==============

MAVEN

==============

=> Maven is a build tool that is used to create the initial project folder structure in order to develop the application.

Ex of build tools:

Java - Maven, Ant, Gradle

Ruby - Rakes

.NET - Nant, MS Build

Python - PyBuilder

=> In the project, developers will write the source code, that source code should be compiled for execution, that compilation of project we will do using MAVEN.

=> Using Maven, we can do project creation, download the dependencies required for the project, unit test execution, and packaging the code.

=> Maven is a free and open-source s/w given by Apache organization.

=> Maven is developed using Java Programming Language.

=> Maven is also called as Java Build Tool.

Uses of Maven:

----

1. Using Maven we can create initial project folder structure.

2. Using Maven we can download the project dependencies.

2.1 Lets say i want to develop a mobile application (BookMyShow) using Java. Here what we need to understand is, just by using Java, we cannot fully develop the project. We might have to use frameworks such as Spring, SpringBoot, RestAPIs, Hybernate, JAP, MicroServices. These additional tools are called as "Dependencies." Earlier developers used to download all these dependencies individually from their respective websites. But now we can use Maven to download all these dependencies required for a project.

2.2. The question is how maven will understand what are the dependencies required for a project?

In maven project, there will be a file called as "pom.xml" file, which is an input file for the maven. Inside the pom.xml file we are going to configure what are the dependencies maven has to download.

2.3. POM - Project Object Model

2.4. Whenever we create a project folder structure using Maven, automatically maven creates the pom.xml file.

2.5. Whatever communication that you want to do with Maven, we have to communicate only through pom.xml file.

3. Using Maven, we can compile the project source code.

3.1. When we are writing a code related to an app. In Java the file will be saved with an extension as ".java" (Demo.java). The code which is available in this file is called as Source Code. Our ultimate goal in writing the java program is to execute it in a computer. But here when i just write a java program will it create an output? NO. Here what we need to do is, this java program should be compiled. Here, whenever we compile the source code, it will generate the byte code in a new file. This new file will have an extension as ".class"

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

3.2. The process of converting the .java file to .class file is called as Compilation. As a devops engineer, we have to do the compilation. Java compiler will do the compilation.

3.3. Whenever we want to execute a java program, that execution shall be done by using a Virtual Machine in Java, which is called as Java Virtual Machine (JVM). This JVM will provide the necessary output as per the source code.

4. Lets assume that my project is having 1000 .java files. Now, compiling all these 1000 .java files will be a troublesome work.

4.1. We can package the java project as a jar or war file using Maven.

jar - Java Archive

war - Web Archive

4.2 Standalone java applications will be executed using jar files.

4.3 Java web application will be executed using war files.

4.4. Once we do the compilation of 1000 .java files, as many as 1000 .class files will get created. Since we cannot give the source code to the client, we will package the .class files as a jar or war file based on the type of application. This jar or war file will be deployed into the webserver (Tomcat). The process of keeping the jar or war file into the web server is known as DEPLOYMENT. Then JVM will execute inorder to get the output.

4.5. Compiling the source code and packaging it is called as Build Activity.

4.6. Standalone java applications - The s/w which is specific to one computer is called as "Standalone Applications." - jar files.

4.7. Java web applications - war file.

==

MAVEN

==

=> MAVEN is developed by using Java Programming Language.

=> If you want to run/work with Maven, we need to install the Java s/w.

=> Once the Java s/w is installed, java provides us two things;

1. JDK - Java Development Kit
2. JRE - Java Runtime Environment

=> JDK contains set of tools to develop java programs.

=> JRE contains a platform/environment which is used to run the java programs.

Note: JDK and JRE folders will be available with Java 8 version only.

=> If you are a developer, you have to install both JDK and JRE.

=> In client system, only JRE is required.

Download & Install Java Software:

~ In order to download the Java Software, we need to create an account in Oracle Website.

~ Weblink: https://www.oracle.com/java/technologies/downloads/#jdk21-windows

Configuration of Java Software:

~ After installation of Java Software, we have to set JAVA_HOME in environment variables.

~ The variables which are used by our PC are called environment variables.

~ Two types of environment variables: 1. User Variables 2. System Variables

Setting up the JAVA_HOME page for system variables:

~ C Drive - Program Files - Java - jdk - Copy the jdk folder path - In the search box of PC, type "Environment" and select 'Edit System Environment Varibles' - A dialogue box will open - Advanced - Environment Variables - A dialogue box will open - System Variables - New - A dialogue box will open - Variable Name: JAVA_HOME, Variable Value: <Paste the path of jdk folder> - OK - OK - OK.

Setting up the path for Java:

~ C Drive - Program Files - Java - jdk - bin - Copy the bin folder path - In the search box of PC, type Environment and select 'Edit System Environment Varibles' - A dialogue box will open - Advanced - Environment Variables - A dialogue box will open - System Variables - Select the "path" - Edit - New - Paste the path - OK - OK - OK.

