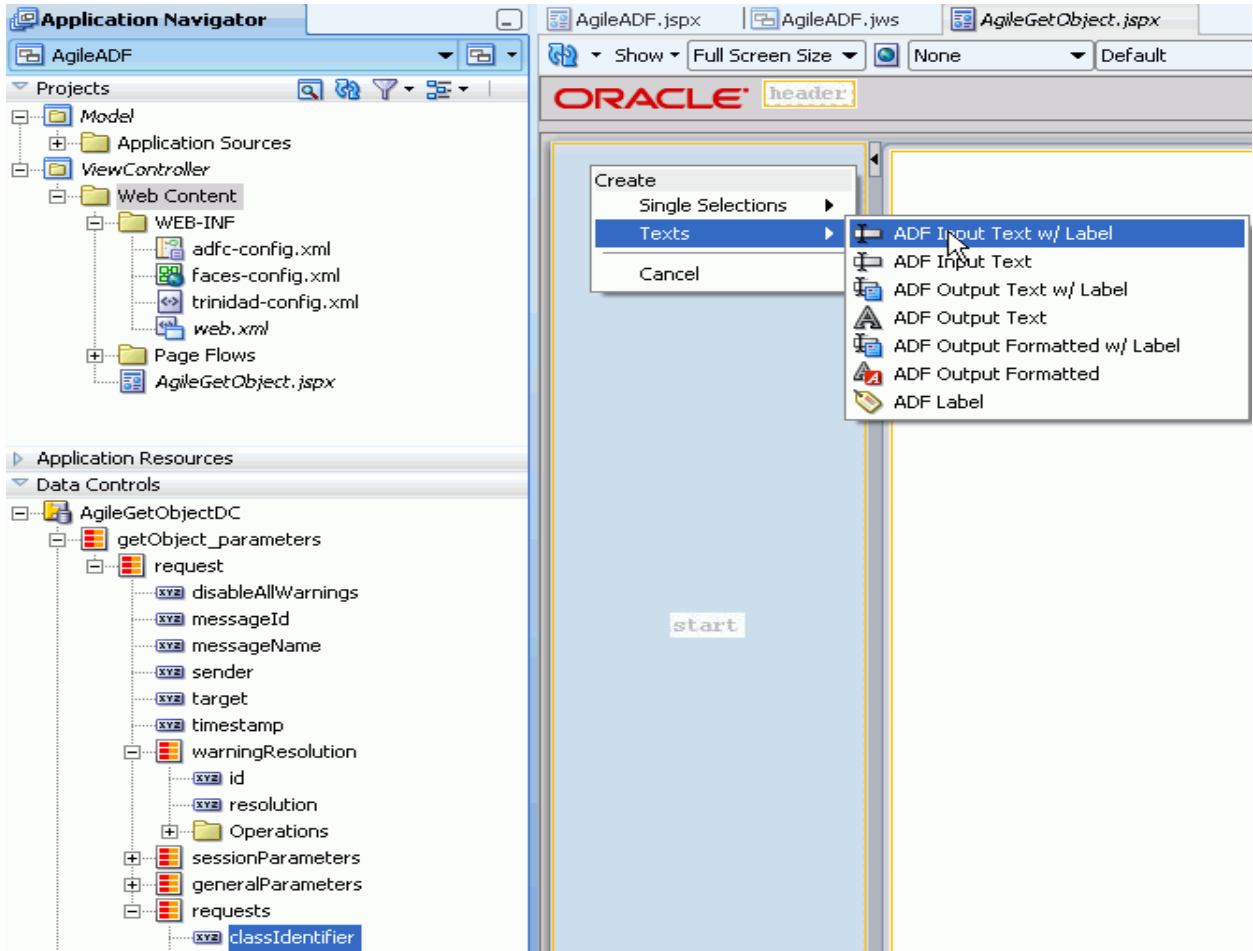
4. In Create JSF Page dialog do as follows:
 - a. In **File Name** field, type **AgileGetObject.jspx**,
 - b. Mark the check box next to **Create as XML document**
 - c. In **Page Template** drop down list, select **Oracle Three Column Layout**
 - d. Click **OK**.



