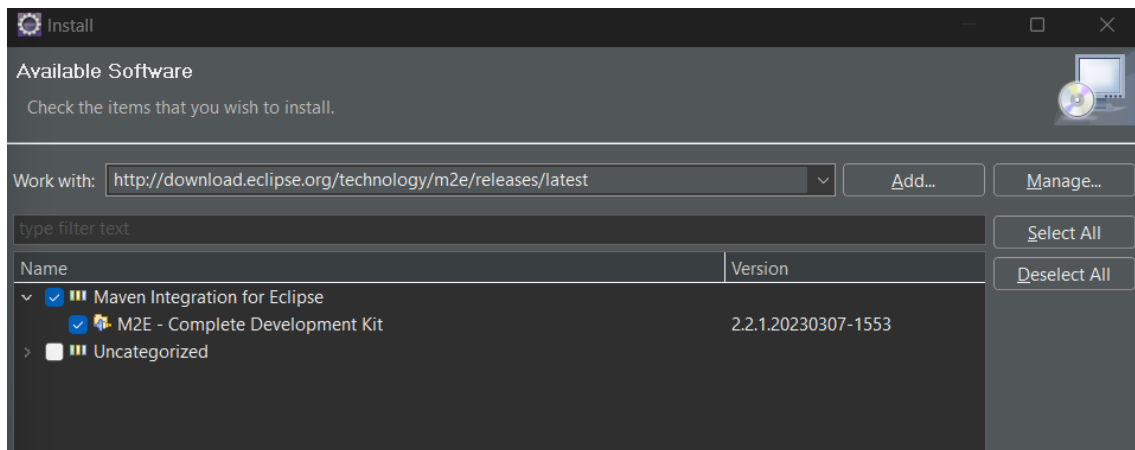


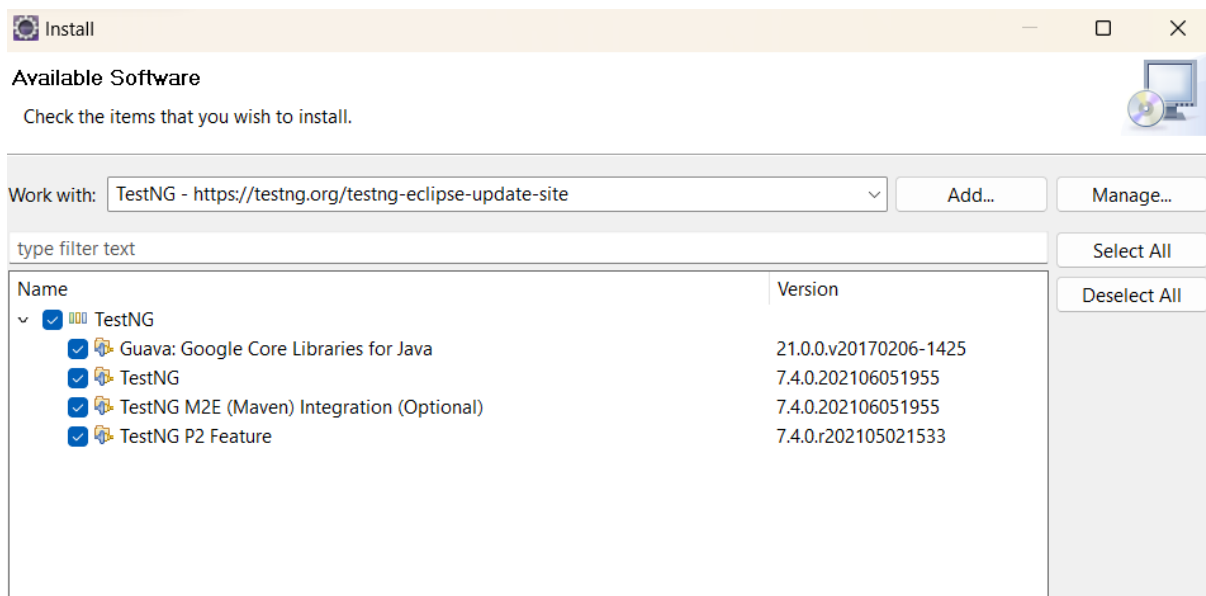
1. Download and Install Eclipse IDE for Java Developers.



