**SelCukes Framework Setup Guide**

Contents

[1) Java setup: 3](#_Toc387770044)

[a) Download and Install Java: 3](#_Toc387770045)

[b) Path environment variable setup 4](#_Toc387770046)

[2) Maven Setup 5](#_Toc387770047)

[a) Download Maven 5](#_Toc387770048)

[b) Path environment variable setup 6](#_Toc387770049)

[3) Eclipse Setup 8](#_Toc387770050)

[a) Eclipse Download 8](#_Toc387770051)

[4) Create Simple Maven Project 13](#_Toc387770052)

[5) Add Dependencies in POM.xml 16](#_Toc387770053)

[6) Create Object Identification configuration file. 21](#_Toc387770054)

[7) Create Step Implementation File (Selenium) 22](#_Toc387770055)

[8) Create a Feature File (Cucumber) 25](#_Toc387770056)

[9) Copy RunCukesTest.java file 27](#_Toc387770057)

[10) Run the project. 27](#_Toc387770058)

[a) Running from eclipse: 27](#_Toc387770059)

[b) Running from command prompt: 28](#_Toc387770060)

[11) Running feature file: 28](#_Toc387770061)

**SelCuke Framework Setup Procedure:**

**Pre-Requisites:**

# Java setup:

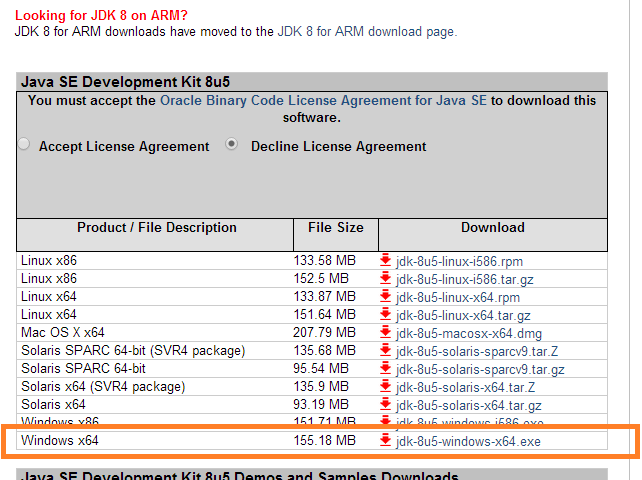
## Download and Install Java:

**Memory Requirements**

On Windows 64-bit operating systems, in 32- or 64-bit mode, the Java runtime requires a minimum of 128MB of memory.

Java SE is freely available from the link  <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>. So you download a version based on your operating system.

**For ex:** For your machine has Windows operating system and is 64 bit machine, then download the exe which is highlighted in the below screenshot.

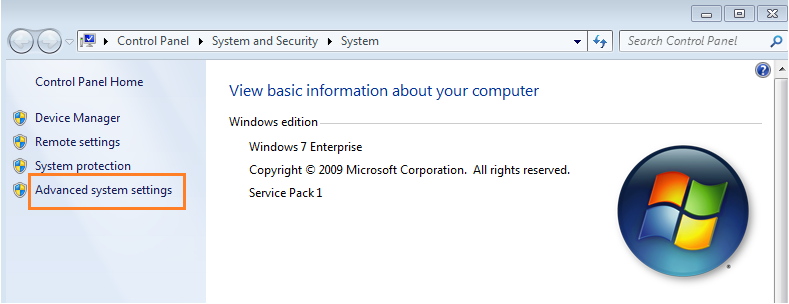


Follow the instructions to download java and run the **.exe** to install Java on your machine. Once you installed Java on your machine, you would need to set environment variables to point to correct installation directories

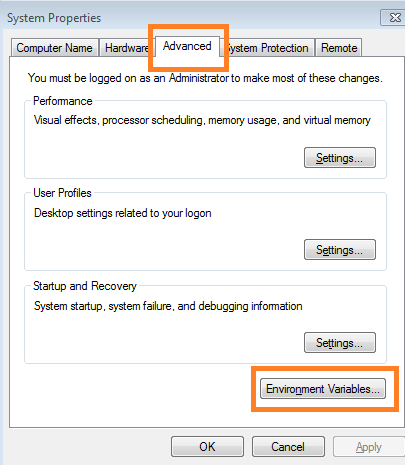
## Path environment variable setup

Assuming you have installed Java in c:\Program Files\java\jdk directory:

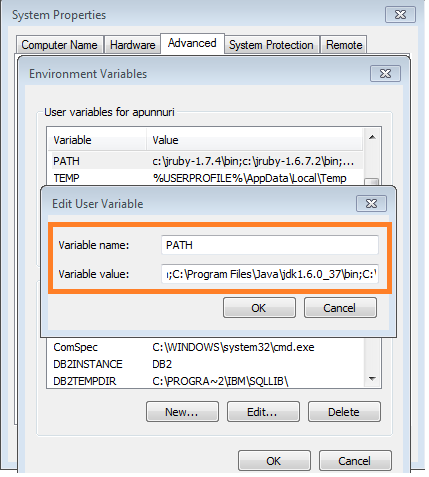
* Right-click on 'My Computer' and select 'Properties'. Select “Advanced System Settings”.



* Click on the 'Environment variables' button under the 'Advanced' tab.



* Now, alter the 'Path' variable so that it also contains the path to the Java executable. Example, if the path is currently set to 'C:\WINDOWS\SYSTEM32', then change your path to read 'C:\WINDOWS\SYSTEM32;c:\Program Files\java\jdk\bin'.



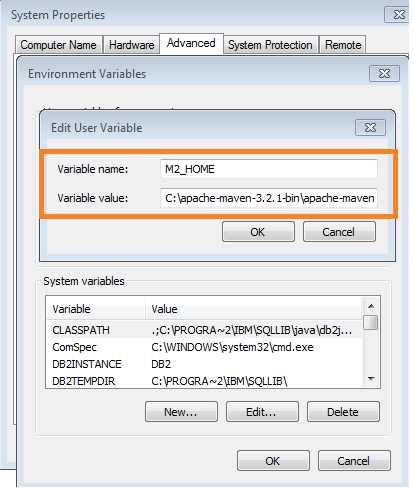
# Maven Setup

## Download Maven

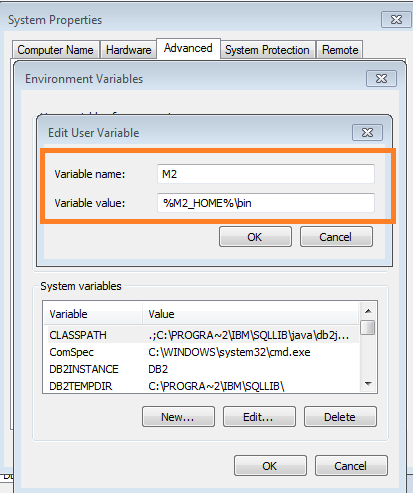
* Download the current stable version of maven binary zip file from this site <http://maven.apache.org/download.cgi>
* Unzip the distribution archive, i.e. apache-maven-3.2.1-bin.zip to the directory you wish to install Maven 3.2.1. These instructions assume you chose C:\Program Files\ApacheSoftwareFoundation. The subdirectory apache-maven-3.2.1 will be created from the archive.

