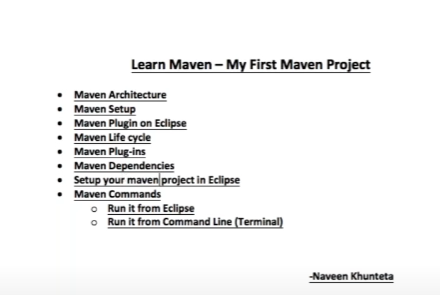
***Maven :***



Maven Is the build automation tool which generates the build.

--------------------------------

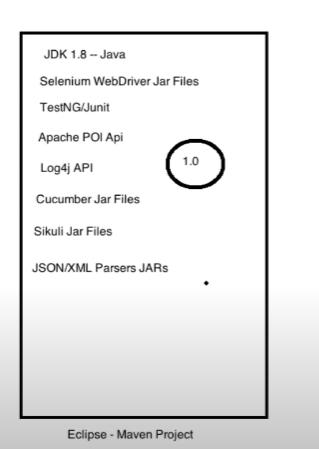
Jdk is necessary for all the things in java , eclipse.

For the selenium , we need the selenium web drivers.Fot the testing/JUNIT , we need the testing/ JUNIT jar files.

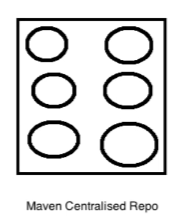
To work on the excel , we need the apache POI jar files.

For log4j ,Sikuli JSON/XML parsers and for all other stuff we need the respective jar files.

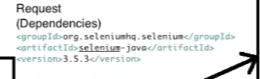
For the typical project to work on , we go to the project and ll the jar files in the add external jar or libraries in the add library section.



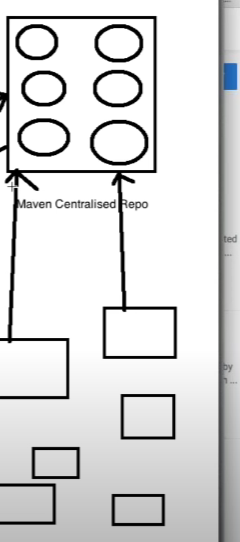
But in maven everything is available as jars in form of dependency as. So for everything in maven there is a centralized repository which consists of the dependency for each .

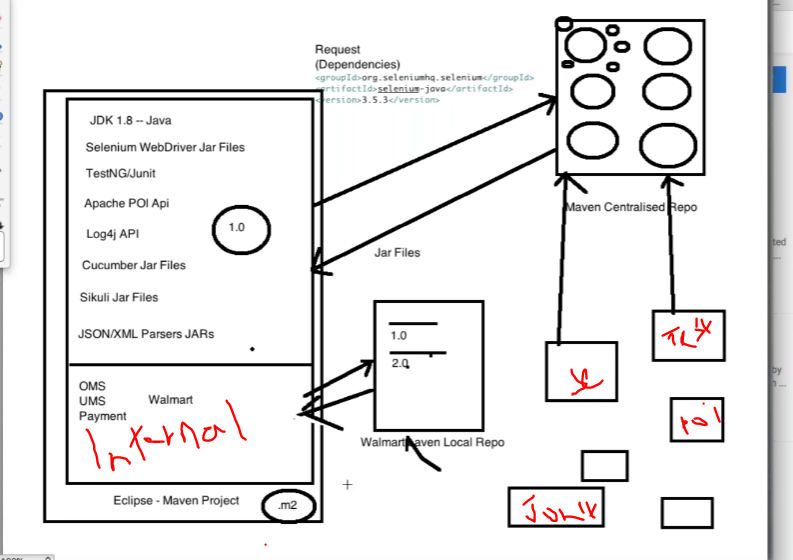


So if there are different companies, like Thought works (Selenium), TestNG etc … they all will provide their dependency in the centralized maven repo as below: in the form of the group ID, artifact id and version.



