MAVEN

- Maven is an automation project management tool.
- It is used to build the code
- once we build the code we will get JAR/WAR/EAR
- Maven is used to add the dependencies to our application.
- Maven is based on POM.xml

POM: Project Object Model

XML: Extensible Markup Language.

- POM.xml contains project related data (metadata, kind of project, kind of output, description, dependencies).
- Maven was developed by Apache software foundations.
- Maven was released in 2004.
- Maven can build any number of projects into desired Output such as .jar, .war and .ear

- .jar = java archive file
- .war = web archive file
- .EAR = enterprise archive

- It is mostly used for java-based projects.
- It was initially released on 13 July 2004.
- Maven is written in java.

MAVEN BUILD LIFE CYCLE:

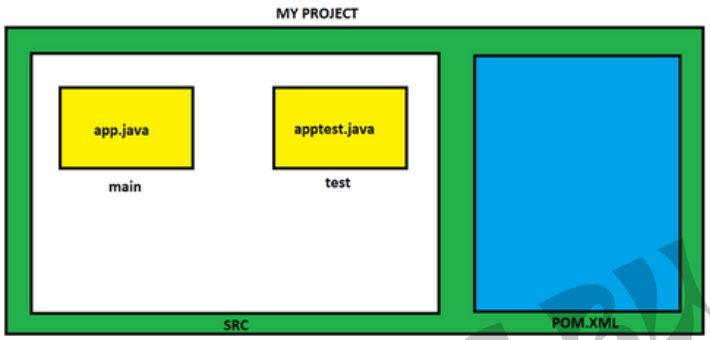
1 Generate Resource
2. Compile Code
3. Unit Test
4. Package (build)
5. install (into Local repo or Artifactory)
6. Deploy (to servers)
7. Clean (to delete all the runtime files)

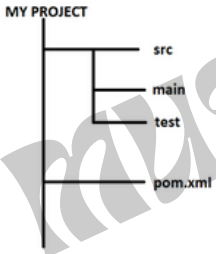
BUILD TOOL:

- it is used to set up everything which is required to run your java code This can be applied to your entire java project.
- It generates source code, compiling code, packaging code to a jar etc.
- POM refers the XML file that have all information regarding project and configuration details
- Main configuration file is in pom.xml.
- It has description of the Project details regarding version and configuration management.
- The XML file is in the Project home directory.

DEVOPS BY MUSTAFA

MAVEN DIRECTORY STRUCTURE:





step-1: download file from dlcdn.apache.org
wget https://dlcdn.apache.org/maven/maven-3/3.8.7/binaries/apache-maven-3.8.7-bin.tar.gz
wget ---> web get : used to download files from browser to server.

step-2: untar the file
tar -zxvf file_name

step-3: install java-1.8.0 version
yum install java-1.8.0-openjdk -y
to check the version: java -version

step-4: install maven
yum install maven -y
to check the version: mvn -v or mvn -version

step-5: generate source code
mvn archetype:generate
Choose a number or apply filter (format: [groupId:]artifactId, case sensitive contains): 1997: (enter)
Choose org.apache.maven.archetypes:maven-archetype-quickstart version: (select any number or keep enter)
Define value for property 'groupId': **DEVOPS**
Define value for property 'artifactId': **BATCH37**
Define value for property 'version' 1.0-SNAPSHOT: : (enter)
Define value for property 'package' DEVOPS: : (enter)

step-6: to compile the code
mvn compile
whenever we compile, target folder will gets created and we can see output on target folder,

step-7: test the source code
mvn test
once the test is done, some test folders will gets created in target folder

step-8: build the code
mvn package
once build is done, jar file will be created. (artifact_id-version-SNAPSHOT.JAR)

step-9: install the code
mvn install

step-10: clean the run time files (executable files)
mvn clean
delets all the target folder

to update java version: update-alternatives --config java

ANT VS MAVEN

Ant	Maven
Ant **doesn't has formal conventions**, so we need to provide information of the project structure in build.xml file.	Maven **has a convention** to place source code, compiled code etc. So we don't need to provide information about the project structure in pom.xml file.
Ant is **procedural**, you need to provide information about what to do and when to do through code. You need to provide order.	Maven is **declarative**, everything you define in the pom.xml file.
There is **no life cycle** in Ant.	There is **life cycle** in Maven.
It is **a tool** box.	It is **a framework**.
It is **mainly a build tool**.	It is **mainly a project management tool**.
The ant scripts are **not reusable**.	The maven plugins are **reusable**.
It is **less preferred** than Maven.	It is **more preferred** than Ant.