Web services in Java

CS 475

1

A Simple Example

- Echo server example
- Java supports web services in core Java
  - JAX-WS (Java API for XML-Web Services)
  - Current version 2.1 (comes with Java 6)
- More typical way to use web services in a production environment is to deploy the service in a web container such as Apache Tomcat or SUN's Java application container Glassfish
- Our example uses Core Java only

2

Implementing a simple web service

- Create the "service endpoint interface"
  - Interface for web service
- Create the "service implementation bean"
  - Class that implements the service
- Create the service publisher
  - In full production mode, one would use a Java application server such as BEA WebLogic, Jboss, Glassfish, etc.
  - Our example will use a simple Java application

3

Service Endpoint Interface

```
package example.echo;  // echo server
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.soap.SOAPBinding;
import javax.jws.soap.SOAPBinding.Style;

@WebService     // This signals that this is a Service Endpoint Interface (SEI)
@SOAPBinding(style = Style.RPC) // Needed for the WSDL
public interface EchoServer {
      @WebMethod   // This signals that this method is a service operation
            String EchoMessage(String strMsg);    }
```

4

Service Implementation Bean

```
package example.echo;
import javax.jws.WebService;
/**
 *  The @WebService property endpointInterface links the
 *   SIB (this class) to the SEI (example.echo.EchoServer).
 *   Note that the method implementations are not annotated
 *  as @WebMethods.
*/

@WebService(endpointInterface = "example.echo.EchoServer")
public class EchoServerImpl implements EchoServer {
      public String EchoMessage(String Msg)    {
            String capitalizedMsg;
            System.out.println("Server: EchoMessage() invoked...");
            System.out.println("Server: Message > " + Msg);
            capitalizedMsg = Msg.toUpperCase();
             return(capitalizedMsg);
       }
}
```

5

Service Publisher

```
package example.echo;
import javax.xml.ws.Endpoint;

public class EchoServerPublisher {
      public static void main(String[ ] args) {
      // 1st argument is the publication URL
      // 2nd argument is an SIB instance

      Endpoint.publish("http://localhost:9876/es", new  EchoServerImpl());
 }
}
```

6

Deploying and testing the echo web service

- Compile the Java code
- Run the publisher

  java example.echo.EchoServerPublisher &

- Testing the web service with a browser

  URL: http://localhost:9876/es?wsdl

7

WSDL for echo service

```
<definitions targetNamespace="http://echo.example/" name="EchoServerImplService">
<types/>
<message name="EchoMessage”> <part name="arg0" type="xsd:string"/>  </message>
<message name="EchoMessageResponse”> <part name="return" type="xsd:string"/> </message>

<portType name="EchoServer">
    <operation name="EchoMessage">
        <input message="tns:EchoMessage"/>
        <output message="tns:EchoMessageResponse"/>
   </operation>
</portType>

<binding name="EchoServerImplPortBinding" type="tns:EchoServer">
   <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>
       <operation name="EchoMessage">
        <soap:operation soapAction=""/>
        <input> <soap:body use="literal" namespace="http://echo.example/"/>  </input>
        <output>   <soap:body use="literal" namespace="http://echo.example/"/> </output>
     </operation>
</binding>

<service name="EchoServerImplService">
<port name="EchoServerImplPort" binding="tns:EchoServerImplPortBinding”>
        <soap:address location="http://localhost:9876/es"/>
</port>
</service>
</definitions>
```

8

EchoClient

```
package example.echo;
import javax.xml.namespace.QName;
import javax.xml.ws.Service;
import java.net.URL;

class EchoClient {
        public static void main(String argv[ ]) throws Exception {
                if (argv.length < 1) {
                System.out.println("Usage: java EchoClient \"MESSAGE\"");System.exit(1);}

            String strMsg = argv[0];
            URL url = new URL("http://localhost:9876/es?wsdl");
               // Qualified name of the service:
               //   1st arg is the service URI
               //   2nd is the service name published in the WSDL
             QName qname = new QName("http://echo.example/", "EchoServerImplService");
             Service service = Service.create(url, qname);
             // Extract the endpoint interface, the service "port".
            EchoServer eif = service.getPort(EchoServer.class);
           System.out.println(eif.EchoMessage(strMsg));
        }
}
```

9

Conclusions

- Simple example of web services in Java
- Some more points
  - Client can be in any language, e.g. Perl
  - Service is not multi-threaded but it is possible to make it multi-threaded
  - Restrictions on types of method parameters and return values

10