Whenever we download the maven , there will be one hidden directory created (as .m2) in our machine



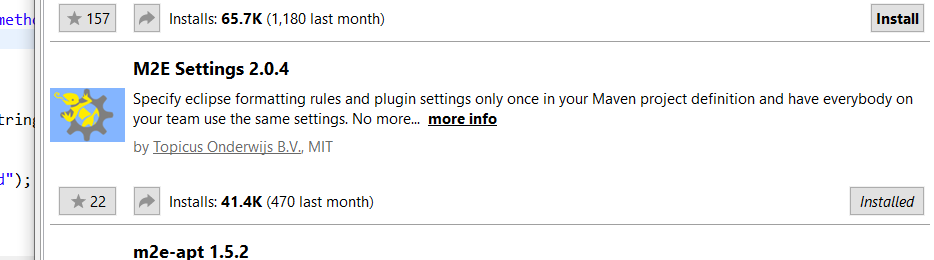


Installing maven on windows:

<https://mkyong.com/maven/how-to-install-maven-in-windows/>

Maven plug in on eclipse:

Fo to eclipse



Maven Lifecycle:

1. Maven compile: compiling all the java classes and code. This compile will be done through Maven compiler plug in.
2. Maven Test : Test here means testcases (Unit test cases,integration,functional – using Junit/TestNG) done through Maven sure fire plugin.
3. Maven resource Jar: These can create 3 resources: JARs/Wars(web archivers)/EARs(enterprise archival files) by Maven resources plugin(to generate the resources).

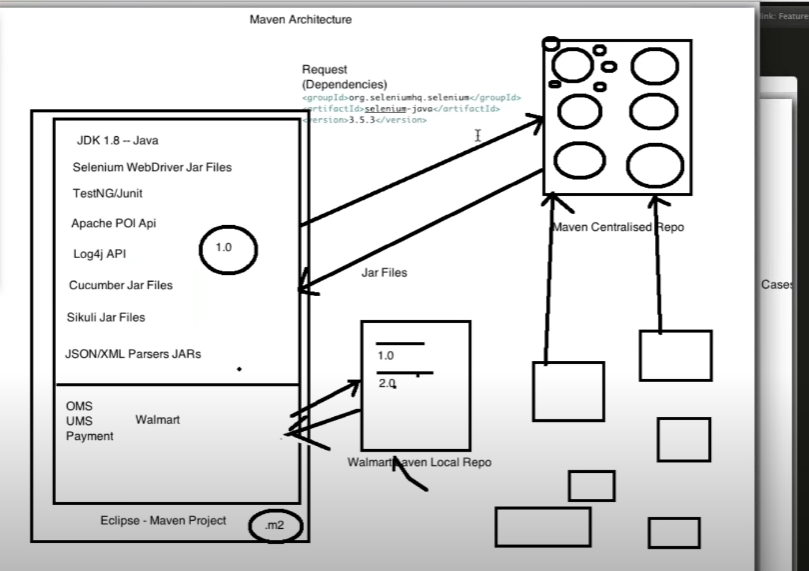
Other same tools available are gradle,ANT.

So from above we can see that maven has 3 plugins

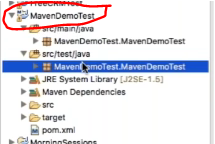
* Maven compiler plugin for compiling
* Maven sure fire plug in for testing
* Maven resource p lug in for generating the resource files

Maven dependency :

It will be either external dependency or internal dependency :It can be maven world wide repo or internal company repo.



Creating Maven project in eclipse :



Then in the pom.xml filr we need to add the dependencies :

For testing we need to add the scope as compile:

