Part 3: Building a Web Service from an Existing WSDL

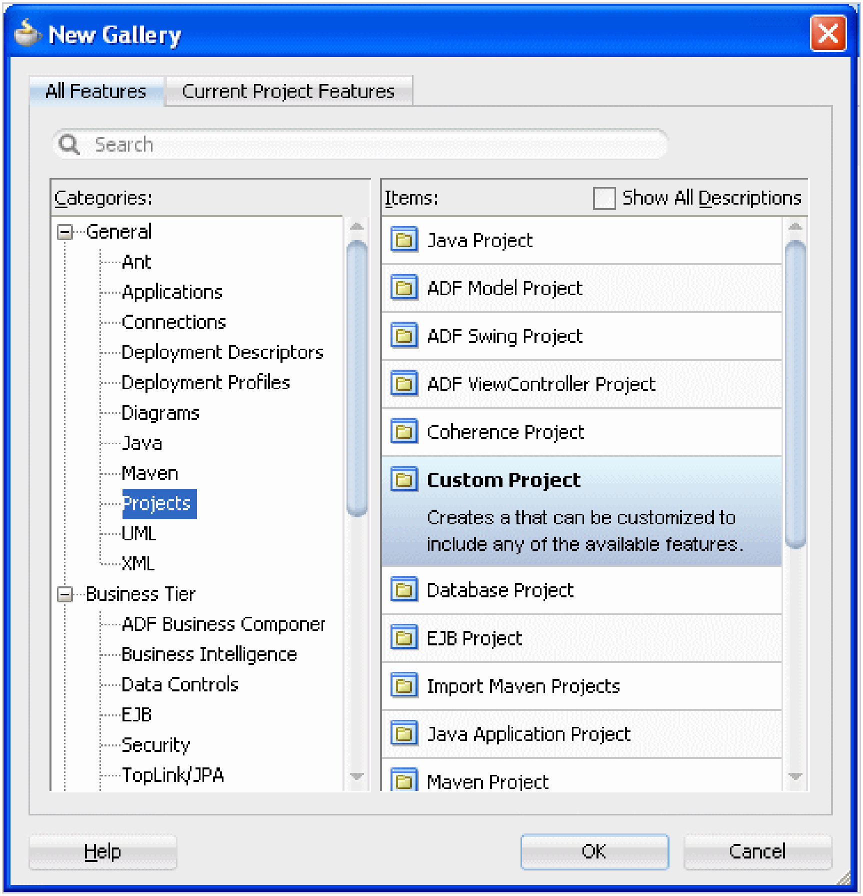
In this section you use the WSDL that was created for you in the last section as the source for a new web service.

A web service is described using Web Service Description Language (WSDL).

Step 1: Creating a Web Service from a WSDL

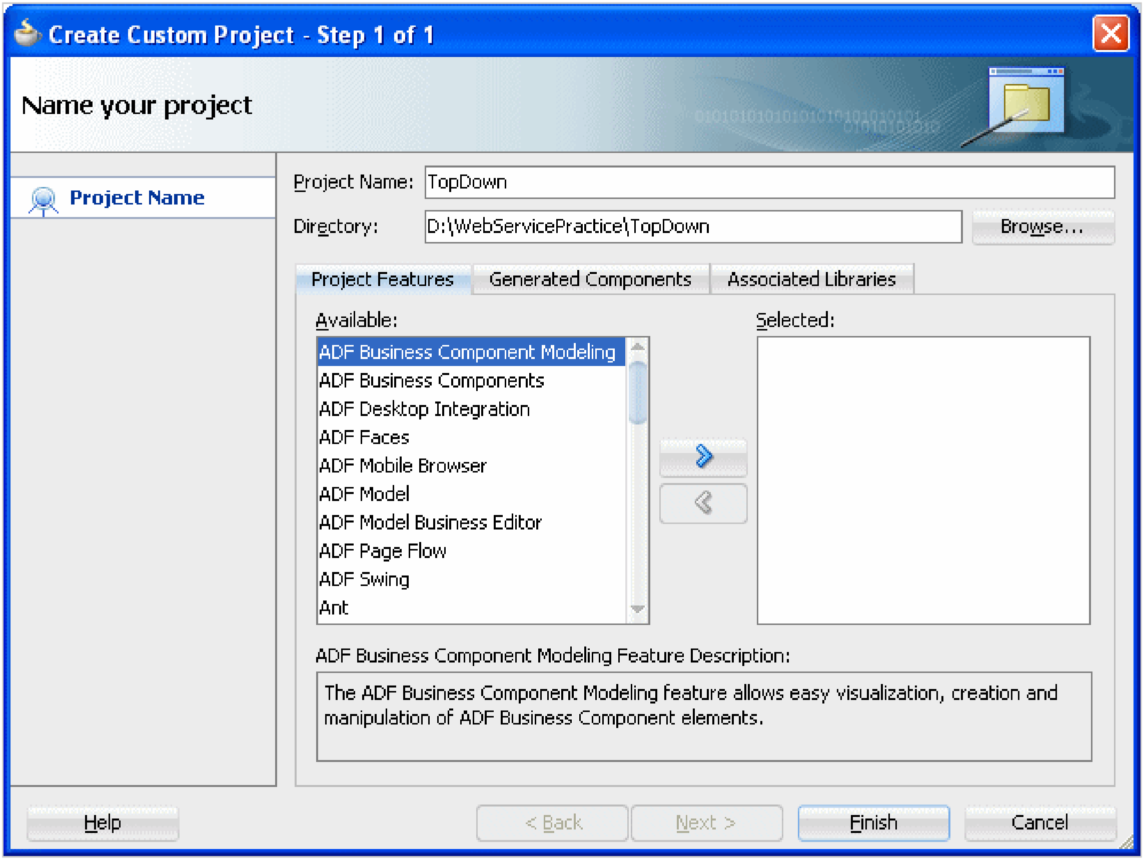
In the previous two sections you created web services using annotation and a wizard. In this section you create a web service from an existing WSDL file. For convenience, you will use the WSDL file you created in the last section of this tutorial.

