**What is a Selenium framework?**

Selenium framework is a code structure for making code maintenance simpler, and code readability better. A framework involves breaking the entire code into smaller pieces of code, which test a particular functionality.

### ****Benefits of Selenium framework****

* Increased code reuse
* Improved code readability
* Higher portability
* Reduced script maintenance

## What is Maven?

Maven is a build tool. It is a software project management tool that provides a new concept of project object model (POM). Maven allows the developer to automate the process of the creation of the initial folder structure, performing the compilation and testing and the packaging and deployment of the final product. It cuts down the good number of steps in the build process and it makes it one-step process to do a build.

A very significant aspect of Maven is the use of repositories to manage jar files.

**Why we should use maven?**

1. We need maven to get all our dependencies automatically, which also allows users to reuse same jars across multiple projects.

2. Each and every engineer in a project must use the same jar dependencies due to the centralized POM.

3. It provides the complete structure with naming conventions which is easy to locate and execute tests.

4. We can automate the complete build procedure.

### Maven Local Repository

This is the place where Maven stores all the project jars files or libraries or dependencies. By default the folder name is ‘***.m2***‘ and by default the location in windows 7 is ‘***Libraries\Documents\.m2***‘.

### Maven Central Repository

Maven central repository is the default location ‘***http://mvnrepository.com/***‘ for Maven to download all the project dependency libraries. For any library required in the project, Maven first looks into the .m2 folder of Local Repository, if it does not find the required library then it looks in Central Repository and downloads the library into the local repository.

### Dependency Keyword

Dependencies are the libraries, which are required by the project. For example Log4j jars, Apache Poi jars, Selenium Jars, etc. Dependencies are mentioned in the Maven pom.xml like this:

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>2.43.1</version>

</dependency>

### Maven POM

POM is a Project Object Model XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects. Some of the configurations that can be specified in the POM are the project dependencies, the plugins or goals that can be executed, the build profiles, and so on.

### Surefire Plugin

The Surefire Plugin is used during the test phase of the build lifecycle to execute the unit tests of an application. It generates reports in 2 different file formats like plain text file, XML files, and HTML files as well. Even if you are using TestNG or Junits framework for reporting, this plugin is a must to use, as it helps Maven to identify tests.

How to Install Maven In Eclipse IDE

<https://www.softwaretestingmaterial.com/install-maven-eclipse-ide/>

<https://www.toolsqa.com/java/maven/how-to-install-maven-eclipse-ide/>

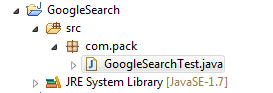
**Basic example program using selenium WebDriver Java**

**Step 1:** First create a Java project. File -> New -> Java Project. And name the Java Project as 'GoogleSearch'.

**Step 2:** Create a Package with name called 'com.pack'. Right Click on the Src folder -> New -> Package.

**Step 3:** Create a class with name 'GoogleSearchTest'. Right Click on the Package folder -> New -> Class

Now the folder structure should look like below.



**Step 4:**

We need to configure the build path using Right Click On Project folder -> Build Path -> Configure Build path. Add selenium jars.

**Step 5:**

Basic script to open Google website and print the title:

**package** com.pack;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** GoogleSearchTest {

**public** **static** **void** main(String[] args) {

String exePath1 = "G:\\Selenium RS 2019\\geckodriver-v0.26.0-win32\\geckodriver.exe";

System.*setProperty*("webdriver.gecko.driver", exePath1);

WebDriver driver = **new** FirefoxDriver();

driver.navigate().to("http://google.com");

String appTitle = driver.getTitle();

System.***out***.println("Application title is :: "+appTitle);

driver.quit();

}

}

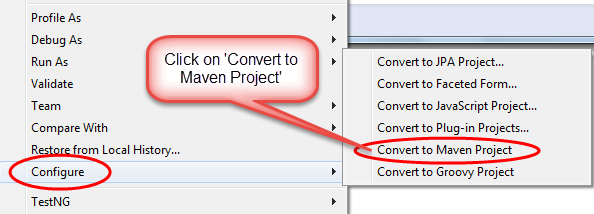
**Output:**

Application title is:: Google

**Basic example program using selenium WebDriver Java in Maven**

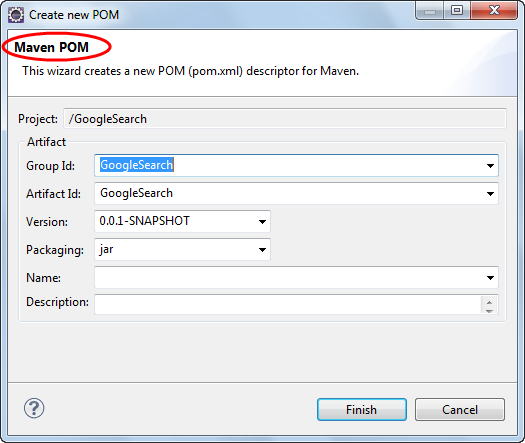
We can convert java project to maven project using below steps: -

**Step 1:** Right Click on Project Folder -> Configure -> Select 'Convert to Maven Project'



**Step 2:**

Displays a screen as below which is responsible to create the POM.xml file.



