ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

# Εισαγωγή στα Δίκτυα Υπηρεσιών

## Assisting Lecture 9 - Top Down SOAP Web Services

Μύρων Παπαδάκης
Τμήμα Επιστήμης Υπολογιστών

# Introduction to Service Networks
# CS-592 – Spring 2015

Assisting Lecture: Top- Down WS with Apache Tomcat, Axis2, Eclipse

Myron Papadakis (myrpap@gmail.com)

# Outline

- Axis2, Tomcat, Eclipse
- Top-Down Web Services
  - Area rectangle calculator
  - Area rectangle calculator client
- Bottom-up Web Services
- SoapUI


# Axis2

- Axis2 is a **SOAP processing engine** with the main function of delivering incoming SOAP messages into correct applications (every application is assumed to be a Web Service)
- Generate code from both service clients (=stubs) and service providers (=skeletons)
  - **WSDL2Java** tool: code generation from WSDL
  - **Java2WSDL**: generate WSDL from code
- *Data binding* is the term used for techniques that handle the conversion between XML and application data structures
- Axis2 provides a variety of data binding frameworks
  - ADB (Axis2 Data Binding): Limitations only for Axis2 Web services
  - XMLBeans
  - JiBX
  - None (AXIOM)
    - AXIOM is Axis2 document model used for XML Message handling
- More information at: http://www.ibm.com/developerworks/webservices/library/ws-java3/

# Axis2 Data Binding

- The primary objective of any data binding framework is to map the XML schema constructs to programming language objects,
  - provide the marshaling (i.e., produce an equivalent XML document from a programing language object structure) and
  - unmarshaling (i.e, produce an equivalent programing language object structure from an XML document) capabilities.

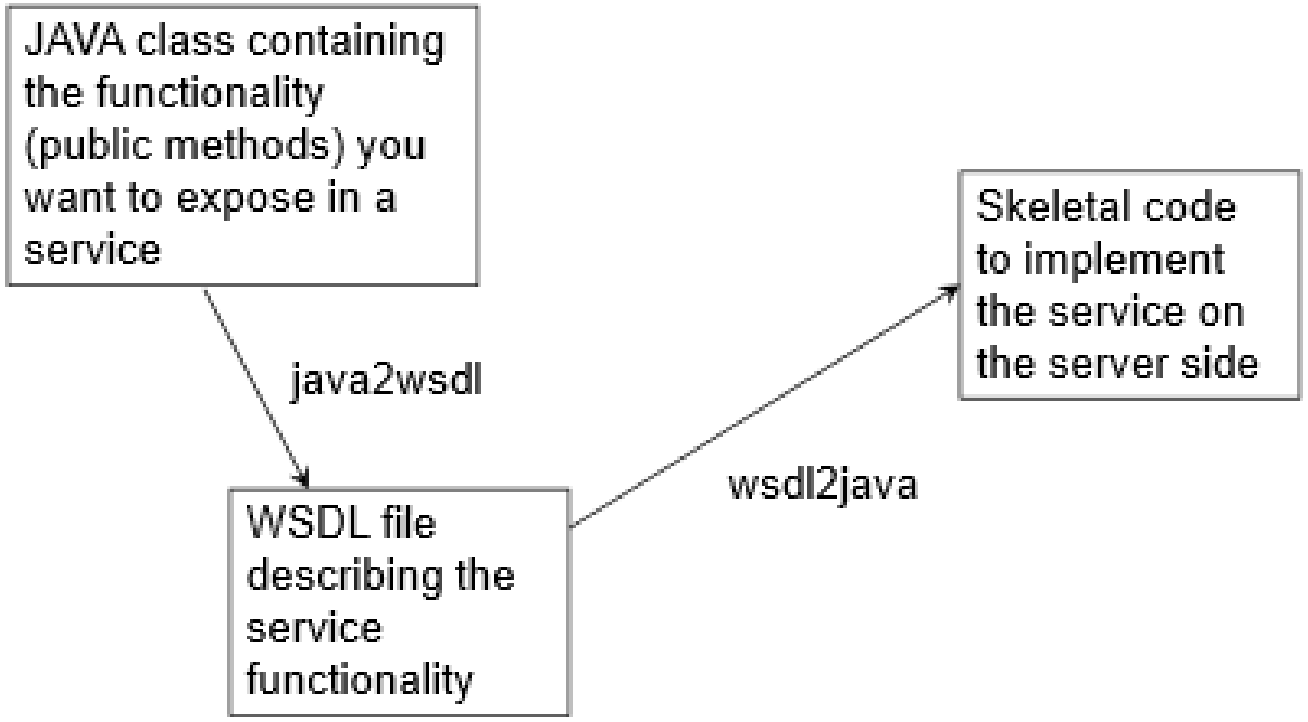
- In fact, ADB is designed to support any programming language with some customized templates and also those who write the programing language.

- Currently it has a full implementation only for the Java programing language, and a relatively less implementation for the C programing language.

# Apache Axis2

- Set of tools to simplify Web Services
  - Services
    - Create a Web Service from any Java class
    - Create Web Service skeletons from WSDL files
    - Build war file for deployment on any Java-based browser
  - Clients
    - Create client stubs from WSDL files
- Usage
  - Eclipse plugins
    - Integrated into Java EE Version of Eclipse
    - Download free from http://www.eclipse.org/downloads/
    - Choose “Eclipse IDE for Java EE Developers”
  - Command-line tools
    - Windows and Unix

# Axis2 > Creating Services

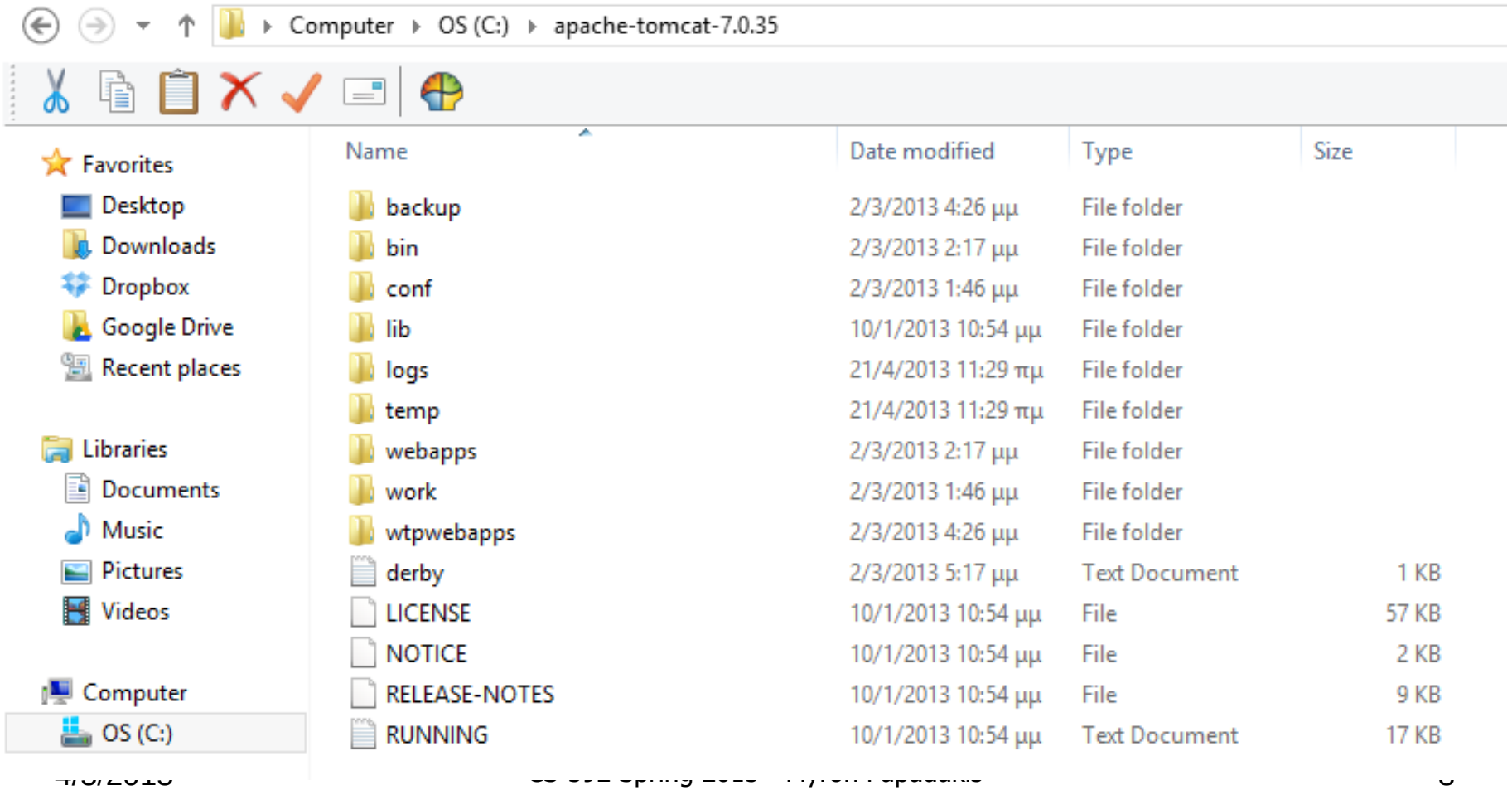


# Apache Tomcat

- Download Apache Tomcat 7
- http://tomcat.apache.org/download-70.cgi
- Building Apache Tomcat requires a JDK to be installed.
- You can download one from http://www.oracle.com/technetwork/java/javase/downloads/index.html or from another JDK vendor.
- Set an environment variable JAVA_HOME to the pathname of the directory into which you installed the JDK release

# Apache Tomcat

- Extract the zip file for example to local disk C:

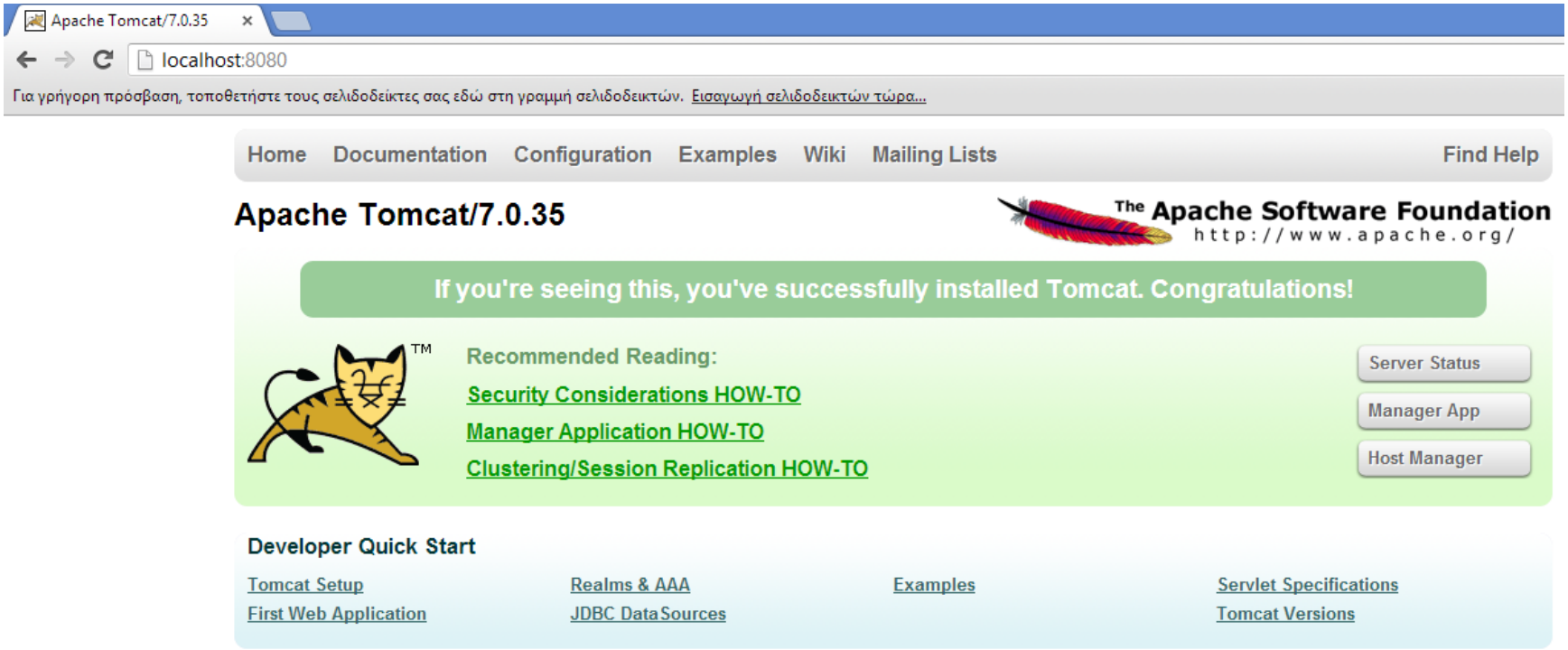


# Apache Tomcat

- Environment Variable: **CATALINA_HOME** to the path of apache tomcat
- Scripts: **startup**, shutdown inside %CATALINA_HOME%/bin
- Test Tomcat: http://localhost:8080/
- **Tomcat instances**
  - if Tomcat has been configured for multiple instances, each instance will have its own CATALINA_BASE configured.

# Test Apache Tomcat

*Tomcat Running on port 8080*



# Multiple Tomcat Instances

- Copy files from the initial tomcat folder to another folder (i.e to C:\tomcat_instance2)
  - Change the following ports of the new instance (conf/server.xml)
  - Shutdown port
  - Connector port
  - Ajp port
  - Redirect port

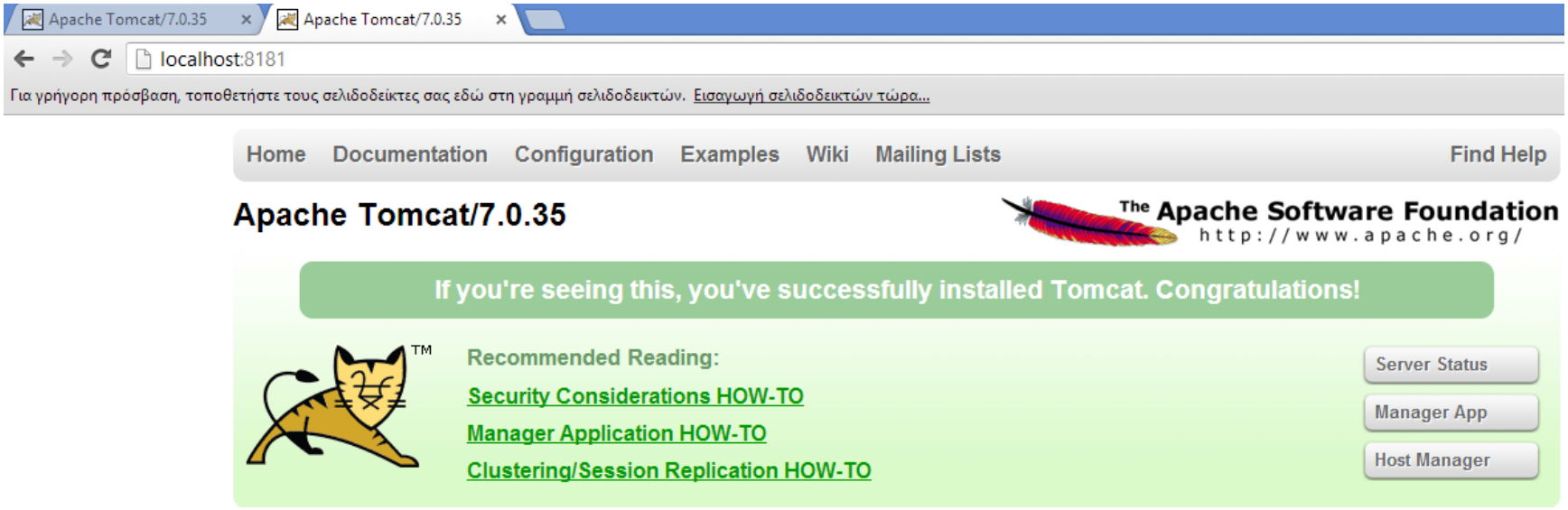
```
21    -->
22    <Server port="8105" shutdown="SHUTDOWN">
```

```
70    <Connector port="8181" protocol="HTTP/1.1"
71               connectionTimeout="20000"
72               redirectPort="81443" />
```

```
89
90        <!-- Define an AJP 1.3 Connector on port 8009 -->
91        <Connector port="8109" protocol="AJP/1.3" redirectPort="81443" />
92
```

