Lab 5: WSDL

In this lab, you will exercise the functionality of WSDL.

Objectives:

- To read an existing WSDL file
- To write a WSDL from scratch and generate a starter service

Part 1: Reading a WSDL Document

In this part, you will examine the WSDL for an existing service. The service is the Weather service from the National Oceanic and Atmospheric Administration(NOAA).

Steps:

_1. The homepage for the service is at:

```
http://graphical.weather.gov/xml
```

Open a Web browser and enter the URL and read about the Web service. Note that there's a link to the WSDL for the service.

Note: If your computer doesn't have Internet access, we've downloaded the WSDL and placed it in **{Lab Installation Directory}/starters/lab05**. You can use your browser's *File - Open* menu to open it from there.

_2. Examine the WSDL, then answer the following questions:

a. What is the *target namespace* of the service (make sure you are looking at the *definitions* element, not a *schema* element)?

_____

b. What are the names of types that this service uses (you only need to list the first three types)?

_____

c. What is the *portType* name?

_____

Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.

d. What is the *style* and *use* of the *NDFDgenByDay* operation? If encoded, what encoding scheme is used?

_____

e. What messages comprise the *NDFDgenByDay* operation?

_____

f. Describe the *parts* for the *NDFDgenByDay* input method.

_____

g. As best you can, write the Java interface that corresponds to the *portType* (you only need to write the first two methods).

h. What is the service endpoint address?

_____

Part 2: Writing WSDL Top Down

In this part, you will write a WSDL from scratch, generate a starter service, then complete and test the service.

A WSDL file describes a Web service -- commonly, developers first create the Web service and then use their Web services toolkit to generate the WSDL file so that client programmers can access the service. This process is referred to as *bottom up*, since the WSDL is generated after the fact.

Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.

However, it is also useful to use a *top down* process where you write the WSDL first, and use it to generate the Web service itself. That process lets architects design the system and hand off the WSDL for implementation by developers. And that's what this part is all about.

The scenario is that you wish to expose an *order status* application as a Web service so that client programs can query the status of an order, given an order number.

Note: The most common cause of problems in this lab is simple typographical errors, so be careful!

Steps:

_1. Start RAD if it's not already running.

_2. Create a new Dynamic Web Project named **lab05Web** associated with *lab05EAR*.

_3. Right-click on the new project's *WebContent/WEB-INF* folder and create a new folder named **wsdl**.

_4. To get started, right-click on the *wsdl* folder and choose *New - Other - Web Services - WSDL* to start the wizard.

   a. On the first wizard page, for the *File name*, enter **orderstatus.wsdl**, then press Next.

   b. On the next page, set the *Target Namespace* to **http://bigcompany.com/orderstatus**, then deselect the *Create WSDL Skeleton* box and press Finish. RAD will create the file and open it in the WSDL editor.

   c. At the bottom of the editor window, note the *Source* and *Design* views. Try switching between the views to see the raw XML and a (for now empty) graphical view. In this lab, you will use the Design view to make writing the WSDL from scratch easier.

_5. Start by creating a *port type* for your service. Follow this procedure:

   a. Right-click on the Design view and choose *Add Port Type*. RAD creates a symbol and lets you enter its name: type **OrderStatus** and press Enter.

   b. Click the *NewOperation* placeholder text and change it to **getOrderStatus** and press Enter.

   c. In the upper-right corner of the Design View, click the *View* drop down and change it to *Advanced*. RAD draws a more complete symbol for the port type.

   d. At the bottom of the RAD window, surface the *Properties* tab (if you don't see it, choose *Window - Show View - Properties* from the RAD menu).

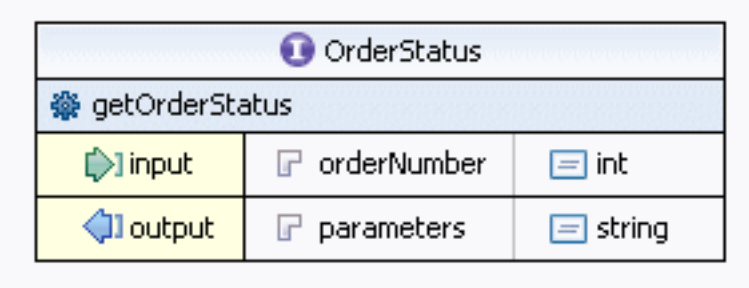
   e. In the port type symbol, for the *Input* message, click the the *parameters* cell to select it.

