Testing/Debugging Business Services using Oracle JDeveloper

Purpose

This tutorial covers how to expose a published Business Service as a web service in Oracle JDeveloper; and how to run and debug Business Service in JDeveloper.

Time to Complete

Approximately 30 minutes

Topics

This tutorial covers the following topics:

Overview
Scenario
Prerequisites
Expose a published Business Service as a web service
Run/Test a published Business Service
Debug a published Business Service
Summary

Overview

Business Service developers use JDeveloper to create Business Services. This tutorial describes steps how to expose a published Business Service as web service; then how to test and debug Business Service in development environment using JDeveloper.

Note: Only published Business Services can be exposed as web services.

Scenario

Expose a published Business Service as a web service, then use web service testing page to test the web service. At last, debug Business Service source code in JDeveloper.

This tutorial uses reference implementation Business Service AddressBookManager as an example.

Prerequisites

To complete this tutorial, you should have:

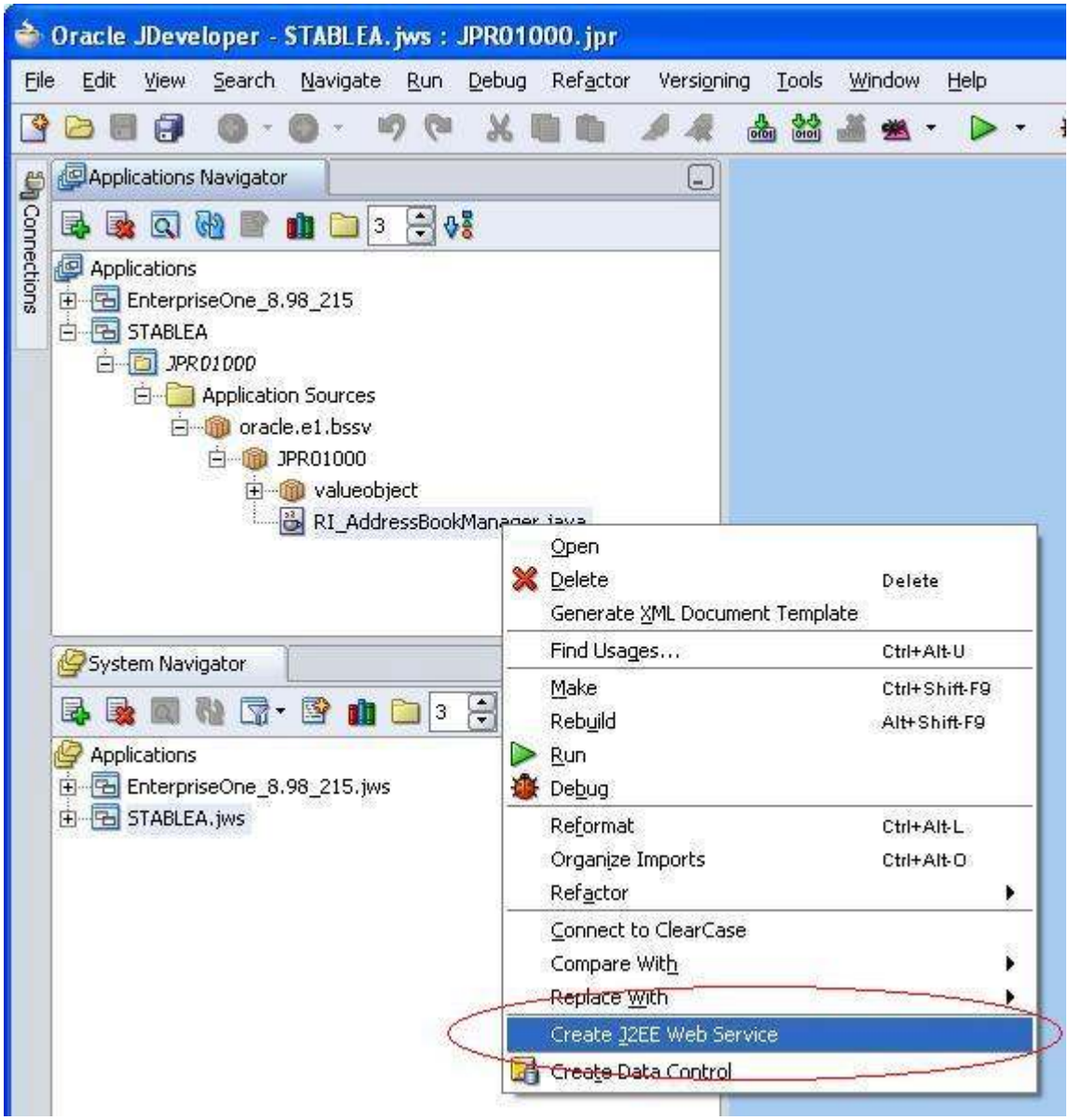
1. JD Edwards EnterpriseOne Windows client installed (Application release 8.12 with Business Service ESU and 8.97 tools release)

