```
============
Build Tools
============

=> Build tools are used to automate project build process

======================
Project Build Process
======================

=> Download required libraries (Ex: Spring, Hibernate, Junit, Log4j...)

=> Add Libraries to build path

=> Compile source code

=> Execute Unit test cases

=> Package project as jar / war file for deployment/execution


######### We can automate above steps using Build Tools ##########

=> We have several build tools in market for Java

1) Ant (Outdated)
2) Maven
3) Gradle

=================
What is Maven ?
=================

=> Maven is a free & open source software

=> Maven s/w developed by Apache Organization

=> Maven s/w developed by using Java Language

Note: To run maven s/w java is mandatory in our system

=> Maven is used to automate Java projects build process

====================
What Maven can do ?
====================

1) Create Project Folder Structure
2) Download Required libraries/dependencies
3) Add libraries to build path
4) Compile source code
5) Execute JUnit test cases
6) Package project as jar / war


==============================
Maven Setup In Windows Machine
==============================

1) Download & Install Java software

2) Set JAVA_HOME

3) Set Java Path
```

4) Verify Java Setup

5) Download & Extract Maven software

6) Set MAVEN_HOME

7) Set Maven Path

8) Verify Maven setup

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

Archetype : It represents Project template

quick-start : Stand-alone Application template
web-app : Web Application template

groupId : It represents organization/company name

Ex: com.tcs, com.infy, in.ashokit...

artifactId: It represents project name

Ex: insurance_app, medical_app, sbi_app...

version : It represents project version number

SNAPSHOT -> Project Under Development

RELEASE -> Project development completed

Maven Dependencies : The libraries required for project development

Ex: Spring, Hibernate, JUnit, ojdbc, mysql-connector, Security .....

Maven Repositories : Repository will maintain will maintain dependencies/libraries

Ex: Central Repo, Remote Repo & Local Repo

Maven Goals : Goals are used to perform project build process

Ex: clean, compile, test, package, install etc...

pom.xml : Project Object Model (Maven Configuration File)

==========
Creating Maven Project in CLI
==========

stand-alone project creation

```
mvn archetype:generate -DgroupId=in.ashokit -DartifactId=sbi_app -DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4 -DinteractiveMode=false
```

Web-app creation

```
mvn archetype:generate -DgroupId=in.ashokit -DartifactId=hdfc_web_app -DarchetypeArtifactId=maven-archetype-webapp -DarchetypeVersion=1.4 -DinteractiveMode=false
```

```
============
Maven Goals
============

clean : To delete target folder

compile : To compile source code (it will store .class files in target)

test : To execute Junit test cases

        test = compile + test

package : To package project as jar / war (it will store in target)

        package = compile + test + package

install : To store our project packaged file into maven repo

        install = compile + test + package + install

#### Note: To perform project build process we can use 'mvn clean install' #####



================================
Maven Project Folder Structure
================================

src/main/java : We can create application source files (.java)

src/main/resources : We can keep project config files (.yml, .properties, .xml)

src/test/java : We will keep Junit classes (Unit test)

src/test/resources : We can keep unit test config files (.yml, .properties, .xml)

target : .class files + packaged files will be stored

pom.xml : Maven Configuration (dependencies)


=========================================
How to add dependencies in Maven Project ?
=========================================

=> Dependencies means the libraries required for the project development

        Ex: Spring, hibernate, Junit, Ojdbc14, mysql-connector etc....

=> We can find maven dependencies in below website

                Website : www.mvnrepository.com

        <dependencies>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>6.0.11</version>
                </dependency>
        </dependencies>

=> When we have above dependency in pom.xml then maven will download all required jars
like spring-core, spring-beans, spring-jcl, spring-aop from cental repository.
```

Note: This is called as Transitive Dependency Management.

Advantages : We no need to add all dependencies (maven will download all required jars)

Dis-Advantages: Unwanted jars will be loaded to JVM (memory wastage)

=> To remove un-wanted jars we can use 'Maven Dependency Exclusion' concept.

```
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-context</artifactId>
        <version>6.0.11</version>

        <exclusions>
                <exclusion>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-aop</artifactId>
                </exclusion>
        </exclusions>

</dependency>
```

==========
Maven Repositories
==========

=> We have 3 types of repositories in Maven

a) Central Repo
b) Remote Repo
c) Local Repo

=> Central Repo is maintaing by Apache Org

=> Every company will maintain their own Remote Repo (Nexus/JFrog) for shared libraries (libraries specific to company projects)

=> Local Repo will be created in our machine (.m2)

Note: When we add dependency in pom.xml first maven will check in local repo if not available then maven will download from central repo / remote repo to local repo. Maven will given dependencies from local repo to project build path.

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

=> In realtime application several modules will be available like below

a) Products List
b) Payment
c) Orders
d) Reports
e) Admin

=> We can develop such kind of project modules development using Maven Multi Module concept.

=> Maven Multi Module supports Parent & Child relationship

=> When we build parent project then all its child modules gets build automatically.

Note: Parent Project Packaging type should be pom. Once parent project got created we can create child projects as Modules.

1) What is Project Build Process
2) Purpose of Build Tools
3) What is Maven
4) Maven Setup in Windows
5) Maven Terminology
6) Maven Project Creation
7) Maven Goals Execution
8) Adding maven dependencies
9) Dependency Exclusion
10) Maven Repositories
11) Maven Multi Module Project
12) Working with Maven in STS IDE