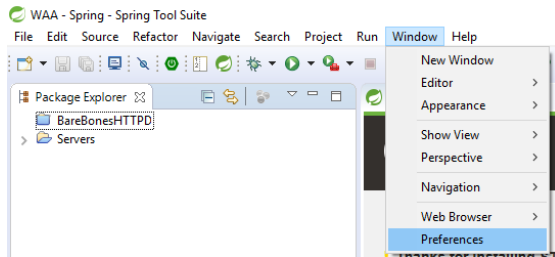


Lab 2

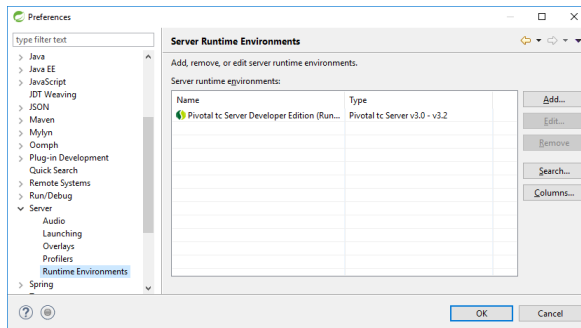
1. Installing Tomcat in STS

STS comes with TC as the development web server. If you prefer to use Tomcat, then follow these steps:

1. Go to the STS Preferences (**Windows\Preferences**)



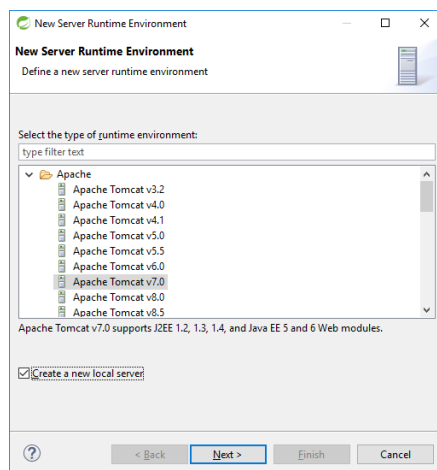
2. Select the **Server\Runtime Environments** from the left panel



3. Press the **Add...** button

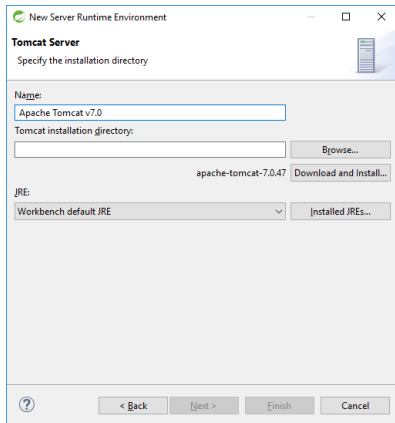
4. For this assignment you can install Apache\Apache Tomcat v7.0. If you have other version already installed or you prefer to install another one then select the corresponding version.

The following instructions are based on v7.0 because the wizard downloads Tomcat for you.



5. Select the **Create a new local server** check box and press **Next**.

6. If you have a Tomcat already installed, press Browse... and find the location of your Tomcat . Otherwise press the **Download and Install...** button, then accept the terms and select where to install Tomcat. **NOTE: Do not install it under Program Files in a Windows OS.**

