Understanding Maven

Dependency Management, Build Lifecycle & More

RAHULRAJ P

What is Maven ?

Maven is a open-source build automation and project management tool widely used for java applications. it automates the source code compilation, dependency management, packaging, and execution of test scripts.

How Maven Works ?

1.Project object model(POM)

pom.xml file contains essential information about the project, such as dependencies, source directories and plugins.

pom.xml (MyMavenProjects)

```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/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.example</groupId>
  <artifactId>MyMavenProjects</artifactId>
  <version>1.0-SNAPSHOT</version>
  <name>Archetype - MyMavenProjects</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
      <version>3.1.2</version>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.8.2</version>
    </dependency>
  </dependencies>
</project>
```

2.Build Life cycle

Default Life Cycle

- validate – Verifies project configuration.
- compile – Compiles source code.
- test - Runs unit tests.
- package - Packages compiled code into a JAR/WAR.
- verify - Runs integration tests.
- install - Installs the package in the local repository.
- deploy - Deploys the package to a remote repository.

mvn install -> All phases installing in the local repository.

Clean Life Cycle

- pre-clean – Executes tasks before cleaning.
- clean – Deletes files generated in previous builds (e.g., target/ directory).
- post-clean – Executes tasks after cleaning.

mvn clean ->Handles project cleaning by removing files generated from previous builds, including the target/ directory and other build-related files.

Site Life Cycle

mvn site -> executes the Maven Site Lifecycle, which generates project documentation and reports. It includes phases like pre-site (prepares for site generation), site (creates documentation), post-site (final adjustments), and site-deploy (publishes the site to a server). This helps in maintaining well-documented projects with

automated reporting.

Dependency management

Maven manages project dependencies by automatically downloading required libraries from repositories. It first checks the local repository, then the central repository, and finally any remote repositories specified in the pom.xml.

Why Dependency Management is Important?

- **Automates fetching and versioning** – Ensures the correct versions of libraries are used.
- **Resolves conflicts** – Manages multiple versions of the same library.
- **Reduces manual effort** – Simplifies dependency handling, improving project maintainability.

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <version>3.1.2</version>
  </dependency>
</dependencies>
```

- **groupId** – Identifies the organization or project that created the library.
- ***artifactId*** – Specifies the exact library or module.
- **version** – Defines the specific release of the library to use.

Transitive Dependency

When you declare a dependency in Maven, it not only fetches the specified library but also its required dependencies, known as Transitive Dependencies.

mvn dependency: tree

```
[INFO] org.example:MyMavenProjects:jar:1.0-SNAPSHOT
[INFO] +- org.springframework.boot:spring-boot-starter-web:jar:3.1.2:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter:jar:3.1.2:compile
[INFO] |  |  +- org.springframework.boot:spring-boot:jar:3.1.2:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-autoconfigure:jar:3.1.2:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-starter-logging:jar:3.1.2:compile
[INFO] |  |  |  +- ch.qos.logback:logback-classic:jar:1.4.8:compile
[INFO] |  |  |  |  +- ch.qos.logback:logback-core:jar:1.4.8:compile
[INFO] |  |  |  |  \- org.slf4j:slf4j-api:jar:2.0.7:compile
[INFO] |  |  |  +- org.apache.logging.log4j:log4j-to-slf4j:jar:2.20.0:compile
[INFO] |  |  |  |  \- org.apache.logging.log4j:log4j-api:jar:2.20.0:compile
[INFO] |  |  |  \- org.slf4j:jul-to-slf4j:jar:2.0.7:compile
[INFO] |  |  +- jakarta.annotation:jakarta.annotation-api:jar:2.1.1:compile
[INFO] |  |  +- org.springframework:spring-core:jar:6.0.11:compile
[INFO] |  |  |  \- org.springframework:spring-jcl:jar:6.0.11:compile
[INFO] |  |  \- org.yaml:snakeyaml:jar:1.33:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter-json:jar:3.1.2:compile
[INFO] |  |  +- com.fasterxml.jackson.core:jackson-databind:jar:2.15.2:compile
[INFO] |  |  |  +- com.fasterxml.jackson.core:jackson-annotations:jar:2.15.2:compile
[INFO] |  |  |  \- com.fasterxml.jackson.core:jackson-core:jar:2.15.2:compile
[INFO] |  |  +- com.fasterxml.jackson.datatype:jackson-datatype-jdk8:jar:2.15.2:compile
[INFO] |  |  +- com.fasterxml.jackson.datatype:jackson-datatype-jsr310:jar:2.15.2:compile
[INFO] |  |  \- com.fasterxml.jackson.module:jackson-module-parameter-names:jar:2.15.2:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter-tomcat:jar:3.1.2:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-core:jar:10.1.11:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-el:jar:10.1.11:compile
[INFO] |  |  \- org.apache.tomcat.embed:tomcat-embed-websocket:jar:10.1.11:compile
[INFO] |  +- org.springframework:spring-web:jar:6.0.11:compile
[INFO] |  |  +- org.springframework:spring-beans:jar:6.0.11:compile
[INFO] |  |  \- io.micrometer:micrometer-observation:jar:1.10.9:compile
[INFO] |  |     \- io.micrometer:micrometer-commons:jar:1.10.9:compile
[INFO] |  \- org.springframework:spring-webmvc:jar:6.0.11:compile
[INFO] |     +- org.springframework:spring-aop:jar:6.0.11:compile
[INFO] |     +- org.springframework:spring-context:jar:6.0.11:compile
[INFO] |     \- org.springframework:spring-expression:jar:6.0.11:compile
[INFO] \- junit:junit:jar:4.8.2:compile
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  23.200 s
[INFO] Finished at: 2025-02-15T17:05:42+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