## Path environment variable setup

* Right-click on 'My Computer' and select 'Properties'. Select “Advanced System Settings”. Click on the 'Environment variables' button under the 'Advanced' tab, then add the M2\_HOME variable in the user variables with the value C:\Program\Files\ApacheSoftwareFoundation\apache-maven-3.2.1. Be sure to omit any quotation marks around the path even if it contains spaces. Note: For Maven 2.0.9, also be sure that the M2\_HOME doesn't have a '\' as last character.



* In the same dialog, add the M2 environment variable in the user variables with the value %M2\_HOME%\bin



* Setting the proxy and local repository (for jar files) in settings.xml
  + 1. Go to **maven installation directory** “C:\Program\Files\ApacheSoftwareFoundation\apache-maven-3.2.1\conf”. Open the settings.xml file.
    2. Under the <proxy> tag, provide the respective proxy settings. This is required by maven to connect to internet and download the dependency jar files.

For ex:

<proxy>

<protocol>http</protocol>

<username>USERNAME</username>

<password>PASSWORD</password>

<host>internet.xxxxx.com</host>

<port>80</port>

<nonProxyHosts>127.0.0.1</nonProxyHosts>

</proxy>

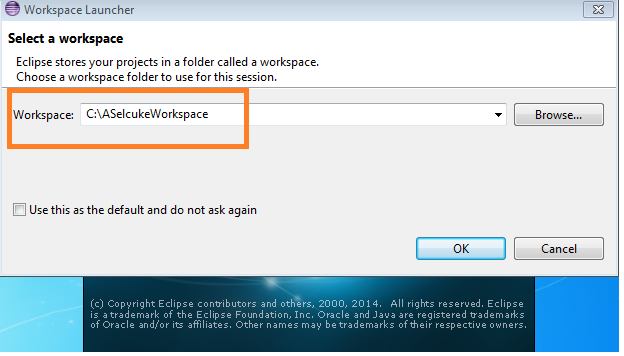
* + 1. Specify the location of local repository in localRepository tag. Maven downloads the dependency jar files and keeps it in the local repository.

For ex: <localRepository>C:\projects\maven\_localrepo</localRepository>

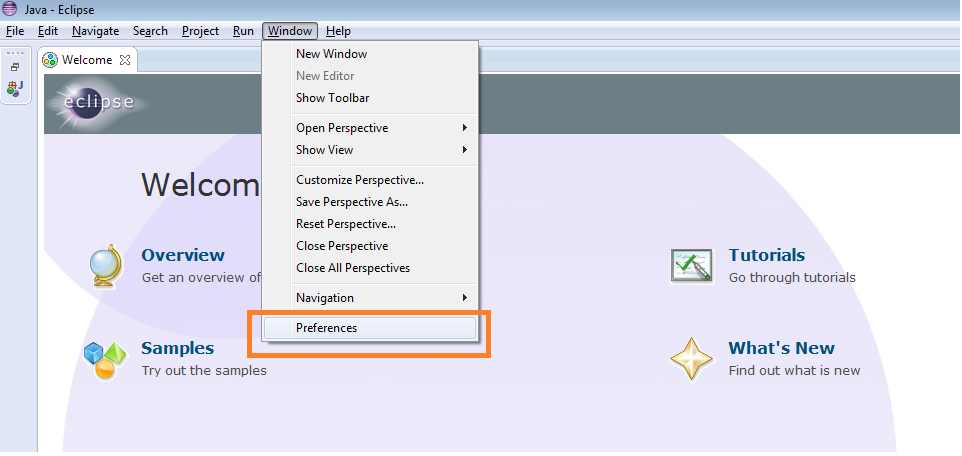
# Eclipse Setup

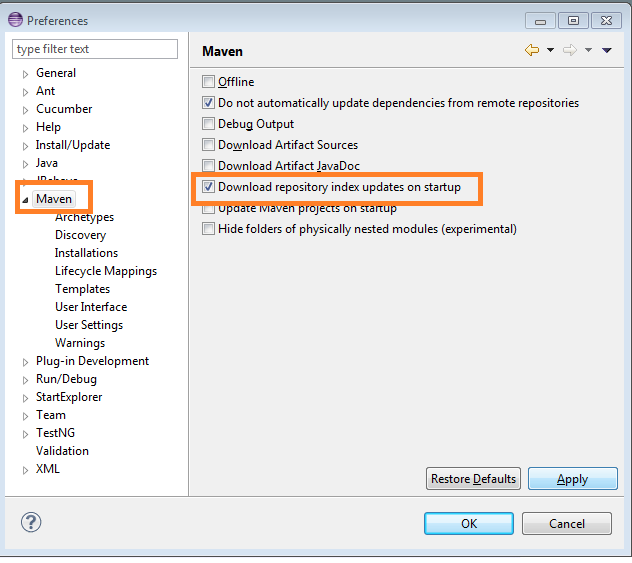
## Eclipse Download

* Download the Eclipse IDE for Java EE (Kepler) from this site <https://www.eclipse.org/downloads/>
* Change following settings in eclipse.
* Open eclipse. Create a workspace in C: drive. (Creating workspace in shared drive is not recommended).