2. Oracle JDeveloper 10.1.3 installed
3. Business Service project created in JDeveloper through OMW.
4. Business Service source files created in JDeveloper.

Expose a published Business Service as a web service

We will go through steps how to expose a published Business Service as a web service in JDeveloper.

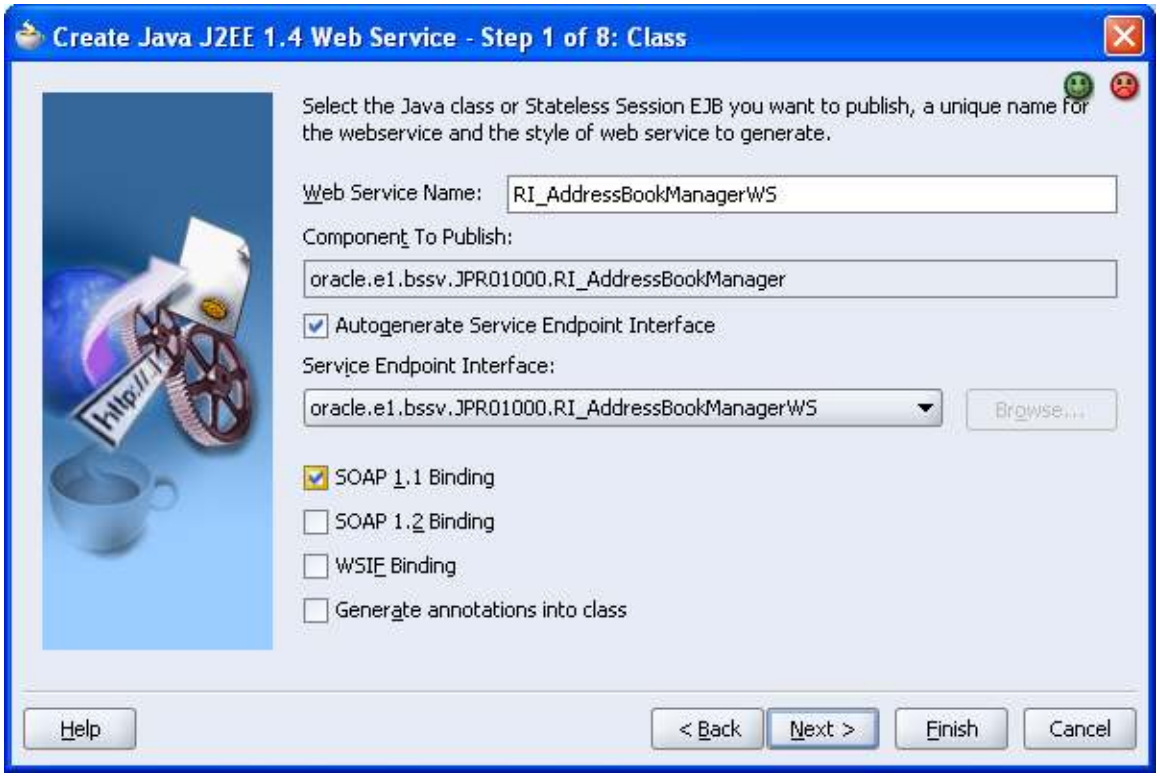
1. In JDeveloper navigation panel, select a published Business Service, e.g. RI_AddressBookManager.java under project JPR01000.



