AN INTRODUCTION: JAVA ENVIRONMENT ANT, AXIS, TOMCAT

Laboratory of Service Design and Engineering

2011/2012

Outline

- Java, ANT, Web Services programming...
- ANT
  - Basic Concepts
  - Example: Basic build file
  - Exercise
- Apache Axis and Tomcat Installation
  - Web Service Middle ware: An overview
  - Axis Installation
  - Deployment of a Java Web Service (JWS)
  - Running a Client with Dll Method

ANT

ANT

- Java-based build tool
- Open source
- Full portable
- Home page: http://ant.apache.org

ANT

- The name is the acronym for "Another Neat Tool“
- James Duncan Davidson, the original author of ANT, describing what Ant is intended to be, says
  - "ants do an extremely good job at building things"
  - "ants are very small and can carry a weight dozens of times their own"

INSTALLING ANT

- Make sure you have already installed JDK 5 or JDK 6 or latest versions
- Download ANT from http://ant.apache.org/bindownload.cgi
- Follow the installation instructions written in http://ant.apache.org/manual/install.html#installing

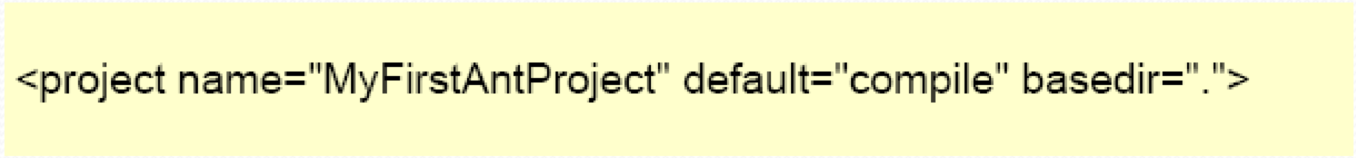
BUILD FILE

```
<?xml version="1.0"?>
  <project name="MyFirstAntProject" default="compile" basedir=".">
     <target name="init">
              <mkdir dir="build" />
     </target>
     <target name="compile" depends="init">
              < javac srcdir="src" destdir="build" />
     </target>
  </project>
```

- **Ant Project:** a collection of named targets. Each build file contains one project.
- **Ant Target:** a fixed series of ant tasks in a specified order that can depend on other named targets.
- **Ant Task:** something that ant can execute jobs, such as compile, create jars, copy.

ANT PROJECT

- The first line of a build file is the Project tag

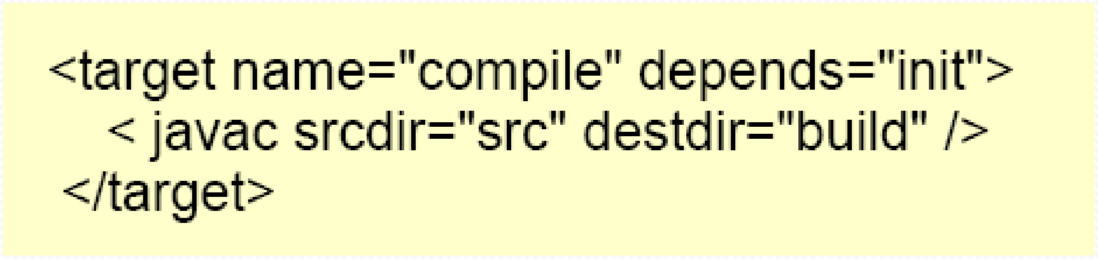


```
<project name="MyFirstAntProject" default="compile" basedir=".">
```

- The name is the name of your project
- The default attribute points to the target which, by default, will start the building process.
- The basedir sets the directory from which all path calculations are done.
- Both the name and basedir attributes are optional fields but the default attribute is required.

ANT TARGET

- A target is a set of tasks waiting to be executed.



```
<target name="compile" depends="init">
    < javac srcdir="src" destdir="build" />
 </target>
```

- A target can be dependent upon another target

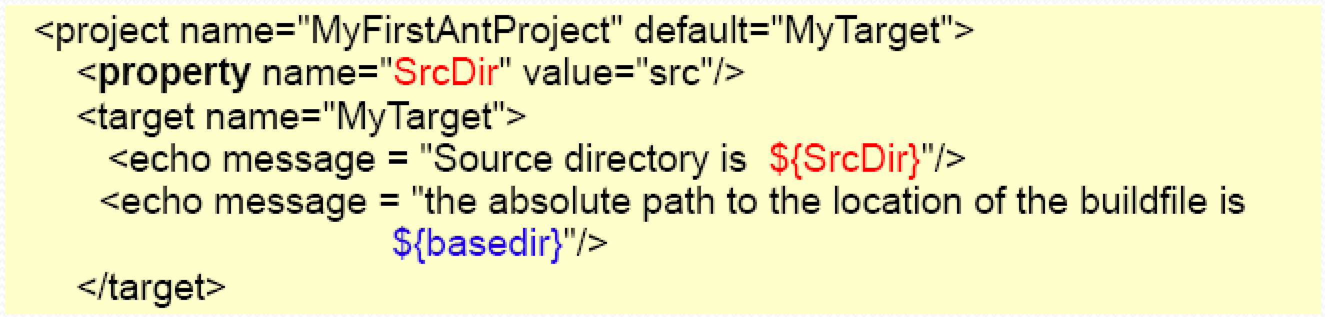
ANT TASK

- Tasks are the snippets of code that get the job done.

```
< javac srcdir="src" destdir="build" />
```

- In the above example, the task of the "compile" target runs javac in the source directory, and places the .class files in the directory "build" which was created in the "init" target (if “build” directory already exists, then it will NOT be create it).
- Notice that "src" and "build" may be defined in Property tags