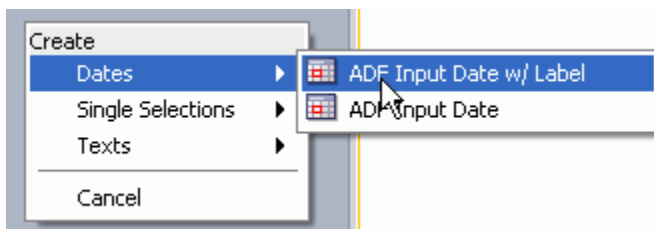
5. The new page opens up in the Editor. In the right most column, right click and select **Delete**.



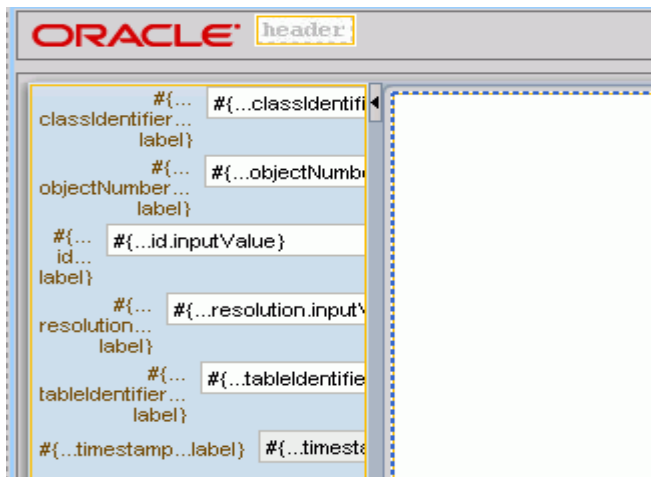
6. In the left pane, expand the **AgileGetObjectDC** node under Data Controls.
7. Drag and drop the **classIdentifier** node to the first column of the page on the right.
8. Choose **Texts > ADF Input Text w/Label** from the menu.



9. Repeat steps 7-8 for the following nodes:
 - objectNumber (requests → objectNumber)
 - id (warningResolution → id)
 - resolution (warningResolution → resolution)
 - tableIdentifier (requests → tableRequests → tableIdentifier)
10. Drag and drop the **timestamp** node (request → timestamp) to the first column of the page on the right and select **Dates > ADF Input Date w/Label** from the displayed menu.

