

Apache Maven

Bojan Tomić
tomicb@fon.rs

Kako se pravi Java projekat (build)

- Osnovni koraci (skraćena procedura bez testiranja)
 - 1)Preuzimanje *.java fajlova sa početne destinacije (obično src folder)
 - 2)Preuzimanje odgovarajućih biblioteka I drugih resursa (*.jar) sa classpath-a
 - 3)Kompajliranje *.java fajlova u *.class fajlove
 - 4)Premeštanje *.class fajlova na željenu destinaciju (obično bin folder)
 - 5)Pakovanje *.class fajlova i drugih resursa u *.jar fajl

Apache Ant

- Alat za pravljenje Java projekta (build) Apache fondacije: <http://ant.apache.org/>
- Podrazumevani alat za pravljenje projekata u Eclipse i NetBeans okruženjima
- Proceduralan i nema podrazumevane opcije
 - Mora se definisati svaki korak
 - Mora se definisati svaka destinacija
- Podešavanja kao XML fajl (obično build.xml)

Apache Ant

```
<project name="MyProject" default="dist" basedir=". ">
  <description>
    simple example build file
  </description>
  <!-- set global properties for this build -->
  <property name="src" location="src"/>
  <property name="build" location="build"/>
  <property name="dist" location="dist"/>

  <target name="init">
    <!-- Create the time stamp -->
    <tstamp/>
    <!-- Create the build directory structure used by compile -->
    <mkdir dir="${build}"/>
  </target>

  <target name="compile" depends="init"
    description="compile the source">
    <!-- Compile the java code from ${src} into ${build} -->
    <javac srcdir="${src}" destdir="${build}"/>
  </target>

  <target name="dist" depends="compile"
    description="generate the distribution">
    <!-- Create the distribution directory -->
    <mkdir dir="${dist}/lib"/>

    <!-- Put everything in ${build} into the MyProject-${DSTAMP}.jar file -->
    <jar jarfile="${dist}/lib/MyProject-${DSTAMP}.jar" basedir="${build}"/>
  </target>

  <target name="clean"
    description="clean up">
    <!-- Delete the ${build} and ${dist} directory trees -->
    <delete dir="${build}"/>
    <delete dir="${dist}"/>
  </target>
</project>
```

Apache Ant

- Nedostaci
 - Nema podrazumevanih opcija (sve se mora definisati svaki put: zadaci, destinacije...)
 - Svaka biblioteka (*.jar) mora da se ručno skine sa Interneta i prekopira u projekat
 - Java projekti nisu međusobno kompatibilni kod različitih IDE-a, a i inače
 - Npr. Eclipse i NetBeans projekat
 - Svaki IDE ima svoj specifičan Ant script tj. build konfiguraciju
 - U projektima se NE implementiraju „najbolje prakse“

Apache Maven

- Alat Apache fondacije za pravljenje Java projekta (build), ali i upravljanje projektima (project management):
<https://maven.apache.org/>
- Dosta besplatnih knjiga i resursa na Internetu, npr. „Maven the Complete Reference“, Sonatype books.
- Podrška u Eclipse, NetBeans ali i drugim okruženjima
- Convention over configuration (sve opcije imaju podrazumevane vrednosti)

Apache Maven

- Automatsko preuzimanje biblioteka sa Maven centralnog repozitorijuma i povezivanje sa projektom
- Maven projekat je, bez izmena, kompatibilan sa svim IDE koje podržavaju Maven
- Predefinisane konfiguracije za određene tipove projekata (arhetipovi - JAR, WAR, Spring...)
- Pogodan za velike projekte (moduli)
- Project Object Model (POM), podešavanja kao XML fajl (pom.xml)

Apache Maven

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>org.codehaus.mojo</groupId>
  <artifactId>my-project</artifactId>
  <version>1.0</version>
</project>
```