ANT PROPERTIES



```
<project name="MyFirstAntProject" default="MyTarget">
   <property name="SrcDir" value="src"/>
   <target name="MyTarget">
     <echo message = "Source directory is  ${SrcDir}"/>
    <echo message = "the absolute path to the location of the buildfile is
                      ${basedir}"/>
   </target>
```

- Ant properties are immutable: once they are set they can not be changed within a build process
- Ant properties can be
  - set anywhere in a build file
  - set in an external property file
  - referenced anywhere in a build file after they are set
- Ant has a set of "built in" properties (e.g. basedir)

Let’s get hands dirty with ANT

Excercie-1 (15 min)

- Make a ANT project as following
  - It should take a couple of Integer values as input and shows result after adding them.
  - Make proper directory structure (src, build, dist, classes etc).
  - Make build.xml file.
  - Run build file to get output into proper classes dir.

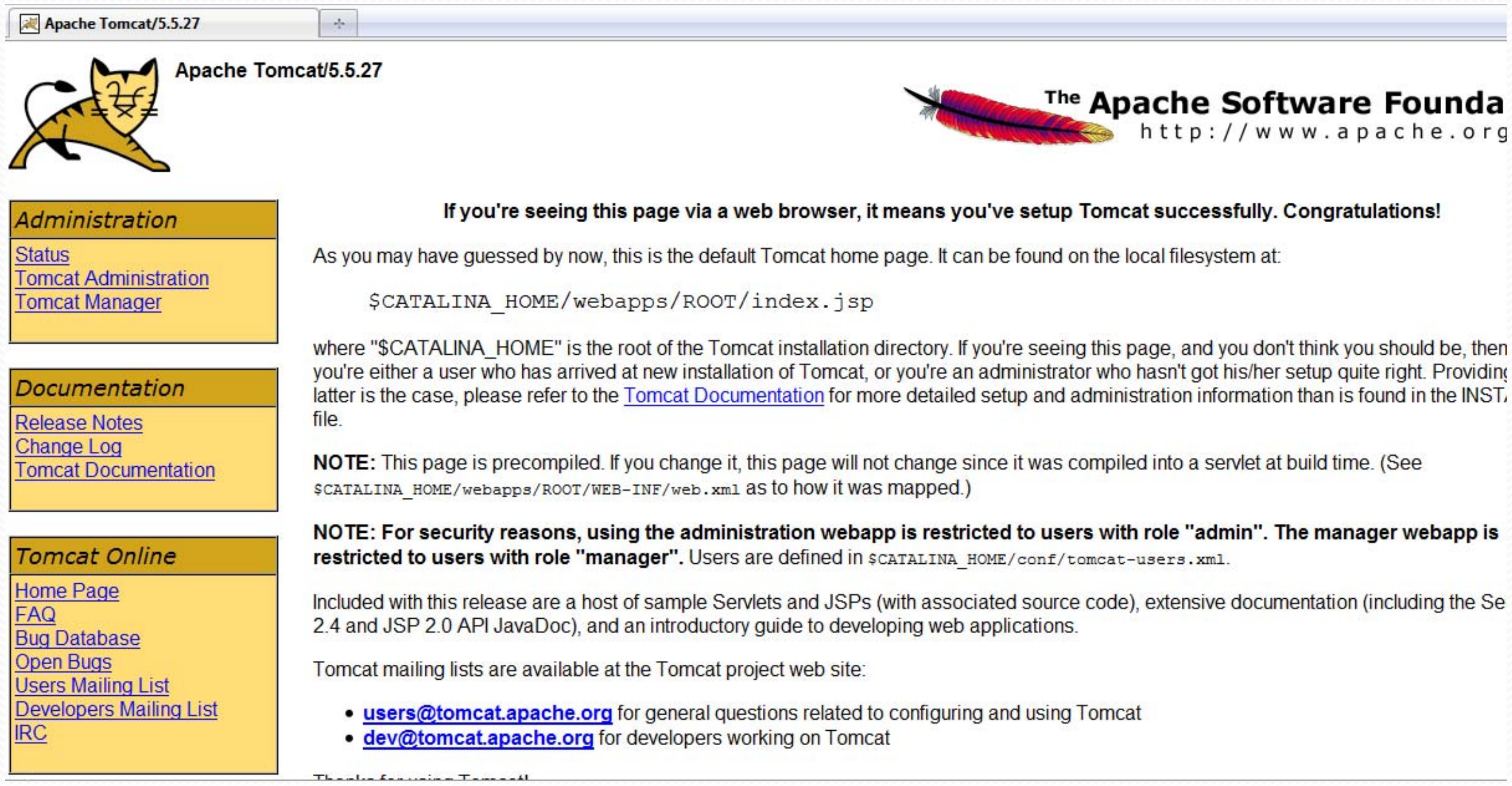
APACHE AXIS & TOMCAT

Tomcat Installation

- Go to: http://tomcat.apache.org/
- Download the zip version of Tomcat 5.5 or latest
- Unzip it somewhere
- Start the server
  - For example, if tomcat is installed in C:\ then, C:\tomcat_home\bin\startup
- Go to http://localhost:8080/

Tomcat Installation 2

You should see page like this:



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.jsp
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providin latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INST file.

NOTE: This page is precompiled. If you change it, this page will not change since it was compiled into a servlet at build time. (See `$CATALINA_HOME/webapps/ROOT/WEB-INF/web.xml` as to how it was mapped.)