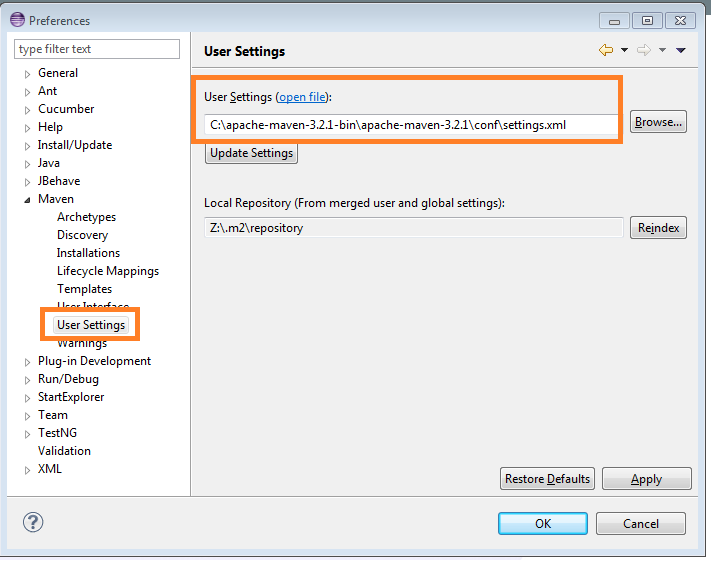


* Click on Window >> Preferences >> Maven. Check the “Download repository index updates on startup.





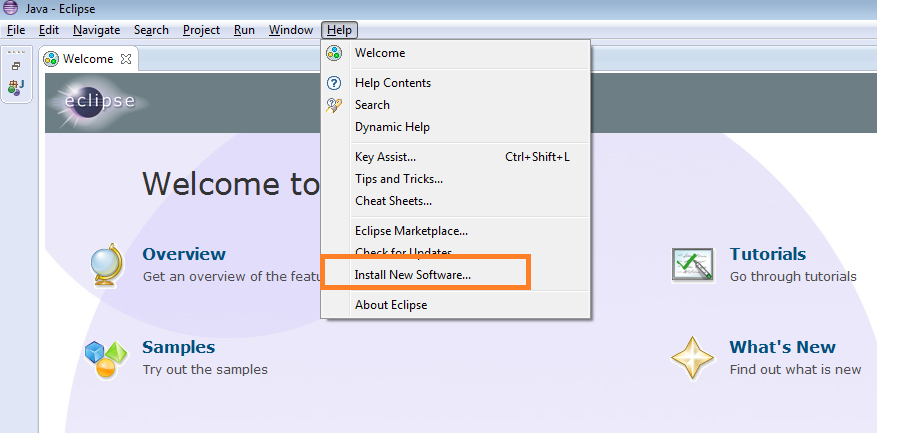
* Click on Window >> Preferences >> Maven >> User Settings. Click on Browse button. Provide the location of the settings.xml file which will be present in the Maven installation directory (in this case it is “C:\Program\Files\ApacheSoftwareFoundation\apache-maven-3.2.1\conf”).

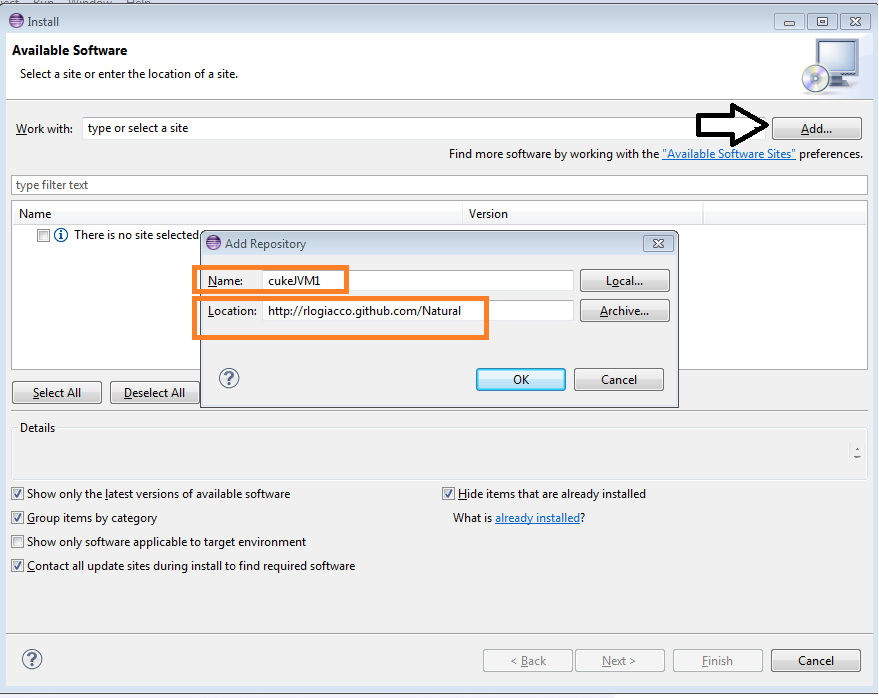


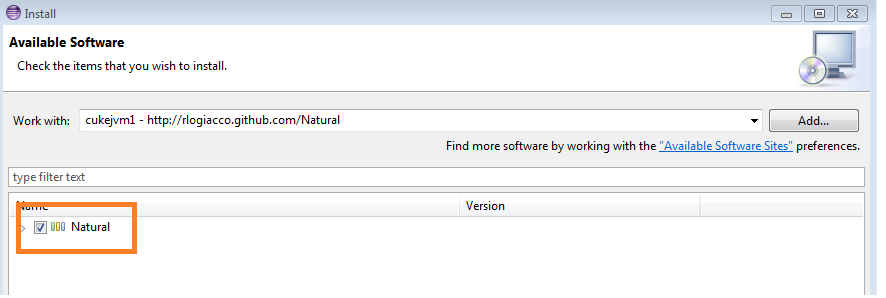
* Add following eclipse plugins to support cucumber JVM.
  + - Go to Help >> Install New software. Click Add button. Specify some name (ex: cukeJVM1) and Enter “<http://rlogiacco.github.com/Natural>” in Location and Click OK button.

Check the “Natural” checkbox and click on Next button. Select “I accept the terms of license agreement” and click on Finish button. Click OK in the warning message that appears while adding the plugin.

This plugin highlights cucumber syntax and run features.



