

# Web Services

# **Let us write Eclipse-Java-Axis2 based Web Services**

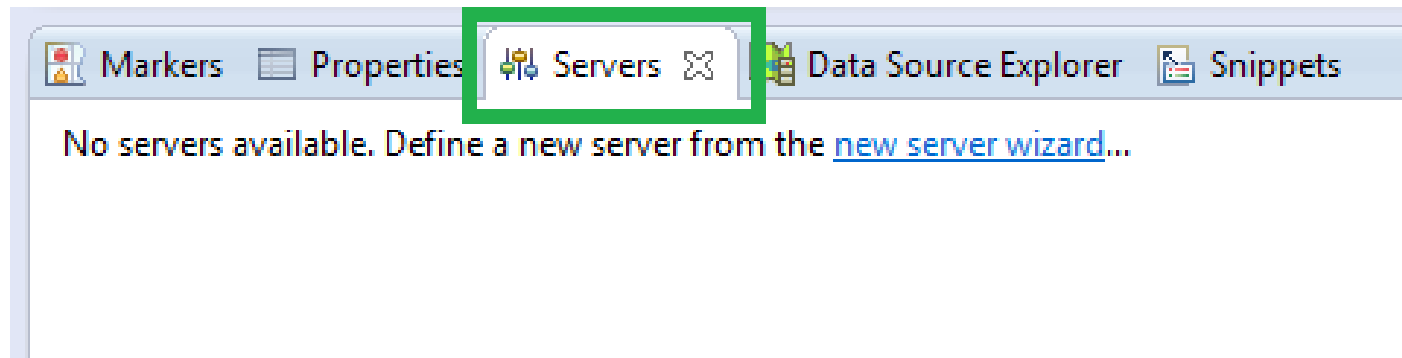
# Setup the development environment

1.To setup the development environment you need to download and extract Eclipse EE and Tomcat servlet container.

- ✓ Eclipse IDE for Java EE Developers
- ✓ Apache Tomcat (Download zip archive)

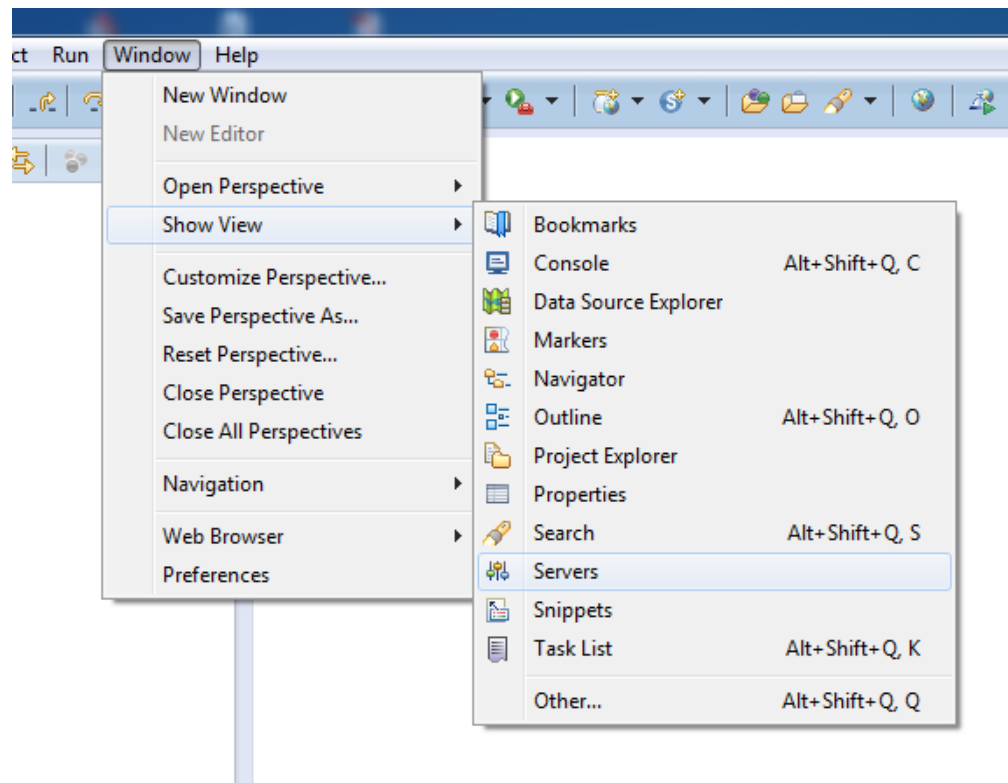
# Setup the development environment

- Double click to open Eclipse. You should be able to see the Servers tab.



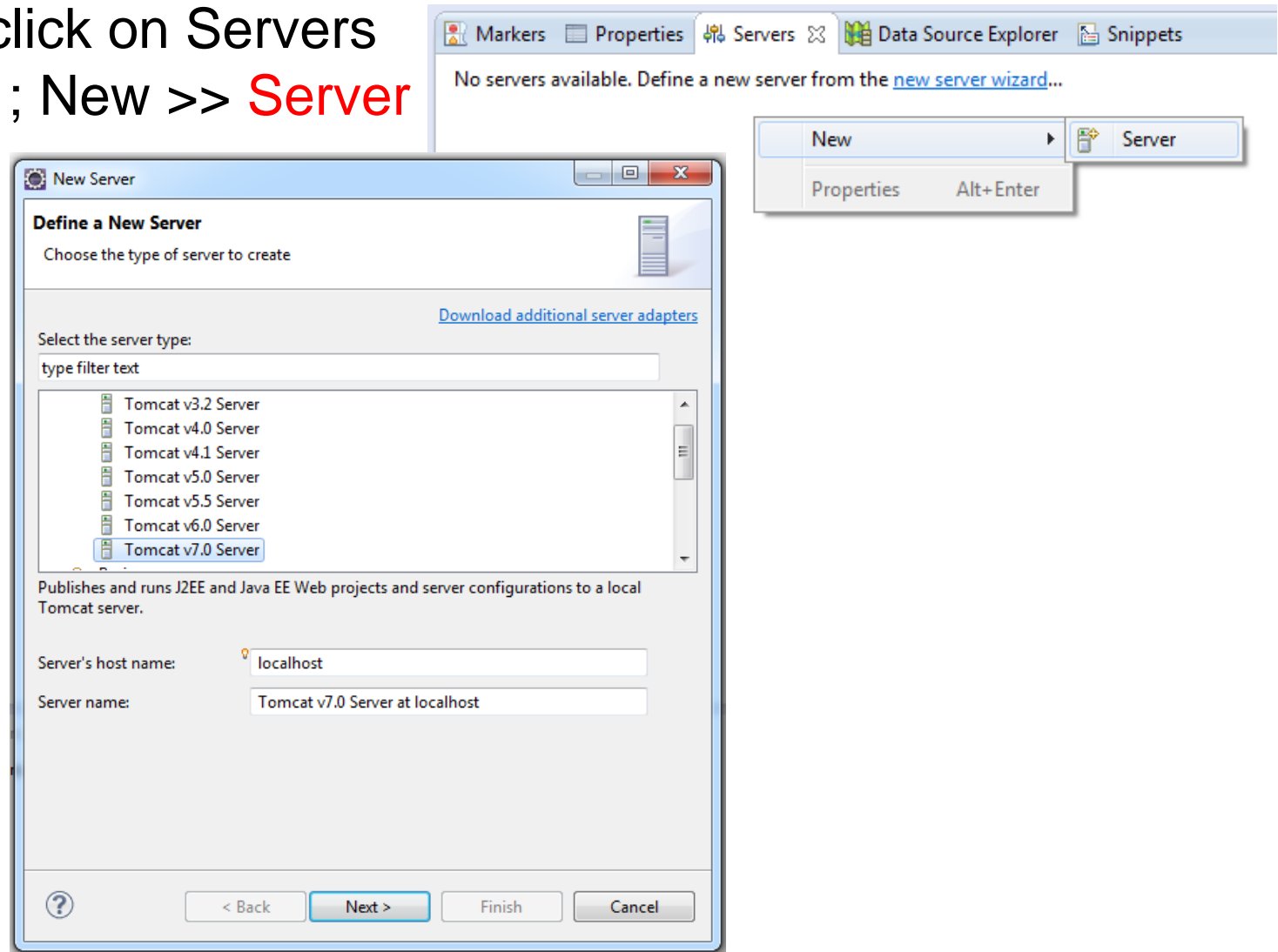
# Setup the development environment

- If you can't see the Servers tab then ;  
Window >> **Show View** and select **Servers**