2. Right mouse click

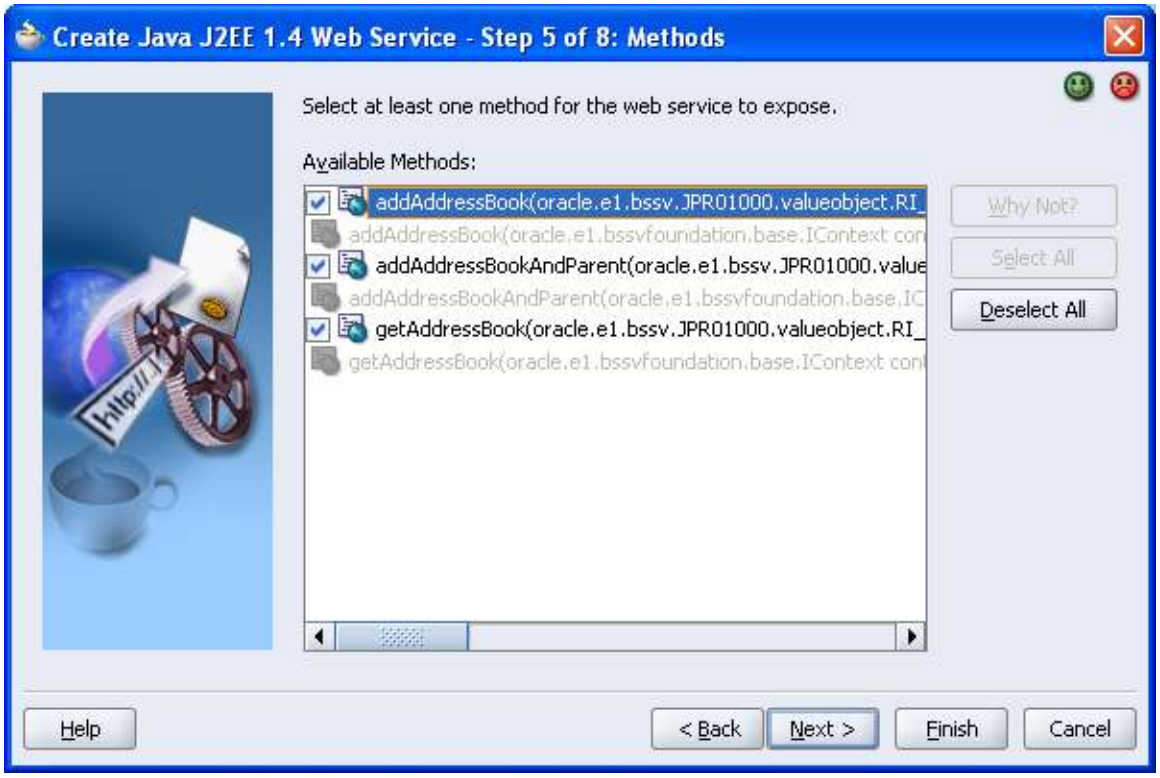
3. Select **Create J2EE Web Service** from the menu
4. In the **Select J2EE Web Service Version** dialog, choose **J2EE 1.4 (JAX-RPC) Web Service** to launch the **Create Java J2EE 1.4 Web Service** wizard.