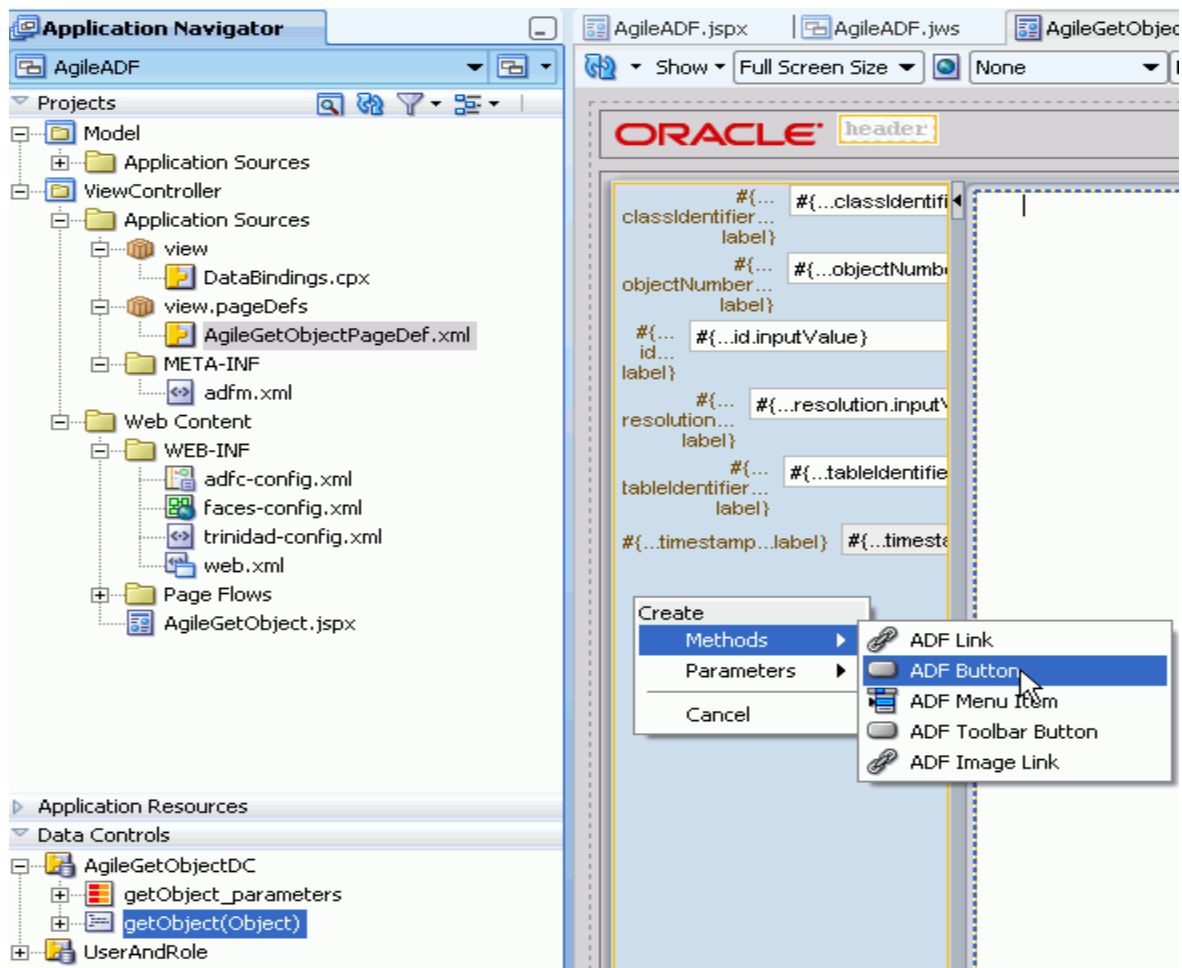


11. Ensure that page contains all the input fields shown in the following illustration.

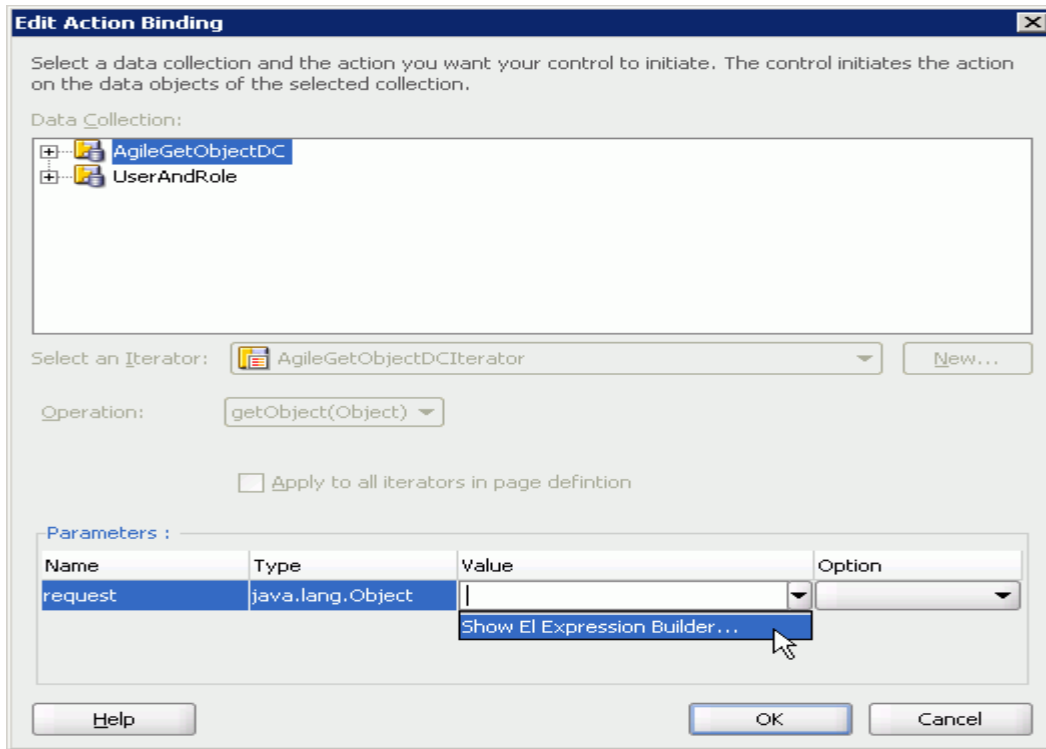


12. Drag and drop the **getObject(Object)** node