NOTE: For security reasons, using the administration webapp is restricted to users with role "admin". The manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

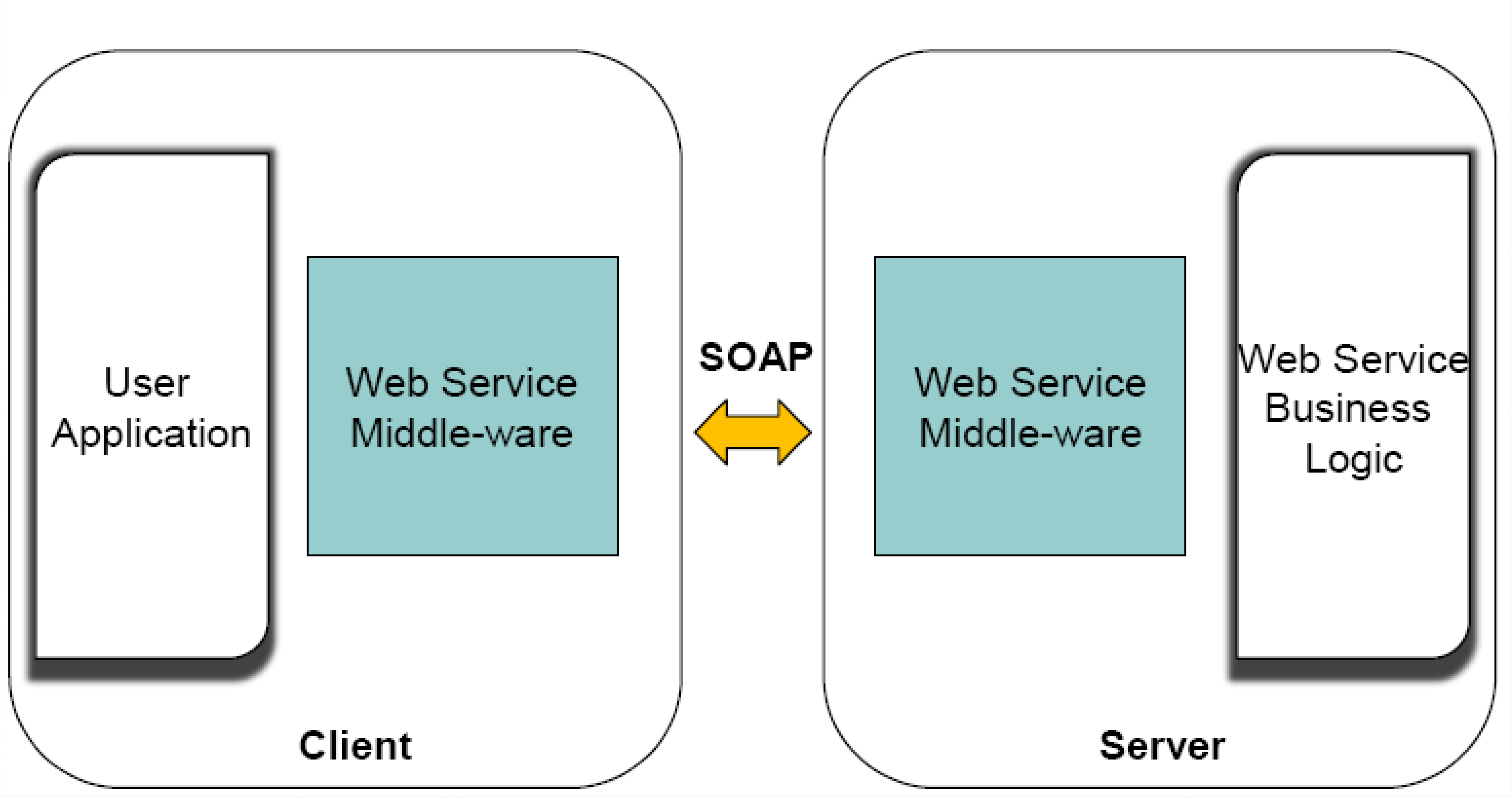
Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation (including the Se 2.4 and JSP 2.0 API JavaDoc), and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

