Getting started with Maven

Create Java project

mvn archetype:generate
-DgroupId=org.yourcompany.project
-DartifactId=application

Create web project

mvn archetype:generate
-DgroupId=org.yourcompany.project
-DartifactId=application
-DarchetypeArtifactId=maven-archetype-webapp

Create archetype from existing project

mvn archetype:create-from-project

Main phases

- clean — delete target directory
- validate — validate, if the project is correct
- compile — compile source code, classes stored in target/classes
- test — run tests
- package — take the compiled code and package it in its distributable format, e.g. JAR, WAR
- verify — run any checks to verify the package is valid and meets quality criteria
- install — install the package into the local repository
- deploy — copies the final package to the remote repository

Useful command line options

-DskipTests=true compiles the tests, but skips running them
-Dmaven.test.skip=true skips compiling the tests and does not run them
-T number of threads:
- 4 is a decent default
- 2C - 2 threads per CPU
- rf, --resume-from resume build from the specified project
- pi, --projects makes Maven build only specified modules and not the whole project
- am, --also-make makes Maven figure out what modules out target depends on and build them too
- o, --offline work offline
- X, --debug enable debug output
- P, --activate-profiles comma-delimited list of profiles to activate
- U, --update-snapshots forces a check for updated dependencies on remote repositories
- ff, --fail-fast stop at first failure

Essential plugins

Help plugin — used to get relative information about a project or the system.
mvn help:describe describes the attributes of a plugin
mvn help:effective-pom displays the effective POM as an XML for the current build, with the active profiles factored in.

Dependency plugin — provides the capability to manipulate artifacts.
mvn dependency:analyze analyzes the dependencies of this project
mvn dependency:tree prints a tree of dependencies

Compiler plugin — compiles your java code.
Set language level with the following configuration:

<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.6.1</version>
  <configuration>
    <source>1.8</source>
    <target>1.8</target>
  </configuration>
</plugin>

Version plugin — used when you want to manage the versions of artifacts in a project’s POM.

Wrapper plugin — an easy way to ensure a user of your Maven build has everything that is necessary.

Spring Boot plugin — compiles your Spring Boot app, build an executable fat jar.

Exec — amazing general purpose plugin, can run arbitrary commands ;)

The big picture

- maven
  - Central
    - Plugins
    - Dependencies
  - Local Repository
    ~/.m2/settings.xml