1. Create a new empty project. Right-click the Annotation project node and select **New**. In the New Gallery, click the **All Features** tab and then select **General > Projects** in the Categories list and **Custom Project** in the Items list. Click **OK**.



Creates a that can be customized to include any of the available features.

2. Name the project **TopDown**, leaving other values at their defaults. Click **Finish**.



Create Custom Project - Step 1 of 1

Name your project

Project Name

Project Name: TopDown

Directory: D:\WebServicePractice\TopDown

Browse...

Project Features | Generated Components | Associated Libraries

Available:

- ADF Business Component Modeling
- ADF Business Components
- ADF Desktop Integration
- ADF Faces
- ADF Mobile Browser
- ADF Model
- ADF Model Business Editor
- ADF Page Flow
- ADF Swing
- Ant

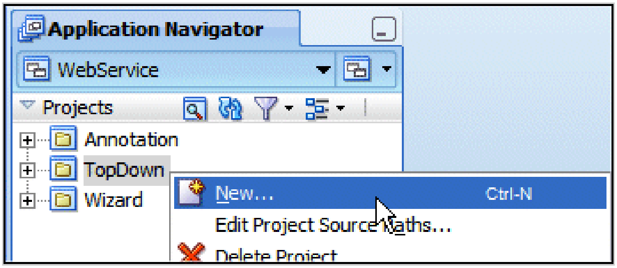
Selected:

ADF Business Component Modeling Feature Description:

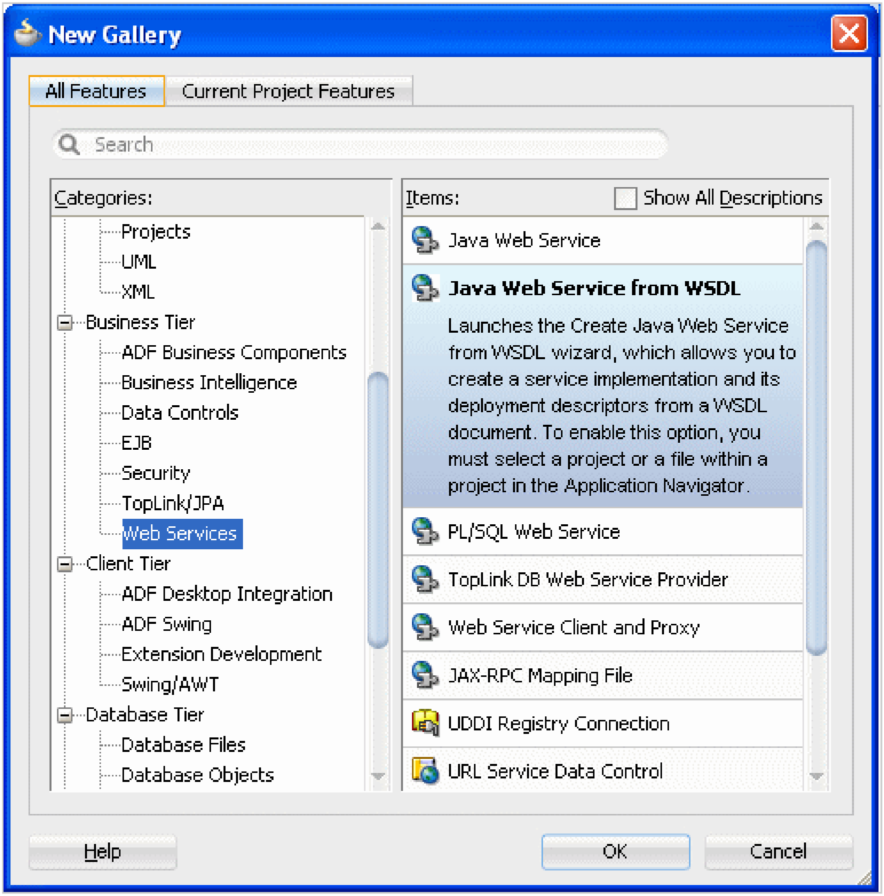
The ADF Business Component Modeling feature allows easy visualization, creation and manipulation of ADF Business Component elements.