      Then in the Properties tab, change the *Reference Kind* to *Type* and then change the *Type* to *int*.

   f. Repeat the above to change the *Output* message to *Type* with a type value of *string*.

   g. For the *Input* message, click the *parameters* placeholder text to select it, then in the Properties, change its Name to **orderNumber**.

      At this point, your WSDL should look like Figure 1:



Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.

Figure 1: Port Type

h. Examine the WSDL Source at this point and ensure that you understand it.

_6. Next, you can create the binding for your port type. Follow these steps:

a. In Design View, right click on the white surface of the view and choose *Add Binding* -- RAD draws a small symbol that represents the binding.

b. In the Properties, change the *Name* to **OrderStatusBinding**.

Still in the Properties, for the *Port Type*, select *OrderStatus*.

c. In the Properties, click the *Generate Binding Content* button.

For the *Protocol*, select *SOAP*.

For the *SOAP Binding Options*, select *rpc literal*, then press Finish.

d. Use the Source view to examine the newly added WSDL content and ensure that you understand it.

_7. Finally, configure the Service:

a. In the Design View, first left-click anywhere in the white surface to deselect the binding, then right-click on the white surface of the view and choose *Add Service*.

Enter **OrderStatus** for the Name.

b. Click on the *NewPort* placeholder text to select it, then in the Properties, enter or select:

```
Name:      OrderStatusPort
Binding:   OrderStatusBinding
Address:   http://localhost:9080/lab05Web/OrderStatusService
```

At this point, your WSDL should look like Figure 1:

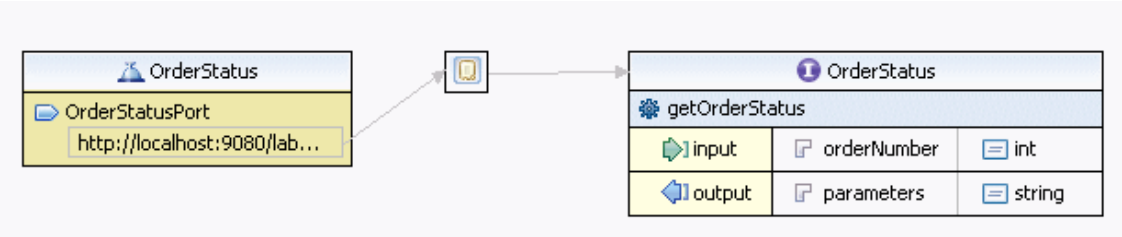


Figure 2: Service

c. Use the Source view to examine the newly added WSDL content and ensure that you understand it.

d. In the Source view, find the *xsd:schema* element at the top of the file, then delete the entire *xsd:schema* element, including the *getOrderStatus* and *getOrderStatusResponse* child elements.

RAD create the schema automatically when you created the WSDL, but it's not needed since you change the messages from "Element" to "Type".

e. Save and close the WSDL.

_8. Now generate a service from the WSDL. Follow these steps:

a. Right-click on the WSDL and choose *Web Services - Generate Java bean skeleton* to start the wizard.

b. On the first wizard page, examine the defaults, then press Next.

Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.

c. On the next wizard page, note the defaults, then press Next.

d. On the next wizard page, note the defaults, then press Finish.

e. In the Project Explorer, under *Java Resources: src*, look in the *com.bigcompany.orderstatus* package for the generated files.

OrderStatus.java is an interface for your service and *OrderStatusBindingImpl.java* is the "starter" Web service implementation. You will complete it in the next step.

Examine these files and ensure that you understand how RAD mapped entries in the WSDL to these Java artifacts and how the JAX-WS annotations work.

_9. Edit OrderStatusBindingImpl.java and add code to the *getOrderStatus* so that it returns a String like:

```
The status of order XX is PENDING
```

where XX is the integer passed to the method.

You should have no compile errors at this point.

_10. To test, in the Enterprise Explorer, expand the *lab05Web/Services* entry, then right-click on *{http://bigcompany.com/orderstatus}OrderStatus* and choose *Test with Web Services Explorer*.

Under *Operations*, click on the *getOrderStatus* link, then enter an integer for *orderNumber* and press the Go button. You should see the returned string in the *Status* pane.

_11. Examine the raw SOAP messages and ensure that they match what you'd expect for the WSDL that you wrote.

_12. Use RAD's *File - Close All* menu to close all open windows.

Part 3: Writing a Standalone Client

Steps:

_1. Using the same techniques as in Lab02, create a simple client for the OrderStatus service that invokes the getOrderStatus method.

Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.

Copyright © Descriptor Systems 2014. All Rights Reserved. Materials may not be reproduced in any form without prior written consent of Joel Barnum.