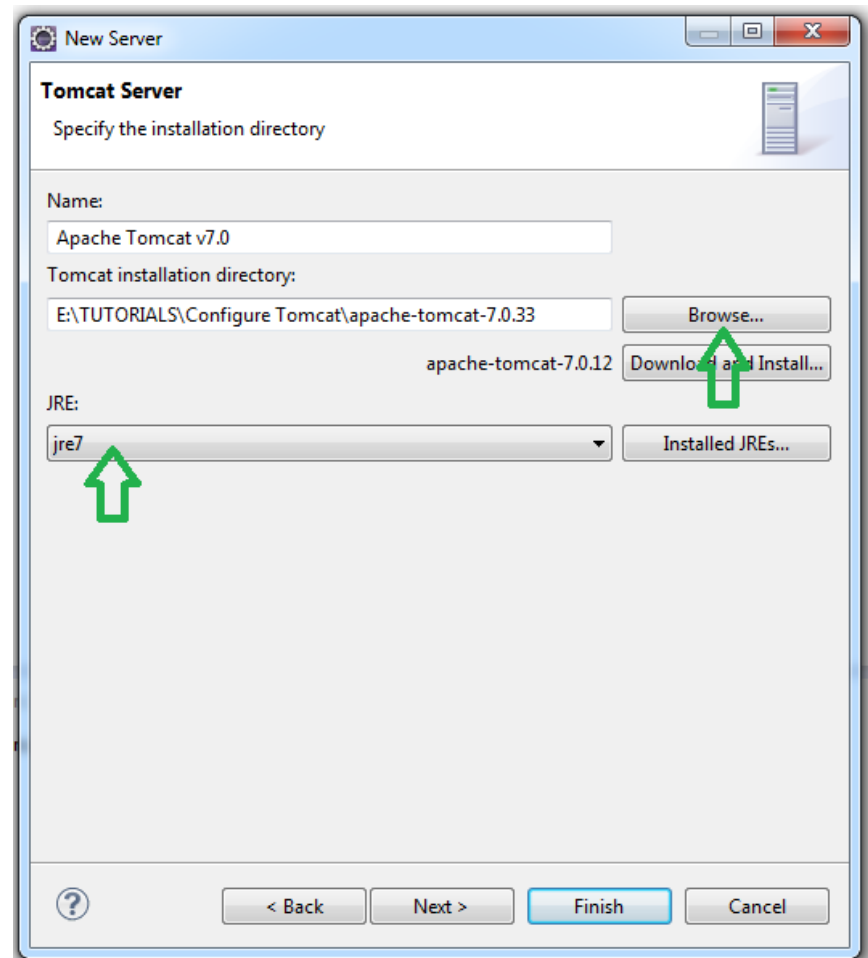


# Setup the development environment

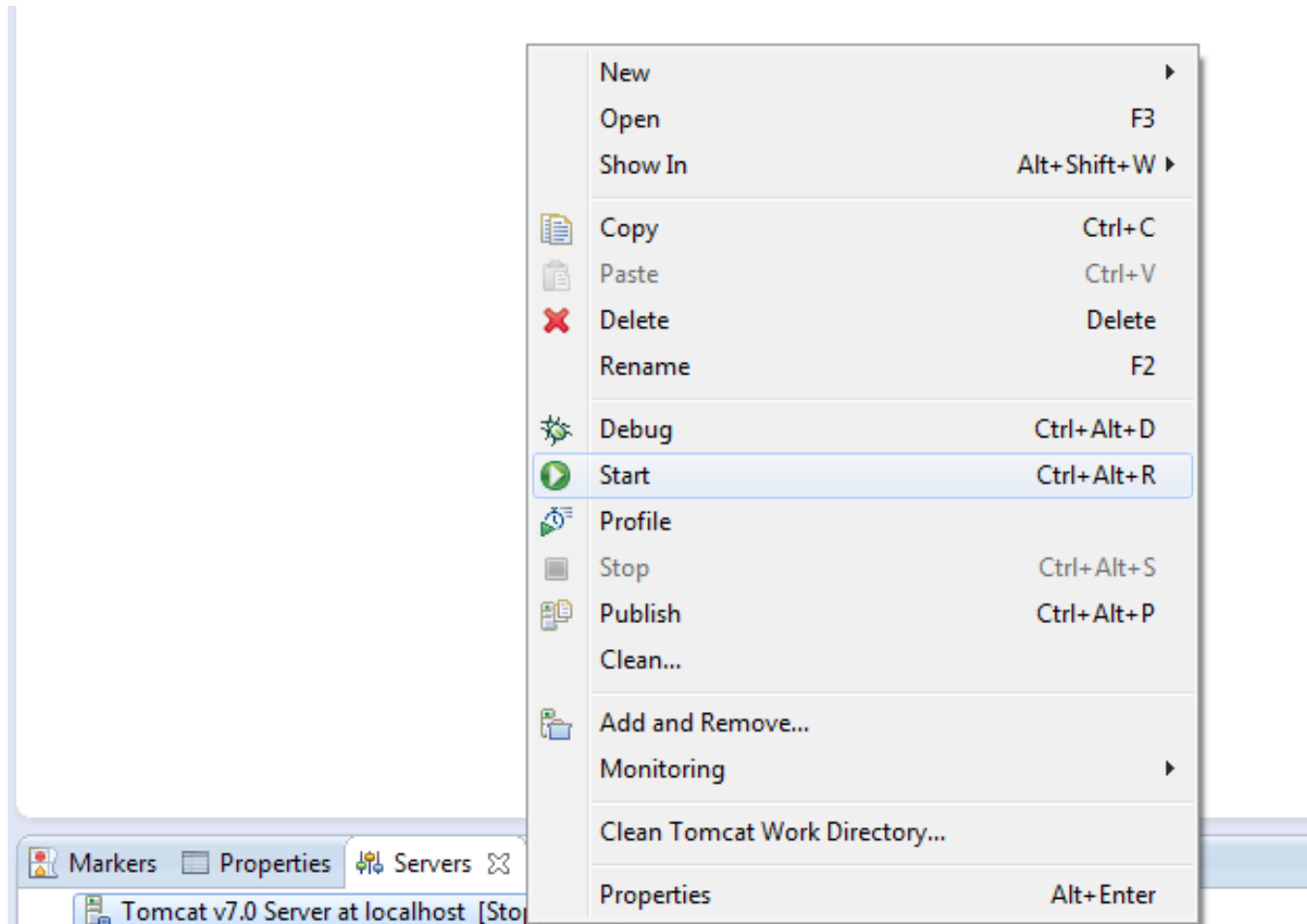
- Right click on Servers  
Select ; New >> **Server**



- Identify **Tomcat installation directory** and select the **jre** that installed in your system and click finish to create the **new server**.