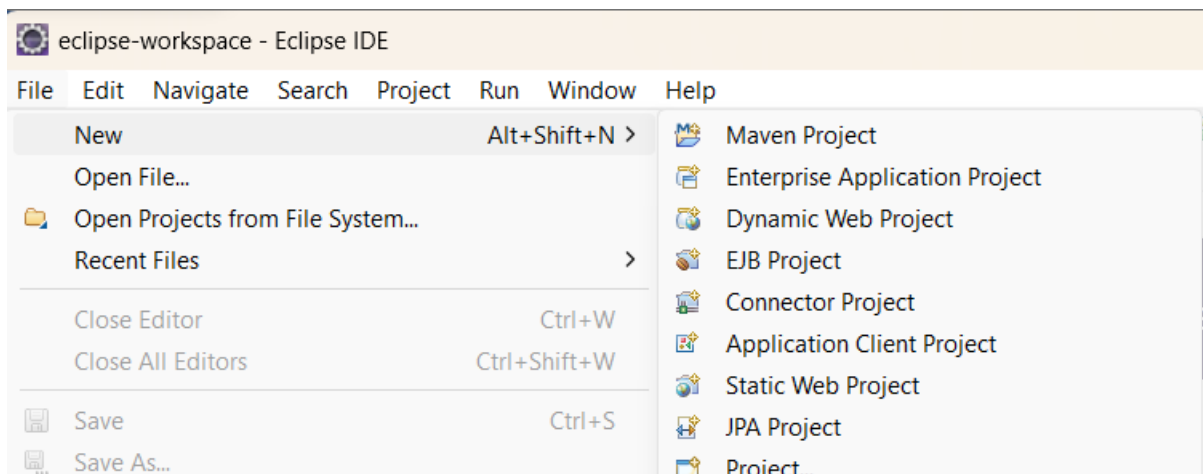
2. Download the m2e plugin. It facilitates the build process.
In the Eclipse IDE, go to Help → Install New Software.
Enter the URL given below in Work with area.



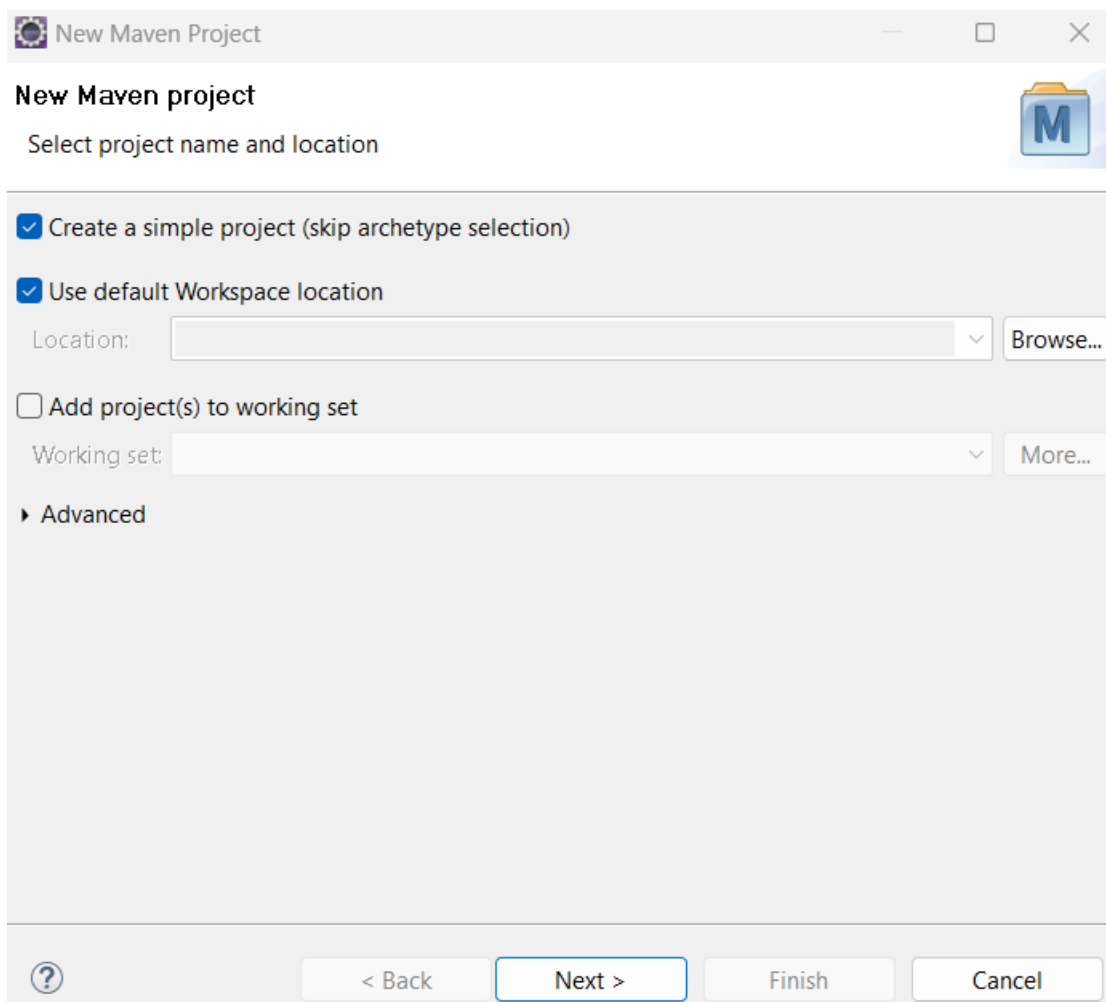
3. We also have to install the TestNG plugin. Refer to the documentation of TestNG and enter the relevant available URL in the Work with area.



