Lab 1

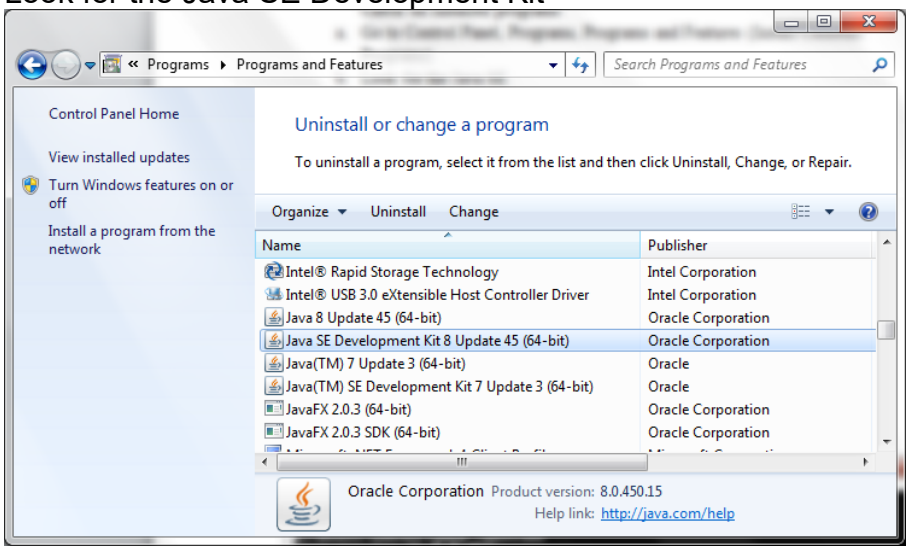
1. Environment Setup

Tool	Version	URL
Java JDK	8u161 or later	http://www.oracle.com/technetwork/java/javase/downloads/index.html
STS	3.9.2	https://spring.io/tools

The main objective of this exercise is to make sure that your development environment is working.

A) Installing Java

1. Verify installed Java version
   Check on Installed programs
   a. Go to Control Panel, Programs, Programs and Features (Install/Uninstall Programs)
   b. Look for the Java SE Development Kit



2. **Installing Java**
   Download Java JDK from Oracle web site
   Run the installer and take note of the Java installation path

B) Installing Spring Tool Suite (STS)

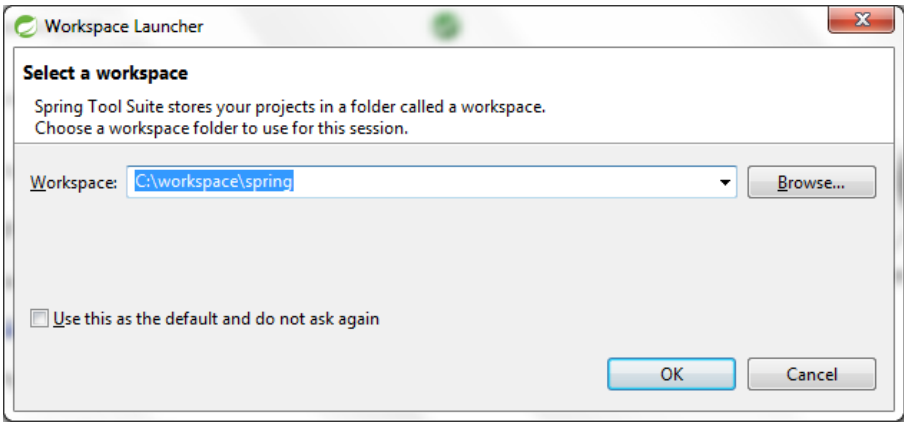
NOTE: STS is a version of Eclipse with some plugins added to improve the development experience. Using STS is not mandatory, if you prefer other IDE is okay but I won't be able to give you support for the IDE.

a. Download STS from Spring.io
b. Unzip STS on a folder outside Program Files
c. Create a shortcut for \sts-bundle\sts-3.9.2.RELEASE**STS.EXE**
d. Run STS.EXE

NOTE: You need internet while using Eclipse (STS)

e. Chose a folder for your projects when prompted to select a workspace

NOTE: DO NOT USE a workspace created by another Eclipse version. Eclipse (STS) stores information and plugins on the workspace. Changing versions may lead to conflicts.

