SSM 重点

Spring Springmvc Mybtis

SSH 介绍

Struts2 Spring Hibernate

Maven

Git

Shiro

Vue

LayUI

Echarts等

# 一 maven简介

maven是基于pom（项目对象模型）的项目管理和构建工具

maven主要做了两件事：

项目构建：清理→编译→测试→生成测试报告→打包→部署

jar包的依赖管理：依赖指的是jar包之间的相互依赖，依赖管理指的就是使用Maven来管理项目中使用到的jar包，Maven管理的方式就是“自动下载项目所需要的jar包，统一管理jar包之间的依赖关系。

Maven资源库：

Maven本地资源库

Maven 的本地资源库是用来存储项目的依赖库，默认的文件夹是 “.m2” 目录，根据需要可以将其更改为另一个文件夹。

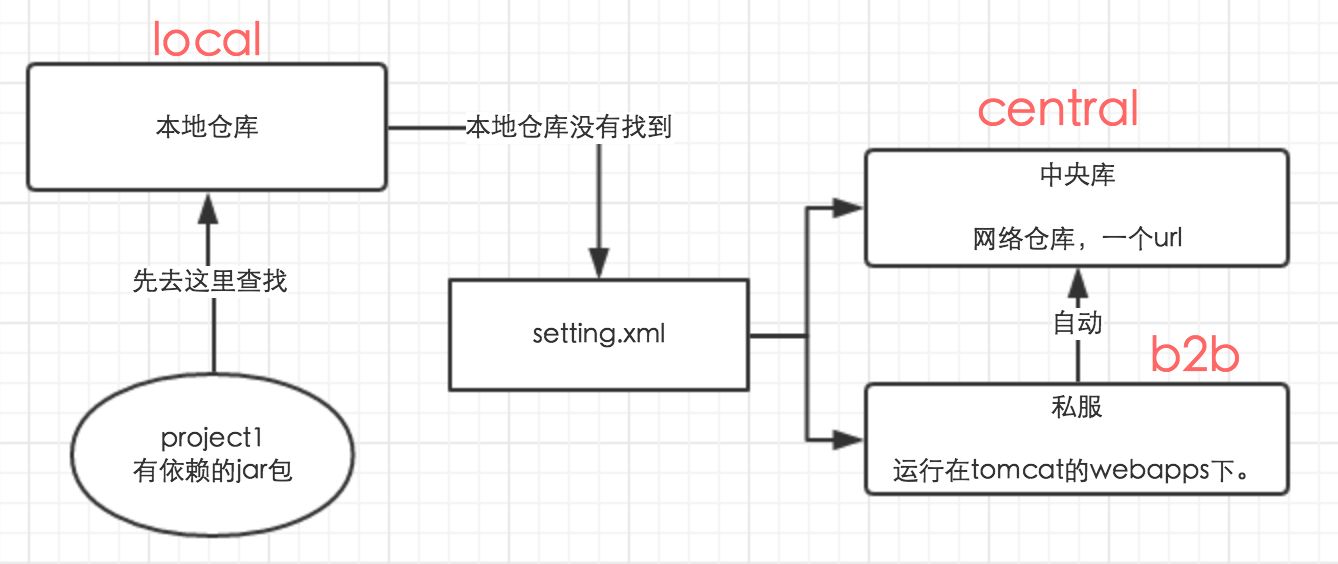
Maven中央仓库（中央存储库）

Maven 中央仓库是 Maven 用来下载所有项目的依赖库的默认位置。

Maven 私服

在局域网内创建一个类似于Maven中央仓库的私有仓库，这个私有仓库将从Maven中央仓库或其它远程仓库下载的构件缓存起来，当用户需要时，先从私有仓库查，没有再由私有仓库向远程仓库找，下载到私有仓库。一般使用Nexus来搭建 Maven私服。

Maven执行时，其中一个首要目标就是检查项目的依赖。依赖是你的项目用到的jar文件（java库）。如果在本地仓库中不存在该依赖，则Maven会从中央仓库下载并放到本地仓库。本地仓库只是你电脑硬盘上的一个目录。你可以根据需要制定本地仓库的位置。你也可以指定下载依赖的远程仓库的地址。



# 二 pom文件

pom.xml，这是Maven的核心配置文件，pom称为Project Object Model（项目对象模型），它用于描述整个Maven项目，所以也称为Maven描述文件。描述项目用到的资源，包括源代码目录、测试代码目录等的位置，以及项目依赖的外部jar包。

每一个项目都有一个POM文件。POM文件即pom.xml，应该放在项目的根目录下。一个项目如果分为多个子项目，一般来讲，父项目有一个POM文件，每一个子项目都有一个POM文件。在这种结构下，既可以一步构建整个项目，也可以各个子项目分开构建。

modelVersion：

　　指定了当前Maven模型的版本号，对于Maven2和Maven3来说，它只能是4.0.0

groupId：

　　该是公司名或是组织名。一般来说groupId是由三个部分组成，每个部分之间以"."分隔，第一部分是项目用途，比如用于商业的就是"com"，用于非营利性组织的就是"org"；第二部分是公司名，比如"baidu"；第三部分是项目名

artifactId：

　　可以认为这是Maven构建的项目名

Version：

　　版本号，SNAPSHOT意为快照，说明该项目还在开发中，是不稳定的版本。在Maven中很重要的一点是，groupId、artifactId、version三个元素生成了一个Maven项目的基本坐标