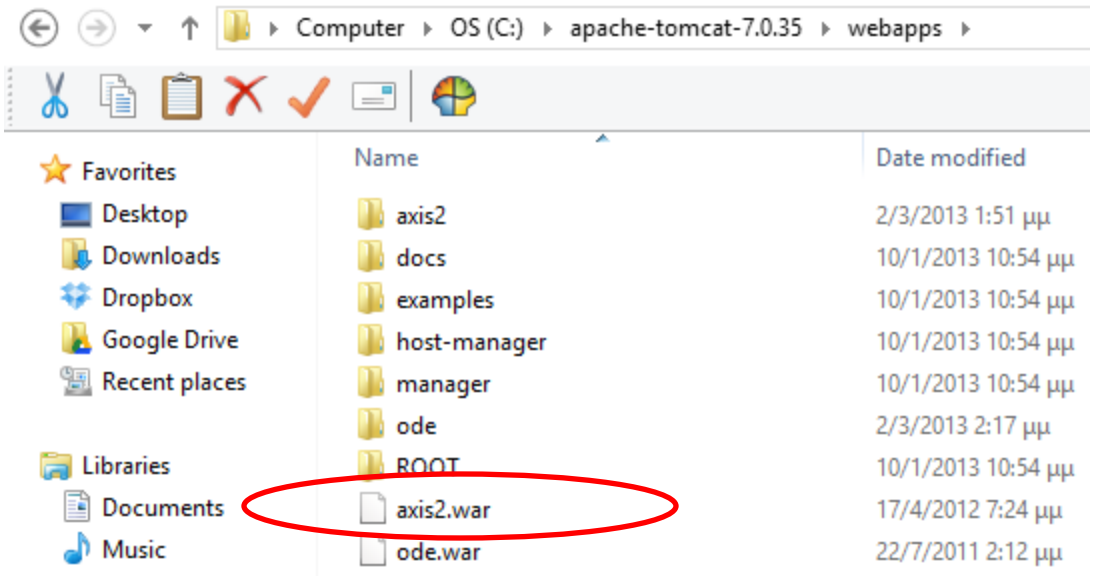
# Testing Additional Tomcat Instances

*2nd Tomcat Instance running on port 8181 (the 1st runs already on port 8080)*



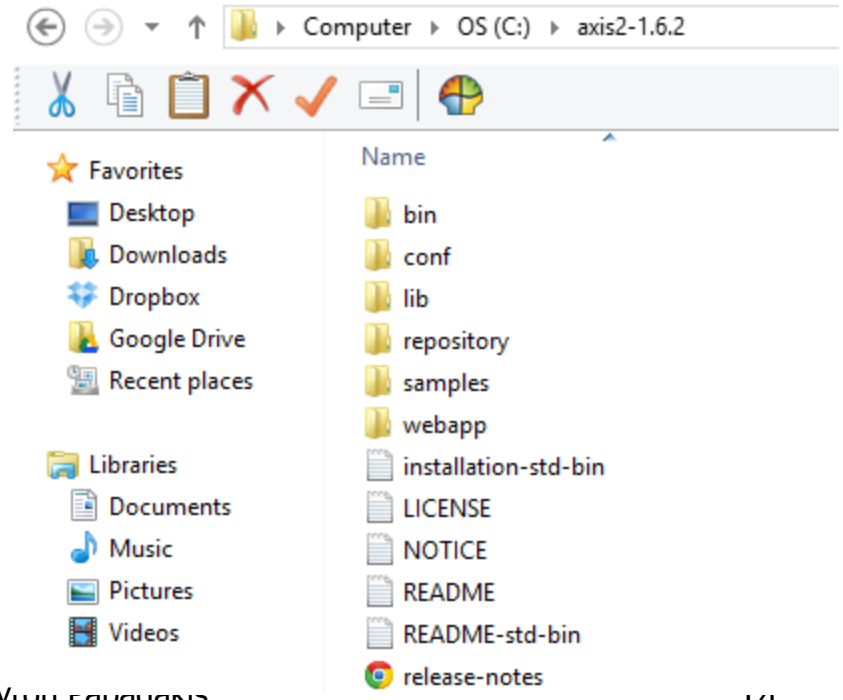
# Axis2 Installation

- http://axis.apache.org/axis2/java/core/download.cgi
- Get the **WAR** distribution for deployment on Tomcat
- Place it into the "*webapps*" folder of tomcat



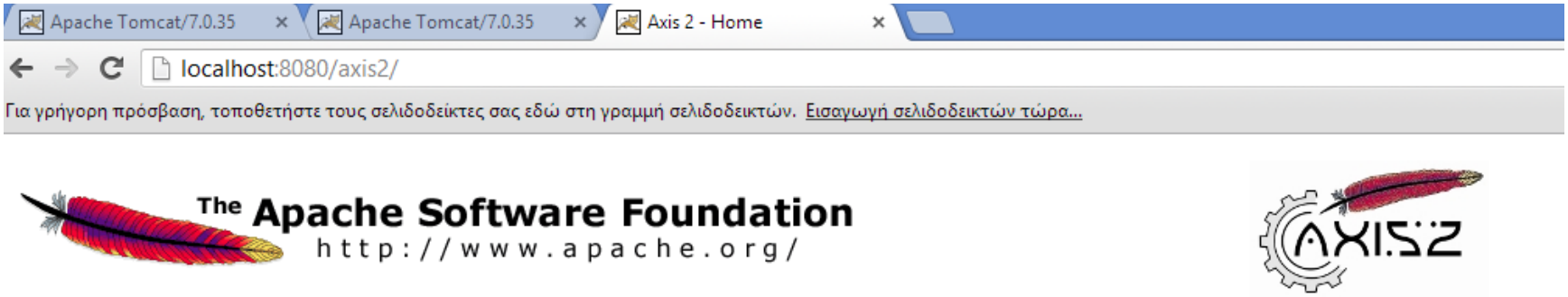
# Axis2 Installation

- Get also the **standard distribution** for service and client programming
- **Choose "zip" under *Binary* Distribution**
- Unzip to folder of your choice, ie "C:\"
- Test Axis2 deployment
- http://localhost:8080/axis2/



# Testing Axis2 Deployment

http://localhost:8080/axis2/

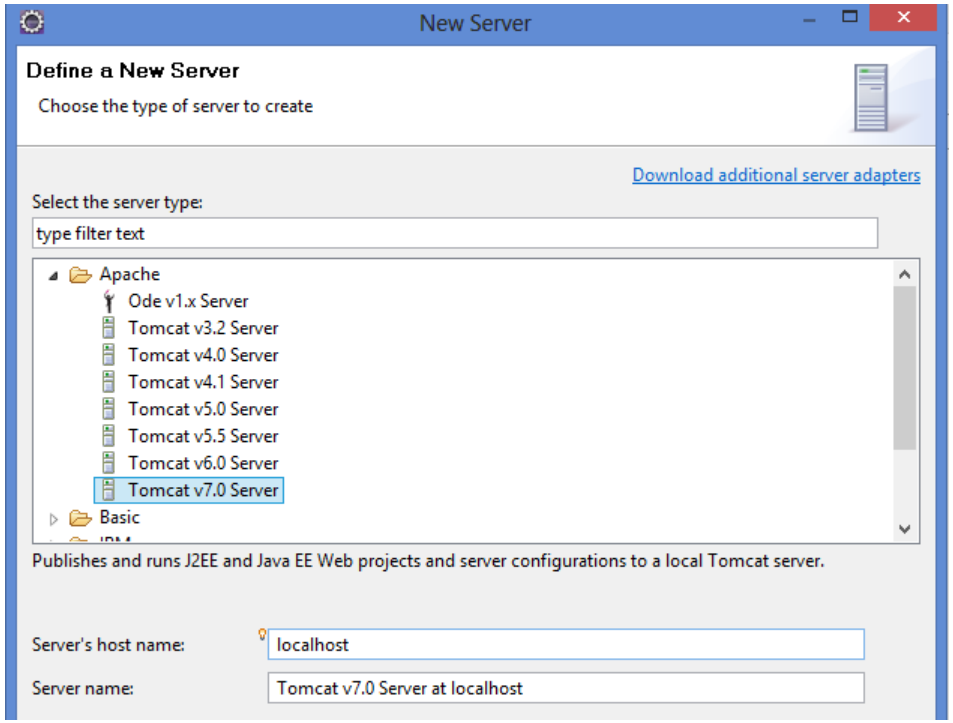


# Eclipse Axis2 Lab Example



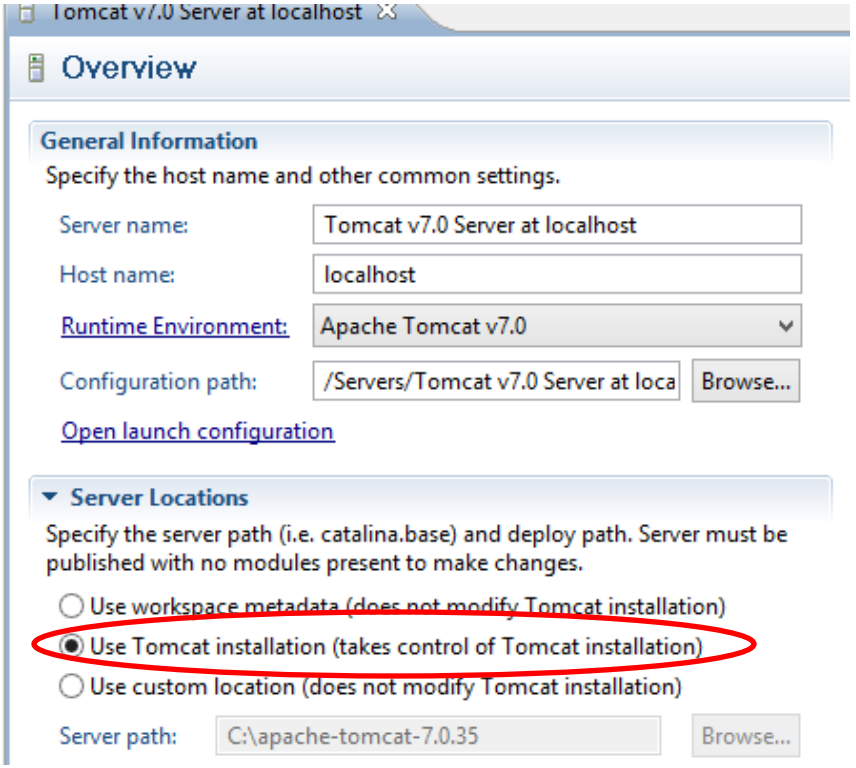
# Eclipse > Add Server

- Download Eclipse for Java EE developers (i.e Helios, Indigo, Juno)
  - Helios is the preferred one
- Window → Show view → Servers → Add server
- Select "***Server Type***" and then under "***Apache***", "***Tomcat VX.0 Server***" .
- Then, define "Tomcat installation Directory", JRE



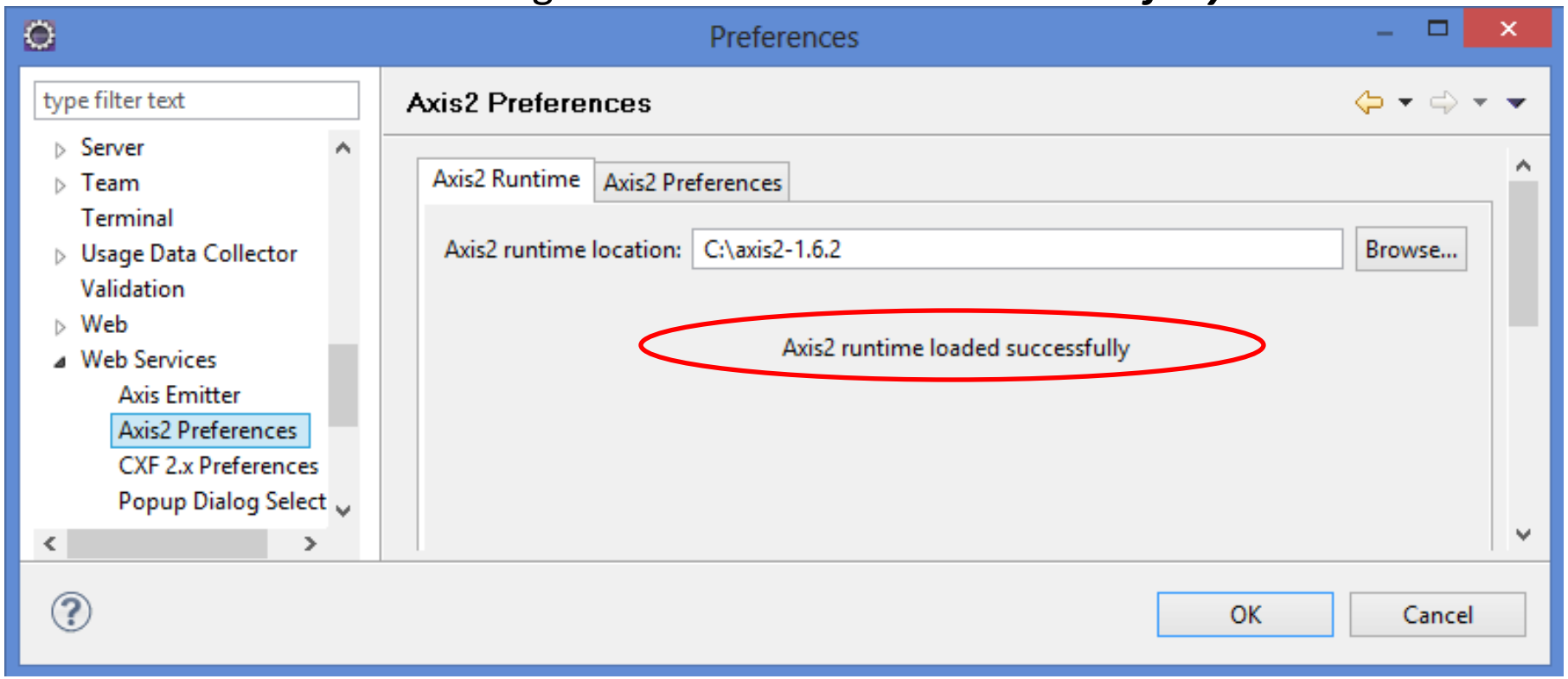
# Eclipse > Add Server

- By default, Eclipse does not modify referenced Tomcat but rather creates an instance of the specified server
- If you want to use the genuine, then select from "**Servers**", right click and select "**Open**".
- Choose "**Use Tomcat Installation**" under "**Server Locations**" and set the server path



