Apache - Maven

Java-Erstellung auf Plugin-Basis

Martin Hoffmann

Übersicht

- Was ist Maven
- Alternative: Ant
- Arbeitsweise von Maven
  - Standard
  - Eigene Konfigurationen & Plugins
  - Arbeiten im Team
  - Aufteilen von Projekten in Unterprojekte
- Beispiele
- Literatur
- Quellen
- Fragen

Was ist Maven

„A maven (yi.=meyvn) is an experienced or knowledgeable person, such as an expert or freak.“ (maven.apache.org)

- Entwickelt um den Erschaffungsprozess des Jakarta Turbine Projektes zu vereinfachen

Maven Zielsetzung:

- Den Erstellungsprozess vereinfachen
- Einheitlichen Umgebung für verschiedene Projekte
- Unterstützung der Verwaltung von Quelletextversionen, Abhängigkeiten, Tests
- Einfache und schnelle Nutzung neuer Pluginversionen

Es gibt doch Ant, wieso noch Maven?

Kurzer Einblick in die Arbeitsweise von Ant

- Erstellen einer *build.xml* in der Ziele beschrieben werden
- Bietet Möglichkeiten um Logs und Dokumentationen zu erstellen
- Skripte

Problem:

- Für jedes Projekt muss eine eigene build.xml erstellt werden
- Überladen von CVS durch jars

Arbeitsweise von Maven

- Standardkonfigurationen
- An das eigene Projekt anpassen
- Eigene Plugins entwickeln
- Teamwork
- Unterprojekte

Standard Projektstruktur

```
/
+- src/
|  +- main/
|  |  +- java/
|  |  |  +- ...
|  |  +- resources/
|  |     +- ...
|  +- test/
|  |  +- java/
|  |  |  +- ...
|  |  +- resources/
|  |     +- ...
|  +- site/
|     +- xdoc/
|        +- ...
+- target/
|  +- ...
+- project.xml
+- README.txt
+- LICENSE.txt
```

Die Plugins

- Beinhalten Haupt- und Zwischenziele
- Verwalten Abhängigkeiten
- Beschreiben ihre Dokumentation

Beispiele:

jar
java:compile

Einige existierende Plugins

- Eclipse
- Ant
- docbook in xdoc
- Erstellen von FAQ für xdoc
- xdoc aus und in html
- Erstellen von pdf aus xdoc
- war/jar
- Wizard
- genapp
- Tomcat

„Mir reicht das nicht, ich mach gern mein eigenes Ding“

- Maven an das eigene Projekt anpassen
- eigene Plugins erstellen

Konfigurationsdateien

- {projekt.home}/maven.xml
  - Eigene Ziele für das Projekt erstellen
- {projekt.home}/project.xml
  - Projekteigenschaften beschreiben
- {projekt.home}/project.properties
  - Plugins konfigurieren
- {user.home}/build.properties
  - Allgemeine Systemkonfigurationen
- {projekt.home}/build.properties
  - Systemkonfigurationen für das Projekt

Bei eigenen Plugins

- {projekt.home}/plugin.jelly
  - Enthält Jelly-Skript für das Plugin
- {projekt.home}/plugin.properties
  - Beinhaltet Standardwerte

Beispiel für projekt.xml

```
<project>
  <groupId>sample</groupId>
  <artifactId>sample-echo</artifactId>
  <version>1.0-SNAPSHOT</version>
  <dependencies>
    <dependency>
      <groupId>log4j</groupId>
      <artifactId>log4j</artifactId>
      <version>1.2.8</version>
    </dependency>
  </dependencies>
  <build>
    <sourceDirectory>src/main/java</sourceDirectory>
    <unitTestSourceDirectory>src/test/java</unitTestSourceDirectory>
    <resources>
      <resource>
        <directory>src/main/resources</directory>
      </resource>
    </resources>
    <unitTest>
      <includes>
        <include>**/*Test.java</include>
      </includes>
    </unitTest>
  </build>
</project>
```

Eigene Plug-ins erstellen

- Wie ein normales Projekt
  $ maven plugin

- Jelly-Skript für Beschreibung
- Plugin in Maven-Stammverzeichnis installieren
- Plugin veröffentlichen und bekannt machen

wichtig: an die Doku denken

„Du bist nicht allein auf der Welt“

Teamwork:

- CVS und Konfigurationen
- Plugins für andere zugängig machen

Unterstützte Quelletextverwaltungssysteme

- CVS
- Subversion
- ClearCase (teileweise)
- Perforce (teileweise)
- StarTeam (teileweise)
- Visual SourceSafe (geplant)

Gemeinsame Konfigurationen

- Die project.xml legt Eigenschaften des Projektes fest
- Projekt kann unterschiedliche Versionen haben
- Funktionen in Plugins kapseln und allgemein zugänglich machen

Plugins mit anderen Teilen

- durch gemeinsames Pluginverzeichnis
- Plattformen benutzen (z.B.: SourceForge, ibiblio.org, ...)

Schritt für Schritt

Unterprojekte und „Vererbung“ von Konfigurationen

Unterprojekte

- Jedes Verzeichnis mit einer project.xml wird als eigenes Projekt angesehen
- Konfigurationsdateien können mit Platzhalter versehen werden

Literatur und Quellen

Literatur:

- Maven: A Developer's Notebook (O'Reilly, Juli 2005)

Quellen:

maven.apache.org

Schluss

Noch Fragen?