Scope：

依赖范围

compile（编译范围） compile是默认的范围，会被打包。

provided（已提供范围） provided依赖只有在当JDK或者一个容器已提供该依赖之后才使用。它们不是传递性的，也不会被打包。

runtime（运行时范围） runtime依赖在运行和测试系统的时候需要，但在编译的时候不需要。

test（测试范围）只有在测试编译和测试运行阶段可用。

system（系统范围）必须显式的提供一个对于本地系统中JAR文件的路径。注意该范围是不推荐使用。

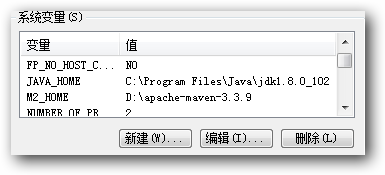
# 三 maven的安装配置

1 下载maven程序包，比如apache-maven-3.3.9，解压后，放在一个路径下

2 创建maven的环境变量

MAVEN\_HOME：maven的程序包路径

Path的最后添加;%MAVEN\_HOME%\bin





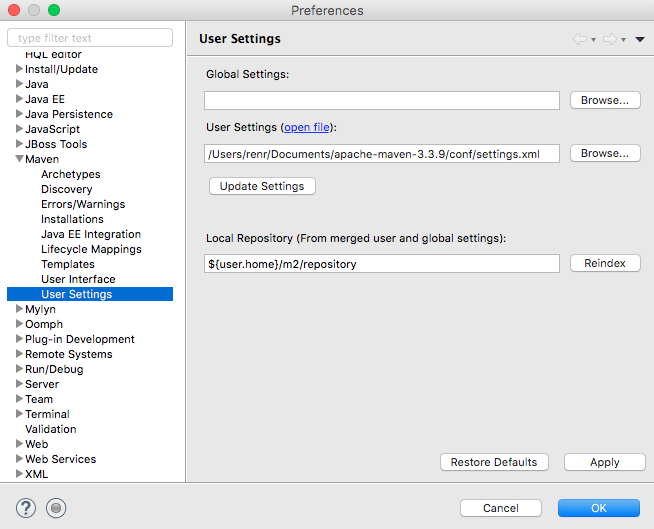


mvn –v 命令，查看版本号，通过该命令可以测试maven是否安装成功

3 eclipse中的配置

如果要修改Local Resposity的路径，需要修改settings.xml文件，然后点击“update settings”按钮





修改仓库的镜像路径：

<!-- 阿里镜像 -->

<mirror>

<id>alimaven</id>

<name>aliyun maven</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

</mirror>

# 四 创建maven工程

## 1 通过命令创建

1)创建Maven的普通Java项目：

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

2) 创建Maven的Web项目：

mvn archetype:generate -DgroupId=com.rr.webapp -DartifactId=mywebapp -DarchetypeArtifactId=maven-archetype-webapp -DinteractivMode=false

-DgroupId　　　　　　　　　组织标识（包名）

　　-DartifactId　　　　　　　　项目名称

　　-DarchetypeArtifactId　　 指定ArchetypeId，maven-archetype-quickstart，创建一个Java Project；maven-archetype-webapp，创建一个Web Project

　　-DinteractiveMode　　　　　　是否使用交互模式

3）其他常用命令

编译源代码

mvn compile

编译测试代码

mvn test-compile

清空

mvn clean

运行测试

mvn test

安装当前工程的输出文件到本地仓库

mvn install

打包

mvn package

先清除再打包

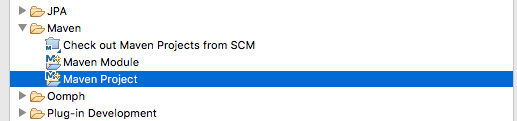
mvn clean package

注：运行jar包命令

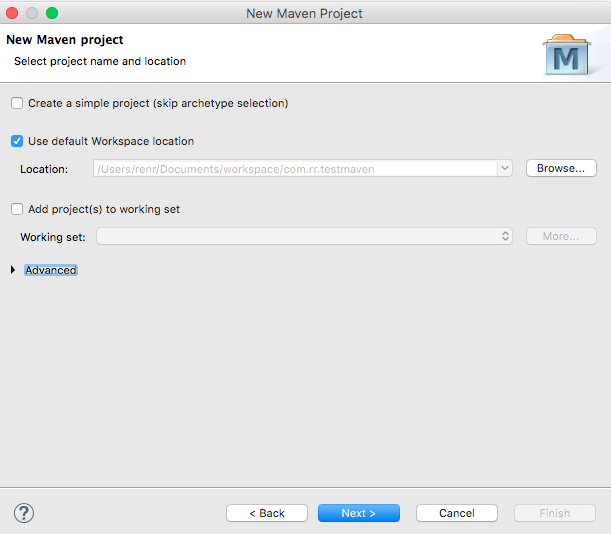
java -cp target\myapp-1.0-SNAPSHOT.jar com.rr.app.App