under Data Controls to the first column of the page on the right (under the previously created fields).

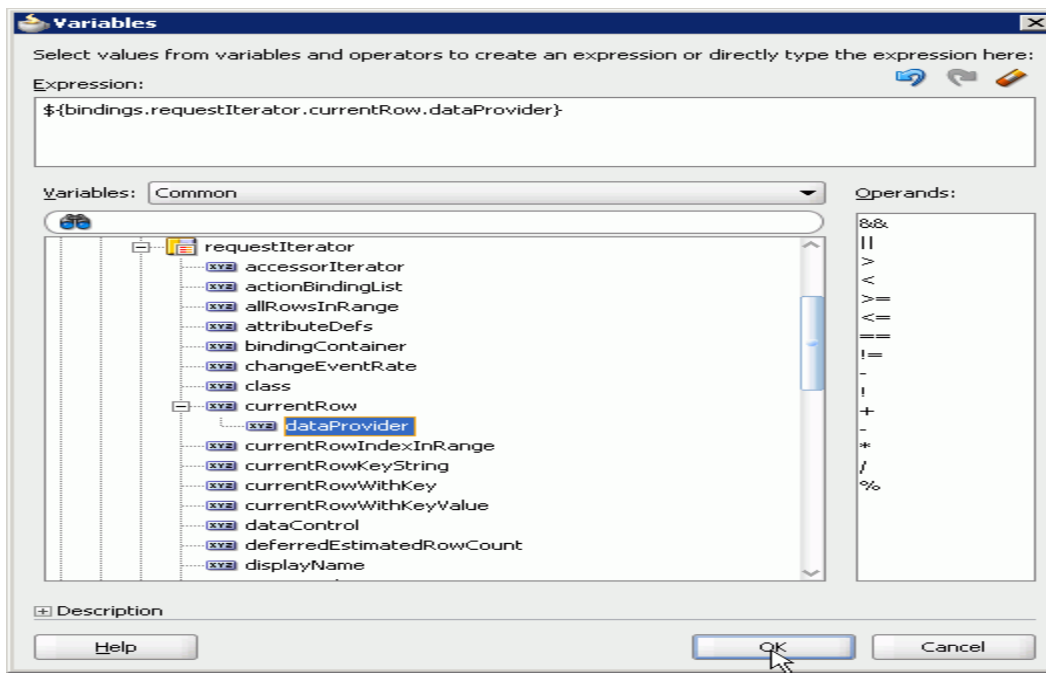
13. Select **Methods > ADF Button** in the drop down menu.