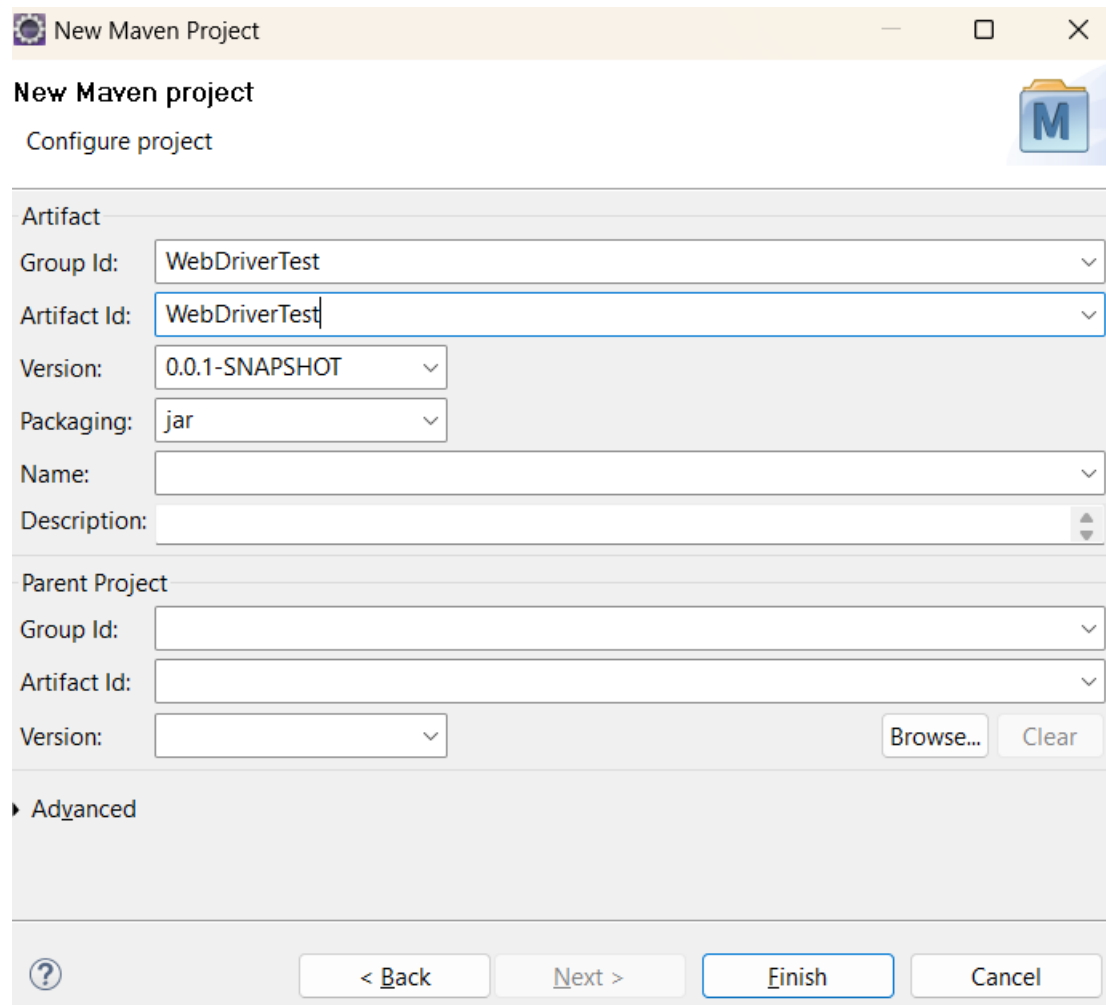
4. Click on File → New → Maven Project



5. Select the following, make sure you're using default workspace location.



6. Click on Next and enter the following details. Then click on Finish.



New Maven Project

New Maven project

Configure project

Artifact

Group Id: WebDriverTest

Artifact Id: WebDriverTest

Version: 0.0.1-SNAPSHOT

Packaging: jar

Name:

Description:

Parent Project

Group Id:

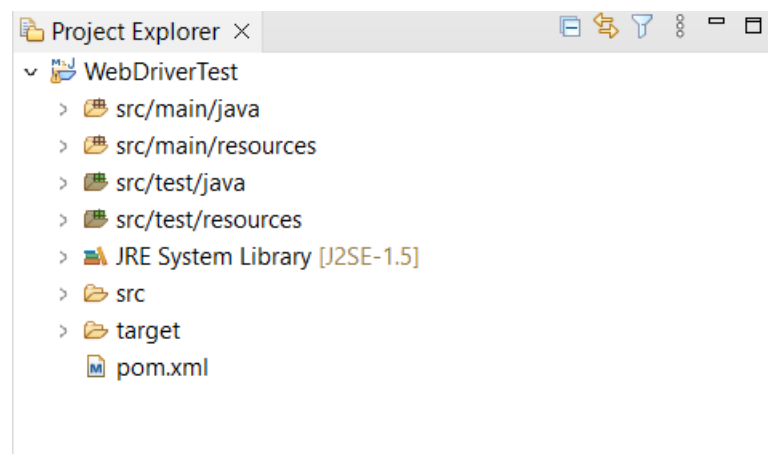
Artifact Id:

Version: Browse... Clear

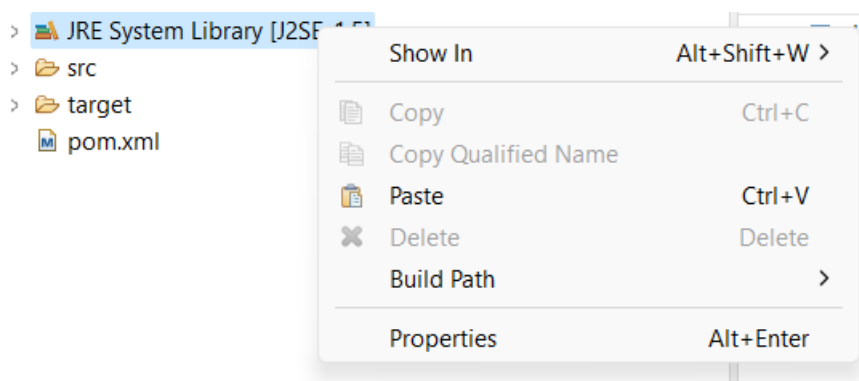
Advanced

< Back Next > Finish Cancel

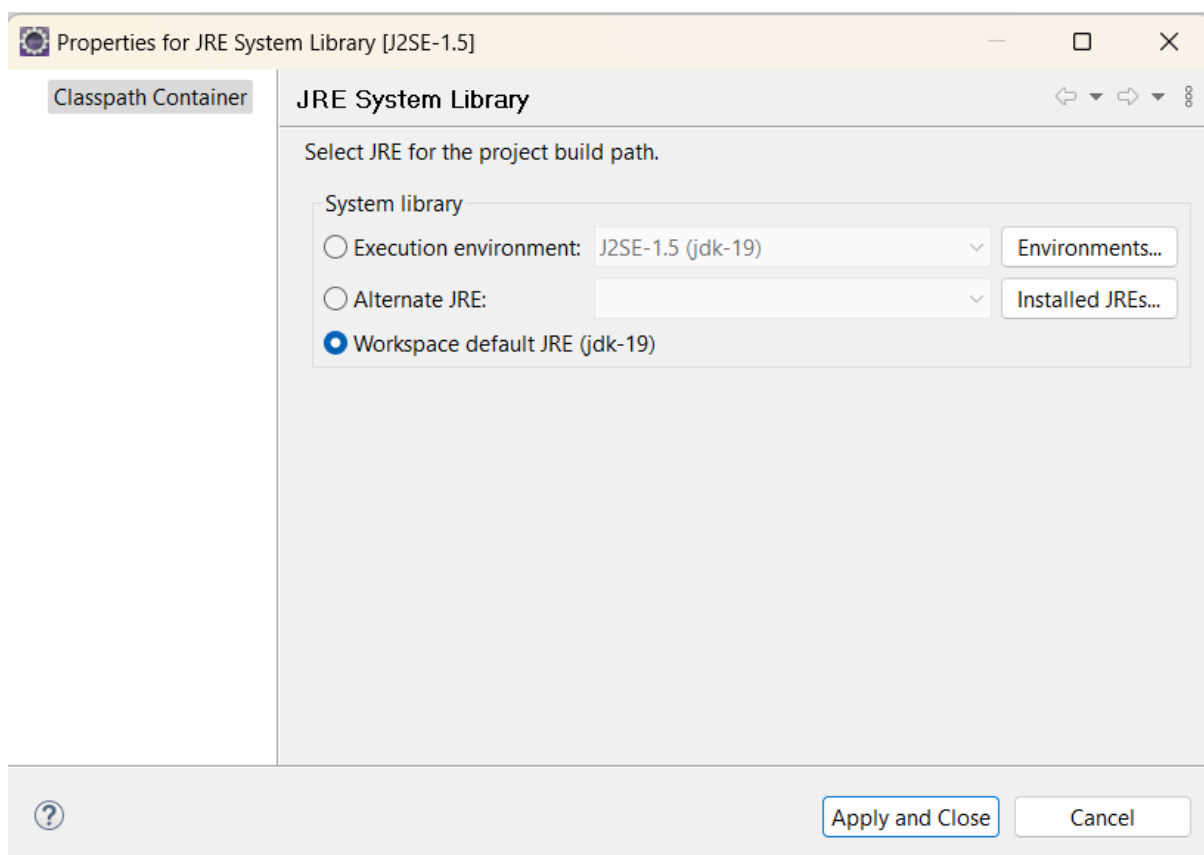
7. The Eclipse IDE will create WebDriverTest with the following structure:



