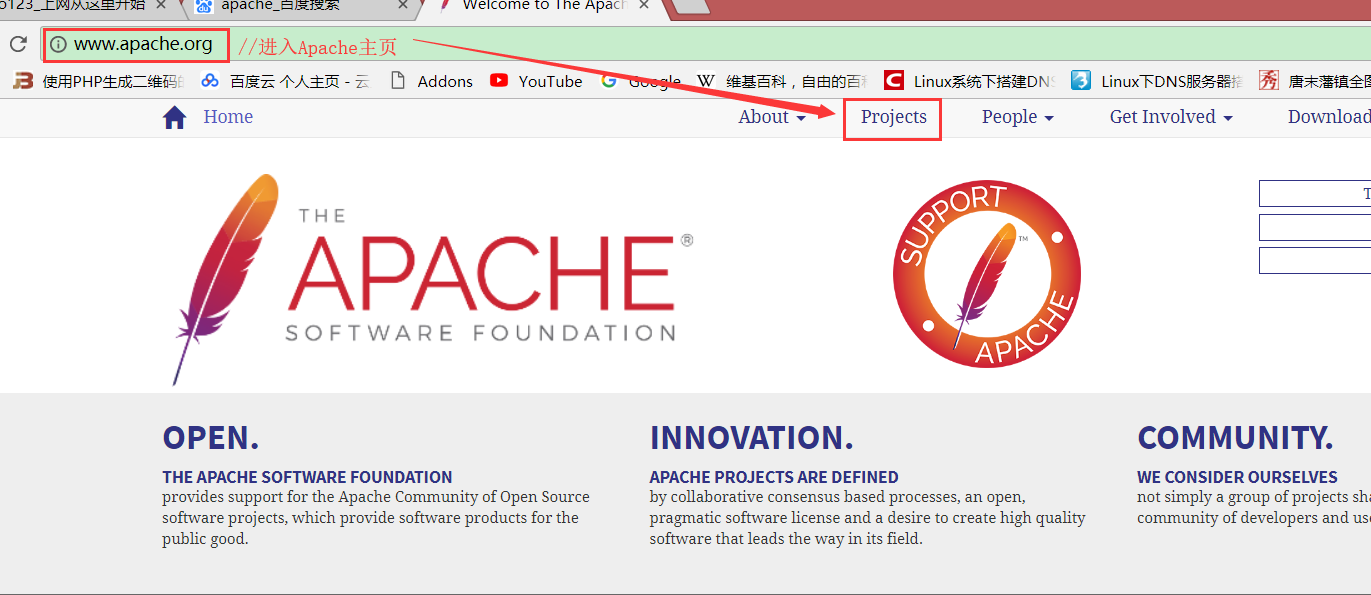
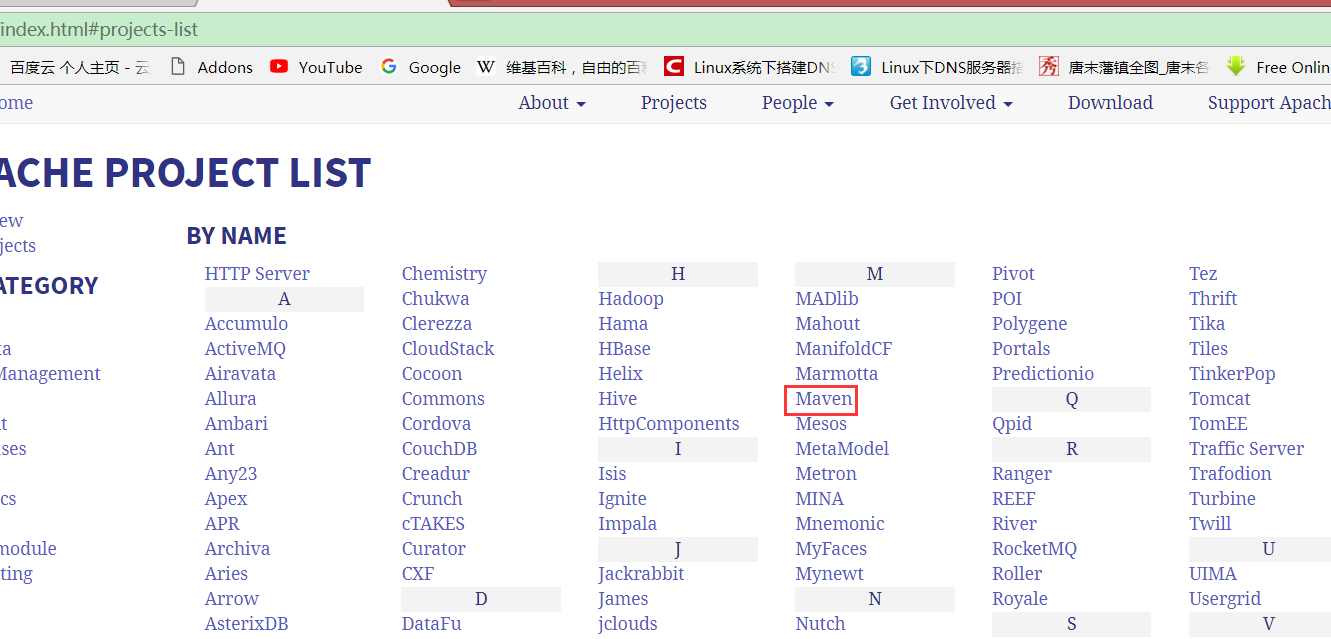
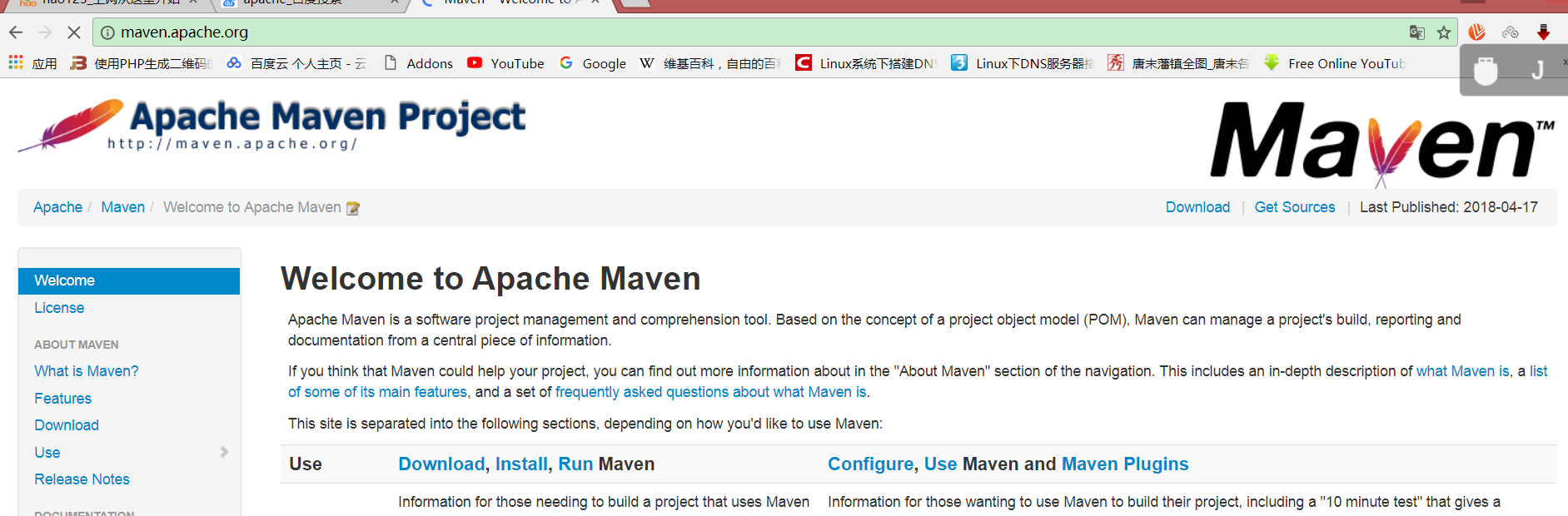
# Maven官网