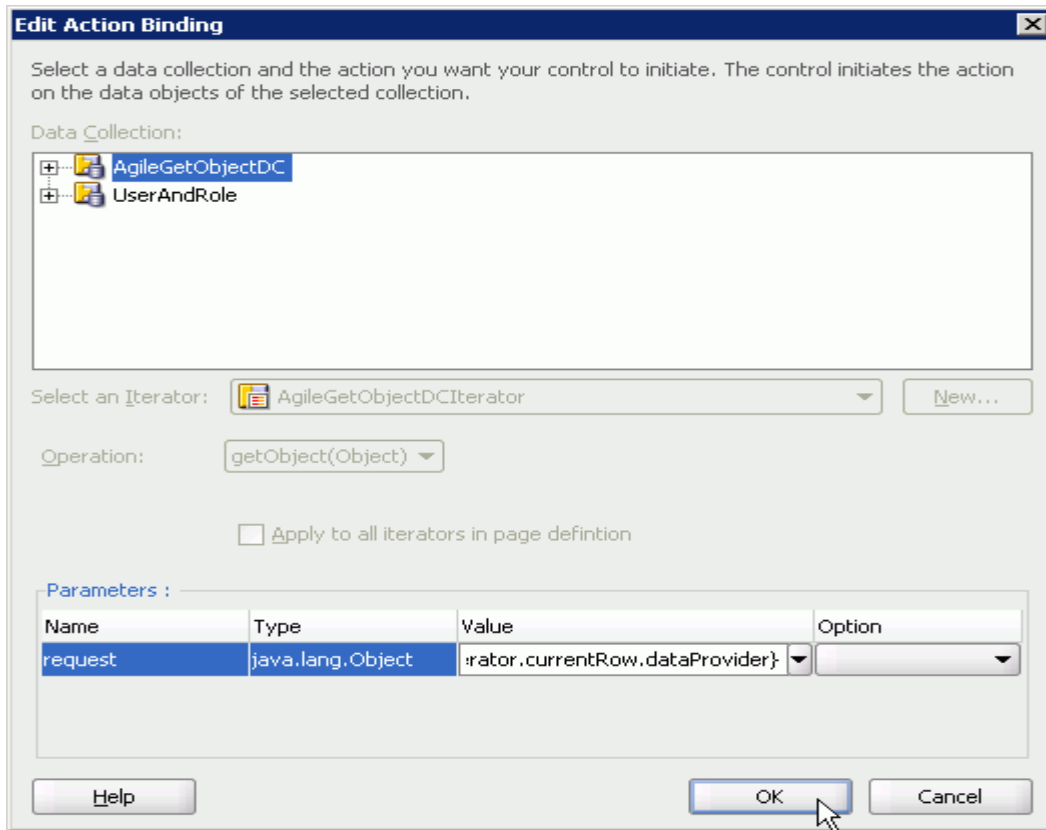
14. In Edit Action Binding dialog, select **EI Expression Builder** from the **Value** drop down list.



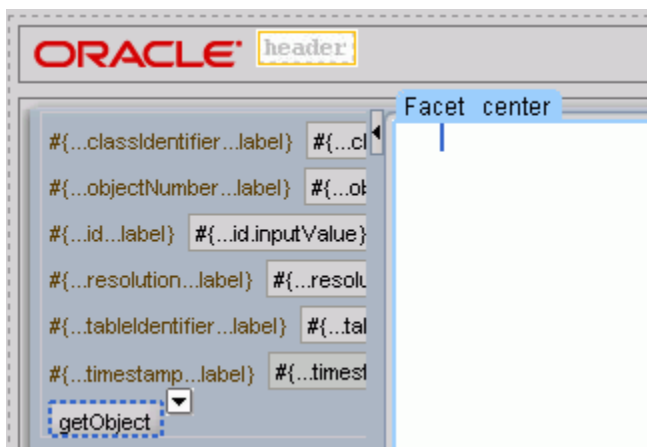
15. Navigate to **ADFBindings** → **bindings** → **requestIterator** → **currentRow** and choose **dataProvider**.
16. Click on **OK**.