How to check whether java is working or not:

~ Open Command Prompt ----> java -version ----> You should see the Java Version Number/Id

How to download Maven from Apache Website:

~ Google ----> Download apache maven (https://maven.apache.org/download.cgi) ----> "Binary Zip Archive" and click on the link next to it ----> The file gets downloaded in the Zip format ----> Extract the zip file ----> Copy the folder ----> Paste the folder in the C Drive

----

Configuration of Maven Software:

Setting up the MAVEN_HOME in system environment variables:

~ C Drive - Paste the extracted zip folder in C Drive - Apache Maven folder - Copy the folder path - In the search box of PC, type Environment and select 'Edit System Environment Varibles' - A dialogue box will open - Advanced - Environment Variables - A dialogue box will open - System Variables - New - Variable Name: MAVEN_HOME, Variable Value: Paste the path of maven folder - OK - OK - OK.

Setting up the path for Maven:

~ C Drive - Apache folder - Bin Folder - Copy the bin folder path - In the search box of PC, type Environment and select 'Edit System Environment Varibles' - A dialogue box will open - Advanced- Environment Variables - A dialogue box will open - System Variables - Select the "path" - Edit - New - Paste the path - OK - OK - OK.

How to check whether Maven is working or not:

~ Open Command Prompt ----> mvn -version ----> You should see the Maven Version Number/Id

==

MAVEN

==

Maven Terminology:

Ex: IBM ---> Project (Hotstar) ---> Java ---> Code ---> hotstar.java (Source Code) ---> Compilation (Java Compiler) ---> hotstar.class ---> JVM ---> Output

=> Archetype: It represents what type of project we are going to create.

It refers to a project template that you can use to quickly set up new projects. It essentially acts as a blueprint containing a basic structure and configuration for a specific type of project.

1) maven-archetype-quickstart ---> It represents the standalone application creation.

2) maven-archetype-webapp ---> It represents the web application creation.

There are 100+ types of architypes

=> group ID: It represents the company name or a project name.

Hotstar

Ex: IBM-Hotstar

=> Artifact ID: It represents the project name or project module name.

Hotstar (movies, cricket,...) ---> Movies, Cricket...

DevOps ---> Introduction to DevOps, Linux, Maven....

An artifact refers to any file that can be downloaded, installed, or deployed during the build process. It's essentially a piece of code or resource that your project either uses or produces.

PhonePe ---> Transactions, Recharge, Tickets, Bills...

Ex: PhonePe-Transactions

=> Packaging: It represents how you want to package your application i.e either war file or jar file.

Standalone App ---> Jar file (multiple .class files will be available)

Web App ---> War file (multiple .class files will be available)

.java file ----> .class file ----> Execute (JVM) ----> Output

1000 .java files ----> Package (jar or war) ----> Execute (JVM) ----> Output

----

How to create a standalone application using Maven

```
mvn archetype:generate -DgroupId=in.StarAgile -DartifactId=01-Maven-SA-App -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

How to create a web application using Maven

```
mvn archetype:generate -DarchetypeArtifactId=maven-archetype-webapp -DgroupId=in.StarAgile -DartifactId=02-Maven-WebApp -DinteractiveMode=false
```

How to create initial project folder structure for standalone application

----

D Drive ---> Create a folder (01-Standalone-App) ---> Open Command Prompt in the created folder path ---> Execute the below command

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

=> Once the project folder structure is created, we will see two things

1. SRC Folder

   1.1. Main Folder - java\in\StarAgile\App.java ---> In ".java" file app source code will be written.

   1.2. Test Folder - java\in\StarAgile\AppTest.java ---> In "AppTest.java" file, code testing (J-Unit Test) related info.

2. pom.xml File

   Without pom.xml file maven will not work. Whatever work we are going to do in maven, everything will be communicated using pom.xml file only.

Note: Whenever we are executing maven goals we should execute all the maven goals in the pom.xml file path only.

How to add dependencies in the pom.xml file

----

https://mvnrepository.com/ ---> Search for required dependency (Spring) ----> Copy the syntax and paste in the pom.xml file. ----> Save the file and exit.

Maven will download the dependencies added in the pom.xml.

How maven will download these dependencies in the backend?

MAVEN REPOSITORIES

=> Maven will download any dependency using a concept called repository.

=> In Maven, we have 3 types of repositories:

1. Central Repo. - Apache Organisation
2. Remote Repo. - Individual company (JFROG)
3. Local Repo. - Our PC (.m2 folder)

By

Kastro Kiran V

LinkedIn: https://www.linkedin.com/in/kastro-kiran/

YouTube: https://www.youtube.com/playlist?list=PLs-PsDpuAuTdOcZa-DDgG8KRbtMI_XRrC