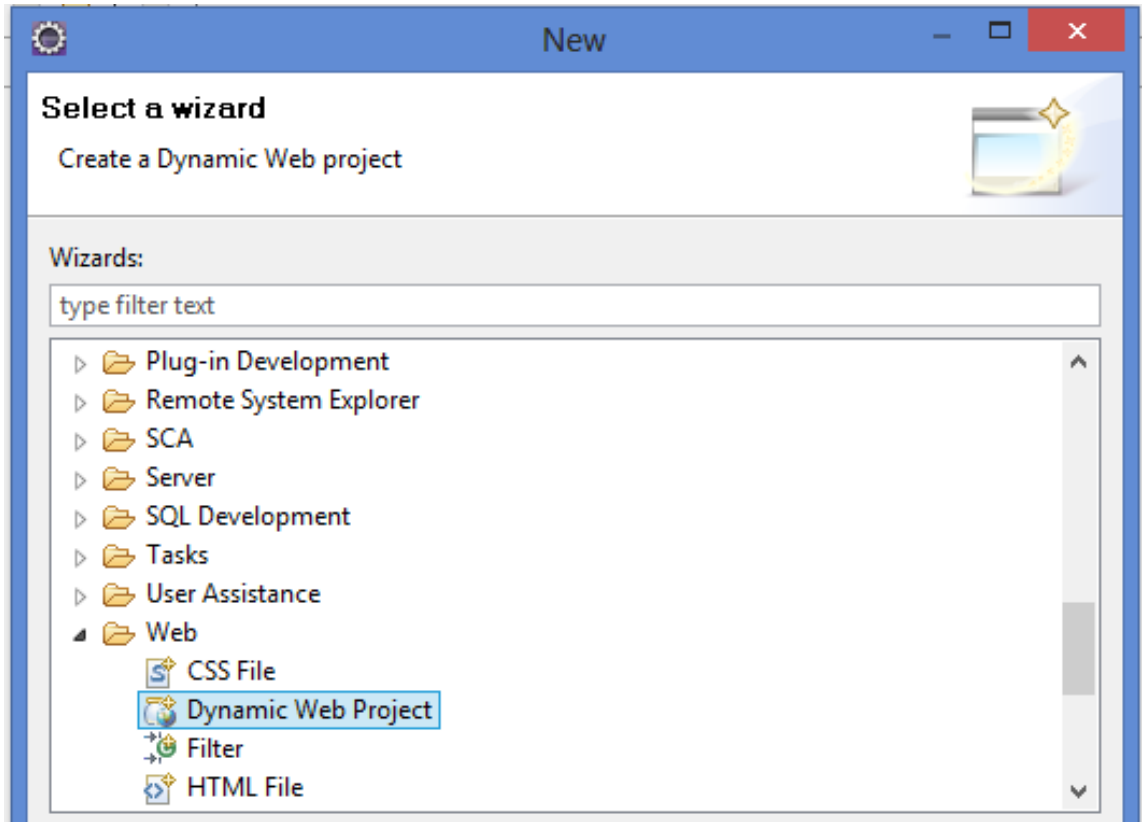
# Tell Eclipse About Axis2

- Window → Preferences → Web Services → Axis2 Preferences
- Set install path of Axis2 Set ***"Axis2 runtime location"*** to the path you have unzipped Axis2 (i.e C:\axis2-1.6.2)
- You will see the message ***"Axis2 runtime loaded succesfully"***.



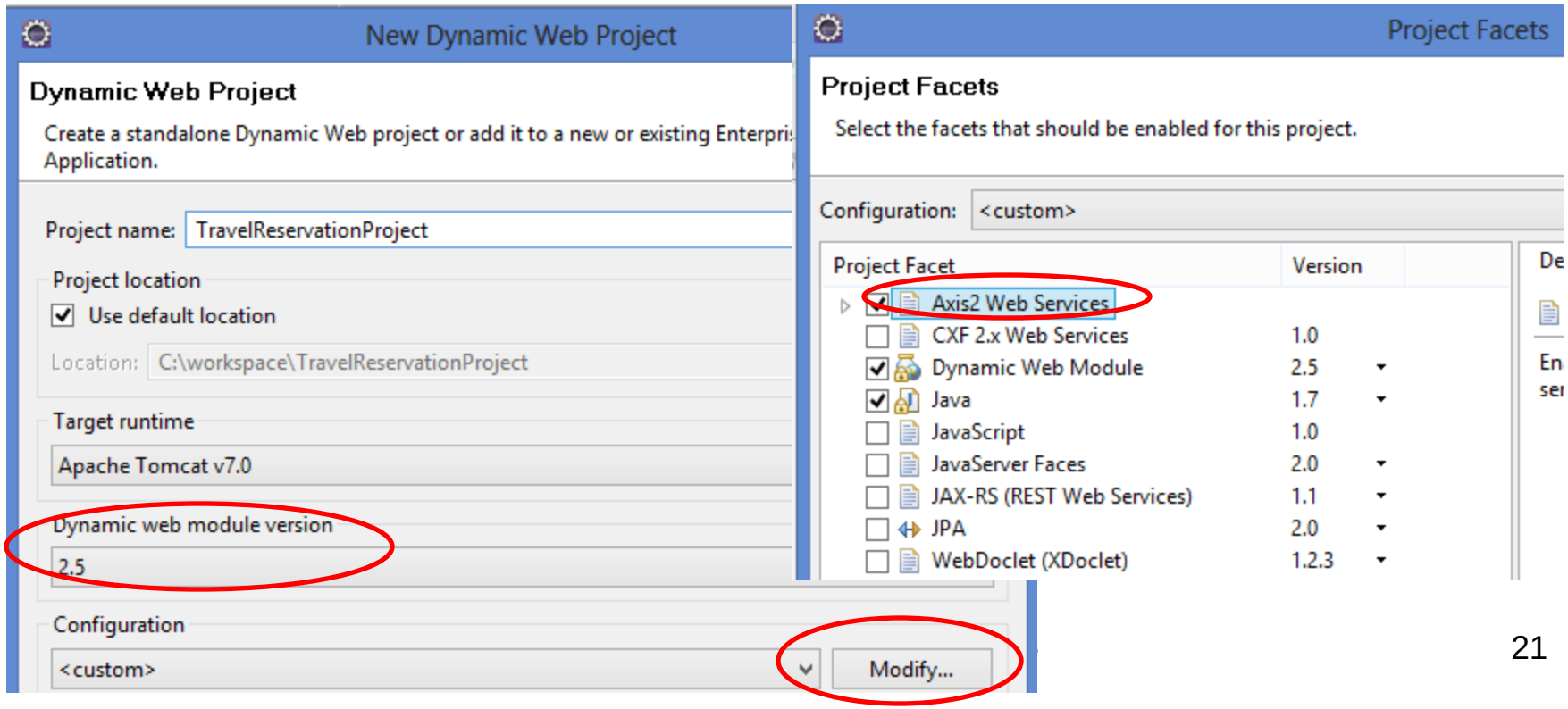
# Eclipse and Axis2 Project

- File → New → Other → Web → Dynamic Web Project



# Eclipse and Axis2 Project

- Set the following:
  - **Target Runtime**: set to the server you want
  - **Dynamic web module version**: Don't use "3", but one of "2.2 ,2.3, 2.4, 2.5"
  - **Under Configuration:** Project Facets add: **"Axis2 Web Services"**



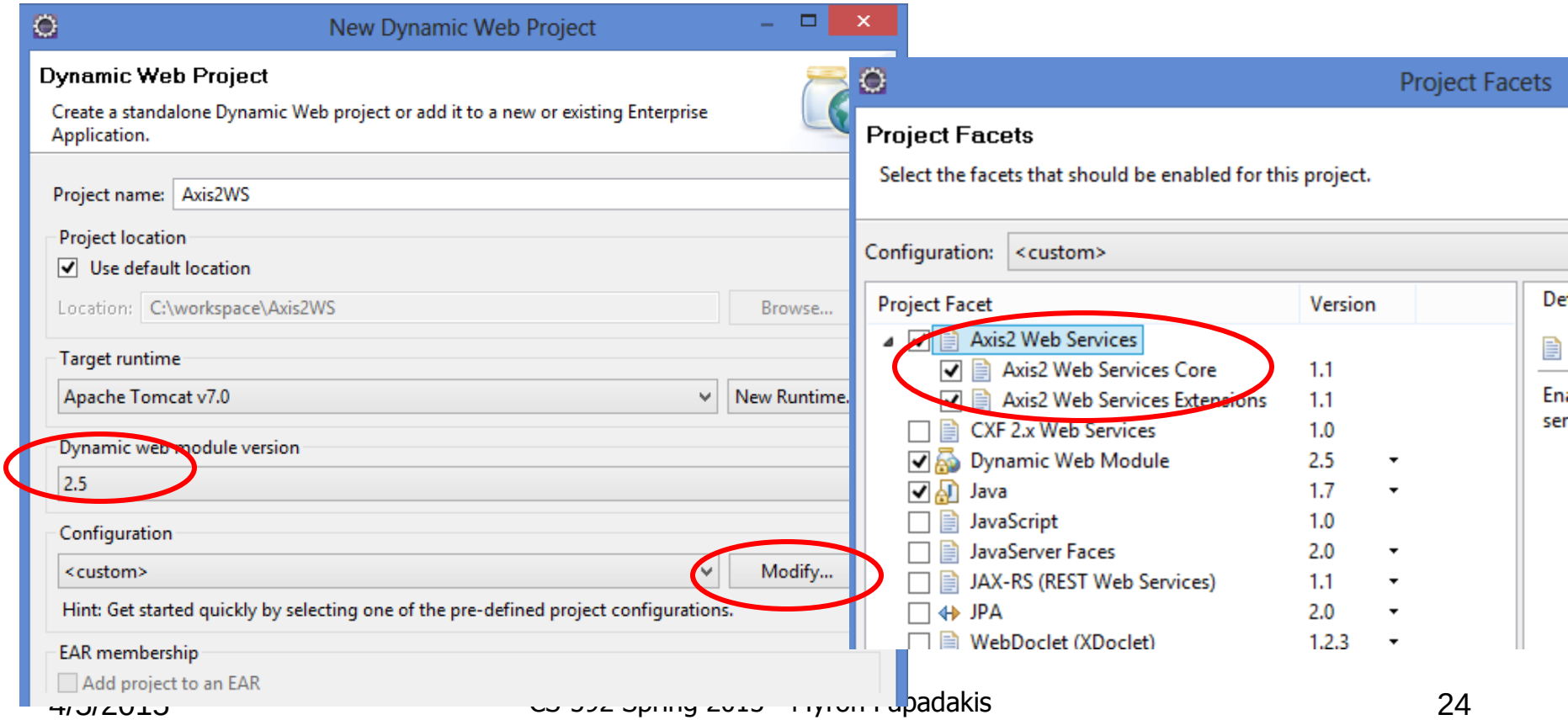
# Eclipse Top-Down WS Example

# Eclipse Lab Example (Top-Down Approach)

- Create a new Web Service that calculates the area of a rectangle. We need a single operation (i.e. calculateRectangleArea)
  - Input: width and height (float).
    - Use a complexType (i.e Dimension) for the input
  - Output: float
- Use the top-down for building this service (first we must define the WSDL file for this service)
- Test the WS using a client (Web Services Explorer, manually written client or SOAPUI)

# Eclipse Lab Example (Area Calculation)

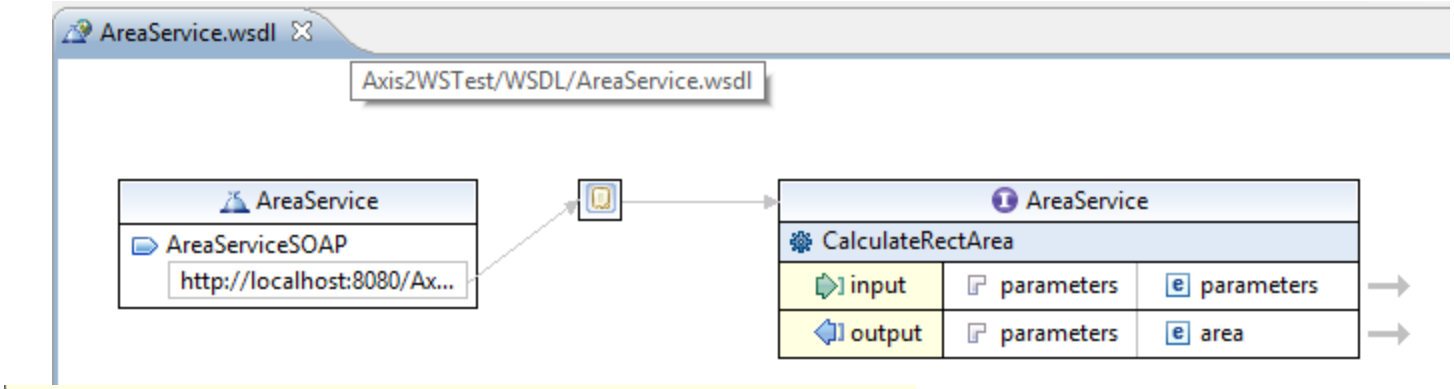
- Create a project with the support of Axis2 features. Open **File -> New -> Other... -> Web -> Dynamic Web Project**

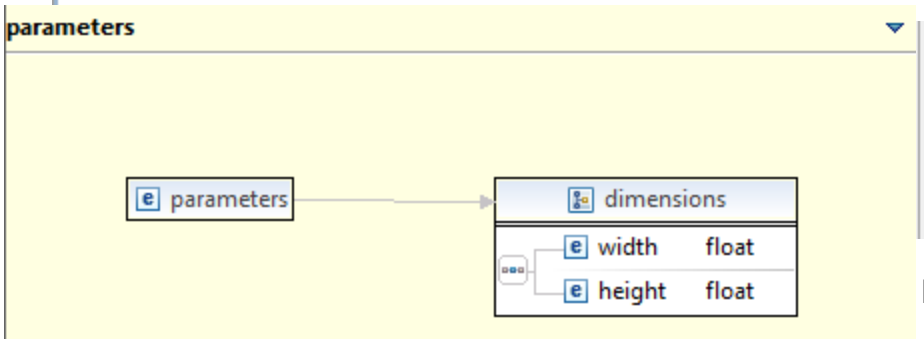


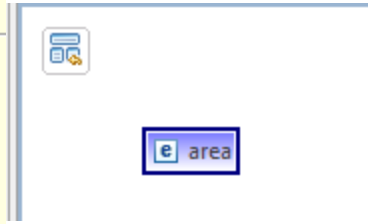
# Eclipse Lab Example > Create WSDL

- Create a folder called WSDL on the Axis2WSTest project.
- Right click on this folder and then New→ Other → Web Services →WSDL and name this file AreaService.wsdl