Help | < Back | Next > | Finish | Cancel

3. Right-click the new **TopDown** project and select **New**.



4. In the New Gallery, select the **Business Tier | Web Services** node in the Categories list. Select the **Java Web Service from WSDL** item and click **OK**.

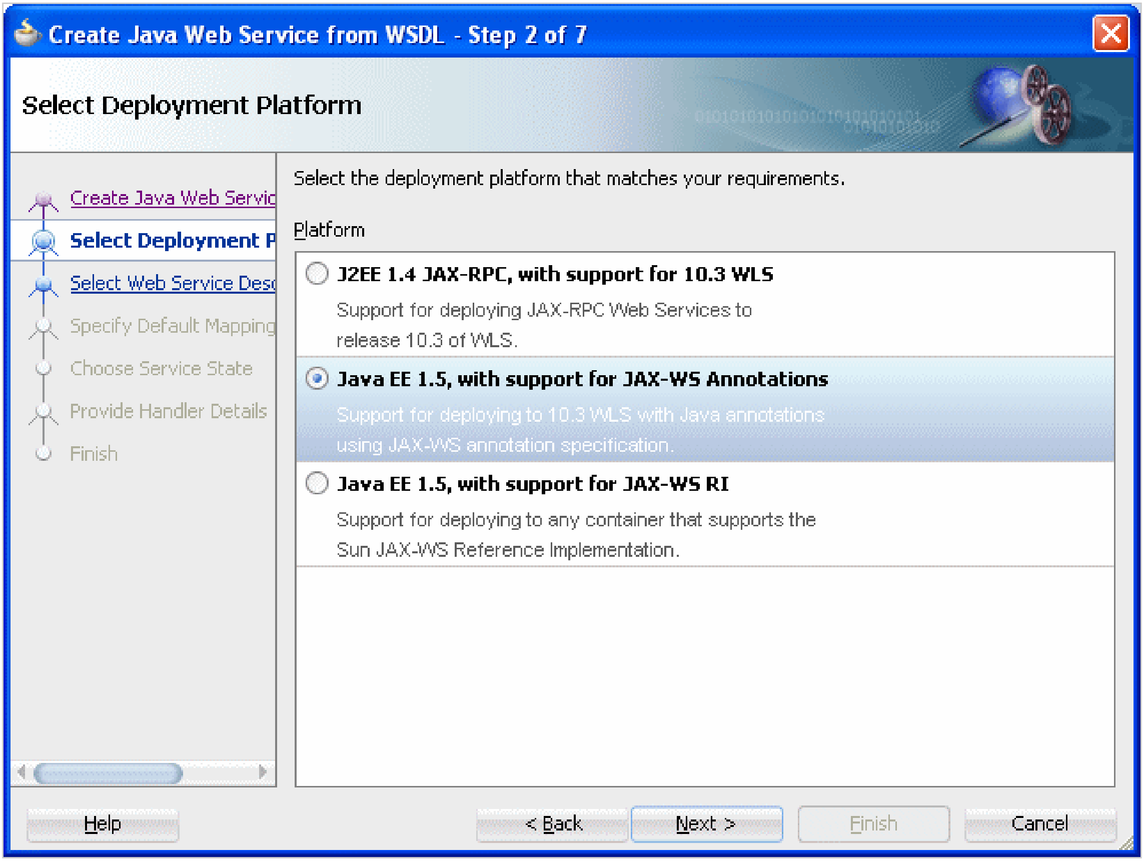


Java Web Service from WSDL

Launches the Create Java Web Service from WSDL wizard, which allows you to create a service implementation and its deployment descriptors from a WSDL document. To enable this option, you must select a project or a file within a project in the Application Navigator.

