# artifactId与groupId

groupId ：the unique identifier of the organization or group that created the project

groupId 是项目组织唯一的标识符，实际对应JAVA的包的结构，是main目录里java的目录结构。

artifactId ：unique base name of the primary artifact being generated by this project

ArtifactID是项目的唯一的标识符，实际对应项目的名称，就是项目根目录的名称。

举例：

<groupId>com.steven.mavendemo</groupId>   
<artifactId>mavendemo</artifactId>   
<version>0.0.1-SNAPSHOT</version>   
<packaging>jar</packaging>   
<name>mavendemo</name>

groupId

定义了项目属于哪个组，举个例子，如果你的公司是mycom，有一个项目为myapp，那么groupId就应该是com.mycom.myapp.   
  
artifacted   
定义了当前maven项目在组中唯一的ID,比如，myapp-util,myapp-domain,myapp-web等。   
  
version   
指定了myapp项目的当前版本，SNAPSHOT意为快照，说明该项目还处于开发中，是不稳定的版本。   
  
name   
声明了一个对于用户更为友好的项目名称，不是必须的，推荐为每个pom声明name，以方便信息交流。

# maven坐标

maven的世界中拥有数量非常巨大的构件，也就是平时用的一些jar,war等文件。   
maven定义了这样一组规则：   
世界上任何一个构件都可以使用Maven坐标唯一标志，maven坐标的元素包括groupId, artifactId, version，package，classifier。   
只要在pom.xml文件中配置好dependancy的groupId，artifact，verison，classifier,   
maven就会从仓库中寻找相应的构件供我们使用。那么，"maven是从哪里下载构件的呢？"   
答案很简单，maven内置了一个中央仓库的地址（http://repol.maven.org/maven2）,该中央仓库包含了世界上大部分流行的开源项目构件，maven会在需要的时候去那里下载。

举例：

<groupId>org.sonatype.nexus</groupId>   
<artifactId>nexus-indexer</artifactId>   
<version>2.0.0</version>

<packaging>jar</packaging>

groupId   
定义当前maven项目隶属的实际项目。   
groupId的表示方式与Java包名的表示方式类似，如： <groupId>org.sonatype.nexus</groupId>   
  
artifactId   
该元素定义实际项目中的一个Maven项目（模块），推荐的做法是使用实际项目的名称作为artifactId的前缀。   
如：<artifactId>nexus-indexer</artifactId>   
在默认情况下，maven生成的构件，其文件名会以artifactId作为开头，如：nexus-indexer-2.0.0.jar。   
  
packaging【可选的，默认为jar】：   
当不定义packaging时，maven会使用默认值jar。   
  
classifier:   
该元素用来帮助定义构件输出的一些附属构件。   
项目构件的文件名是坐标相对应的，一般的规则为：artifact-version.packing

# maven常用命令

编译源码

mvn compile

编译测试代码

mvn test-compile

打包

mvn package

在本地repository中安装jar

mvn install

清除产生的项目

mvn clean

上传到私服

mvn deploy

强制检查更新

mvn clean install -U

maven打包注意xml等配置文件需要单独使用<resources>标签来说明需要打进包里的文件。

# maven私服搭建