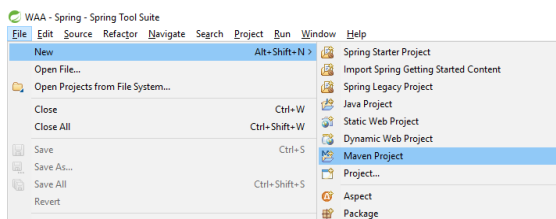


Once the wizard finishes the download it will enable the Finish button

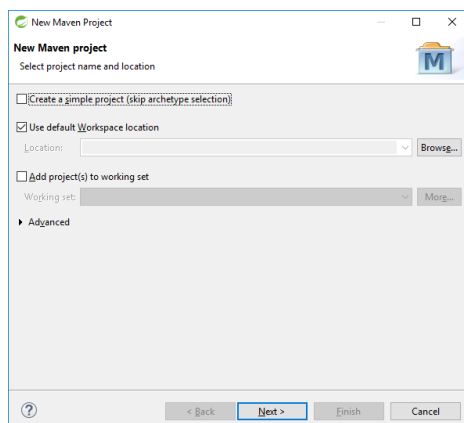
7. Press **Finish**

2. Creating a Servlet or JSP Project (Step by Step)

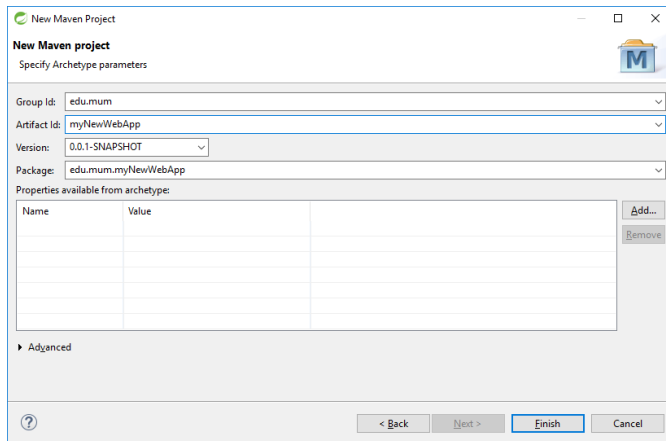
1. Create a Maven Project (**New\Maven Project**)



2. Verify that the **Create a simple project (skip archetype selection)** checkbox is checked and press **Next**.

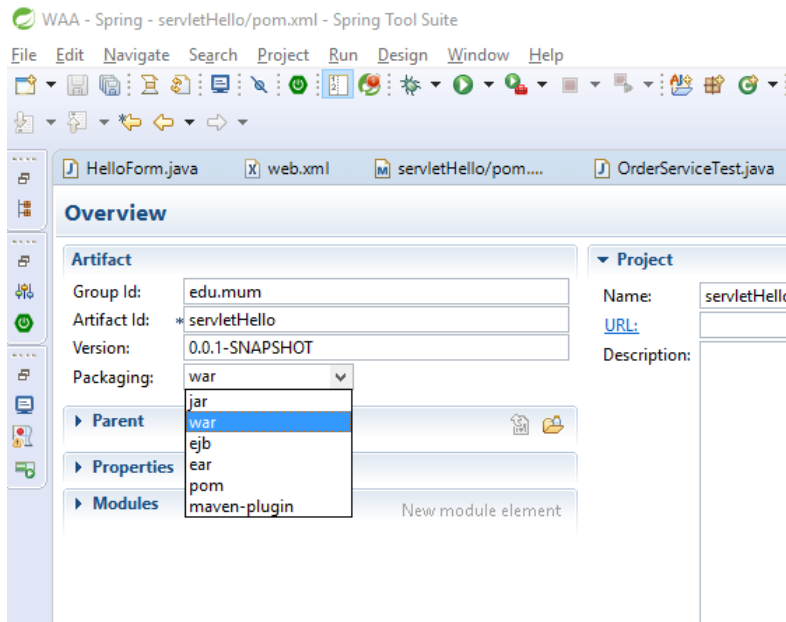


3. Set the name of the project's group and artifact, and press **Finish**.



The 'New Maven Project' dialog box in Spring Tool Suite. It shows the 'Specify Archetype parameters' tab. The 'Group Id' is set to 'edu.mum', 'Artifact Id' is 'myNewWebApp', 'Version' is '0.0.1-SNAPSHOT', and 'Package' is 'edu.mum.myNewWebApp'. Below these fields is a table for 'Properties available from archetype:' with columns 'Name' and 'Value'. At the bottom are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'.