*http://localhost:8080/Axis2WSTest/services/AreaService*

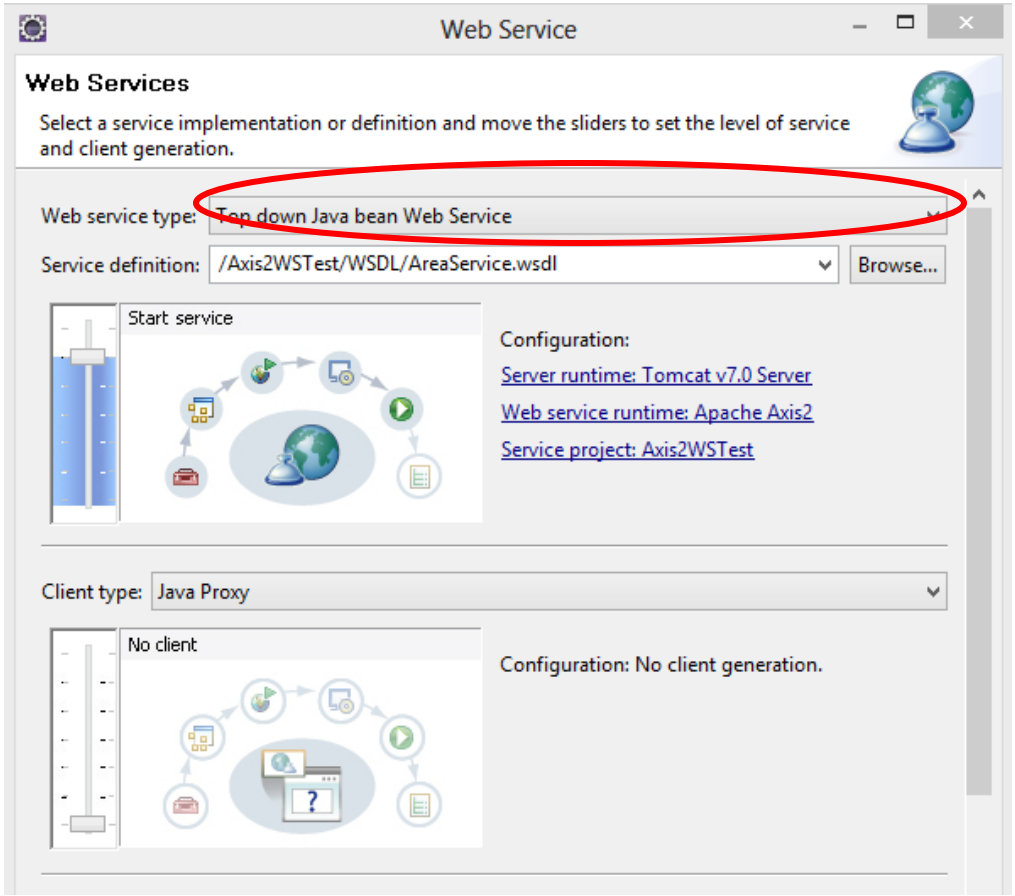


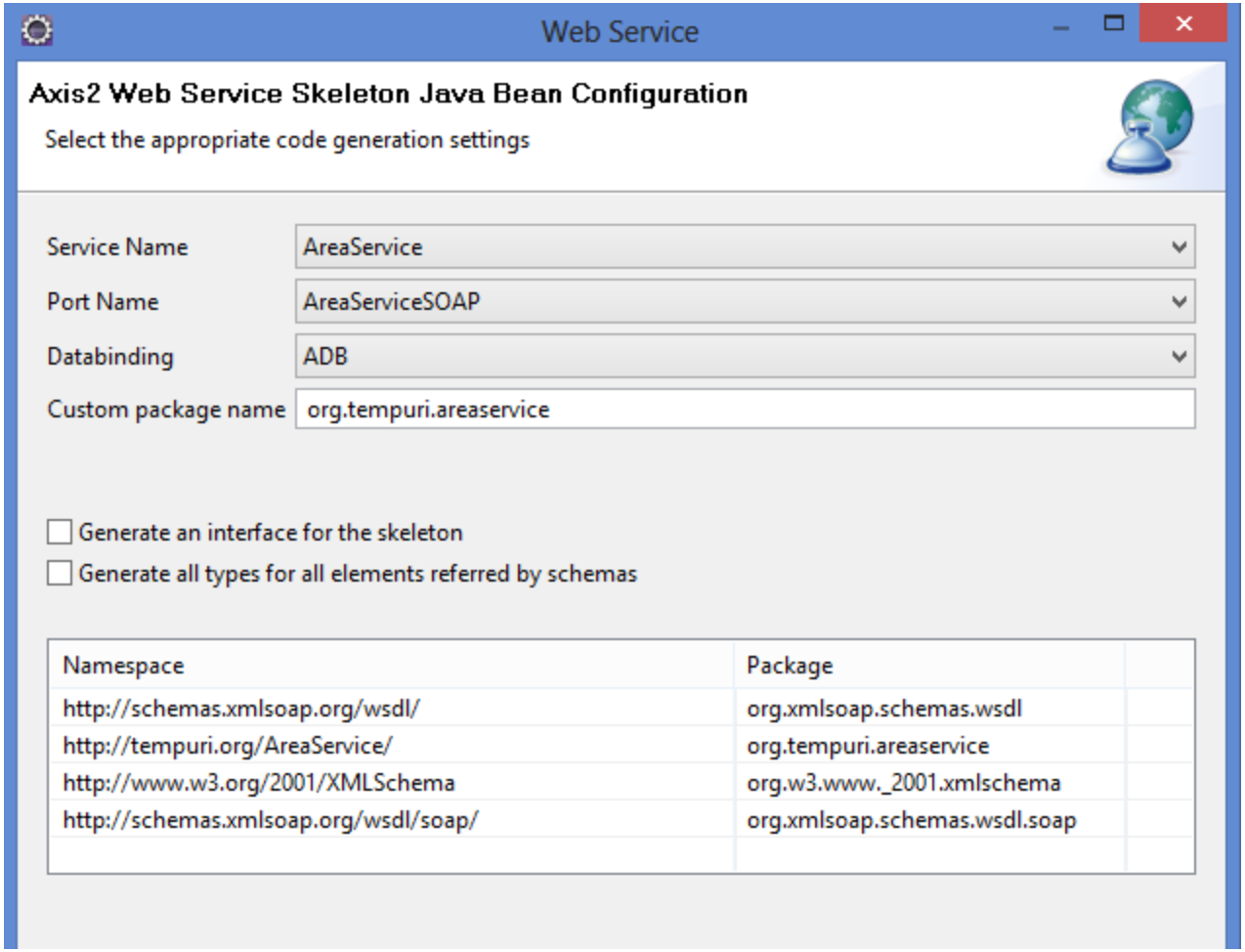



# Eclipse Lab Example

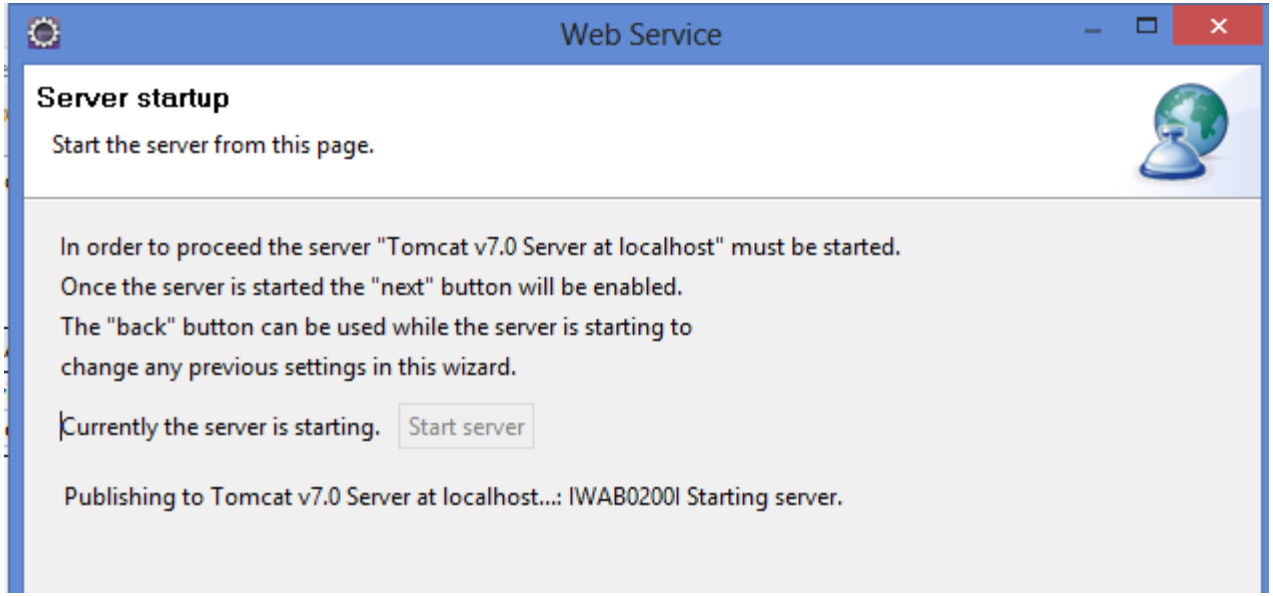
- By selecting AreaService.wsdl source the Open File -> New -> Other... -> Web Services -> Web Service