Conflict Resolution in Dependency

When two dependencies require different versions of the same library, Maven resolves the conflict using the Nearest-Wins Strategy:

- Nearest-Wins Strategy – The version closest to the project in the dependency tree is used.
- Explicit Declaration – If a version is explicitly defined in the pom.xml, Maven prioritises it over transitive versions.
- Dependency Management – The <dependencyManagement> section can be used to enforce a specific version across the project.

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
      <version>3.1.2</version> <!-- Version is controlled here -->
    </dependency>
  </dependencies>
</dependencyManagement>

<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <version>3.1.0</version> <!-- Version is controlled here -->
  </dependency>
</dependencies>
```

Maven selects version 3.1.0 of spring-boot-starter-web because the version specified directly in the <dependencies> section takes

precedence over the version defined in <dependencyManagement>. The <dependencyManagement> section only provides a default version when a dependency is declared without a version in <dependencies>. Since spring-boot-starter-web explicitly has version 3.1.0 in <dependencies>, Maven ignores the 3.1.2 version from <dependencyManagement> and resolves 3.1.0 as confirmed by the mvn dependency:tree output.

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn dependency:tree | grep spring-boot-starter-web

[INFO] \- org.springframework.boot:spring-boot-starter-web:jar:3.1.0:compile
rahul-ts426@rahul-ts426 MyMavenProjects %
```

Optimizing dependency Management

Sometimes, transitive dependencies include unnecessary libraries, increasing project size, and causing conflicts. You can exclude them using the <exclusions> tag in pom.xml. For example, to exclude spring-boot-starter-logging from spring-boot-starter-web

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <version>3.1.2</version>
    <exclusions>
      <exclusion>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-logging</artifactId>
      </exclusion>
    </exclusions>
  </dependency>
</dependencies>
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn dependency:tree | grep spring-boot-starter-logging

rahul-ts426@rahul-ts426 MyMavenProjects %
```

If the exclusion is applied correctly, there should be no output related to spring-boot-starter-logging.

Why Use dependencyManagement?

- Ensures all modules in a multi-module project use the same dependency version.
- Prevents version conflicts caused by transitive dependencies.
- Simplifies dependency declarations by keeping versions centralised.

Single Module Project Structure

A single module project contains only one pom.xml, and all the code is inside a single directory.

```
my-single-module-app/
|── src/
|    ├── main/
|    |    ├── java/com/example/app/ (Java source files)
|    |    ├── resources/ (Configuration files)
|    ├── test/
|    |    ├── java/com/example/app/ (Test files)
```

```
8 |── pom.xml  (Maven Project Configuration)
```

Multi-Module Project Structure

A multi-module project has a parent POM that manages multiple sub-modules.

