1.$M2_HOME/conf/settings.xml和~/.m2/settings.xml的区别
前者是全局范围的，整台机器上的所有用户都会直接受该配置的影响，而后者是用户范围的，只有当前用户才会受到该配置影响。
推荐使用用户范围的settings.xml

2.pom.xml文件
project是所有pom.xml的根元素，声明POM相关的命名空间及xsd元素，不是必须的，但是使用这些属性能够让第三方工具帮助我们快速编辑POM。
modelVersion指定了当前POM模型的版本，对于Maven2及Maven3来说，它只能是4.0.0。
groupId定义了项目属于哪个组，这个组往往和项目所在的组织或公司存在关联。例如：　com.mycom.myapp。
artifactId定义了当前Maven项目在组织中唯一的ID。
version指定了Hello World项目当期的版本----0.0.1-SNAPSHOT。
SNAPSHOT意为快照，说明项目还处于开发中，是不稳定的版本。
name元素声明了一个对于用户更为友好的项目名称，不是必须的。
dependencies元素可以包含多个dependency元素以声明项目的依赖。
scope元素，scope为依赖范围，如果依赖范围为test则表示该依赖只对测试有效。如果不声明依赖范围，那么默认值就是compile,表示该依赖对主代码和测试代码都有效。

3.代码目录
Maven约定：
在项目的根目录中放置pom.xml,在src/main/java目录中放置项目的主代码，在src/test/java中放置项目的测试代码。

4.由于历史原因，Maven的核心插件之一----compiler插件默认只支持编译Java1.3,因此需要配置该插件，使其支持更高的版本。

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-compiler-plugin</artifactId>
```

```
  <configuration>
    <source>1.8</source>
    <target>1.8</target>
    <encoding>utf8</encoding>
  </configuration>
</plugin>
```

5.默认打包生产的jar是不能够直接运行的，因为带有main方法的类信息不会添加到manifest中（打开jar文件中的META-INF/MANIFEST.MF文件，将无法看到Main-Class 一行）。为了生产可执行的jar文件，需要借助maven-shade-plugin。plugin元素在POM中的相对位置在<project><build><plugins>下面。可以配置mainClass，项目在打包的时会将信息放到MANIFEST.MF中。

1.依赖范围包括：compile、test、provided、runtime 、system,用来控制依赖与三种classpath(编译classpath、测试classpath、运行classpath)的关系。

表 5-1　依赖范围与 classpath 的关系

依赖范围（Scope）	对于编译 classpath 有效	对于测试 classpath 有效	对于运行时 classpath 有效	例　子
compile	Y	Y	Y	spring-core
test	—	Y	—	JUnit
provided	Y	Y	—	servlet-api
runtime	—	Y	Y	JDBC 驱动实现
system	Y	Y	—	本地的，Maven 仓库之外的类库文件

2.排除依赖、归类依赖、优化依赖

3.快照版本
当项目经过完善的测试后需要发布的时候，就应该将