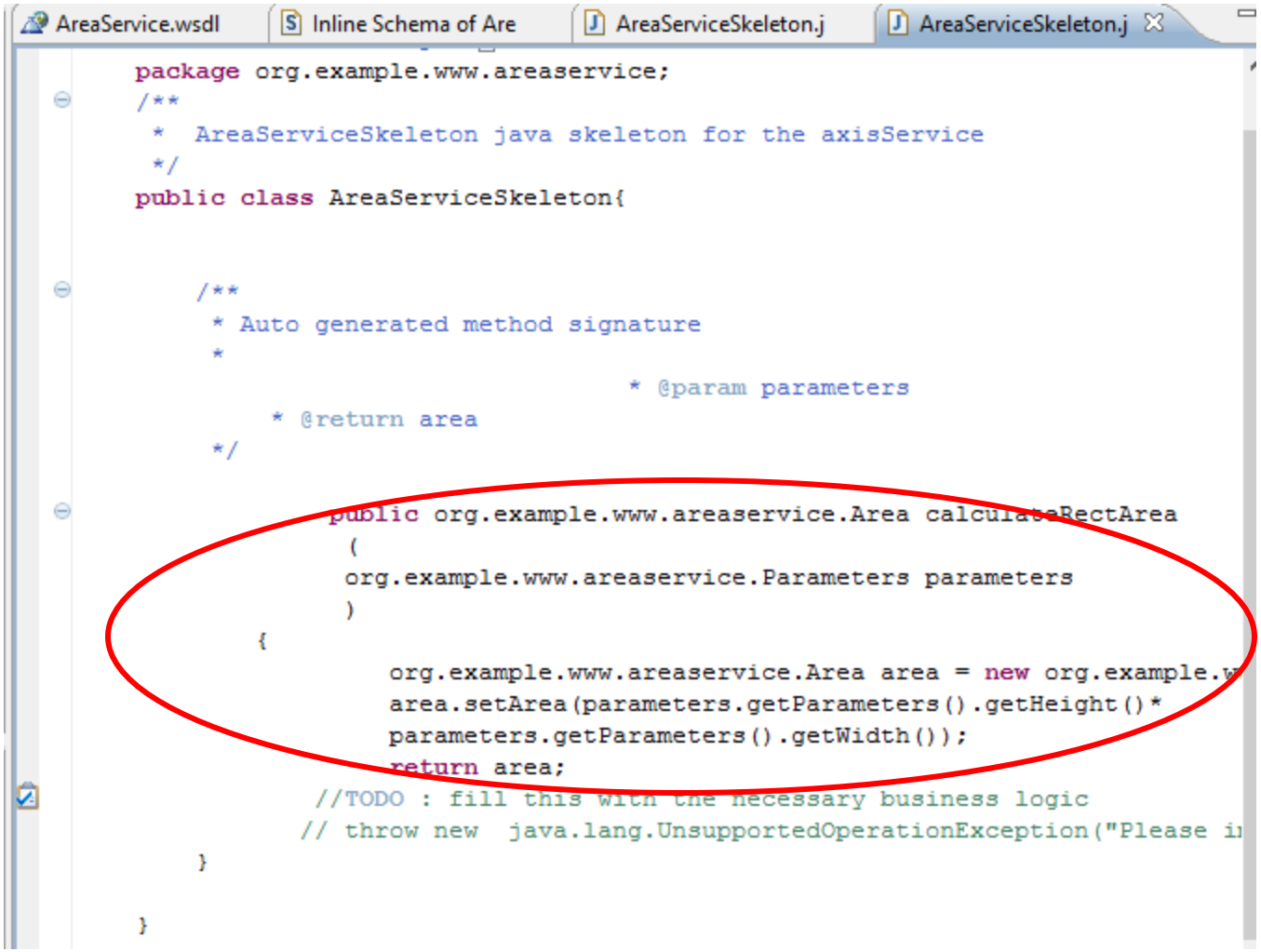
# Skeleton JAVA bean configuration page



# Eclipse Lab Example

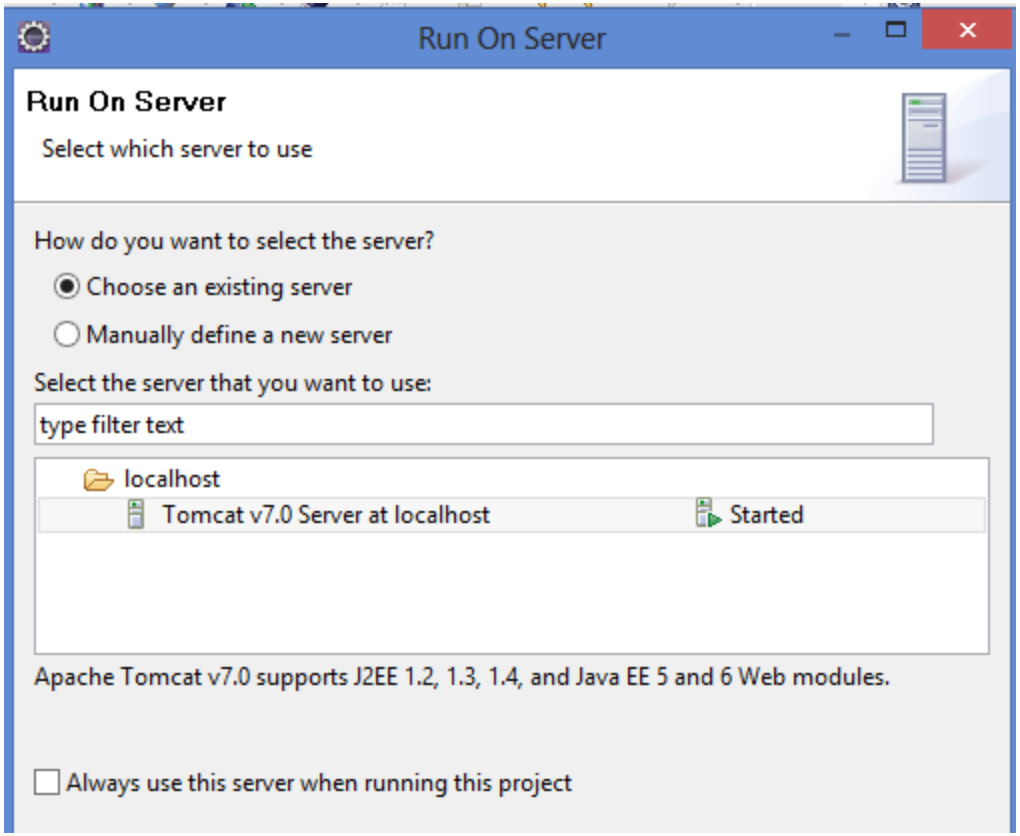



# Area Service Skeletion

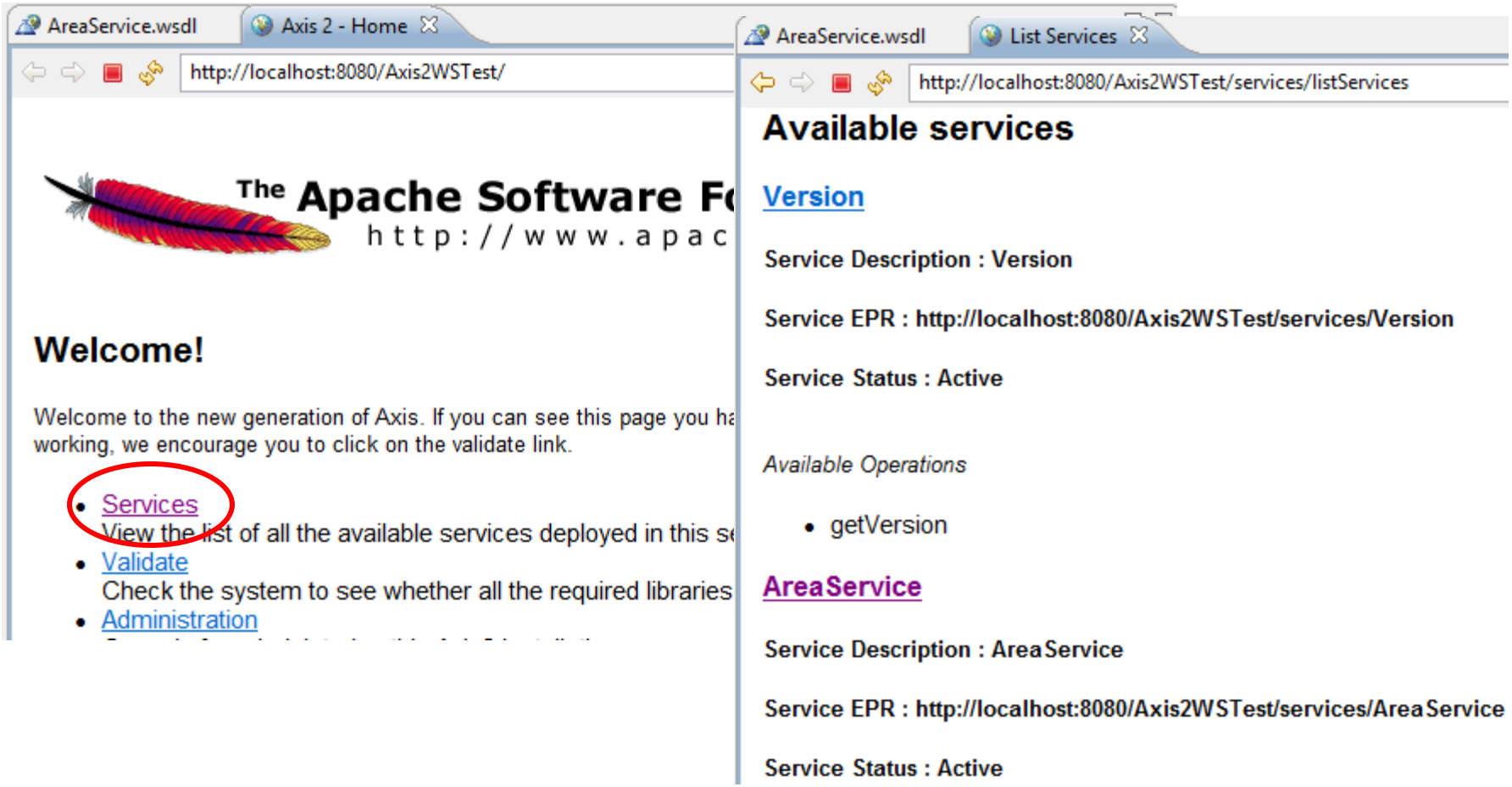


# Running the project

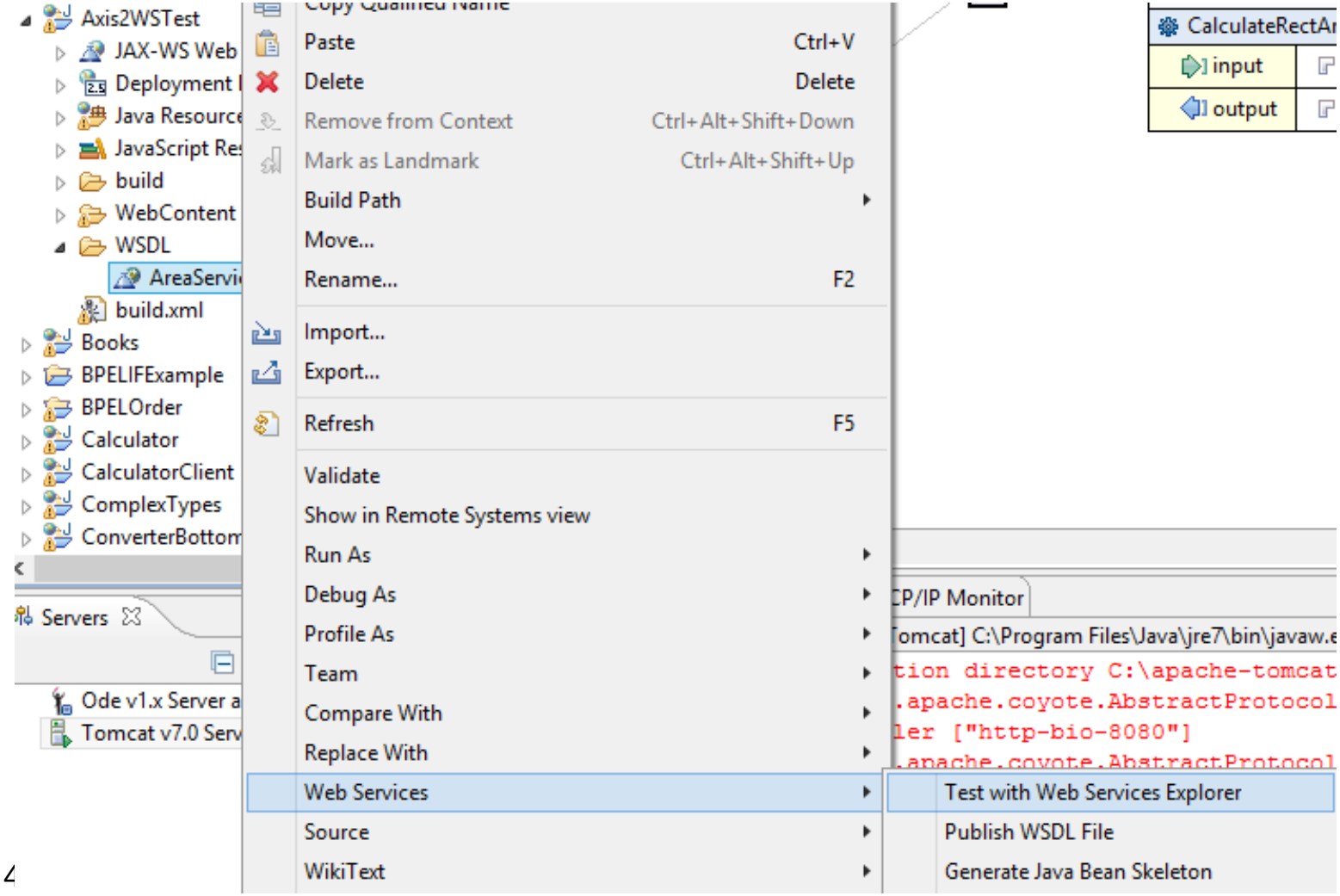
- Right Click the Project and select Run As → Run on Server
- Restart the server if prompted to do so



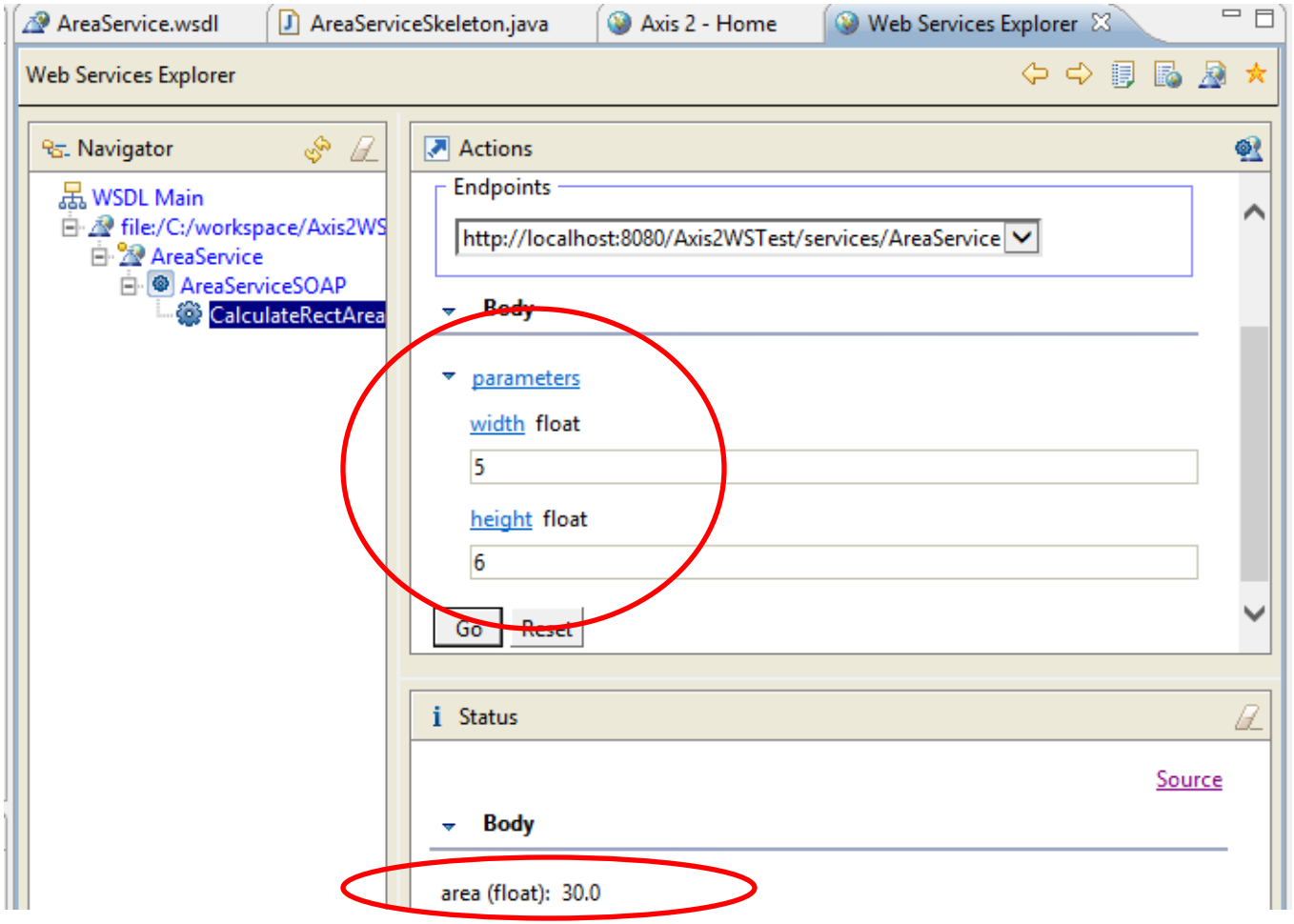
# Axis2 Services



# Web Service Client



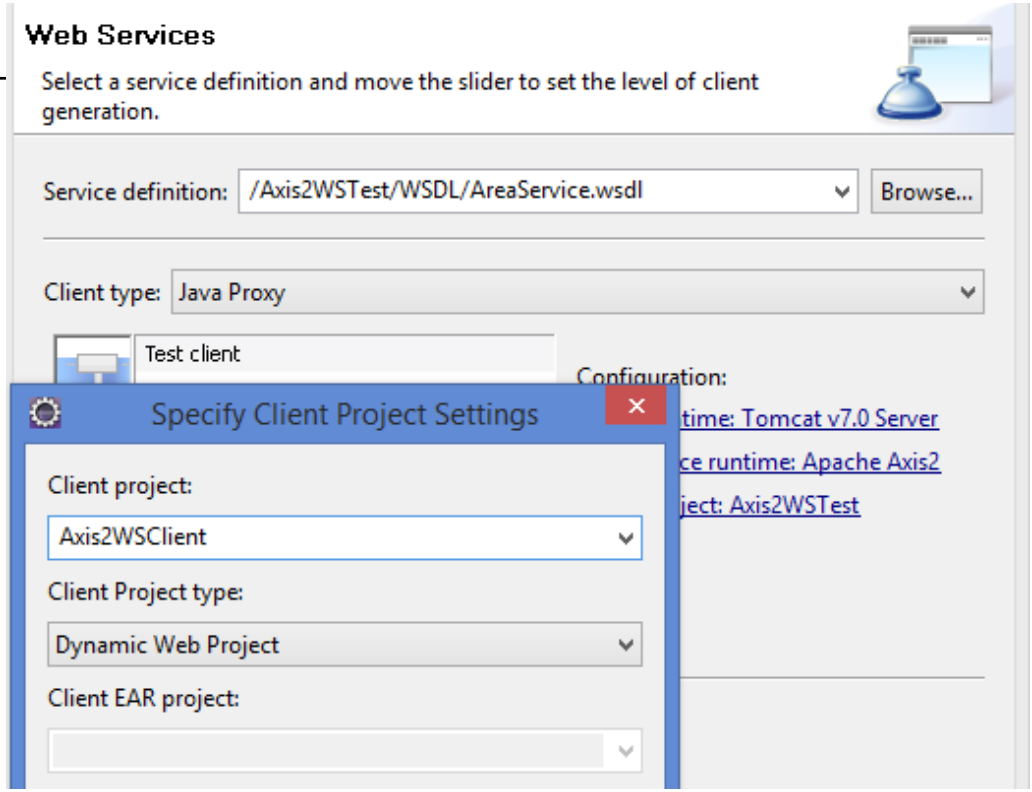
# Web Service Client



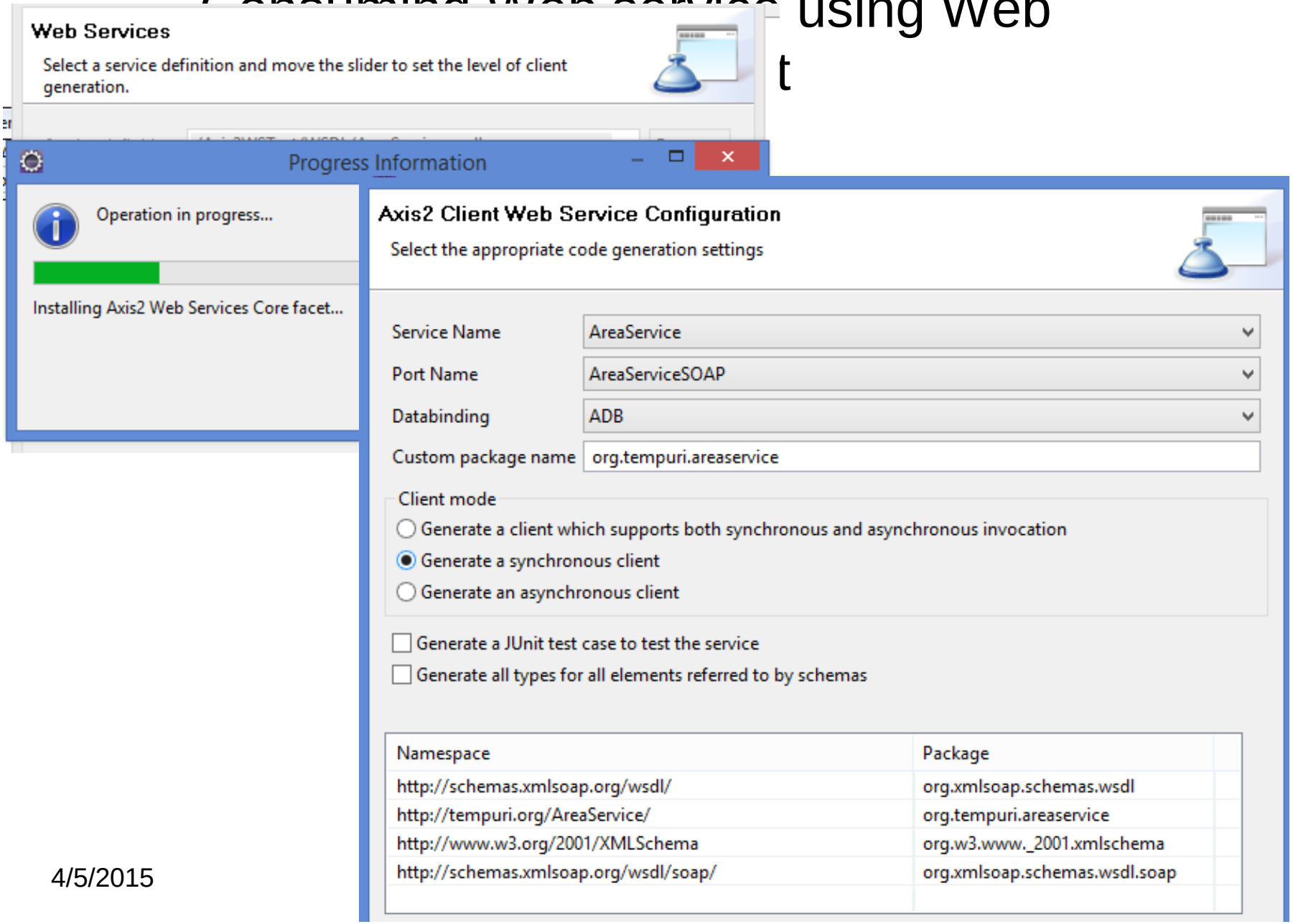
# Consuming Web service by writing a Web Service Client

