Chapter - 5 Build Tool Maven 7/03/24

Q What is Maven?

Ans Apache Maven is a Software project management and ~~compth~~ comprehension tool. Based on the concept of a Project Object Model (POM), Maven can manage a projects build reporting and documentation from a central piece of information. primarily used for Java Projects.

Q What Maven can do?

- Model our Software project..
- Centralize project information.
- Manage our build process.
- Gather data about our Software project and the build itself.
- Document the Software and our projec

* Advantages of Maven over Ant

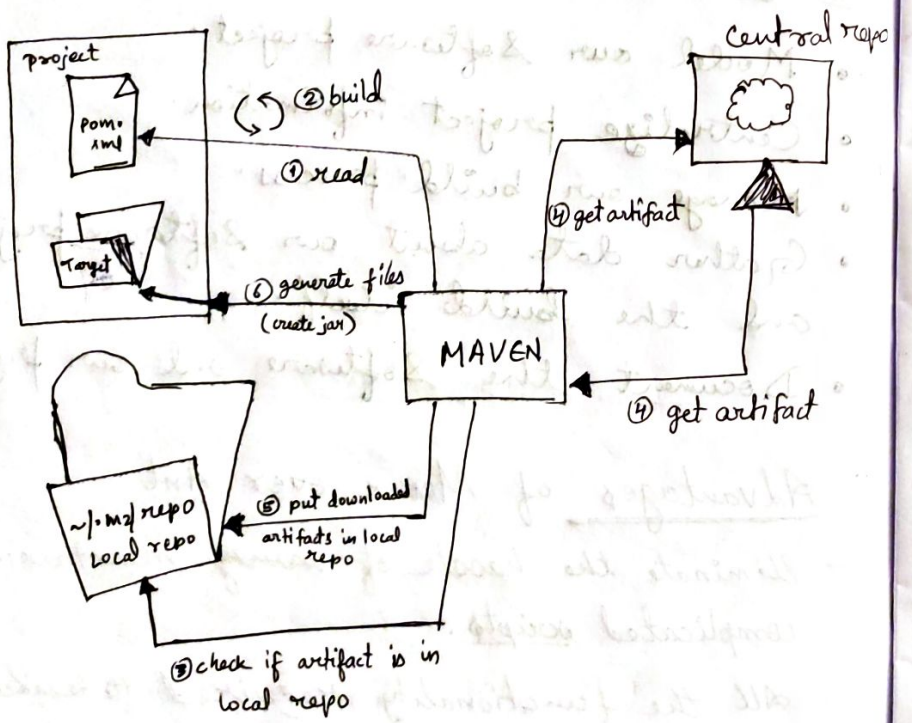
→ Eliminate the hassle of ~~manag~~ maintaining complicated scripts.

→ All the functionality required to build our project that is clean, compile, copy resources, install, deploy etc is built right into the Maven.

→ cross project reuse - and has no convenient way to reuse target across projects, but Maven does provide this functionality.

→ Logic & looping Maven provides conditional logic, looping constructs and reuse mechanisms which were missing in Ant.

How Maven Works? Img



Theory in next pages. (+3 pages)

Terms related to ~~Maven~~ Maven

① POM :

→ Project Object Model.

→ fundamental unit of work in ~~Maven~~ Maven

→ It is a XML file that contains information about the project & configuration details used by Maven to build the project.

→ project.xml in Maven 1

→ pom.xml in Maven 2 and 3 onwards.

→ Project Type (Packaging - JAR, WAR, EAR)

→ Minimal pom.xml

```
<project>
    <modelVersion> 4.0.0 </modelVersion>
    <groupId> com.mycompany.app </groupId>
    <artifactId> my-app </artifactId>
    <version> 1 </version>
</project>
```

② Archetype :

- In general ⇒ a primitive type or an original model or pattern from which all other things of same kind are made.
- for Maven : A template of a project which is combined with some user inputs to produce a working Maven project that has been tailored to the user's reqmts.

③ Plugins :

→ A plugin provides a set of goals that can be executed to perform a task.

→ There are Maven plugins for building, testing, source control management, running a web server, generating Eclipse project files and much more.

→ Some basic plugins are included in every project by default, & and they have sensible default settings

④ Goals :

→ A goal is like an Ant target

→ We can write our own plugins with goals, or use those that are provided by Maven

→ For Eg: A Java project can be compiled with the compiler-plugins compile goal by running `mvn compiler:compile`

⑤ Artifact :

→ It is a unique identifier or a simple name given to a module within a group.

→ There can be multiple modules in a group.