```
1 my-multi-module-app/
2 |── module1/
3 |    ├── src/
4 |    |    ├── main/
5 |    |    |    ├── java/com/example/module1/ (Java source files)
6 |    |    |    ├── resources/ (Configuration files)
7 |    |    ├── test/
8 |    |    |    ├── java/com/example/module1/ (Test files)
9 |    ├── pom.xml
10 |── module2/
11 |    ├── src/
```

```
│   │   ├── main/
│   │   │   ├── java/com/example/module2/ (Java source files)
│   │   │   ├── resources/ (Configuration files)
│   │   ├── test/
│   │   │   ├── java/com/example/module2/ (Test files)
│   ├── pom.xml
├── pom.xml  (Parent POM)
```

Repository

Maven repositories are directories that store packaged JAR files along with metadata, which Maven uses to download dependencies and their versions recursively until all required dependencies are resolved and stored locally. There are three main types of Maven repositories:

- **Local Repository** (~/.m2/repository/) – Cached dependencies downloaded on the local machine.
- **Central Repository** (Maven Central) – Default remote repository hosted by Maven.
- **Remote Repository** – Custom repositories configured by organizations for

internal dependencies.

Plugins

Maven uses plugins to extend its functionality. Plugins are collections of goals that perform specific tasks, such as compiling code, running tests, or generating documentation. These plugins help automate the build lifecycle and ensure consistency across projects.

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.8.1</version>
      <configuration>
        <source>11</source>
        <target>11</target>
      </configuration>
    </plugin>
  </plugins>
</build>
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn compile -X | grep "maven-compiler-plugin"

[DEBUG] Goal:         org.apache.maven.plugins:maven-compiler-plugin:3.8.1:compile (default-compile)
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[DEBUG] org.apache.maven.plugins:maven-compiler-plugin:jar:3.8.1
[DEBUG] Created new class realm plugin>org.apache.maven.plugins:maven-compiler-plugin:3.8.1
[DEBUG] Importing foreign packages into class realm plugin>org.apache.maven.plugins:maven-compiler-plugin:3.8.1
[DEBUG] Populating class realm plugin>org.apache.maven.plugins:maven-compiler-plugin:3.8.1
[DEBUG]   Included: org.apache.maven.plugins:maven-compiler-plugin:jar:3.8.1
[DEBUG] Loading mojo org.apache.maven.plugins:maven-compiler-plugin:3.8.1:compile from plugin realm ClassRealm[plugin>org.apache.maven.plugins:maven-compiler-plugin:3.8.1, parent: jdk.internal.loader.Clas
sLoaders$AppClassLoader@5c29bfd]
[DEBUG] Configuring mojo execution 'org.apache.maven.plugins:maven-compiler-plugin:3.8.1:compile:default-compile' with basic configurator -->
[DEBUG]   (f) mojoExecution = org.apache.maven.plugins:maven-compiler-plugin:3.8.1:compile {execution: default-compile}
rahul-ts426@rahul-ts426 MyMavenProjects %
```

Build Automation

Maven automates the build process by executing a series of predefined tasks based on the project's lifecycle and configuration. This automation simplifies development by managing dependencies, compiling code, running tests, and packaging applications efficiently. It ensures consistency across different environments, reducing manual effort and minimizing errors in the build process.