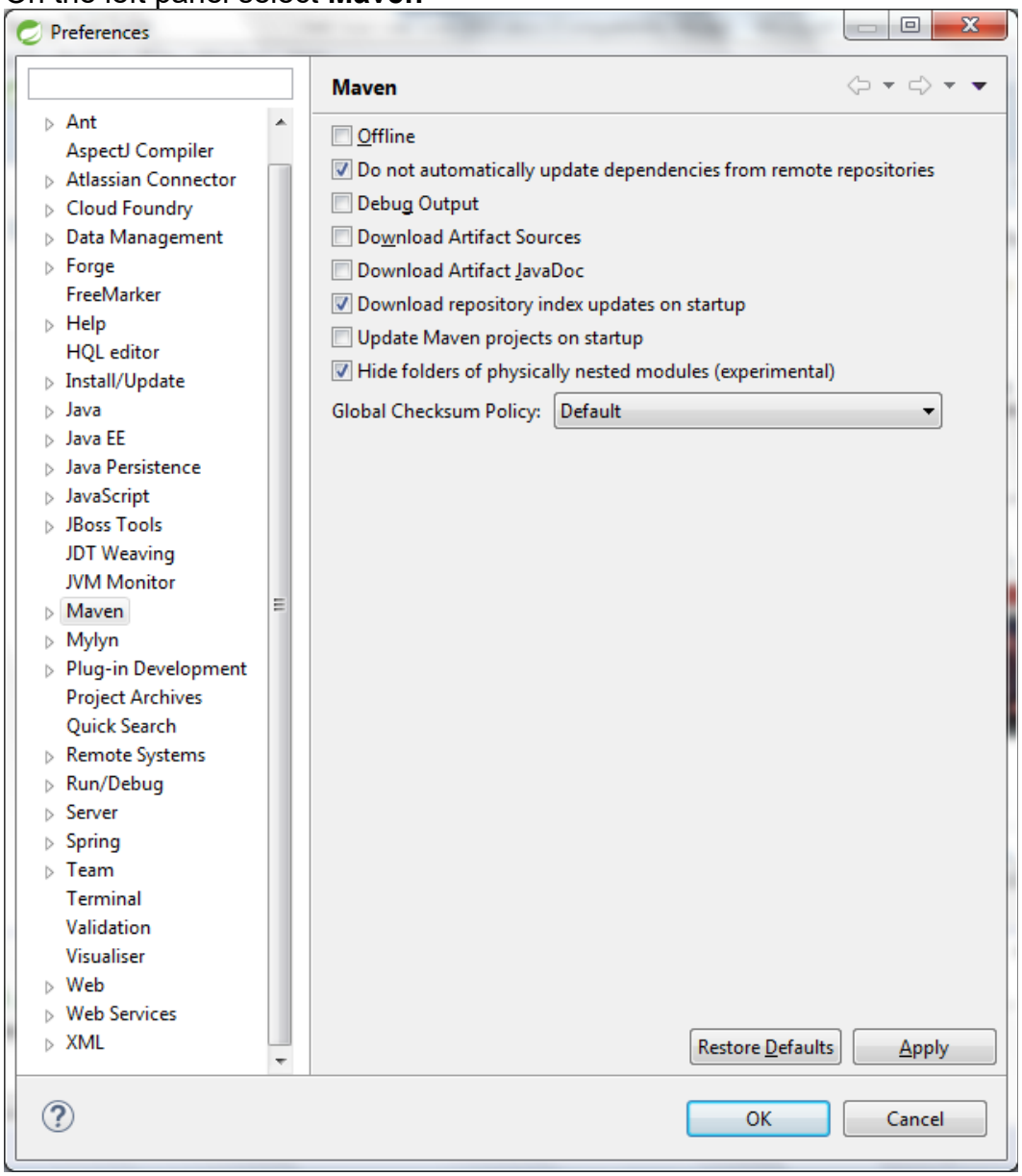


A workspace allows you to group similar projects

f. Go to Help, Check for updates and install them if there are any available
   - Restart application when requested

g. Go to Windows, Preferences, Java, Installed JREs
h. **Remove the installed JRE** that is set to default (if it is not already set to JDK path)
i. Press **Add New…**
j. Select directory for the **installed JDK** (for example C:\Program Files\Java\jdk1.8.0_74)
k. Press **Finish**
l. Check (on the left boxes in the list**)** **the new JDK** as the default JRE

m. On the left panel select **Maven**



n. Check the "**Download repository index updates**" option (this is optional)

NOTE: this option will populate the information required by maven to help you identify dependencies

o. Press **Ok** to accept the preference changes
p. Restart STS and wait for the index to be populated (Network is required) – Be patient

2. Problem 1 – Web Server

1. Import the BareBonesHTTPD project
2. Add the following functionality to the program
    a) Define a document-root-folder where the application will be able to read files
    b) Use the HTTP request URI to match files and folders under the document-root-folder.
    NOTE: The full path cannot be outside of the document-root-folder.
    c) The HTTP response payload should return the requested file. Map error codes with corresponding HTTP response status codes.

3. Problem 2 – Web Container

1. Use the project from Problem 1 or import the BareBonesHTTPD project
2. Add the following functionality to the program
    a) Define a Java class that will produce an HTML response for each of the following URI
        /welcome.web
        /contacts.web

    Generate an HTML dynamically, for each page, add a counter for example. The classes must return different HTML content.

```
<!DOCTYPE html>
<html>
<head>
<title>Class Name</title>
</head>
<body>

<p>Some static text</p>
<p>Some generated text</p>

</body>
</html>
```

    b) Change the BareBonesHTTPD to match the URI with the corresponding class to send the respond.
    c) In case the HTTP request URI is different, the program should continue with the expected response (if it is based on Problem 1, then it should return the file)

If that was too easy!

    d) Adjust the program to read the names of the classes and their URI from a configuration file (you can define the format).
    e) Use reflection to instantiate objects for the classes
    f) Change the BaereBonesHTTPD to use the configuration file to execute the class method to process the HTTP request send to the related URI.