# Maven的使用

<http://mvnrepository.com> ：jar包的下载地址

Maven主要用于项目中jar包的管理和项目构建管理。

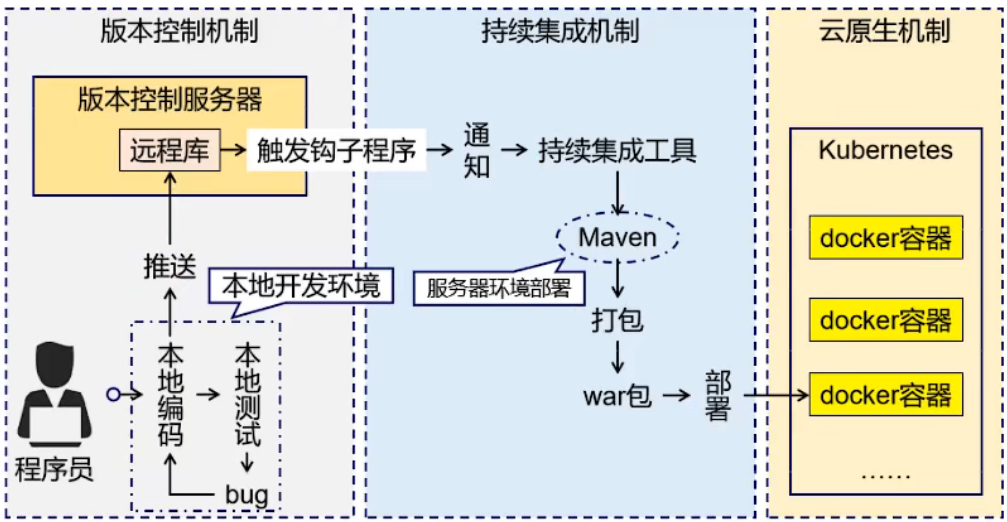
## Maven的用途

### 本地jar包依赖管理

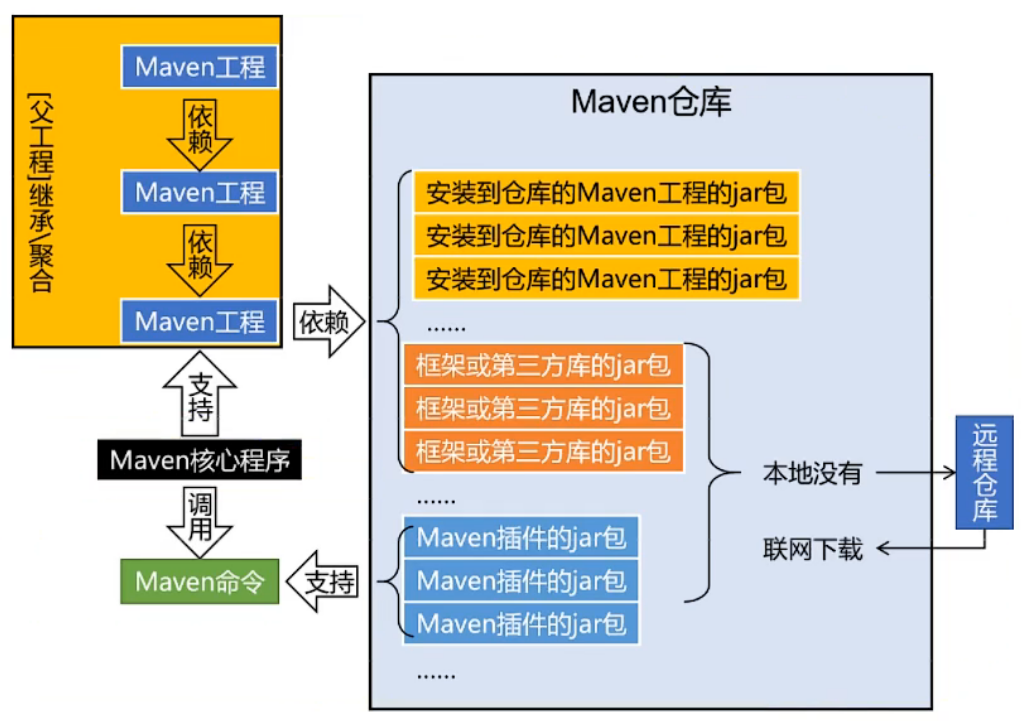
### 项目构建管理

#### 与IDE结合的项目构建

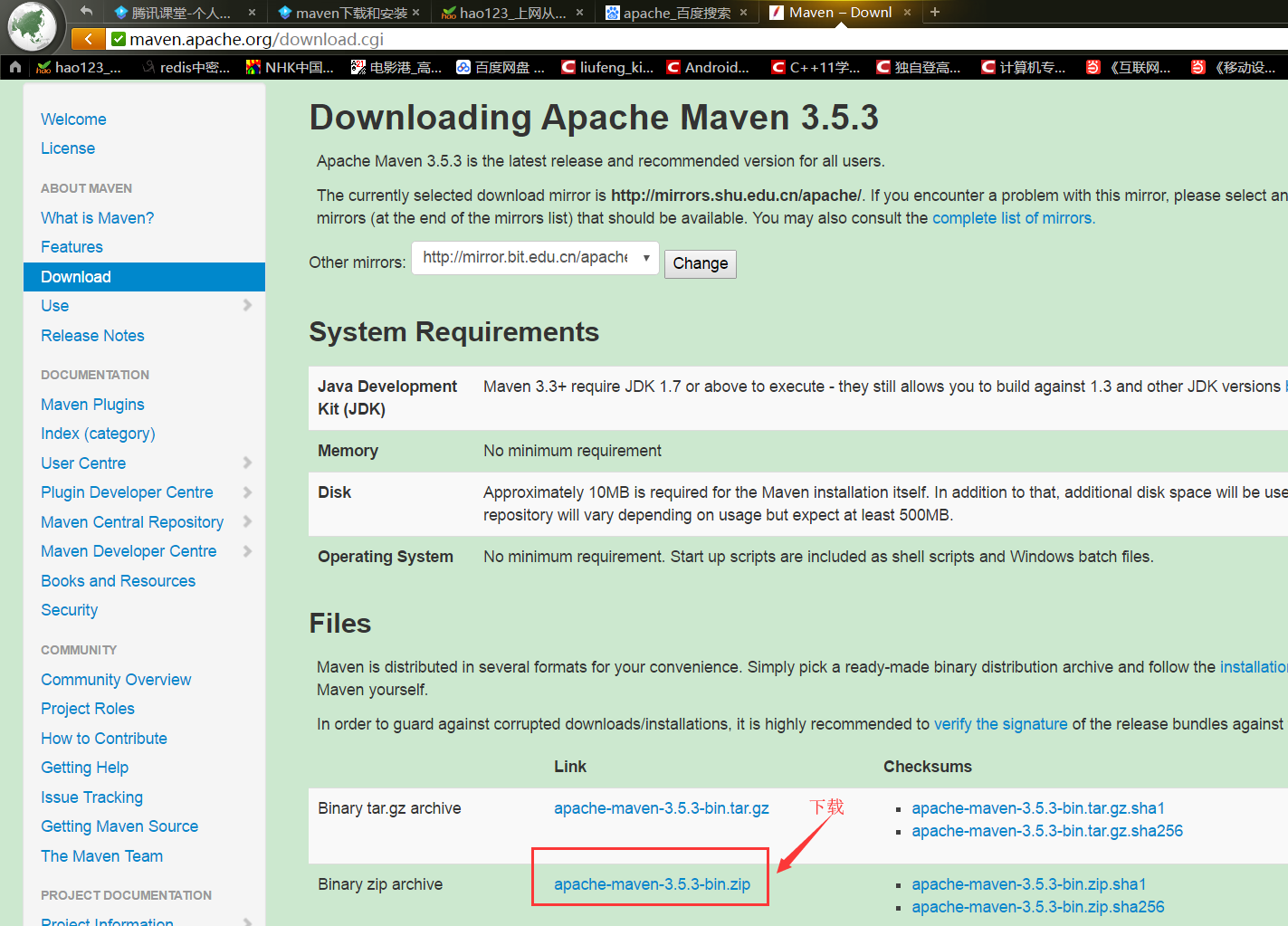
#### 脱离IDE环境的项目构建

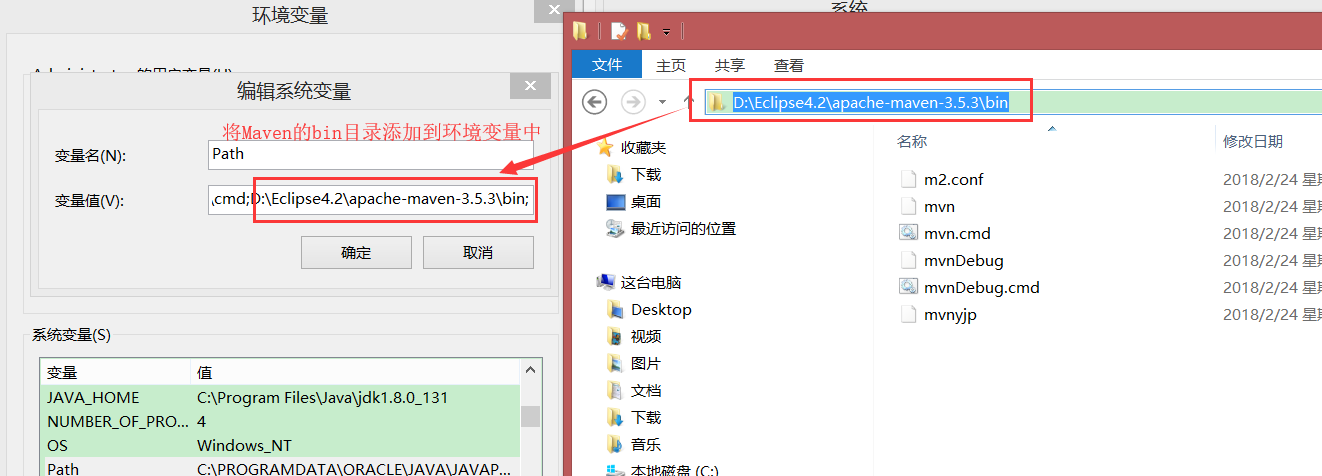


## Maven的工作机制



## 下载与安装

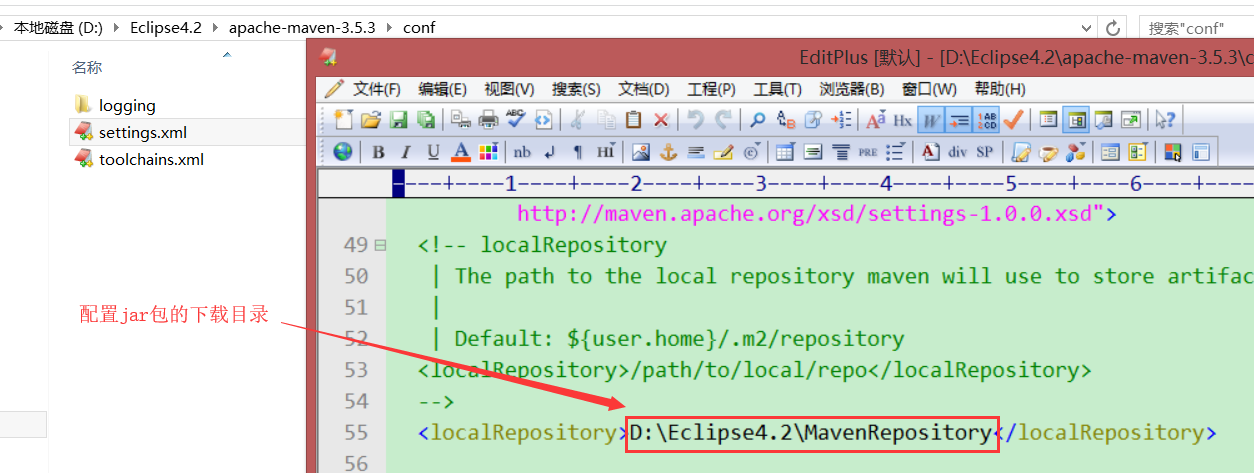