5. Click **Next** on the welcome page if shown
6. In **Create Java J2EE 1.4 Step 1 of 8** dialog, enter a name in **Web Service Name** field, e.g. RI_AddressBookManagerWS. Value in **Service Endpoint Interface** field changes automatically reflecting the value entered in **Web Service Name** field. Take defaults for rest of fields. Ensure **Component to Publish** has the published Business Service class, e.g. oracle.e1.bssv.JPR01000.RI_AddressBookManager. In addition, ensure that only **SOAP 1.1 binding** and **Autogenerate Service Endpoint Interface** checkboxes are selected

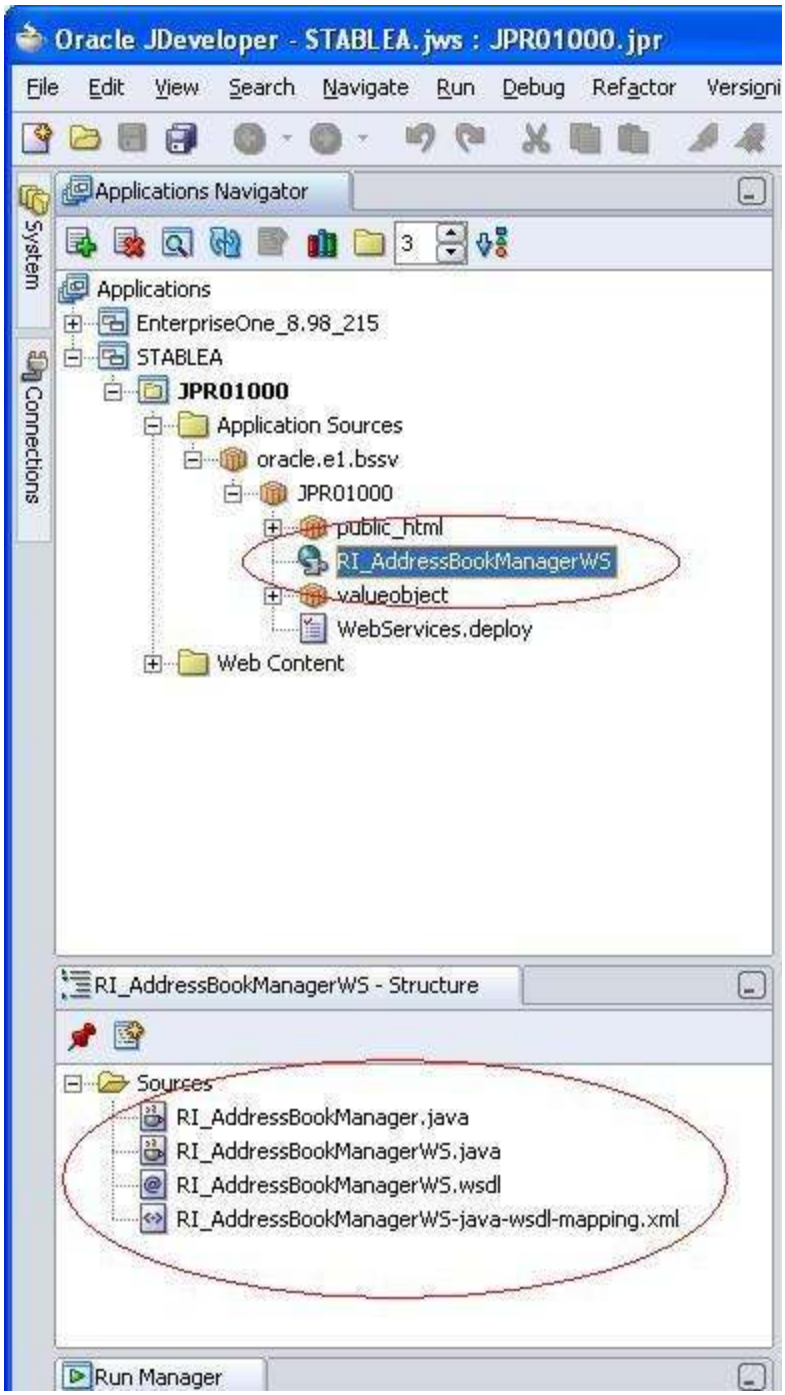


7. Click **Next**
8. In **Create Java J2EE 1.4 Step 2 of 8** dialog, select **Document/Literal** from **SOAP Message Format** combo box.

