## **MAVEN**

# Why Maven?

## When we create normal java project, we need all kinds of library to work on our projects.

## So, we download each library from different websites in form of jar files and then we attach them to our projects.

## Suppose tmrw if I want to upgrade all libraries to latest version then I have to remove the existing version of libraries from project and again have to download and again have to add to our project.

## Now there comes maven.

## Prerequisite for Maven is Java

## Maven is build tool. Whenever u create maven project by default it will provide pom.xml file

## Whole maven project is controlled by pom.xml file.

## By default, maven gives project structure when u create it.

## Generate reports.

## Generate project related documentation.

## We can package the project and submit to client.

# Two types of entries we have in pom.xml

## Dependencies

## Plugins

# Dependencies

## To maintain the third-party library, which will automatically download third party library and then if I want to upgrade the third-party library just change the version number in dependency it will upgrade automatically.

## Dependencies are responsible for downloading the third-party library and drivers which are related to our maven project.

## Maven project internally create maven local repository. As soon as if we try to download one dependency it will automatically download all third-party libraries related to that one dependency as jars from remote maven repository and store it in maven local repository.

## All the libraries are available inside the remote maven repository. It will automatically download from there.

# mvnrepository.com

# Plugins:

## Contains project related configurations (surefire plugin, compile plugin) are specified.

## Suppose I want to compile the project we need compiler plugin.

## Suppose I want to execute whole project then I need surefire plugin.

## Suppose I want to package my project then I need some plugin

## Suppose I want to start and stop the server every time as per my web application then I need one plugin.

## By default, eclipse will come with the maven software you don’t need to install maven separately.

# Creating Maven Project in eclipse

## Maven Archetype is basically a template.

## If web applications, then take one type of template or spring boot then take another type of template.

## **Group ID**: Company Name(com.accy)

## **Artifact ID**: Project Name(Myproject)

## When u want to package then select any option like jar, pom or war (default is jar)

## src/main/java – contains development source code.

## src/main/resources – contains third party source code file(config.properties ->data required to develop the application)

## src/test/java – unit test code, test code

## src/test/resources – test resource code

## src and target used to store temporary files during runtime by maven.

## POM.xml – Project Object Model

## Scope in dependency: Till wat phase its applicable

# Maven follows Maven Build Life Cycle

## Validate the code.

## Compile the code.

## Test the code.

## Package the code - Automatically whole project will be packaged and created as jar file in default and that jar file you can share it with another project or team members

## Integration test – sometimes this project is integrated with another project (parent project)

## Verify code.

## Install –create jar file for whole project and will be available in local repository.

## Deploy – install jar in remote repository to use by other projects.

# When Run as-> Maven Test

## It will validate the project, compile and test the project(whatever phase u r executing it will execute the previous available phases also)

# Goals:

# What u exactly want to do

## Clean-> click Run (Project will be cleaned. Unnecessary files will be cleared) using clean-plugin (default plugin)

## Run as-> Maven Clean

## Run as -> Maven Compile -> compiles the project(default). Uses resource plugin(default) and compile plugin.

# To create xml file in maven:

## Right click on project package->TestNg-> Convert to Testng

## It will automatically create xml file.

# Sure-File Plugin

## Maven project reports will be inside sure-file reports ->target folder.

## When u run pom.xml file it will execute only first TC remaining test cases will not be executed which is a problem in maven

## So, to overcome that we must add maven surefire plugin in pom.xml to execute all TestNG test cases.

## We must tell pom.xml to execute testng.xml file. For that we have to add suitexmlfile under configuration tag.

## We can also specify multiple xml file under the suitexml files to run using pom.xml

# To run Maven Commands from CLI then maven software is required to be installed in our OS

# Two environmental variable we have to create:

## Maven\_Home – maven installed path

## M2\_Maven- maven installed path

# To check version:

# mvn -version

## It gives internally what java is using and also JRE details and maven version

# How to run maven project from CLI

## Get the location of project.

## Cd project location

## Maven commands always look for pom.xml so wherever pom.xml is located we must execute our command from that location.

# mvn validate

## Validate the project is correct and all necessary information is available in the project.

# mvn clean

## Remove all files generated by previous build. All temporary files will be cleaned up from target folder.

# mvn compile

## Compiles the source code of the project.

# mvn test

## validate, compiles all the test and then comes to the test phase and execute the test using surefire plugin.

# mvn package

## Create a jar file of our whole project. Execute everything like validate, compile, test and finally package(create) whole project into jar file.

# mvn install

## Clean, compile, test, package(create) the project into jar file and install this jar file into local repo.

# mvn deploy

## Clean, compile, test, package(create) the project into jar file and install this jar file into local repo and finally copy the jar file into remote repo(we have to specify configuration to deploy).

# Execution Phase:

# Run with Maven pom.xml within eclipse

## We have to add 2 plugin (maven -compiler-plugin and sure-file-plugin) under build and under plugin tag

## maven -compiler-plugin compiling all code if there are any issue.

## sure-file-plugin - running ur testng xml file.

## Click pom.xml and Run as Maven test.

## First time u wil get error to fix that go to window-> preferences->Add ->Standard VM->jdk path->select jdk .

# Run thr Maven CLI(without eclipse)

## Install maven software separately based on OS.

## Add Environment variables for maven and Java home not to get the error.

## mvn -version

## Get the location of project.

## Give path for current project directory using “cd project location”.

## mvn clean install

## clean command – clean up all execution happens previously.

## install command – start execution of complete project.

## Success execution – Build Success

## Failed execution – Build Failure

# Run using run.bat(instead of manually typing the commands in CLI we can use run.bat to store the commands)

## In project folder create bat (executable file) file(run.bat)

## Open bat file and give the commands (project directory command and maven clean install)

## Double click on bat file and it will execute our test cases.

# Run using Jenkins

# Download Jenkins

# Configure Jenkins:

## URL [**http://localhost:8080**](http://localhost:8080)( in project Devops wil share the URL)

## To unlock Jenkins in Jenkins secret folder you will found the password

## Install suggested plugins.

## Create admin users (in Jenkins wat are the project u r responsible for u can see only that DevOps team will give access only to that)

## Save and finish.

## We are using maven project in eclipse so install maven plugin in Jenkins.

## For that go to manage Jenkins (all plugins we can install from this, and all configurations can be done) and manage plugins.

## Select all maven related plugins.

## Click install without restart.

## Run automation thru Jenkins.