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 preuzmanje 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>
```

Apache Maven

- Maven je alat sa kojim se radi iz konzole:
  - mvn install
  - mvn test
- Da bi se koristio, potrebno ga je instalirati
- Mnogi IDE za Javu imaju podršku za Maven
  - Interna instalacija (ne treba instalirati na računar)
  - GUI podrška kroz menije, prozore itd.

Apache Maven

- Maven koordinate (svaki projekat)
  - Group id (često obrnut domen: rs.ac.bg.fon.nprog)
  - Artifact id (lokalni naziv projekta: biblioteka)
  - Version (npr. 1.2.0)
- Projektni arhetipovi
  - Java aplikacija, web aplikacija, Spring aplikacija...
- Načini „pakovanja“ projekta
  - JAR, WAR, EAR, POM…

Apache Maven

- Maven radni ciklus - uprošćen („build cycle“):
  1) Proveravaju se fajlovi sa kodom običnih klasa
  2) Proveravaju se resursi projekta
  3) Proveravaju se biblioteke potrebne za kompajliranje
  4) Kompajliraju se osnovne klase
  5) Fajlovi sa kompajliranim kodom idu u /target
  6) Proveravaju se fajlovi sa kodom test klasa
  7) Proveravaju se test resursi projekta
  8) Proveravaju se biblioteke potrebne za testiranje
  9) Kompajliraju se test klase
  10) Fajlovi sa kompajliranim test kodom idu u /target
  11) Pokreću se test klase odn. Testovi
  12) Kompajlirane klase i resursi se pakuju u npr. JAR fajl u /target

Apache Maven

- Standardna struktura projekta
  - src/main/java Izvorni kod običnih Java klasa
  - src/main/resources Resursi neophodni za rad
  - src/test/java Izvorni kod Java test klasa
  - src/test/resources Resursi neophodni za testove
  - pom.xml Maven konfiguracioni fajl
  - target Putanja gde se smešta izlaz – JAR sa spakovanim projektom, rezultati testova, dokumentacija

Apache Maven

- Fajl pom.xml „nasleđuje“ konfiguraciju:
  - Od tzv. „super pom-a“ koji ima već predefinisanu konfiguraciju u vidu najboljih praksi
  - „convention over confirguration“
  - Može i od drugog pom.xml fajla („parent pom“)
  - Zato je pom.xml kratak
- U samom pom.xml fajlu se konfigurišu samo stvari specifične za taj projekat

Apache Maven

- Fajl pom.xml često sadrži i :
  - Opšte o projektu: opis projekta, link ka VCS repozitorijumu, ko su programeri
  - Properties koji konfigurišu različite stvari:
    - project.build.sourceEncoding (karakteri, npr. - UTF-8)
    - maven.compiler.target (kompajlirana verzija Jave – default 1.6, ali se može staviti 1.8)
    - maven.compiler.source (source verzija Jave – default 1.6, ali se može staviti 1.8, 1.9 isl.)

Apache Maven

- Rad sa drugim bibliotekama
  - Pronalaženje u Maven Central repozitorijumu preko 3 koordinate: https://search.maven.org/
  - Ili u nekom drugom Maven repozitorijumu
  - Unošenje u pom.xml kao „dependency“

```
<dependency>
    <groupId>dom4j</groupId>
    <artifactId>dom4j</artifactId>
    <version>1.6.1</version>
</dependency>
```

Apache Maven

- Rad sa drugim bibliotekama (nastavak)
- Version (tag u okviru „dependency“ taga)
  - Tačna verzija koja se traži
    - 1.6.1 - samo verzija 1.6.1 biblioteke se traži za projekat
  - Raspon verzija ([od,do] (od, do) (od,do] ili [od, do))
    - (, 1.6.1] - znači sve verzije pre 1.6.1 ali uključuje i 1.6.1
    - [0.5, 1.6.1) - od verzije 0.5 do 1.6.1 ali NE uključuje i 1.6.1

Apache Maven

- Rad sa drugim bibliotekama (nastavak)
- Scope (dodatni tag u okviru „dependency“ taga)
  - compile – default vrednost za scope ako nije naveden. Potreban radi kompajliranja i pokretanja. Pakuje se u izlazni fajl (npr. WAR)
  - provided – biće obezbeđen u runtime okruženju, npr. web server, enterprise container (npr. Spring). Ne pakuje se u izlazni fajl.
  - runtime – potreban za pokretanje i testiranje, ali ne kompajliranje (npr. JDBC drajver...)
  - test – potrebna samo za testiranje (Junit, TestNG...)
  - system – slično kao provided, samo mora da se obezbedi apsolutna putanja u lokalnom file sistemu.

Apache Maven

- Rad sa drugim bibliotekama (nastavak)
  - Maven onda prekopira traženu biblioteku i sve zavisne biblioteke u LOKALNI Maven repozitorijum na disku
  - (Windows) c:/Users/####/.m2
  - (Linux) /home/####/.m2

Apache Maven

- Maven životni ciklusi (lifecycle)
  - Clean Lifecycle (clean)
  - Default Lifecycle (default)
  - Site Lifecycle (site)
- Svaki ciklus se sastoji iz faza(phases) i ciljeva (goals) koji obuhvataju te faze
- Svaki cilj ima svoj maven plug-in koji ga izvršava
- Plug-inovi se mogu konfigurisati ali i menjati

Apache Maven

- Maven clean životni ciklus, cilj clean, faze:
  1)pre-clean
  2)clean
  3)post-clean
- Briše target folder odnosno sve što je generisano.

Apache Maven

- Maven site životni ciklus, ciljevi site:site i site:deploy, faze:
  1)pre-site
  2)site
  3)post-site
  4)site-deploy
- Generiše mini sajt sa podacima o projektu.

Apache Maven

- Maven default životni ciklus, faze:
  - validate
  - ...
  - compile
  - ….
  - test
  - ...
  - deploy
- Konkretne faze zavise od pakovanja (JAR, WAR, EAR...)
- https://books.sonatype.com/mvnref-book/reference/lifecycle-sect-structure.html

Apache Maven

- Maven default životni ciklus za JAR pakovanje, faze i ciljevi (u zagradi):
  1)Process-resources (resources:resources)
  2)Compile (compiler:compile)
  3)Process-test-resources (resources:testResources)
  4)Test-compile (compiler:testCompile)
  5)Test (surefire:test)
  6)Package (jar:jar)
  7)Install (install:install)
  8)Deploy (deploy:deploy)
- Ako se pozove neki cilj, izvrše se i svi ovi pre njega!!!