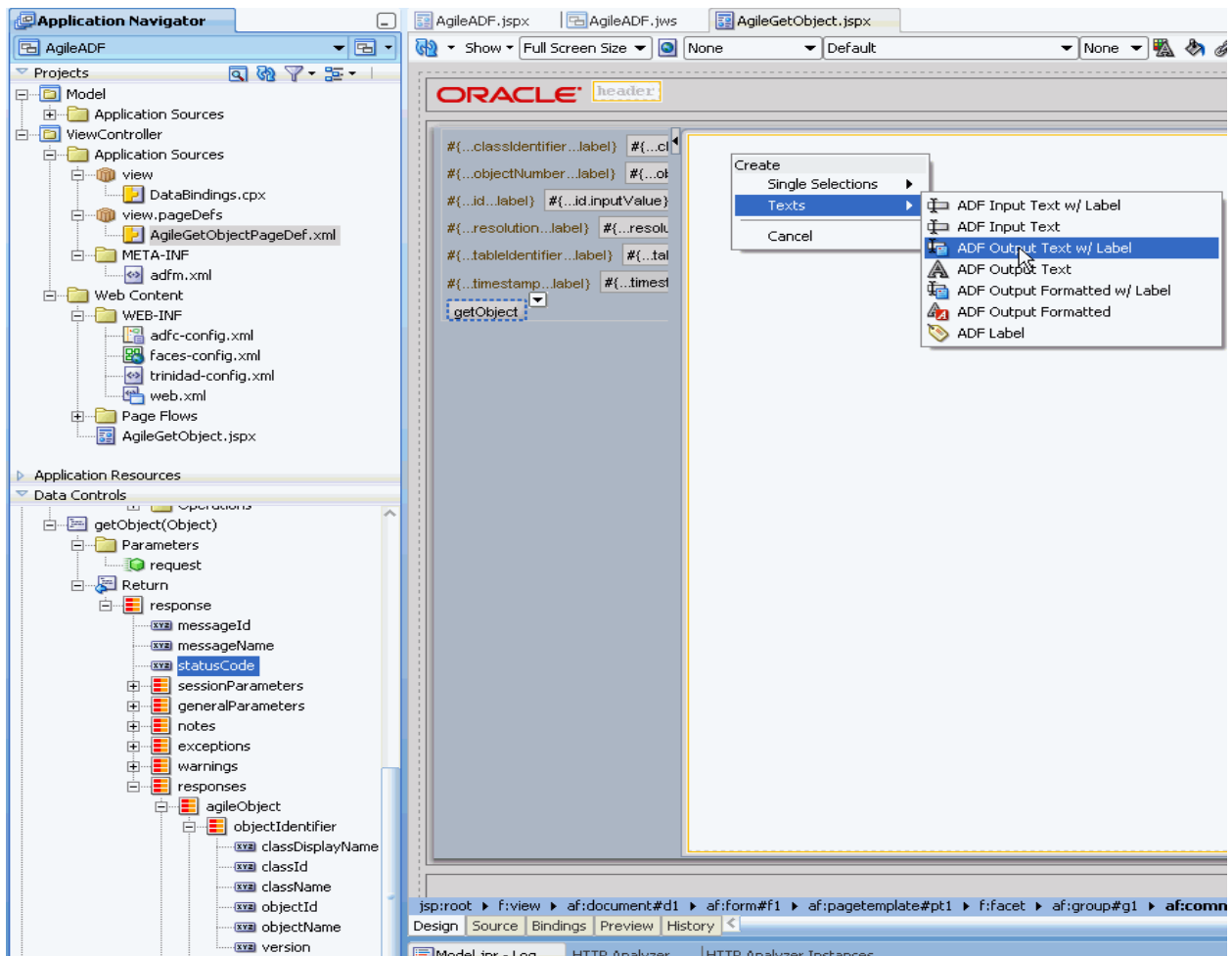
17. Click **OK**.



The following page appears. Note the recently added **getObject** button.