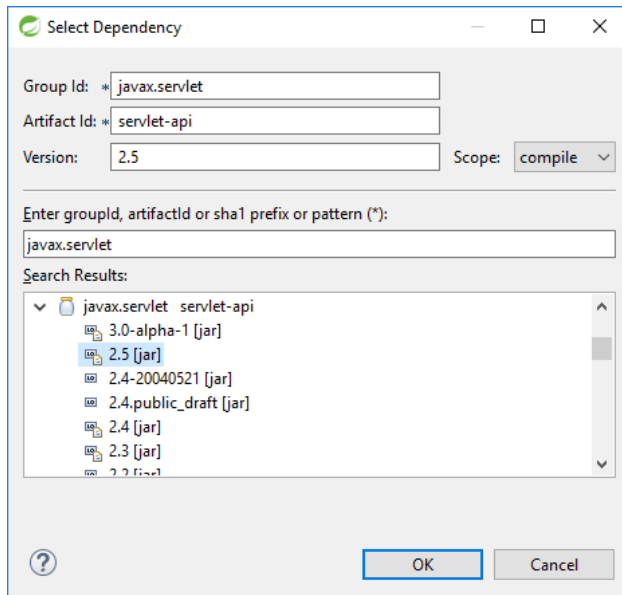
4. Open your pom.xml file and set the Packaging to **war**



The Spring Tool Suite IDE interface showing the 'pom.xml' file. The 'Overview' tab is active, displaying the 'Artifact' section with fields for 'Group Id: edu.mum', 'Artifact Id: *servletHello', 'Version: 0.0.1-SNAPSHOT', and 'Packaging: war'. A dropdown menu is open for 'Packaging', showing options: 'jar', 'war' (selected), 'ejb', 'ear', 'pom', and 'maven-plugin'. The 'Project' section on the right shows 'Name: servletHello' and 'URL:'. The 'Parent' and 'Modules' sections are also visible.

Or set it directly in the XML `<packaging>war</packaging>`

5. On the pom.xml file and go to the Dependencies tab and press Add. Filter by artifact and select javax.servlet servlet-api (the latest version is 2.5)



Alternatively, you could add your dependencies directly in the XML using the pom.xml tab

```
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>servlet-api</artifactId>
  <version>2.5</version>
</dependency>
```

6. Create a new folder under **/src/main/webapp**

Structure of the project

\pom.xml

\src\main\java

\src\main\resource

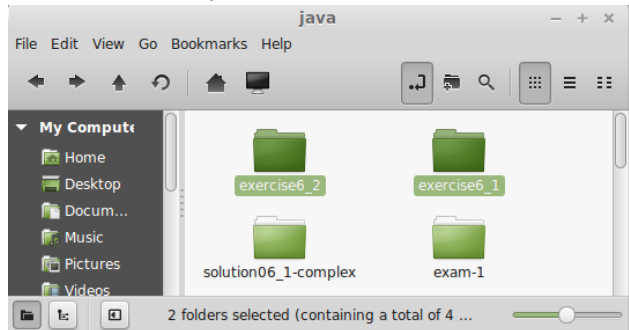
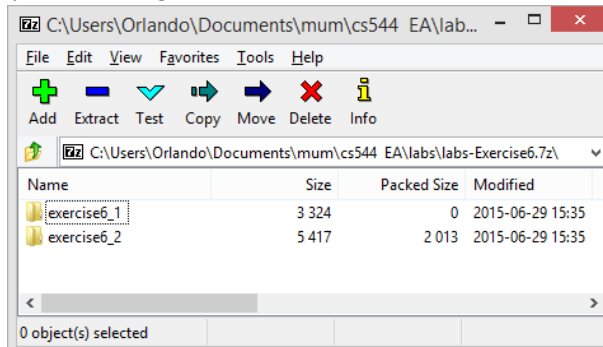
\src\main\webapp\WEB-INF

