

JVM Build Tools Cheat Sheet

Shared concepts · all three tools

Coordinates group:artifact:version[:classifier] (Ivy: org, name, rev). All three read the Maven repository layout: org/apache/commons/commons-csv/1.14.1/

Transitive a dependency brings its own (exclude the unwanted).

Conflicts Maven: **nearest** wins, then first declared · Gradle: **highest** requested · Ivy: a conflict manager.

SNAPSHOT vs release a SNAPSHOT is re-checked (updatePolicy daily by default, -U forces); a release is immutable.

Ranges / dynamic [1.0,2.0) · 4.+ (Gradle, Ivy) · latest.release, latest.integration

BOM / platform Maven import scope · Gradle platform()

Lock & verify Gradle gradle.lockfile, verification-metadata.xml · Maven: no core lockfile, trusted checksums · Ivy ivy:deliver, checksums=

Wrapper vs installed mvnw/gradlew pin the tool (check distributionSha256Sum); Ant has none.

Incremental Ant: per task, by timestamp · Maven: per plugin · Gradle: task inputs/outputs, build cache.

Reproducible Ant modificationtime (1.10.2) · Maven project.build.outputTimestamp (default in 3.10) · Gradle: default since 9.0

Apache Ant 1.10.18 · Java 8+ · build.xml

ant [-f other.xml] [**target**] default target / other file

ant -p targets that have a description

ant -Dname=value property; wins over the file

ant -k keep going after a failed target

ant -lib dir JARs (antlibs, Ivy) on Ant's classpath

ant -diagnostics Java, ANT_HOME, optional tasks

-q / -v / -d · ANT_OPTS quiet / verbose / debug · JVM options

```
<project name="tidepool" default="jar">
<property name="build" location="build"/>
<target name="compile">
<mkdir dir="${build}/classes"/>
<javac srcdir="src/main/java"
destdir="${build}/classes" release="21"
includeantruntime="false"/>
</target>
<target name="jar" depends="compile">
<jar destfile="${build}/tidepool.jar"
basedir="${build}/classes">
<manifest><attribute name="Main-Class"
value="io.tidepool.Main"/></manifest></jar>
</target>
</project>
```

\$(name) properties are immutable: first definition wins

includes="/*.java"** ** any depth of directories; selectors (<contains>, <date>...) narrow a fileset

mappers <globmapper from="*" to="*.bak"/>; flatten, regexp, package...

filterchains <filterchain><headfilter lines="2"/>; expandproperties, replacetokens(@token@)

junitlauncher JUnit Platform (JUnit 5+; 4 via Vintage)

java fork="true" · exec own JVM · a program

depends="a,b" each target runs at most once

if= unless= · if:set target / any task (ant:if)

import / include reuse files: overridable / prefixed

subant · ant · antcall other build files · re-run a target

xmlns:x="antlib:pkg" antlib namespace (Ivy, Resolver)

macrodef with @{attribute}

```
<macrodef name="compile-module">
<attribute name="module"/>
<attribute name="release" default="21"/>
<sequential><javac srcdir="@{module}/src/main/java"
destdir="@{module}/build/classes"
release="@{release}"/></sequential>
</macrodef>
<compile-module module="tidepool-model"/>
```

Custom task in ten lines

```
public class HelloTask extends Task {
private String message;
public void setMessage(String m) { message = m; }
@Override
public void execute() throws BuildException {
if (message == null)
throw new BuildException("message?");
log(message);
}
}
<taskdef name="hello"
classname="io.tidepool.ant.HelloTask"/>
```

Ivy 2.6.0: ivy.xml, tasks, conflict managers

```
<conf name="compile"><conf name="test"/>
<dependency org="org.apache.commons"
name="commons-csv" rev="1.14.1"
conf="compile->default"/> <!-- conf mapping -->
<ivy:resolve/>
<ivy:cachepath pathid="cp" conf="compile"/>
<ivy:retrieve
pattern="lib/[conf]/[artifact]-[revision].[ext]"/>
<ivy:publish resolver="shared" pubrevision="1.0.0"/>
```

Conflict managers (ivysettings.xml): latest-revision, latest-time, latest-compatible.

