Repositories - Specifying an alternate location

Working at an IT development-company it has many advantages to proxy the official repositories, but some projects will not be worked on anymore or are transfered to another company. What to do? Explain everything in big documents and refer to this wiki? No, one wants to project just to work. Nobody can guarantee the future releases of Maven won't break your project. And another wish: if somebody just wants to check out if the project is good to be used, it does not want to take a day for it.

Finding a good location

The biggest pain is to get the repositories at a specific place, so it is isolated, i.e. in the project-tree. I chose "project/tools/maven-2.x.x" to have the complete Maven in. So the most logical location would be "project/tools/maven-2.x.x/repository". So I copy a fresh install of the used Maven into the tools-dir.

Edit settings.xml

In project/tools/maven-2.x.x/config you'll find settings.xml which needs to be updated to have the current repository-location in it. Just before "</settings>" we put

```
  <localRepository>file://${MVN_HOME}/repository</localRepository>
```

This works, because config.xml reads environment-variables. If you found this page, you might have tried i.e. ${project.basedir} and have noted it is not set.

Edit pom.xml

The POM should be updated to use the local repository:

```
<repositories>
  <repository>
    <id>local</id>
    <name>Local</name>
    <url>file://${project.basedir}/toolss/maven-2.0.9/repository</url>
  </repository>
</repositories>
```

Make it work out of the box

I always provide two files:

- readme.txt : contains project-description, autors, history and how to get the project up-and-running fast ("mvn eclipse:eclipse" and that kind of things).
- build.bat / build.sh : this should take care of everything to build the project. Optional: JAVA_HOME should be edited to point to a JDK, since I do not like to add a complete JDK into the tools-dir and assume a Java-developer has it installed on his machine. I just do a check if "JAVA_HOME" is set. When the customer wants a source-CD, I make an exception.

Other developers will understand what these files wil do and probably just run "build" without reading the . Understand and remember that people only want to put time in a project, when they think it's worth it. So make your script user-friendly and the txt very short, so it gives you a chance to try again.

The build-script sets the JAVA_HOME (I use "rem set JAVA_HOME=<my own location>" or "# export JAVA_HOME=<my own location>" to give a hint) and MVN_HOME (should point to "%CD%/project/tools/maven-2.x.x" or "`pwd`/project/tools/maven-2.x.x") and calls /project/tools/maven-2.x.x/bin/mvn.bat or /project/tools/maven-2.x.x/bin/mvn.sh to build the project giving the developers-build of the project. Afterwards it show which Java-version it used and which maven-location was used, so the user has feedback when the build fails.