\src\main\webapp\WEB-INF\web.xml

3. Importing Maven Projects (Step by Step)

1. Unzip the project folders into your workspace:

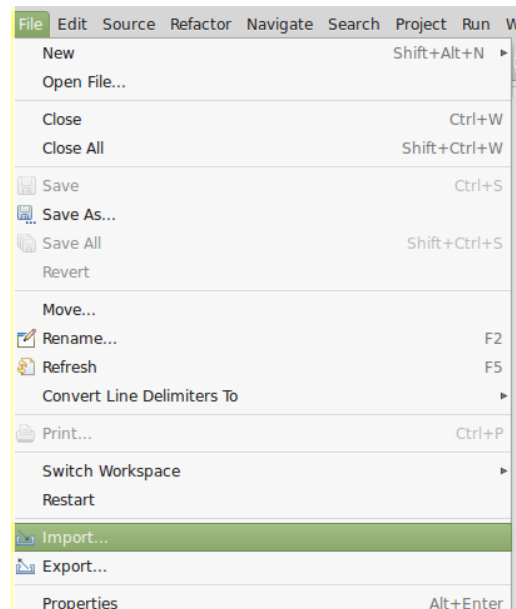
(These images are not based on the zip files I've provided so far)



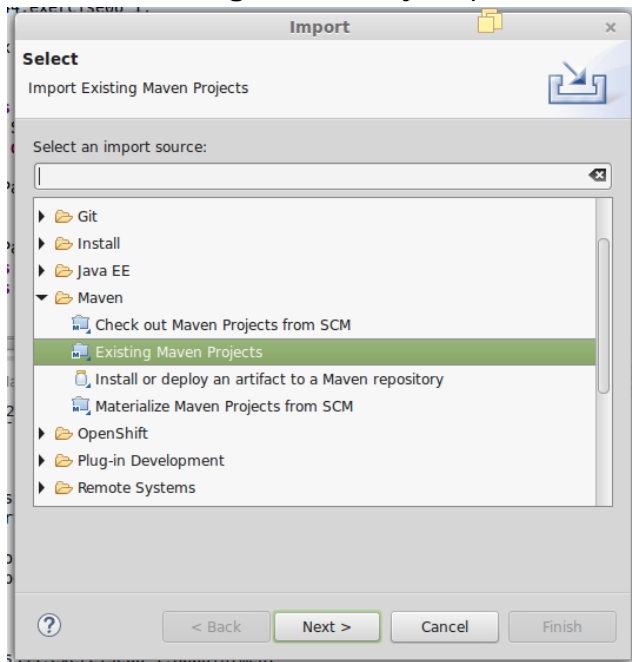
2. On Windows systems, verify that none of the project files are set to hidden

3. Open STS and select the corresponding workspace

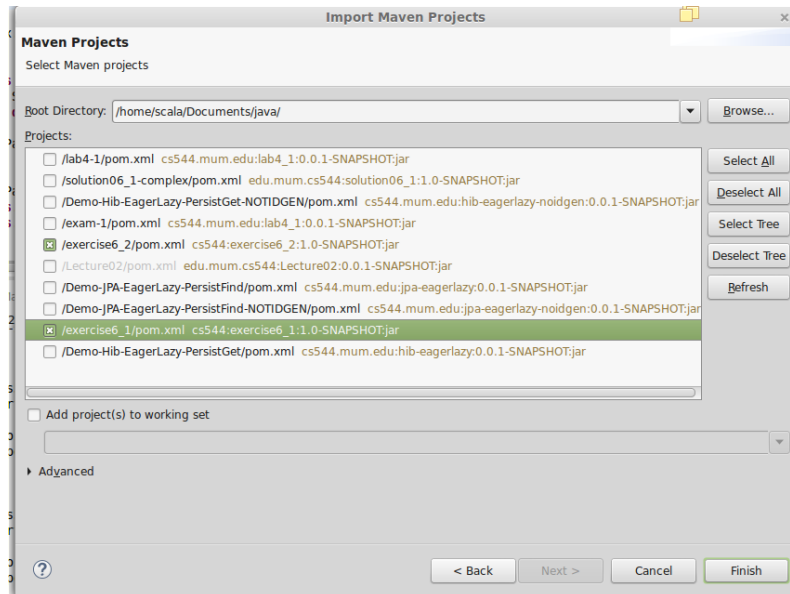
4. Choose the **File / Import...** option



