Maven Installation Guide

Dipl.-Inf. Manfred Wolff
in Zusammenarbeit mit der
NEUSTA GmbH

Installation und Grundkonfiguration von Maven

Die aktuelle Version von Mavin ist zur Zeit die Version 1.0. Beim Schreiben dieses How-To war der Release-Candidate 3 (RC3) gerade veröffentlicht worden. Auf der Homepage von Maven unter http://maven.apache.org kann in der Download-Sektion die aktuelle Version von Maven heruntergeladen werden. Falls Maven unter Windows betrieben werden soll, empfehle ich nicht den Windows-Installer zu benutzen, sondern die gezippte Version.

Maven kann in einem Verzeichnis der Wahl entpackt werden. Dieses Verzeichnis bezeichne ich im folgenden als maven.home. Maven verlangt zwei kleine Anpassungen:

1. Setzen des maven.home Verzeichnisses. Hierfür wird die Environment-Variable MAVEN_HOME gesetzt[1].

Linux:
MAVEN_HOME=/opt/maven-1.0-rc3
Windows:
SET MAVEN_HOME=[Lfw:]\Programme\maven-1.0-rc3

2. Erweiterung des Suchpfades. Dabei wird die PATH-Variable um den Pfad erweitert, in dem sich die Startscripts für Maven befinden.

Linux:
PATH=${PATH}:$[MAVEN_HOME}/bin
Windows:
SET PATH=%PATH%;%MAVEN_HOME%\bin

Ausprobiert werden kann die Installation sofort auf einer Shell durch die Eingabe von `maven -v`.

Maven meldet sich dann mit der entsprechenden Versionskennung. Die Kommandozeile von Maven kennt folgende Parameter:

-D	--define arg	Definiert eine Systemvariable.
-E	--emacs	Produziert Loggingausgaben ohne „Verzierungen“.
-P	--plugin-help	Gibt Hilfeausgaben für ein bestimmtes Plugin.
-X	--debug	Produziert Debug Ausgaben
-b	--nobanner	Verhindert die Anzeige des Logos
-d	--dir arg	Setzt das Arbeitsverzeichnis (wird mit der Option -p und -f ignoriert.
-e	--execution	Zeigt den Stack-Trace bei der Ausführung an.
-f	--find arg	Setzt die Projektdatei und das Arbeitsverzeichnis, in dem die Projektdatei gesucht wird.
-g	--goals	Zeigt alle verfügbaren Ziele an.
-h	--help	Zeigt die Hilfestellung von Maven an.
-i	--info	Zeigt Systeminformationen an.
-o	--offline	Ignoriert entfernte Repositories.
-p	--pom arg	Setzt die Projektdatei.
-q	--quiet	Vermindert die Ausgaben auf der Konsole.
-u	--usage	Zeigt die Hilfestellung im Kontext des aktuellen Projekts an.
-v	--version	Zeigt Versionsinformationen an.

Der nächste Schritt ist die Einrichtung eines lokalen Repositories. Ohne auf die genaue Funktion von Repositories and dieser Stelle genau eingehen zu wollen hier einige Anmerkungen:

- Sinn von Library-Repositories ist es eine Bibliothek (`jar`-File) genau einmal zur Verfügung zu stellen. Somit können die `jar`-Archive für alle bearbeiteten Projekte genutzt werden ohne immer wieder Versionen zu kopieren.

1 Falls die Umgebungsvariable %HOME% auf dem Windowssystem nicht gesetzt ist, muss der gesamte Pfad angegeben werden. Dieser ist unter Windows 2000 und Windows XP unter [Lfw:]\Dokumente und Einstellungen\[user] zu finden.

- Es gibt verschiedene Stufen von Repositories. Auf jeden Fall gibt es ein zentrales Repository auf dem Entwicklungsrechner.

Das lokale Repository wird wie folgt angelegt:

```
Linux
$MAVEN_HOME/bin/install_repo.sh $MAVEN_HOME/repository
Windows
%MAVEN_HOME%\bin\install_repo.bat %MAVEN_HOME%\repository
```

Die erfolgreiche Installation des lokalen Repositories kann sofort überprüft werden, weil einige `jar`-Archive bereits nach Anlage sofort in das Repository überführt werden.

