WebServices

**Created By:** Rupendra Sharma

**WebServices**

Web services are platform and technology independent B2B (Business to Business) component. Web services expose business method for other application on the web in a technology independent manner. Web services facilitate interoperability.  
 Before web service interoperability was achieved with the help of CORBA (Common object request broker architecture). CORBA based interoperability has following problem:

|  |
| --- |
| **NOTE:** CORBA is a protocol, implementation responsibility is to provided only 2 company. |

1. Limited vendor support, only 2 vendors were there for CORBA implementation.
2. Third party prop righted protocol and implementation were required for interoperability.

Limitations of CORBA were eliminated by web services with the help of following three xml based technologies.

1- SOAP (Simple object access protocol)

2- WSDL (web services Description language)

3- UDDI (Universal Description Discovery & integration)  
  
**1- SOAP:** SOAP is an xml based protocol which is used to represents object and method in text format.  
This protocol facilitates transfer of method invocation information and method result in text format from the client to the web services and web service to client.  
For each web service invocation an SOAP packet is created. This SOAP packet is preloaded in to HTTP packets which are transmitted over the network.

**2- WSDL:** WSDL is an xml based language which contains xml data types and elements to describe methods, there parameters, return types etc.  
For each web services a WSDL document is created this document contains all the information of wed services in xml format.

**3- UDDI:** To facilitate automatic discovery of web services on the web, concept of UDDI server was evolved. A UDDI server is used by the web service developers to publish their WSDL document. These documents are search by the web service client.  
  
2.4- Using the information of WSDL, a SOAP packet is created for the method call.

2.5- SOAP packet is pay loaded into HTTP packet.

2.7- SOAP packet is extracted and passed by the proxy on the server.

3.2- A SOAP packet is created for the request.

3.3- SOAP packet is pay loaded into HTTP packet.

3.4- HTTP packet is sent to the client.

3.5- SOAP packet is extracted and passed by the proxy at the client end.

In order to package a simple class as a web service different means can be used. All the application server provide the Ant task for packaging classes as web services, various plug-ins are available for the IDE,s , helper classes are provided by the framework such as spring.

**<servicegen> and <service>** Ant task are provided by Weblogic server to package a POJO class as a web service. In order to use these Ant task following jar files must be available in the class path:

1- C:\bea\weblogic81\server\lib\**webservices.jar**

2- C:\bea\weblogic81\server\lib\**webserviceclient.jar**

|  |
| --- |
| **Simple POJO class** |
| package mypack;  public class AdderSubtractor  {  public int add(int x, int y)  {  return(x+y);  }  public int subtract(int x, int y)  {  return(x-y);  }  } |
| For packaging as a web services need to **build.xml** file.  <project name="buildWebService" default="package" basedir=".">  <target name="compile">  <javac srcDir="." destDir="." />  </target>  **<target name="package" depends="compile">**  **<servicegen destEar="as.ear" contextURI="as">**  **<service javaClassComponents="mypack.AdderSubtractor" targetNameSpace="http://localhost:7001" serviceName="AS" serviceURI="/AS" generateTypes="true" expandMethods="true">**  **<client packageName="asClient" />**  **</service>**  **</servicegen>**  **</target>**  </project> |
| To make web service ear file (as.ear) need to run Ant tool. Before Running Ant tool need to set the path and classpath as following.  File Name- **p.bat**  @echo off  **set path=%path%;C:\bea\weblogic81\server\bin**  **set classpath=.;C:\bea\weblogic81\server\lib\webservices.jar;C:\bea\weblogic81\server\lib\webserviceclient.jar;**E:\PROGRAMS\WebServices\AS\_client.jar  **Steps(Give following command):**  1- p (p.bat)  2- Ant  Build successful ……..  After building successful deploy **as**.**ear** on weblogic Application server.  On deployment time choose **Your Deployment Resource - (Applications)**.  Under Testing tab choose Launch Test Page <http://localhost:7001/as/AS>. In this page under **Recommendation:** choose Click here to download WSDL document . Set classpath for WSDL jar file in p.bat file. To see the content of WSDL jar file give following command:  **jar -tf AS\_client.jar** |
| **Create console based client for using web services:**  import asClient.\*;  public class Client  {  public static void main(String args[])  {  try  {  int a=Integer.parseInt(args[0]);  int b=Integer.parseInt(args[1]);  **String wsdlUrl = "http://localhost:7001/as/AS?WSDL";**  **AS service = new AS\_Impl( wsdlUrl );**  **ASPort port = service.getASPort();**  int s= port.add(a,b);  int d= port.subtract(a,b);    System.out.println("Sum : "+s);  System.out.println("Subtract : "+d);    }  catch(Exception e)  {  System.out.println(e);  }  }  }  **NOTE:**  1- To compile Client.java need to set the WSDL jar file path (**AS\_client**) in classpath environment variable.  2- First run **p.bat** for setting classpath of WSDL jar file (**AS\_client**).  3- Now compile the **Client.java** and run it for testing. |
| **Output :**  E:\PROGRAMS\WebServices>java Client 10 2  Sum : 12  Subtract : 8 |