5. In the Create Java Web Service from WSDL wizard, click **Next** to dismiss the Welcome page.

6. In the Select Deployment Platform step of the wizard, ensure that the **Java 1.5, with support for JAX-WS Annotation** is selected as the deployment platform, and then click **Next**.



Select the deployment platform that matches your requirements.

Platform

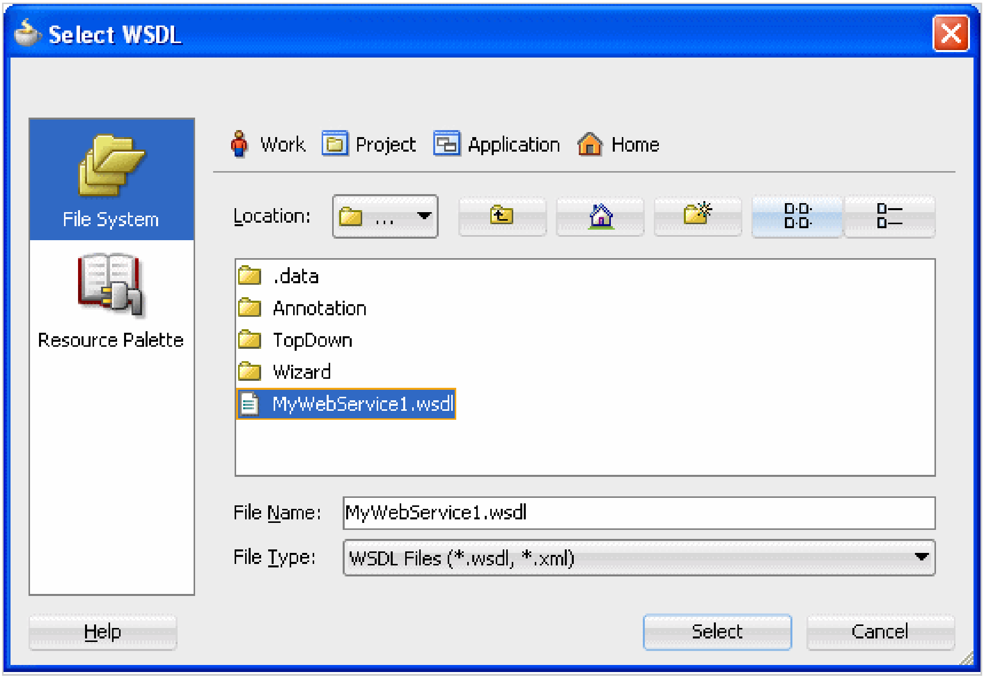
○ **J2EE 1.4 JAX-RPC, with support for 10.3 WLS**
Support for deploying JAX-RPC Web Services to release 10.3 of WLS.

● **Java EE 1.5, with support for JAX-WS Annotations**
Support for deploying to 10.3 WLS with Java annotations using JAX-WS annotation specification.

○ **Java EE 1.5, with support for JAX-WS RI**
Support for deploying to any container that supports the Sun JAX-WS Reference Implementation.