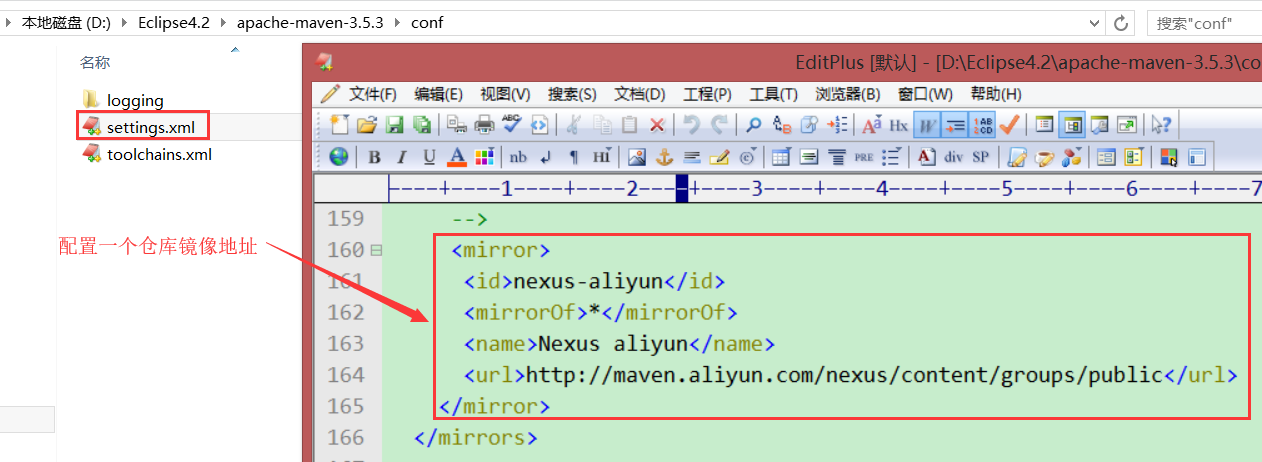
## 配置全局Maven管理环境

Maven仓库：<http://mvnrepository.com>

### 配置本地jar包存放目录



### 配置国内阿里云镜像地址



### 配置Maven的jdk编译版本

在conf/settins.xml的profiles节点下配置



### 配置JAVA\_HOME和MAVEN\_HOME

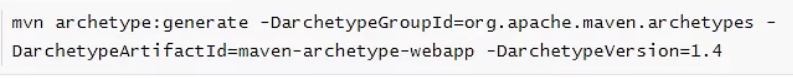
配置到环境变量即可

## 命令行创建Maven项目

### 创建普通Java工程

mvn archetype:generate

### 创建Web工程

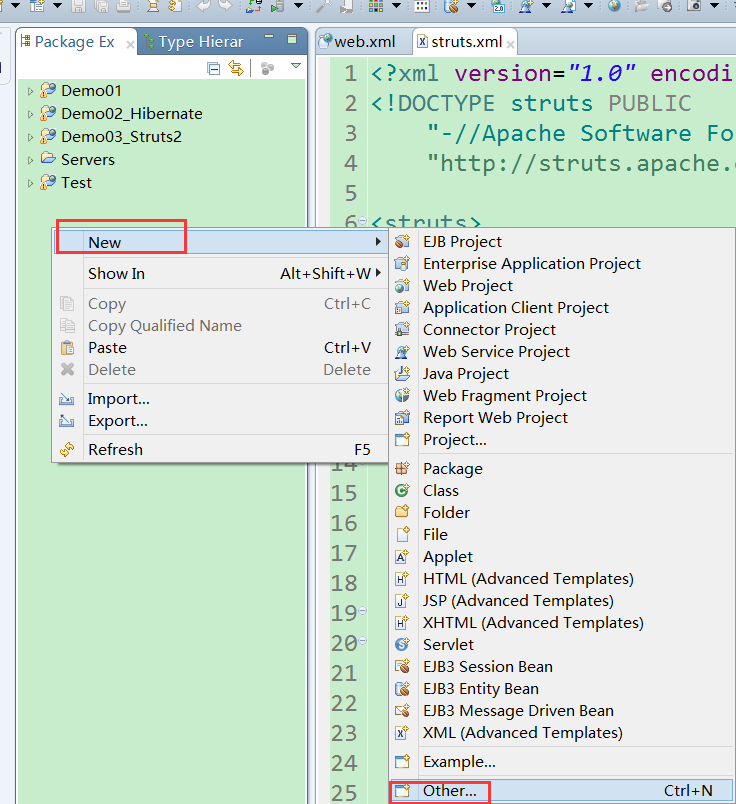