**Step 3:**

Now we need to add the Maven Dependency for selenium under dependencies tag.

**Do Google search with selenium java maven dependency.** Below is the link

<https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java>

**Step 4:**

Place the below code inside dependencies tag.so then we don't need to change anything in pom.xml file.

<dependencies>

<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>3.141.59</version>

</dependency>

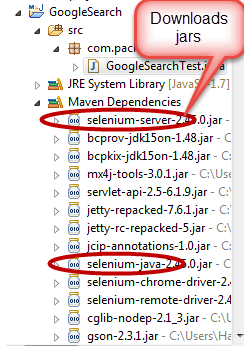
</dependencies>

**Step 5:**

Now remove the jar file which we have uploaded manually in the build path. Now maven will take care of the jar files. It will download all the required jars based on the dependency provided in pom.xml file, once we save the pom.xml file.

**Step 6:**

Refresh the project and see the jars files that are downloaded under maven dependencies.



Now you can execute the same java file and the see the same output as earlier.

**Create Selenium WebDriver Maven Project Using TestNG and Eclipse IDE**

**Note:** Test cases reside under the **src > test > java > PackageName** will only be considered as a test by Maven, rest will be ignored if you put your test cases in some other folder.

Step 1: First create a maven project and name it as 'FirstDemo'.

Step 2: Create a package com.google.tests and create a class 'GoogleHomePageTest.java'

Step 2: Create a class 'GoogleHomePageTest.java'

Step 3: Add Tests in 'GoogleHomePageTest.java' class.

Step 4: Add TestNg and Selenium Dependencies to maven pom.xml file.

Step 5: Now add maven Surefire Plug-in and maven-compiler-plugin to pom.xml

Step 6: Execute tests using 'maven test' from pom.xml

**package** com.google.tests;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.testng.Assert;

**import** org.testng.annotations.AfterClass;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**public** **class** GoogleHomePageTest {

**private** WebDriver driver;

String appURL = "http://google.com";

@BeforeClass

**public** **void** testSetUp() {

String exePath1 = "G:\\Selenium RS 2019\\geckodriver-v0.26.0-win32\\geckodriver.exe";

System.*setProperty*("webdriver.gecko.driver", exePath1);

driver = **new** FirefoxDriver();

}

@Test

**public** **void** verifyGooglePageTittle() {

driver.navigate().to(appURL);

String getTitle = driver.getTitle();

Assert.*assertEquals*(getTitle, "selenium");

}

@AfterClass

**public** **void** tearDown() {

driver.quit();

}

}

**How to Create Selenium Maven Project in Eclipse IDE:**

<https://www.seleniumeasy.com/maven-tutorials/simple-example-program-using-webdriver-maven-java>

<https://www.seleniumeasy.com/maven-tutorials/how-to-execute-selenium-webdriver-testng-xml-using-maven>

<https://www.softwaretestingmaterial.com/create-selenium-maven-project/>

<https://www.techbeamers.com/create-selenium-webdriver-maven-project/>

<https://examples.javacodegeeks.com/enterprise-java/testng/testng-maven-project-example/>

<https://howtodoinjava.com/testng/how-to-execute-testng-tests-with-maven-build/>

How to Install Maven on Windows

<https://www.toolsqa.com/java/maven/how-to-install-maven-on-windows/>

<https://maven.apache.org/download.cgi>

<https://www.toolsqa.com/java/maven/create-new-maven-project-eclipse/>

**Data driven framework:**

<https://chercher.tech/java/maven-integration-selenium-webdriver>

<https://www.edureka.co/blog/selenium-framework-data-keyword-hybrid-frameworks>

<https://www.edureka.co/blog/create-selenium-maven-project/>

<https://www.toolsqa.com/selenium-webdriver/data-driven-testing-excel-poi/>

<http://learn-automation.com/data-driven-framework-in-selenium-webdriver/>

<https://www.softwaretestinghelp.com/data-driven-framework-selenium-apache-poi/>

<https://www.ecanarys.com/Bogs/ArticleID/216/Data-driven-testing-in-selenium-webdriver-using-excel>

<https://www.swtestacademy.com/selenium-webdriver-tutorial-java-testng/>

**Date picker:**

<https://www.swtestacademy.com/datepicker-using-selenium/>

**Maven commands to clean, compile and install from command prompt**

<https://www.softwaretestinghelp.com/maven-project-setup-for-selenium-selenium-tutorial-24/>

<https://www.toolsqa.com/java/maven/configure-selenium-continuous-integration-maven/>