7. In the Select Web Service Description step, you specify the source WSDL to be used in creating the web service.

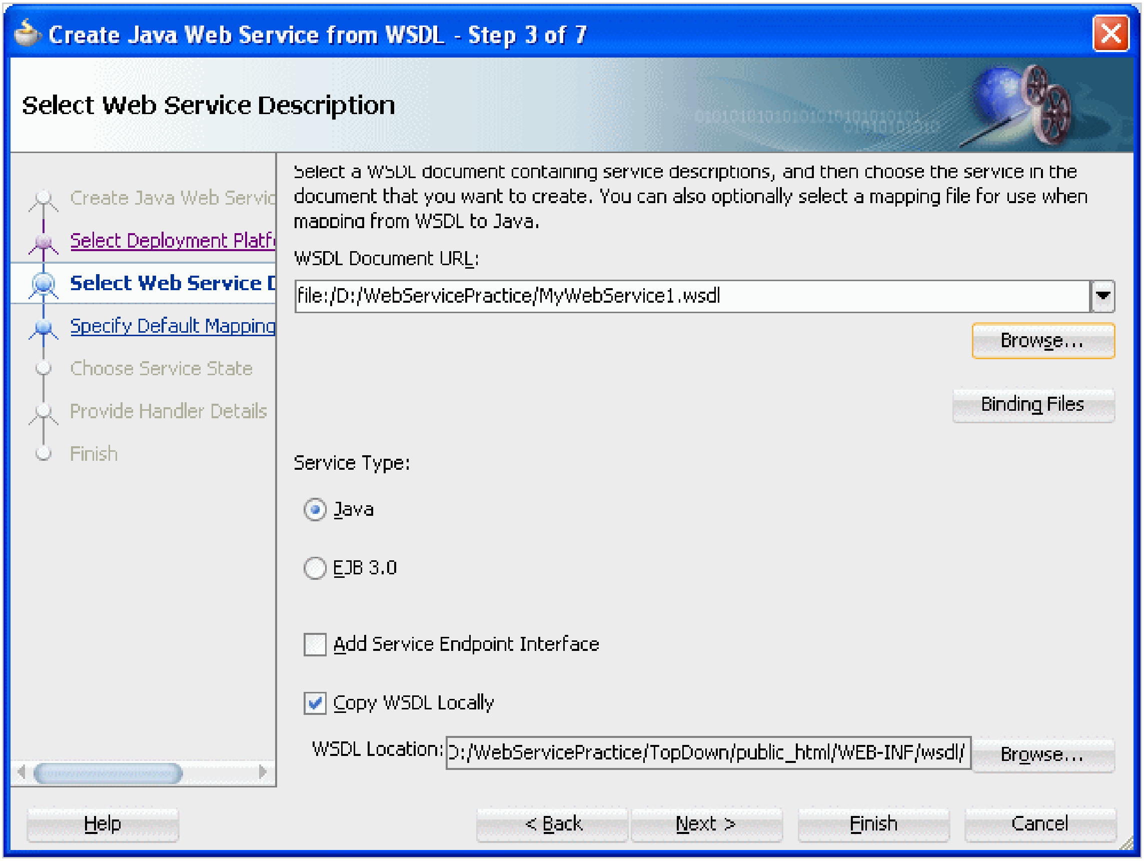
Click **Browse** and navigate to the WSDL you earlier saved (should be something like MyWebService1.wsdl). Select it and click **Select.**



File Name: MyWebService1.wsdl

File Type: WSDL Files (*.wsdl, *.xml)

8. Notice the file path displays in the WSDL Document URL field. Click **Next**.



Select a WSDL document containing service descriptions, and then choose the service in the document that you want to create. You can also optionally select a mapping file for use when mapping from WSDL to Java.

WSDL Document URL: file:/D:/WebServicePractice/MyWebService1.wsdl

Service Type:

● Java

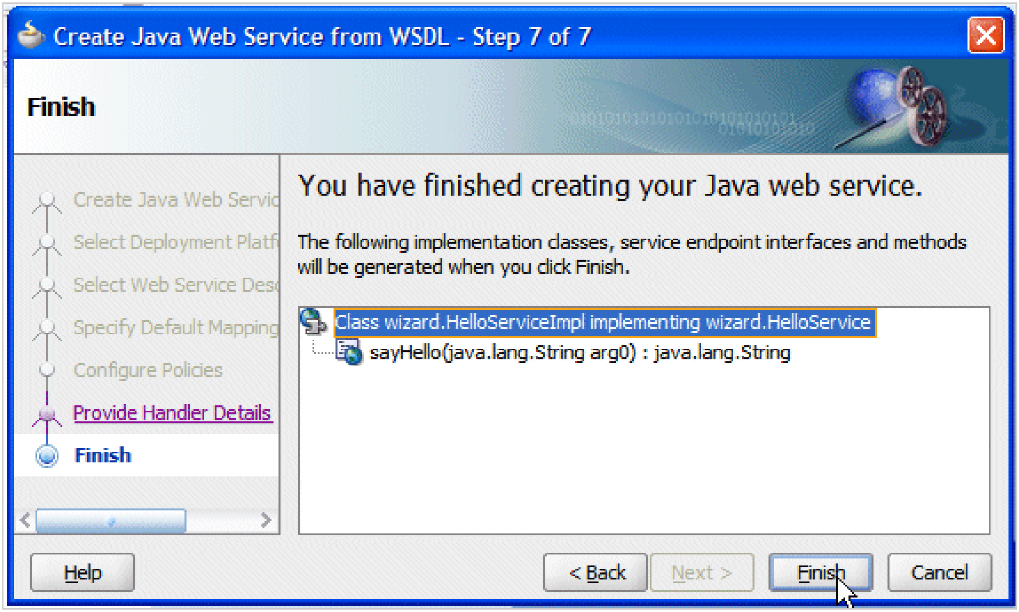
○ EJB 3.0

□ Add Service Endpoint Interface

☑ Copy WSDL Locally

WSDL Location: D:/WebServicePractice/TopDown/public_html/WEB-INF/wsdl/

9. Either click **Finish** to complete the process, or click **Next** to view all the defaulted options then click **Finish**. The image below shows the last page of the wizard.