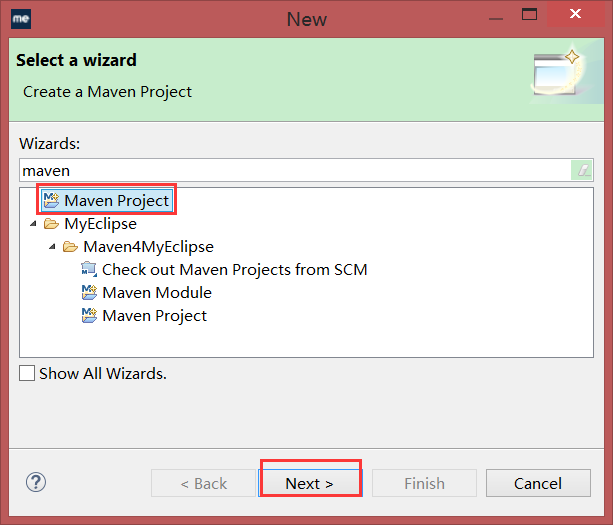


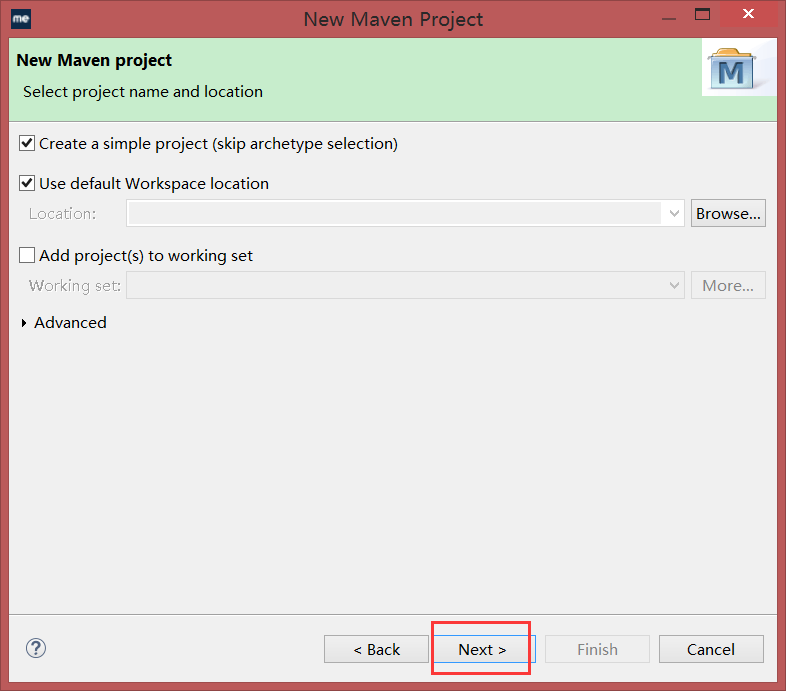
web.xml位于WEB-INF目录下

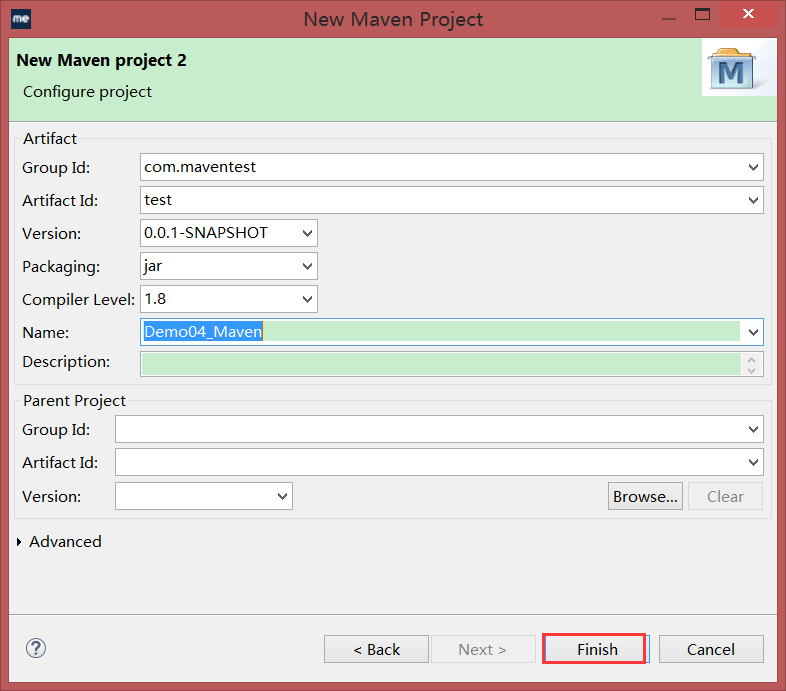
## MyEclipse创建Maven项目

MyEclipse中自动集成了一个Maven环境，但是也可以自定义配置自己下载好的。

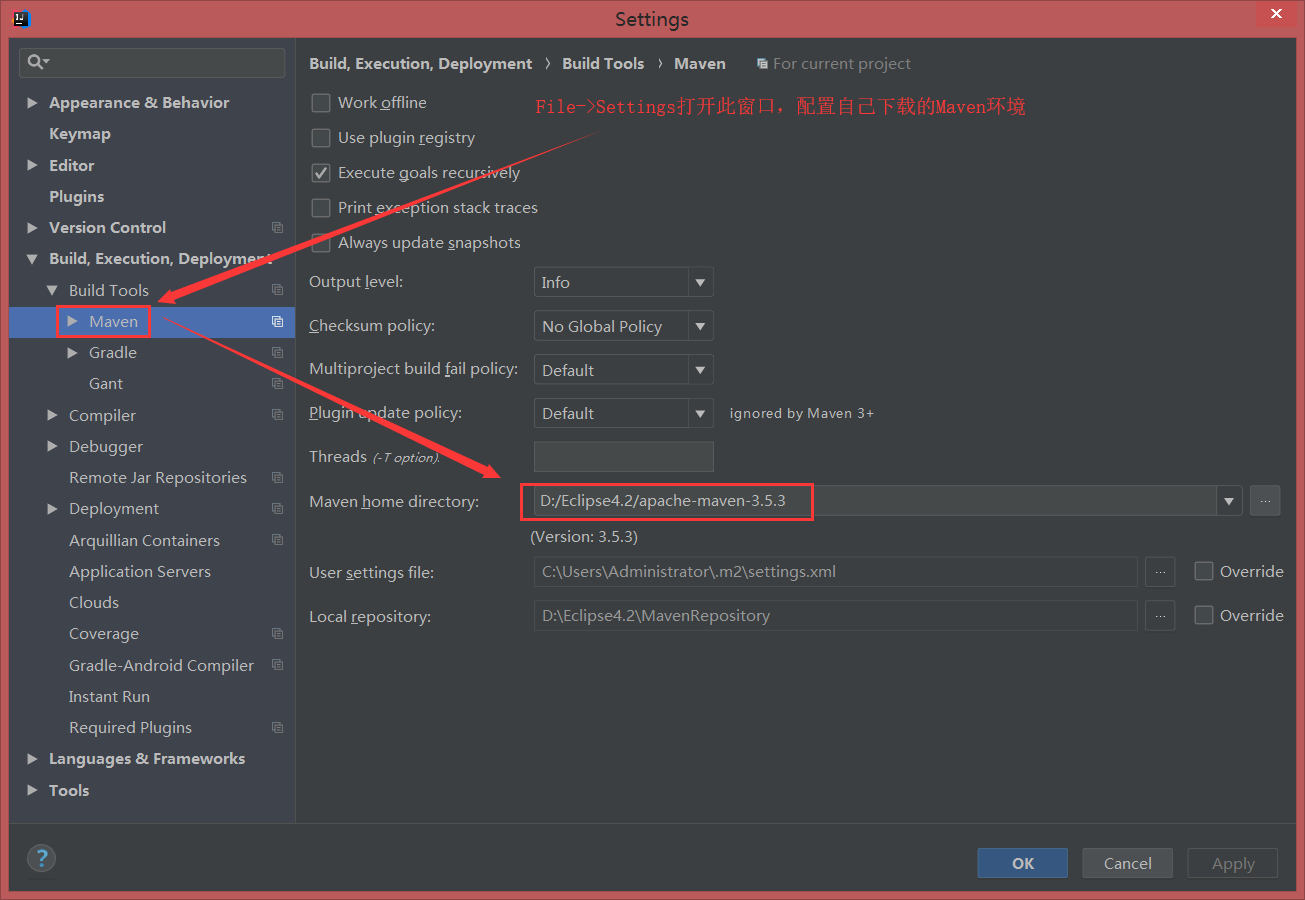


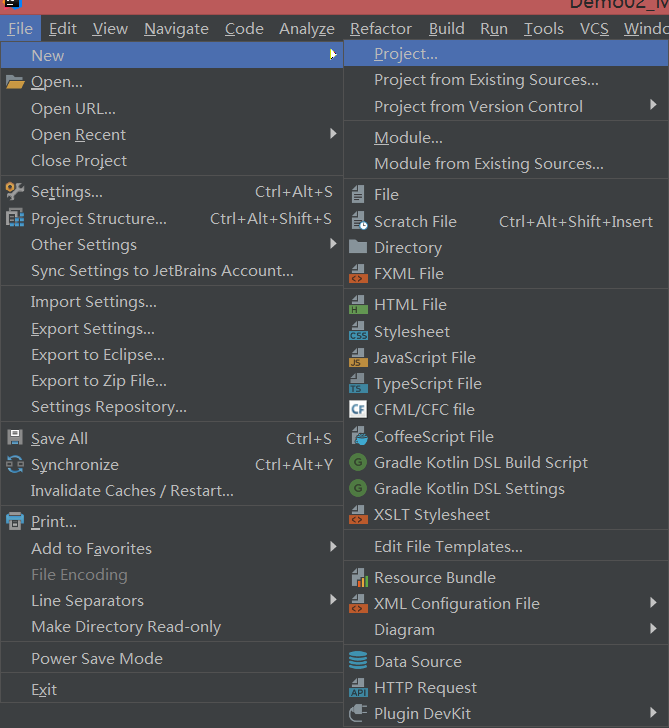


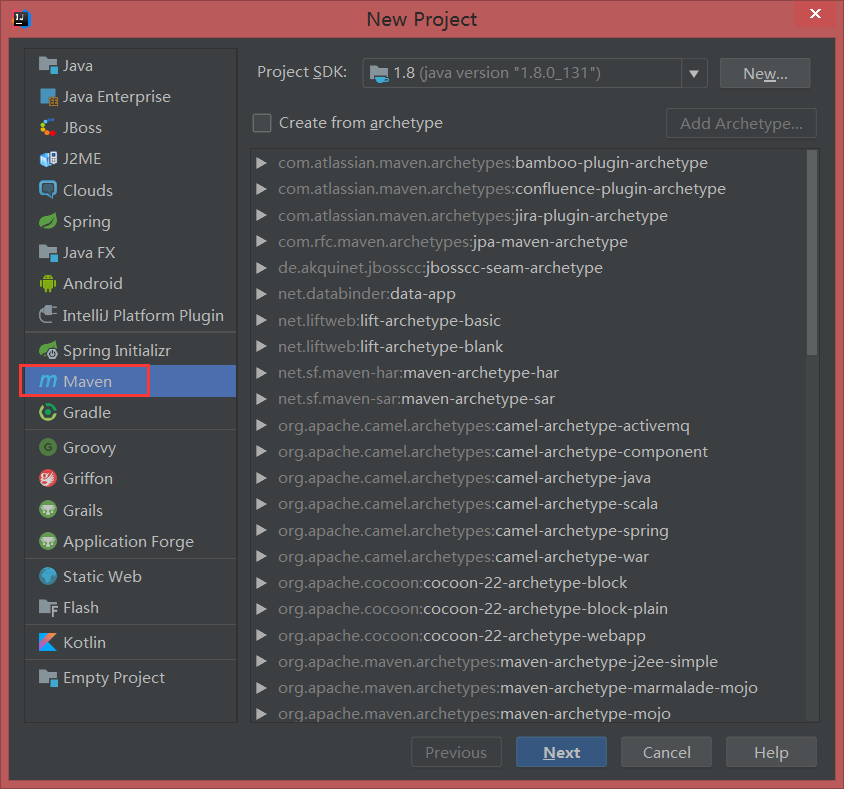