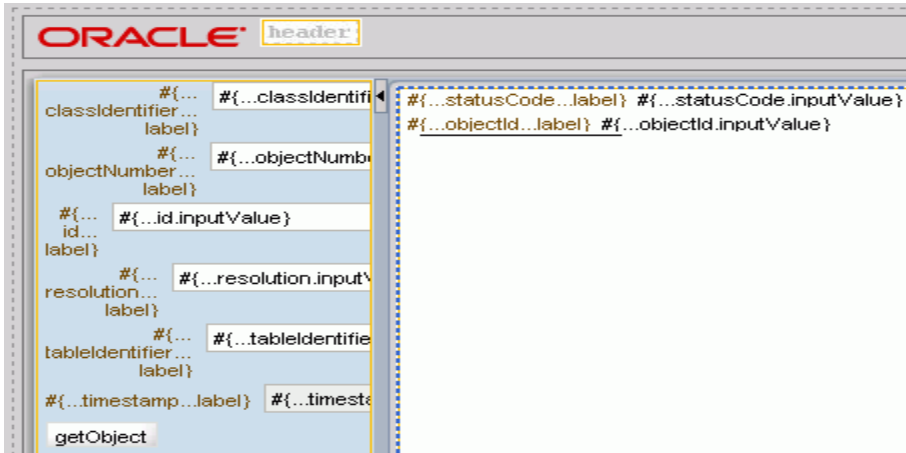


18. Expand the **getObject(Object)** → **Return** node in the Data Controls palette on the left as shown.
19. Drag and drop the **statusCode** node to the second column of the page on the right. Choose **Texts > ADF Output Text w/Label** from the displayed menu.



20. Repeat above step for the **objectId** node (Return → response → responses → agileObject → objectIdentifier → objectId).

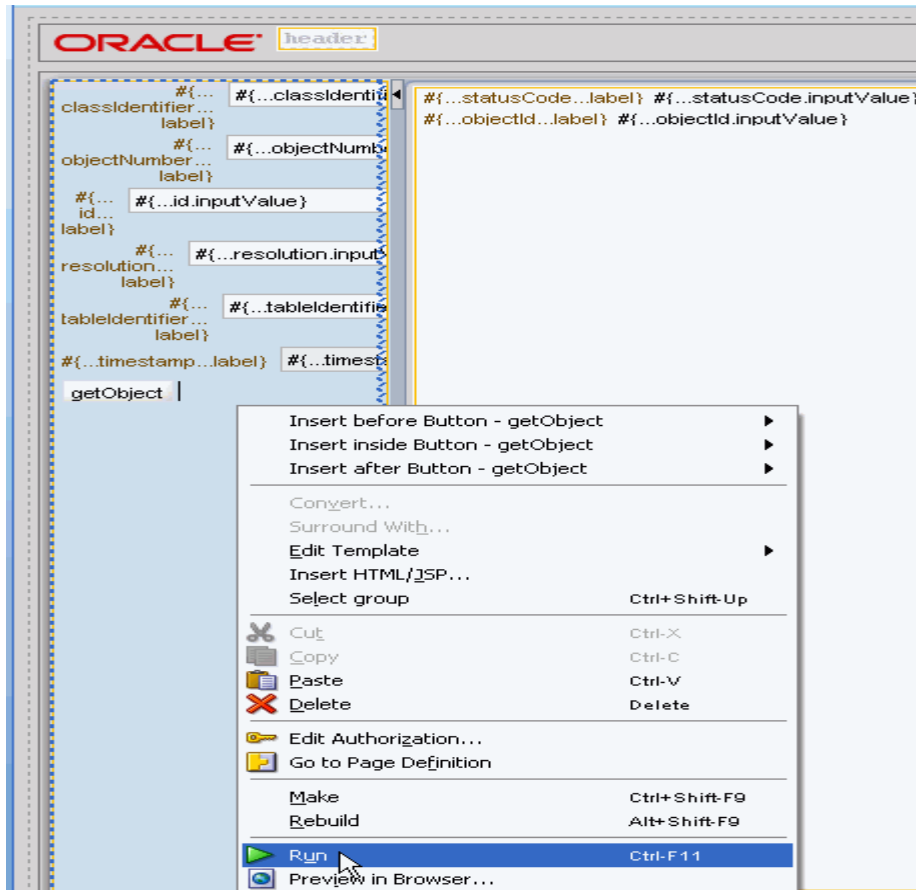
This completes the JSF page creation and integration with the web service data control. Your page should include the following input and output fields.