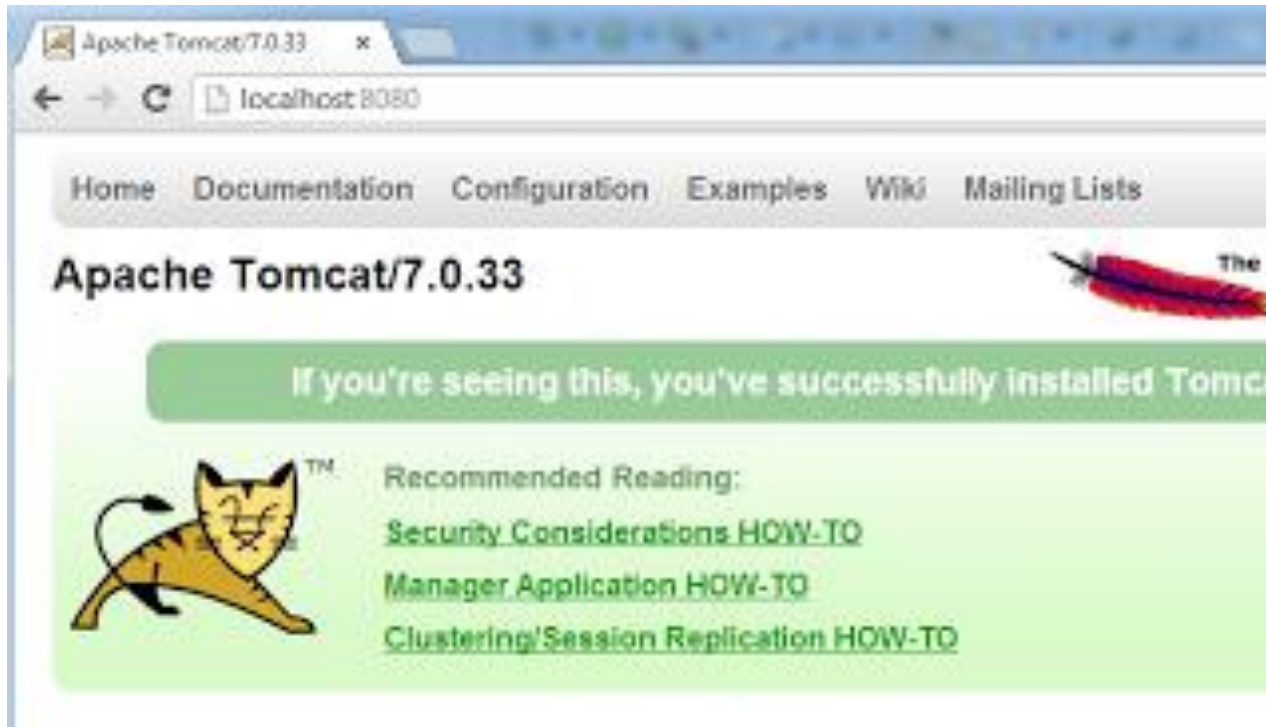


- Start the server right click on Tomcat v7 server in Servers tab and select Start.

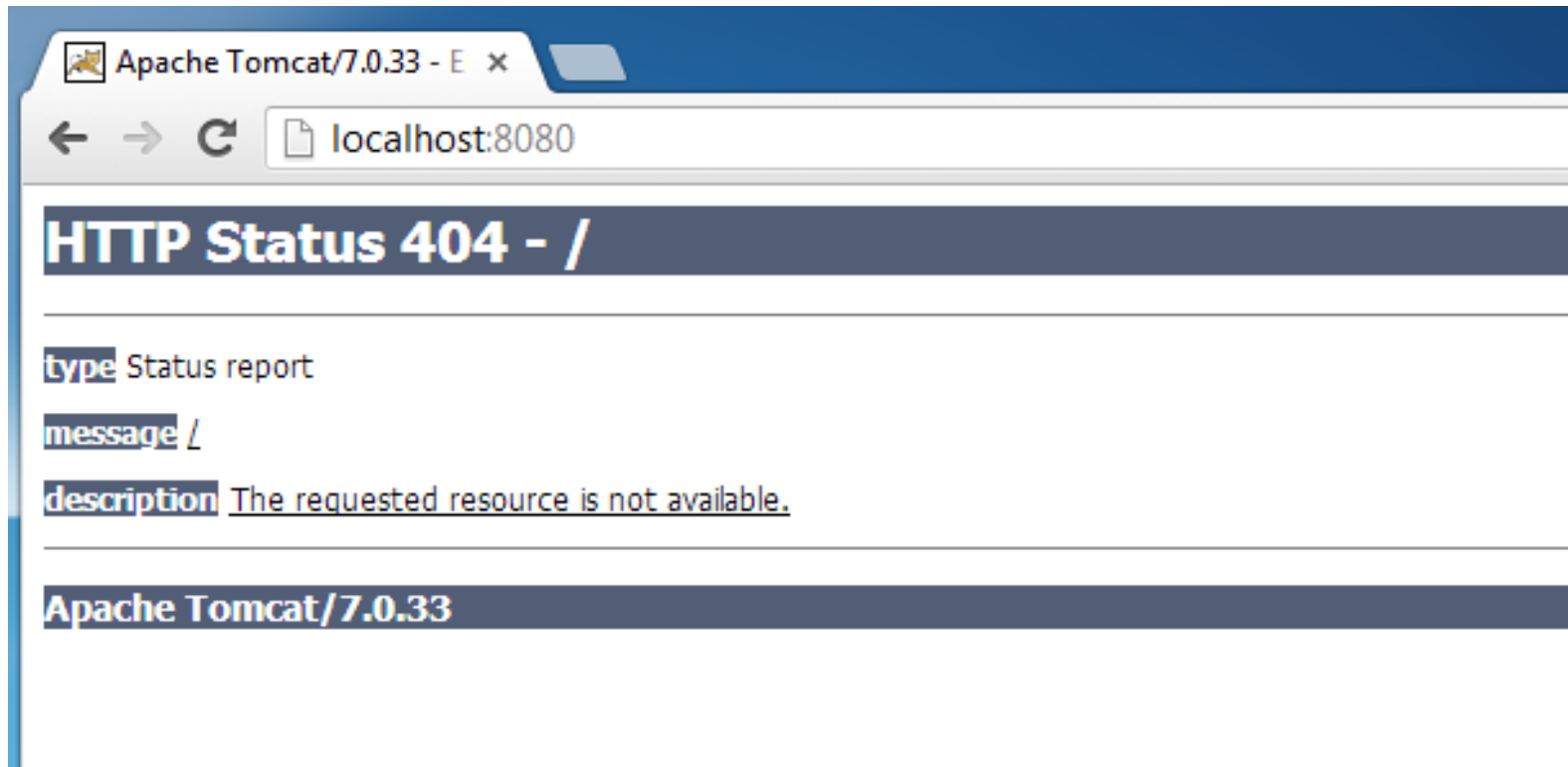




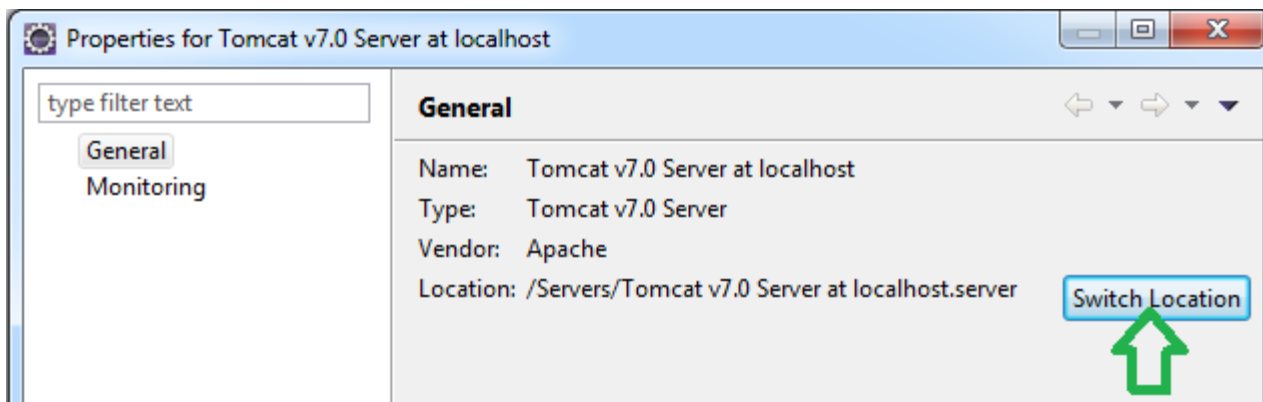
- To check the server open Chrome and go to **localhost:8080** you should be able to see Tomcat start page.



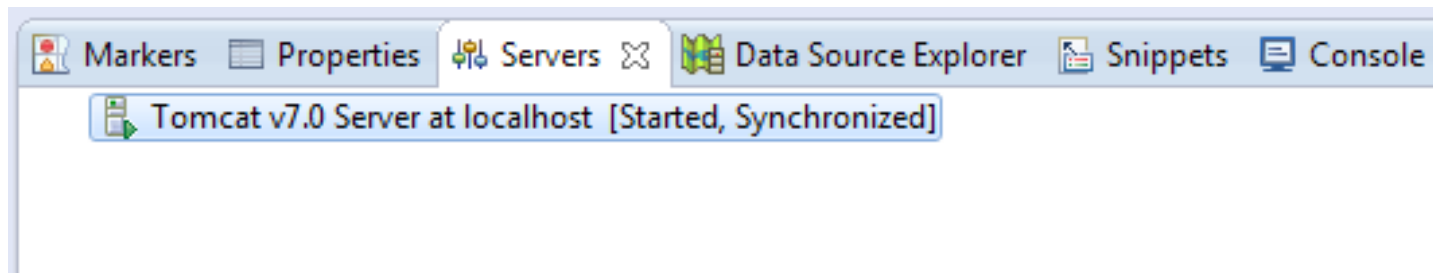
- If you are getting "**requested resource is not available**" error you need to change the file location of the Tomcat.



