**MAVEN**

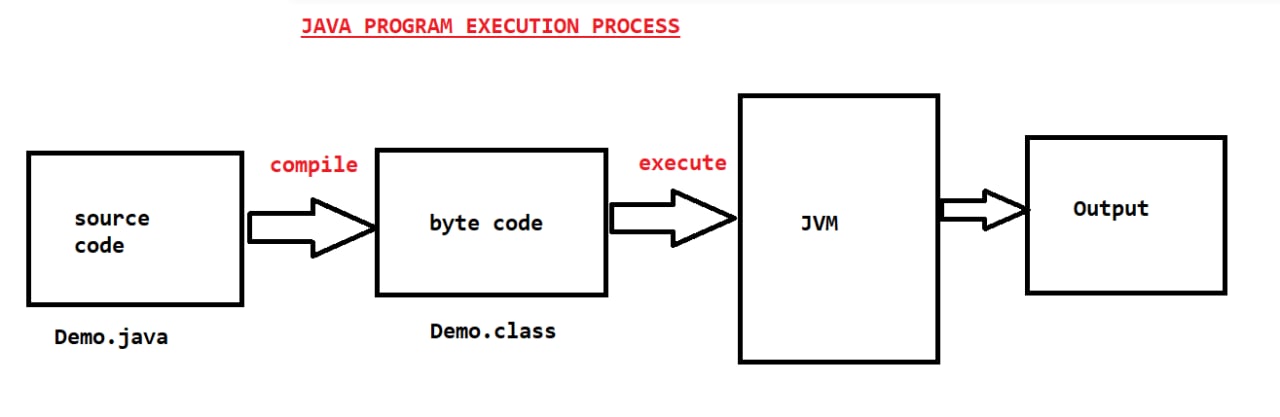
* Maven is a free and open source software given by **Apache Organization**
* Maven s/w is developed using **Java programming language**
* Maven is used to perform **Build Automation** for java projects
* Maven is called as **Java Build Tool**

**JAVA**

* Java is a programming language
* Java language developed by sun Microsystem Company
* Oracle Company acquired Sun Microsystem
* Java is under license of **Oracle Company**
* Java is a high level programming language
* Java is simple programming language
* Java program files will have .java as extension

Ex: Demo.java, Hello.java, Driver.java, Calculator.java etc.

* We can't execute .java files directly
* Java Programs should be converted into Machine understandable format to execute
* Java Programs (.java file) contains source code



* We need to compile java source code into byte code using java compiler (javac)

**Ex: javac Demo.java**

* When we compile java code it will create .class file
* We need to execute .class file to run the java program

**Ex: java Demo**

* When we run java program using java command, JVM will start and it will execute java program

**Note:** JVM stands for Java Virtual Machine

* JVM will convert byte code into machine understandable code
* Java project contains several java programs (.java files)
* We need to compile project source code into byte code
* When we compile project source code we will get .class files
* To deploy java project, we will package all .class files as JAR or WAR file

JAR: Java Archive

WAR: Web archive

**What we can do using maven**

1) We can create **initial project folder structure** using maven

2) We can download "**project dependencies**" using maven

(ex: springboot, hibernate, kafka, redis, email, log4j, junit, security...)

* Java is a free s/w given by sun microsystem
* To develop one java project we will use several frameworks like spring, hibernate etc. along with Java
* We need to download those frameworks and we should add to our java project
* These frameworks we are using in our project are called as our project dependencies
* Instead of we are downloading dependencies, we can tell to maven s/w to download dependencies

**Note:** Required dependencies we will add in **"pom.xml"** file then maven s/w will download them

* pom stands for **project object model**
* When we create maven project then pom.xml file will be created automatically
* **pom.xml** will act as **input file** for maven software

3) We can **compile project source code** using maven

**Note:** Compilation means converting java source code into byte code

Compilation

Demo.java Demo.class

**Note:** java project means collection of java programs.

4) We can package java project as **jar or war file** using maven

JAR - Java Archive

WAR - Web Archive

**Standalone java applications** will be executed as a **jar file.**

**Java Web Applications** will be executed as a **war file.**

**Maven Installation**

1) Download and install Java software

* When we install java we will below 2 things

**a) JDK (Java Development Kit)**

**b) JRE (Java Runtime Environment)**

* JDK contains set of tools to develop java programs
* JRE contains platform/environment which is used to run java programs

**Link To Download Java:** <https://www.oracle.com/in/java/technologies/javase/javase8-archive-downloads.html>

2) Set JAVA\_HOME in Environment Variables (System Environment Variables)

* User Environment Variables: Specific to particular account which logged in our PC
* System Environment Variables: For All Accounts

JAVA\_HOME = C:\Program Files\Java\jdk1.8.0\_202

**Note: Environment Variables will be used by operating system to find out location of software**

3) Set Path for JAVA (Go to System Environment Variables Environment Variables System Variables Select Path and Click on Edit then add JDK path)

Path = C:\Program Files\Java\jdk1.8.0\_202\bin

4) Verify Java installation by executing below command in "**Command Prompt"**

**java -version**

**Note:** It should display java version which we have installed

5) Download Maven software from Apache website

Link to download Maven: https://maven.apache.org/download.cgi

File Name: apache-maven-3.8.5-bin.zip (Binary Archive)