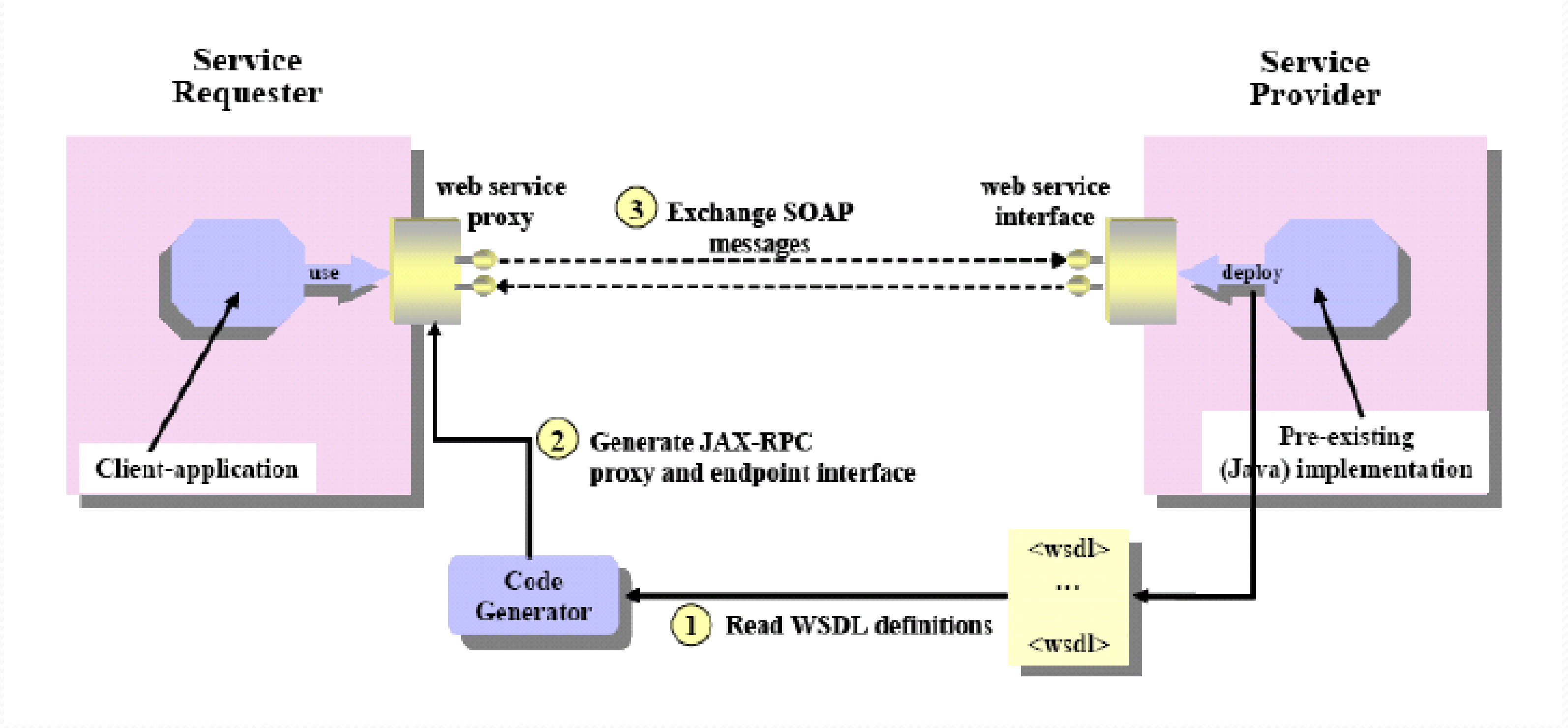
- **users@tomcat.apache.org** for general questions related to configuring and using Tomcat
- **dev@tomcat.apache.org** for developers working on Tomcat

AXIS
Web Service: Middle Ware

Web Service Middle-Ware

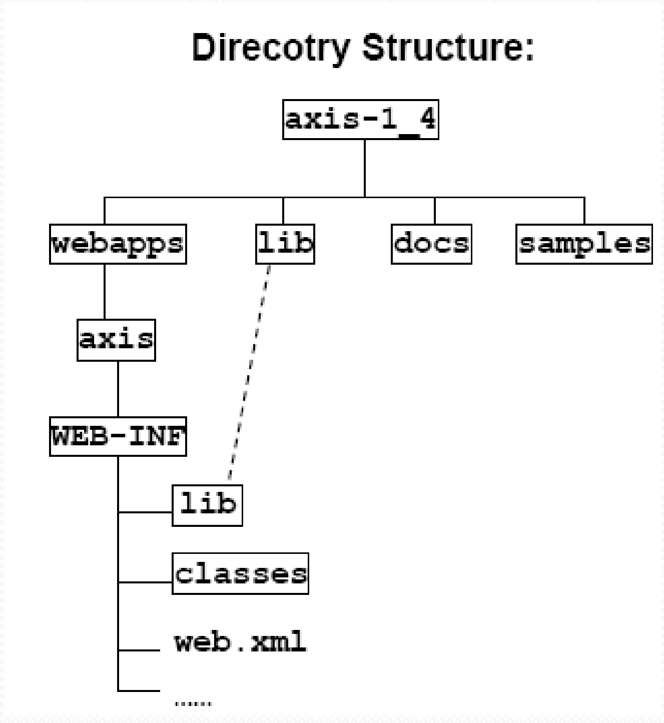


Web Service Middle-Ware 2



Axis Installation & Deployment

- Download Axis from http://ws.apache.org/axis/
- Unzip it somewhere
- Copy **webapps\axis** to **webapps** directory of Tomcat.
- Restart Tomcat