- In order to do that right click on server in the Servers tab and select properties. Then from the server properties window click **Switch Location and Apply, OK.**



- Next double click on the server to open **Overview.**



- In the Overview window change **Server locations** to **Use Tomcat installation** and save changes.

\*Tomcat v7.0 Server at localhost

### Overview

**General Information**  
Specify the host name and other common settings.

Server name: Tomcat v7.0 Server at localhost

Host name: localhost

**Runtime Environment:** Apache Tomcat v7.0

Configuration path: /Servers/Tomcat v7.0 Server at loca

[Open launch configuration](#)

**Server Locations**  
Specify the server path (i.e. catalina.base) and deploy path. Server must be published with no modules present to make changes.

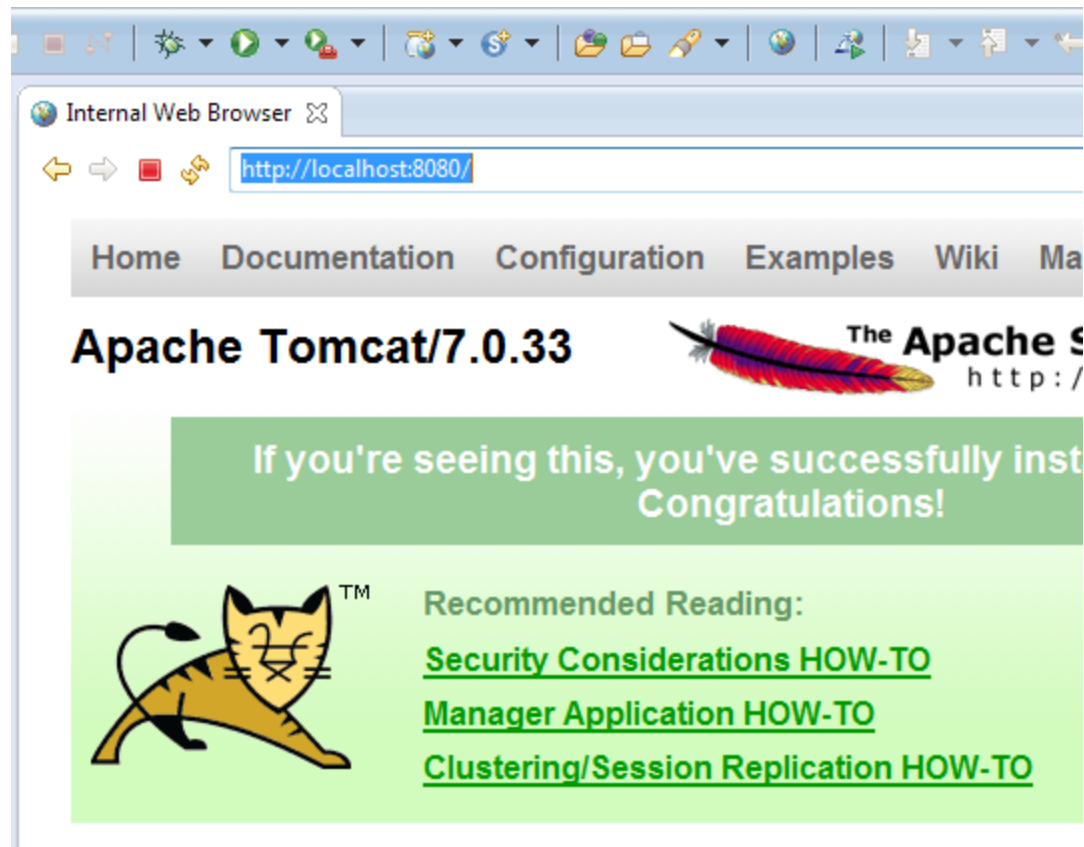
Use workspace metadata (does not modify Tomcat installation)

Use Tomcat installation (takes control of Tomcat installation)

Use custom location (does not modify Tomcat installation)

Overview Modules

- To open Eclipse Internal web browser select **Window >> Show View >> Other >> General >> Internal web Browser**




# Setup the development environment

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

## Apache Tomcat/7.0.59

The Apache Software Foundation  
<http://www.apache.org/>

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:

- [Security Considerations HOW-TO](#)
- [Manager Application HOW-TO](#)
- [Clustering/Session Replication HOW-TO](#)

Server Status  
Manager App  
Host Manager

### Developer Quick Start

- [Tomcat Setup](#)
- [Realms & AAA](#)
- [Examples](#)
- [Servlet Specifications](#)
- [First Web Application](#)
- [JDBC DataSources](#)
- [Tomcat Versions](#)

### Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 7.0 access to the manager application is split between different users.  
[Read more...](#)

- [Release Notes](#)
- [Changelog](#)
- [Migration Guide](#)
- [Security Notices](#)

### Documentation

- [Tomcat 7.0 Documentation](#)
- [Tomcat 7.0 Configuration](#)
- [Tomcat Wiki](#)

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

- [Tomcat 7.0 Bug Database](#)
- [Tomcat 7.0 JavaDocs](#)
- [Tomcat 7.0 SVN Repository](#)

### Getting Help

#### FAQ and Mailing Lists

The following mailing lists are available:

- [tomcat-announce](#)  
Important announcements, releases, security vulnerability notifications. (Low volume).
- [tomcat-users](#)  
User support and discussion
- [taglibs-user](#)  
User support and discussion for [Apache Taglibs](#)
- [tomcat-dev](#)  
Development mailing list, including commit messages

localhost:8080/manager/status

Documentation Get Involved Miscellaneous Apache Software Foundation

9:57 PM 2/17/2015

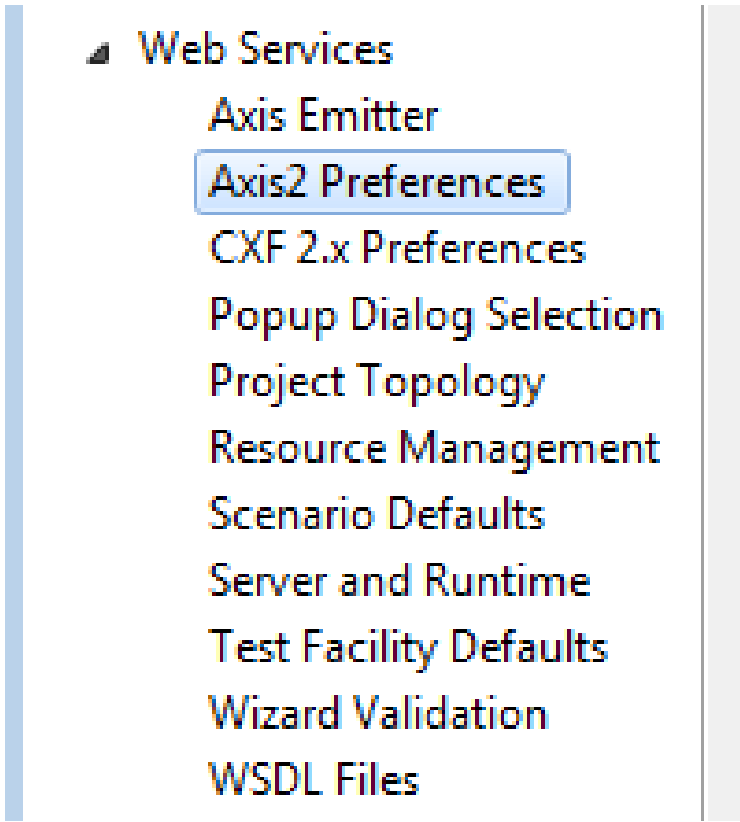
# Create java web service in Eclipse using Axis2

- First you need to download and extract Apache Axis2 SOAP engine.

