POM stands for Project Object Model.

The POM contains information about the project and various configuration detail used by Maven to build the project(s).

Some of the configuration that can be specified in the POM are following −

* project dependencies
* plugins
* goals
* build profiles
* project version
* developers
* mailing list

<project xmlns = "http://maven.apache.org/POM/4.0.0"

xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation = "http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.companyname.project-group</groupId>

<artifactId>project</artifactId>

<version>1.0</version>

</project>

**groupId**

This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects.

**artifactId**

This is an Id of the project. This is generally name of the project.

**version**

This is the version of the project

What is Build Lifecycle?

A Build Lifecycle is a well-defined sequence of phases, which define the order in which the goals are to be executed

|  |  |  |
| --- | --- | --- |
| **Phase** | **Handles** | **Description** |
| prepare-resources | resource copying | Resource copying can be customized in this phase. |
| validate | Validating the information | Validates if the project is correct and if all necessary information is available. |
| compile | compilation | Source code compilation is done in this phase. |
| Test | Testing | Tests the compiled source code suitable for testing framework. |
| package | packaging | This phase creates the JAR/WAR package as mentioned in the packaging in POM.xml. |
| install | installation | This phase installs the package in local/remote maven repository. |
| Deploy | Deploying | Copies the final package to the remote repository. |

### mvn compiler:compile

This command compiles the java source classes of the maven project.

### mvn compiler:testCompile

This command compiles the test classes of the maven project.

### mvn package

This command builds the maven project and packages them into a JAR, WAR, etc

### mvn install

This command builds the maven project and installs the project files (JAR, WAR, pom.xml, etc) to the local repository.

### mvn deploy

This command is used to deploy the artifact to the remote repository

### mvn validate

This command validates the maven project that everything is correct and all the necessary information is available

### mvn dependency:tree

This command generates the dependency tree of the maven project.

### mvn dependency:analyze

This command analyzes the maven project to identify the unused declared and used undeclared dependencies. It’s useful in reducing the build size by identifying the unused dependencies and then remove it from the pom.xml file.

### mvn archetype:generate

Maven archetypes is a maven project templating toolkit. We can use this command to generate a skeleton maven project of different types, such as JAR, web application, maven site, etc.

### mvn site:site

This command generates a site for the project. You will notice a “site” directory in the target after executing this command. There will be multiple HTML files inside the site directory that provides information related to the project.

### mvn test

This command is used to run the test cases of the project using the maven-surefire-plugin.

### mvn compile

It’s used to compile the source Java classes of the project.

### mvn verify

This command build the project, runs all the test cases and run any checks on the results of the integration tests to ensure quality criteria are met.