Setting up the WPS as an Multiple Module Eclipse Project

-Walkthrough-

1. Download the latest Eclipse Version
   (http://www.eclipse.org/downloads/)

2. Install Maven2
   (http://maven.apache.org/download.html)

   Unzip maven-2.0.9-bin.zip to the directory you wish to install Maven 2.0.9. These instructions assume you chose C:\Program Files\Apache Software Foundation\maven-2.0.9

   Add the bin directory to your path, by opening up the system properties (WinKey + Pause), selecting the "Advanced" tab, and the "Environment Variables" button, then editing the PATH variable in the user variables. eg. "C:\Program Files\Apache Software Foundation\maven-2.0.9\bin";%PATH%

   In the same dialog, make sure that JAVA_HOME is set to the location of your JDK, e.g. C:\Program Files\Java\jdk1.5.0_02

   Run `mvn --version` to verify that it is correctly installed.

   Run `mvn` to generate the local repository. (You will see some errors, but don`t mind)

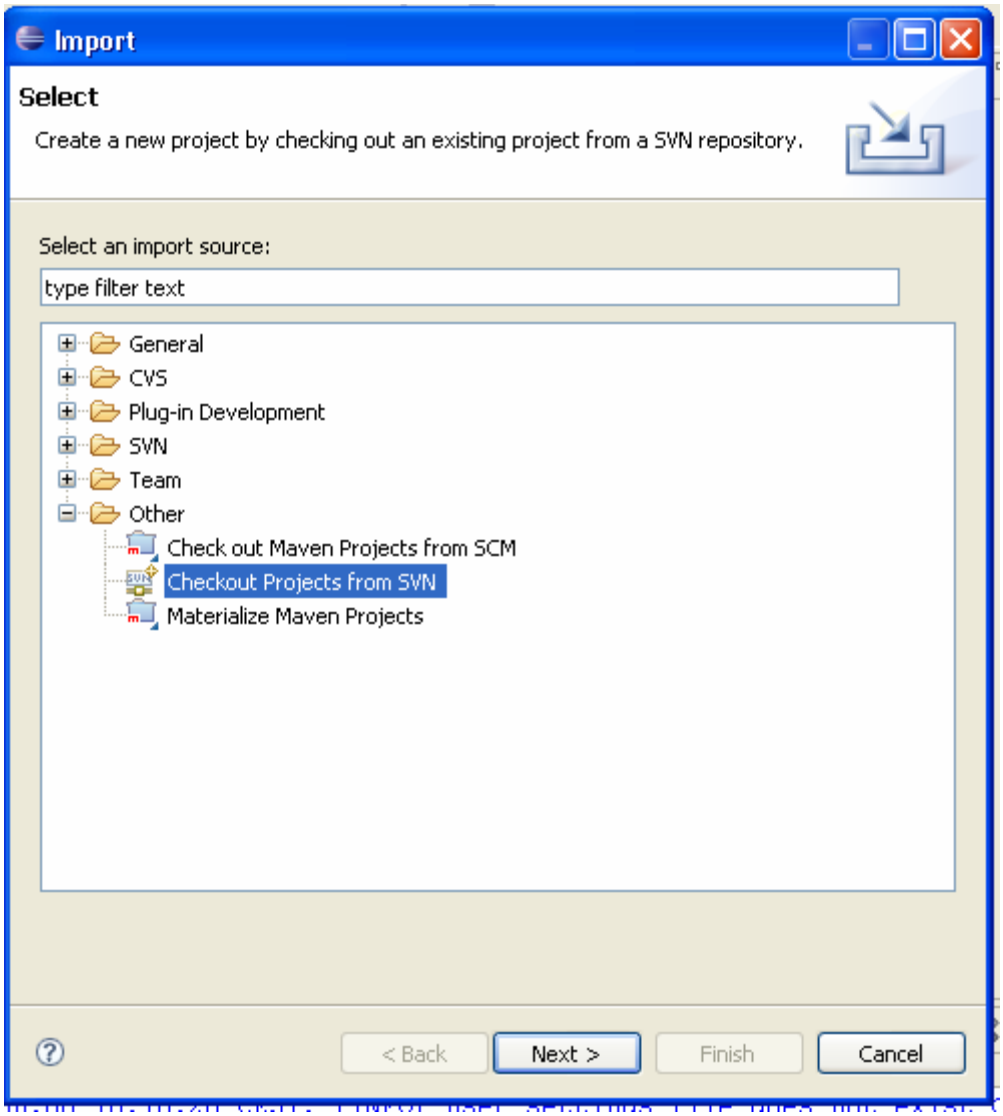
   Create a new folder `repository` located under the .m2 folder. The default path is:

   `C:\Documents and Settings\<your login name>\.m2`

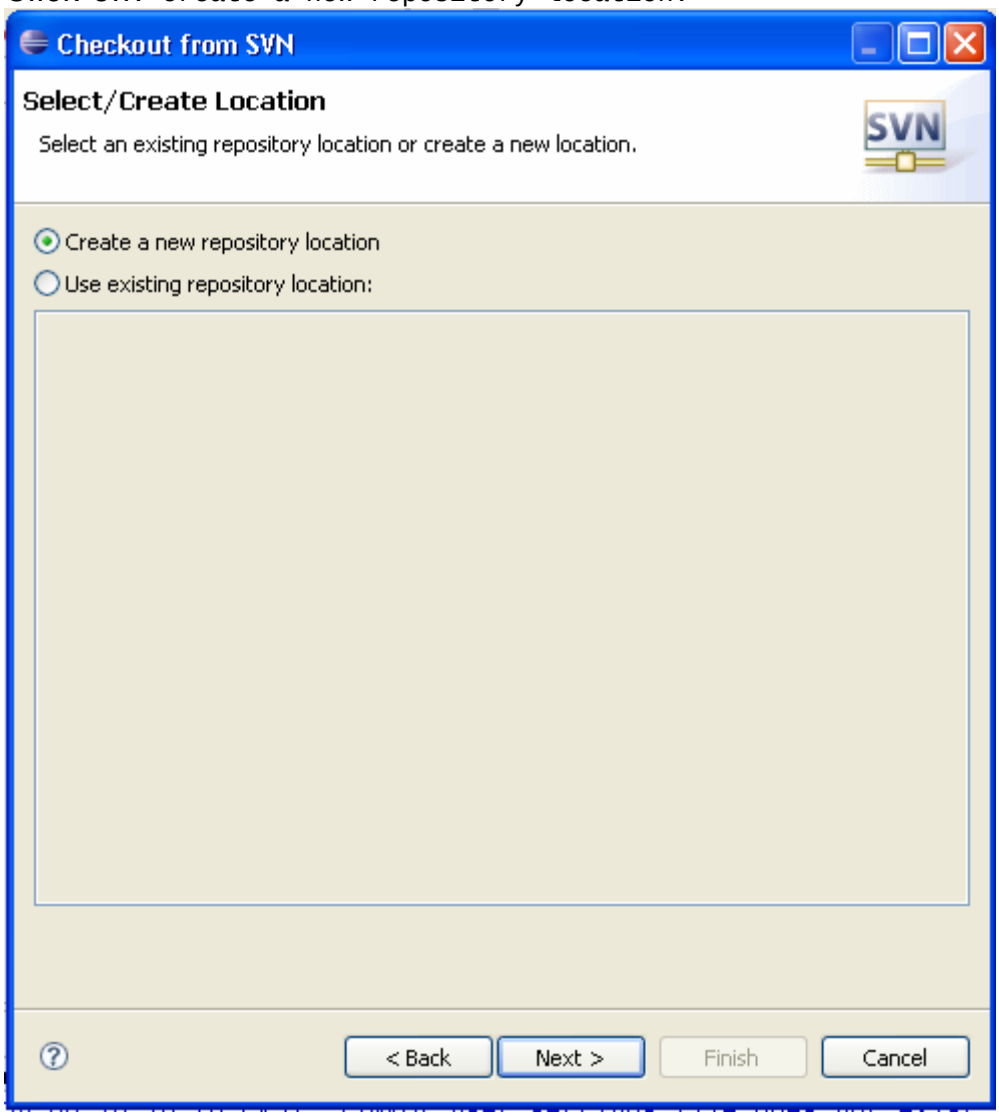
3. Install the Maven2 Plugin
   `Eclipse->Help->Software Updates->Find and Install-> Search for new Features to install->new remote site . Enter http://m2eclipse.sonatype.org/update/`
   as the update URL.

4. Install a SVN client
   We recommend to install Subversion as your Eclipse SVN client, but it should work with any SVN client. To install Subversion, `Eclipse->Help->Software Updates->Find and Install-> Search for new Features to install->new remote site . Enter` http://subclipse.tigris.org/update_1.2.x
   as the update URL and follow the steps.