Apache Axis2 Binary Distribution

# Configure Axis2

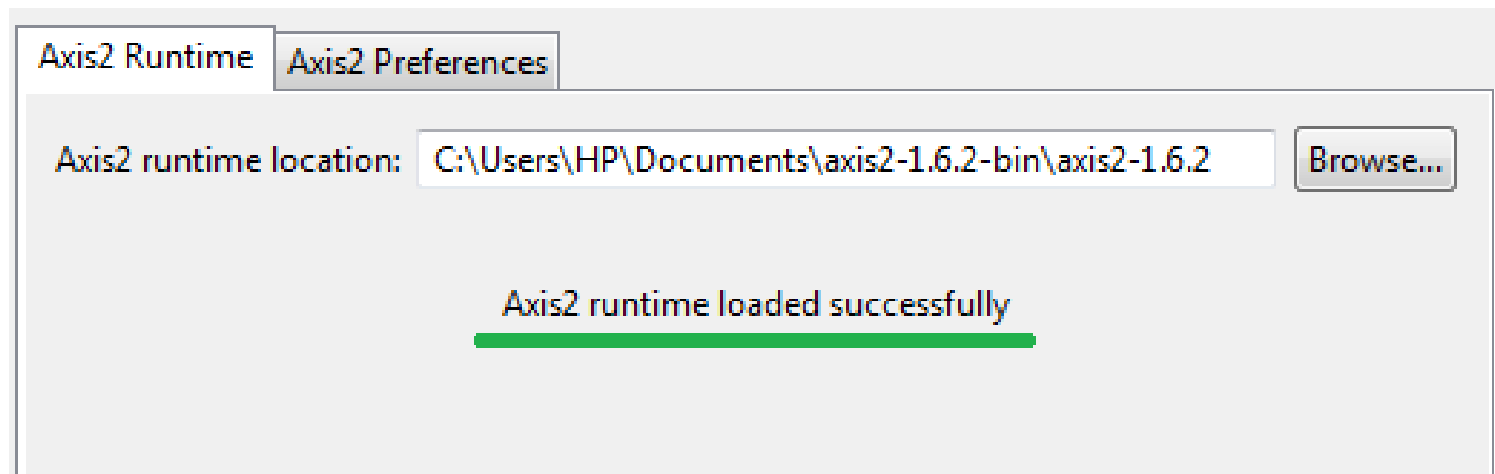
- In Eclipse go to **Window >> Preferences >> Web Services >> Axis2 Preferences**



▲ Web Services  
Axis Emitter  
Axis2 Preferences  
CXF 2.x Preferences  
Popup Dialog Selection  
Project Topology  
Resource Management  
Scenario Defaults  
Server and Runtime  
Test Facility Defaults  
Wizard Validation  
WSDL Files



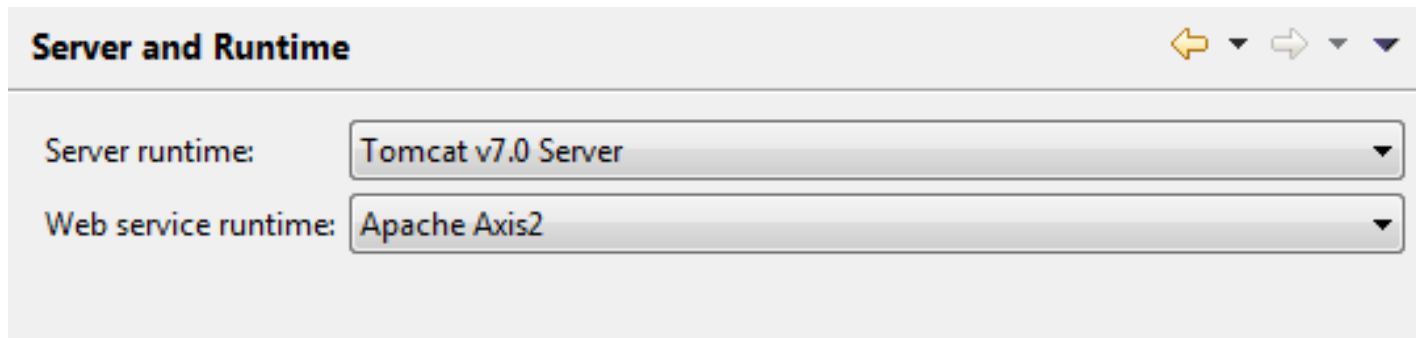
- Then Browse to axis2 extracted location. If you configure correctly Eclipse will show this message "**Axis2 runtime loaded successfully**" then click on **Apply** button.



- To set server & runtime in the same Preferences window click on **Server & Runtime**.

- ▲ Web Services
  - Axis Emitter
  - Axis2 Preferences
  - CXF 2.x Preferences
  - Popup Dialog Selection
  - Project Topology
  - Resource Management
  - Scenario Defaults
  - Server and Runtime**
  - Test Facility Defaults
  - Wizard Validation
  - WSDL Files

- Then select Tomcat 7 as **Server runtime** & Axis2 as **Web service runtime**.

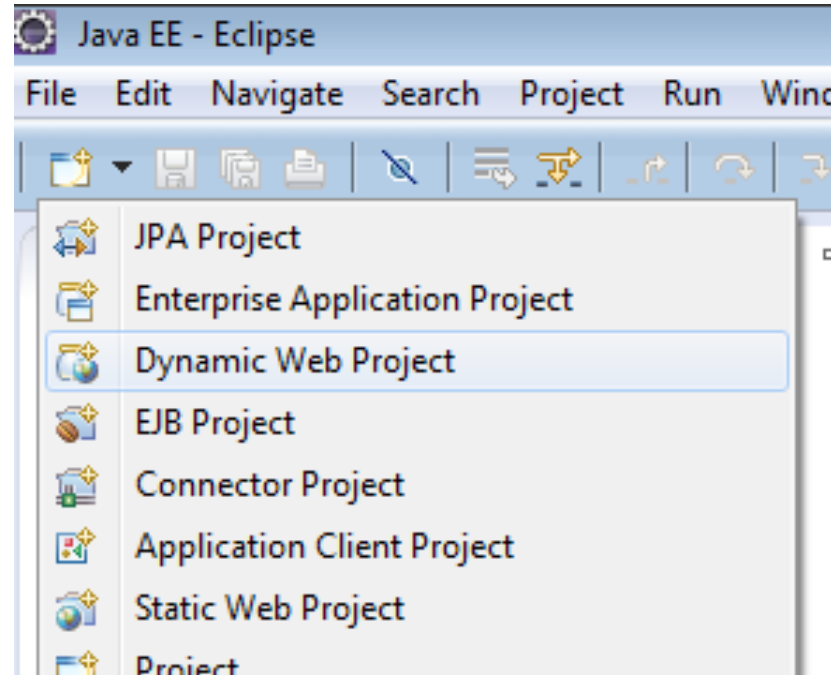


The screenshot shows a dialog box titled "Server and Runtime". It contains two dropdown menus. The first dropdown, labeled "Server runtime:", is set to "Tomcat v7.0 Server". The second dropdown, labeled "Web service runtime:", is set to "Apache Axis2". There are navigation arrows in the top right corner of the dialog box.

- To complete configurations finally click **Apply** and **OK**.

# Creating the Dynamic web project

- In Eclipse EE create new Dynamic web project. Give project name



- Select following settings

**Dynamic Web Project**  
Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location

Use default location

Location:

