Maven =

Maven is a build automation tool but it is much more than that is a project management tool. Building a project means compiling the source code running the tests which could be unit tests as well as integration tests packaging the complied code into jar files building these jar files into web archive or a war file deploying these wars on to the servers other tasks all these tasks can be automated using tools such as maven.

Maven ueses **convention over configuration** that is

In that project(base) directory there are four child folder.

Project-Name(Base Project Directory)

**src/main/java** – In that folder all the source code are present.

**src/main/resource** – In that folder all the resources files means property files or XML configuration that are use in our java classes are present.

**src/test/java** – In that folder all our unit and integration tests are present.

**src/test/resources** – In that folder all the resources files means property files or XML configuration that are use in our test are placed.

Execute structure of maven –

Firstly we run command a **mvn install** maven will compile the source code under the src/main/java run the unit tests under the src/test/java and if test passed it will create a bundle or package the compiled class into **jar** file if it is a standalone java application and if it is web application it will bundle it into **war** file onto the web application.

The folder structure is slightly different for java application as well as web application.

Archetypes=

Maven provides several Archetypes(tempelets) we can execute a command with an archetype and it will create the folder structure the standard folder structure required by maven for us.

There are different types of archetypes like standalone, webapp, EAR, etc

Why Maven-

1. Maven provides **common interface** for developers.
2. Maven is not a just build management tool but it can also manage **dependencies** that is our project is depend on other project.

<dependency>

<artifactid> junit.jar …..

</dependency>

1. In **repository** we puts all the artifacts and plugins.
2. **Reuse** - Maven is very lightweight when we download and install it grabs whatever it needs as plugins.

To create maven project and run in command line –

1. Select the folder
2. mvn archetype:generate -DgroupId=com.bharath -DartifactId=hellomaven -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

archetype:generate = It is a maven goal. It decribes a unit of work in maven. After -D are the parameter to this goal & this goals needs certain information to work group id is like as package name.

artifactId = It is name of our project

archetypeArtifactId= It is show us that what type of our project is creating.

1. Then click the enter button
2. Change the directory to project directory
3. To run maven project use below command –

java -cp target/jar-file name package-name class-name

for eg. java -cp target/hellomaven-1.0-SNAPSHOT.jar com.tejal App

Plugin & Goals-

Maven plugin is a collection of one or more goals.

For eg. archetype(plugin):generate(goals)

Install(plugin):install(goals)

Goals means it is a specific task which we usually run independently or it can be part of bigger build. These goals also takes parameters such as archetype:generate goals have several parameters like as group id , artifact id , etc. & they also have default parameters value such as archetypeArtifactId=maven-archetype-quickstart.

POM XML=

POM means Project Object Model is key to operate Maven. Maven reads pom.xml file to accomplish its configuration and operations. It is an XML file that contains information related to the project and configuration information such as **dependencies**, **source directory**, **plugin**, **goals etc**. used by Maven to build the project.

* **project-**It is the root element of the pom.xml file.
* **modelVersion-**modelversion means what version of the POM model you are using.
* **groupId-**groupId means the id for the project group. It is unique and Most often you will use a group ID which is similar to the root Java package name of the project like we used the groupId com.project.tejal.
* **artifactId-**artifactId used to give name of the project you are building.in our example name of our project is SumOFNumbers.
* **version-**version element contains the version number of the project. If your project has been released in different versions then it is useful to give version of your project.
* **dependencies-**dependencies element is used to defines a list of dependency of project.
* **dependency-**dependency defines a dependency and used inside dependencies tag. Each dependency is described by its groupId, artifactId and version.
* **name-**this element is used to give name to our maven project.
* **scope-**this element used to define scope for this maven project that can be compile, runtime, test, provided system etc.
* **packaging-**packaging element is used to packaging our project to output types like JAR, WAR etc.

Life cycle phases of Maven –

Each life cycle phase is associated with one or more goals.

The process-resources phase is assocaiated with resources:resources.

The complile phse is associated with complier:complier.

The test phase is associated with surefire:test

The package phase is associated with jar:jar

* + process-resources = Copy and process the resources into the destination directory, ready for packaging phase.
  + Compile= Source code compilation is done in this phase.
  + Test= Tests the compiled source code suitable for testing framework.
  + Package = This phase creates the JAR/WAR package as mentioned in the packaging in POM.xml.

Repositories =

Default maven location-

<http://repo.maven.apache.org/maven2/>

The central Maven repository is created Maven community. Maven looks in this central repository for any dependencies needed but not found in your local repository. Maven then downloads these dependencies into your local repository.

If the plugin are not found in central maven repo then we have to use enterprise level repository. In that there are some free products also commercial products. A local repository is a directory on the machine of developer. This repository contains all the dependencies Maven downloads.

Create maven project in Eclipse –

1. File Menu -> select Project
2. Select Maven Project and click Next
3. Select use default workspace location and click Next
4. Select Archetype

For Java standalone project-

Group Id = org.apache.maven.archetypes

Artifact Id = maven-archetype-quickstart

For Web Application project-

Group Id = org.apache.maven.archetypes

Artifact Id = maven-archetype-webapp

1. Then enter the group id, artifact id, version and package.

Group Id – com.tejal

Artifact Id – HelloWorld

Version – version of your project

Package – It same as group id

1. Then click Finish and the project is created.

Maven Commands –

**mvn –version:**This command helps us in knowing the current version of Maven that is installed.

**mvn- package:**This command is used to execute all Maven phases until the package phase. It does the job of compiling, verifying and building the project. It builds the jar file and places it in the specified folder under the specified project.

**mvn clean install:**This maven command helps in executing a clean build life cycle and installs build phase in the default build cycle. This build life cycles may have its build phases and inside each build, there are different build goals. Also, this ensures that the build target is being removed for a new build and adds the clean target.

**mvn compile:**This command is used to compile the source code. It also compiles the classes that are stored at a particular target or class.

**mvn test:**Maven also provides the facility of unit testing particular codes. It runs the tests using suitable testing frameworks.

**mvn deploy:** Maven also has the facility of deploying the code for the project. This deployment is done in an integration or release environment. It copies all final package to the remote repository and it becomes available for sharing with other developers.