Eg - If a group has a module called webapp, its artifact id will be "webapp".

⇒ GroupId :- name of the company.
Eg - com.example.app

⇒ ArtifactId ⇒ Project Name

⑥ Version :

→ It is an identifier for the release or build number of the project.

→ Two types of version :

⇒ Releases : A version that is assumed never to change.

⇒ Snapshot : used by projects during development & it implies that development is still occuring & project may change.

*** Build Life Cycle 10 marks

→ In Maven, the build is run using a predefined ordered set of steps called the build life-cycle.

→ The indivisual steps are called phases, and the same phases are run for ~~or every phase~~ every maven build using default life-cycle, no matter what it will produce

→ The build tasks that will perform during each phase are determined by the configuration in the project file, and in particular selected packaging.

→ Some of the most commonly used & life cycle phases in the default life-cycle are:-

1) ⇒ validate — checks build prerequisites

2) ⇒ compile — compiles the source code identified for the build.

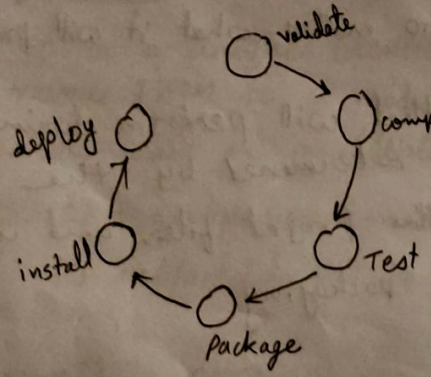
3) ⇒ Test — runs a unit tests for the compiled code.

4) ⇒ package — assembles the compiled code into a binary build result & like jar file

~~5) ⇒ install — shares the build with other projects on the same machine.~~

6) ⇒ deploy — publishes a the build into a remote repository for other projects to use.

5) install — The package is installed into local repository, as making it available as a dependency for other projects locally.



9/03/024

Q. What is Maven Repository ? & its types ?

In Maven, a repository is a directory where all the project jars, library jar, plugins or any other project specific artifacts are stored and can be easily used by Maven.

There are 2 types of repositories:-

① Local repo ② Remote repo ~~③ Central repo~~

① Local Repository :- .m2

→ This repository is stored on the local machine.

→ When we run a Maven build, it downloads required dependencies from central repository and stores them in our local repository.

→ Default location of Local Repository :-
'${user.home}/.m2/repository'.

② Remote Repository (Global)

→ This repository is located on a remote server. Maven can download ~~these~~ dependencies from these repositories when needed.

→ Central repository is the default remote repository for Maven, which contains a large number of commonly used libraries.

→ Other remote repositories can also be configured in the "pom.xml" file to download dependencies from them.

Ugad

Q. How does Maven works?

Maven is a build automation tool primarily used for Java projects. It simplifies the build process by providing a standard way to build and manage projects, including dependencies, build lifecycle and project structure. Here is how Maven works:

(i) Project Object Model (POM):
Maven uses POM file usually named as pom.xml that contains project configurations and settings.

(ii) Dependencies and repositories

dependencies: These are external java libraries required for our projects. Maven manages them ~~from~~ by downloading jar files from repositories.

repositories: These are directories containing packaged JAR files. Local repo resides on the machine's Hard Drive. If Maven doesn't find dependencies locally, it fetches them from a central Maven repository.

ii) Build lifecycle :

Maven defines a set of build phases (eg. compile, test, package, install, deploy) that represents the sequence of steps to build a project. Each phase is associated with a set of goals, which are executed to achieve that phase's tasks.

iv) Build profiles :-

Build profiles allows to build the project with different configurations.

For eg - We might need profiles for local development, testing or production.

v) Build plugins :-

⇒ plugins perform specific tasks during the build process. We can add POM plugins to our POM file.

Overall, Maven simplifies the build process and promotes best practices by providing a standardized way to manage projects & their dependencies.

diagram here.

MAVEN Installation

→ download from official website.
→ unzip
→ create environment variable by pasting the directory (till "\bin" depth).

Creating a Maven project

→ open cmd

→ type the following and press Enter :

```
mvn archetype:generate -DgroupId=com.example.app
-DartifactId=helloworld -DarchetypeArtifactId=maven-archetype
-quickstart -DarchetypeVersion=1.4 -DinteractiveMode=false
```

default project creation command.

Maven Dependencies

Eg. Junit ⇒ used for testing the app.

Maven Plugins

They enhance Maven's capabilities and perform specific tasks during build process.

Eg - Compiler Plugin
Jar Plugin
Javadoc Plugin
Release plugin
deploy plugin