1.A Build life cycle

Sample Maven Project Structure

Assume a simple Maven project structure like this:

sample-project

├── pom.xml

└── src

├── main

│ └── java

│ └── com

│ └── example

│ └── Sample.java

└── test

└── java

└── com

└── example

└── SampleTest.jav

The `clean` phase removes all files generated by the previous builds. This ensures a clean slate before starting a new build.

Execute the `clean` phase:

```bash

mvn clean

#### 2. `compile`

The `compile` phase compiles the source code of the project.

Execute the `compile` phase:

mvn compile

3. `test`

The `test` phase executes tests using a suitable unit testing framework (like JUnit) for the project.

Execute the `test` phase:

```bash

mvn test

```

The `package` phase packages the compiled code (along with resources) into distributable formats such as JAR, WAR, or EAR.

Execute the `package` phase:

mvn package

5. `install`

The `install` phase installs the packaged artifact into the local Maven repository, making it available for other projects.

Execute the `install` phase:

```bash

mvn install

6. `deploy`

The `deploy` phase copies the final packaged artifact to a remote repository for sharing it with other developers or projects.

Execute the `deploy` phase (requires a configured remote repository, such as Nexus or Artifactory):

mvn deploy