Build Life Cycle Commands

```
mvn validate
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn validate
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< org.example:MyMavenProjects >----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.041 s
[INFO] Finished at: 2025-02-15T23:53:01+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Verifies POM Syntax** – Ensures pom.xml is correctly structured.

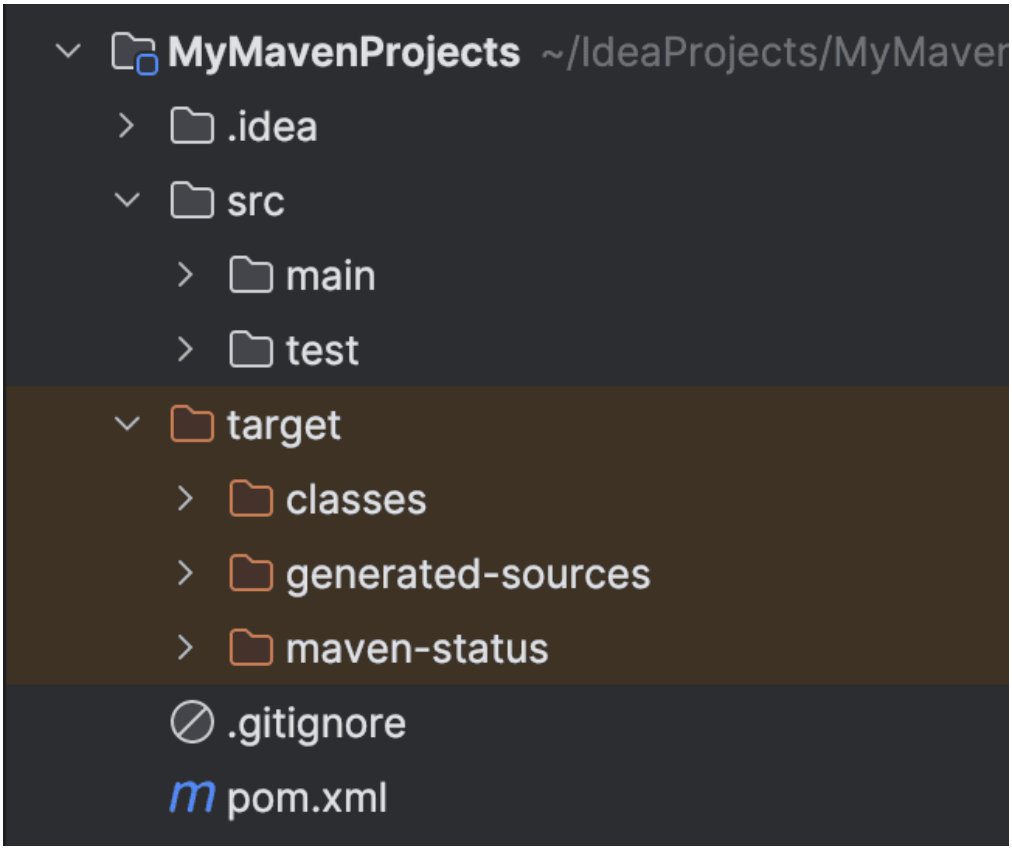
- **Checks Required Plugins** – Confirms necessary build plugins are configured.
- **Validates Dependency Availability** – Resolves dependencies without downloading if they exist locally.
- **Checks Module Structure** – Ensures correct hierarchy in multi-module projects.
- **Validates Profiles** – Confirms active profiles are properly defined and can be applied.

```
mvn compile
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn compile
[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< org.example:MyMavenProjects >----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/classes
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.663 s
[INFO] Finished at: 2025-02-16T09:01:35+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Translates Java source files** (.java) into bytecode (.class).
- **Does not package, test, or execute the code**—only compiles it.
- Uses **dependencies and compiler settings from** pom.xml.

- **Stores compiled** .class files in target/classes.
- **Loads dependencies** - Downloads required .jar files from repositories.
- **Finds source files** - Searches for .java files in src/main/java.
- **Uses compiler plugin** - Calls maven-compiler-plugin for compilation.
- **Validates code** - Ensures there are no syntax errors before proceeding.



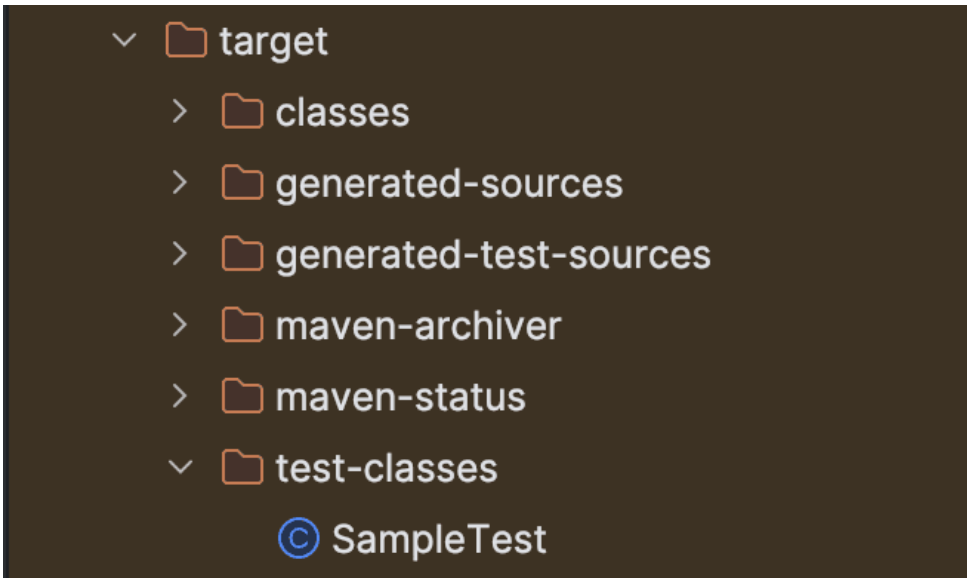

```
mvn test
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< org.example:MyMavenProjects >----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /Users/rahul-ts426/IdeaProjects/MyMavenProjects/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ MyMavenProjects ---
[INFO] Surefire report directory: /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.072 s
[INFO] Finished at: 2025-02-16T09:27:39+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Only runs tests** inside src/test/java.
- **Depends on** mvn compile, so if compilation fails, mvn test will fail.
- **Compiles source code** - Calls mvn compile if not already done.
- **Compiles test code** - Compiles test classes in src/test/java.
- **Loads test dependencies** - Downloads JUnit/TestNG dependencies.
- **Runs tests** - Uses the Surefire Plugin to execute test cases.

- **Surefire Plugin required** - If missing, mvn test won't run any tests.