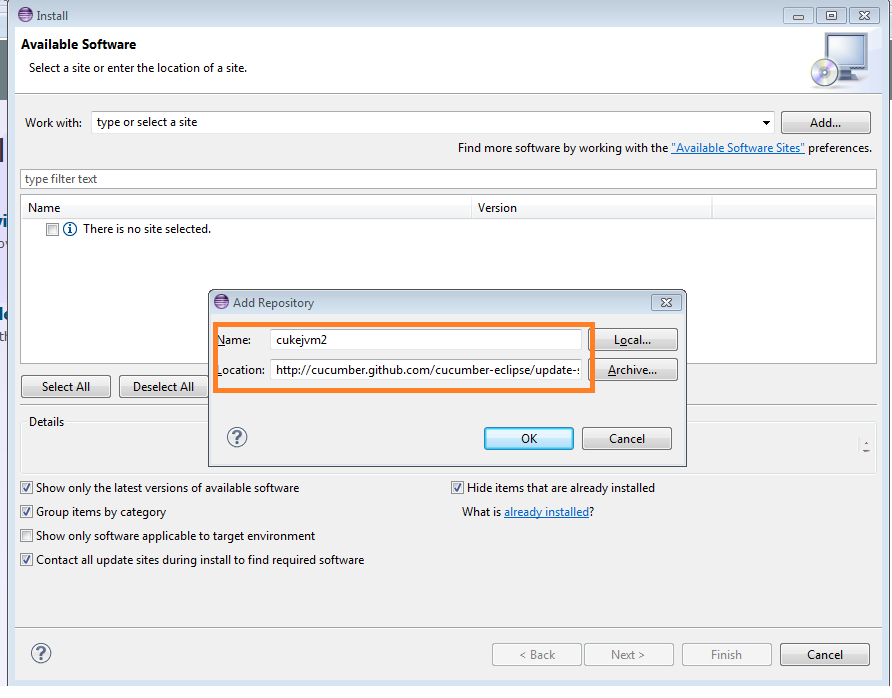


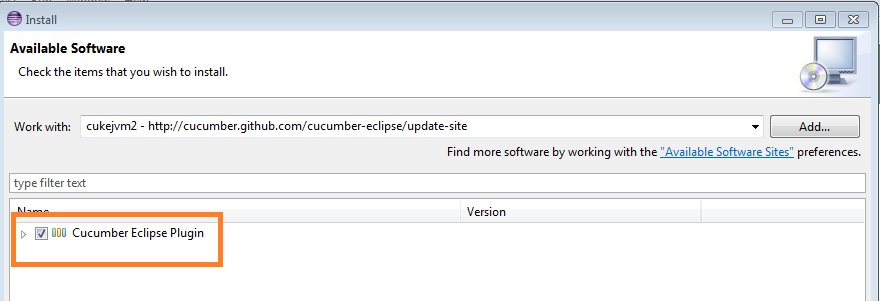


* + - Go to Help >> Install New Software. Click Add button. Specify some name (ex: cukeJVM2) and Enter “<http://cucumber.github.com/cucumber-eclipse/update-site>” in Location and Click OK button.

Check the “Cucumber Eclipse Plugin” checkbox and click on Next button. Select “I accept the terms of license agreement” and click on Finish button. Click OK in the warning message that appears while adding the plugin.

This plugin link Cucumber steps and Selenium Step Implementations.

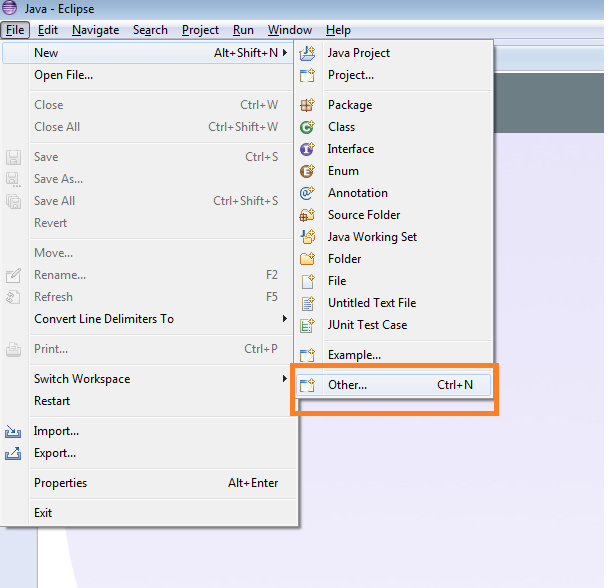




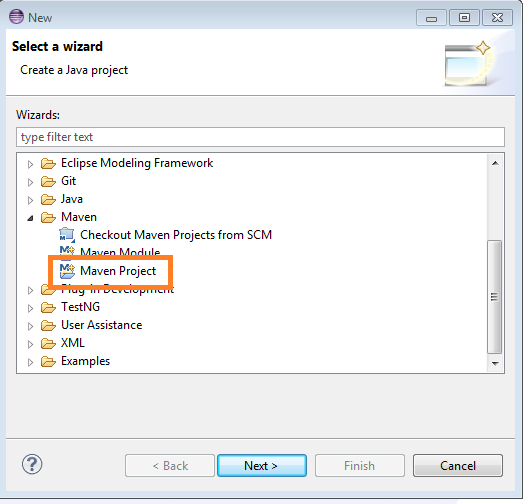
**SelCukes Framework Setup**

# Create Simple Maven Project

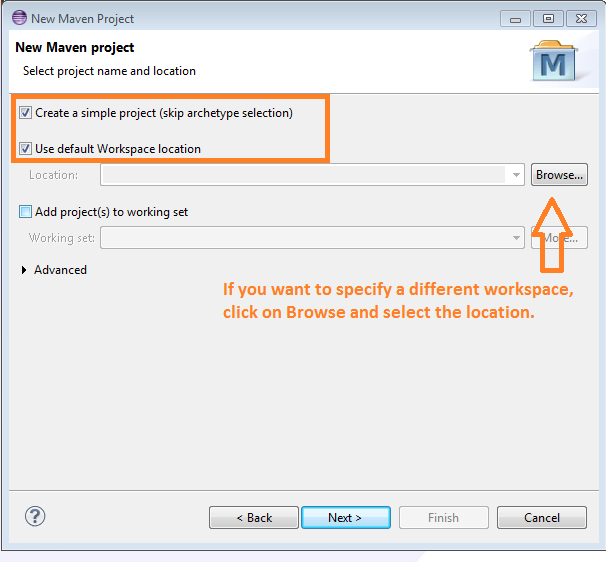
* Follow these steps to create a simple maven project:
* Click on File menu >> Click on New >> Click on Other.