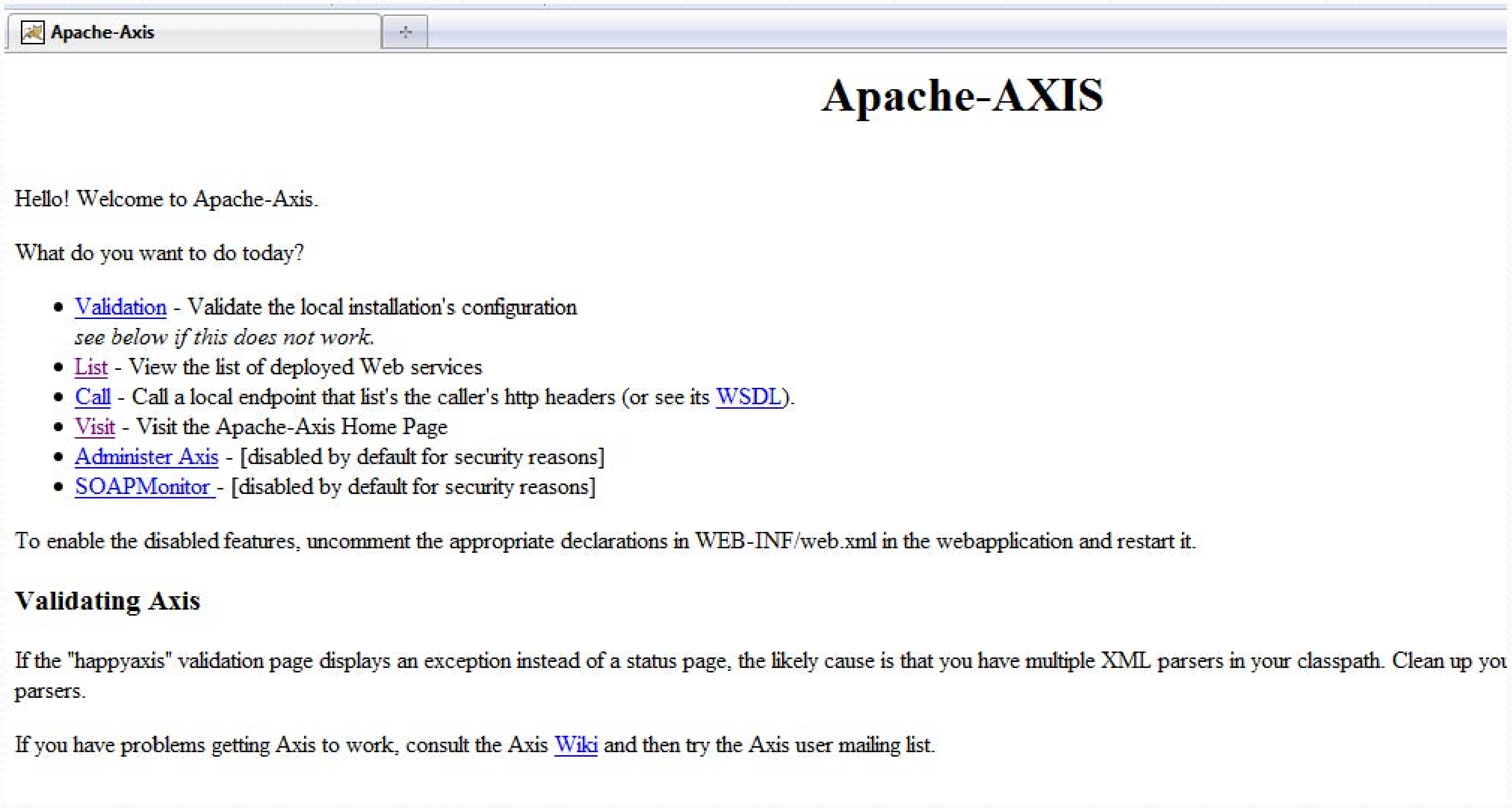


Axis Installation & Deployment

- set **AXIS_HOME**=c:\axis
- set **AXIS_LIB**=%AXIS_HOME%\lib
- set **AXISCLASSPATH**=%AXIS_LIB%\axis.jar;%AXIS_LIB%\commons-discovery.jar; %AXIS_LIB%\commons-logging.jar;%AXIS_LIB%\jaxrpc.jar;%AXIS_LIB%\saaj.jar; %AXIS_LIB%\log4j-1.2.8.jar;%AXIS_LIB%\xml-apis.jar;%AXIS_LIB%\xercesImpl.jar

Test the Deployment

- Point browser to http://localhost:8080/axis
- You should see page like this



Apache-Axis

Apache-AXIS

Hello! Welcome to Apache-Axis.

What do you want to do today?

- Validation - Validate the local installation's configuration
  see below if this does not work.
- List - View the list of deployed Web services
- Call - Call a local endpoint that list's the caller's http headers (or see its WSDL).
- Visit - Visit the Apache-Axis Home Page
- Administer Axis - [disabled by default for security reasons]
- SOAPMonitor - [disabled by default for security reasons]

To enable the disabled features, uncomment the appropriate declarations in WEB-INF/web.xml in the webapplication and restart it.

Validating Axis

If the "happyaxis" validation page displays an exception instead of a status page, the likely cause is that you have multiple XML parsers in your classpath. Clean up you parsers.

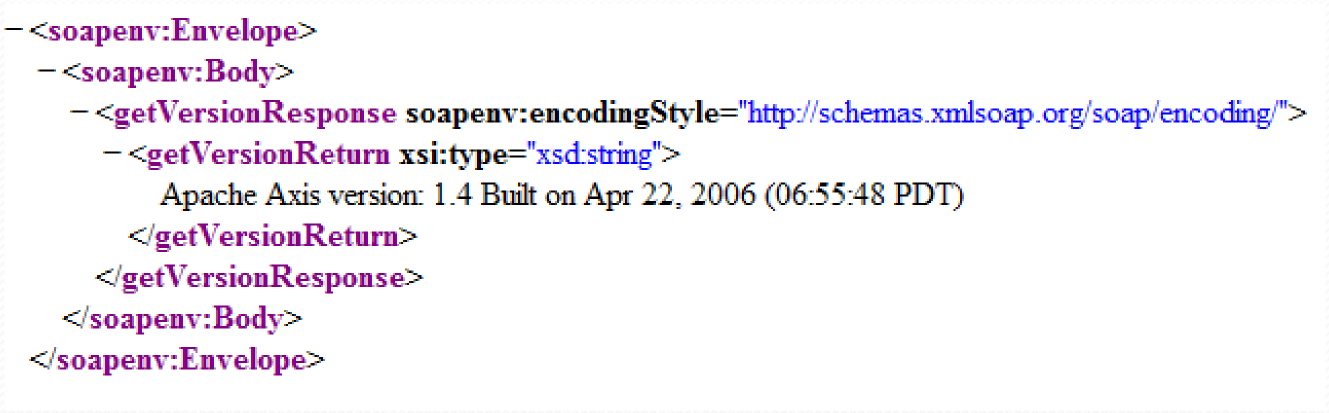
If you have problems getting Axis to work, consult the Axis Wiki and then try the Axis user mailing list.