9. Click **Next**
10. In **Create Java J2EE 1.4 Step 3 of 8** dialog, click **Next**
11. In **Create Java J2EE 1.4 Step 4 of 8** dialog, click **Next**
12. In **Create Java J2EE 1.4 Step 5 of 8** dialog, ensure the methods to expose as web services are selected.



13. Click **Next**
14. In **Create Java J2EE 1.4 Step 6 of 8** dialog, click **Finish**
15. Web service is created for the selected published Business Service.

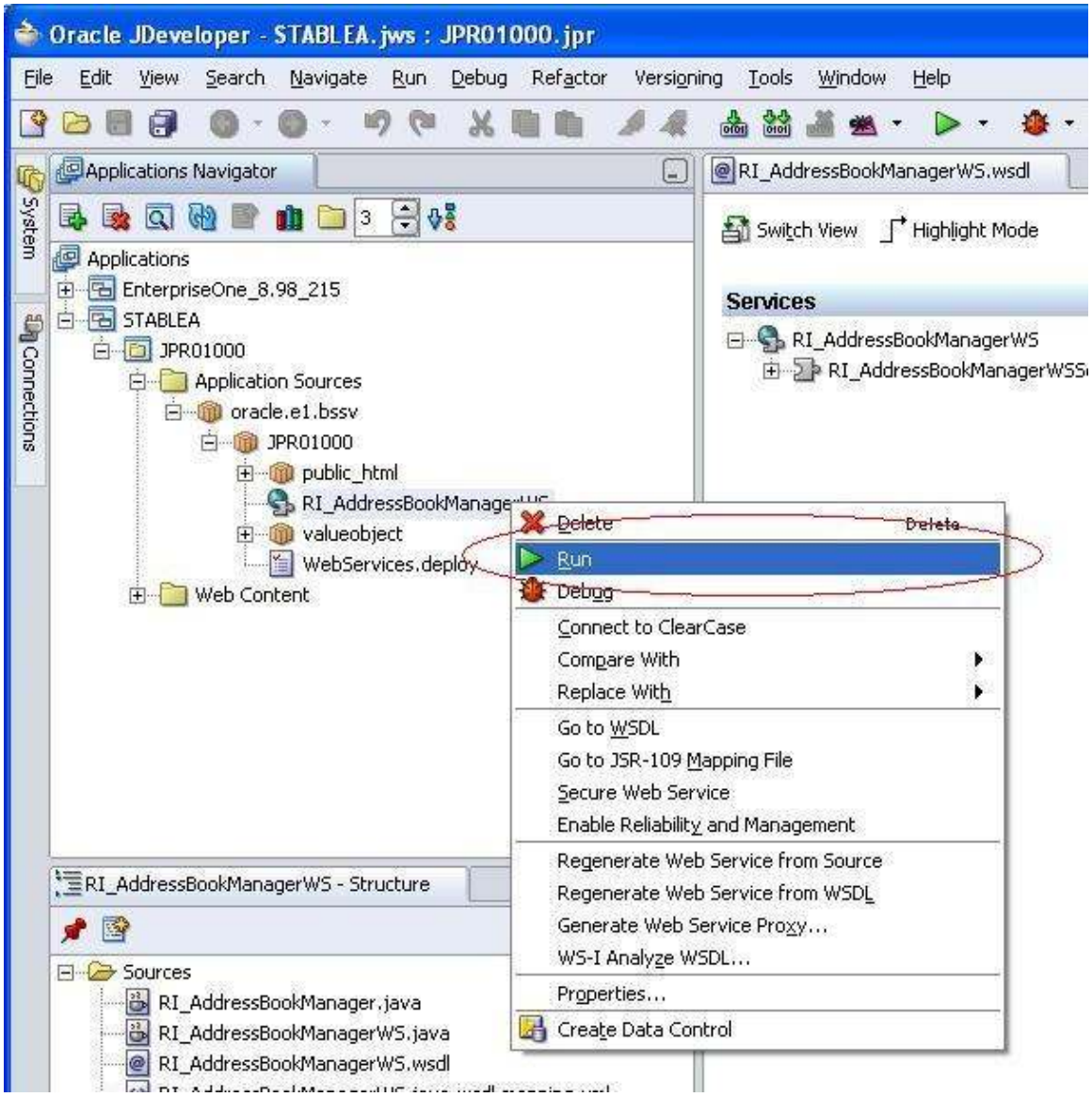


Run/Test a published Business Service

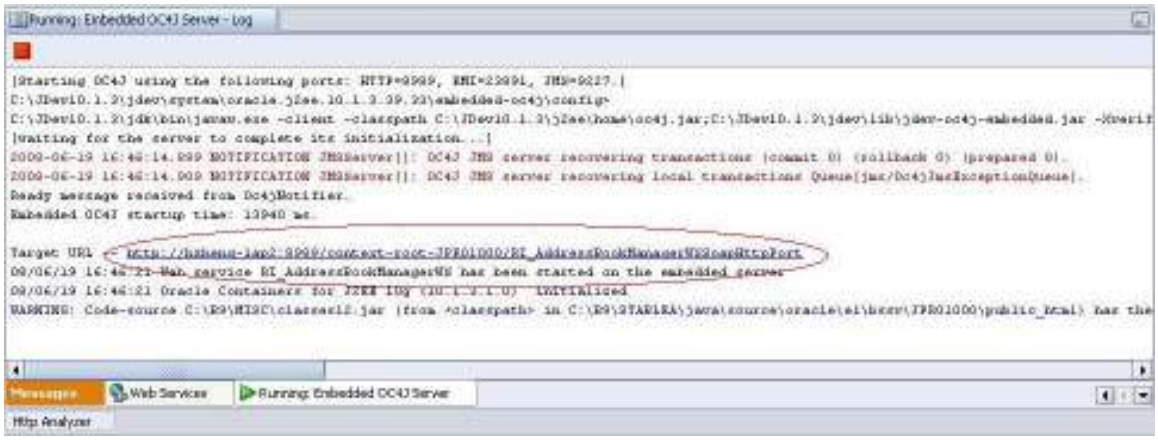
We will go through steps how to run and test the published Business Service.