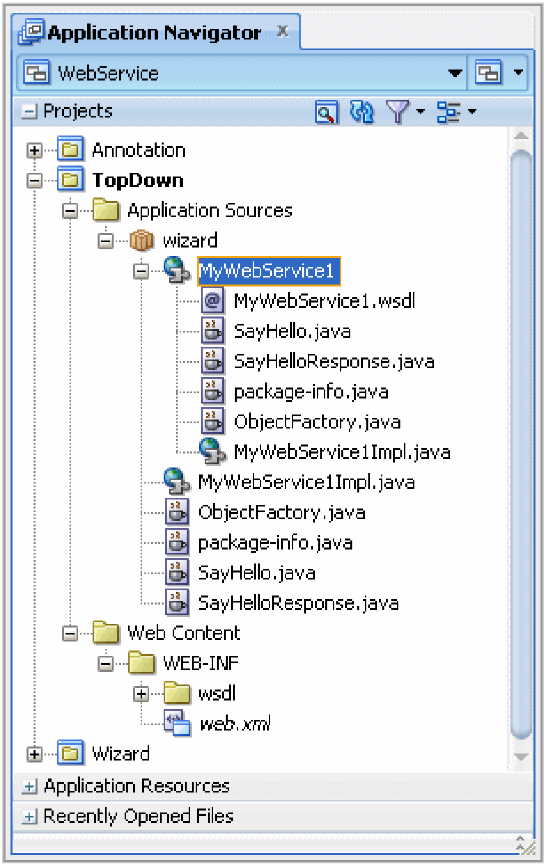
You have finished creating your Java web service.

The following implementation classes, service endpoint interfaces and methods will be generated when you click Finish.

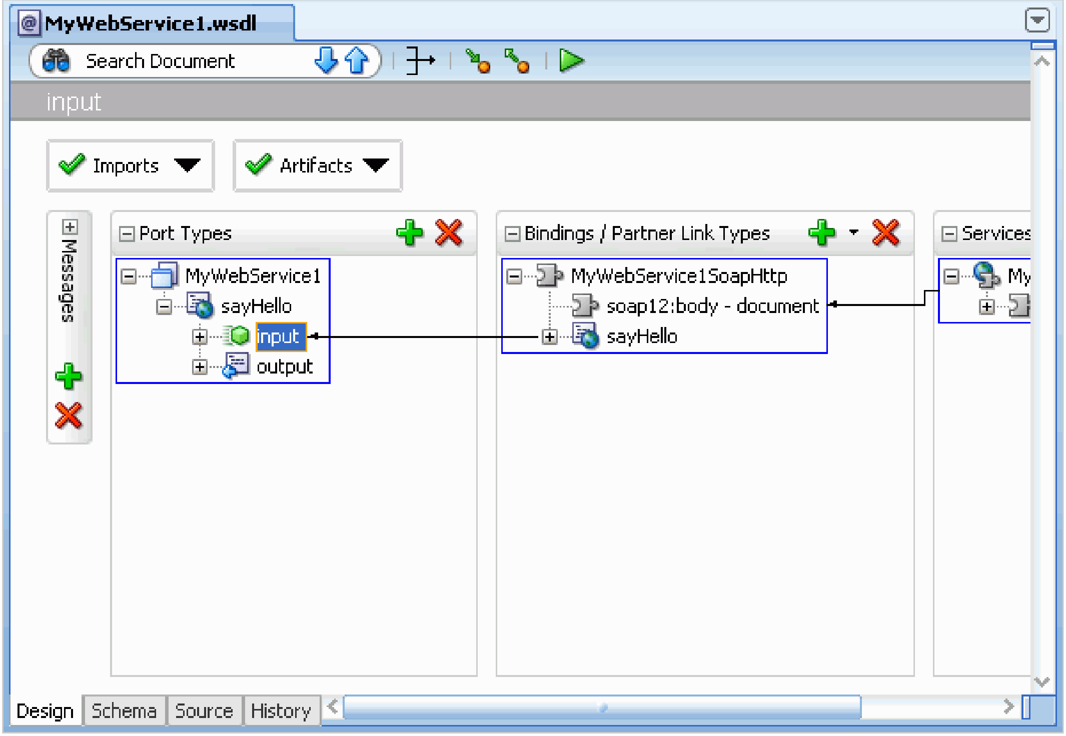
Class wizard.HelloServiceImpl implementing wizard.HelloService
sayHello(java.lang.String arg0) : java.lang.String