Einstellungen, die für alle Projekte gelten, werden in der Datei `build.properties` im eigenen Homeverzeichnis gesetzt. Hier ein Beispiel mit den wichtigsten Einstellungen (die Pfade müssen bei Bedarf auf das eigene System angepaßt werden):

```
# Maven ${user.home}/build.properties
# General local settings – overrides
# ${project.home}/build.properties and
# ${project.home}/project.properties


# proxy settings - needed for remote repository
maven.proxy.host=proxy.mwolff.org
maven.proxy.port=3129


# local dir for plugins and repository
# defaults to ${user.home}/.maven
# use local folder to avoid too many data in user.home
maven.home.local=/opt/maven-1.0-rc3


# adding remote repository
maven.repo.remote=http://www.ibiblio.org/maven
```

Damit ist die Installation und Grundkonfiguration von Mavin abgeschlossen.

Konfiguration von Maven

Grundsätzlich ist es so, dass Maven eine ganz bestimmte Verzeichnisstruktur eines Projekts fordert (siehe nächstes Kapitel). Diese Struktur ist zwar anpassbar, es empfiehlt sich aber bei Projektneuanlage genau diese Struktur zu übernehmen. Mit Hilfe der Maven-Standard-Properties ist diese Projektstruktur anpaßbar.

Maven wird mit Hilfe von verschiedenen Property-Dateien konfiguriert, die in folgender Reihenfolge abgearbeitet werden:

- `${project.home}/project.properties`
- `${project.home}/build.properties`
- `${user.home}/build.properties`

Dabei gewinnt die letzte Definition. Das bedeutet, das gleichartige Definitionen immer überschrieben werden (keine immutable Semantik, wie bei Ant Property-Definitionen).

Nachdem die Kette der Property-Dateien abgearbeitet wurde, werden zusätzlich Einstellungen übernommen, die über System-Properties eingestellt wurde (`-D` Option).

Das Verhalten von Maven wird über die folgenden Standardeinstellungen vorgenommen. Hier eine Auswahl der wichtigsten Einstellungen. Anhand dieser Einstellungen wird auch der Verzeichnisbaum ersichtlich, den Maven standardmäßig erwartet.

Property	Beschreibung	Default
`maven.build.dest`	Das Verzeichnis, in dem die generierten Klassen (`*.class`) abgelegt werden	`${maven.build.dir}/classes`

maven.build.dir	Das Verzeichnis, in dem Maven alle Buildergebnisse, Reports etc. ablegt. Dieses Verzeichnis kann in den build.properties, die im user.home liegen verändert werden.	${basedir}/target
maven.build.src	Das Verzeichnis, in dem alle generierten Sourcen abgelegt werden	${maven.build.dir}/source
maven.conf.dir	Das Verzeichnis, in dem die Konfigurationen abgelegt werden.	${basedir}/conf
maven.docs.src	Das Verzeichnis für die Benutzerspezifische Dokumentation	${basedir}/xdocs
maven.mode.online	Zeigt an, ob man online ist, oder nicht.	true
maven.plugin.dir	Zeigt an, wo die Plug-Ins von maven zu finden sind.	${maven.home}/plugins
maven.repo.central	Zeigt den Host an, auf dem das Deployment stattfinden soll (dist:deploy)	login.ibiblio.org
maven.repo.central.directory	Das Verzeichnis, in dem die Distribution kopiert werden soll (dist:deploy)	/public/html/maven
maven.repo.local	Das lokale Repository für die jar-Dateien	${maven.home.local}/repository
maven.repo.remote	Das entfernte Repository für die jar-Dateien. Es können verschiedene Repositories mit Hilfe von Kommata separiert werden.	http://www.ibiblio.org/maven
maven.repo.remote.enabled	Gibt an, ob das entfernte Repository genutzt werden soll	true
maven.scp.executable	Die Datei für ssh-copy	scp
maven.src.dir	Das Verzeichnis für die source Dateien	${basedir}/src
maven.ssh.executable	Die Datei für den ssh Zugriff	ssh

Die project.xml

Maven basiert auf ein Project Object Model (POM). Die Objekte sind dabei:

- Projektname, -version, -beschreibung und die Lizenz.
- URL der Projekt-Homepage, URL des Problem- und Feature-Managements bzw. URL der Mailinglisten (subscribe, unsubscribe, archiv).
- Zugang zum CVS Repository.
- Projektmitglieder (comitter) und Personen, die Teile zum Projekt beisteuern (contributer).
- Abhängigkeiten der verschiedenen jar-Archive.
- Verzeichnisse für die Sourcen und Tests.
- Liste der Reports, die generiert weren sollen.