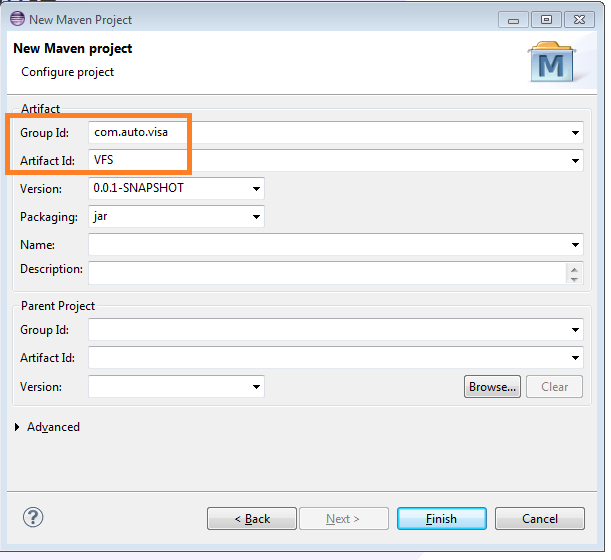
* Click on Maven project >> Click on Next button >> Select “Create a simple project” checkbox.



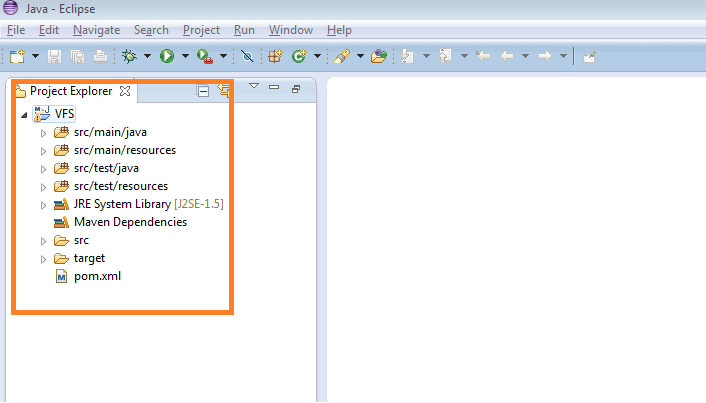
* If you want to use the default workspace location, don’t specify any workspace location, just click on Next. Otherwise, click on Browse and select the appropriate workspace location and Click on Next button.



* Specify the Group Id something like “com.auto.xxxxx” which is project folder structure for maven.
* Specify the Artifact Id. This will be your project name.
* Click on Finish button.



A simple maven project is created.



# Add Dependencies in POM.xml

POM.xml is the file which specifies the dependencies/jars that we are using in the project. There are five dependencies to be added in POM.xml for this project. They are:

* Cucumber-java.jar

This jar supports cucumber to be used with java.

* Cucumber-junit.jar

This jar supports cucumber to be used with junit.

* Selenium-java.jar

This jar provides all the selenium APIs used for automation.

* Junit.jar

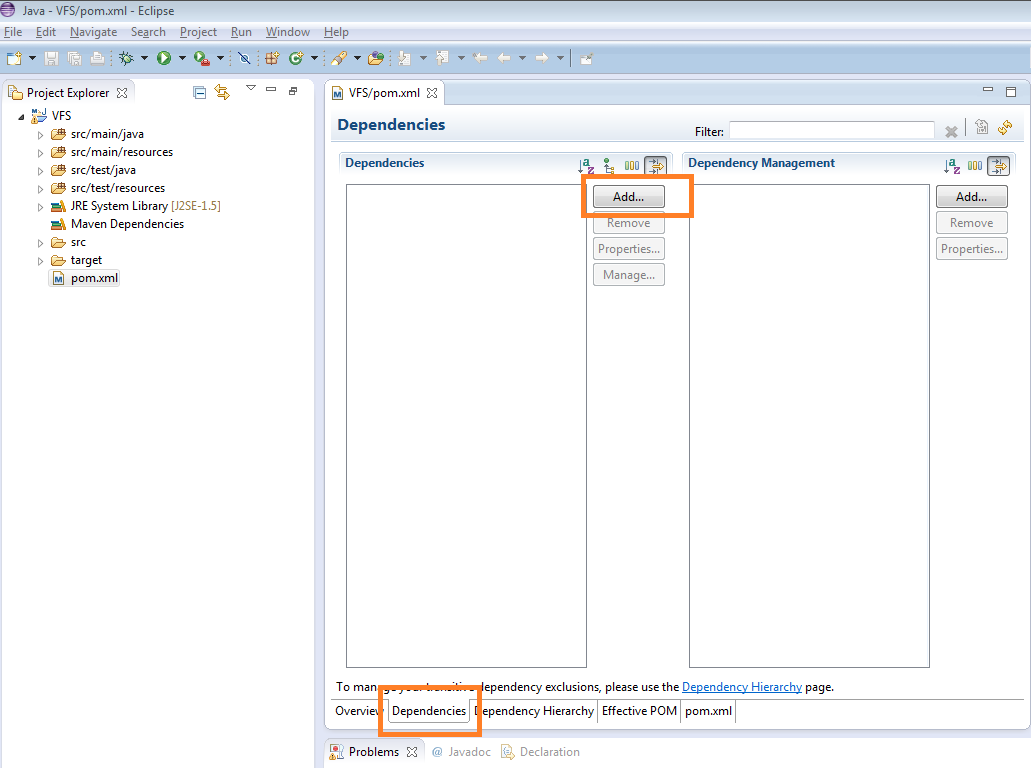
This jar provides all junit supported features.

* Commons-configuration.jar

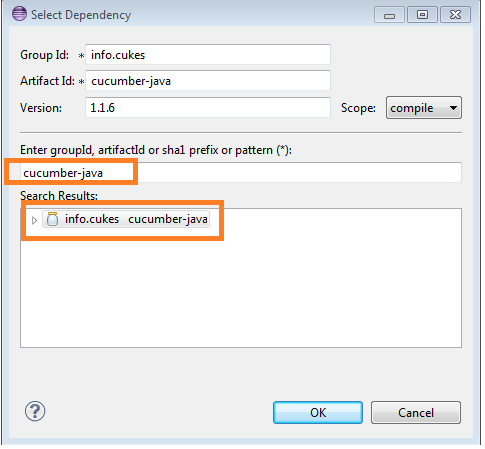
This jar is used to read the xml, properties files.

