Introduction to Maven

Simplifying Java Development and Project Management

Name: Rama Shanker

Date: 21/01/2025

What is Maven?

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

It follows the Project Object Model (POM) to manage dependencies, build processes, and configurations.

Developed by Apache Software Foundation.

Key Features

Dependency Management: Automatically handles library dependencies.

Standardized Build Process: Provides a consistent project structure.

Extensible Plugin System: Add functionalities through plugins.

Repository Support: Access to central and private repositories for dependencies.

Support for Multi-Module Projects: Manage multiple projects under one build process.

Maven Lifecycle

Phases of the Build Lifecycle:

Clean: Cleans the build directory.

Validate: Validates the project is correct and all necessary information is available.

Compile: Compiles the source code.

Test: Runs unit tests using testing frameworks like JUnit.

Package: Packages the compiled code into a JAR/WAR file.

Install: Installs the package into the local repository.

Deploy: Deploys the final package to a remote repository.

Benefits of Maven

Simplifies Dependency Management:

Automatically resolves and downloads required libraries.

Enhances Consistency:

Predefined structure and lifecycle ensure standardization.

Improves Collaboration:

Easily share dependencies and configurations across teams.

Extensible and Scalable:

Support for custom plugins and large-scale projects.

Integration Support:

Works seamlessly with IDEs like IntelliJ IDEA, Eclipse, and CI/CD tools like Jenkins.

Basic Maven Commands

mvn clean: Cleans up the project by deleting the target directory.

mvn compile: Compiles the source code.

mvn test: Executes unit tests.

mvn package: Packages the project into a distributable format (e.g., JAR, WAR).

mvn install: Installs the package to the local repository.

mvn deploy: Deploys the project to a remote repository.

Maven Directory Structure

Standard Directory Layout:

```
project/
|-- src/
|   |-- main/
|   |   |-- java/ (Source code)
|   |   |-- resources/ (Configuration files)
|   |-- test/
|       |-- java/ (Test code)
|       |-- resources/ (Test configurations)
|-- target/ (Compiled code and packages)
|-- pom.xml (Project configuration)
```

How Maven Manages Dependencies

POM File: The heart of Maven.

```
<dependencies>
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>5.3.5</version>
    </dependency>
</dependencies>
```

Repositories:

Local Repository: Stored on the developer's machine.

Central Repository: Maintained by Maven.

Remote Repository: Custom repositories for specific teams/organizations.

Common Maven Plugins

Surefire Plugin:

Executes unit tests.

Compiler Plugin:

Compiles Java source code.

Jar Plugin:

Packages the code into a JAR file.

Checkstyle Plugin:

Ensures adherence to coding standards.

Deploy Plugin:

Automates deployment to remote repositories.

Conclusion

Maven simplifies project management, dependency resolution, and builds.

It ensures consistency and scalability for Java-based projects.

Learn Maven to improve productivity and standardization in software development.