1. Click on **RI_AddressBookManagerWS** node in the navigator panel to select it.
2. Right mouse click

3. Choose **Run** option

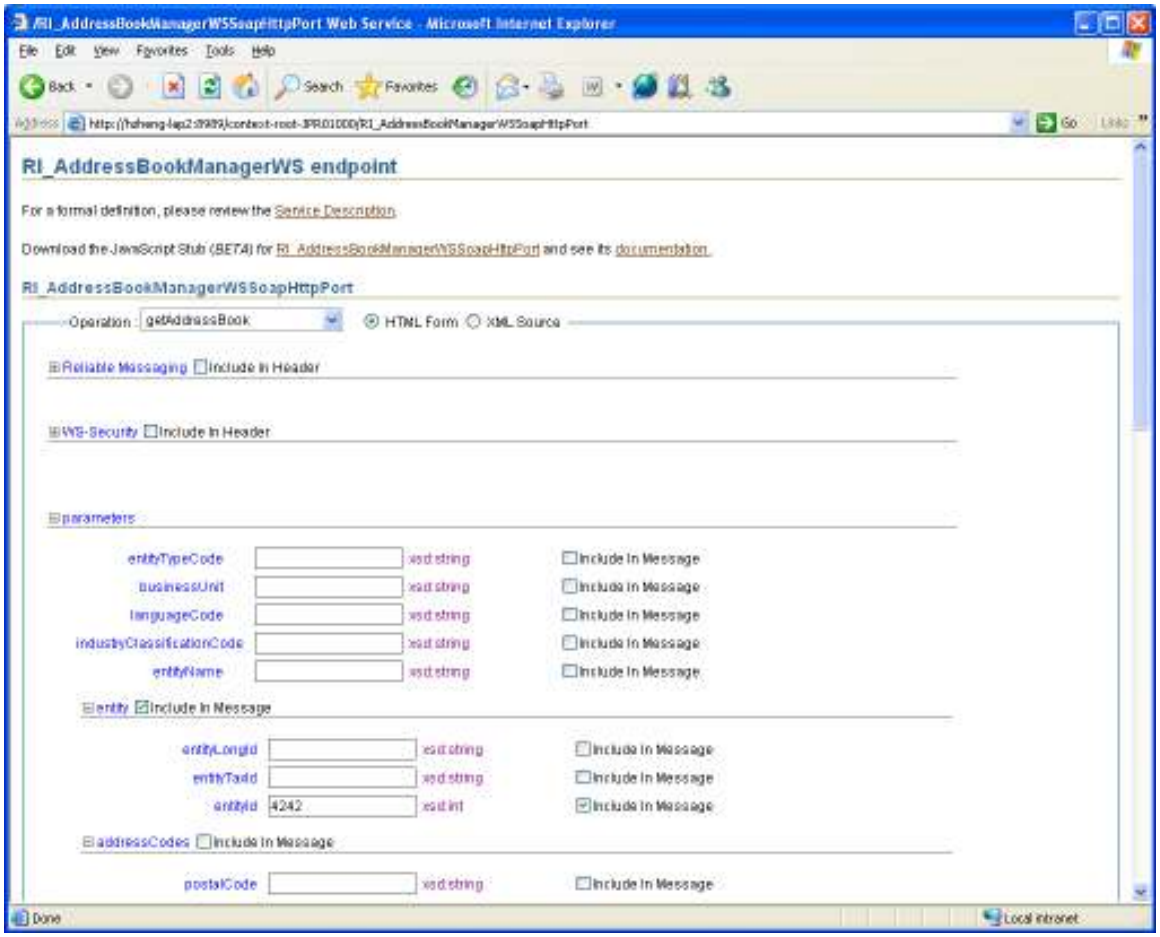


4. This will start "embedded OC4J server" and display messages under "Running: Embedded OC4J Server" message log panel.
5. Wait till you see the Business Service endpoint URL similar to the one listed here.

