**Creating a Maven Project**

**Steps to Create a Maven Project**

1. Open Eclipse  
   Launch your Eclipse IDE.
2. Go to:  
   File → New → Project...
3. Select Project Type:  
   Choose: Maven → Maven Project
4. Click Next
5. Set Workspace Location:
   * Keep default or choose a custom location.
6. Click Next
7. Choose Archetype:
   * Select: maven-archetype-quick start
   * If not visible, check the box: Include snapshot archetypes
8. Click Next
9. Fill Maven Coordinates:
   * Group ID: com. mycompany.app
   * Artifact ID: MyApp
10. Click Finish  
    Eclipse will generate a Maven project with the default structure.

**Sample pom.xml**

<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.mycompany.app</groupId>

<artifactId>MyApp</artifactId>

<version>1.0-SNAPSHOT</version>

<!-- Dependencies Section -->

<dependencies>

<!-- TestNG for unit testing -->

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.9.0</version>

<scope>test</scope>

</dependency>

<!-- Selenium (optional for automation testing) -->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>4.20.0</version>

</dependency>

</dependencies>

</project>

**Maven Build Lifecycle**

Maven defines a series of build phases, grouped into a lifecycle. The most commonly used is the **default lifecycle**.

**Build Phases:**

| **Phase** | **Description** |
| --- | --- |
| validate | Validates the project structure |
| compile | Compiles source code (src/main/java) |
| test | Runs unit tests (src/test/java) |
| package | Packages compiled code into .jar or .war |
| verify | Verifies the package (integration tests) |
| install | Installs the package to the local .m2 repo |
| deploy | Deploys the package to a remote repo |

**Maven Project Directory Structure Explained**

**src/main/java/**

* This is where you write your **application source code**.
* Example: App.java (your main Java class).

Anything in this folder will be compiled during the compile phase.

**src/test/java/**

* This is where you put your **test code** (unit tests).
* Example: AppTest.java (typically uses JUnit or TestNG).

Tests here are executed during the test phase.

**target/ *(not shown above but created after build)***

* Created automatically by Maven when you run mvn compile, package, etc.
* Contains compiled .class files and the final .jar or .war.

**pom.xml**

* The core Maven configuration file.
* Contains:
  + Project metadata (groupId, artifactId, version)
  + Dependencies
  + Plugins
  + Build configuration

**Folder Purpose:**

* **src/main/java/**: Application logic (e.g., App.java)
* **src/test/java/**: Test cases (e.g., AppTest.java, using TestNG or JUnit)
* **target/**: Automatically created after build. Contains:
  + Compiled .class files
  + Final .jar or .war package

**Flowchart**

Start

↓

Create Maven Project (MyApp)

↓

src/main/java/

→ Write application code (App.java)

↓

src/test/java/

→ Write test code (AppTest.java)

↓

pom.xml is generated

↓

Run Maven Lifecycle:

- mvn compile → Compiles App.java

- mvn test → Runs AppTest.java

- mvn package → Creates JAR/WAR in target/

↓

Check target/ directory for output

↓

Run or deploy the built JAR/WAR

↓

Deploy the builts