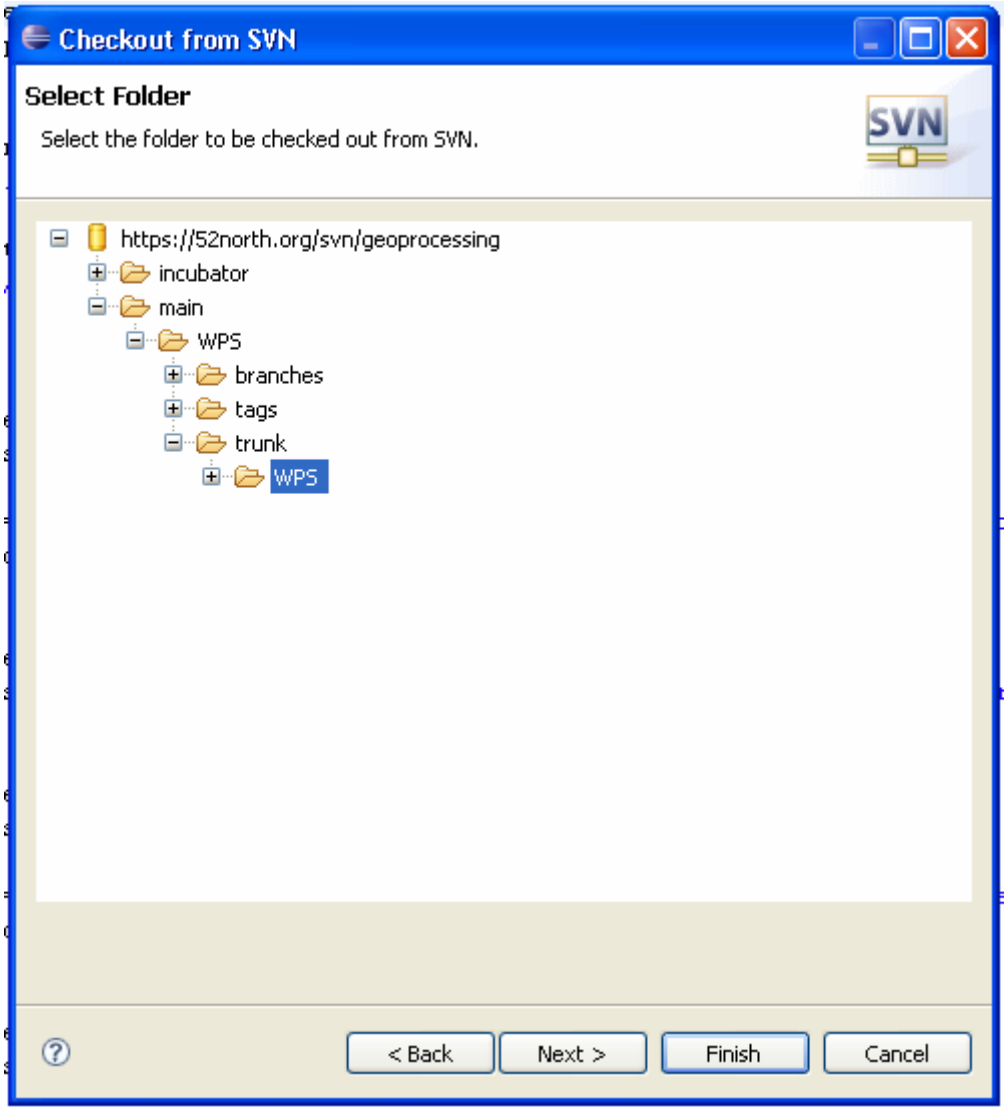
5. Get the source from the 52North SVN Eclipse->File->Import