<!-- https://mvnrepository.com/artifact/org.testng/testng -->

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.13</version>

<scope>compile</scope>

</dependency>

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

<dependency>

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

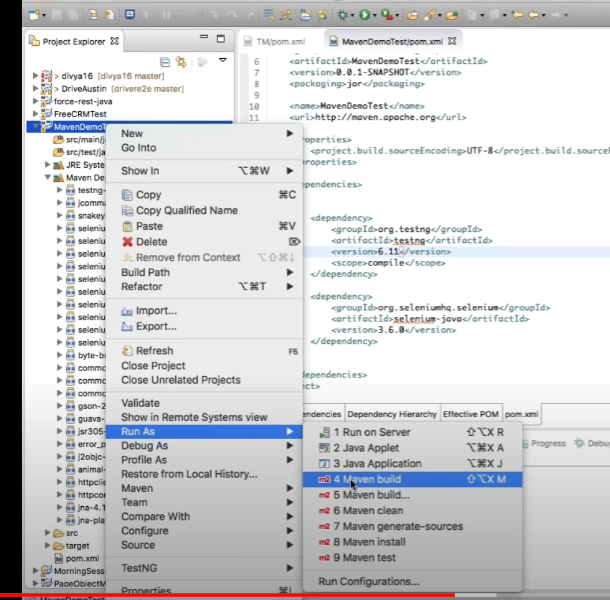
<artifactId>selenium-java</artifactId>

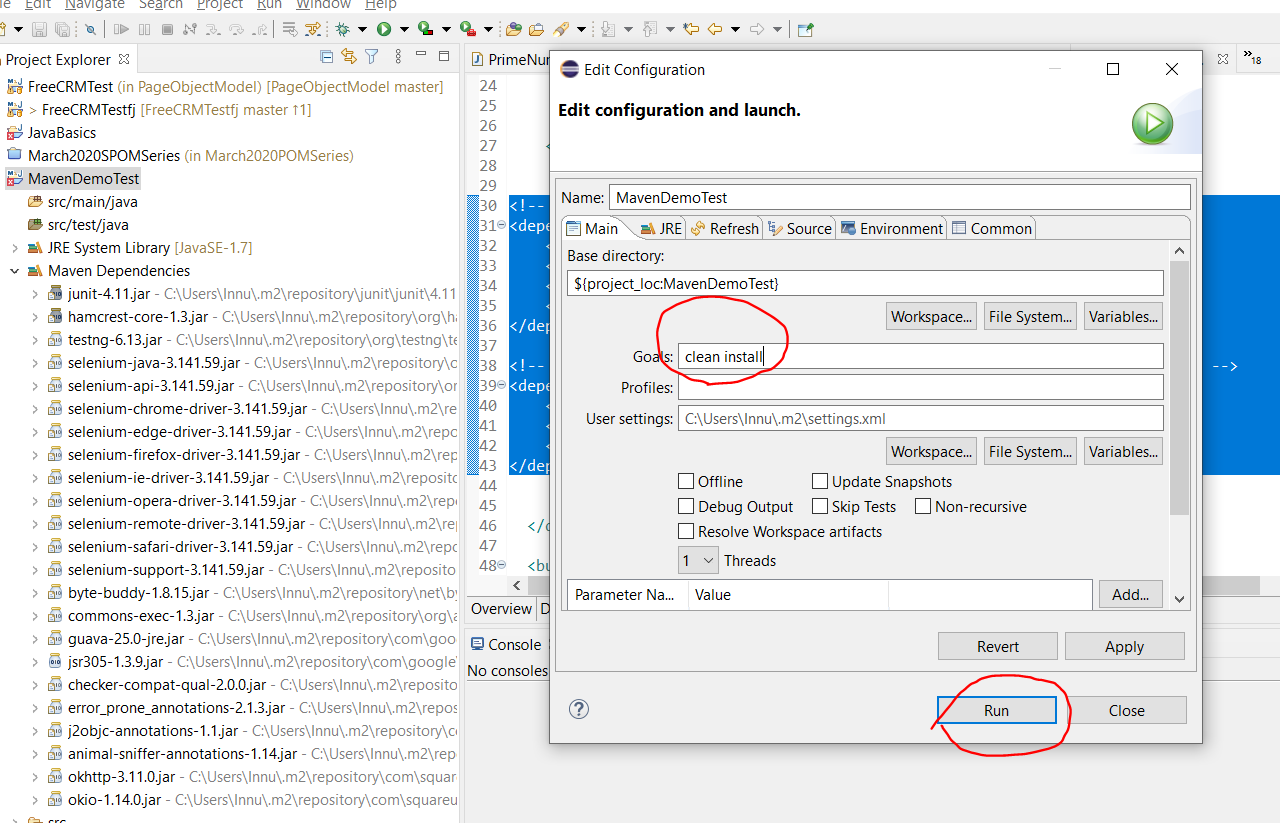
<version>3.141.59</version>

</dependency>

=====

Some times even after we have all the dependencies in eclipse it doesn’t get downloaded properly.For this , we need to do clean install :