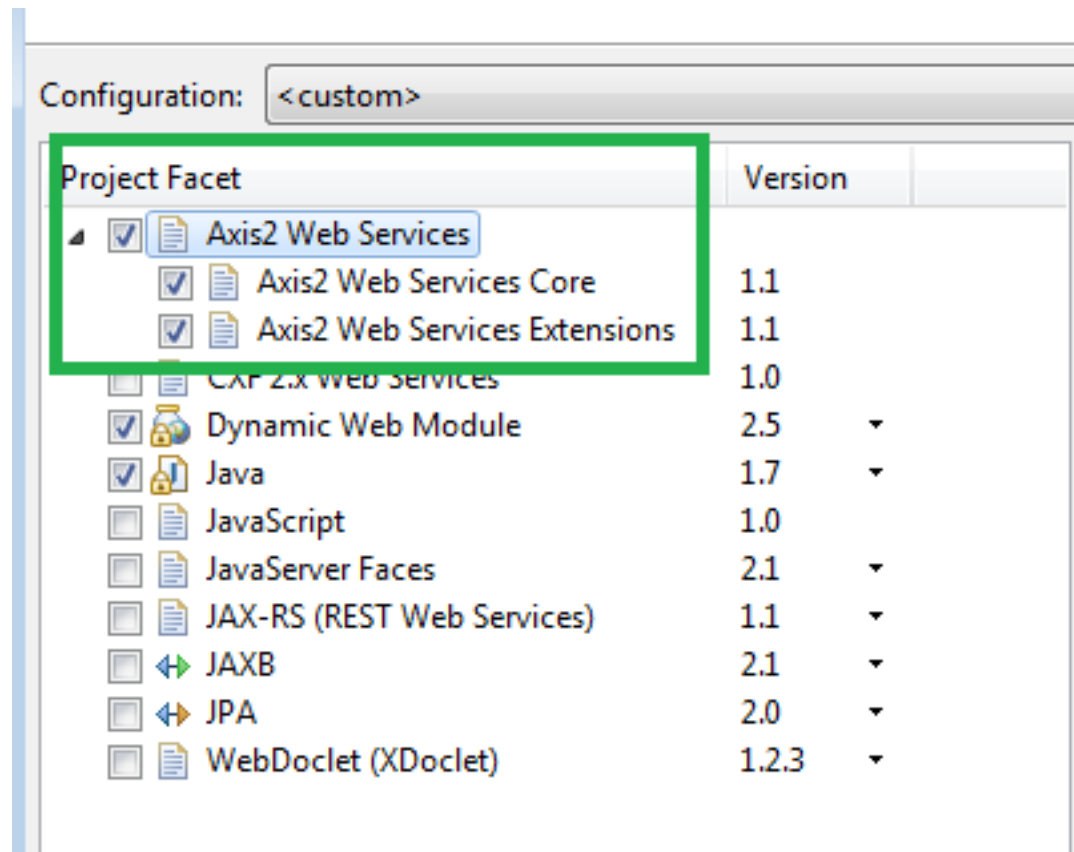
Target runtime

Dynamic web module version

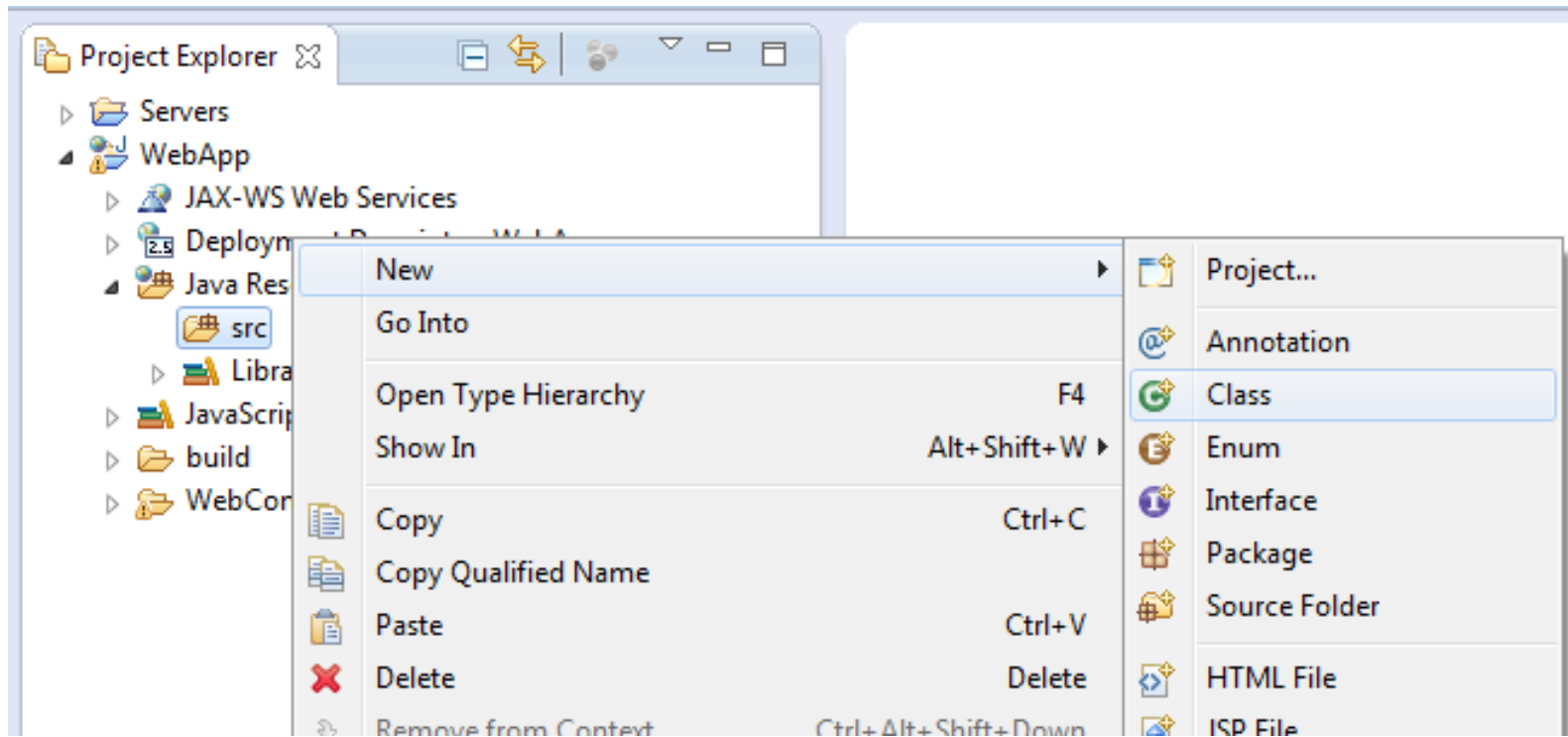
Configuration

Hint: Get started quickly by selecting one of the pre-defined project configurations.

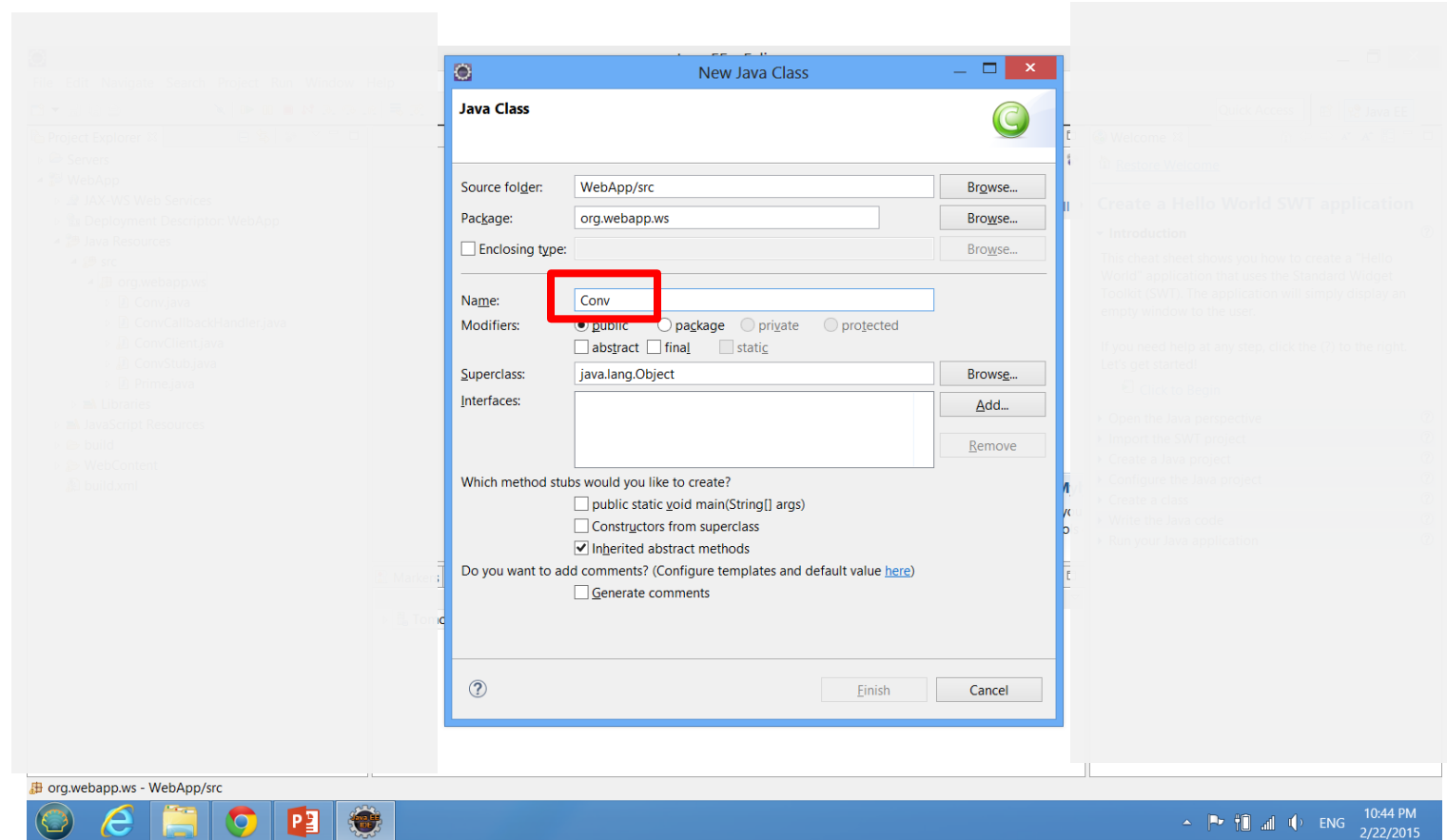
- Select Axis2 web services in opening window.



