Apache Maven Commands and Guide

Introduction to Apache Maven

Apache Maven is a build automation and dependency management tool primarily used for Java projects. It uses a project object model (POM) file to define project structure, dependencies, and build configurations.

General Maven Commands

Verify Maven Installation

Command: mvn -v

Displays the installed Maven version and system information.

Create a Maven Project

Command: mvn archetype:generate -DgroupId=com.example -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

Creates a new Maven project with the specified groupId, artifactId, and archetype.

Build a Project

Command: mvn clean install

Cleans previous builds and compiles, tests, and packages the project.

Clean the Project

Command: mvn clean

Deletes the target directory to remove previous build artifacts.

Compile the Code

Command: mvn compile

Compiles the source code of the project.

Test the Project

Command: mvn test

Runs unit tests defined in the project.

Package the Project

Command: mvn package

Compiles the code and packages it into a .jar or .war file as specified in the pom.xml.

Execute a Project

Command: mvn exec:java -Dexec.mainClass="com.example.Main"

Executes the main class of the project. Ensure the exec plugin is added to the pom.xml.

Dependency Management Commands

Add Dependencies

Command: Add dependencies manually to the <dependencies> section in pom.xml:

```
<dependencies>
 <dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-core</artifactId>
  <version>5.3.12</version>
 </dependency>
</dependencies>
```

Adds a new dependency to your Maven project.

View Dependency Tree

Command: mvn dependency:tree

Displays the dependency hierarchy of the project, including transitive dependencies.

Analyze Dependencies

Command: mvn dependency:analyze

Identifies unused dependencies and potential issues in the project.

Download Dependencies

Command: mvn dependency:resolve

Downloads and resolves all specified dependencies in the pom.xml file.

List All Dependencies

Command: mvn dependency:list

Lists all dependencies of the project.

Copy Dependencies

Command: mvn dependency:copy-dependencies -DoutputDirectory=./libs

Copies project dependencies to the specified directory.

Plugin Commands and Guide

Adding a Plugin

Command: Include a plugin in the <plugins> section of pom.xml:

```
<build>
 <plugins>
  <plugin>
   <groupId>org.apache.maven.plugins</groupId>
   <artifactId>maven-compiler-plugin</artifactId>
   <version>3.8.1</version>
   <configuration>
    <source>1.8</source>
    <target>1.8</target>
   </configuration>
  </plugin>
 </plugins>
</build>
```

Adds a compiler plugin to configure source and target Java versions.

List Plugins

Command: mvn help:active-profiles

Lists all active profiles and associated plugins in the project.

Run a Specific Plugin Goal

Command: mvn <plugin-group>:<plugin-artifact>:<goal>

Executes a specific plugin goal. For example:
mvn org.apache.maven.plugins:maven-clean-plugin:clean

Install a Plugin

Command: mvn install:install-plugin -DgroupId=<groupId> -DartifactId=<artifactId> -Dversion=<version>

Installs a custom plugin locally.

Configure a Plugin

Command: Customize plugin behavior by adding configuration in pom.xml:

```
<plugin>
 <groupId>org.apache.maven.plugins</groupId>
 <artifactId>maven-surefire-plugin</artifactId>
 <version>3.0.0-M5</version>
 <configuration>
  <includes>
   <include>**/Test*.java</include>
```

```
  </includes>
 </configuration>
</plugin>
```

Configures the Surefire plugin to include specific test files.

Verify Plugin Version

Command: mvn plugin:help -Dplugin=<plugin-name>

Displays information about the specified plugin, including its version. For example: mvn plugin:help -Dplugin=compiler