PROGRAMMING ADVANCED

MAVEN

HOE KAN MAVEN ONS WERK AANGENAMER MAKEN?

- Wat is het, wat doet het, welke problemen lost het op?
  - biedt ons uniforme project-structuur
    - maakt convention over configuration mogelijk
  - vnl. om onze projecten te bouwen
    - … en alle afhankelijkheden te beheren (repository)

WAAR VIND IK HET?

- http://maven.apache.org

HOE INSTALLEER IK HET?

- uitpakken zip bestand
- omgevingsvariabele M2_HOME aanmaken en toevoegen aan pad
- verifiëren op de CLI

Voor voorbeelden, zie cursus - pagina 5 en 6

ONS 1E MAVEN PROJECT

- De ganse project-structuur wordt gegenereerd via 1 simpel commando
- Er bestaan templates - of je kan je eigen maken - die je project configureren met 3rd-party afhankelijkheden (bvb een game-engine, een gui-toolkit, ..)
- mvn archetype:create -DgroupId=be.pxl -DartifactId=FirstApplication

ONS 1E MAVEN PROJECT

- Volgende structuur wordt voor ons gegenereerd
- mvn archetype:create -DgroupId=be.pxl -DartifactId=project

```
project
  +--src
      +--main
          +--java
          +--resources
      +--test
          +--java
          +--resources
  +-- target
        +--classes
        +--test-classes
  pom.xml
```

POM

- POM (project object model)
  - in hoofdmap van het project
  - beschrijft het project (structuur, afhankelijkheden, plugins voor de lifecycle, ..)

DEPENDENCIES

- Dependencies zijn verwijzingen naar kant-en-klare bibliotheken(jars) die je kan gebruiken in je project
- Meestal zijn deze te vinden in remote repositories (bvb. voor Spring, Hibernate, ..)
- Maar je kan deze ook krijgen van je eigen teams (bvb. iemand in het team heeft een pdf-rapportage-engine gemaakt en deze gedeeld als een library
- Dependencies kan je downloaden uit een (remote) repository
- Dependencies zijn **transitief**
- **Dependency maven configuratie opzoeken op** http://mvnrepository.com

DEPENDENCY TOEVOEGEN

- Dependencies voeg je toe in de pom.xml
- **Dependency maven configuratie kan je opzoeken op** http://mvnrepository.com

```
<dependency>
    <groupId>com.sparkjava</groupId>
    <artifactId>spark-core</artifactId>
   <version>2.3</version>
</dependency>
```

SCOPE

- Dependencies kunnen scoped zijn
- Scope duidt aan *waar* deze dependencies gebruikt worden
- Voornaamste scopes:
  - compile
  - provided
  - runtime
  - test

PHASES & GOALS

Clean Lifecycle
pre-clean
clean
post-clean

Default Lifecycle	
validate	test-compile
initialize	process-test-classes
generate-sources	test
process-sources	prepare-package
generate-resources	**package**
process-resources	pre-integration-test
compile	integration-test
process-classes	post-integration-test
generate-test-sources	verify
process-test-sources	**install**
generate-test-resources	**deploy**
process-test-resources	

Site Lifecycle
pre-site
site
post-site
site-deploy

PLUGINS

- voegen extra functionaliteit toe aan lifecycle events
- overzicht: https://maven.apache.org/plugins/index.html
- http://www.eclipse.org/jetty/documentation/current/jetty-maven-plugin.html

VOORNAAMSTE COMMANDO'S

- mvn:archetype:generate
- mvn package
- mvn clean
- mvn compile
- mvn install
- mvn test
- OF... mvn clean compile install

GLOBAL & REMOTE REPOSITORIES

- globale repository verwijst standaard naar http://repo1.maven.org/maven2
- modules opzoeken kan je via http://mvnrepository.com
- er bestaan ook custom repositories (bvb. JBOSS Hibernate)
- bijkomende repositories voegen we toe in ofwel
  - pom.xml (project specifiek)
  - settings.xml (globaal)

GLOBAL & REMOTE REPOSITORIES

- voorbeeld:

```
<dependency>
    <groupId>com.sparkjava</groupId>
    <artifactId>spark-core</artifactId>
    <version>2.3</version>
</dependency>
```

LOKALE REPOSITORY

- soort lokale cache
  - je hoeft niet telkens alle dependencies van het internet te downloaden
- lokatie: .m2 folder (hidden)
- eigen artifacts installeren: `mvn install`

EIGEN REPOSITORY VOOR DE ONDERNEMING

- we willen onze code niet beschikbaar stellen op het internet
- we willen onze code wel delen binnen ons bedrijf
- Meestgebruikte: Artifactory, Nexus
- installeren in eigen repository door mvn deploy (zie cursus p.26)

PROPERTIES & RESOURCE FILTERING

- De pom kan geconfigureerd worden mbov properties, dewelke verder in de pom gebruikt kunnen worden
- omgevingsvariabelen + systeemvariabelen
- zelf-gedefinieerde properties

UITVOERBARE JAR BESTANDEN

- om af te leveren aan de eindgebruiker
- uitvoerbaar bestand
- manifest bevat de naam van de main class + afhankelijkheden
- creatie adhv maven plugins
  - maven-jar-plugin
  - maven-dependency-plugin

WHAT ELSE?

- http://mvnrepository.com/
- http://www.sonatype.org/nexus/go/
- https://www.jfrog.com/open-source/

HOE MAVEN GEBRUIKEN IN INTELLIJ IDEA (DEMO)