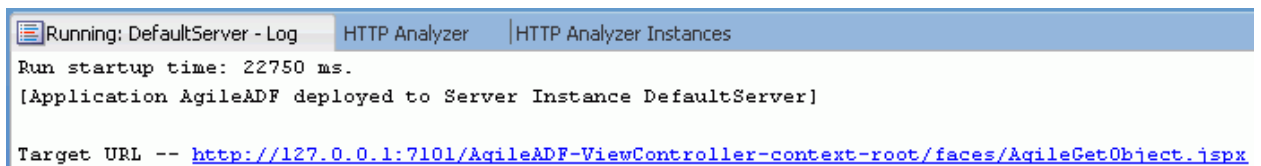
21. **File > Save All.**

4 Testing

1. Right click on the page and select **Run**.



This starts the embedded WebLogic Server within JDeveloper to deploy the application. You can monitor the progress at the bottom of the screen (in the Running: DefaultServer - Log tab). Deployment is complete when the Target URL is displayed as shown below.



Upon successful deployment, the page will automatically load into a browser window. If this doesn't happen, click on the Target URL (shown above) to view the page.

2. Type the input data as shown below (use the calendar icon to select a date for the timestamp field).
3. Click on **getObject**.

ORACLE

classIdentifier

* objectNumber

id

resolution

tableIdentifier

timestamp

The results from the web service call are displayed to the right, indicating a successful call to obtain the objectId.

ORACLE

classIdentifier

* objectNumber

id

resolution

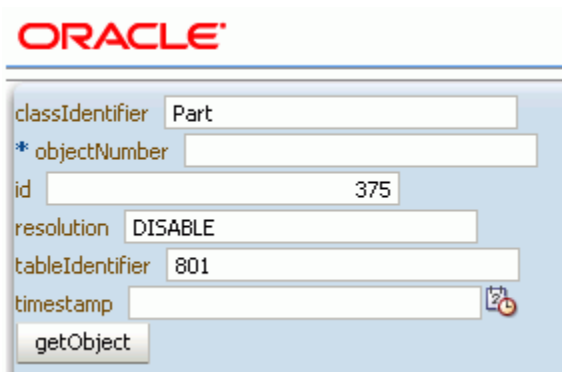
tableIdentifier

timestamp

statusCode SUCCESS
objectId 6164358

5 Appendix: Using hardcoded values

Using hardcoded values for some of the input fields can be especially optimal if the underlying web service invocation requires a number of inputs, but many of them have static values. In this tutorial, since **objectNumber** and **timestamp** are the only two fields that vary, we can hardcode the other inputs as shown and eliminate the need for the user to input the static values for every invocation.

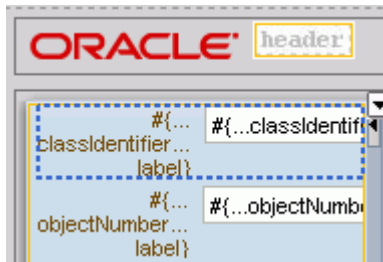


The screenshot shows an Oracle ADF web form with the following fields and values:

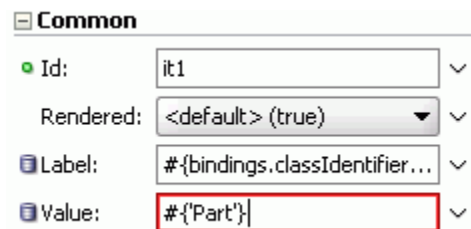
- classIdentifier: Part
- * objectNumber: (empty)
- id: 375
- resolution: DISABLE
- tableIdentifier: 801
- timestamp: (empty)
- getObject: (button)

Implementation

1. Follow the steps in the tutorial, up to and including Chapter 3.
2. Click on the classIdentifier label in the design window to bring up the property inspector.



3. Enter `#{'Part'}` in the Value field as shown.



The screenshot shows the Oracle ADF property inspector for the classIdentifier label. The Value field is highlighted with a red box and contains the value `#{'Part'}`.

4. Repeat steps 2 and 3 for the other fields (**id**, **resolution**, **tableIdentifier**) and assign the individual values as shown.

```
#{...classIdentifier...label} Part
    #{...
objectNumber...
    label}
#{...id...label} 375
#{...resolution...label} DISABLE
#{...tableIdentifier...label} 801
#{...timestamp...label} #{...timesta
getObject
```

5. **File > Save All** and continue with Section 4 in the tutorial to test. The only inputs required are **objectNumber** and **timestamp**.