6) Extract Maven Zip file -> Copy extracted maven folder and paste it in "C" drive

7) Set MAVEN\_HOME in System Environment Variables

MAVEN\_HOME = C:\apache-maven-3.8.5

8) Set Path for Maven in System Environment Variables

Path: C:\apache-maven-3.8.5\bin

9) Open Command Prompt and verify Maven Installation using below command

**mvn –version**

**Maven Terminology**

Archetype

Groupid

Artifactid

Packaging

* Archteype represents what type of project we want to create

maven-archetype-**quickstart**: It represents java **standalone application**

maven-archetype-**webapp**: It represents java **web application**

**Note:** Maven providing 1500+ archetypes

* groupId represents company name or project name
* artifactId represents project name or project module name
* packaging represents how we want to package our java application (jar or war)

**Creating standalone application using maven**

1) Create one folder for maven practice

2) Open Command prompt from that folder

3) Execute below command to create maven project

* **mvn archetype:generate -DgroupId=in.ashokit -DartifactId=01-Maven-App -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false**

4) Once project created verify project folder structure

01-Maven-App

- src

- main

-java

- test

-java

- pom.xml

src/main/java: Application source code (.java files)

src/test/java: Application Unit Test code (.java files)

pom.xml: Project Object Model (Maven configuration file)

5) We can add dependencies in pom.xml file

6) We can find maven dependencies in www.mvnrepository.com website

7) Add below dependency in pom.xml file

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>5.2.22.RELEASE</version>

</dependency>

**How maven will download dependencies**

* Maven will download dependencies using repository

In Maven we have 3 types of repositories

1) Central Repository

2) Remote Repository

3) Local Repository

Central repository is maintaining by **apache organization**

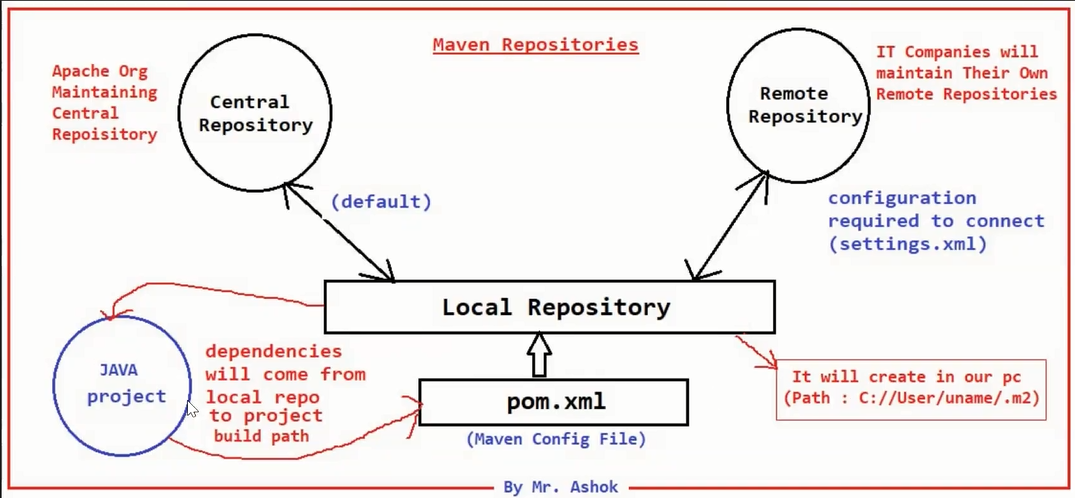
**Every company** will maintain their **own remote repository**

**Local repository** will be created in our system (Location: C://users/<uname>/**.m2**)

* When we add dependency in pom.xml, maven will search for that dependency in local repository. If it is available it will add to project build path.
* If dependency not available in local repository then maven will connect to Central Repository or Remote Repository based on our configuration.

**Note**: By default maven will connect with central repository. If we want to use remote repository then we need to configure remote repository details.

**Note**: Every software company will maintain their own remote repository (Ex: JFrog)



**Configuring Remote Repository (Configure in pom.xml file)**

<repositories>

<repository>

<id>id</id>

<url>jfrong-repo-url/</url>

</repository>

</repositories>

**Maven Goals**

To perform project build activities maven provided several goals for us

1. **Clean**
2. **Compile**
3. **Test**
4. **Package**
5. **Install**

* **Clean goal** is used **to delete target folder**
* Compile goal is used to compile project source code. Compiled code will be stored in target folder

Compile

.java ------------> .class

* **Test goal** is used to **execute unit test code** of our application (junit code)
* **Package goal** is used **to generate jar or war file** for our application based on packaging type available in pom.xml file.

**Note**: jar or war file will be created in target folder.

* **Install goal** is used **to install our project as a dependency** in maven local repository.

**Note**: Every maven goal is associated with maven plugin. When we execute maven goal then respective maven plugin will execute to perform the operation.

**Syntax**: mvn <goal-name>

**Note**: We need to execute maven goals from project folder

Creating web application using maven

* **mvn archetype:generate -DarchetypeArtifactId=maven-archetype-webapp -DgroupId=in.ashokit -DartifactId=01-maven-web-app -DinteractiveMode=false**