Maven

- Maven is a powerful project management tool that is based on pom (project object model). It is used for projects build, dependency and documentation.

- Problem without maven

  ① adding set of Jars in each project
  ② creating the right project structure
  ③ Building and deploying the project.

- What it does?

  ① It makes a project easy to build
  ② It provides uniform build process
  ③ It provides project information
    - log document
    - cross referenced sources
    - mailing list
    - dependency list
    - unit test report

  ④ It is easy to migrate for new features of maven.

Apache

Maven helps to manage

- Builds
- Documentation
- Reporting
- SCMs
- Releases
- Distribution
- mailing list

A build tool take care of everything for building a process it does following,

1. Generate source code
2. Generate documentation from source code
3. Compiler source code
4. Packages compiled code into JAR & ZIP file.
5. installs the packaged code in local repository, server repository or central repository.

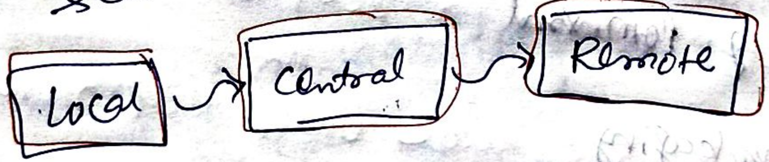
* maven Repository

A maven repository is a directory of packaged JAR file with pom.xml.

maven searches for the dependencies in the repositories, There are 3 types of maven repository.

1. local Repository
2. Central Repository
3. Remote Repository.

maven searches for the dependencies in following order



* Local : maven local repository is located in your local system.

Ex:- c:\users\abc\.m2

* Central Repository

maven central repository is located on the web. it has been created by the apache maven community itself.

http://repo1.maven.org/maven2/.

maven Remote Repository

located on the web, most of libraries can be missing from central repository to add in project we have to define manually in the Pom.xml.

Remote Repository were also be private to company.

* Pom.xml

file contains information of project and configuration information for the maven to build the project such as dependendencies, build directory, source directory, test source directory, plugin, goals etc.

- dependencies: Any library on which a java project is dependent on. to run or build is called dependency.

  Ex: selenium, POI, log4j, testNG ...

- surefire plugin:

  To execute the testcases and identify the tests when using TestNG.

- elements of Pom.xml

  1. packaging
  2. name
  3. url
  4. dependencies
  5. dependency
  6. scope
  7. Project
  8. plugin

```
<dependencies >
    <dependency >
        <groupId>org. testng </groupId>
        <version> 7.4.0 </version>
        < scope> test </scope>
    </dependency>
</dependencies>
```

Note: Refer the POM.xml Provided (Contains Dependencies plugin's).

~~plugins~~

General phrases used in maven ★

→ groupId : Generally groupId refers to domain id. for best practices company name is used as groupId. It identifies project uniquely.

→ artifactId : it is basically name of the Jar without version.

→ version : the tag is used to create a version of the project.

★ maven phases (Build life cycle):

① clean: deletes all artifacts and targets which are created already.

② Compile: used to compile the source code of the project.

③ test : test the compiled code and these tests donot require to be packaged or deployed.

④ Package : Package is used to convert your project into jar or war etc ---

⑤ install : installs the package into the local repository for use of another project.

Setup maven with command prompt

↳ ① download the apache-maven from

maven.apache.org/download.cgi

download

↳ apache-maven-3.8.4-bin.zip

② Extract and place in cdrive

③ Set the maven-HOME and path, Java-HOME variables in Environment settings.

④ Create the maven project with pom.xml updated

⑤ open the cmd and change dir to current project.

```
> cd /d D:\path of project dir. ↵
> ~~maven test~~ ↵
> mvn -version ↵
> mvn test ↵
```

↳ ② Setup maven for Eclipse

① go to new software under Help menu in eclipse

or

eclipse market place → search for maven → maven integration for eclipse (m2e)

② create maven project and pom.xml then update maven project.

③ Run As maven test

```
<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.atipune</groupId>
 <artifactId>MavenTest</artifactId>
 <version>0.0.1-SNAPSHOT</version>
  <build>
               <plugins>
                  <plugin>
                  <groupId>org.apache.maven.plugins</groupId>
                  <artifactId>maven-compiler-plugin</artifactId>
                  <version>2.3.2</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>3.0.0-M5</version>
    <configuration>
     <suiteXmlFiles>
             <suiteXmlFile>testng.xml</suiteXmlFile>
     </suiteXmlFiles>
    </configuration>
   </plugin>
 </plugins>
 </build>

<dependencies>
        <!-- https://mvnrepository.com/artifact/org.testng/testng -->
              <dependency>
                 <groupId>org.testng</groupId>
                 <artifactId>testng</artifactId>
                 <version>7.4.0</version>
                 <scope>test</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>

              <!-- https://mvnrepository.com/artifact/com.microsoft.edge/msedge-selenium-tools-java -->
              <dependency>
                 <groupId>com.microsoft.edge</groupId>
                 <artifactId>msedge-selenium-tools-java</artifactId>
                 <version>3.141.0</version>
              </dependency>
```

```
<!-- https://mvnrepository.com/artifact/commons-io/commons-io -->
<dependency>
    <groupId>commons-io</groupId>
    <artifactId>commons-io</artifactId>
    <version>2.11.0</version>
</dependency>

<!-- https://mvnrepository.com/artifact/commons-logging/commons-logging -->
<dependency>
    <groupId>commons-logging</groupId>
    <artifactId>commons-logging</artifactId>
    <version>1.2</version>
</dependency>

<!-- https://mvnrepository.com/artifact/com.aventstack/extentreports -->
<dependency>
    <groupId>com.aventstack</groupId>
    <artifactId>extentreports</artifactId>
    <version>5.0.1</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi</artifactId>
    <version>3.17</version>
</dependency>
```

```
                        <!-- https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml -->
                        <dependency>
                            <groupId>org.apache.poi</groupId>
                            <artifactId>poi-ooxml</artifactId>
                            <version>3.17</version>
                        </dependency>

                        <!-- https://mvnrepository.com/artifact/org.apache.logging.log4j/log4j-core -->
                        <dependency>
                            <groupId>org.apache.logging.log4j</groupId>
                            <artifactId>log4j-core</artifactId>
                            <version>2.14.1</version>
                        </dependency>

            <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
            <dependency>
                <groupId>mysql</groupId>
                <artifactId>mysql-connector-java</artifactId>
                <version>8.0.15</version>
            </dependency>
</dependencies>
 </project>
```