**Pom.xml file:**

How does maven know what resources are required for building your project? This is through an xml file which is called pom.xml.

In pom.xml we mention all the resources in the form of resources and dependencies.

When you run any command like mvn deploy it searches for the pom.xml file in the local directory and runs the command.

mvn -f pom.xml deploy

mvn deploy – will search for the pom.xml in the current directory

mvn clean deploy – clean command will clear the target directory each time.

**Example pom.xml file:**

POM means project object model. It is an XML representation of a Maven project held in a file “pom.xml”. The below is an example pom.xml file.

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.google.finance</groupId>

<artifactId>quick.analytics</artifactId>

<packaging>jar</packaging>

<version>1.0-SNAPSHOT</version>

<name>Test Version</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

Maven Coordinates:

groupId, artifactId and version are the maven coordinates.

groupId: This is the unique name given to the project so that it can be identified uniquely across all the projects. Generally, this groupId is the company domain name in the reverse order with out www followed by department followed by anything else you want. For example if google is developing a project called “quick analytics” by the finance department then this group id can be “com.google.finance”

artifactId: Artifact is a file, usually a JAR that gets deployed to a Maven repository. Generally this is the project name so this can be “quick.analytics”.

version: This is the version for the jar/war/ear files generated. Generally first test files will be generated so these are named as 1.0-SNAPSHOT. If everything works then version will be changed to the actual version like 1.0.

The other elements in the pom.xml are

packaging: here you mention whether you want to build a jar file or war file or ear file.

name: This the name of jar file created. If it is a test version (SNAPSHOT) we are creating, we will give name like Test Version etc otherwise will give the actual name.

This is the basic information we will provide.

There are another two main types of information we have to specify.

1. Dependency information
2. Plugin information.

**Dependency Information:**

There are two dependencies we have to use.

1. Selenium dependency
2. TestNG dependency

Go to google and search for “Selenium Maven Dependency” and go to the maven repository link and copy the dependency section. The following is the link.

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

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>3.14.0</version>

</dependency>

Similarly get TestNG dependency. You can get it from

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

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.14.3</version>

<scope>test</scope>

</dependency>

Now under the dependencies section, update the xml with the above two.

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.google.finance</groupId>

<artifactId>quick.analytics</artifactId>

<packaging>jar</packaging>

<version>1.0-SNAPSHOT</version>

<name>Test Version</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>3.14.0</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.14.3</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

Plugin Information:

Go to google and search for “Maven Surefire Plugin xml” and you will get the following.

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

</plugin>

</plugins>

</pluginManagement>

</build>

In addition to the above since we want to run the tests from TestNG, we need to have the configuration section as well in <plugin>

Add it to the pom.xml above dependencies.

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

So the plugins section becomes,

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</pluginManagement>

</build>

In the above tesng.xml is the TestNG xml file we provide.

So now the overall pom.xml look like

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.google.finance</groupId>

<artifactId>quick.analytics</artifactId>

<packaging>jar</packaging>

<version>1.0-SNAPSHOT</version>

<name>Test Version</name>

<url>http://maven.apache.org</url>

<build>

<pluginManagement>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</pluginManagement>

</build>

<dependencies>

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>3.14.0</version>

</dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.14.3</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

This forms our whole pom.xml for running our selenium tests. There might a lot of other dependencies and plugins available but those are used by developers or devops engineers and completely for build or packaging purpose. For running selenium tests with TestNG this xml should be fine.