- Right click on **src** folder then **New >> Class**



- Give class name as well as package name





- Add following code to the **class**

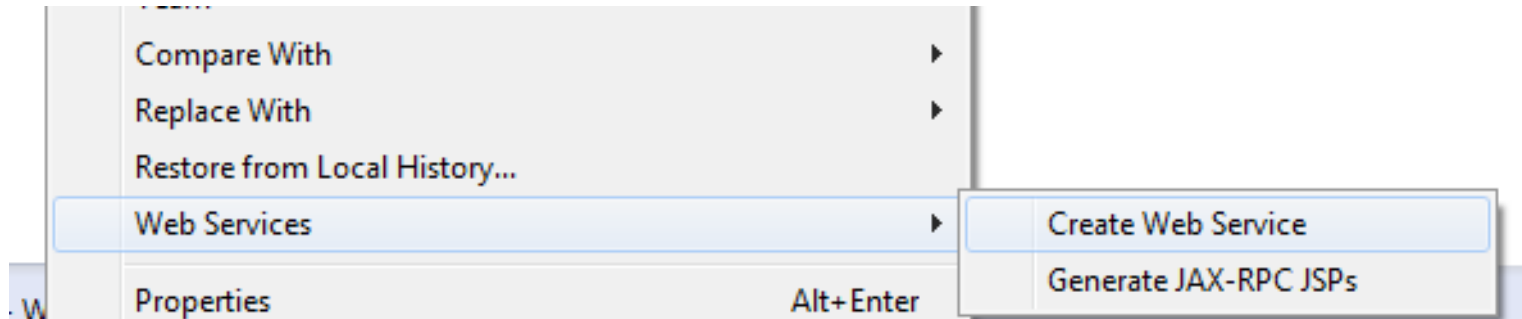
The screenshot shows the Eclipse IDE interface. The main editor window displays the following Java code in `Conv.java`:

```
1 package org.webapp.ws;  
2  
3 public class Conv {  
4     public double ftoC(double x)  
5     {  
6         return (5.0/9.0)*(x-32.0);  
7     }  
8  
9     public double ctoF(double y)  
10    {  
11        return ((9.0/5.0)*y)+32.0;  
12    }  
13 }
```

The Project Explorer on the left shows the project structure: `WebApp` > `src` > `org.webapp.ws`. The Console window at the bottom shows `Tomcat v7.0 Server at localhost [Stopped]`. The right-hand side of the IDE displays a 'Welcome' page with a 'Create a Hello World SWT application' tutorial.

# Creating the web service

- Right click on your service class (not the project).  
Select **Web Services >> Create Web Service**



- In Web Service window tick to the Monitor the Web service check box. Other default settings are **OK**. No need to change those to this web service.
- Click **Next**.

Web Service

### Web Services

Select a service implementation or definition and move the slider

Web service type:

Service implementation:

**Start service** Configurat  
[Server run](#)  
[Web servic](#)  
[Service pr](#)

**No client** Configurat

Client type:

Publish the Web service

Monitor the Web service

Do not show me this dialog box again.

- Next window, select Generate a default services.xml file.

Axis2 Web Service Java Bean Configuration

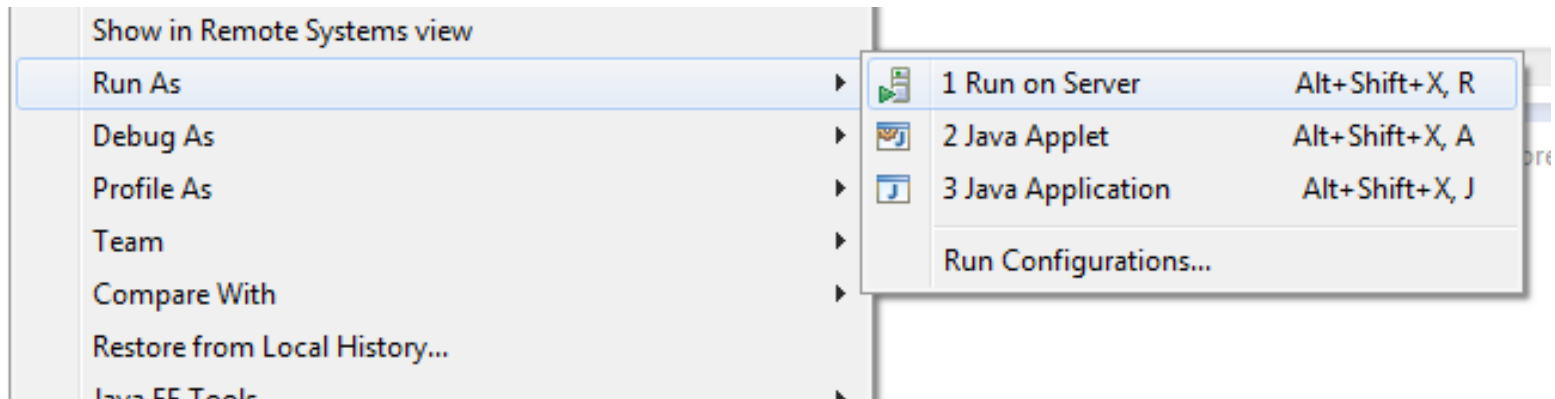
Use an existing services.xml file

Generate a default services.xml file

- Then click Next and finally click **Finish**.

# Testing the project

- Perfect..... Now you should have a web service. To set the service Right click on your project and select **Run As >> Run on Server**.



- Click **next & click Finish**.

- If everything went fine you should see following page in Eclipse internal web browser.

The Apache Software Foundation  
<http://www.apache.org/>

AXIS2

### Welcome!

Welcome to the new generation of Axis. If you can see this page you have successfully deployed the Axis2 Web Application. However, to ensure that Axis2 is properly working, we encourage you to click on the validate link.

- [Services](#)  
View the list of all the available services deployed in this server.
- [Validate](#)  
Check the system to see whether all the required libraries are in place and view the system information.
- [Administration](#)  
Console for administering this Axis2 installation.

Copyright © 1999-2006, The Apache Software Foundation  
Licensed under the [Apache License, Version 2.0](#).

7:53 PM  
2/23/2015

- Select **Conv** which is your web service class name. Under **Available Operations** you can see your web service operation.

Conv

Service Description : Please Type your service description here

Service EPR : <http://localhost:8080/WebApp/services/Conv>

Service Status : Active

*Available Operations*

- ctoF
- ftoc

Version

Service Description : Version

Service EPR : <http://localhost:8080/WebApp/services/Version>

- After clicking on **Conv** you can see the WSDL file. Copy WSDL link to the clip board.

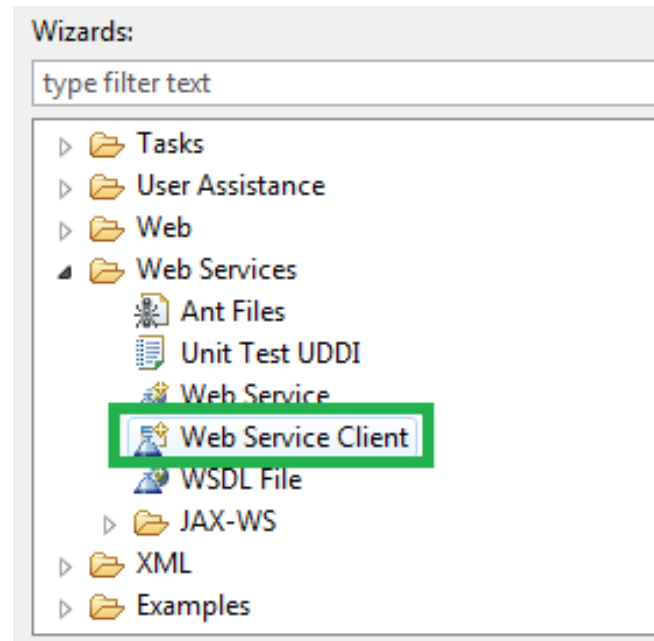
After clicking on **Conv** you can see the WSDL file. Copy WSDL link to the clip board.