```
#Skipping Test
mvn test -Dmaven.test.skip=true

#Force Maven to continue Even if Tests Fail
mvn test -Dmaven.test.failure.ignore=true

#Run a Single Test class
mvn test -Dtest=MyTest

#Runs a specific method in a Test Class
#If you have a test class MyTest and a method testLogin, run:
mvn test -Dtest=MyTest#testLogin test
```

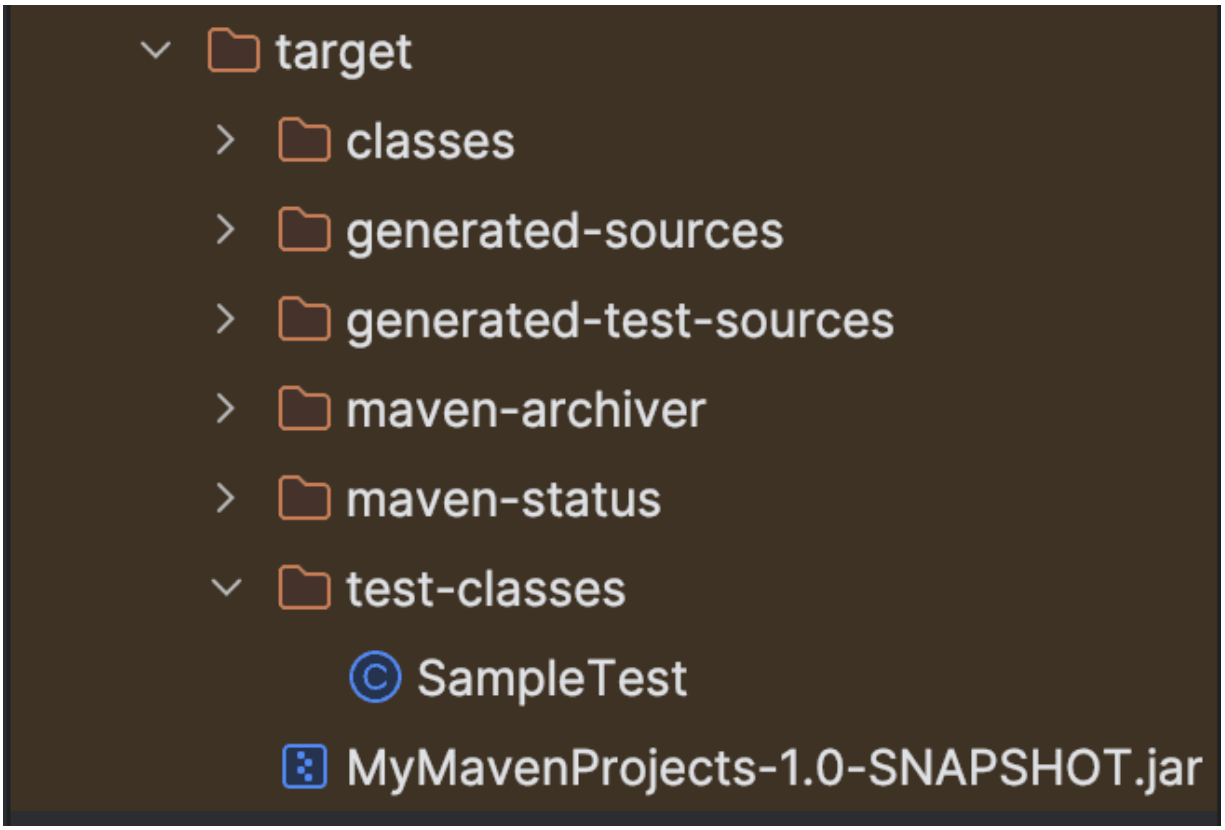