All diese Einstellungen werden in der `project.xml` definiert.

```
<project>
  <!-- Die POM-Version, welche aktuel genutzt werden soll.  -->
  <pomVersion>3</pomVersion>

  <!-- Ein eindeutiger Name für das Projekt. -->
  <id>app</id>

  <!-- Eine kurze Beschreibung für das Projekt. -->
  <name>Example Application</name>

  <!-- Die Version der Anwendung. -->
  <currentVersion>1.0</currentVersion>
```

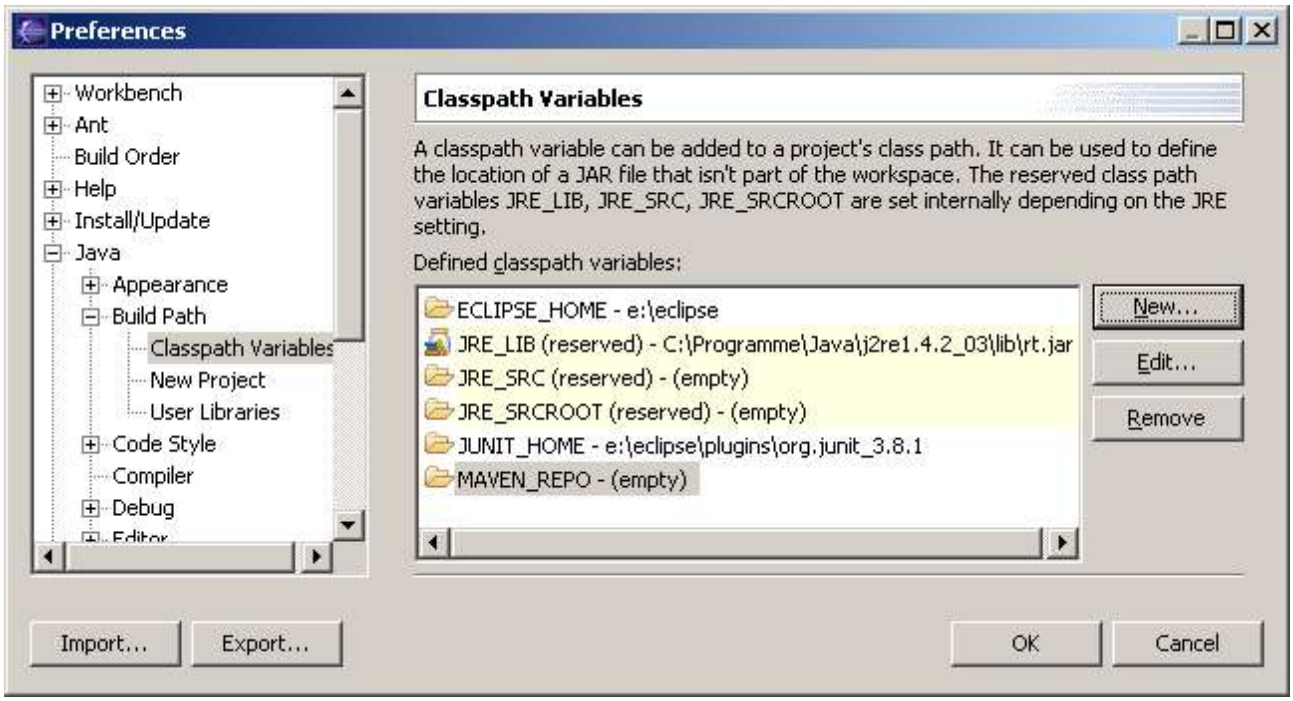


```
<!-- Details über die Organisation des Projekts.-->
<organization>
  <name>Apache Software Foundation</name>
  <url>http://www.apache.org/</url>
  <logo>http://maven.apache.org/images/jakarta-logo-blue.gif</logo>
</organization>

<!-- Das Jahr, in dem das Projekt gestartet ist. -->
<inceptionYear>2004</inceptionYear>
<package>example.app</package>
<logo>http://maven.apache.org/images/maven.jpg</logo>
<description>A collection of example projects showing how to use
   maven in different situations</description>

<!-- Eine kleine Beschreibung des Projekts. -->
<shortDescription>How to use maven in different situations</shortDescription>

<!-- Die Homepage des Projekts. -->
<url>http://maven.apache.org/reference/plugins/examples/</url>
<issueTrackingUrl>http://nagoya.apache.org/scarab/servlet/scarab/</issueTrackingUrl>
<siteAddress>jakarta.apache.org</siteAddress>
<siteDirectory>/www/maven.apache.org/reference/plugins/examples/</siteDirectory>
<distributionDirectory>/www/maven.apache.org/builds/</distributionDirectory>

<!-- Informationen über das cvs system. -->
<repository>
<connection>scm:cvs:pserver:anoncvs@cvs.apache.org:/home/
  cvspublic:maven-plugins/examples</connection>
  <url>http://cvs.apache.org/viewcvs/maven-plugins/examples/</url>
</repository>

<!-- Mailinglisten des Projekts, falls vorhanden. -->
<mailingLists/>

<!-- Entwickler des Projekts. -->
<developers/>

<!-- Jar-Archive, die dieses Projekt benutzt. -->
<dependencies/>

<!-- Build-Informationen für dieses Projekt. -->
<build>
  <nagEmailAddress>turbine-maven-dev@jakarta.apache.org</nagEmailAddress>
  <sourceDirectory>src/java</sourceDirectory>
  <unitTestSourceDirectory>src/test</unitTestSourceDirectory>
  <unitTest>
    <includes>
      <include>**/*Test.java</include>
    </includes>
    <excludes>
      <exclude>**/NaughtyTest.java</exclude>
    </excludes>
  </unitTest>
  <resources>
    <resource>
      <directory>src/conf</directory>
      <includes>
        <include>*.properties</include>
      </includes>
    </resource>
  </resources>
 </build>
</project>
```