# Consuming Web service using Web Service Client

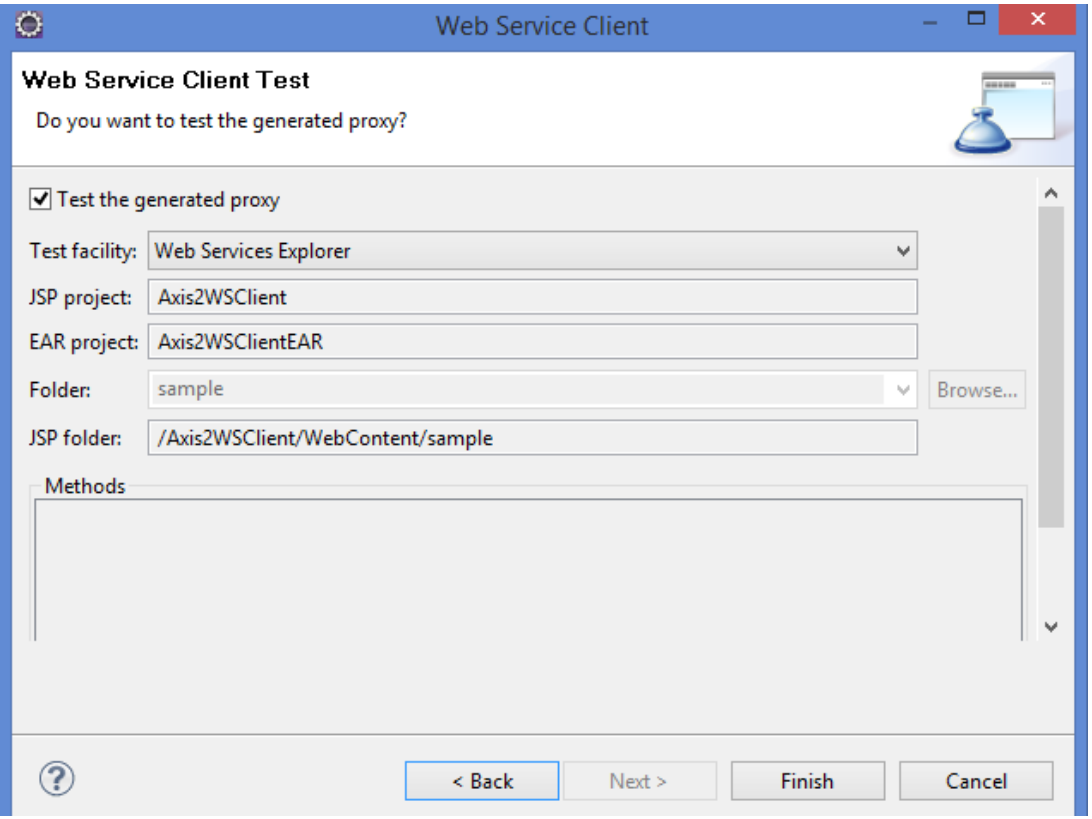
- Select the AreaService.wsdl file in the Axis2Test/WSDL folder Open File -> New -> Other... -> Web Services -> Web Service Client.
- Click Next.
- Move the client slider to the Test Client position.
- Click the Client project link under Configuration and type in Axis2WSClient as the name of the Client project in the dialog that appears:

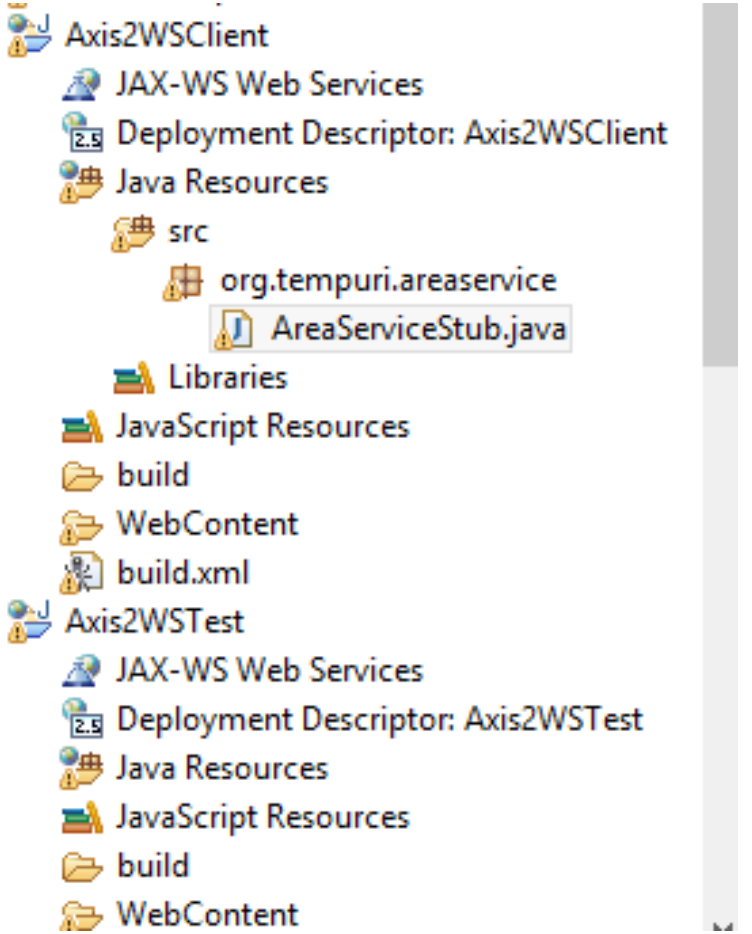


# Consuming Web service using Web t

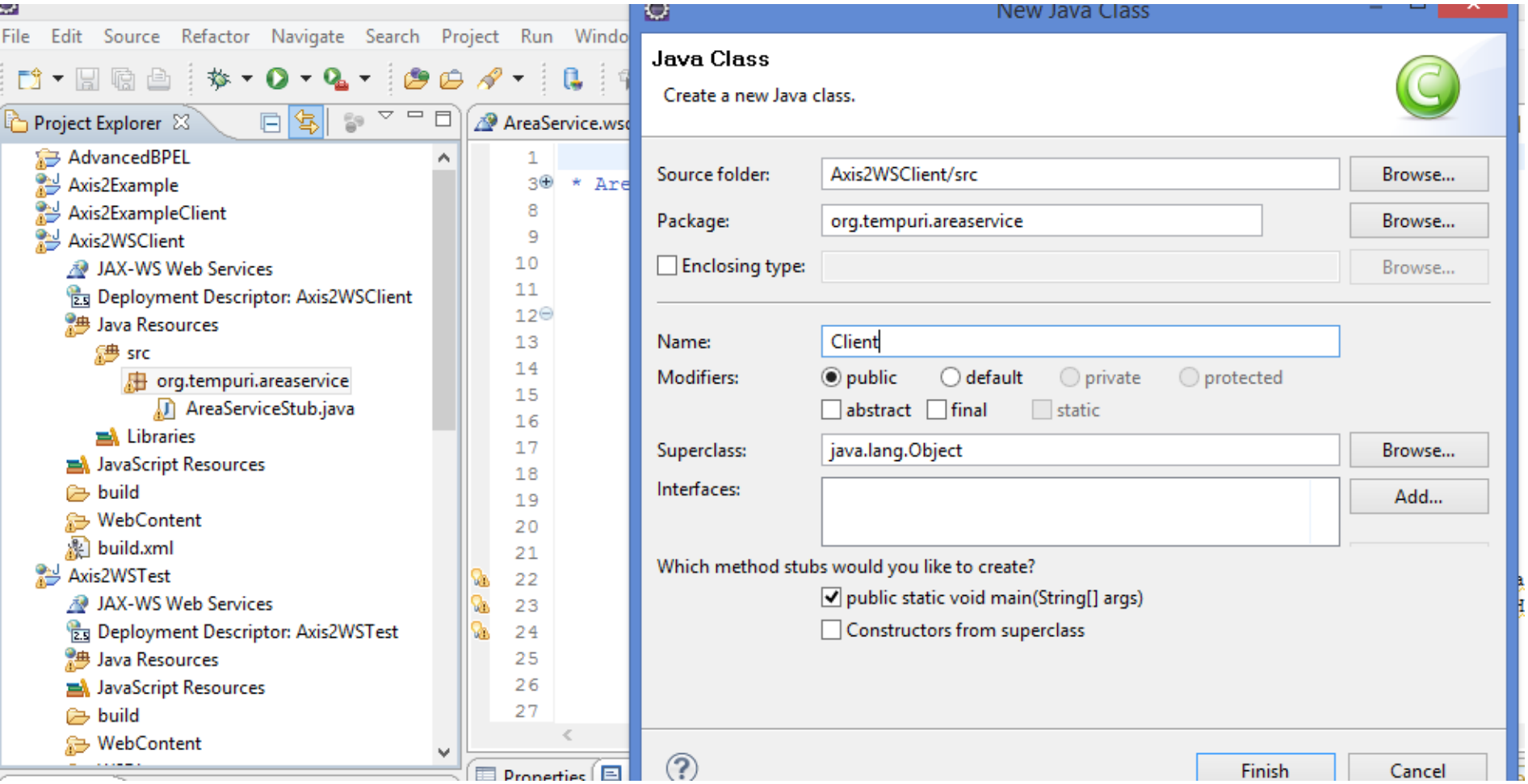


# Consuming Web service using Web Service Client



# Consuming Web service using Web Service Client



# Consuming Web service using Web Service Client

AreaService.wsdl | http://localhost:8080/Axis2WSTest/ser | Web Services Explorer

```
16    public static void main(String[] args) {
17
18        AreaServiceStub stub;
19        try {
20            stub = new AreaServiceStub();
21            Parameters parameters = new Parameters();
22            Dimensions dimensions = new Dimensions();
23            float height = 0, width = 0;
24
25            Scanner scanner = new Scanner(System.in);
26            System.out.println("Enter height:");
27            height = scanner.nextFloat();
28            dimensions.setHeight(height);
29
30            System.out.println("Enter width:");
31            width = scanner.nextFloat();
32            dimensions.setWidth(width);
33
34            parameters.setParameters(dimensions );
35            Area resp = stub.calculateRectArea(parameters);
36            float area = resp.getArea();
37            System.out.printf("The area
```

Properties | Console | TCP/IP Monitor

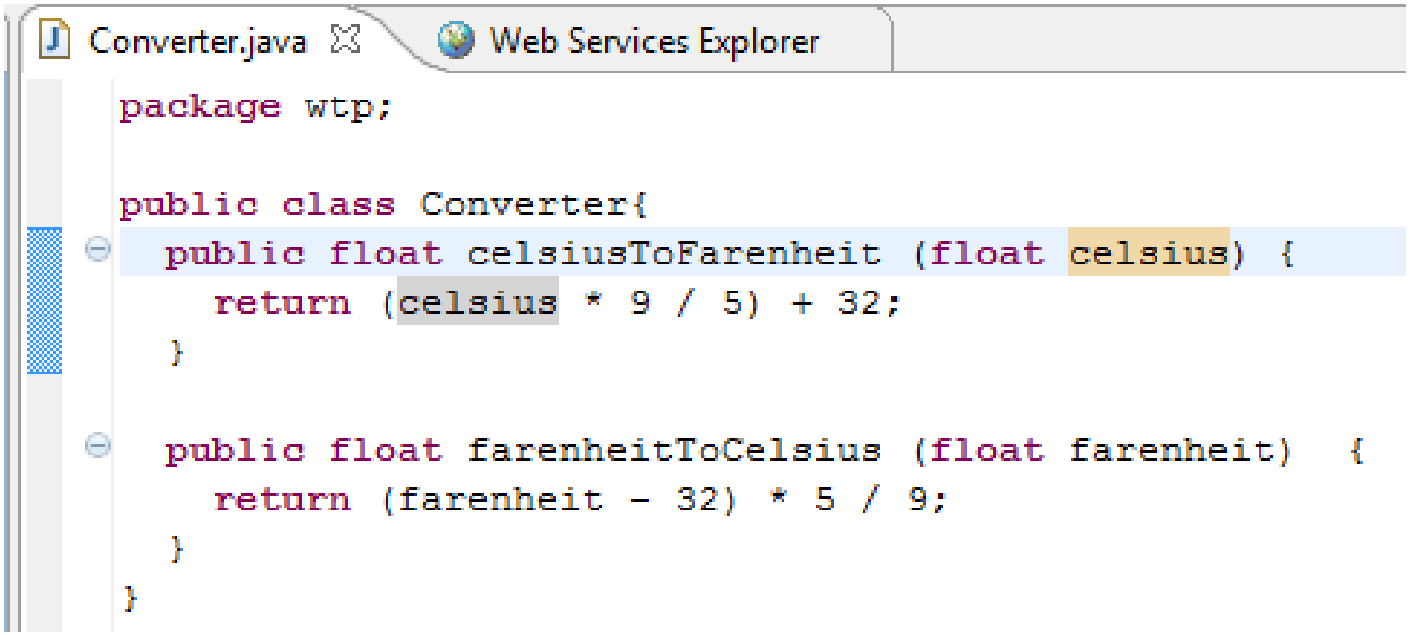
```
<terminated> Client (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (A
Enter height:
5
Enter width:
6
The area is 30.0
```