8. Click on JRE System Library → Properties.



9. Check that the Workspace default JRE is selected here. Click on Apply and Close.



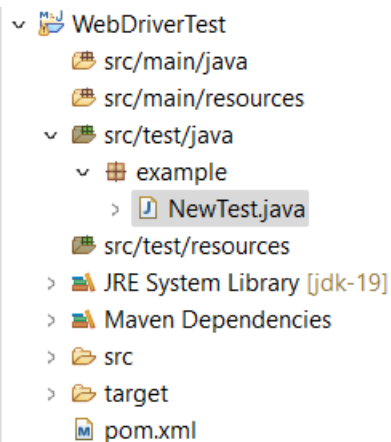
10. Select the pom.xml file and Open it in Editor Section.



11. Add the Selenium, Maven, TestNG, Junit dependencies to pom.xml in the <project> node:

```
6⑥ <dependencies>
7⑦ <dependency>
8  <groupId>junit</groupId>
9  <artifactId>junit</artifactId>
10 <version>3.8.1</version>
11 <scope>test</scope>
12 </dependency>
13⑧ <dependency>
14 <groupId>org.seleniumhq.selenium</groupId>
15 <artifactId>selenium-java</artifactId>
16 <version>3.12.0</version>
17 </dependency>
18⑨ <dependency>
19 <groupId>org.testng</groupId>
20 <artifactId>testng</artifactId>
21 <version>7.4.0</version>
22 <scope>test</scope>
23 </dependency>
24 </dependencies>
25 </project>
```

12. Creating a new TestNG Class. Create a new package named package → Create a new file named NewTest.java



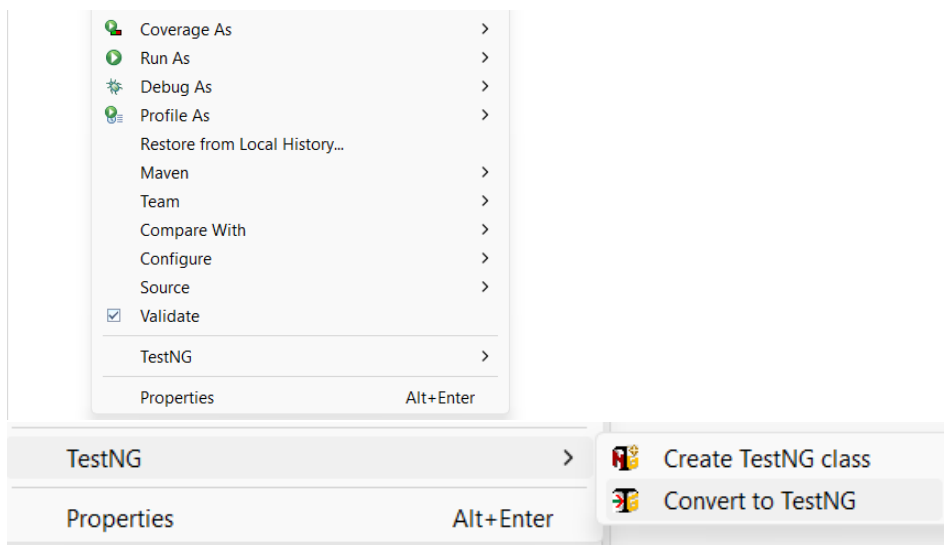
```

v WebDriverTest
  src/main/java
  src/main/resources
  v src/test/java
    v example
      > NewTest.java
  src/test/resources
  > JRE System Library [jdk-19]
  > Maven Dependencies
  > src
  > target
  pom.xml
```

13. Enter the following code in the NewTest.java file. This verifies the title of Selenium Page.

```
*WebDriverTest/pom.xml  .project  .classpath  *NewTest.java ×
1 package example;
2
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.firefox.FirefoxDriver;
5 import org.testng.Assert;
6 import org.testng.annotations.Test;
7 import org.testng.annotations.BeforeTest;
8 import org.testng.annotations.AfterTest;
9 public class NewTest {
10 private WebDriver driver;
11 @Test
12 public void testEasy() {
13 driver.get("http://demo.guru99.com/test/guru99home/");
14 String title = driver.getTitle();
15 Assert.assertTrue(title.contains("Demo Guru99 Page"));
16 }
17 @BeforeTest
18 public void beforeTest() {
19 System.setProperty("webdriver.gecko.driver",
20 "<PATH\\TO\\geckodriver.exe");
21 driver = new FirefoxDriver();
22 }
23 @AfterTest
24 public void afterTest() {
25 driver.quit();
26 }
27 }
```

