```
=============
Apache Maven
=============

=> Maven is a build tool

=> Maven is free & open source

=> Maven is developed using java language

Note: Maven is used only for java projects build process

=> It is used to automate project build process

         a) Download required libraries/dependencies

                        Ex: spring, junit, log4j, kafka, redis ....

     b) Compile source code

     c) Execute Unit test cases

     d) package application as jar or war


Note: The main aim of maven is to simplify java projects build process.

==============
Maven Setup
==============

### Reference Video : https://www.youtube.com/watch?v=hV1OWzYpzxo

Step-1 : Install Java + Setup JAVA_HOME + Setup Java Path

                JAVA_HOME = C:\Program Files\Java\jdk-17

                Path = C:\Program Files\Java\jdk-17\bin

Step-2 : Download maven software as zip file & extract it

                URL : https://maven.apache.org/download.cgi

Step-3 : Copy maven folder into C drive (optional)

Step-4 : Setup MAVEN_HOME + setup MAVEN PATH

                        MAVEN_HOME = C:\apache-maven-3.9.6

                        Path : C:\apache-maven-3.9.6\bin

Step-5 : Verify maven installation in cmd

                        $ mvn -v


==================
Maven Terminology
==================

1) Archetype : Type of project (quick-start / web)

2) groupId : Organization Name (in.ashokit)
```

```
3) artifactId : Project name

4) version : SNAPSHOT / RELEASE

5) packaging : jar or war

6) dependencies : libraries (jars)

7) Repositories : dependencies location
                                             - central / Remote / Local

8) goals : To perform maven build process
                                 - clean
                                 - compile
                                 - test
                                 - package

===============================
Maven Project Creation in CLI
===============================

# synatax to create mvn project using cmd

mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -DarchetypeArtifactId=maven-
archetype-quickstart -DarchetypeVersion=1.4 -DinteractiveMode=false


# navigate into project directory

cd my-app

# execute maven goals

mvn clean
mvn compile
mvn test
mvn package
mvn install

mvn clean package => clean + compile + test + package

mvn clean install => clean + compile + test + package + install


===============================
Maven Project Creation in IDE
===============================

=> By default maven is integrated with all java based IDEs

        Ex: Eclipse, STS, IntelliJ IDEA...

=> We can create maven project directley in above IDEs

                File -> New -> Maven -> Maven Project

=> We can execute maven goals from IDE directley

                Right Click on project -> Run as -> Maven Build

=======================================================
Q) Can we customize project build jar or war name ?
=======================================================
=> Yes we can do that by configuring <finalName> in pom.xml file

 <build>
          <finalName>app</finalName>
  </build>

=> If we don't configure final name then it will take name like below

                artifactId + version.jar/war

====================
Maven Dependencies
====================

=> Libraries required to develop our java applications are called as Maven dependencies

                a) spring-core
                b) spring-jdbc
                c) spring-boot-starter
                d) jackson
                e) junit
                f) log4j

=> We need to add required dependencies in maven project pom.xml file.

=> When we add dependencies in pom.xml file then maven will download those dependencies and will add
to project build path.

=> Maven will take care of "transitive-dependency" management.

                        spring-context => core + beans + aop + jcl

                        spring-jdbc ==> spring-jdbc + spring-tx


=> Maven dependencies we can find in below website

                        url : www.mvnrepository.com


<dependencies>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>6.1.4</version>
                </dependency>
</dependencies>

=> If we want remove some child dependencies then we need to use dependency exclusion concept.

                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>6.1.4</version>
                        <exclusions>
                                <exclusion>
                                        <groupId>org.springframework</groupId>
                                        <artifactId>spring-aop</artifactId>
                                </exclusion>
                        </exclusions>
                </dependency>


======================
Maven Repositories
```

```
======================

=> Mvn Repository is a location where maven dependencies/libraries will be stored.

=> Maven tool will deal with 3 types of repositories

                1) Central Repository (public)

                2) Remote repository (private -> company specific)
                                                     (Nexus/Jfrog)

                3) Local Repository (in our machine)


=> Local Repository will be available in our machine

                        Path : C://Users/<name>/.m2

=> Central Repository will be maintained by Apache org. It is public repository.

=> Remote Repositories are called as private repositories.      These are project/company specific.

                Ex: Nexus repo & JFrog Repo.


=> When we add dependency in pom.xml file, maven will search for it in local repo. If not available
then it will search in central repo (it will download from central to local).


Note: We need to modify maven settings to connect with remote repository.

===========================
Maven Dependency Scope
===========================

=> Depenendency scope will decide when maven should load that dependency into our application.

compile
runtime
test
provided
system
import

===========================
Maven Multi Module Project
===========================

=> Creating project with parent-child relation.

=> Create Maven project with packaging type as 'pom'. It will act as parent.

=> Create child project as maven module.

                        Flipkart-App    (Parent)

                                - admin   (module-1)
                                - reports (module-2)

Note: When we execute maven goals parent project then it will reflect on modules also.


========
Summary
========
1) What is Build Tool
2) What is Build Process
3) Maven Introduction
4) Maven Setup in Windows
5) Creating Maven Projects
6) Maven Project Folder Structure
7) Maven Terminology
8) Maven Dependencies
9) Maven Repositories
                - local
                - central
                - remote (nexus/jfrog)

10) Maven Goals
11) Depenency Scopes
12) Maven Multi Module
```