After creating the project in eclipse we run directly run 🡪TestNG

But if ew don’t want to create the testing reports and run directly from maven then goto the project and right click and Run as 🡪 Maven test.

But if there is some remote machine which we need to run the command maven then we need to go to directory in the cmd :

Directory of D:\MyWorkSpace\Java\MavenDemoTest

Maven clean install will create the build (jar files).If we want to build then we need to run the cmd: mvn clean install else if we want to directly run tests then we can do mvn test

07/22/2020 06:16 PM <DIR> .

07/22/2020 06:16 PM <DIR> ..

07/22/2020 06:16 PM 1,132 .classpath

07/22/2020 05:57 PM 565 .project

07/22/2020 05:57 PM <DIR> .settings

07/22/2020 06:08 PM 3,418 pom.xml

07/22/2020 05:56 PM <DIR> src

07/22/2020 06:16 PM <DIR> target

07/22/2020 06:21 PM <DIR> test-output

3 File(s) 5,115 bytes

6 Dir(s) 100,718,297,088 bytes free

D:\MyWorkSpace\Java\MavenDemoTest>ls

'ls' is not recognized as an internal or external command,

operable program or batch file.

D:\MyWorkSpace\Java\MavenDemoTest>mvn clean install

[INFO] Scanning for projects...

[INFO]

[INFO] --------------------< MavenDemoTest:MavenDemoTest >---------------------

[INFO] Building MavenDemoTest 0.0.1-SNAPSHOT

[INFO] --------------------------------[ jar ]---------------------------------

[WARNING] The POM for com.beust:jcommander:jar:1.66 is missing, no dependency information available

[INFO]

[INFO] --- maven-clean-plugin:3.1.0:clean (default-clean) @ MavenDemoTest ---

[INFO] Deleting D:\MyWorkSpace\Java\MavenDemoTest\target

[INFO]

[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ MavenDemoTest ---

[WARNING] JAR will be empty - no content was marked for inclusion!

[INFO] Building jar: D:\MyWorkSpace\Java\MavenDemoTest\target\MavenDemoTest-0.0.1-SNAPSHOT.jar

[INFO]

[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ MavenDemoTest ---

[INFO] Installing D:\MyWorkSpace\Java\MavenDemoTest\target\MavenDemoTest-0.0.1-SNAPSHOT.jar to C:\Users\Innu\.m2\repository\MavenDemoTest\MavenDemoTest\0.0.1-SNAPSHOT\MavenDemoTest-0.0.1-SNAPSHOT.jar

[INFO] Installing D:\MyWorkSpace\Java\MavenDemoTest\pom.xml to C:\Users\Innu\.m2\repository\MavenDemoTest\MavenDemoTest\0.0.1-SNAPSHOT\MavenDemoTest-0.0.1-SNAPSHOT.pom

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 5.590 s

[INFO] Finished at: 2020-07-22T18:30:15-04:00

[INFO] ------------------------------------------------------------------------

Now if we don’t want to execute the testcases and just want the build, we can ignore the test cases

D:\MyWorkSpace\Java\MavenDemoTest>mvn package -DskipTests

[INFO] Scanning for projects...

[INFO]

[INFO] --------------------< MavenDemoTest:MavenDemoTest >---------------------

[INFO] Building MavenDemoTest 0.0.1-SNAPSHOT

[INFO] --------------------------------[ jar ]---------------------------------

[WARNING] The POM for com.beust:jcommander:jar:1.66 is missing, no dependency information available

[INFO]

[INFO] --- maven-resources-plugin:3.0.2:resources (default-resources) @ MavenDemoTest ---

[INFO] Using 'UTF-8' encoding to copy filtered resources.