14. Right click on WebDriverTest. Select TestNG → Convert to TestNG. A testing.xml file will be created.

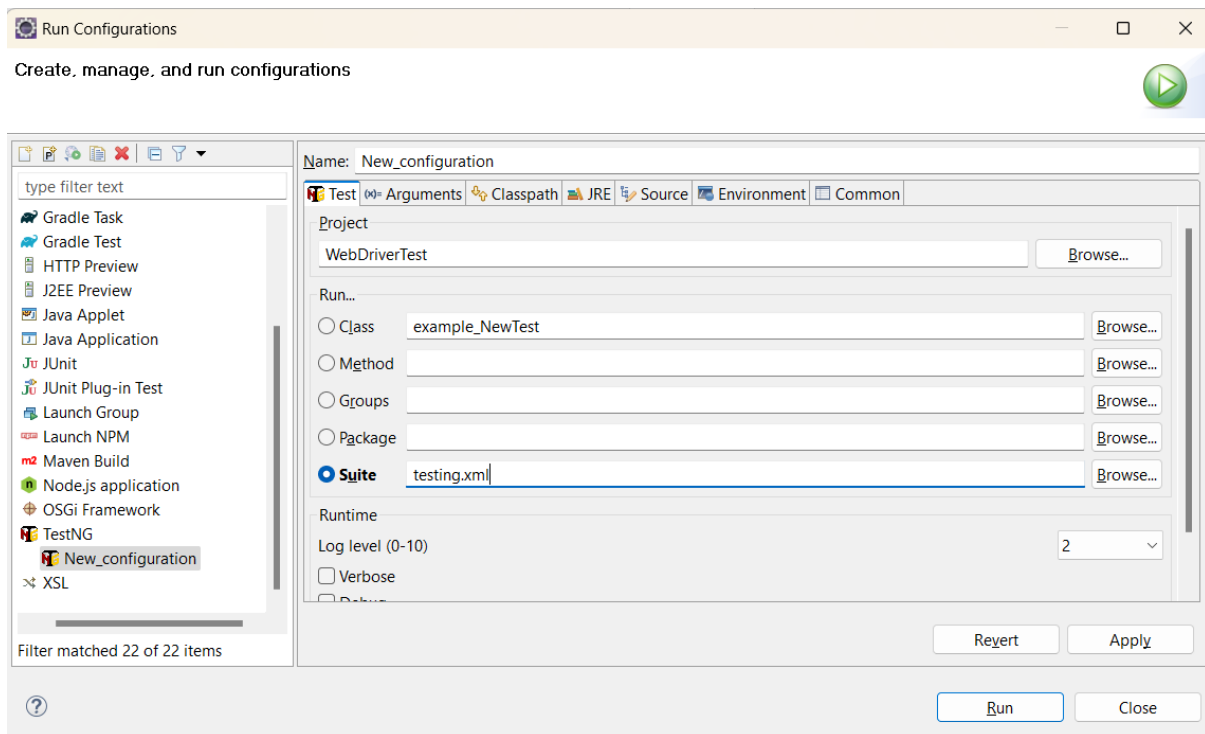


```
WebDriverTest/pom.xml  .project  .classpath  NewTest.java  testng.xml  testng.xml ×
https://testng.org/testng-1.0.dtd (doctype)
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
3 <suite name="Suite">
4   <test thread-count="5" name="Test">
5     <classes>
6       <class name="example.NewTest"/>
7     </classes>
8   </test> <!-- Test -->
9 </suite> <!-- Suite -->
10
```

15. Now, we need to run the test through this testing.xml

Go to Run → Run Configurations → TestNG.

Type the following details and select Suite as testing.xml. Then click on Run.



Make sure that the configuration of Java is same and you have geckodriver.exe (or any browser driver executable) in the path of your workspace.

```

Markers Properties Servers Data Source Explorer Snippets Terminal Console × Results of running suite
<terminated> New_configuration [TestNG] C:\Program Files\Java\jdk-19\bin\javaw.exe (01-May-2023, 4:39:35 pm – 4:39:51 pm) [pid: 13088]
JavaScript error: resource://gre/modules/AsyncShutdown.sys.mjs, line 727: Error: Phase "profile-be
JavaScript error: resource://gre/modules/AsyncShutdown.sys.mjs, line 727: Error: Phase "profile-be

=====
Suite
Total tests run: 1, Passes: 1, Failures: 0, Skips: 0
=====

```

16. We also need to add

- a. maven-compiler-plugin
- b. maven-surefire-plugin
- c. testng.xml to pom.xml

The maven-surefire-plugin is used to configure and execute tests. Here plugin is used to configure the testing.xml for TestNG test and generate test reports. The maven-compiler-plugin is used to help in compiling the code and using the particular JDK version for compilation. Add all dependencies in the following code snippet, to pom.xml in the <plugin> node:

```

</dependencies>
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.8.1</version>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
    </plugin>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>2.12</version>
      <inherited>true</inherited>
      <configuration>
        <suiteXmlFiles>
          <suiteXmlFile>testng.xml</suiteXmlFile>
        </suiteXmlFiles>
      </configuration>
    </plugin>
  </plugins>
  <pluginManagement>
    <plugins>
      <!--This plugin's configuration is used to store Eclipse m2e
settings only. It has no influence on the Maven build itself.-->

