1.mvn archetype:generate -B -DarchetypeGroupId=org.apache.maven.archetype -DarchetypeArtifactId=maven-archetype-webapp -DarchetypeVersion=1.0 -DgroupId=com.enn.scheduling -DartifactId=EnergyScheduling -Dversion=1.0-SANPSHOT -Dpackage=com.enn.scheduling
2. mvn -Dwtpversion=1.5 eclipse:eclipse

maven dependency:tree

Maven常用命令：
1. 创建Maven的普通java项目：
   mvn archetype:create
   -DgroupId=packageName
   -DartifactId=projectName
2. 创建Maven的Web项目：
    mvn archetype:create
    -DgroupId=packageName
    -DartifactId=webappName
    -DarchetypeArtifactId=maven-archetype-webapp
3. 编译源代码： mvn compile
4. 编译测试代码：mvn test-compile
5. 运行测试：mvn test
6. 产生site：mvn site
7. 打包：mvn package
8. 在本地Repository中安装jar：mvn install
9. 清除产生的项目：mvn clean
10. 生成eclipse项目：mvn eclipse:eclipse
11. 生成idea项目：mvn idea:idea
12. 组合使用goal命令，如只打包不测试：mvn -Dtest package
13. 编译测试的内容：mvn test-compile
14. 只打jar包: mvn jar:jar
15. 只测试而不编译，也不测试编译：mvn test -skipping compile -skipping test-compile
    （ -skipping 的灵活运用，当然也可以用于其他组合命令）

16. 清除eclipse的一些系统设置:mvn eclipse:clean

ps:

一般使用情况是这样，首先通过cvs或svn下载代码到本机，然后执行mvn eclipse:eclipse生成ecllipse项目文件，然后导入到eclipse就行了；修改代码后执行mvn compile或mvn test检验，也可以下载eclipse的maven插件。

mvn -version/-v 显示版本信息
mvn archetype:generate 创建mvn项目
mvn archetype:create -DgroupId=com.oreilly -DartifactId=my-app 创建mvn项目

mvn package 生成target目录，编译、测试代码，生成测试报告，生成jar/war文件
mvn jetty:run 运行项目于jetty上,
mvn compile 编译
mvn test 编译并测试
mvn clean 清空生成的文件
mvn site 生成项目相关信息的网站
mvn -Dwtpversion=1.0 eclipse:eclipse 生成Wtp插件的Web项目
mvn -Dwtpversion=1.0 eclipse:clean 清除Eclipse项目的配置信息(Web项目)
mvn eclipse:eclipse 将项目转化为Eclipse项目

在应用程序用使用多个存储库

```
<repositories>
    <repository>
        <id>Ibiblio</id>
        <name>Ibiblio</name>
        <url>http://www.ibiblio.org/maven/</url>
    </repository>
    <repository>
        <id>PlanetMirror</id>
        <name>Planet Mirror</name>
        <url>http://public.planetmirror.com/pub/maven/</url>
    </repository>
```

</repositories>

mvn deploy:deploy-file -DgroupId=com -DartifactId=client -Dversion=0.1.0 -Dpackaging=jar -Dfile=d:\client-0.1.0.jar -DrepositoryId=maven-repository-inner -Durl=ftp://xxxxxxx/opt/maven/repository/

发布第三方Jar到本地库中：

mvn install:install-file -DgroupId=com -DartifactId=client -Dversion=0.1.0 -Dpackaging=jar -Dfile=d:\client-0.1.0.jar

-DdownloadSources=true

-DdownloadJavadocs=true

mvn -e 显示详细错误 信息.

mvn validate 验证工程是否正确，所有需要的资源是否可用。
mvn test-compile 编译项目测试代码。 。
mvn integration-test 在集成测试可以运行的环境中处理和发布包。
mvn verify 运行任何检查，验证包是否有效且达到质量标准。
mvn generate-sources 产生应用需要的任何额外的源代码，如xdoclet。

本文来自CSDN博客，转载请标明出处：
http://blog.csdn.net/lifxue/archive/2009/10/14/4662902.aspx

常用命令：
mvn -v 显示版本
mvn help:describe -Dplugin=help 使用 help 插件的 describe 目标来输出 Maven Help 插件的信息。
mvn help:describe -Dplugin=help -Dfull 使用Help 插件输出完整的带有参数的目标列
mvn help:describe -Dplugin=compiler -Dmojo=compile -Dfull 获取单个目标的

信息,设置 mojo 参数和 plugin 参数。此命令列出了Compiler 插件的compile 目标的所有信息
mvn help:describe -Dplugin=exec -Dfull 列出所有 Maven Exec 插件可用的目标
mvn help:effective-pom 看这个“有效的 (effective)”POM，它暴露了 Maven的默认设置

mvn archetype:create -DgroupId=org.sonatype.mavenbook.ch03 -DartifactId=simple -DpackageName=org.sonatype.mavenbook 创建Maven的普通java项目，在命令行使用Maven Archetype 插件
mvn exec:java -Dexec.mainClass=org.sonatype.mavenbook.weather.Main Exec 插件让我们能够在不往 classpath 载入适当的依赖的情况下，运行这个程序
mvn dependency:resolve 打印出已解决依赖的列表
mvn dependency:tree 打印整个依赖树

mvn install -X 想要查看完整的依赖踪迹，包含那些因为冲突或者其它原因而被拒绝引入的构件，打开 Maven 的调试标记运行
mvn install -Dmaven.test.skip=true 给任何目标添加maven.test.skip 属性就能跳过测试
mvn install assembly:assembly 构建装配Maven Assembly 插件是一个用来创建你应用程序特有分发包的插件

mvn jetty:run 调用 Jetty 插件的 Run 目标在 Jetty Servlet 容器中启动 web 应用
mvn compile 编译你的项目
mvn clean install 删除再编译

mvn hibernate3:hbm2ddl 使用 Hibernate3 插件构造数据库