[INFO] skip non existing resourceDirectory D:\MyWorkSpace\Java\MavenDemoTest\src\main\resources

[INFO]

[INFO] --- maven-compiler-plugin:3.8.0:compile (default-compile) @ MavenDemoTest ---

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-resources-plugin:3.0.2:testResources (default-testResources) @ MavenDemoTest ---

[INFO] Using 'UTF-8' encoding to copy filtered resources.

[INFO] skip non existing resourceDirectory D:\MyWorkSpace\Java\MavenDemoTest\src\test\resources

[INFO]

[INFO] --- maven-compiler-plugin:3.8.0:testCompile (default-testCompile) @ MavenDemoTest ---

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-surefire-plugin:2.22.1:test (default-test) @ MavenDemoTest ---

[INFO] Tests are skipped.

[INFO]

[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ MavenDemoTest ---

[WARNING] JAR will be empty - no content was marked for inclusion!

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

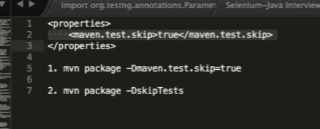
[INFO] Total time: 1.534 s

[INFO] Finished at: 2020-07-22T18:39:25-04:00

[INFO] ---

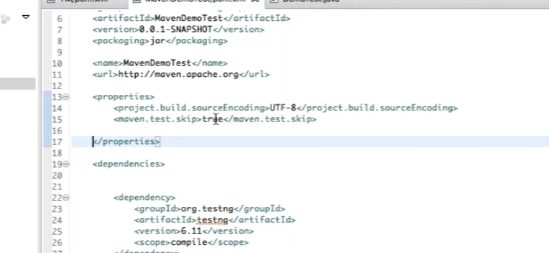
There are two ways to skip the test cases :

1. mvn package -DskipTests
2. mvn package -Dmaven.test.skip=true (for this to work , we need to add this to the pom.xml

 ---------------------------------------------------------------------

Pom.xml:

<maven.test.skip>true</maven.test.skip>



If we don’t add the above property in the pom.xml (<maven.test.skip>true</maven.test.skip>

) and run the line in cmd:

D:\MyWorkSpace\Java\MavenDemoTest>mvn package -Dmaven.test.skip=true

[INFO]

[INFO] --- maven-compiler-plugin:3.8.0:compile (default-compile) @ MavenDemoTest ---

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-resources-plugin:3.0.2:testResources (default-testResources) @ MavenDemoTest ---

[INFO] Not copying test resources

[INFO]

[INFO] --- maven-compiler-plugin:3.8.0:testCompile (default-testCompile) @ MavenDemoTest ---

[INFO] Not compiling test sources

[INFO]

[INFO] --- maven-surefire-plugin:2.22.1:test (default-test) @ MavenDemoTest ---

[INFO] Tests are skipped.

[INFO]

[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ MavenDemoTest ---

[WARNING] JAR will be empty - no content was marked for inclusion!

INFO] ------------------------------------------------------------------------

And after adding in pom.xml , its building the jar file

[INFO] Nothing to compile - all classes are up to date

[INFO]

[INFO] --- maven-resources-plugin:3.0.2:testResources (default-testResources) @ MavenDemoTest ---

[INFO] Not copying test resources

[INFO]

[INFO] --- maven-compiler-plugin:3.8.0:testCompile (default-testCompile) @ MavenDemoTest ---

[INFO] Not compiling test sources

[INFO]

[INFO] --- maven-surefire-plugin:2.22.1:test (default-test) @ MavenDemoTest ---

[INFO] Tests are skipped.

[INFO]

[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ MavenDemoTest ---

[INFO] Building jar: D:\MyWorkSpace\Java\MavenDemoTest\target\MavenDemoTest-0.0.1-SNAPSHOT.jar

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 1.578 s

[INFO] Finished at: 2020-07-23T10:56:45-04:00

[INFO] ------------------------------------------------------------------------