Follow these steps to add the above dependencies to the project:

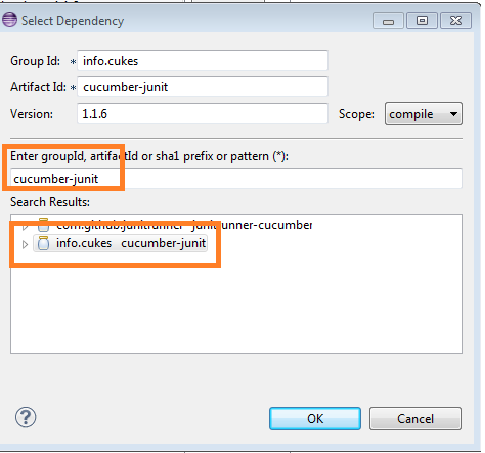
1. Open the pom.xml file in eclipse. Click on Dependencies tab. Click on Add button.



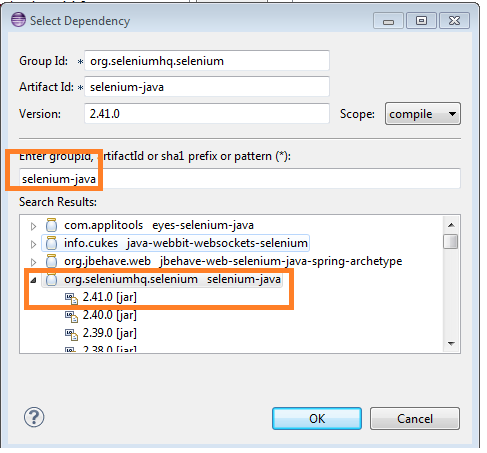
1. In the “Enter group id, artifact id” textbox, enter “cucumber-java”. The jar will be displayed in the Search result. Expand it and select the latest stable version of jar.



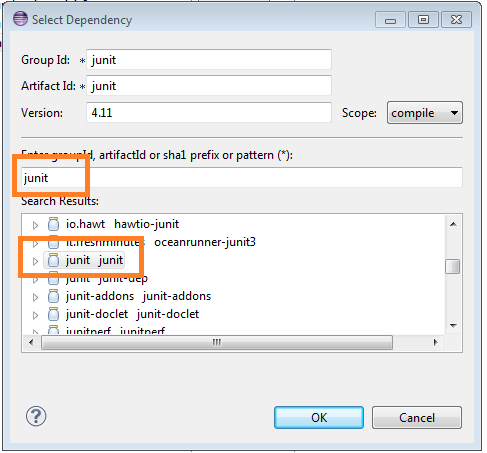
1. In the “Enter group id, artifact id” textbox, enter “cucumber-junit”. The jar will be displayed in the Search result. Expand “info.cukes cucumber-java” and select the latest stable version of jar.



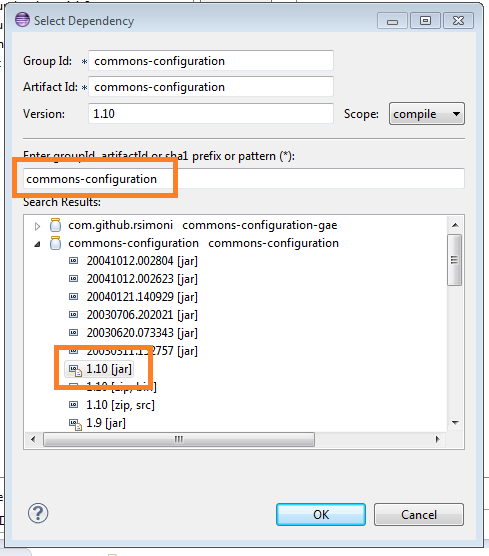
1. In the “Enter group id, artifact id” textbox, enter “selenium-java”. The jar will be displayed in the Search result. Expand “org.seleniumhq.selenium selenium-java” and select the latest stable version of jar.



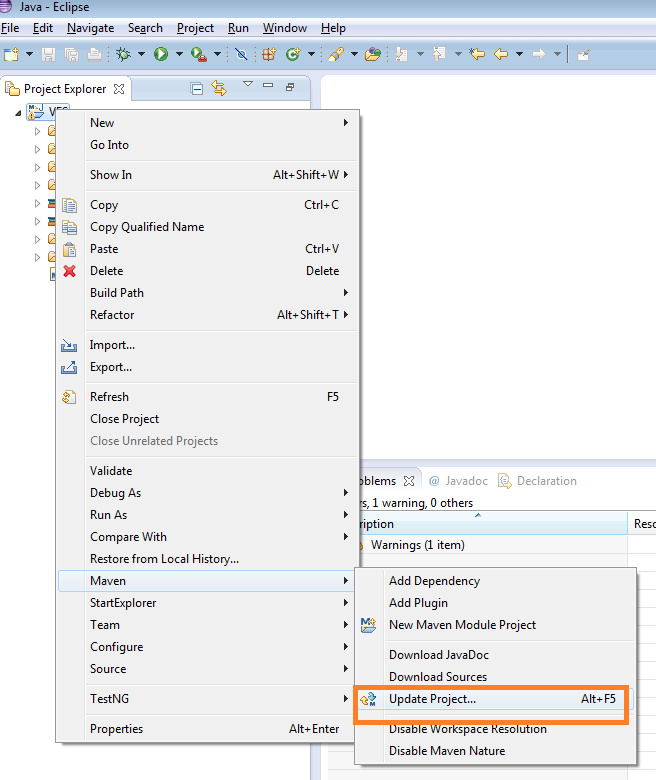
1. In the “Enter group id, artifact id” textbox, enter “junit”. The jar will be displayed in the Search result. Expand “junit junit” and select the latest stable version of jar.



1. In the “Enter group id, artifact id” textbox, enter “commons-configuration”. The jar will be displayed in the Search result. Expand “commons-configuration commons-configuration” and select the latest stable version of jar.



Click on Save button. If any errors exist in project now, right click on Project >> Select Maven >> Click on Update Project.



# Create Object Identification configuration file.

The objects have to be stored in an xml file. The objects then read using the commons-configuration API. Please refer the example below, which specifies how the objects are stored and how they are read in the step implementation.

**Storing the objects in xml file.**

Follow these steps to create objects-config.xml file and store the objects in this file.

* 1. Under the src\main\resources folder, create file “object-config.xml”.
  2. The structure of xml file is left to user discretion. The structure in this example is as below.