Resolver Ant Tasks 2.0.0 <resolver:resolve> with <dependency coords="org.apache.commons:commons-csv:1.14.1"/> into a path; resolver:install, resolver:deploy

Apache Maven 3.10.0 · Java 8+ · pom.xml

```
mvn clean verify build, unit + integration tests
mvn install / deploy to ~/.m2/repository / remote
-pl :engine -am / -amd + modules it needs / that need it
-T 1C · -o · -U 1 thread/core · offline · update SNAPSHOTS
-B -ntp · -e · -X batch, no progress · stack traces · debug
-P prod · -Dkey=val activate a profile · set a property
-s file · -t file user settings.xml · toolchains.xml
-DskipTests · -Dtest=X do not run tests · one test class
help:effective-pom after inheritance and profiles
dependency:tree · :analyze graph · used-undeclared/unused
mvn wrapper:wrapper generate mvnw (wrapper 3.3.4)
```

Lifecycle clean · default · site (jar bindings)

validate → ... → process-resources resources:resources → **compile** compiler:compile → test-compile → **test** surefire:test → **package** jar:jar → integration-test → **verify** → **install** install:install → **deploy** deploy:deploy. A phase runs all phases before it.

```
<project xmlns="http://maven.apache.org/POM/4.0.0">
<modelVersion>4.0.0</modelVersion>
<parent>...<relativePath>../pom.xml</relativePath>...
<groupId>io.tidepool</groupId>
<artifactId>tidepool-model</artifactId>
<version>1.0.0</version>
<SNAPSHOT</version>
<properties>
```

```
<maven.compiler.release>21</maven.compiler.release>
</properties>
<dependencyManagement><dependencies><dependency>
<groupId>org.junit</groupId>
<artifactId>junit-bom</artifactId>
<version>6.1.3</version>
<type>pom</type><scope>import</scope>
</dependency></dependencies></dependencyManagement>
</project>
```

compile default; every classpath, transitive

provided compile + test; supplied at runtime

runtime / test runtime + test / test only

import BOM in dependencyManagement, type pom

optional> · <exclusions> not passed on · cut a transitive

parent> vs <modules> inheritance vs aggregation (reactor)

\$(revision) CI-friendly version; flatten plugin on Maven 3

settings.xml <localRepository>, <servers>, <mirrors>, <profiles>

<mirrorOf>*; external:*, central:*, !lab; external:http:* = 3.8.1 HTTP blocker

<activation> activeByDefault, jdk, os, property, file; or -P

release · toolchains maven.compiler.release; toolchains pick a JDK

Surefire / Failsafe *Test, Test*, *Tests, *TestCase / *IT, IT*, *ITCase

shade / assembly uber JAR / custom archives

Central sources + Javadoc JARs, GPG, central-publishing-maven-plugin 0.12.0, Portal token in <server>

enforcer requireMavenVersion, requireJavaVersion, dependencyConvergence, requireUpperBoundDeps, bannedDependencies, requirePluginVersions

Mojo skeleton (packaging maven-plugin)

```
@Mojo(name = "check-stations", threadSafe = true,
defaultPhase = LifecyclePhase.VALIDATE)
public class CheckStationsMojo extends AbstractMojo {
@Parameter(property = "tidepool.file")
private File file;
public void execute() throws MojoExecutionException,
MojoFailureException { ... }
}
```

3.9 → 3.10 breaking changes

POM validation duplicate deps/plugins, bad version chars, parent relativePath: errors

classpath breadth-first level order; - Daether.system.dependencyVisitor=preOrder restores

super POM drops release-profile and plugin management (pin plugins); sets UTF-8 and outputTimestamp