```

```

54<    <plugin>
55      <groupId>org.eclipse.m2e</groupId>
56      <artifactId>lifecycle-mapping</artifactId>
57      <version>1.0.0</version>
58      <configuration>
59        <lifecycleMappingMetadata>
60          <pluginExecutions>
61            <pluginExecution>
62              <pluginExecutionFilter>
63                <groupId>
64                  org.apache.maven.plugins
65                </groupId>
66                <artifactId>
67                  maven-compiler-plugin
68                </artifactId>
69                <versionRange>
70                  [3.8.2,)
71                </versionRange>
72              <goals>
73                <goal>testCompile</goal>
74              </goals>
75            </pluginExecutionFilter>
76            <action>
77              <ignore></ignore>
78            </action>
79          </pluginExecution>
80        </pluginExecutions>
81      </lifecycleMappingMetadata>

```

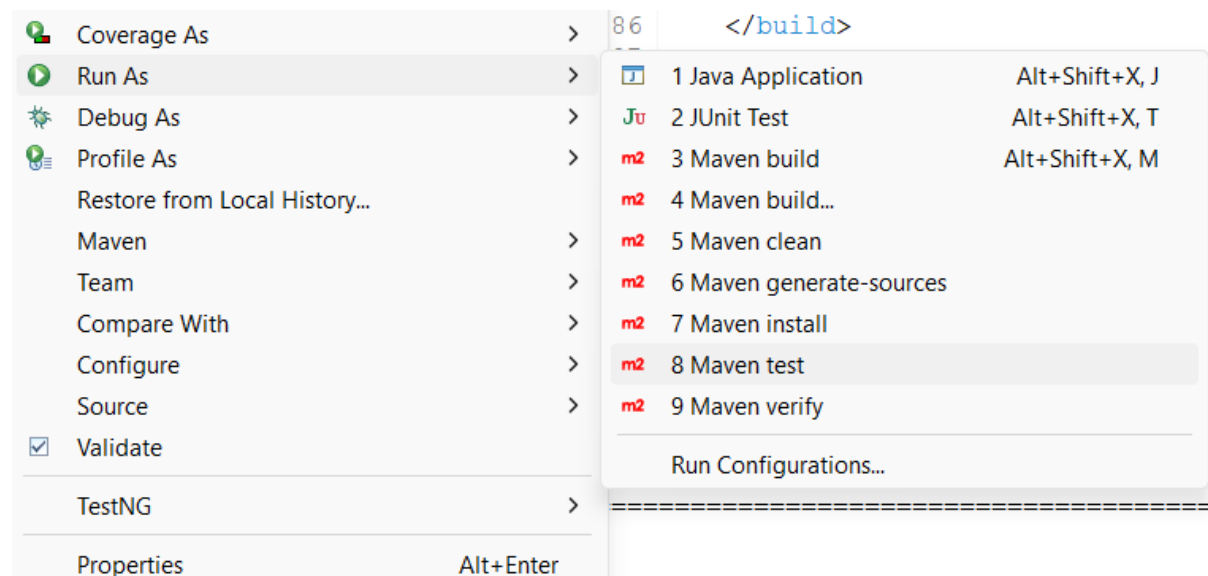


```

73         <goal>testCompile</goal>
74     </goals>
75 </pluginExecutionFilter>
76 <action>
77     <ignore></ignore>
78 </action>
79 </pluginExecution>
80 </pluginExecutions>
81 </lifecycleMappingMetadata>
82 </configuration>
83 </plugin>
84 </plugins>
85 </pluginManagement>
86 </build>
87
88
89 </project>

```

17. Right click on WebDriverTest → Run as → Maven Test



Make sure that the build finishes successfully.

```

Overview | Dependencies | Dependency Hierarchy | Effective POM | pom.xml
Markers | Properties | Servers | Data Source Explorer | Snippets | Terminal | Console | Results of running
<terminated> C:\Program Files\Java\jdk-19\bin\javaw.exe (01-May-2023, 4:50:13 pm) [pid: 21852]
Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 21.584 s
[INFO] Finished at: 2023-05-01T16:50:37+05:30
[INFO] -----