Click on: `create a new repository location.`

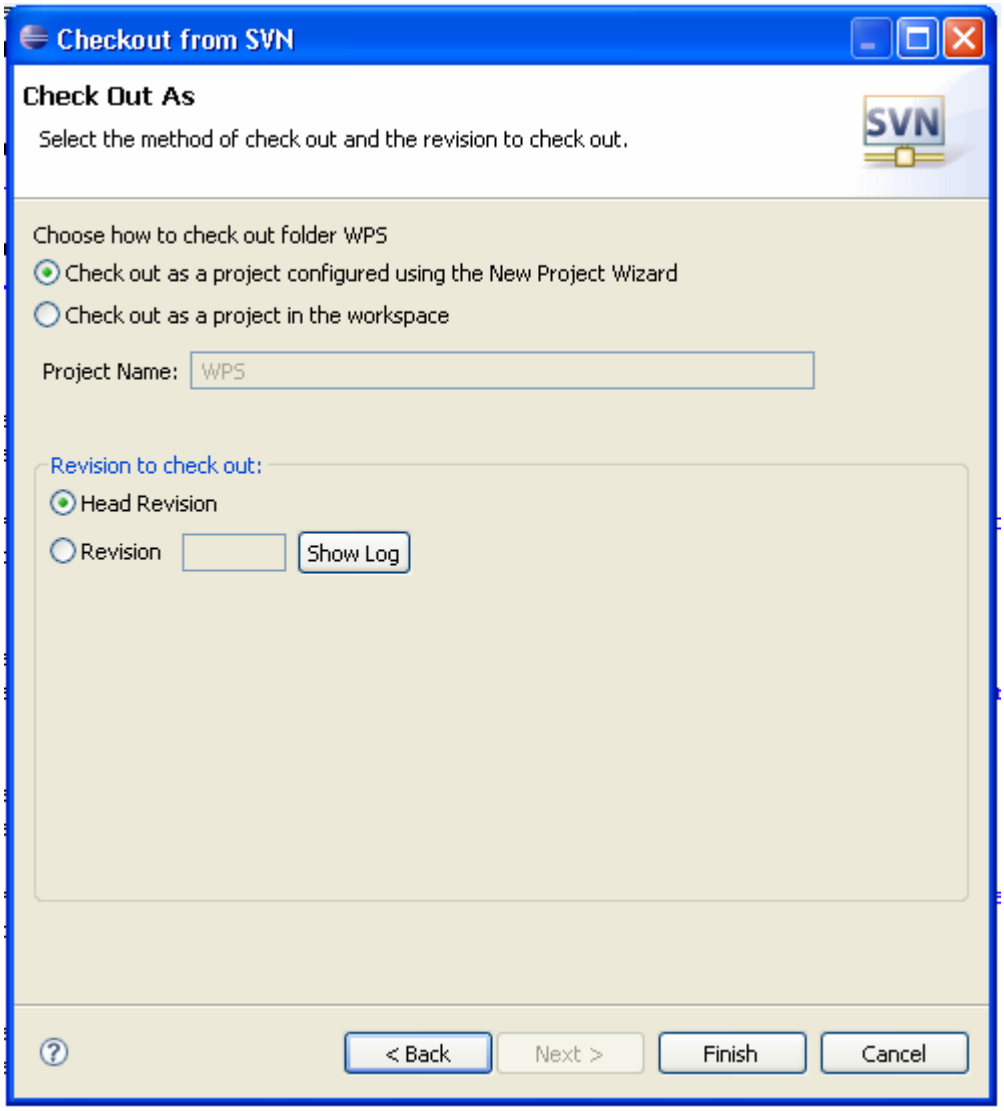


Use the following URL:
https://52north.org/svn/geoprocessing

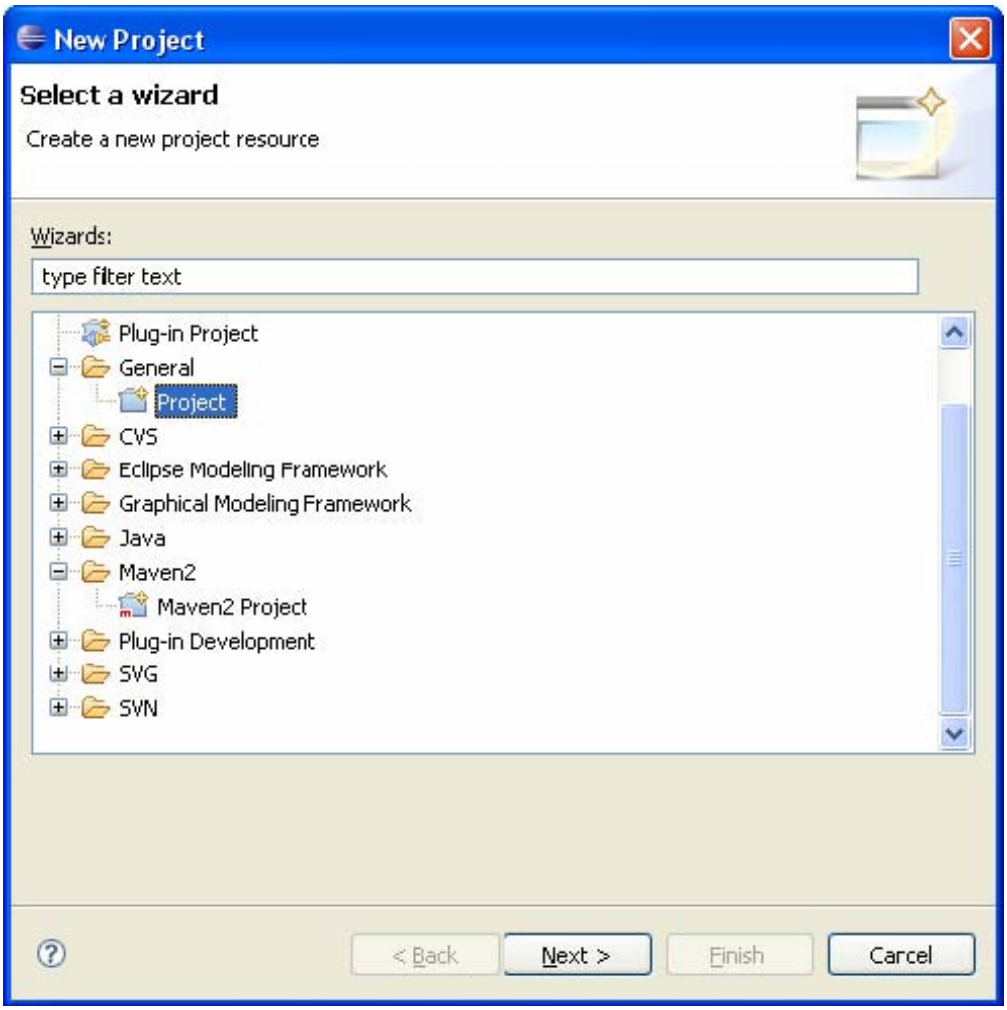


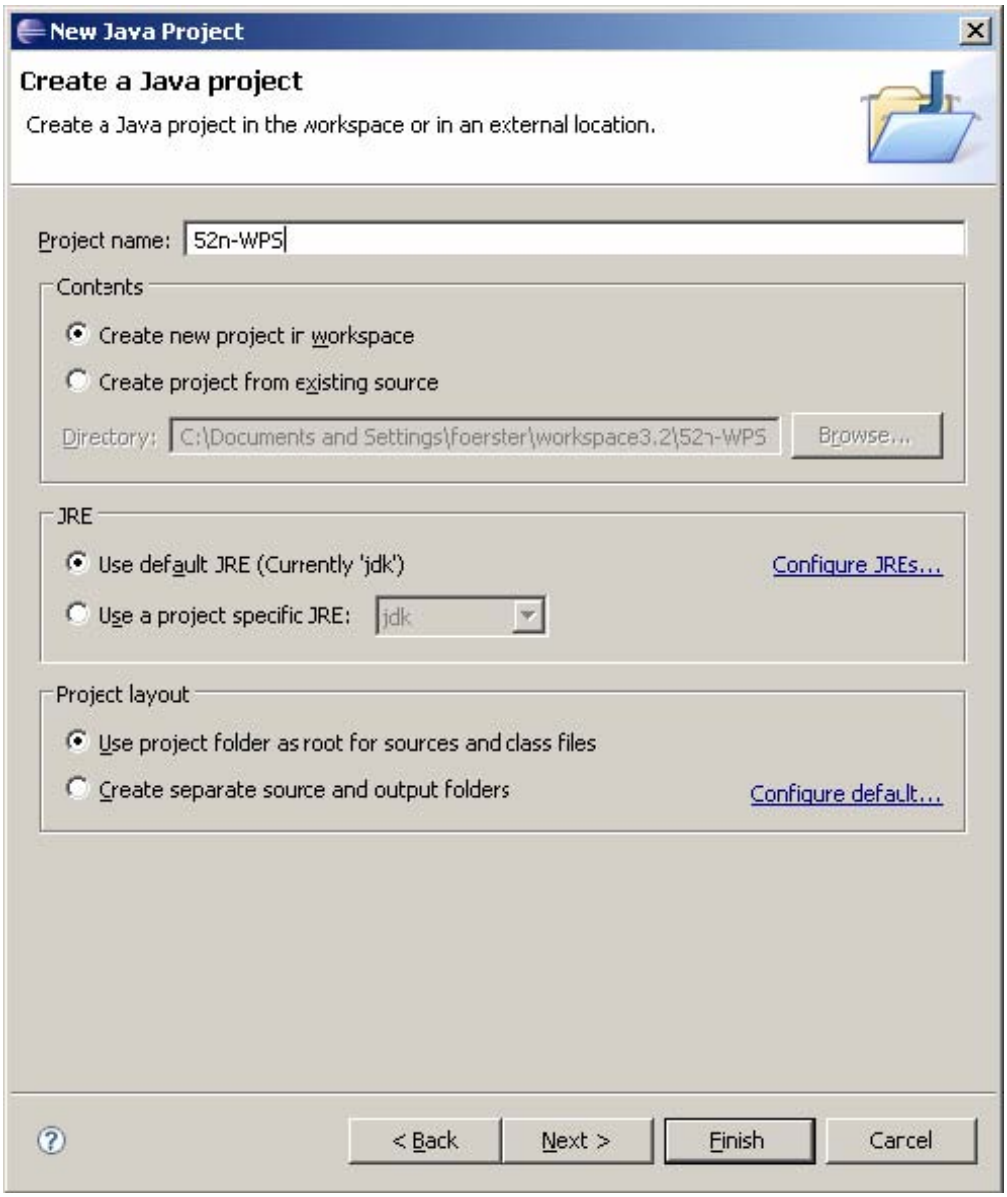
Navigate to main→WPS→trunk→WPS

Click next.



Click Finish. And create a new Project as shown in the next screenshot.

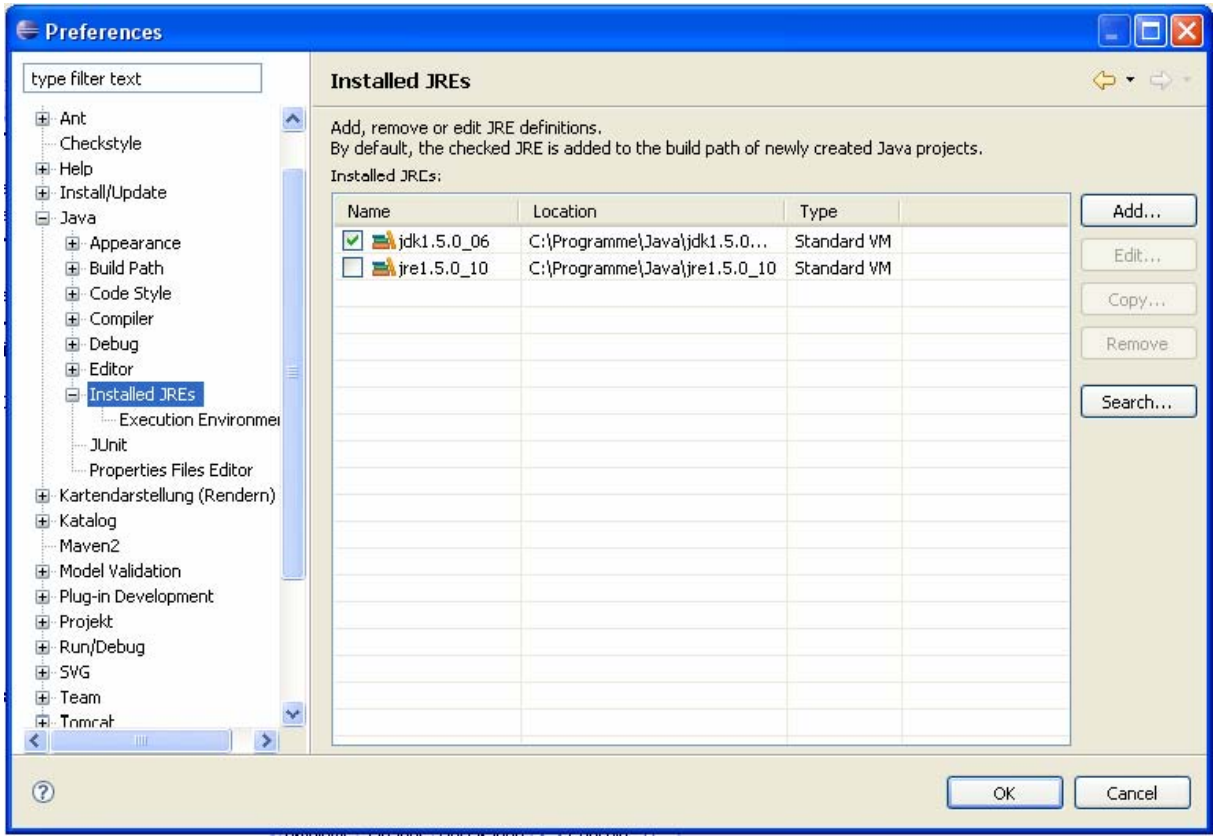

New Project
Select a wizard
Create a new project resource
Wizards:
type filter text
Plug-in Project
General
Project
CVS
Eclipse Modeling Framework
Graphical Modeling Framework
Java
Maven2
Maven2 Project
Plug-in Development
SVG
SVN
< Back
Next >
Finish
Cancel



5. Setting the project as an Maven2 Project

And make sure that you are using a JDK:

Go to Windows->Preferences->Java->Installed JREs



Edit the `settings.xml` from your conf folder located under the maven install folder.