Validate AXIS

- Follow the link http://localhost:8080/axis/happyaxis.jsp
- This must show you
  - Happyaxis.jsp page
  - It verifies the needed and optional libraries

Test a SOAP Endpoint

- Now it's time to test a service, follow this link
- http://localhost:8080/axis/services/Version?method=getVersion
- It shows you something like this



```
- <soapenv:Envelope>
  - <soapenv:Body>
    - <getVersionResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      - <getVersionReturn xsi:type="xsd:string">
          Apache Axis version: 1.4 Built on Apr 22, 2006 (06:55:48 PDT)
        </getVersionReturn>
      </getVersionResponse>
    </soapenv:Body>
  </soapenv:Envelope>
```

How to create a web service

- There are two basic methods

1. Implement the service (write a java class) and produce the WSDL

   JAVA → WSDL

2. Create the WSDL and produce the java class

   WSDL → JAVA

How to produce WSDL

- There are three basic steps

1. <u>**Instant deployment**</u> (JWS)
   - Generates the WSDL from a URL
   - For simple services
2. <u>**WSDD Deployment (**Web Service Deployment Descriptor)</u>
   - Generates a WSDL from a URL
   - For wsdd deployed services
3. Use <u>**Java2WSDL**</u> tool
   - Generates WSDL from java service implementation classes

Deploying web service: JWS

- Steps
  - Copy “axis” directory from “webapps” directory of the Axis distribution to the deployment directory of server.
    - For Tomcat it is “webapps” folder
  - Make a Java program (Any simple) and copy it to “axis” directory
    - No special code needed.
    - All public, non-static methods exposed.

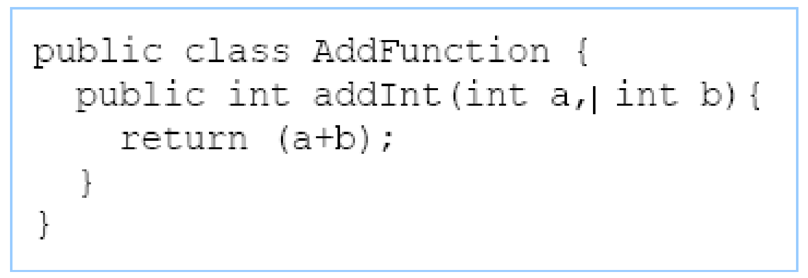
Deploying web service: JWS

- Change the file extension from ".java" to ".jws"
  - To get WSDL go to :

  http://host:port/axis/file-name.jws?wsdl
- A file ".class" would be created under the directory "…\webapps\axis\WEB-INF\jwsClasses"
  - If the class is in a package, the appropriate subdirectory is created under "…\jwsClasses"

JWS Web Service

- Create a Java Class using following code

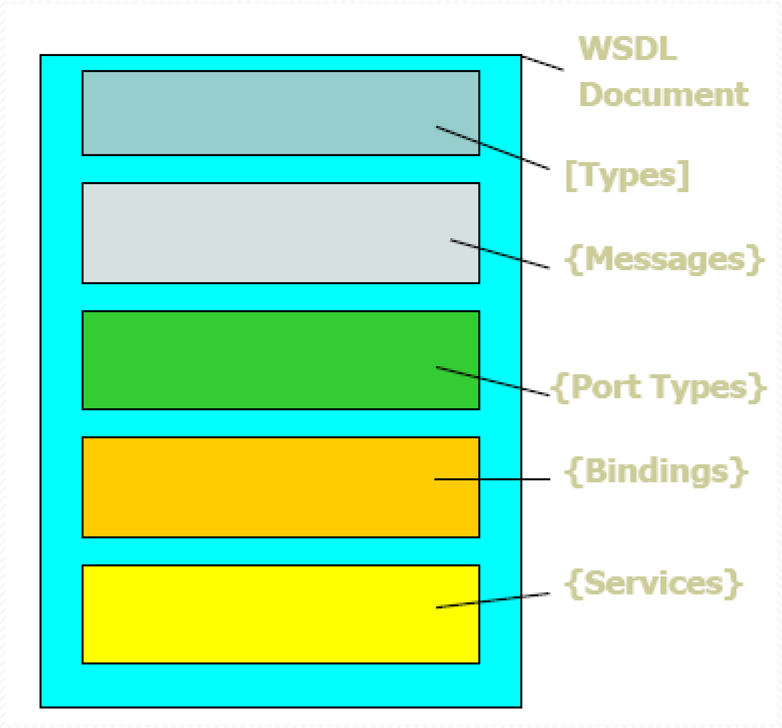


```
public class AddFunction {
  public int addInt(int a, int b){
    return (a+b);
  }
}
```

- Change the extension to ".JWS"
- Make sure the class name is as exactly file name
- Copy it to "webapps/axis" directory.
- That's It.

