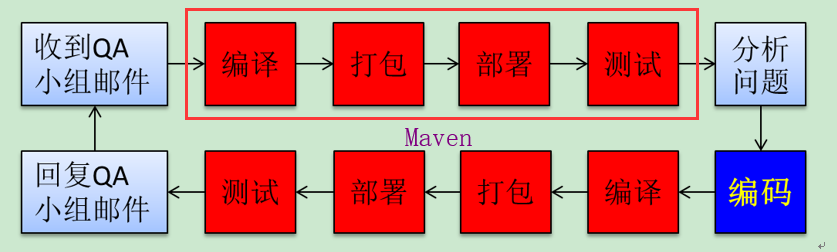
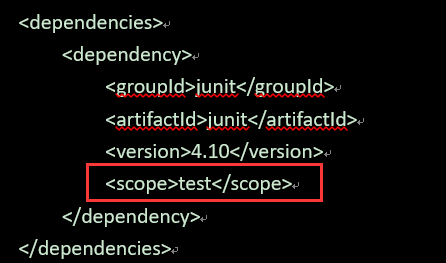
Maven：



通过Maven命令可以程序编译，打包(War,jar,pom)，部署程序(运行)，测试(运行Test类)



Scope：表示当前的jar包在什么时候才会被引用(compile，test，provided，Runtime)

Compile：编译，测试，运行时都有效(默认)

Test：测试时有效

Provided：编译，测试时有效，运行时无效(如Servlet-api，由运行时tomcat提供)

Runtime：运行时有效，编译，测试无效(如JDBC，只有在运行的时候才会去连接数据库)，只有在运行时才会抛出异常

本地仓库是指将jar包下载到本地仓库中，远程仓库是指当本地没有jar包时，需要到指定的远程仓库中下载，可以在setting.xml的mirrors标签下指定远程仓库的地址

Maven的生命周期：

Maven一共有3套生命周期：clean，Default，site，3套生命周期是独立的

Clean：clean生命周期一共包含3个阶段：

Pre-clean：clean之前需要完成的工作

Clean：移除上一次构建生成的文件

Post-clean：clean之后完成的工作

当输入mvn：clean命令时，执行的是pre-clean，clean两个阶段

Site：包含4个阶段：

Pre-site：执行需要在生成站点文档之前的工作

Site：生成项目站点文档

Post-site：生成完项目站点后，为部署做准备

Site-deploy：将生成的站点部署在特定服务器上

Site是生成站点，site-deploy是发布站点

Default：最重要的生命周期，绝大部分的工作都在Default生命周期中



在输入mvn clean deploy 表示先执行clean生命周期的clean阶段，再执行Default生命周期的deploy阶段

Maven的核心仅仅定义了抽象的生命周期，具体的任务都是交由插件完成的。

每个插件都能实现多个功能，每个功能就是一个插件(Maven的插件)目标。

Maven的生命周期与插件目标相互绑定，以完成某个具体的构建任务。

例如：compile就是插件maven-compiler-plugin的一个目标；

pre-clean是插件maven-clean-plugin的一个目标。

Maven生命周期的各个阶段都是有插件来完成的，用户可以在build标签下自定义maven插件的版本信息

Maven常用命令：

Mvn archetype ：generate 创建一个maven项目，但是一般都是工具创建

Mvn test-compile 编译测试代码

Mvn package 只是对程序打包

Mvn install 对程序打包，并将包保存到本地仓库中

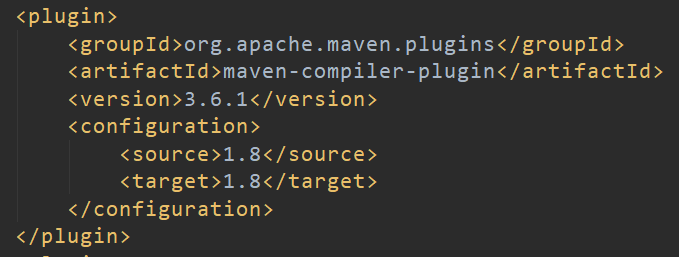
Mvn deploy 将打包后的包上传到私服上

Mvn dependency ：list 列出maven程序的依赖关系

Mvn dependency ：tree 以树状的方式列出maven程序的依赖关系

Maven常用插件：

Maven就是有一系列插件组合成的，maven自身有很多插件，比如maven-compiler-plugin插件，就是执行mvn compile命令的插件，也可以在pom文件中声明其他插件