```
mvn package
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< org.example:MyMavenProjects >----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /Users/rahul-ts426/IdeaProjects/MyMavenProjects/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ MyMavenProjects ---
[INFO] Surefire report directory: /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ MyMavenProjects ---
[INFO] Building jar: /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/MyMavenProjects-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.739 s
[INFO] Finished at: 2025-02-16T13:59:56+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Runs all previous** lifecycle phases (compile, test).
- **Generates** a JAR (.jar) or WAR (.war) inside the target/ directory.
- Uses maven-jar-plugin (for JAR) or maven-war-plugin (for WAR) for packaging.
- **Runs unit tests** before packaging unless explicitly skipped (-DskipTests).
- **Creates the final artifact** inside target/, e.g., target/myapp.jar.

- Uses maven-shade-plugin for creating a fat JAR (Uber JAR) with dependencies.
- **Supports executable** JAR/WAR creation with all required dependencies.




```
mvn verify
```

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn verify
[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< org.example:MyMavenProjects >----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /Users/rahul-ts426/IdeaProjects/MyMavenProjects/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ MyMavenProjects ---
[INFO] Surefire report directory: /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ MyMavenProjects ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.737 s
[INFO] Finished at: 2025-02-16T14:31:24+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Executes** integration tests, code quality checks, and verification before installation.
- **Runs unit tests** using maven-surefire-plugin.
- **Runs integration tests** using maven-failsafe-plugin.
- **Ensures all tests** pass before proceeding to installation or deployment.
- **Validates** the correctness of the packaged artifact.
- **Verifies** dependencies, configurations, and plugin executions.