10. Click **Save All** to save your work.

11. The Application Navigator now displays the Java web service and all the Java files.

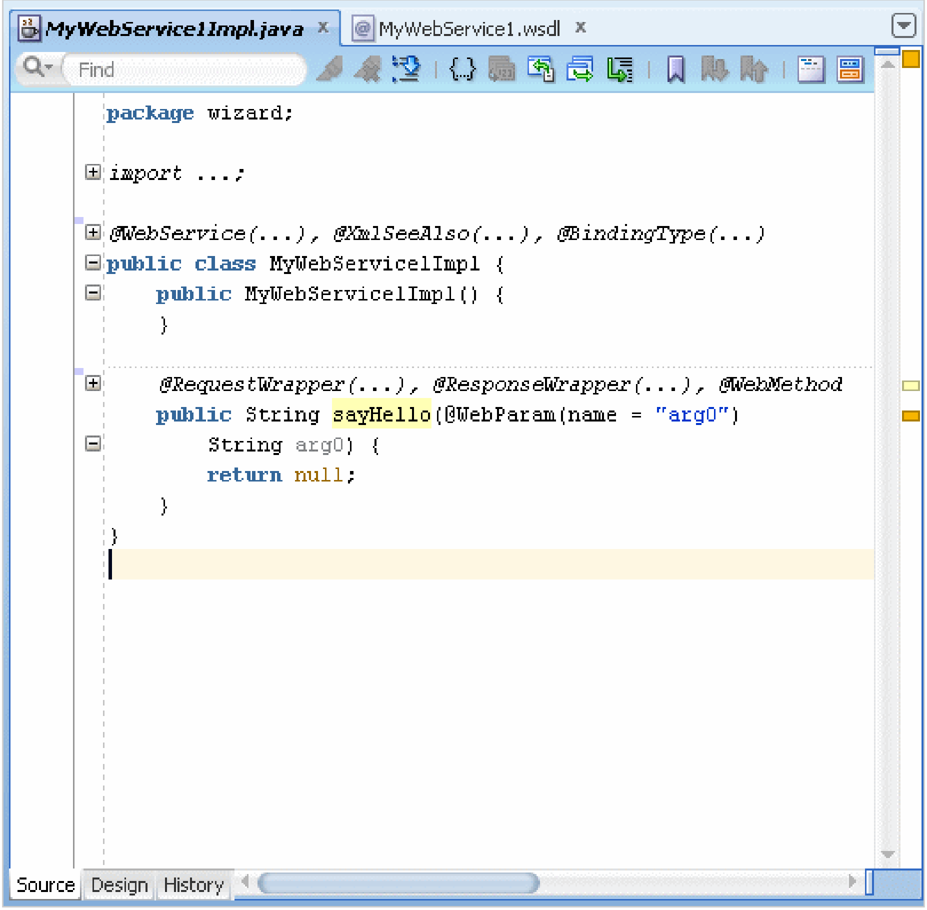


12. Double-click **MyWebService1.wsdl** in the Application Navigator. Notice the design of the WSDL in the editor.



Remember that since the web service was created from the top down, the wizard created all the method headers, but you need to provide the method bodies.

13. Set the `sayHello()` method to return something more reasonable. Double-click the **MyWebService1Impl.java** file to open it in the editor.



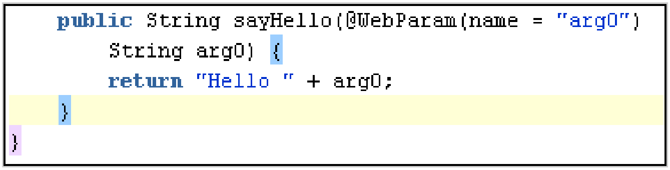
```
package wizard;

import ...;

@WebService(...), @XmlSeeAlso(...), @BindingType(...)
public class MyWebService1Impl {
    public MyWebService1Impl() {
    }

    @RequestWrapper(...), @ResponseWrapper(...), @WebMethod
    public String sayHello(@WebParam(name = "arg0")
        String arg0) {
        return null;
    }
}
```

14. Set the return to **"Hello " + arg0**. This returns the word Hello followed by the value that is entered in the argument.



```
    public String sayHello(@WebParam(name = "arg0")
        String arg0) {
        return "Hello " + arg0;
    }
}
```