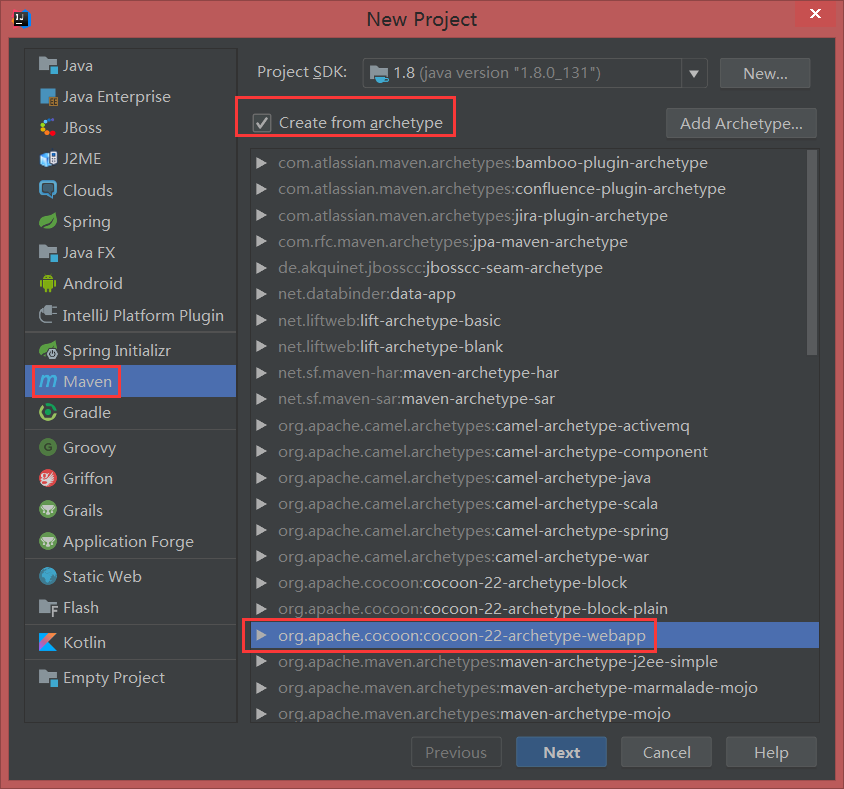
## Idea创建Maven项目

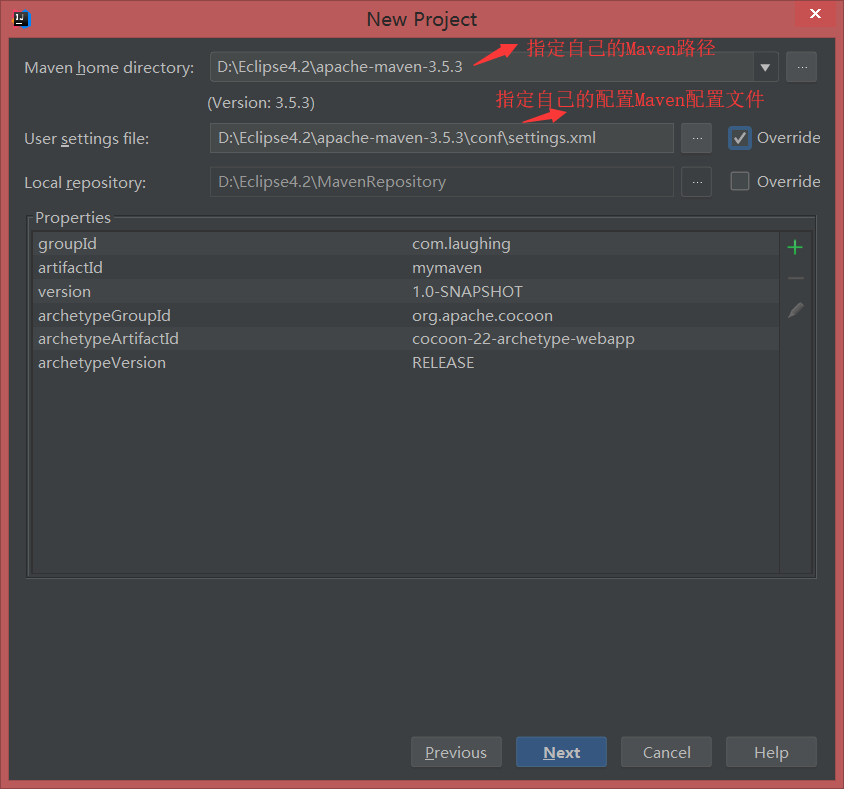






## idea创建Maven Web项目



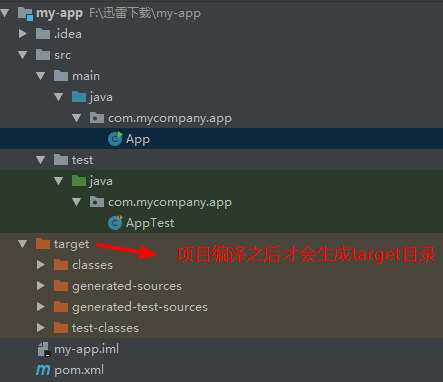


## 2.7 maven项目的目录结构

如果需要maven工具来构建管理项目就必须按照maven的规范约束创建目录结构。

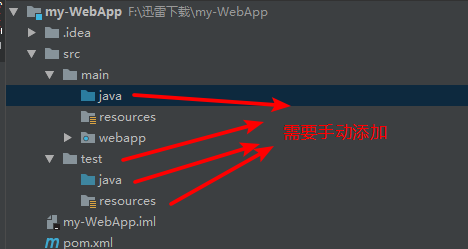
* 创建普通Java项目：

mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false



* 创建web项目：

mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-WebApp -DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false



## 2.8 maven常用命令

* mvn clean：删除清理编译后产生的target目录；
* mvn compile：编译src/main下面的源码并生成target目录，在target目录下的classes目录下生成对应的.class文件；
* mvn test-compile：编译src/test下面的源代码;
* mvn test：编译src/test下的源码，在target目录下生成test-classes目录并生成对应的.class文件；
* mvn package：按照pom文件中配置的打包方式打包当前项目；
* mvn install：把当前项目打包后发布到本地配置的maven仓库目录中；
* mvn site：在target目录下生成项目说明文档；
* mvn dependency:tree 查看当前项目的依赖信息；

## maven生命周期

compile<test<package<install（执行范围从小到大，后面的命令执行时会预先依次执行前面的命令）;

## 依赖版本冲突解决



## 依赖包的作用范围

即<dependency>/<scope>节点的取值

* 运行时无效表示在打包时相关依赖不会参与打包；

### compile(默认值)

在main和test下都有效，编译时和运行时都有效

### test

只在运行单元测试时有效，只对test目录有效，对main目录无效；编译时有效，运行时无效；

### provided

对main、test目录有效，运行时无效。用于生产环境已经提供了相关依赖包的前提下。

### system

### runtime

### import

## maven打包方式

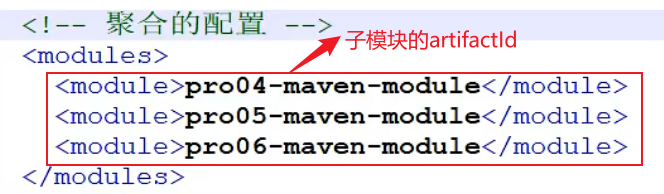
* jar
* war
* pom：作为管理其他工程的工程打包方式

## 依赖排除

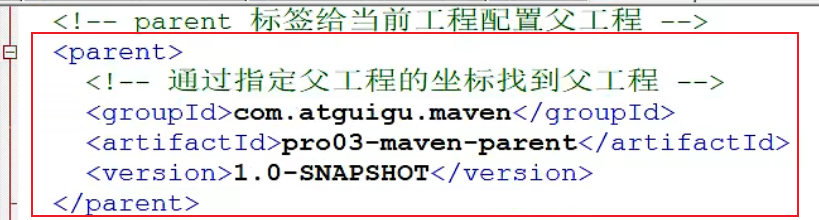


## 工程继承

* 只有打包方式为pom的工程才能管理其他工程；
* 父工程一般不写业务代码；
* 父工程pom包含子模块：



* 子模块pom指定父工程：



* 在父工程pom中统一管理依赖信息



* 子模块要使用哪些依赖包，还得在自己的pom中明确加上(版本不用再指定)；