# Bottom-up Example

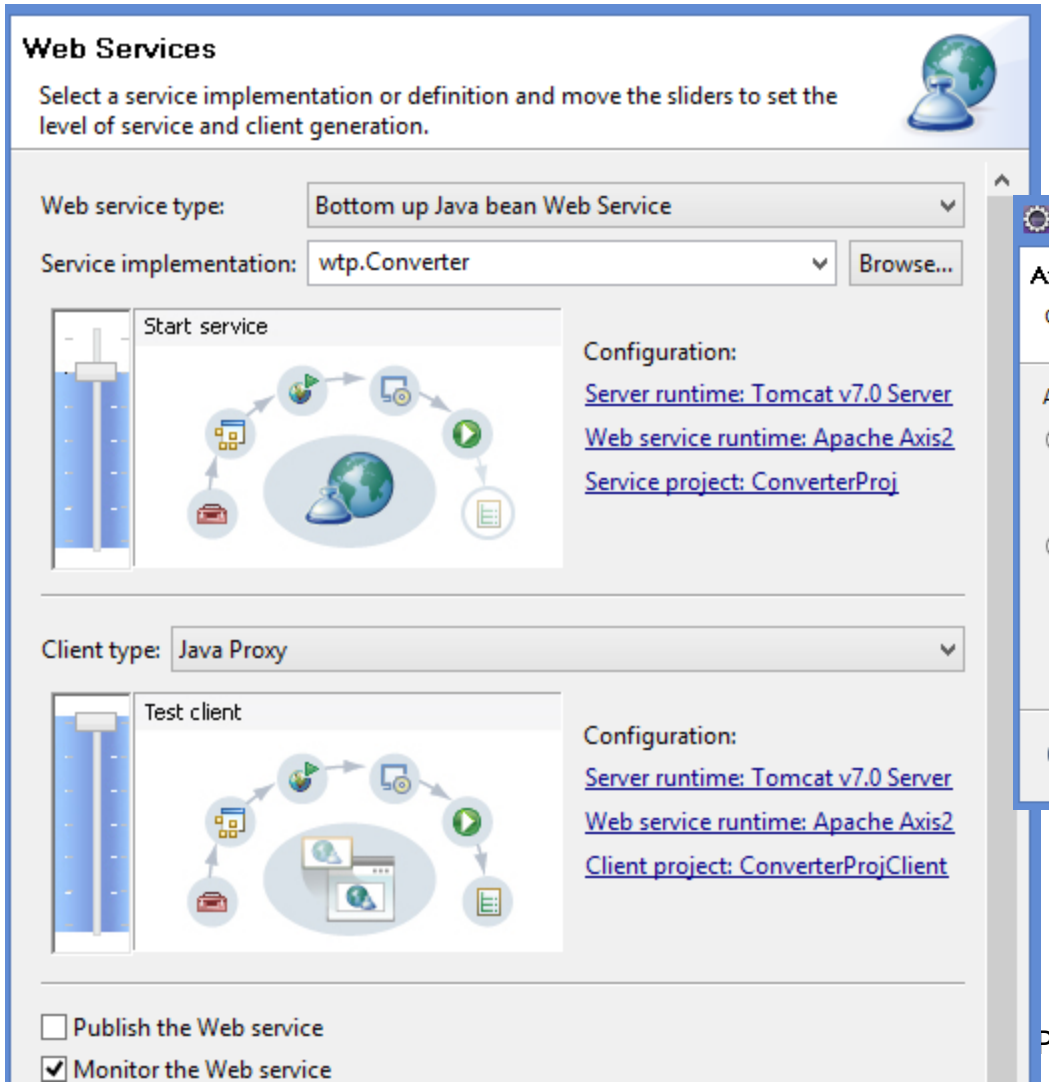
Temperature Example

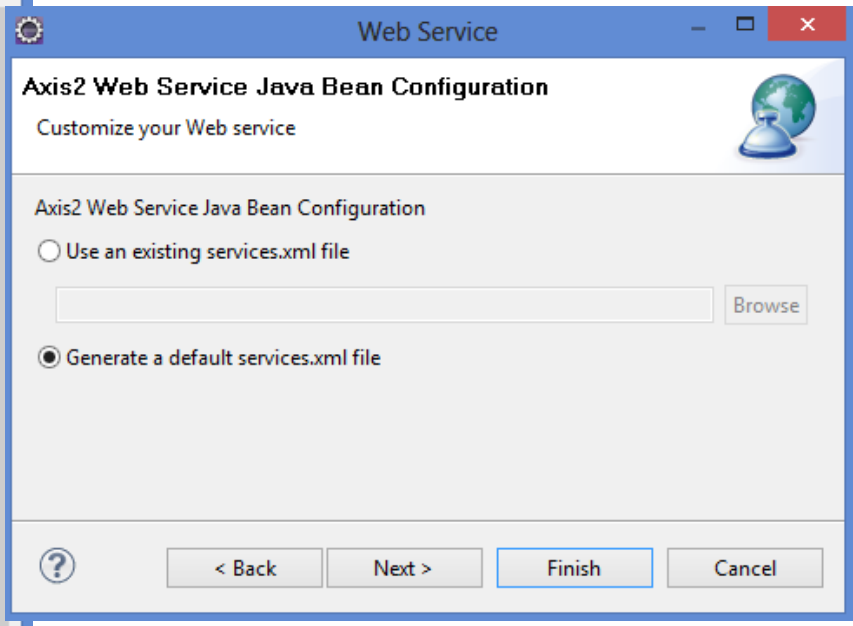
# Eclipse Lab Example : Creating a Bottom Up Web Service

- Create a new Dynamic Project and a new Java class
- Select the **Converter.java** file.
- Open **File -> New -> Other... -> Web Services -> Web Service**

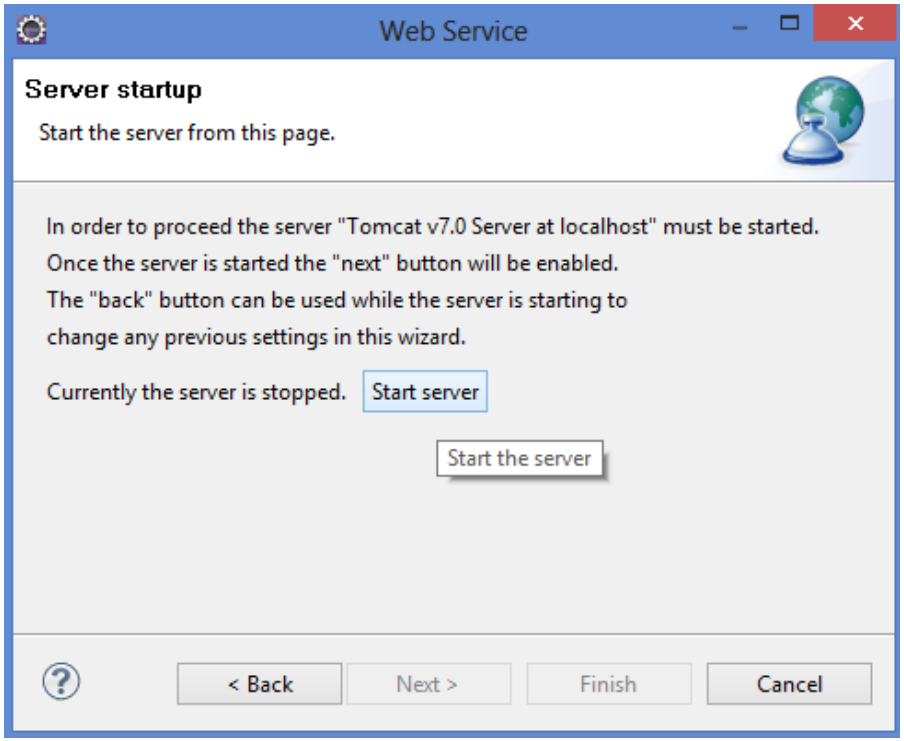


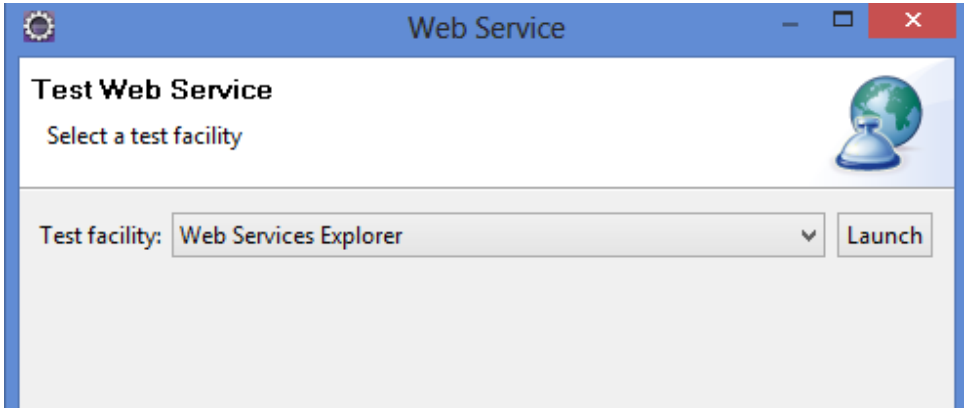
# Eclipse > Creating a Bottom Up Web Service



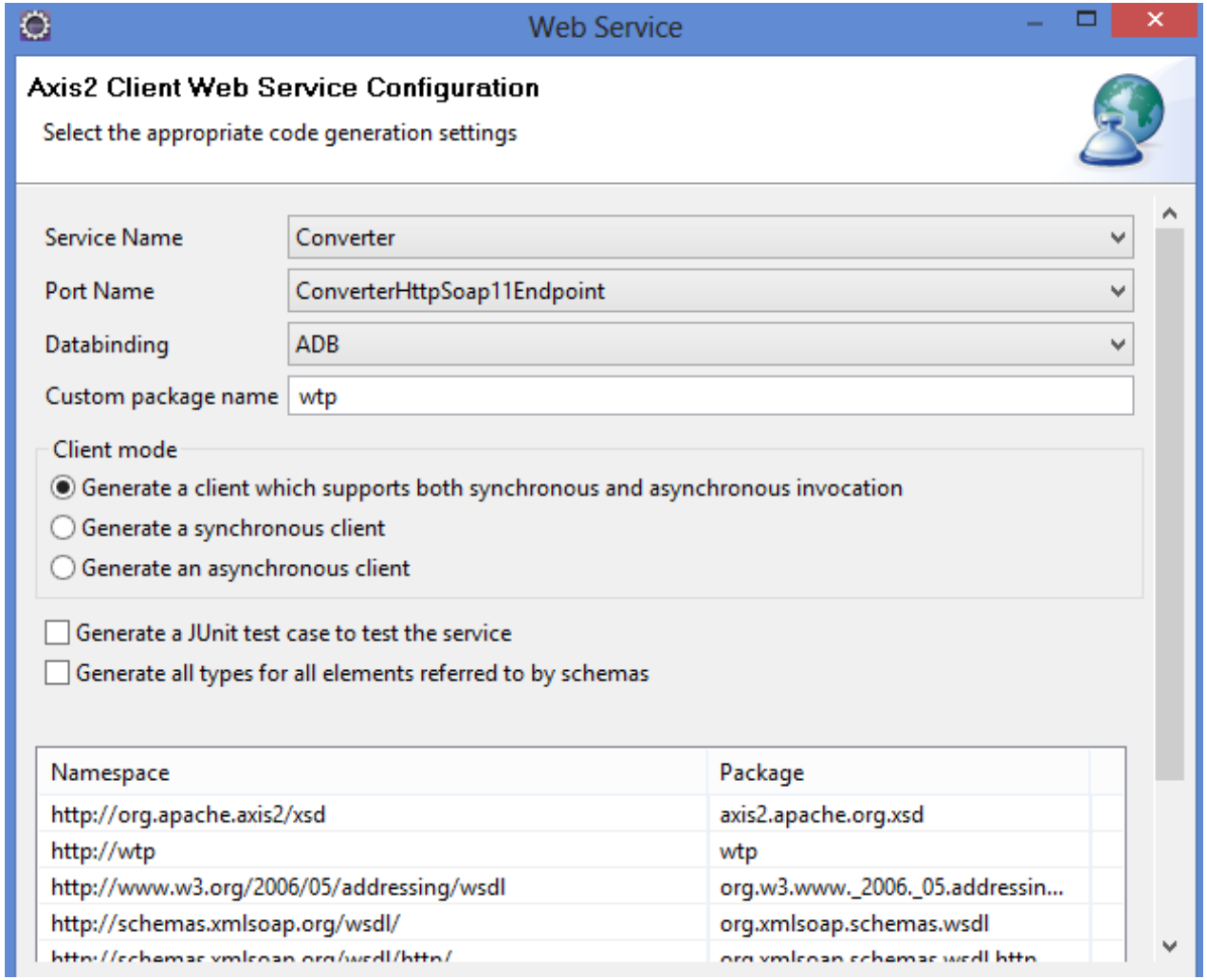



# Eclipse > Creating a Bottom Up Web Service

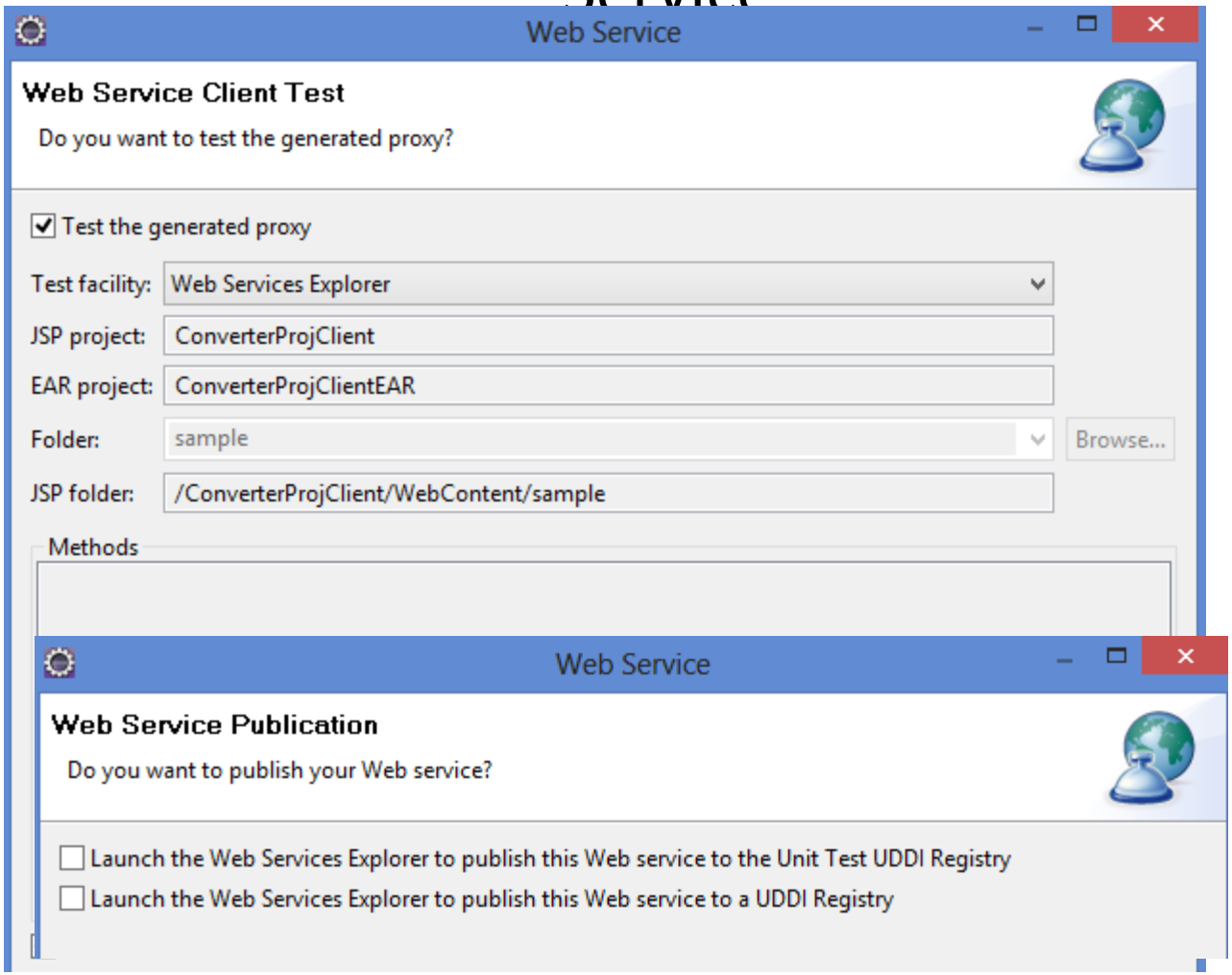


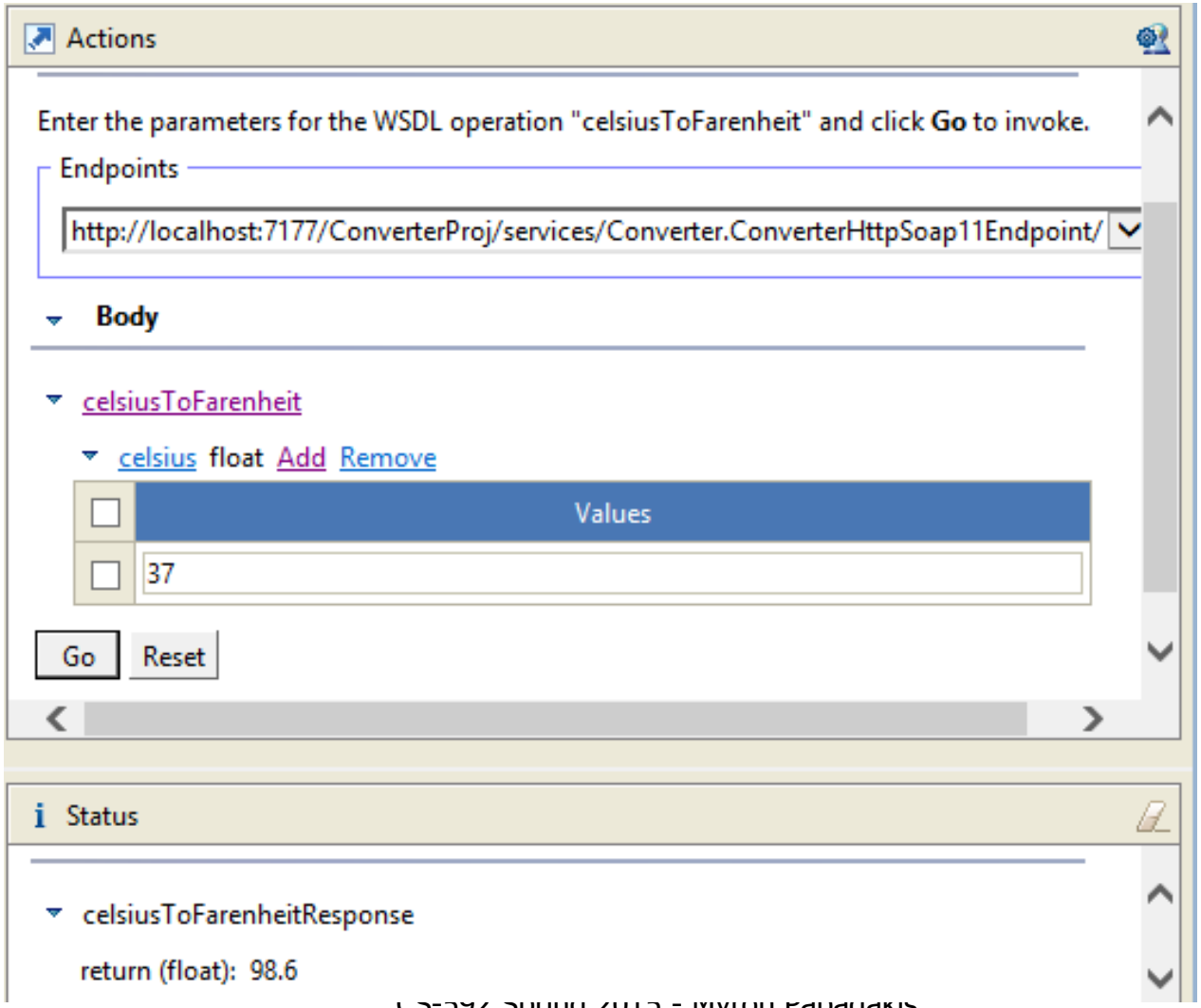
# Eclipse > Creating a Bottom Up Web Service



