Day 29

Assignment 1: Build Lifecycle

Demonstrate the use of Maven lifecycle phases (clean, compile, test, package, install, deploy) by executing them on a sample project and documenting what happens in each phase.

A)

To demonstrate the use of Maven lifecycle phases, let's go through the steps using a sample Maven project. Each phase will be executed, and the results will be documented.

1. Setting Up the Sample Project

First, we create a sample Maven project using the Maven Archetype plugin.

```
mvn archetype:generate -DgroupId=com.example -DartifactId=my-sample-project -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

Navigate to the project directory:

```
cd my-sample-project
```

2. Project Structure

The project structure will look like this:

```
my-sample-project
├── pom.xml
└── src
    ├── main
    │   └── java
    │       └── com
    │           └── example
    │               └── App.java
    └── test
        └── java
            └── com
            └── example
                └── AppTest.java
```

3. Maven Lifecycle Phases

3.1. Clean Phase

The clean phase deletes the target directory, which contains all the build artifacts from previous builds.

Execute the clean phase:

```
mvn clean
```

What happens:

Maven removes the target directory and all its contents.

3.2. Compile Phase

The compile phase compiles the source code located in src/main/java.

Execute the compile phase:

```
mvn compile
```

What happens:

Maven compiles the Java source files and places the compiled .class files in the target/classes directory.

3.3. Test Phase

The test phase runs the unit tests located in src/test/java.

Execute the test phase:

```
mvn test
```

What happens:

Maven compiles the test code and runs the tests using a unit testing framework (e.g., JUnit).

Test results are stored in the target/surefire-reports directory.

3.4. Package Phase

The package phase packages the compiled code into a JAR file.

Execute the package phase:

```
mvn package
```

What happens:

Maven packages the compiled classes and resources into a JAR file located in the target directory, typically named my-sample-project-1.0-SNAPSHOT.jar.

3.5. Install Phase

The install phase installs the packaged JAR into the local Maven repository.

Execute the install phase:

```
mvn install
```

What happens:

Maven installs the JAR file into the local Maven repository (~/.m2/repository), making it available for other local projects to use as a dependency.

3.6. Deploy Phase

The deploy phase uploads the packaged JAR to a remote repository, typically for sharing with other developers.

Before executing this phase, ensure your pom.xml is configured with the correct remote repository details.

Execute the deploy phase:

```
mvn deploy
```

What happens:

Maven uploads the JAR file to the configured remote repository.

Summary of Each Phase

clean: Removes previous build artifacts (target directory).

compile: Compiles the source code to bytecode (target/classes).

test: Runs unit tests and generates test reports (target/surefire-reports).

package: Packages the compiled code into a JAR file (target/my-sample-project-1.0-SNAPSHOT.jar).

install: Installs the JAR file into the local repository (~/.m2/repository).

deploy: Uploads the JAR file to a remote repository.

By running through these phases, Maven automates the build and management of the project, ensuring that each step of the lifecycle is handled in a consistent and repeatable manner.