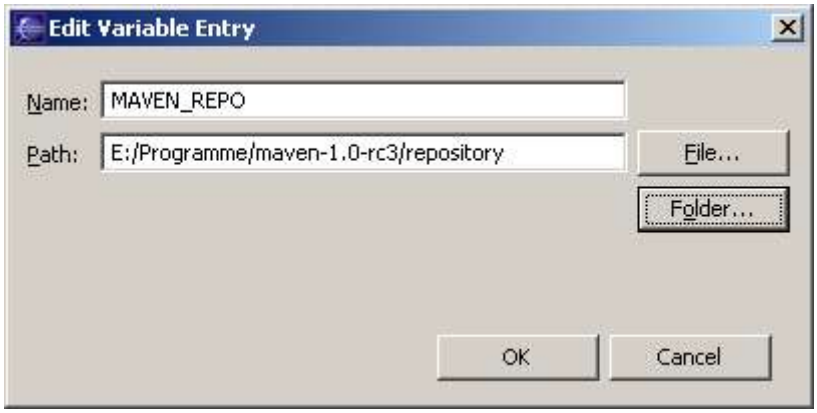
Für die project.xml existiert ein XML-Schema im marven.home Verzeichnis der jeweiligen Distribution (`maven-project.xsd`).

Integration in Eclipse

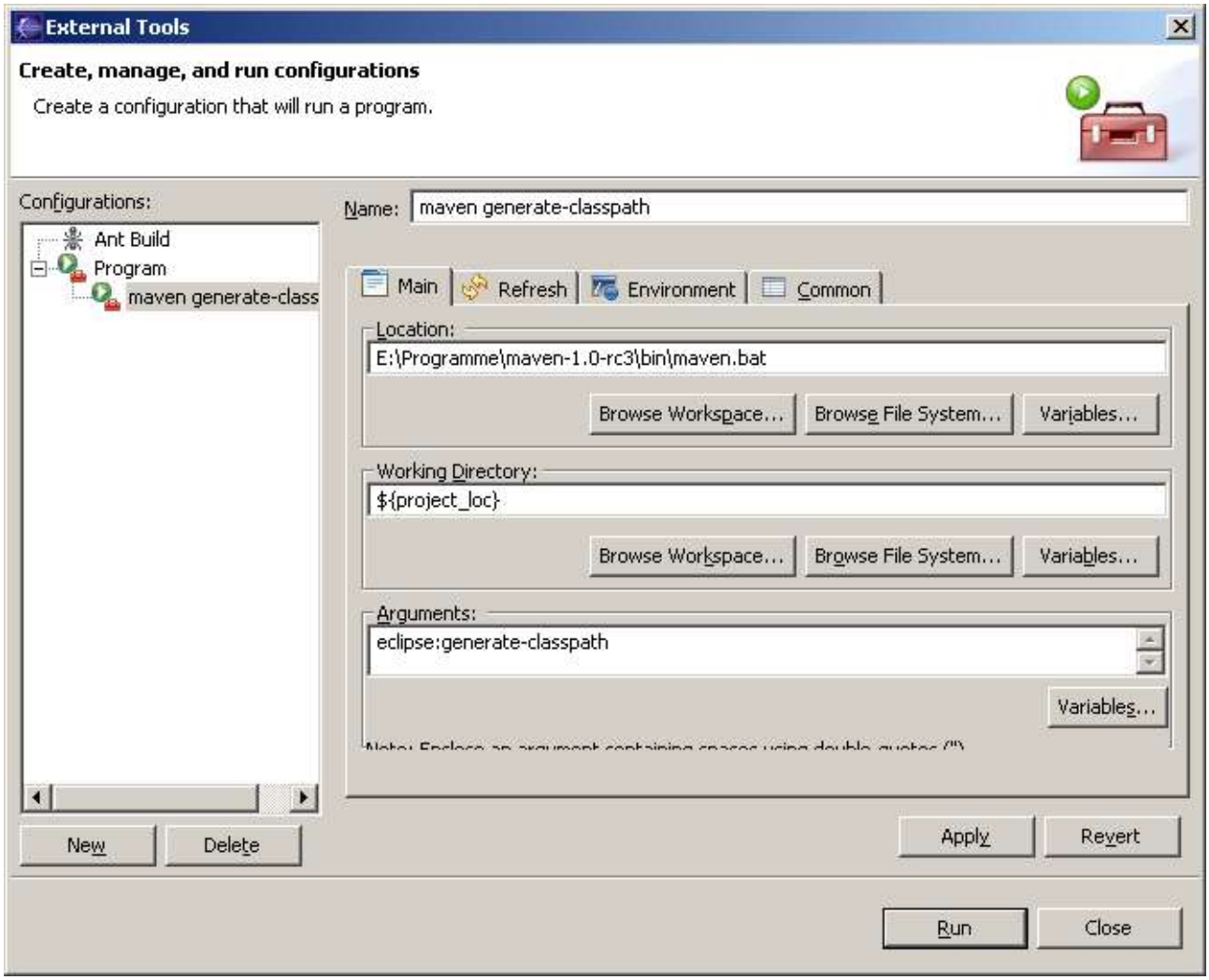
Um Inkonsistenzen zwischen den verschiedenen Repositories und dem Build-Path in Eclipse zu vermeiden, gibt es ein Eclipse-Plugin, welches die Integration von Maven in Eclipse vereinfacht. Der erste Schritte ist die Vereinbarung der Eclipse-Classpath-Variable MAVEN_REPO die anzeigt, wo das lokale Repository physisch auf der Festplatte zu finden ist. Unter Eclipse 3.0 werden diese Variablen unter dem Menüpunt **Window|Preferences** vereinbart.



Durch Betätigung des Schalters **New** geht ein weiterer Dialog auf, in dem die Variable vereinbart werden kann.



Der nächste Schritt ist die Einbindung der verschiedenen Maven Plugin Aufrufe. Dazu wird zu jedem Aufruf ein Eintrag in den externen Tools definiert.



Folgende Dinge sind in dem Dialog zu setzen:

- **Name**: Name, der in der Auswahlliste für die externen Tools erscheint.
- **Location**: Pfad zur maven Startdatei.
- **Working Directory**: ${project_loc} ist eine Eclipse-Variable, die immer auf das gerade aktuell gewählte Projekt zeigt.
- **Arguments**: Bezeichnet den Aufruf des Plugins und des Ziels.

Mit Hilfe dieser Technik können jetzt alle relevanten Maven Goals als externe Programm installiert werden.