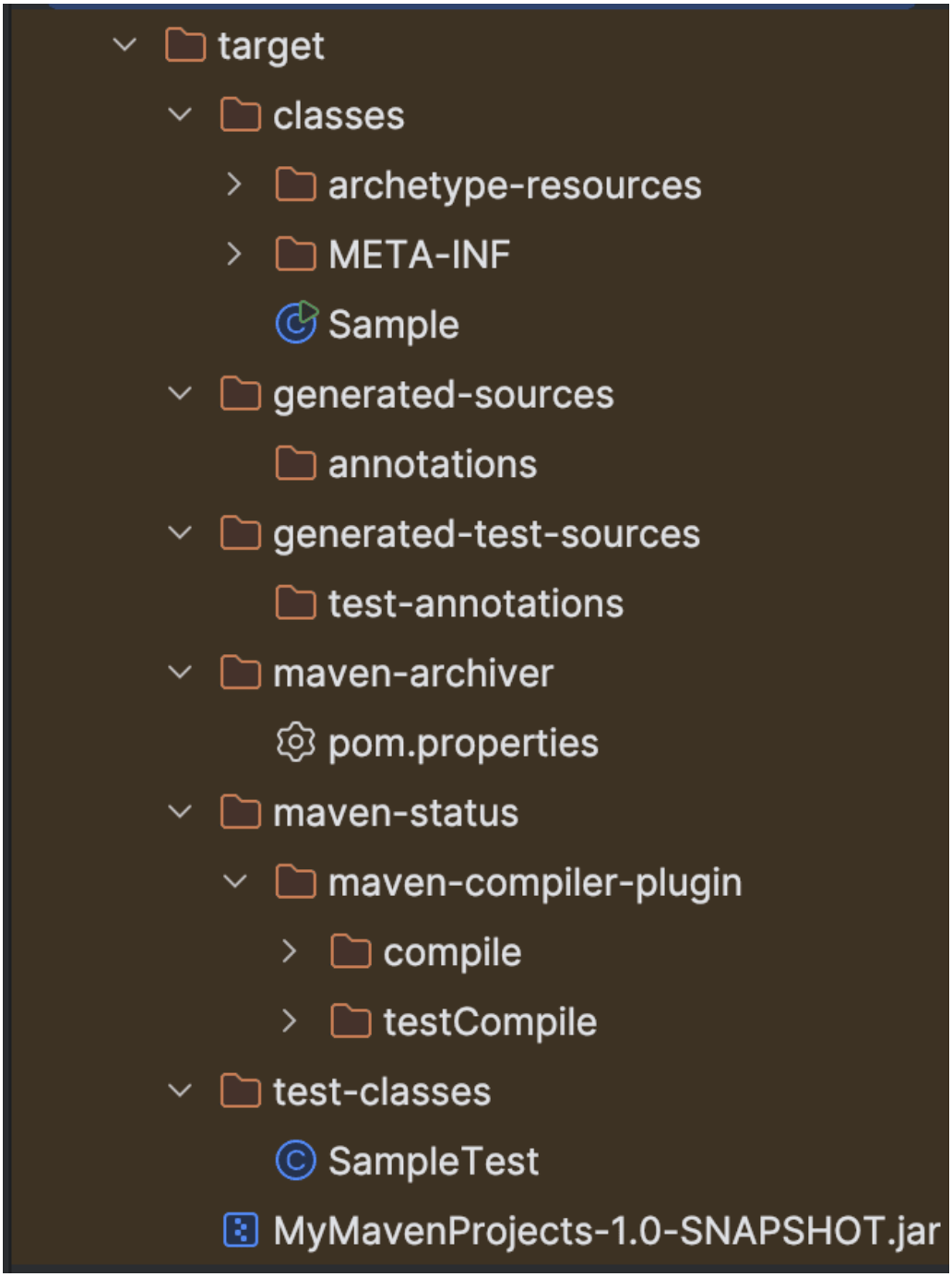
1 mvn install

```
rahul-ts426@rahul-ts426 MyMavenProjects % mvn install
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< org.example:MyMavenProjects >-----------------------
[INFO] Building Archetype - MyMavenProjects 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ MyMavenProjects ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /Users/rahul-ts426/IdeaProjects/MyMavenProjects/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ MyMavenProjects ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ MyMavenProjects ---
[INFO] Surefire report directory: /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/surefire-reports

-------------------------------------------------------
 T E S T S
-------------------------------------------------------

Results :

Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ MyMavenProjects ---
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ MyMavenProjects ---
[INFO] Installing /Users/rahul-ts426/IdeaProjects/MyMavenProjects/target/MyMavenProjects-1.0-SNAPSHOT.jar to /Users/rahul-ts426/.m2/repository/org/example/MyMavenProjects/1.0-SNAPSHOT/MyMavenProjects-1.0
SNAPSHOT.jar
[INFO] Installing /Users/rahul-ts426/IdeaProjects/MyMavenProjects/pom.xml to /Users/rahul-ts426/.m2/repository/org/example/MyMavenProjects/1.0-SNAPSHOT/MyMavenProjects-1.0-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.764 s
[INFO] Finished at: 2025-02-16T15:18:36+05:30
[INFO] ------------------------------------------------------------------------
rahul-ts426@rahul-ts426 MyMavenProjects %
```

- **Runs all previous lifecycle phases** (compile, test, package, verify).
- **Installs the built artifact** (JAR/WAR) into the local repository (~/.m2/repository).
- Ensures the project is available for **local dependency** resolution before deployment.
- **Creates** a JAR/WAR file and places it in the local Maven repository.
- **Local repository** (~/.m2/repository) acts as a cache for dependencies and installed artifacts.
- **Central repository** (default remote repository) is used for

downloading dependencies.

- **Remote repository** (custom/private) stores dependencies for private or company projects.



```
1 mvn deploy
```

```
[rahul-ts426@rahul-ts426 MyMavenProjects % mvn deploy
```

- **Runs all previous lifecycle** phases (compile, test, package, install).
- **Packages** the project and uploads artifacts (JAR, WAR, POM) to a remote repository.
- **Installs artifacts** into the local repository (~/.m2/repository).
- **<distributionManagement>** in pom.xml defines release and snapshot repository URLs.
- **Authentication for deployment** is managed via settings.xml using <server> credentials.
- Use **mvn clean deploy** to deploy artifacts to the configured remote repository.
- **Snapshot versions** (1.0.0-SNAPSHOT) go to the snapshot repository, releases (1.0.0) to the release repository.
- **Debug deployment** issues using mvn clean deploy -X.
- **Repository managers** like Nexus and Artifactory are used for private repositories.

THANK YOU