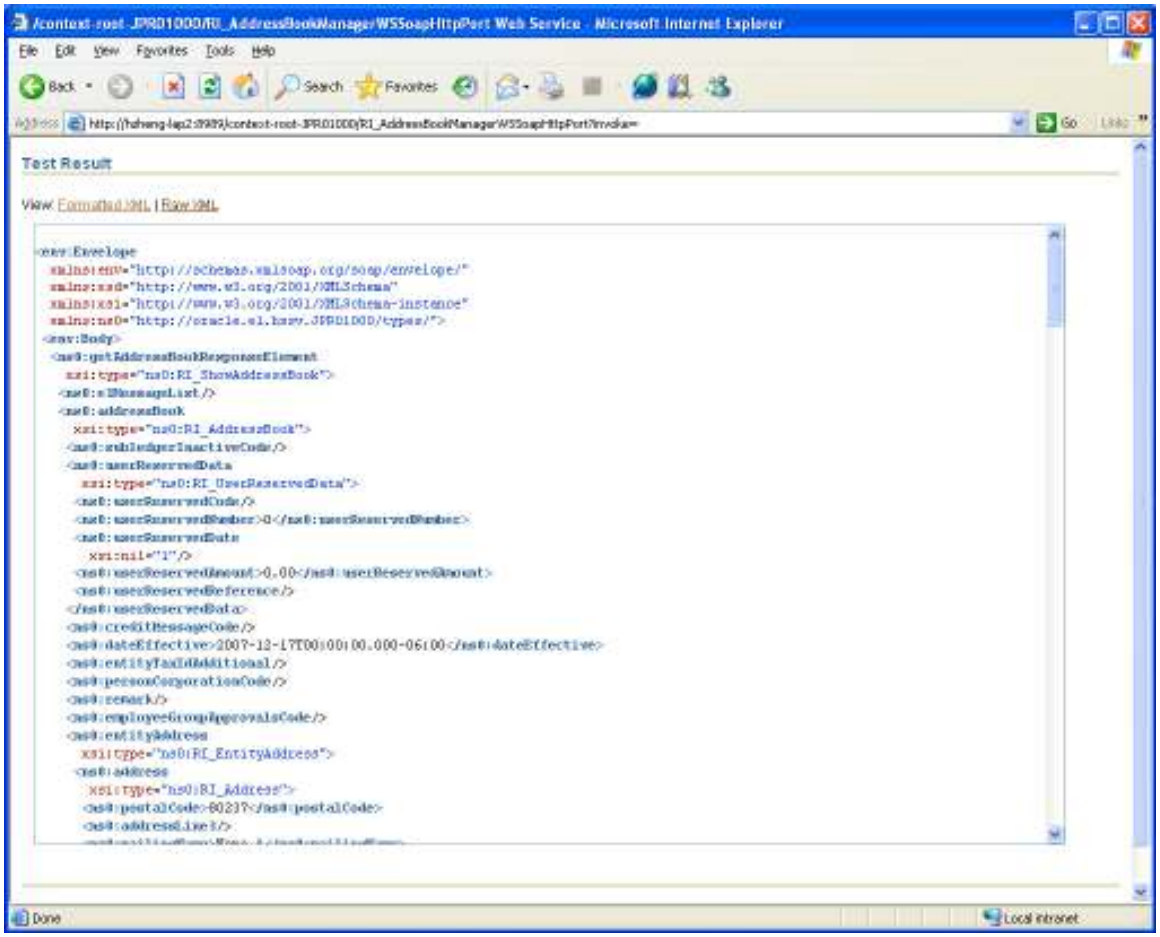


6. Click on the URL. A browser window will open displaying web service testing page.
7. Select an operation from **Operation** combo box, e.g. getAddressBook for RI_AddressBookManager reference implementation.
8. Enter needed values
    a. Please ensure int fields have non-blank value (if not using, set to ZERO).
    b. Please ensure xsd:dateTime fields have complete date & time information (such as 2006-02-02T08:05:00.000-06:00)
    c. After filling out necessary data, you can switch to XML Source using radio button at the top and capture your input data. This will help you for future run and diagnostic.

Note: getAddressBook operation only requires entityId value. So uncheck **Include In Message** checkboxes for all fields under **parameters** section except **entityId** field. Enter a valid address book number, e.g. 4242, in **entityId** field.



9. Click **Invoke** at the bottom of this testing page. This will invoke published Business Service as web service and display response from the Business Service. Click **Formatted XML link** to view a formatted output.



10. Stop the web service by clicking stop button (red square) in "Running: Embedded OC4J Server" panel.

Debug a published Business Service

We will go through steps how to debug a published Business Service.

1. Set break points in Business Service Java files, e.g. at getAddressBook() in RI_AddressBookManager.java.

2. Click on **RI_AddressBookManagerWS** node in the navigator panel to select it.
3. Right mouse click
4. Choose **Debug** option
5. This will start "embedded OC4J server" and display messages under "Debugging: Embedded OC4J Server" message log panel.
6. Click on the Business Service endpoint URL. A browser window will open displaying web service testing page.
7. Select an operation from **Operation** combo box, e.g. getAddressBook for RI_AddressBookManager reference implementation.
8. Fill in appropriate values and click **Invoke** button at the bottom.
9. The debugging cursor should stop at the first hit breakpoint in JDeveloper.
10. Use the debugging features in JDeveloper to debug Business Service code.
11. Stop the web service by clicking stop button (red square) in "Debugging: Embedded OC4J Server" message log panel when finish debugging.

Summary

A Business Service developer is able to complete the whole life cycle of development tasks in JDeveloper – developing Business Services, exposing them as web services, testing and debugging Business Services.