**Steps to create the JAX-WS webservice and client**

Reference: <http://javapostsforlearning.blogspot.in/2013/03/jaxws-web-service-eclipse-tutorial.html#idc-cover>

1. Create 2 java projects one for the provider and one for the client.

**Provider side**

1. In the provider project create an interface:

package com.lps.test.webservice;

import javax.jws.WebMethod;

import javax.jws.WebService;

import javax.jws.soap.SOAPBinding;

import javax.jws.soap.SOAPBinding.Style;

import javax.jws.soap.SOAPBinding.Use;

@WebService

@SOAPBinding(style = Style.DOCUMENT, use=Use.LITERAL)

public interface TestWebservice {

@WebMethod

public String checkLoanStatus(String loanNumber);

}

1. Implement the interface:

**package** com.lps.test.webservice;

**import** javax.jws.WebService;

@WebService(endpointInterface="com.lps.test.webservice.TestWebservice")

**public** **class** TestWebserviceImpl **implements** TestWebservice {

@Override

**public** String checkLoanStatus(String loanNumber) {

**return** "Loan:"+loanNumber+" is active.";

}

}

1. Create a class to publish the webservice:

**package** com.lps.test.webservice;

**import** javax.xml.ws.Endpoint;

**public** **class** TestWebservicePublisher {

**public** **static** **void** main(String[] args) {

Endpoint.*publish*("http://localhost:8080/services/TestWebservice",**new** TestWebserviceImpl());

}

}

1. When running the publishing class you might get an exception:

Wrapper Class Package.Jaxws.MethodName Is Not Found. Have You Run APT To Generate Them?

To resolve this error,

1. open up command prompt and go to the “bin” directory that contains the class files for the interface, impl and publisher classes.
2. Run this command:

set CLASSPATH=.;%CLASSPATH%

wsgen -keep -cp . com.lps.test.webservice.TestWebservice

1. That creates a “jaxws” directory and generates java code for request&response of the webservice operation.
2. Copy the jaxws directory (from under bin) to the src directory
3. Run the publisher class as standalone java program, and this will publish our test webservice!

You will see the following messages:

Apr 23, 2014 11:24:31 AM com.sun.xml.internal.ws.model.RuntimeModeler getRequestWrapperClass

INFO: Dynamically creating request wrapper Class com.lps.test.webservice.jaxws.CheckLoanStatus

Apr 23, 2014 11:24:31 AM com.sun.xml.internal.ws.model.RuntimeModeler getResponseWrapperClass

INFO: Dynamically creating response wrapper bean Class com.lps.test.webservice.jaxws.CheckLoanStatusResponse

You can access the wsdl file for the test webservice using the URL similar to the following:

<http://localhost:8080/services/TestWebservice?wsdl>

Here is the WSDL:

<?xml version="1.0" encoding="UTF-8"?>

<!-- Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6. -->

<!-- Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.1.6 in JDK 6. -->

[<definitions name="**TestWebserviceImplService**" targetNamespace="**http://webservice.test.lps.com/**" xmlns="**http://schemas.xmlsoap.org/wsdl/**" xmlns:xsd="**http://www.w3.org/2001/XMLSchema**" xmlns:tns="**http://webservice.test.lps.com/**" xmlns:soap="**http://schemas.xmlsoap.org/wsdl/soap/**"><types><xsd:schema>](http://localhost:8080/services/TestWebservice?wsdl)<xsd:import schemaLocation="**http://localhost:8080/services/TestWebservice?xsd=1**" namespace="**http://webservice.test.lps.com/**"/></xsd:schema></types>[<message name="**checkLoanStatus**">](http://localhost:8080/services/TestWebservice?wsdl)<part name="**parameters**" element="**tns:checkLoanStatus**"/></message>[<message name="**checkLoanStatusResponse**">](http://localhost:8080/services/TestWebservice?wsdl)<part name="**parameters**" element="**tns:checkLoanStatusResponse**"/></message>[<portType name="**TestWebservice**"><operation name="**checkLoanStatus**">](http://localhost:8080/services/TestWebservice?wsdl)<input message="**tns:checkLoanStatus**"/><output message="**tns:checkLoanStatusResponse**"/></operation></portType>[<binding name="**TestWebserviceImplPortBinding**" type="**tns:TestWebservice**">](http://localhost:8080/services/TestWebservice?wsdl)<soap:binding style="**document**" transport="**http://schemas.xmlsoap.org/soap/http**"/>[<operation name="**checkLoanStatus**">](http://localhost:8080/services/TestWebservice?wsdl)<soap:operation soapAction="**"/>**[**<input>**](http://localhost:8080/services/TestWebservice?wsdl)**<soap:body use="literal"/></input>**[**<output>**](http://localhost:8080/services/TestWebservice?wsdl)**<soap:body use="literal"/></output></operation></binding>**[**<service name="TestWebserviceImplService"><port name="TestWebserviceImplPort" binding="tns:TestWebserviceImplPortBinding">**](http://localhost:8080/services/TestWebservice?wsdl)**<soap:address location="http://localhost:8080/services/TestWebservice"/></port></service></definitions>**

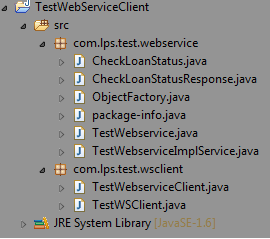
**Client Side**

1. Generate the client code from this wsdl:

cd %project\_home%/src

wsimport -s . <http://localhost:8080/services/TestWebservice?wsdl>

It generates the webservice stub code (about 4 or 5 java files)



1. Create a client program

Identify the auto generated client class (that has the @WebServiceClient) and use it in the client program to get the port, interface etc..

**package** com.lps.test.wsclient;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** java.util.Collections;

**import** java.util.HashMap;

**import** java.util.Map;

**import** javax.xml.namespace.QName;

**import** javax.xml.ws.BindingProvider;

**import** javax.xml.ws.Service;

**import** javax.xml.ws.WebServiceRef;

**import** javax.xml.ws.handler.MessageContext;

**import** com.lps.test.webservice.TestWebservice;

@WebServiceRef(TestWebservice.**class**)

**public** **class** TestWSClient {

**public** **static** **void** main(String[] args) {

TestWebserviceImplService port = **new** TestWebserviceImplService();

TestWebservice svc = port.getTestWebserviceImplPort();

svc.checkLoanStatus("1234");

}

}

1. Running the above client program invokes the webservice.