Maven Java build tool

CONCEPTS

Name a phase and Maven runs every earlier phase too: mvn package also compiles and tests.

validate check POM
↓
compile main sources
↓
test unit tests
↓
package jar / war
↓
verify integration tests
↓
install to ~/.m2
↓
deploy to remote repo

Default lifecycle: asking for a phase runs all phases before it

BUILD

`mvn compile`
Compile main sources

`mvn test`
Run unit tests

`mvn package`
Build the artifact into target/

`mvn verify`
Run integration tests and checks

`mvn install`
Copy artifact to local repo ~/.m2

`mvn clean`
Delete the target/ directory

`mvn clean install`
Fresh build, then install locally

OPTIONS

`-DskipTests`
Compile tests but do not run them

`-o / --offline`
Use only the local repository

`-q / --quiet`
Print errors only

`-e / --errors`
Show full stack traces

OPTIONS (cont.)

`-U / --update-snapshots`
Force re-check of SNAPSHOT deps

`-T <n>`
Build modules in parallel with n threads

`-D<key>=<value>`
Set a system property

MULTI-MODULE

`mvn -pl <module>`
Build only this module

`mvn -pl <module> -am`
Also build what it depends on

`mvn -rf :<module>`
Resume the reactor from a module

DEPENDENCIES

`mvn dependency:tree`
Print the resolved dependency tree

`mvn dependency:tree -Dincludes=<g>:<a>`
Find who pulls in a library

INSPECT AND DEBUG

`mvn help:effective-pom`
Show the POM after inheritance

`mvn help:evaluate -Dexpression=<expr> -q -DforceStdout`
Print one POM value

`mvn -v`
Show Maven, Java and OS versions

`mvn -Dtest=<Class>#<method>`
Run one test method

PROFILES AND PROPERTIES

`mvn -P<id>`
Activate a profile (-Pa,b for several)

WRAPPER AND PROJECT

`mvn wrapper:wrapper`
Add mvnw and mvnw.cmd to the project

`./mvnw <args>`
Run with the pinned Maven version