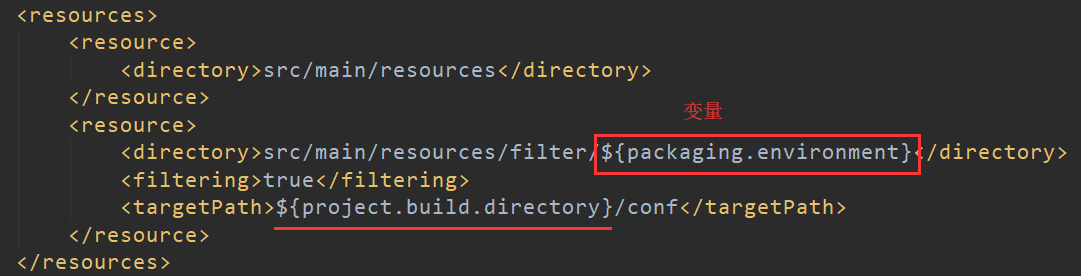


也可以对插件的属性进行修改

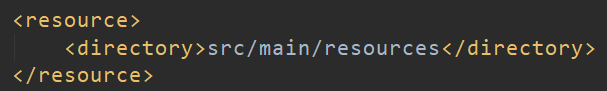
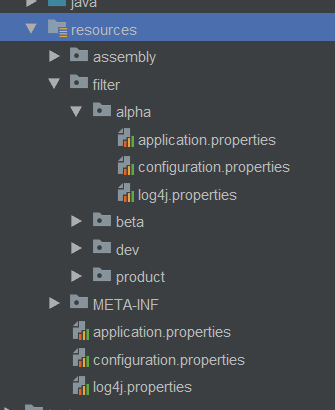
maven-resources-plugin插件：对资源文件进行过滤，可以通过mvn命令，根据命令执行加载不同的资源文件