credentials scoped to repository origins · bad toolchains fail

Maven 4 · 4.0.0-rc-7 · release candidate · Java 17+

modelVersion 4.1.0 new POM model; consumer POM published

<subprojects> replaces <modules>

<packaging>bom real BOM; consumer POM says pom

before: / after: · all / each hooks per phase · umbrella phases

mvn -r resume from the last failure

mvnup check|apply · mvnenc upgrade tool · password encryption

Gradle 9.8.1 · runs on Java 17-27 · build.gradle.kts

```
./gradlew build assemble + check (always the wrapper)
tasks --all · build -x test list tasks · exclude one
:app:test --tests 'io.*' one project, filtered
--continue · -t go on after a failure · continuous
```

```
--offline · --refresh-dependencies no network · re-check
--write-locks write gradle.lockfile
--build-cache · --configuration-cache · --scan
dependencies · dependencyInsight --dependency x tree · why
wrapper --gradle-version 9.8.1 upgrade; run ./gradlew wrapper
again
Phases: initialization (settings) → configuration (scripts) → execution
(tasks).
// settings.gradle.kts
pluginManagement { includeBuild("build-logic") }
include("tidepool-model", "tidepool-engine",
"tidepool-cl1")
includeBuild("../tidepool-model") // composite
// build.gradle.kts
plugins { id("tidepool.java-conventions")
`java-library` ; `maven-publish` }
java { toolchain {
languageVersion = JavaLanguageVersion.of(21) } }
dependencies {
api(libs.commons.csv)
testImplementation(platform(libs.junit.bom))
testImplementation(libs.junit.jupiter)
}
dependencyLocking { lockAllConfigurations() }
testing { suites { // JVM Test Suite
register<JvmTestSuite>("integrationTest") {
useJUnitJupiter()
dependencies { implementation(project()) } } } }
publishing { publications {
create<MavenPublication>("maven") {
from(components["java"]) } } }
# gradle/libs.versions.toml
[versions]
junit = "6.1.3"
[libraries]
commons-csv = "org.apache.commons:commons-csv:1.14.1"
junit-bom = { module = "org.junit:junit-bom",
version.ref = "junit" }
api compile + runtime, exposed to consumers
implementation compile + runtime, hidden
compileOnly / runtimeOnly one classpath only
testImplementation tests (+ testRuntimeOnly)
annotationProcessor annotation processors
platform() / enforcedPlatform() BOM / forced BOM
rich versions require, strictly, prefer, reject
abstract class Hello : DefaultTask() {
@get:Input abstract val station: Property<String>
@get:OutputDirectory
abstract val out: DirectoryProperty
@TaskAction fun greet() = println(station.get())
}
tasks.register<Hello>("hello") { station = "SF" }
```

Convention plugin: build-logic/src/main/kotlin/tidepool.java-conventions.gradle.kts; lazy tasks.register, Property/Provider.

gradle.properties
org.gradle.caching=true
org.gradle.parallel=true
org.gradle.configuration-cache=true
org.gradle.jvmargs=-Xmx512m -Dfile.encoding=UTF-8
gradle/wrapper/gradle-wrapper.properties
distributionSha256Sum=<SHA-256 of the distribution>

Same job / idea	Ant (+ Ivy)	Maven	Gradle
Compile / test / package	ant compile · test · jar (your targets)	mvn compile · test · package	./gradlew compileJava · test · jar
Clean / full build	ant clean / ant (default target)	mvn clean / mvn verify	./gradlew clean / build
Install / publish	resolver:install / ivy:publish, resolver:deploy	mvn install / mvn deploy	publishToMavenLocal / publish
Show dependencies	ivy:report	mvn dependency:tree	./gradlew dependencies
Build files	build.xml + ivy.xml (+ ivysettings.xml)	pom.xml (+ .mvn/, settings.xml)	settings.gradle.kts + build.gradle.kts
Compile / runtime / test libs	Ivy confs (compile / runtime / test)	scopes compile, provided / runtime / test	implementation, api / runtimeOnly / testImplementation
Shared versions	properties, Ivy revision property	dependencyManagement + BOM import	version catalog, platform(), constraints
Shared build logic	common.xml pulled in with import	parent POM + pluginManagement	convention plugin in build-logic
Environment variants	properties + if: / unless:	profiles (-Pci)	project property (-Pci)
Several modules	targets that call subant	reactor: <modules> in an aggregator	multi-project: include() in settings
Pin the tool version	— (install Ant 1.10.18)	mvnw (mvn wrapper:wrapper)	gradlew (wrapper task)

Then → now	since
Ant antcall, Ant-Contrib → macrodef, import / if:	1.6 / 1.9.1
Ant junit → junitlauncher	1.10.3
Maven Maven 2 (EOL 2014) → 3.10 / 4 RC	3.10.0
Maven HTTP repositories → HTTPS only	3.8.1
Central OSSRH → Central Publisher Portal	2025-06-30
Tools EMMA, Cobertura → JaCoCo; FindBugs → SpotBugs	
Gradle task x << {} → doLast, tasks.register	5.0
Gradle compile, runtime → implementation, api; maven, uploadArchives → maven-publish	7.0
Gradle apply plugin: → plugins {} · allprojects → convention plugins	
Gradle Groovy DSL → Kotlin DSL (gradle init default)	8.2
Gradle ext versions → version catalogs	7.4
Gradle JDK 8 → JVM 17+ to run Gradle	9.0