Adding Unmanaged Dependencies to a Maven Project

Last Updated: 29 August 2013
java

Table of Contents

Some Java applications have dependencies that aren’t available in a public Maven repository. This guide shows you how to add these libraries in your application project and tell Maven how to find them.

Pick groupId, artifactId and version parameters

Let’s say your app depends on the library mylib.jar which is not in any public Maven repository. First you must define a groupId, artifactId and version for the library. These parameters may not matter to you, but Maven requires this information for all dependencies.
So let’s use:

- groupId: com.example
- artifactId: mylib
- version: 1.0 (or whatever version your lib is, if you have it versioned)

Create a local Maven repository directory

Your project root should look something like this to start with:

```
yourproject
+- pom.xml
+- src
```

Add a standard Maven repository directory called repo for the group com.example and version 1.0:

```
yourproject
+- pom.xml
+- src
+- repo
```

Deploy the Artifact Into the Repo

Maven can deploy the artifact for you using the mvn deploy:deploy-file goal:

```
mvn deploy:deploy-file -Durl=file:///path/to/yourproject/repo/ -Dfile=mylib-1.0.jar
-DgroupId=com.example -DartifactId=mylib -Dpackaging=jar -Dversion=1.0
```

Your project will contain some Maven metadata and your JAR file.

```
yourproject
+- pom.xml
+- src
+- repo
   +- com
      +- example
         +- mylib
            +- maven-metadata.xml
            +- ...
            +- 1.0
               +- mylib-1.0.jar
               +- mylib-1.0.pom
               +- ...
```

Update Pom file

Now edit your pom.xml and add this repository to your <repositories/> element. You may have to create the <repositories/> element if you dont have one.

```
<repositories>
    <!--other repositories if any-->
    <repository>
        <id>project.local</id>
        <name>project</name>
        <url>file:${project.basedir}/repo</url>
    </repository>
</repositories>
```

Now you can add this jar as a dependency the way your normally would.

```
<dependency>
    <groupId>com.example</groupId>
    <artifactId>mylib</artifactId>
    <version>1.0</version>
</dependency>
```

Commit to Git

Dont forget to add and commit your local repo folder to git.

```
$ git add repo
$ git commit -m 'adding project local maven repo, with mylib.jar 1.0'
```

The next time you push your project the dependency will resolve and your application will build without an issue. Since the local repo folder and jars are checked in with your application code they will remain private to your app.