The screenshot shows a web browser window with the address bar containing the URL `localhost:8080/WebApp/services/Conv?wsdl`. A red box highlights this URL, and a red arrow points to it with the text "Copy this". Below the address bar, the browser displays the WSDL file content, which is an XML document. The XML content is as follows:

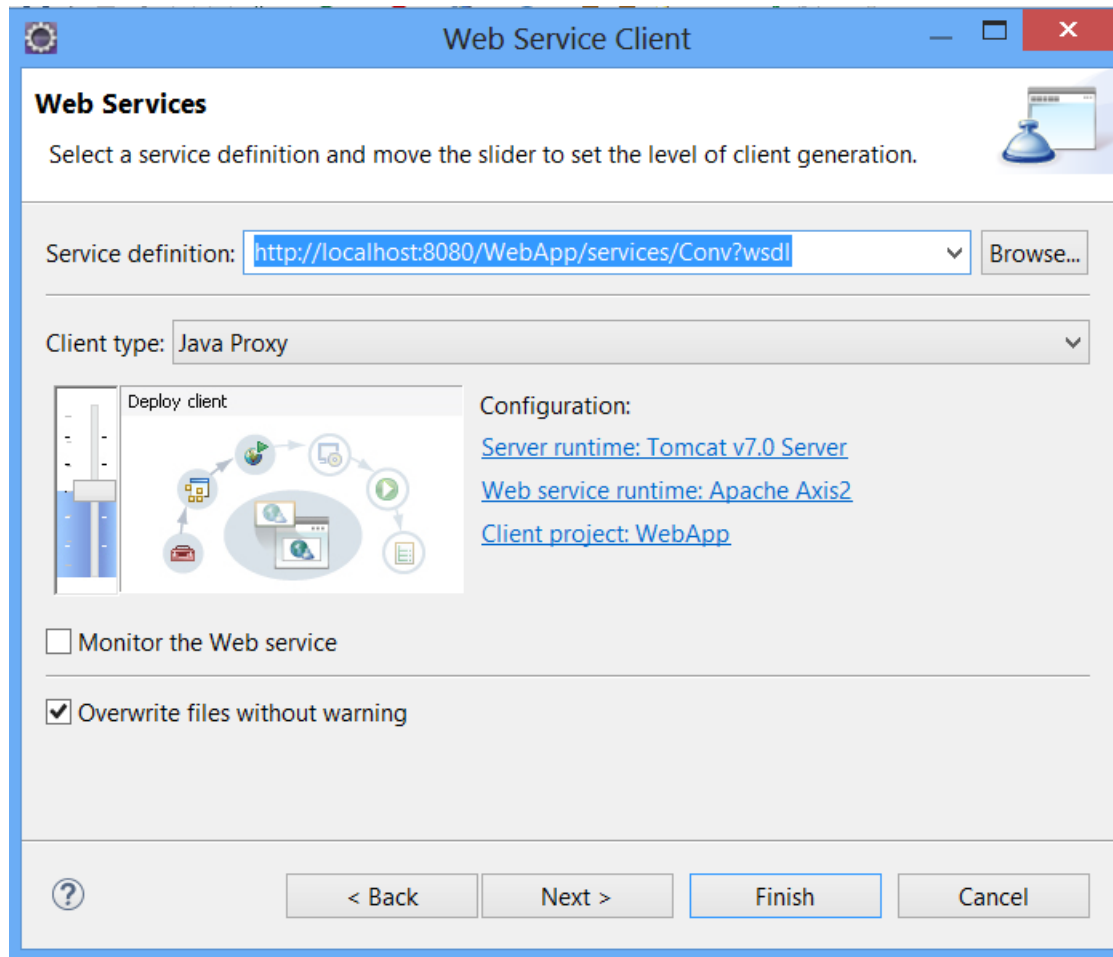
```
<?xml version="1.0" encoding="UTF-8" ?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:ns1="http://org.apache.axis2/xsd" xmlns:ns="http://ws.webapp.org"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
targetNamespace="http://ws.webapp.org">
  <wsdl:documentation>Please Type your service description here</wsdl:documentation>
  <wsdl:types>
    <xs:schema attributeFormDefault="qualified" elementFormDefault="qualified" targetNamespace="http://ws.webapp.org">
      <xs:element name="ftoC">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="x" type="xs:double"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="ftoCResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="return" type="xs:double"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="ctoF">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="y" type="xs:double"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="ctoFResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="return" type="xs:double"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:message name="ctoFRequest">
    <wsdl:part name="return" type="xs:double"/>
  </wsdl:message>
  <wsdl:message name="ctoFResponse">
    <wsdl:part name="return" type="xs:double"/>
  </wsdl:message>
  <wsdl:portType name="ftoC">
    <wsdl:operation name="ftoC">
      <wsdl:input message="ctoFRequest" type="http://schemas.xmlsoap.org/wsdl/http:POST"/>
      <wsdl:output message="ctoFResponse" type="http://schemas.xmlsoap.org/wsdl/http:POST"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="ftoCSoap12" type="ftoC">
    <wsdl:soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  </wsdl:binding>
  <wsdl:binding name="ftoCSoap" type="ftoC">
    <wsdl:soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  </wsdl:binding>
  <wsdl:binding name="ftoCHttp" type="ftoC">
    <wsdl:http:binding style="application" transport="http://schemas.xmlsoap.org/soap/http"/>
  </wsdl:binding>
  <wsdl:service name="ftoC">
    <wsdl:port name="ftoCSoap12" binding="ftoCSoap12" location="http://localhost:8080/WebApp/services/Conv?wsdl" style="document"/>
    <wsdl:port name="ftoCSoap" binding="ftoCSoap" location="http://localhost:8080/WebApp/services/Conv?wsdl" style="document"/>
    <wsdl:port name="ftoCHttp" binding="ftoCHttp" location="http://localhost:8080/WebApp/services/Conv?wsdl" style="application"/>
  </wsdl:service>
</wsdl:definitions>
```



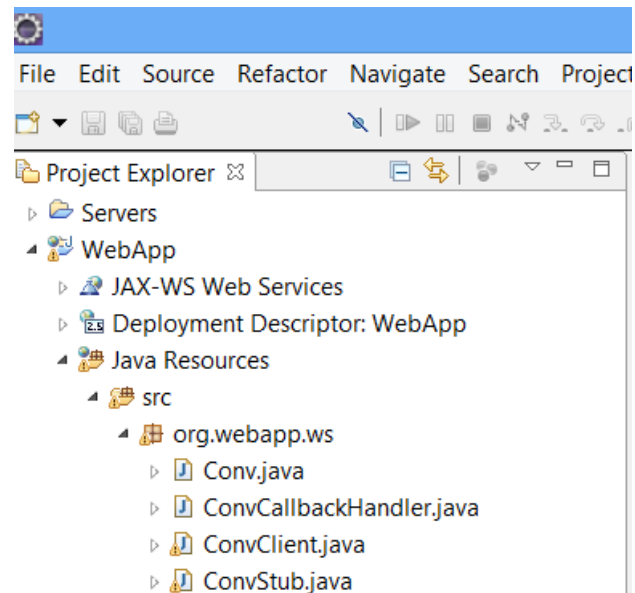
- Now we have to create the web service client. Right click the web service class select **New >> Other >> Web Services >> Web service Client.**



- Click Next. Paste link of the wsdl file to the **Service definition** box.



- Then create new class.



- Paste this code to newly created class. This client class also in the same package with web service class.

```
Conv.java ConvClient.java x
1 package org.webapp.ws;
2 import org.webapp.ws.ConvStub.FtoC;
3 //import java.util.Scanner;
4
5 public class ConvClient {
6     public static void main(String[] args) throws Exception {
7         ConvStub stub = new ConvStub();
8         // Scanner sc=new Scanner(System.in);
9         //System.out.println("Enter the value of Fahrenheit");
10        //double n = sc.nextDouble();
11
12        //Creating the request
13        org.webapp.ws.ConvStub.FtoC request = new org.webapp.ws.ConvStub.FtoC();
14        request.setX(-40.0);
15
16        //Invoking the service
17        org.webapp.ws.ConvStub.FtoCResponse response = stub.ftoC(request);
18        System.out.println("FtoC is " + response.get_return());
19    }
20 }
```

- Right click Client class and **run** the client as a java application.