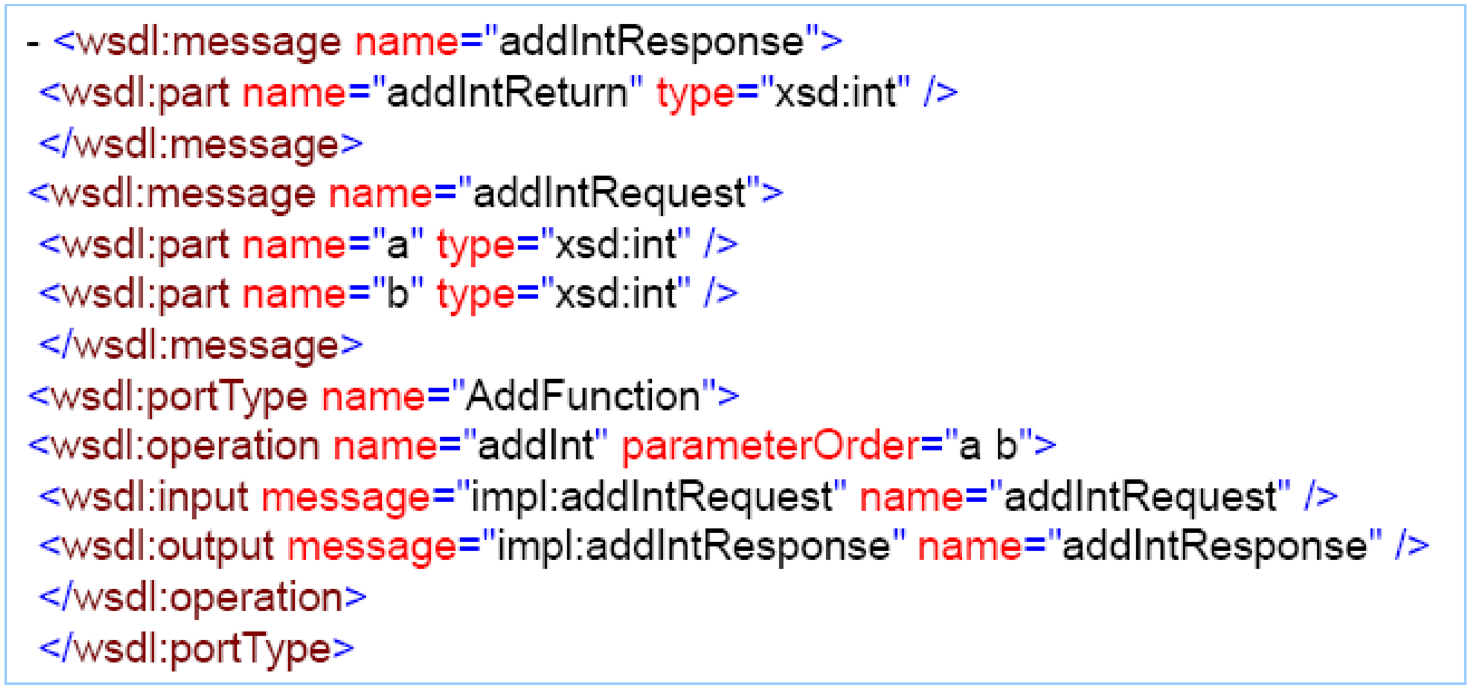
WSDL Refresh

- A WSDL document describes
  - What the service can do
  - Where it resides
  - How to invoke it



The WSDL File

- Point your browser to
  - http://localhost:8080/axis/AddFunction.jws?wsdl
  - Examine its WSDL file.



```
- <wsdl:message name="addIntResponse">
 <wsdl:part name="addIntReturn" type="xsd:int" />
 </wsdl:message>
<wsdl:message name="addIntRequest">
 <wsdl:part name="a" type="xsd:int" />
 <wsdl:part name="b" type="xsd:int" />
 </wsdl:message>
<wsdl:portType name="AddFunction">
<wsdl:operation name="addInt" parameterOrder="a b">
 <wsdl:input message="impl:addIntRequest" name="addIntRequest" />
 <wsdl:output message="impl:addIntResponse" name="addIntResponse" />
 </wsdl:operation>
 </wsdl:portType>
```

Exercise (10 min)

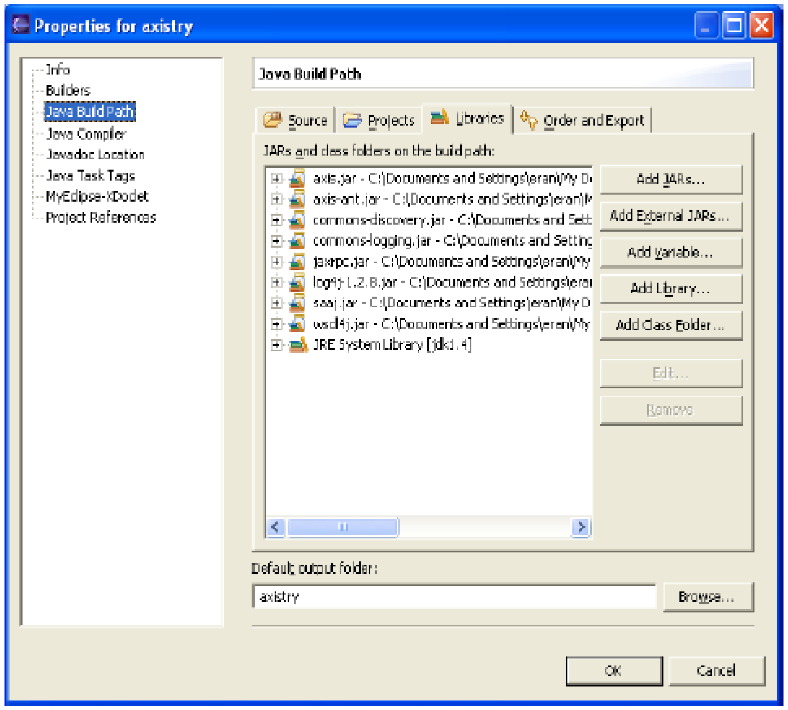
- Create new or use your existing addition class
- Deploy it as web service into AXIS
- Generate it’s WSDL by following steps explained earlier

Invoking JWS Service

- Two options
  - Using Dynamic Invocation Interface (DII)
  - Using Stubs generated from service WSDL description

