

MAVEN

Build automation and dependency management tool

Maven is a build automation and dependency management tool used primarily for Java projects.

MAVEN = "Maven Is a Project Management Tool"

Think of Maven as a **project manager** that helps you:

- Download and manage required libraries (dependencies)
- Compile code
- Package the code into .jar or .war
- Run tests
- Create documentation

Use of Maven in Spring Boot :

In Spring Boot projects, Maven:

- Automatically manages all required libraries
- Simplifies project structure
- **Provides build lifecycle** (clean, compile, install, package)
- Helps developers avoid **manual downloads** of JARs

Working of Maven

1. POM File (pom.xml):

- POM = Project Object Model
- It's an XML file that defines:
 - Project name, version
 - Dependencies
 - Plugins
 - Build configuration

Every Maven project must have a pom.xml.

Example pom.xml for Spring Boot

```
<project xmlns="http://maven.apache.org/POM/4.0.0">  
  <modelVersion>4.0.0</modelVersion>  
  
  <groupId>com.marvellous</groupId>  
  <artifactId>springboot-demo</artifactId>  
  <version>0.0.1-SNAPSHOT</version>  
  <packaging>jar</packaging>  
  
  <parent>  
    <groupId>org.springframework.boot</groupId>  
    <artifactId>spring-boot-starter-parent</artifactId>  
    <version>3.2.0</version>  
  </parent>  
  
  <dependencies>  
    <!-- Spring Boot Web Dependency -->  
    <dependency>  
      <groupId>org.springframework.boot</groupId>  
      <artifactId>spring-boot-starter-web</artifactId>  
    </dependency>  
  
    <!-- Spring Boot Test Dependency -->  
    <dependency>  
      <groupId>org.springframework.boot</groupId>  
      <artifactId>spring-boot-starter-test</artifactId>  
      <scope>test</scope>  
    </dependency>
```

```
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
</project>
```

Common Maven Terminologies

Term	Meaning
Artifact	A build output (e.g., .jar, .war)
GroupId	Like a package name (e.g., com.marvellous)
ArtifactId	Project name (e.g., springboot-demo)
Version	Project version
Repository	Central place where JARs are stored (like Maven Central)
Dependencies	External libraries your project needs

Maven Build Lifecycle

Phase	Description
clean	Deletes the old compiled files
compile	Compiles Java code
package	Creates .jar or .war
install	Installs into local Maven repo
test	Runs unit tests
deploy	Publishes to remote repo

Maven Commands for Spring Boot

1. Project Compilation

`mvn compile`

Compiles the Java source code in the `src/main/java` directory.

2. Clean Build Folder

`mvn clean`

Deletes the `target/` directory (removes old compiled files).

3. Package Project

`mvn package`

Compiles and packages the code into a `.jar` or `.war` inside the `target/` folder.

- Example Output:

`target/Marvellous-0.0.1-SNAPSHOT.jar`

4. Run Spring Boot Application

`mvn spring-boot:run`

Starts the Spring Boot application without manually compiling or packaging.

Most commonly used during development!

5. Run Unit Tests

`mvn test`

Executes all unit test cases from the `src/test/java` folder.

6. Build and Run Together (Clean + Package)

`mvn clean package`

First cleans old build, then compiles and creates the `.jar`.