# Eclipse > Creating a Bottom Up Web Service



# Eclipse > Creating a Bottom Up Web Service



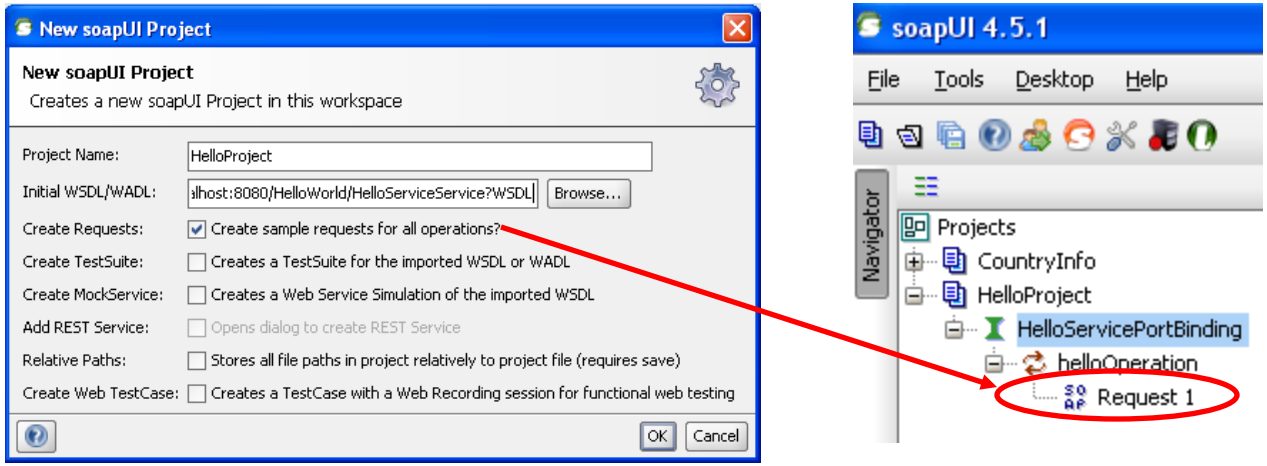
# SoapUI

# Soap UI tool

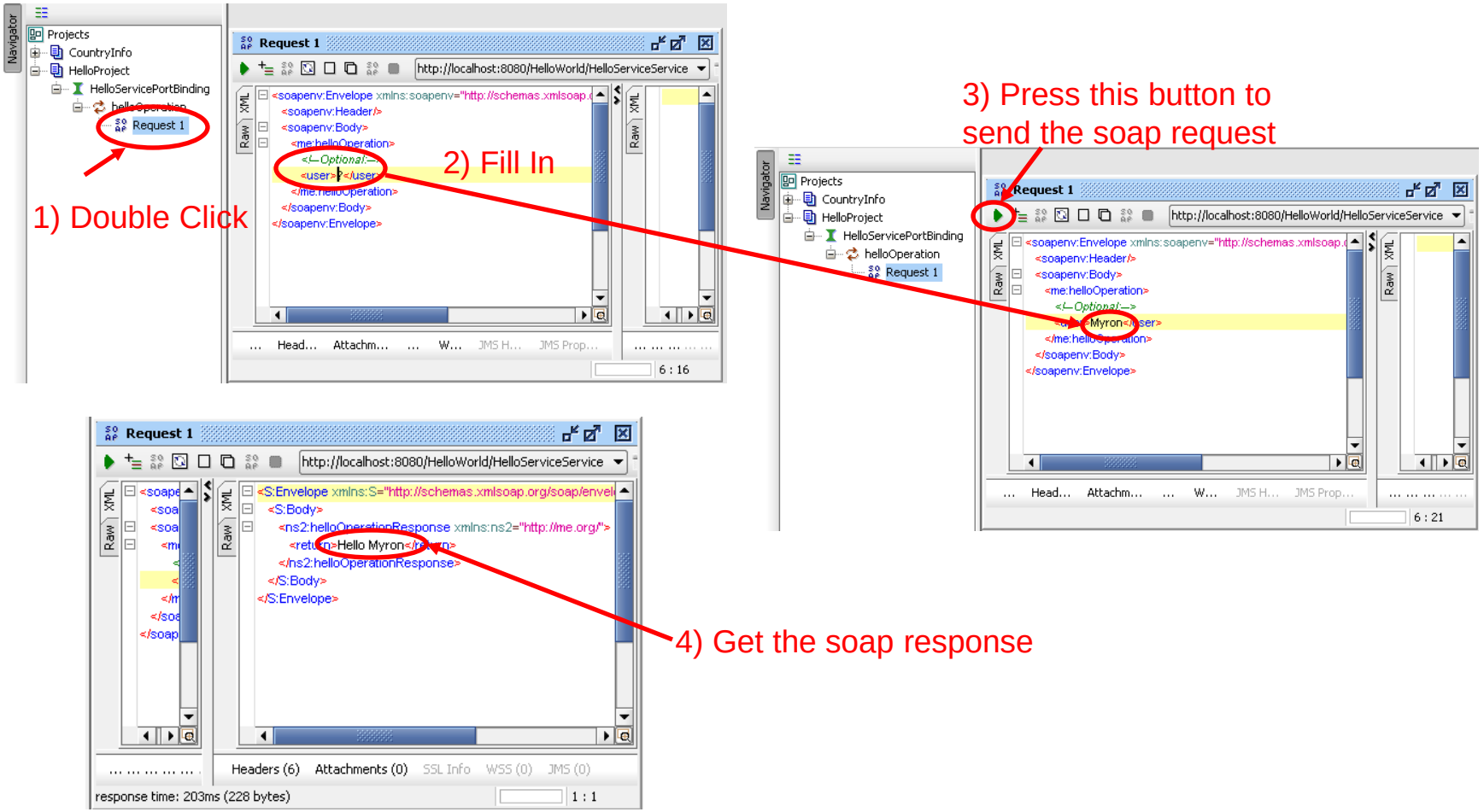
- http://www.soapui.org/Working-with-soapUI/getting-started.html
- Plugins for netbeans and eclipse also exist.
- Some slides from the 1st Assisting Lecture follow next..

# SOAPUI Example 1 (1/2)

- Assume a web service “HelloService” that has a single operation which takes as input a single string from a user (e.g “Myron”) and returns a greeting message to the user (e.g “Hello Myron”)
- Assume that the WSDL of this web service is located in the following url: http://localhost:8080/HelloWorld/HelloServiceService?WSDL
- Invoke this web service using the SOAPUI tool
  - Create a new project (*File > New soapUI Project),* then set the project’s name and the url of the WSDL and press ok

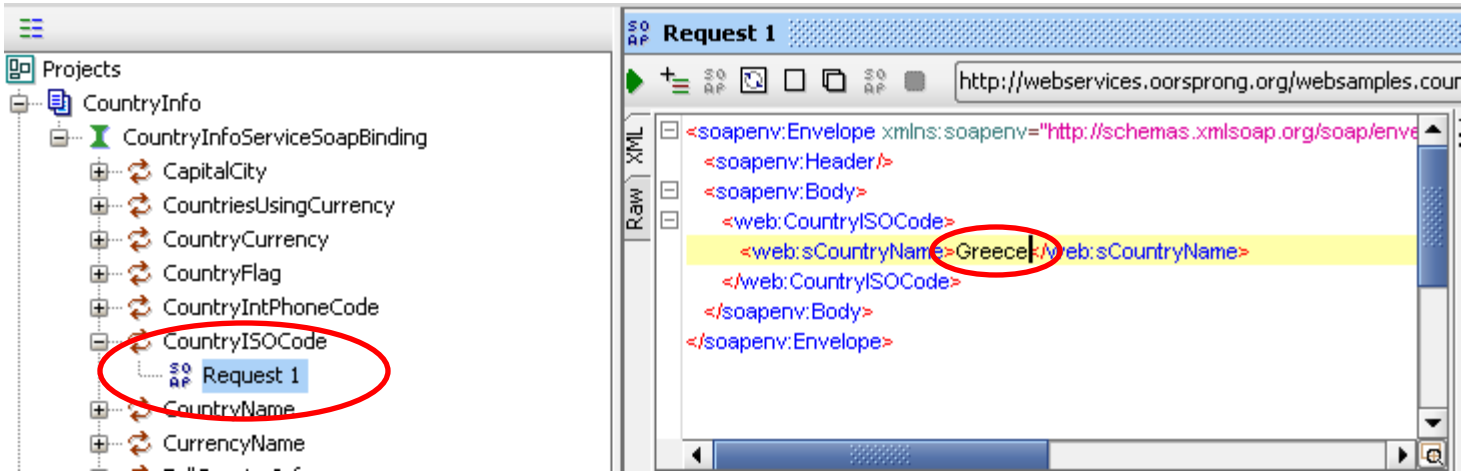


# SOAPUI Example 1 (2/2)

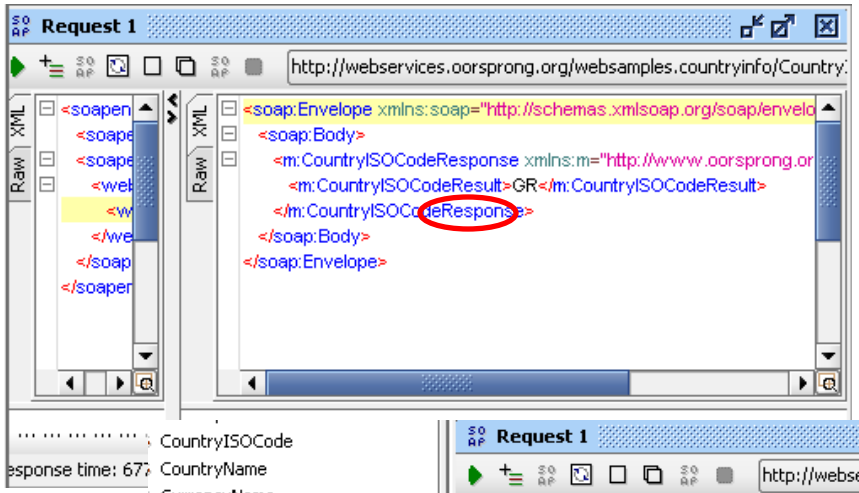


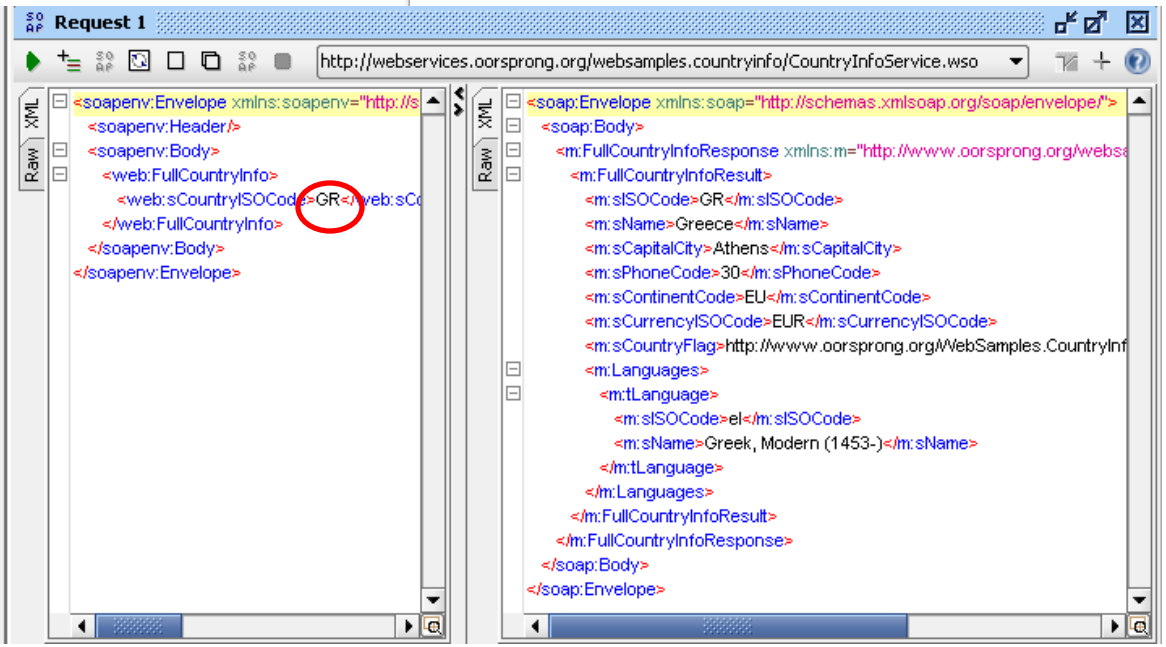
# SOAPUI Example 2 (1/2)

- Create a new project “CountryInfo” and set the WSDL to the following url:
  - http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL
  - This service provides several operations for retrieving information about a particular country
  - Find the iso code of “Greece” by sending a request (soap message)



# SOAPUI Example 2 (2/2)




- The response shows that the iso code of Greece is “GR”
- Use this code to make a request to the operation “FullCountryInfo”

# References

- **Bottom-up Approach:** http://www.eclipse.org/webtools/community/tutorials/BottomUpAxis2WebService/bu_tutorial.html

- **Top-down Approach:** http://www.eclipse.org/webtools/community/tutorials/TopDownAxis2WebService/td_tutorial.html

- http://www.soapui.org/Working-with-soapUI/getting-started.html

# Τέλος Ενότητας

# Χρηματοδότηση

•Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

•Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Κρήτης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

•Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

# Σημειώματα

# Σημείωμα αδειοδότησης

- Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά Δημιουργού - Μη Εμπορική Χρήση - Παρόμοια Διανομή 4.0 [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».


[1] http://creativecommons.org/licenses/by-nc-nd/4.0/

•Ως **Μη Εμπορική** ορίζεται η χρήση:

–που δεν περιλαμβάνει άμεσο ή έμμεσο οικονομικό όφελος από την χρήση του έργου, για το διανομέα του έργου και αδειοδόχο

–που δεν περιλαμβάνει οικονομική συναλλαγή ως προϋπόθεση για τη χρήση ή πρόσβαση στο έργο

–που δεν προσπορίζει στο διανομέα του έργου και αδειοδόχο έμμεσο οικονομικό όφελος (π.χ. διαφημίσεις) από την προβολή του έργου σε διαδικτυακό τόπο

•Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

.

# Σημείωμα Αναφοράς