5. Chose **Existing Maven Project**, press **Next**



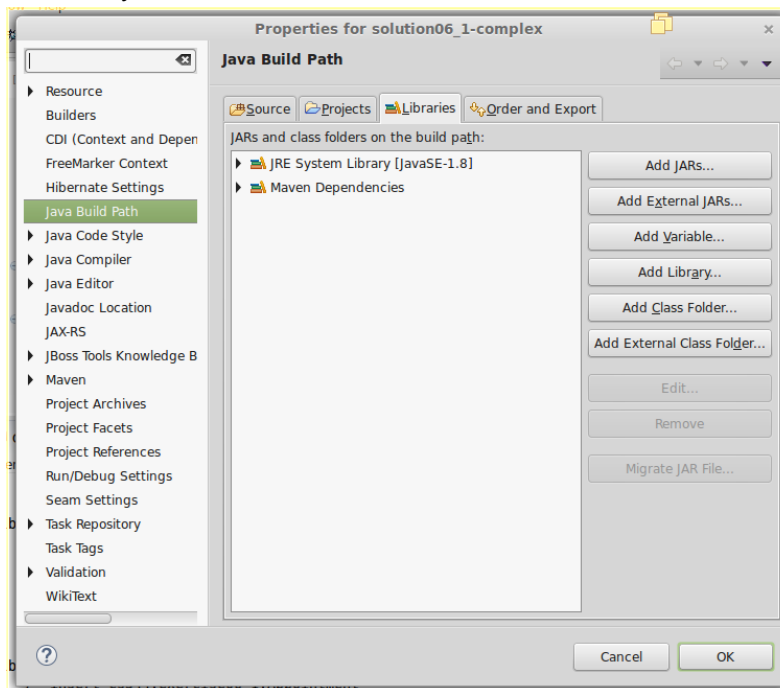
6. Select your workspace as the Root Directory (Use the Browse... button to refresh the list of projects)



7. **Deselect All** and mark the projects that you would like to import. Press **Finish**

8. **Run As / Maven Clean** to clear the target folder and then **Run As / Maven Install** to download jar dependencies and compile code.

NOTE: In some cases you may need to **Configure the Build Path** for your project to set the correct JRE System Library



4. Web App versions

When creating your web.xml you have to must set the web-app schema according to your target JavaEE environment. If you already have a web.xml created by a wizard or a Maven archetype, then you must verify it:

JavaEE 7 Servlet 3.1

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
  version="3.1">
</web-app>
```

JavaEE 6 Servlet 3.0

```
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0">
</web-app>
```

JavaEE 5 Servlet 2.5

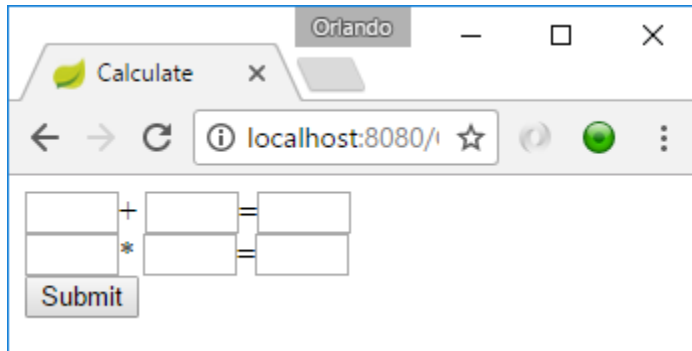
```
web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
  version="2.5">
</web-app>
```

5. Problem A: Servlet Calculator

Using only a servlet, create a web application that will render an HTML page that will allow the user to perform the functions of a simple calculator (addition, multiplication).

Use a POST method to submit the form values.

The following is a sample page to give you an idea of the expected form (but you are free to make it better):

A screenshot of a web browser window. The title bar shows 'Orlando' and standard window controls. The browser tab is labeled 'Calculate'. The address bar shows 'localhost:8080/'. The page content features a simple calculator interface with two rows of input fields. The first row has a plus sign between the first two fields and an equals sign between the second and third. The second row has a multiplication sign between the first two fields and an equals sign between the second and third. Below the input fields is a 'Submit' button.

Remember that you need to register your servlets in the web.xml file.

6. Problem B: JSP Calculator

Improve your calculator creating a JSP page instead of a servlet.

Note: you can add this option creating just a page in the previous project. There is no need to register a servlet for this page.

Test your calculator with concurrent clients with the help of your classmates.