Add the following tags: Under the <settings> tag <localRepository>

```
C:\Documents and Settings\<your login name>\.m2\repository

</localRepository>
```

and under the <profiles> tag

```
<profiles> <profile> <id>52n-start</id>
 <repositories>
  <repository>
   <id>n52-releases</id>
       <name>52n Releases</name>
       <url>http://52north.org/maven/repo/releases</url>
   <releases>

     <enabled>true</enabled>
   </releases>
   <snapshots>
```

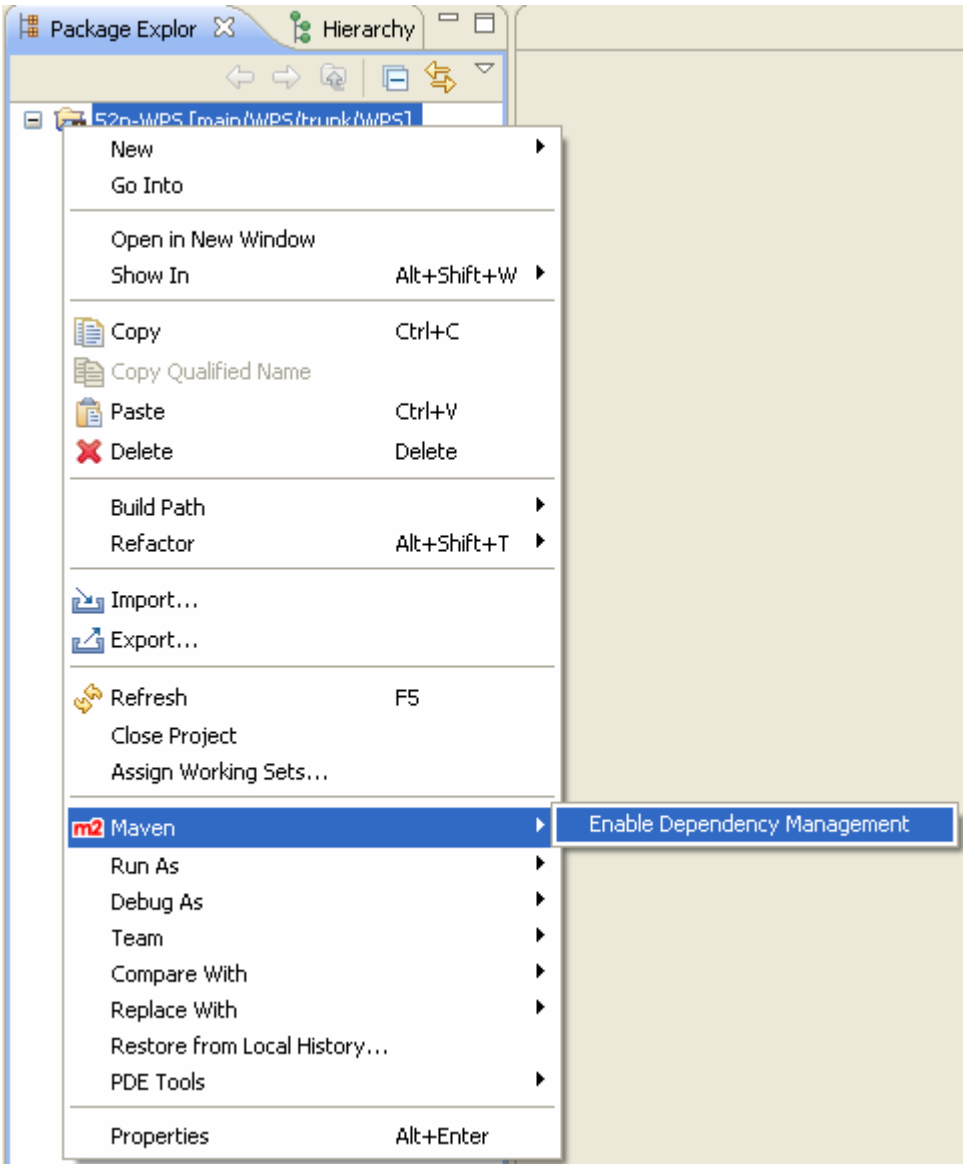
```
      <enabled>false</enabled>
     </snapshots>
   </repository>
   <repository>
   <id>geotools</id>
      <name>Geotools repository</name>
      <url>http://maven.geotools.fr/repository</url>

    </repository>
   <repository>
   <id>Refractions</id>
      <name>Refractions repository</name>
      <url>http://lists.refractions.net/m2</url>

    </repository>
<repository> <id>Apache</id>     <name>Apache repository</name>
   <url>http://repo1.maven.org/maven2</url>
  </repository> </repositories> </profile>
```

and under the <settings> tag

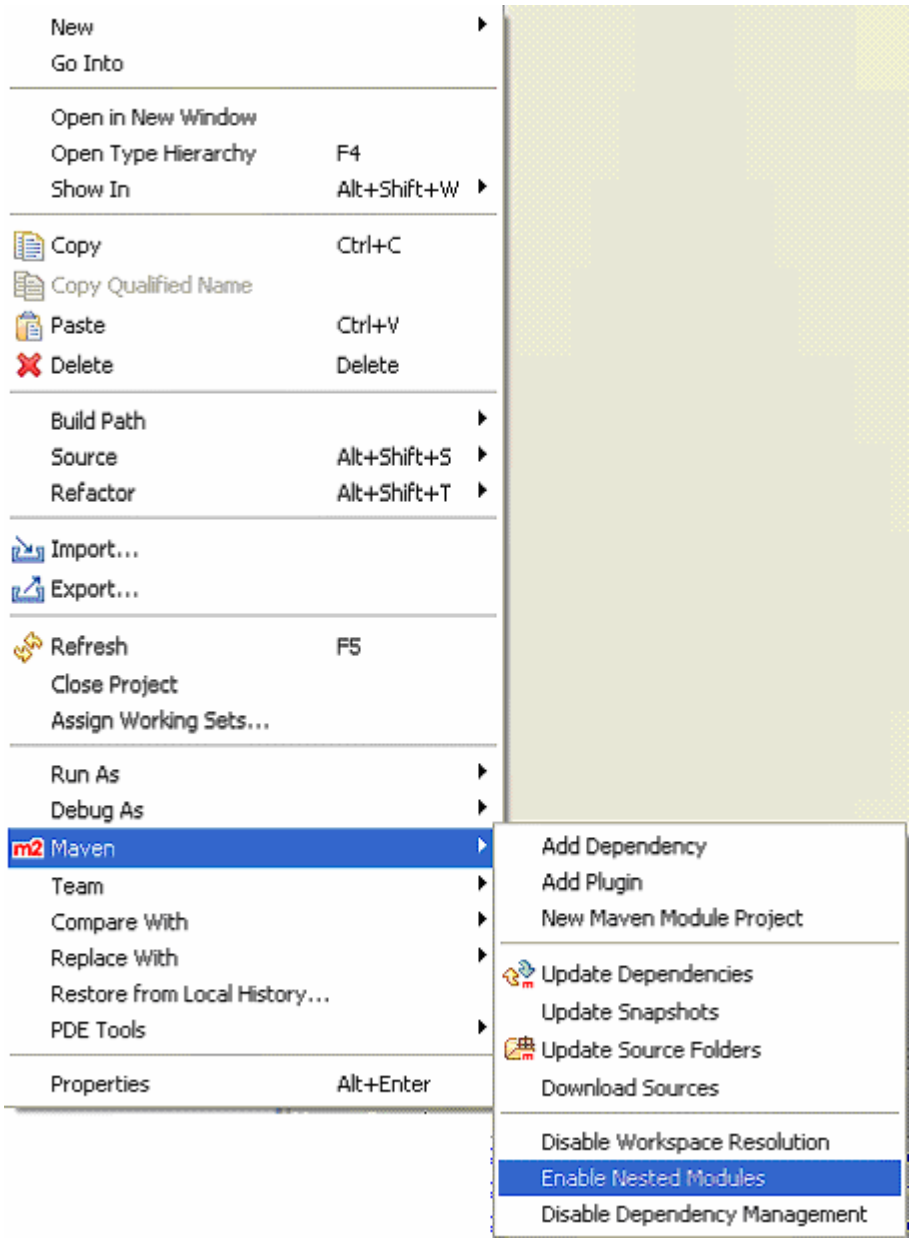
```
  <activeProfiles>
     <activeProfile>52n-start</activeProfile>
     </activeProfiles>
```

Now you are able to transform the downloaded project to a maven2 project.

Right click on your project in the package explorer. Select maven→Enable Dependeny Management as seen below:

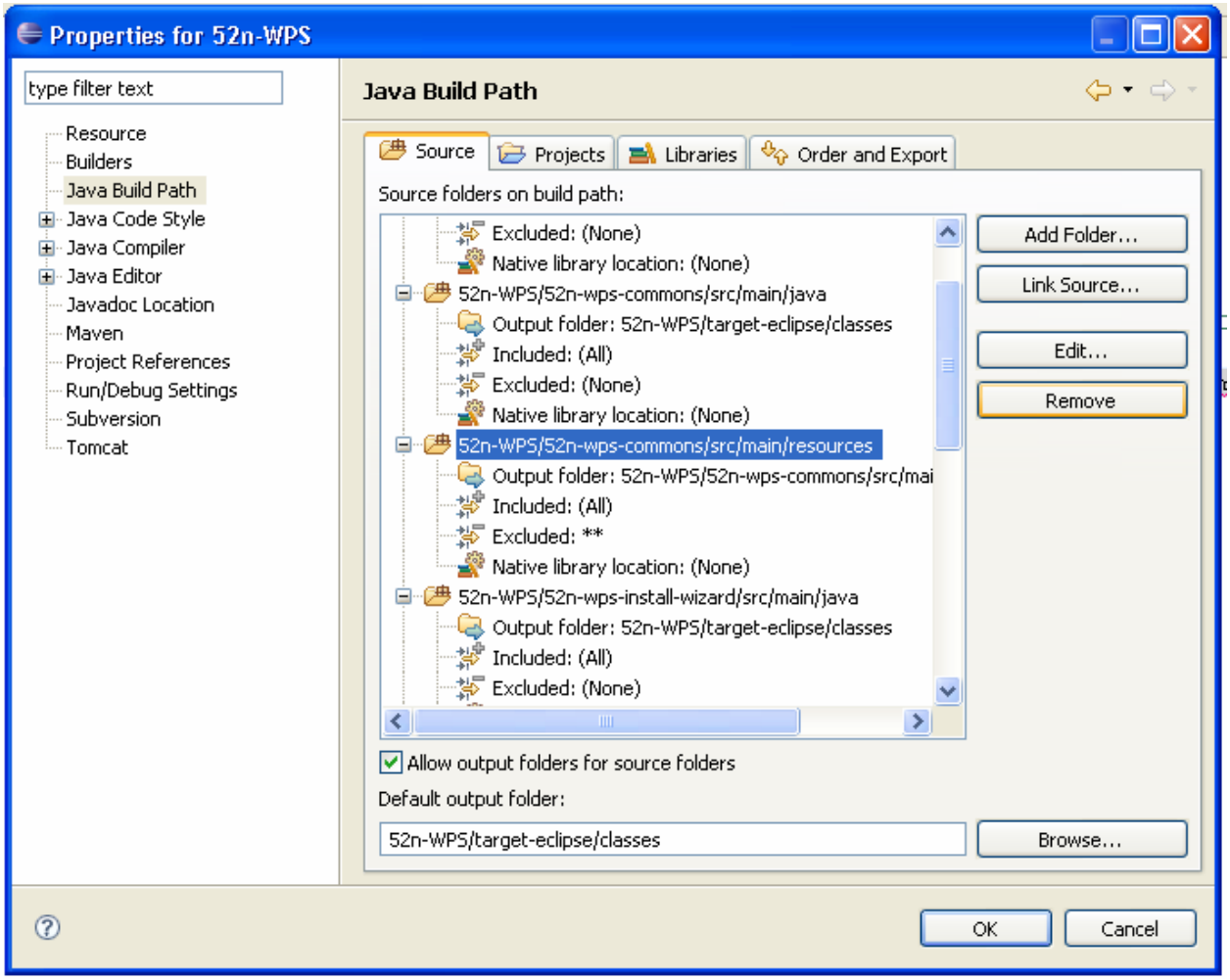


Again, right click on your project, maven→Enable Nested Modules.



Afer that, again right click on your project. Go to properties→java build path→ source

And remove the resources folder from the source folder list.


Properties for 52n-WPS
type filter text
Resource
Builders
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Maven
Project References
Run/Debug Settings
Subversion
Tomcat
Java Build Path
Source
Projects
Libraries
Order and Export
Source folders on build path:
Excluded: (None)
Native library location: (None)
52n-WPS/52n-wps-commons/src/main/java
Output folder: 52n-WPS/target-eclipse/classes
Included: (All)
Excluded: (None)
Native library location: (None)
52n-WPS/52n-wps-commons/src/main/resources
Output folder: 52n-WPS/52n-wps-commons/src/mai
Included: (All)
Excluded: **
Native library location: (None)
52n-WPS/52n-wps-install-wizard/src/main/java
Output folder: 52n-WPS/target-eclipse/classes
Included: (All)
Excluded: (None)
Add Folder...
Link Source...
Edit...
Remove
Allow output folders for source folders
Default output folder:
52n-WPS/target-eclipse/classes
Browse...
OK
Cancel

6. Configure the WPS

The WPS Configuration can be done via a XML file.

The precompiled .war file contains the wps_config.xml in the config folder.

The config XML file can be found in development version under 52n-wps-webapp/src/main/webapp/config/wps_config.xml

A sample Configuration can be found below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSpy v2007 sp2 (http://www.altova.com)-->
<WPSConfiguration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://n52.org/wps schema_v1.xsd" xmlns="http://n52.org/wps">
```

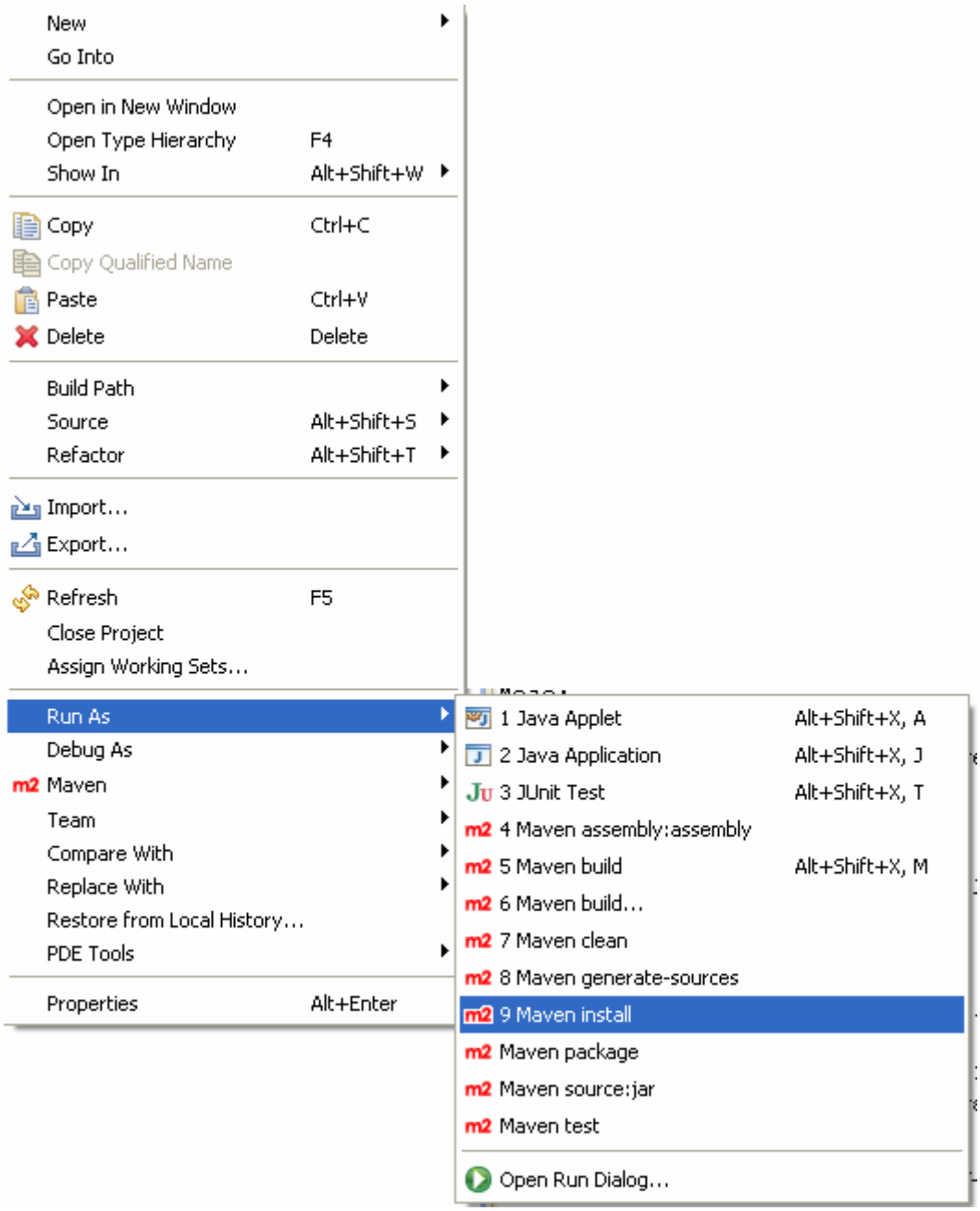
```
    <Datahandlers>
        <ParserList>
            <Parser name="SimpleGMLParser"
className="org.n52.wps.io.xml.SimpleGMLParser"/>
            <Parser name="GML2BasicParser"
className="org.n52.wps.io.xml.GML2BasicParser">
                <Property name="fcBufferTimeout">10000</Property>
            </Parser>
            <Parser name="AsciiGrassParser"
className="org.n52.wps.io.binary.AsciiGrassParser"/>
            <Parser name="DummyParser" className="org.n52.wps.io.xml.DummyParser"/>
            <Parser name="DummyParser" className="org.n52.wps.io.xml.OandMParser"/>
        </ParserList>
        <GeneratorList>
            <Generator name="SimpleGMLGenerator"
className="org.n52.wps.io.xml.SimpleGMLGenerator"/>
            <Generator name="GML2BasicGenerator"
className="org.n52.wps.io.xml.GML2BasicGenerator">
                <Property name="featureTransformerIncludeBounding">false</Property>
                <Property name="featureTransformerDecimalPlaces">4</Property>
            </Generator>
            <Generator name="AsciiGrassGenerator"
className="org.n52.wps.io.binary.AsciiGrassGenerator"/>
            <Generator name="KMLGenerator" className="org.n52.wps.io.xml.KMLGenerator"/>
            <Generator name="DummyParser" className="org.n52.wps.io.xml.OandMGenerator"/>
        </GeneratorList>
    </Datahandlers>
    <AlgorithmRepositoryList>
        <Repository name="LocalAlgorithmRepository"
className="org.n52.wps.server.LocalAlgorithmRepository">
            <Property
name="Algorithm">org.n52.wps.server.algorithm.SimpleBufferAlgorithm</Property>
            <Property
name="Algorithm">org.n52.wps.server.algorithm.simplify.DouglasPeuckerAlgorithm</Property
>
            <Property
name="Algorithm">org.n52.wps.server.algorithm.simplify.TopologyPreservingSimplificationAl
gorithm</Property>
        </Repository>
    </AlgorithmRepositoryList>
    <Server hostport="8080" includeDataInputsInResponse="false" hostname="localhost"
computationTimeoutMilliSeconds="5" cacheCapabilites="true">
        <Database/>
    </Server>
</WPSConfiguration>
```

The main structure lists all parser, generators and Algorithm Repositories. The default AlgorithmRepository is the LocalAlgortihmRepository. This repository acts as a container for all "local" processes. "Local" means in this context, that the java class files exist on the same machine and the same JVM.

However, other parsers, generatorsm, repositories and general server settings can be added and configured here.

7. Compile

To compile your maven project, right click on your project→run as → maven install



As a result, you should see in the console a:

```
Problems  Javadoc  Declaration  Console
<terminated> Executing install in C:/Dokumente und Einstellungen/bs1980x/Desktop/Müll/WPS Test Install/52n-WPS [Maven Build] C:\Programme\Ja
[INFO] Reactor Summary:
[INFO] ------------------------------------------------------------------------
[INFO] 52north processing ..................................... SUCCESS [2.640s]
[INFO] 52north 52n-wps-commons ............................... SUCCESS [3.203s]
[INFO] 52north 52n-wps-io .................................... SUCCESS [17.746s]
[INFO] 52north 52n-wps-server ................................ SUCCESS [19.527s]
[INFO] 52north 52n-wps-nstall-wizard library ................. SUCCESS [2.109s]
[INFO] 52n WPS Web Application ............................... SUCCESS [21.574s]
[INFO] 52north 52n-wps-client library ........................ SUCCESS [6.983s]
[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1 minute 14 seconds
[INFO] Finished at: Wed Jun 18 10:45:45 CEST 2008
[INFO] Final Memory: 7M/44M
[INFO] ------------------------------------------------------------------------
```

8. Deploy the WPS as a Web application
You can choose many ways to deploy a Web Application in tomcat. In the following only one way is described for Tomcat 5.5:

Go to your Tomcat home directory → conf → Catalina → localhost

Create a new wps.xml file with the following content:
```
<Context path="/wps" privileged="true" docBase="<path to your WPS>\52n-wps-webapp\target\52n-wps-webapp-1.0-SNAPSHOT" debug="1"/>
```

where <path to your WPS> points to the folder where your wps project resides.

Save the file.

9. Set up the eclipse tomcat plugin
Download and Install the Sysdeo Tomcat plugin from here:
http://www.eclipsetotale.com/tomcatPlugin.html

Restart Eclipse.

Note the new symbols : 

Configure your tomcat via:
Window→Properties→ Tomcat
Enter your TOMCAT_HOME directoy.

To start tomcat click on: 

Your tomcat will start and your WPS will be available.

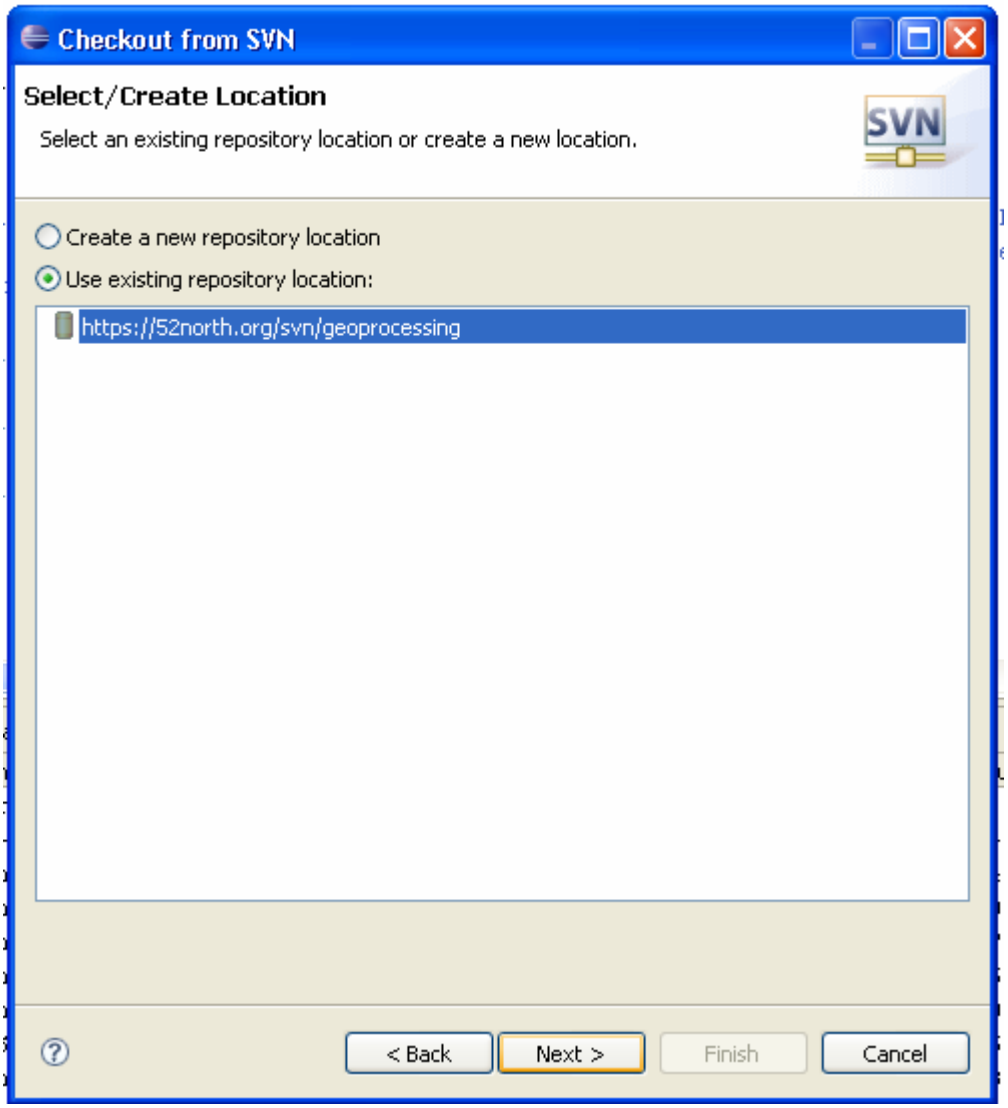
Whenever you make changes to your source code:

1. Compile again as described above.
2. Restart Tomcat.

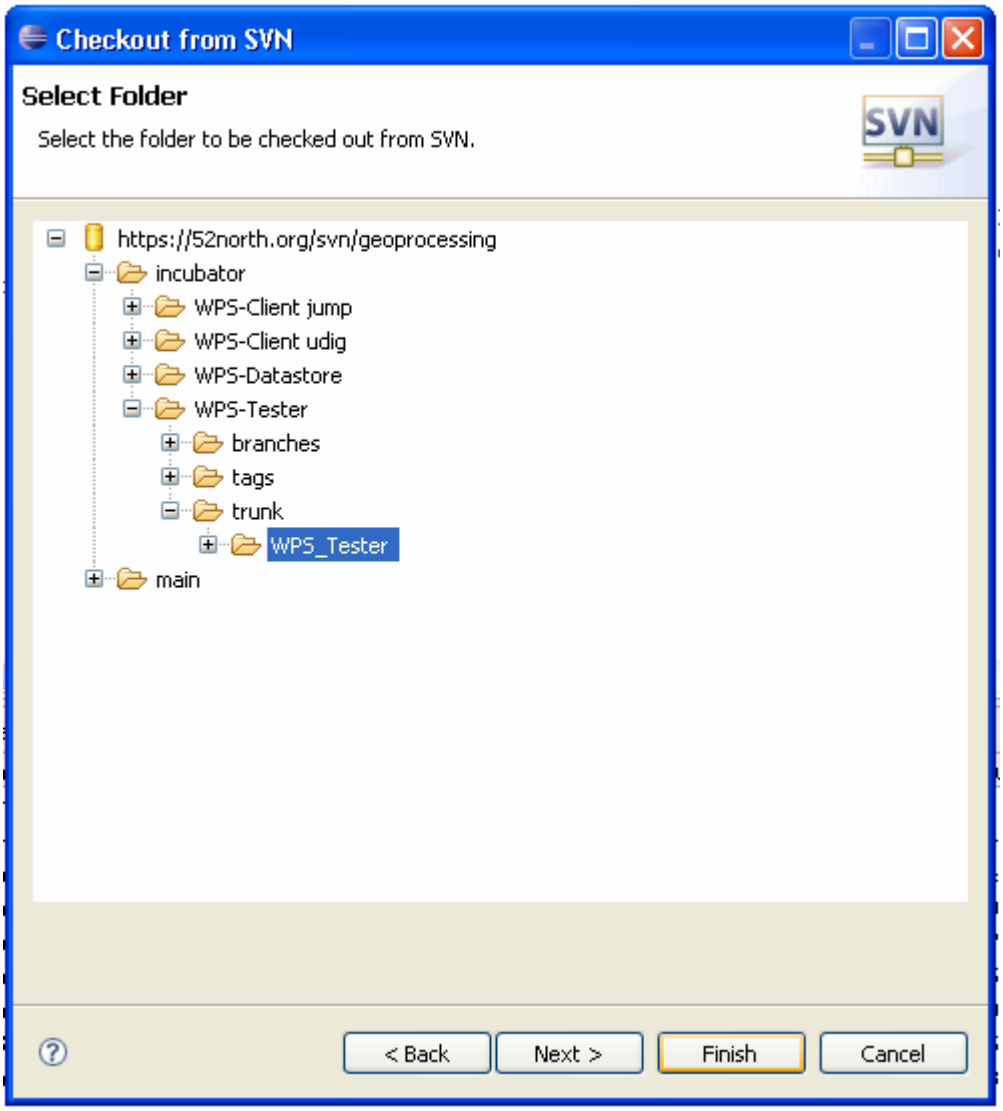
10. Test your installation

You can test your application with an additional project:

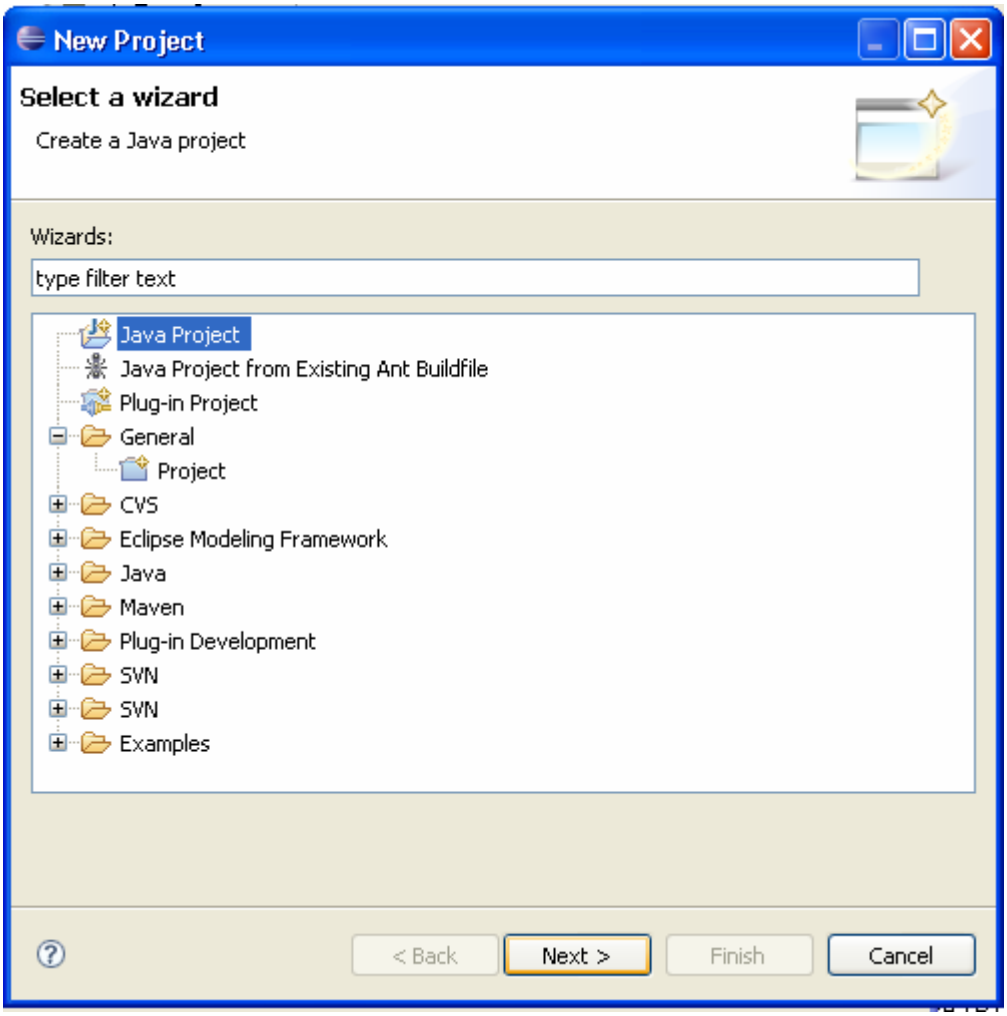
Import a new Project to your Workspace. Use again the SVN client with the existing URL:



This time, go to the incubator → WPS_Tester→Trunk→ WPS_Tester

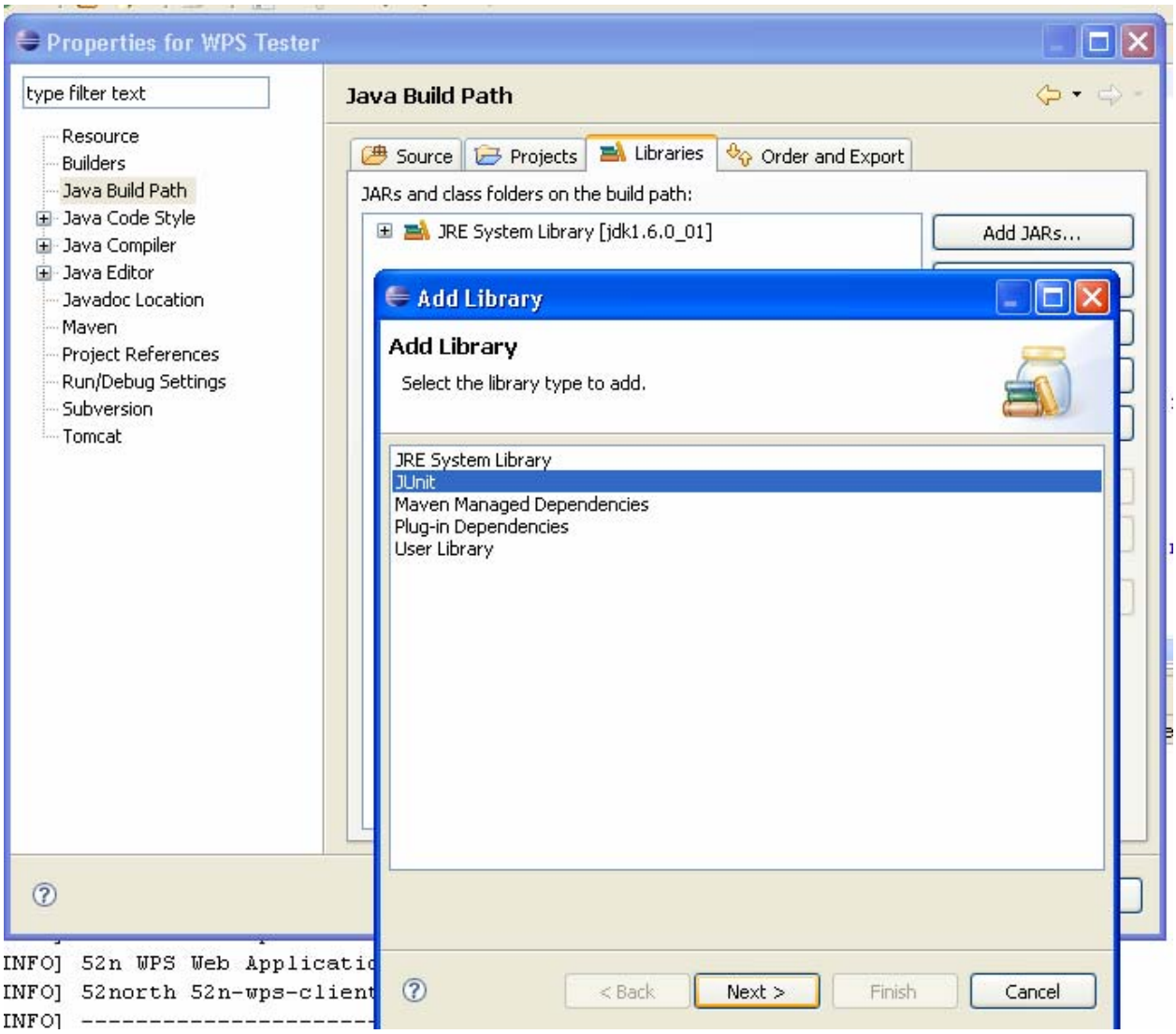


Click finish and create a new Java Project:

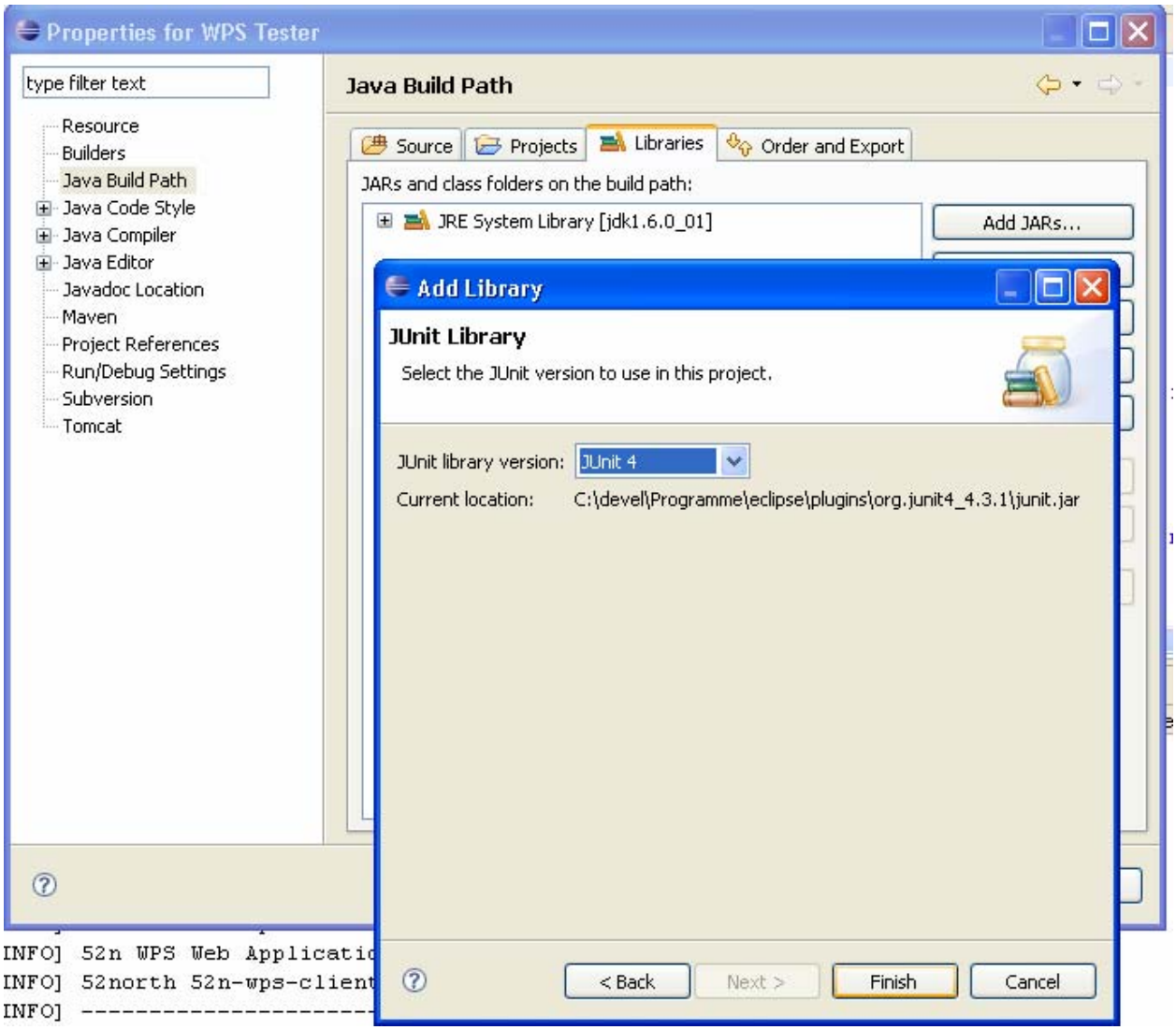


Now you have to add Junit to the classpath via:

Right mouse click on your project→properties→java build path → libraries → add Library. Select JUnit.

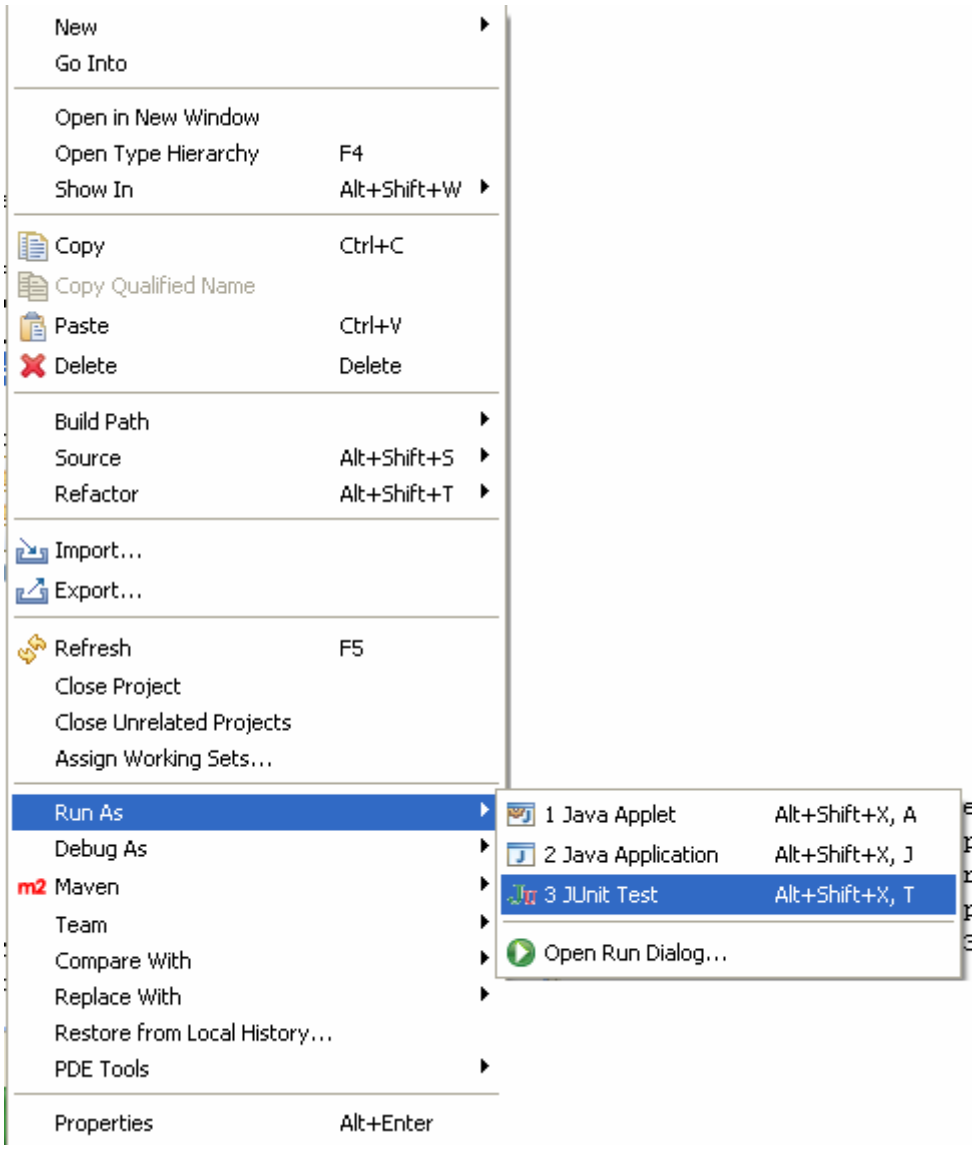


You need to add Junit4 and click finish.

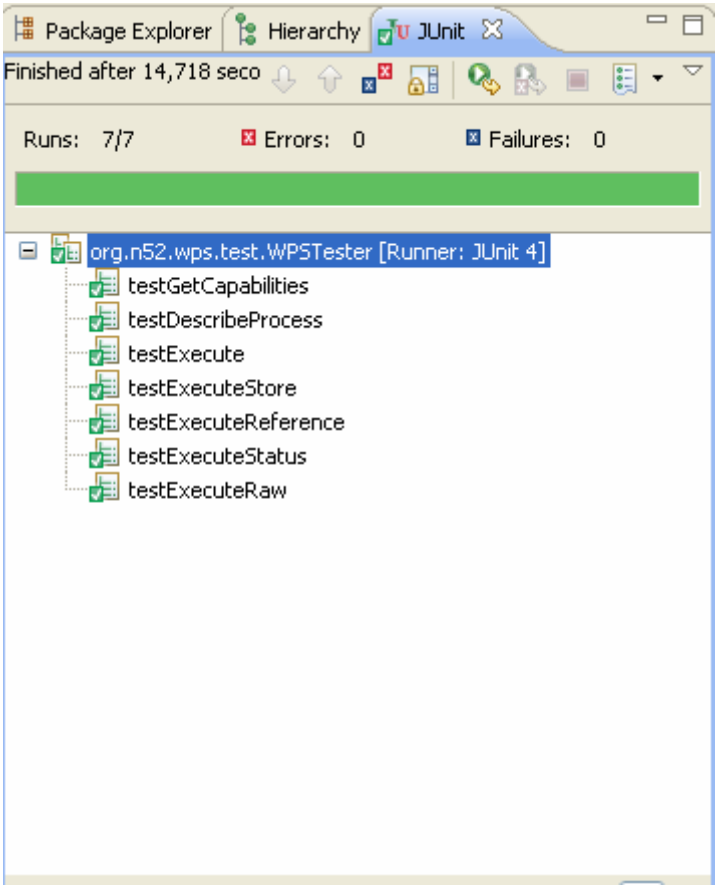


If your tomcat is not started-Start it now through 

After that, right click on your WPS_Tester project → run as → Junit 3 Test



The result should look like:



Your WPS is up and running. Have fun.