Maven is a software project management and comprehension tool. Based on the concept of a Project Object Model (POM), Maven can manage project’s build, reporting and documentation from a central piece of information.

Project Object Model has a set of standards, a project lifecycle, a dependency management system, and logic for executing plugin goals at defined phases in a lifecycle.

Maven project directory structure:

└───maven-project

    ├───pom.xml

    ├───README.txt

    ├───NOTICE.txt

    ├───LICENSE.txt

    └───src

        ├───main

        │   ├───java

        │   ├───resources

        │   ├───filters

        │   └───webapp

        ├───test

        │   ├───java

        │   ├───resources

        │   └───filters

Maven Life Cycle:

1. Validate

Validate the project is correct

1. Compile

Compile the source code of the project

1. Test

Test the compiled source code using a unit testing framework

1. Package

Package the compiled code into distributable format, jar in case of java.

1. Integration-test

Deploy the package into an environment where integration test can be run.

1. Install

Install the package into local repository for use as dependency for other projects locally.

1. Deploy

This phase is performed during release environment. Copies the final package to remote repository for sharing with other developers and projects.

Plugin goals can be attached to each lifecycle phase. Maven executes the goals attached to each phase.

**Installing Maven:**

Download the maven binary files from <https://maven.apache.org/download.cgi> Extract them into desired directory and add the below Environmental variables. Ensure JAVA\_HOME is set and points to JDK installation folder. Append absolute path of bin in the extracted maven to PATH variable. To check the version of maven we use the command mvn –v

**Creating a Maven Project:**

Maven project can be created using a plugin **archetype.**

To create a maven project of name **Sample** with package **org.abc** we execute the following command

$ mvn archetype:generate –DgroupId=org.abc –DartifactId=Sample –DarchetypeArtifactId=maven-archetype-quickstart

After executing the above command, A project named Sample is created with following structure.

└───Sample

    ├───pom.xml

    └───src

        ├───main

        │   ├───java

        │   ├───App.java

        ├───test

           ├───java

           ├───TestApp.java

**Executing a maven Project**

The build life cycle of the mvn is as below:

1. process-resources
2. compile
3. process-test-resources
4. test-compile
5. test
6. package
7. install
8. deploy

To perform any lifecycle method we execute the command **mvn <method>**. By executing any method all the previous methods are executed in-order to execute mentioned lifecycle method. Executing mvn install, it frames a dependency tree based on the project configuration pom.xml on all the sub projects under the super pom.xml (the root POM) and downloads/compiles all the needed components in a directory called .m2 under the user's folder. To execute compiled code with reference to the generated **jar** execute the command

F:\git\_practise\sample>java -cp target/sample-1.0-SNAPSHOT.jar com.abc.App

O/p: Hello World!