Simple Client Test: Using DII with Eclipse

- Create new java Project
- Add all the jars under "…\webapps\axis\WEB-INF\lib" to the java build path libraries (using external jars)
- Create a new class, called "TestAddFunction"

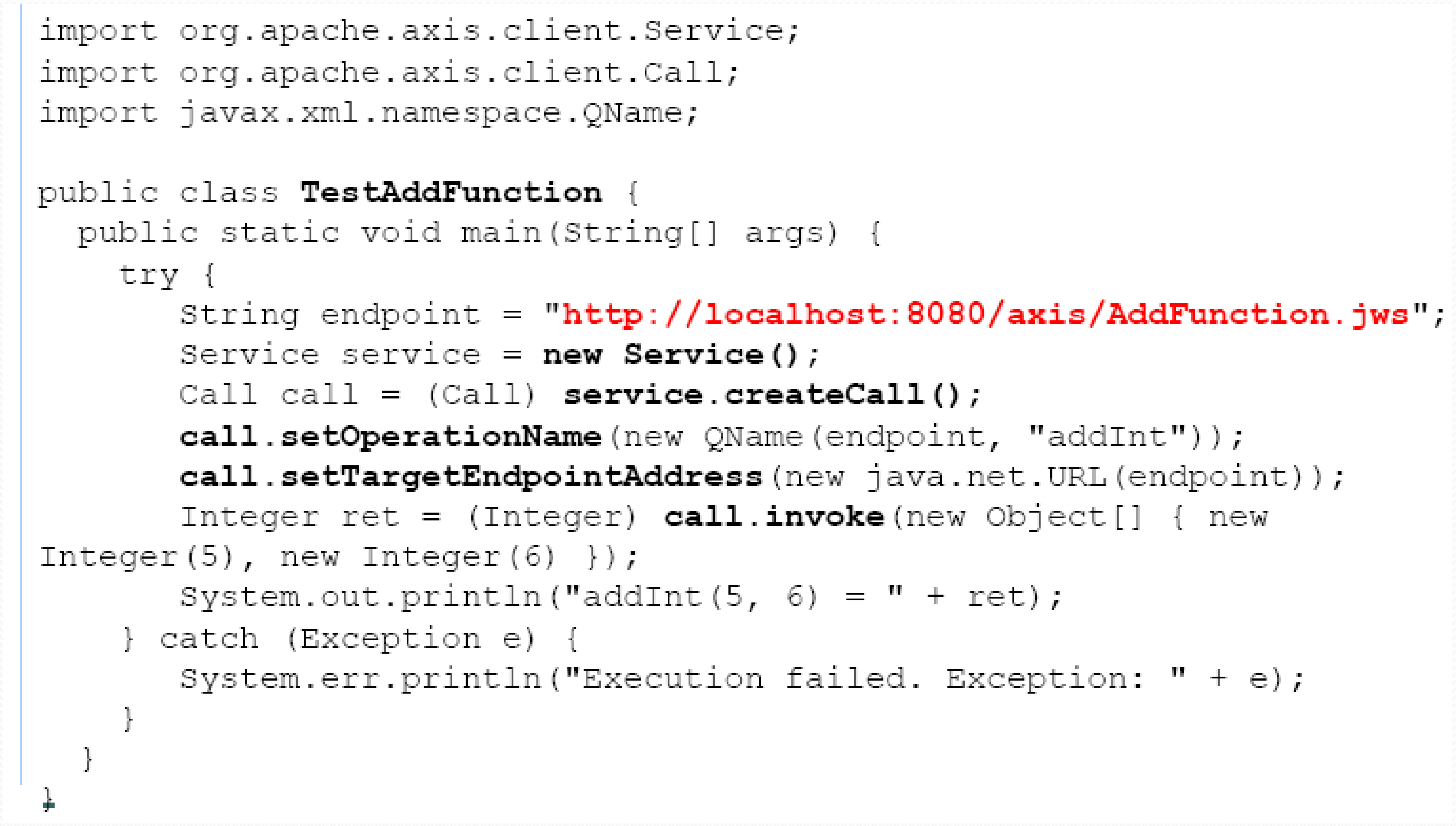


JARs and class folders on the build path:

```
axis.jar - C:\Documents and Settings\eran\My D
axis-ant.jar - C:\Documents and Settings\eran\M
commons-discovery.jar - C:\Documents and Sett
commons-logging.jar - C:\Documents and Settinc
jaxrpc.jar - C:\Documents and Settings\eran\My
log4j-1.2.8.jar - C:\Documents and Settings\erar
saaj.jar - C:\Documents and Settings\eran\My D
wsdl4j.jar - C:\Documents and Settings\eran\My
JRE System Library [jdk1.4]
```

Default output folder: axistry

Client Code: DII Method



```
import org.apache.axis.client.Service;
import org.apache.axis.client.Call;
import javax.xml.namespace.QName;

public class TestAddFunction {
  public static void main(String[] args) {
    try {
       String endpoint = "http://localhost:8080/axis/AddFunction.jws";
       Service service = new Service();
       Call call = (Call) service.createCall();
       call.setOperationName(new QName(endpoint, "addInt"));
       call.setTargetEndpointAddress(new java.net.URL(endpoint));
       Integer ret = (Integer) call.invoke(new Object[] { new
Integer(5), new Integer(6) });
       System.out.println("addInt(5, 6) = " + ret);
    } catch (Exception e) {
       System.err.println("Execution failed. Exception: " + e);
    }
  }
}
```

References

- Java api documentation: http://java.sun.com/javase/6/docs/api/
- Web Service: Axis

  http://ws.apache.org/axis/
- Gaia Trecarichi slides for Web Languages course