CIS 764 Tutorial
By Vamsee Raja Jarugula.

Title: Developing Contract Driven Web Services using JDeveloper.

Web Link: http://www.oracle.com/technology/obe/obe1013jdev/10131/10131_wstopdown/wstopdown.htm

In this tutorial we use JDeveloper to develop a Java web Service using a top-down approach.

The following screen shots demonstrate as to how this tutorial is implemented.

In the Create Project dialog, enter the Project Name WSDLDocument and click OK.
In the New Gallery, expanding **General** in the Categories list and selecting **XML**; then from the Items list select **XML Schema**. Clicking **OK**
In the Create XML Schema dialog, enter a Name of **CreditRating.xsd**, then click **OK**.
You now begin to add elements to the schema. Elements are like variables that are global to the web service.
You can use the design editor to add elements.

In the design editor, click **exampleElement**, and type **ssn** as the name.
Right-click ssn and select **Set Type** from the context menu.
Click inside the element and overtype the default name with **rating**.
Creating the WSDL Contract Document

Next you create the WSDL contract document, and import the schema you created in the previous steps, into it.

1. Right-click the WSDLDocument project and choose **New** from the context menu. Expand the Business Tier node in the New Gallery, and choose **Web Services** in the Categories pane, and **WSDL Document** in the Items pane.
Notice that the CreditRating WSDL has been created and opened in the editor. Ensure that you are in the Design tab for the editor window and the Structure window. If not, click the Design tab at the bottom left in each window. The windows should look like the screenshot below.
In the Insert XML Schema Item dialog, choose `schema` and click OK.
Ensure that the Properties Inspector window is visible. If it is not, choose View-->Property Inspector. In the PI, Specify CreditRating.xsd as the id and http://myCompany.com/CreditRating/types as the schema namespace.
Now that you can see the values in the Definitions tag more clearly, add the following line at the end of the tag:

```xml
xmlns:types="http://myCompany.com/CreditRating/types"
```
<definitions>
  <xs:schema id="creditRating.xsd"
    targetNamespace="http://myCompany.com/CreditRating"
    attributeFormDefault="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="http://www.w3.org/2001/XMLSchema"
    namespace="http://myCompany.com/CreditRating/types">
    <wsdl:types>
      <xsd:schema id="creditRating.xsd"
        targetNamespace="http://myCompany.com/CreditRating"
        attributeFormDefault="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="http://www.w3.org/2001/XMLSchema"
        namespace="http://myCompany.com/CreditRating/types"></xsd:schema>
    </wsdl:types>
  </xs:schema>
</definitions>
In the New Message dialog, enter the Message Name CreditRatingRequestMessage, then click OK.
In the New Part dialog, enter a Part Name of ssn and types:ssn as the Part Type.
In the New Part dialog, enter a Part Name of rating and types:rating as the Part Type.
In the New Part dialog, enter a Part Name of error and types: error as the Part Type.
In the New Port Type dialog, enter the Port Type Name of **CreditRating**, then click **OK**.
In the New Operation dialog, enter the Operation Name `processCreditRating`.

The Operation Type should be `Request Response`.

From the Input drop-down list, select `tns:CreditRatingRequestMessage`, which is one of the messages you defined earlier.

From the Output drop-down list, select `tns:CreditRatingResponseMessage`, which is another of the messages you defined earlier.

Select **Add Fault**.

Enter a Fault Name of `NegativeCredit`.

From the Fault drop-down list, select `tns:CreditRatingFaultMessage`, which is the third message you defined earlier.

Click **OK**.
In the New Binding dialog, enter a binding Name of CreditRatingSoapHttp. Leave all other fields as defaults. The Port Type defaults to the Port Type that you created. Click OK.
In the design editor, expand the binding that you just created to see the details.

Notice that when you click any element, the Design tab of the editor shows its relationship to other elements.
In the Insert Service dialog, enter a name of **CreditRatingService**, then click **OK**.

In the New Port dialog, enter a Port Name of **CreditRatingServiceSoapHttpPort**. The Binding Name defaults to the binding that you created. Click **OK**.
In the Insert Address dialog, for the location enter any string, such as tbd, because it is not necessary at this stage to specify where the finished service will run. Click OK.
In the Create Project dialog, enter a Project Name of Service, then click **OK**.
Right-click the Service project and select **New** from the context menu.

In the New Gallery, expand **Business Tier** and select **Web Services** in the Categories list, then select **Java Web Service from WSDL** in the Items list. Click **OK**.
Click **Next** on the Welcome Page of the Create J2EE 1.4 Java Web Service from WSDL wizard.

On the Web Service Description page of the wizard, from the WSDL Document URL drop-down list select the **CreditRating.wsdl** that you just created. Click **Next**.
By default, JDeveloper creates WSDLs with an endpoint that uses port 8888, because that is the default that would be used with an external server. Because you are testing the web service with the embedded (not external) OC4J, you should change the default port number to the port being used by the embedded OC4J server. To see what port that is, you can invoke Tools > Embedded OC4J Server Preferences, then click Startup in the tree at the left. The port number for the embedded server is shown in the HTTP field.