**Note: The highlighted content below is only for reference. Do not use the same to test any test case.**

**<configurations>**

**<registration>**

**<fullname locator="id">**coursera-signup-fullname**</fullname>**

**<confirmemail locator="id">**coursera-signup-email-confirm

**</confirmemail>**

**<email locator="id">**coursera-signup-email**</email>**

**<password locator="id">**coursera-signup-password**</password>**

**<terms locator="id">**coursera-signup-agree**</terms>**

**</registration>**

**<WelcomePage>**

**<fullname locator="id">**coursera-fullname**</fullname>**

**</WelcomePage>**

**</configurations>**

In the above example**, <registration>** is page name. Under registration page, there are fields “fullname”,”confirmemail”,”email”,”password”,”terms”.

**<fullname locator="id">coursera-signup-fullname</fullname>** specifies the fullname locator type is **ID** and the locator value is **“coursera-signup-fullname”.**

Same pattern is followed for other fields.

**The objects of a different page are grouped separately under appropriate tags**.

For ex: The objects under the WelcomePage are grouped under the <WelcomePage> </WelcomePage> tags.

**Reading the data from object-config.xml file:**

Follow these steps to read the data from object-config.xml file:

The object identifier is passed from the test step in the feature file as below.

**Then I enter "Test Account2" into "registration.fullname" text field**

Note: Ignore the parent node i.e. configurations and specify the remaining proper xpath of the object identifier.

Ex: **registration.fullname, registration.confirmemail**

* 1. The config object is created in the Step Implementation file under @Before annotation (in this example).

**config = new XMLConfiguration("object-config.xml");**

* 1. In order to retrieve the webelement, use **getElement(config.getString(identifier));**

which gives the web element which can be used as per our requirement.

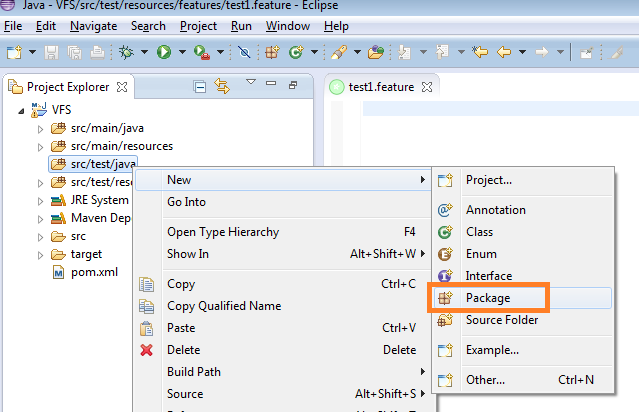
**A sample file is provided below to test a scenario for CoursEra registration page. You can use the same file to run a simple test case and understand the framework. Please refer the attached object-config.xml file.**

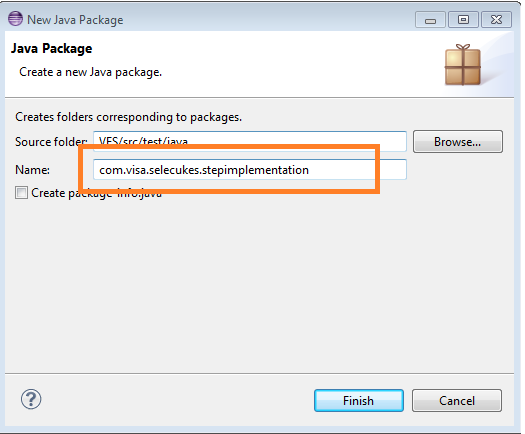
****

# Create Step Implementation File (Selenium)

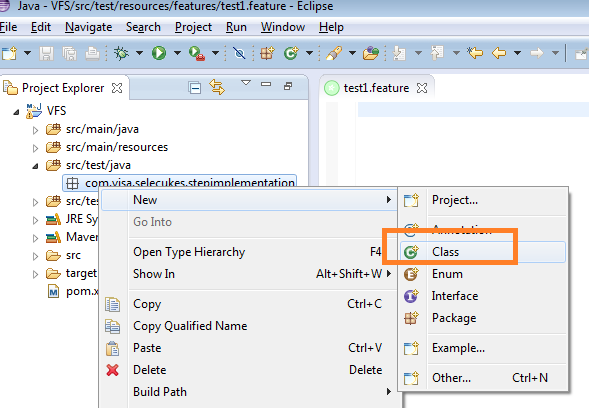
The backend steps which contain the actual automation code should be written in this file.

1. Under the src/test/java, create a new package “com.xxxxx.selecukes.stepimplementation”.

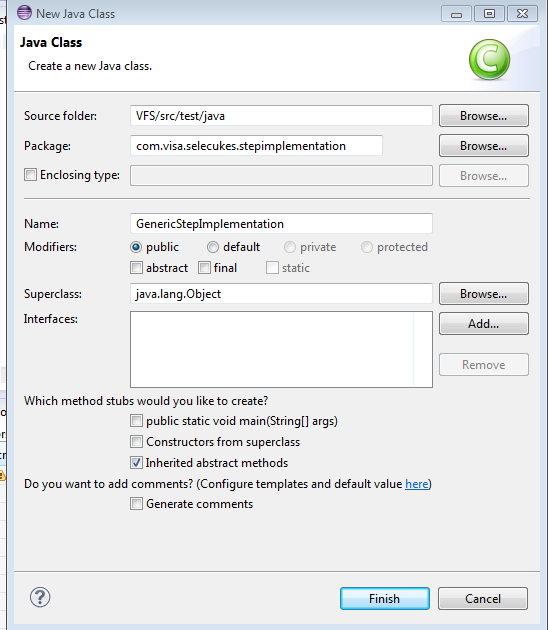




1. Right click on stepimplementation package and select New >> Class.



1. Specify the appropriate file name



1. Click on Finish.
2. The step implementation java file is created.
3. Under @Before annotation, write the code for creating driver instance and config object.
4. Under the @After annotation, write the code for tear down process.
5. All the step implementations start with either of the following tags @Given, @When, @Then.