```

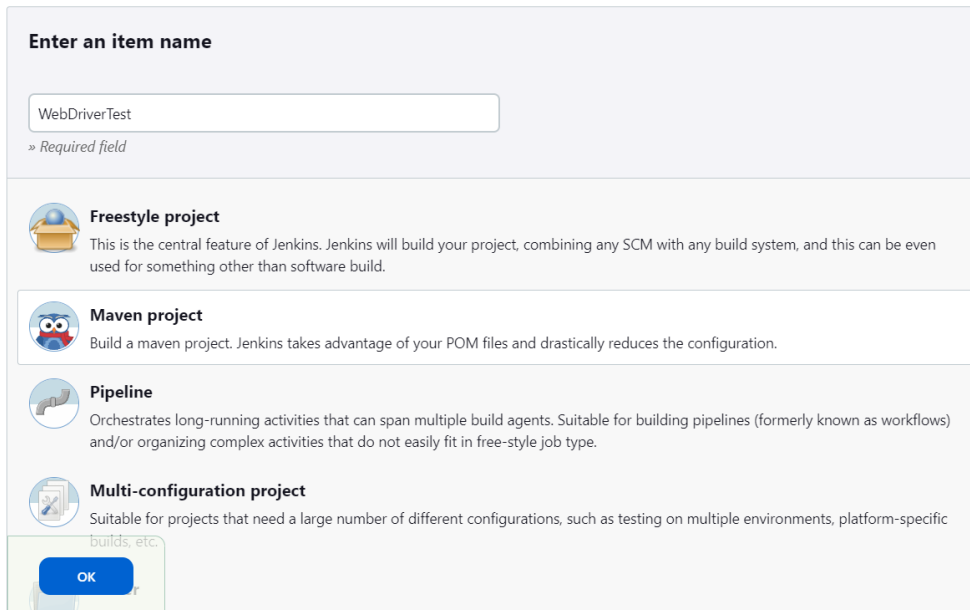
Setting up Selenium Project on Jenkins

18. Go on to the Jenkins Dashboard → New Item.

Type the Item Name and select Maven Project. Click OK.

Note: Make sure Maven Integration Plugin is installed. Otherwise install it beforehand itself.

Without this plugin you won't be able to see the Maven project option in Jenkins.



Enter an item name

WebDriverTest

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

OK

19. Go to the Build Section of your new job.

- In the Root POM textbox, enter the path to your pom.xml file
- In the Goals and options section, enter “clean test”
- Click on Save

Build

Maven Version

! Jenkins needs to know where your Maven is installed.
Please do so from the tool configuration.

Root POM ?

D:\Subrato\eclipse-workspace\WebDriverTest\pom.xml

Goals and options ?

clean test

Advanced ▼

20. For the Maven Version warning/error above:

- Go to Global Tools Configuration
- Maven → Install from Apache.

- i. Alternatively, if you have Maven installed on your local machine, you can mention the path here. This will take lesser time as compared to installing it from Apache.
- ii. Apache one will take time while building for the first time. But the successive builds will take less time.

Maven

Maven installations ^ Edited

Maven installations

List of Maven installations on this system

Add Maven

Maven Name

Maven

☒ Install automatically ?

Install from Apache

Version

3.9.1

Add Installer

Add Maven

21. Click on Build Now. Wait for some time as the process gets finished.

Console Output

```
Started by user Subrato Tapaswi
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\WebDriverTest
Unpacking https://repo.maven.apache.org/maven2/org/apache/maven/apache-maven/3.9.1/apache-maven-3.9.1-bin.zip to
C:\ProgramData\Jenkins\.jenkins\tools\udson.tasks.Maven_MavenInstallation\Maven on Jenkins
Parsing POMs
Discovered a new module WebDriverTest:WebDriverTest WebDriverTest
Modules changed, recalculating dependency graph
Established TCP socket on 51124
[WebDriverTest] $ java -cp C:\ProgramData\Jenkins\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven35-agent-
1.14.jar;C:\ProgramData\Jenkins\.jenkins\tools\udson.tasks.Maven_MavenInstallation\Maven\boot\plexus-classworlds-
2.6.0.jar;C:\ProgramData\Jenkins\.jenkins\tools\udson.tasks.Maven_MavenInstallation\Maven\conf\logging-jenkins-maven3-agent.Maven35Main
C:\ProgramData\Jenkins\.jenkins\tools\udson.tasks.Maven_MavenInstallation\Maven C:\ProgramData\Jenkins\war\WEB-INF\lib\remoting-
3107.v665000b_51092.jar C:\ProgramData\Jenkins\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven35-interceptor-1.14.jar
C:\ProgramData\Jenkins\.jenkins\plugins\maven-plugin\WEB-INF\lib\maven35-interceptor-1.14.jar 51124
```

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 17.708 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[JENKINS] Recording test results

[INFO] -----

[INFO] BUILD SUCCESS

[INFO] -----

[INFO] Total time: 01:05 min

[INFO] Finished at: 2023-05-01T17:03:28+05:30

[INFO] -----

Waiting for Jenkins to finish collecting data

[JENKINS] Archiving D:\Subrato\eclipse-workspace\WebDriverTest\pom.xml to WebDriverTest\WebDriverTest\0.0.1-SNAPSHOT\WebDriverTest-0.0.1-

SNAPSHOT.pom

channel stopped

Finished: SUCCESS