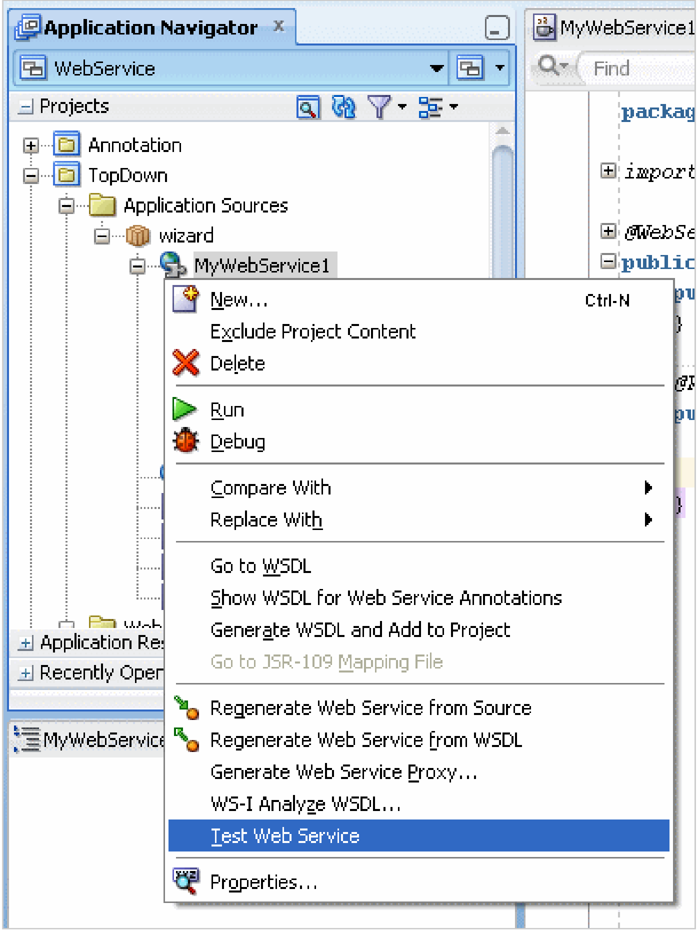
15. Click **Save All** to save your work.

Step 2: Testing the Web Service

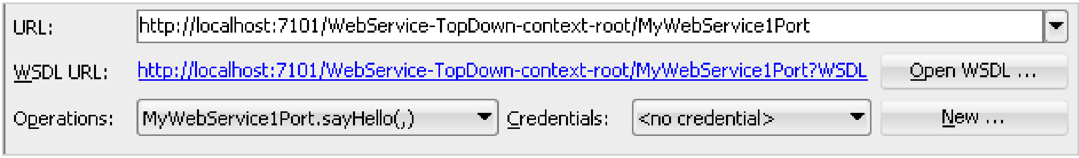
In this section you compile, deploy and test the web service. Just as before, you use the HTTP Analyzer for testing the web service. When you test web services using the analyzer, the service is compiled and deployed to the integrated server. The analyzer is then invoked, enabling you to send and receive values from the web service.

1. In the Application Navigator, right-click the **MyWebService1** node and from the context menu, select **Test Web Service**.

   This option invokes the integrated server, deploys the service, and then starts the HTTP Analyzer.



The top portion of the HTTP Analyzer editor displays the URL for the web service, WSDL URL, and exposed Operations.



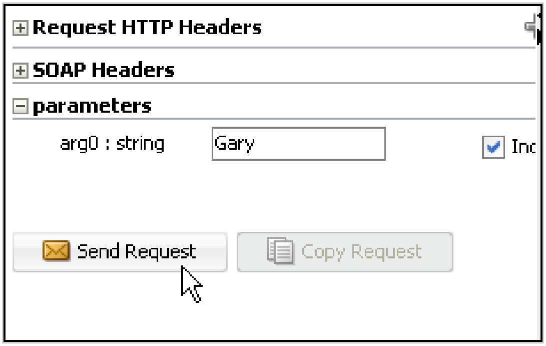
URL: http://localhost:7101/WebService-TopDown-context-root/MyWebService1Port

WSDL URL: http://localhost:7101/WebService-TopDown-context-root/MyWebService1Port?WSDL

Operations: MyWebService1Port.sayHello(,)

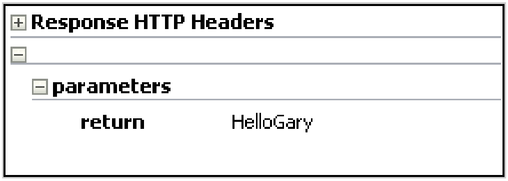
Credentials: <no credential>

2. In the Request area, enter <your name> in the **arg0** field and click **Send Request**.



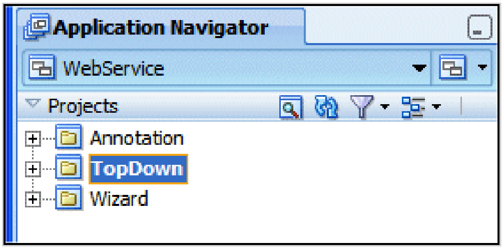
arg0 : string Gary

The analyzer then sends the request to the service and after a bit, the return parameter is displayed.



return HelloGary

3. Close all the tabs in the editor and collapse the **TopDown** node in the Application Navigator.



Previous 1 2 **3** 4 5 6 Next