添加的插件

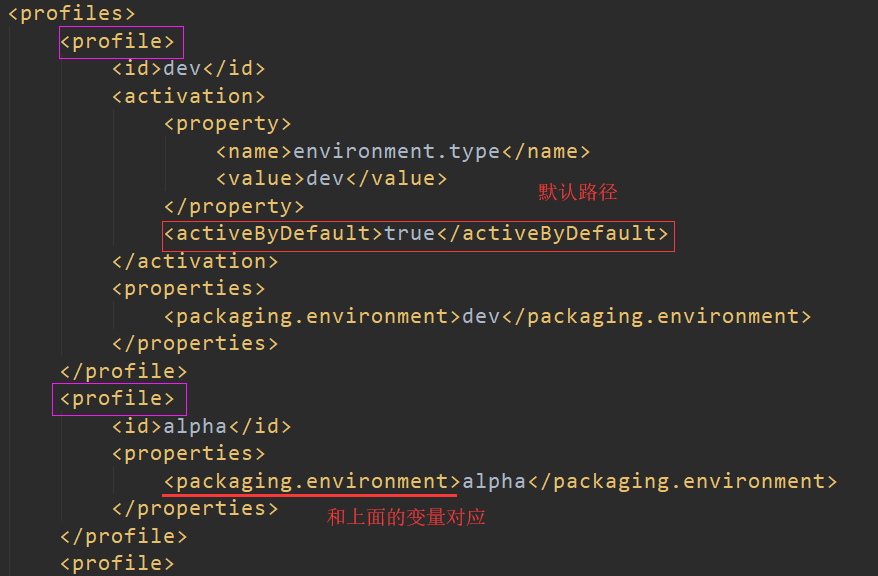


定义资源的路径



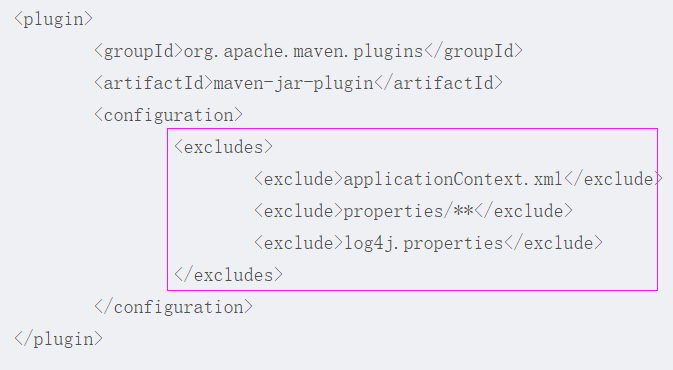
如果在输入命令是，没有指定变量的参数，默认执行第一个resource

定义完资源路径后，定义profile，不同的参数加载不同路径下的资源

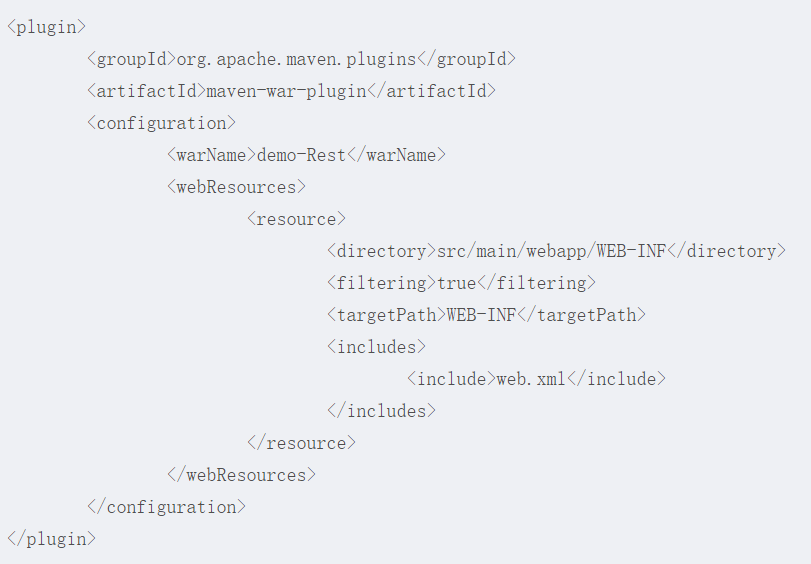


mvn install -P alpha：表示在执行命令的时候，会去加载packaging.environment=alpha下的资源

maven-jar-plugin：在对程序进行打包时，指定默写文件不被打到jar包中

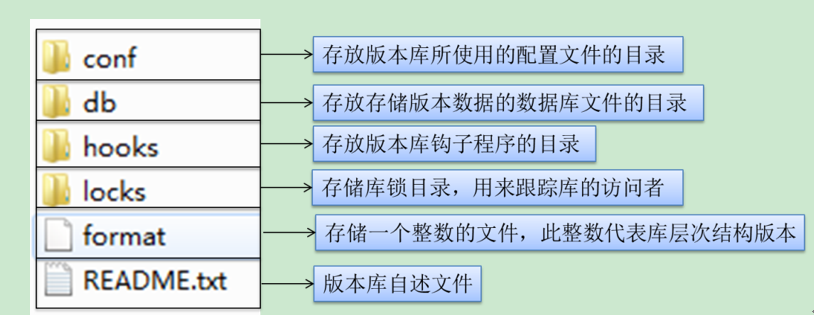


maven-war-plugin：如果是个web项目，使用此插件可以的webapps下指定的文件不打包，和maven-jar-plugin差不多，只是使用场景不同



SVN：

通过svnadmin create D:\Dev创建一个版本库(也可以通过TortoiseSVN创建)



Svn checkout svn：D:\Dev(版本库的位置) 当前文件夹纳入svn的管理

Svn add Xxx 文件纳入SVN管理

Svn commit Xxx -m 文件提交到SVN

Svn在commit之前必须update

SVN的使用和GIT差不多

Git是分布式的，SVN是集中式的

Git下载下来后，在本地就可以查看所有的log，而SVN却需要联网

SVN是将代码放在Server端，而Git是将代码放在本地

SVN适合集中管理，GIT适合分布式开发，强调个体