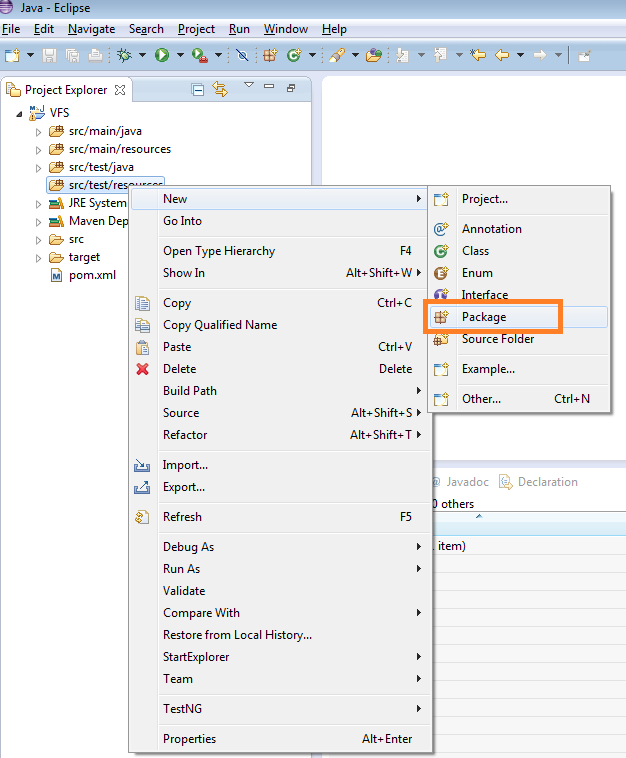
**A sample file is provided below to test a scenario for CoursEra registration page. You can use the same file to run a simple test case and understand the framework.**  Please use the attached generic implementation file for reference.

****

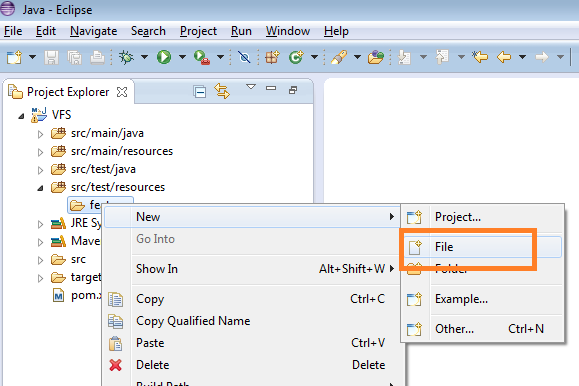
# Create a Feature File (Cucumber)

The feature file should contain the test cases/ test scenario outline. This file contains the test cases in the general readable format (gherkin).Follow these steps to create a feature file. The folder structure in the project is user discretion.

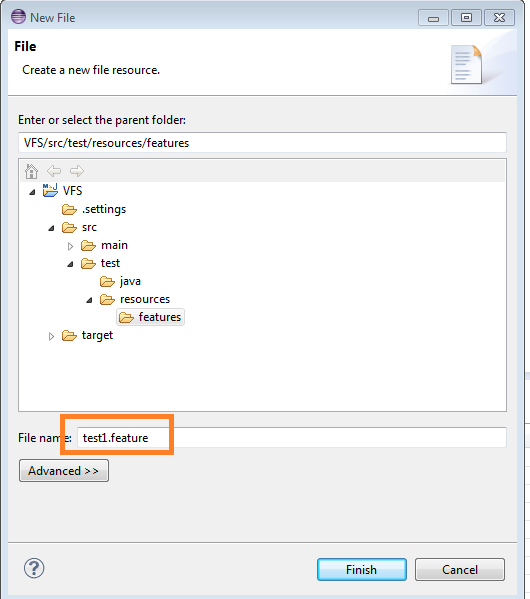
* + 1. Under the src/test/resources, create a new package “features”.



* + 1. Right click on features folder and select New >> File.



* + 1. Specify the appropriate test file name with .feature extension.



* + 1. Click on Finish.

The feature file is created.

The feature file contains the test case in human readable format along with certain annotations to segregate the test cases for execution. A simple test case is identified as a Scenario. A test case which requires parameterization is identified as Scenario Outline. Each test case follows the format Given, When, Then.

**A sample file is provided below to test a scenario for CoursEra registration page. You can use the same file to run a simple test case and understand the framework.**  Please refer the attached feature file for reference.

****

# Copy RunCukesTest.java file

RunCukesTest.java file specifies the feature files which have to be run and also the report type to be generated.

Save the below RunCukesTest.java file in the folder where the StepImplementation.java file is created. In this case, it is “<WorkspaceProjectLocation>\src\test\java\com\xxxxx\selcukesfw”.

****

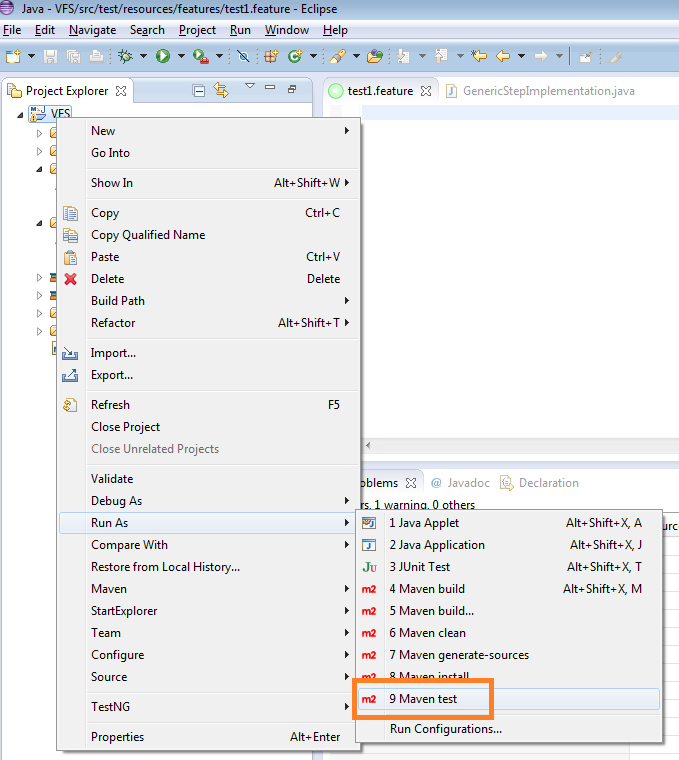
# Run the project.

There are 2 ways of running the project.

## Running from eclipse:

* + - * Right click on Project folder >> Select Run As >> Click on Maven test.

The project runs successfully and all the appropriate test cases will be executed.



## Running from command prompt:

* + - * Open command prompt. Go to Start menu >> Select Run >> Enter cmd. Press Enter.
      * Navigate to the project location. Enter cd <Project location>. Press Enter.
      * Enter mvn and press Enter.

The project runs successfully and all the appropriate test cases will be executed.

# Running feature file:

In order to run a feature file from eclipse, follow these steps:

* Right click feature file.
* Select Run As.
* Click on “Run Feature”.