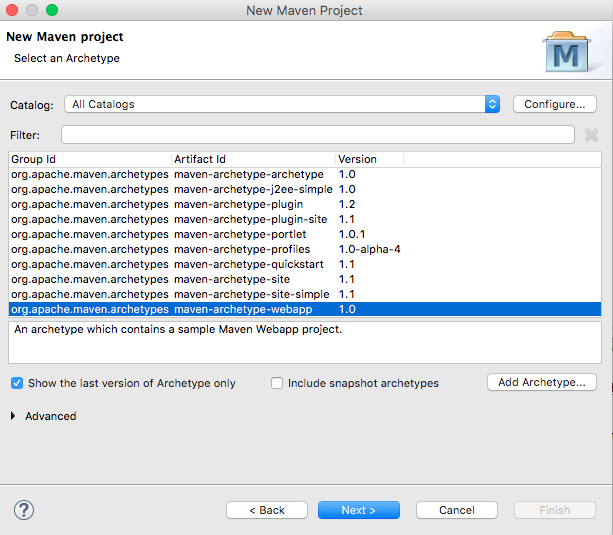
## 2 通过eclipse创建

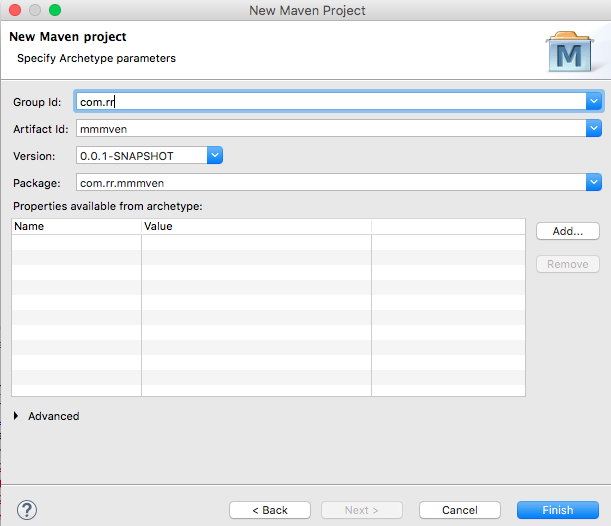
右键，Other-》Maven



使用archetype方式：







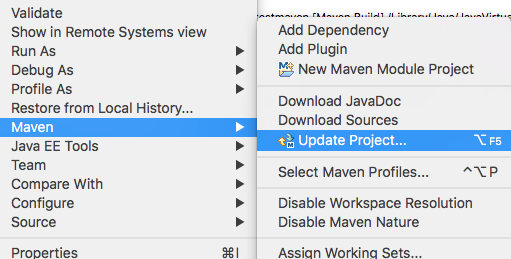
针对javaweb项目，pom中可以引入相关插件

tomcat7-maven-plugin可以让项目使用maven的tomcat插件部署项目，而不用将项目部署在外部的tomcat服务器上

该插件可以指定应用的路径

maven-compiler-plugin可以设置编译程序使用的java版本，因为创建的maven项目默认使用的jdk版本低，造成项目报错。

使用该插件后，update一下，显示的jre的版本就变了



<plugins>

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat7-maven-plugin</artifactId>

<version>2.2</version>

<configuration>

<uriEncoding>UTF-8</uriEncoding>

<path>/testmaven</path>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

<compilerVersion>1.8</compilerVersion>

<encoding>UTF-8</encoding>

</configuration>

</plugin>

</plugins>

创建工程出现的问题：

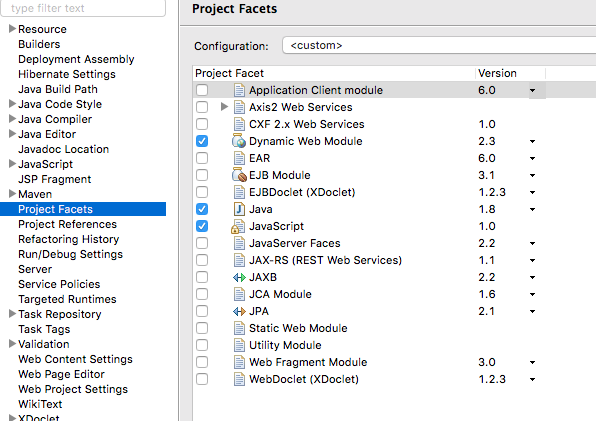
1)目录显示不全 没有resource、test等目录



可以在configure build path中将jre重新绑定下

2) servlet版本问题

web.xml中和Project Facets不一致



可以修改下面的文件



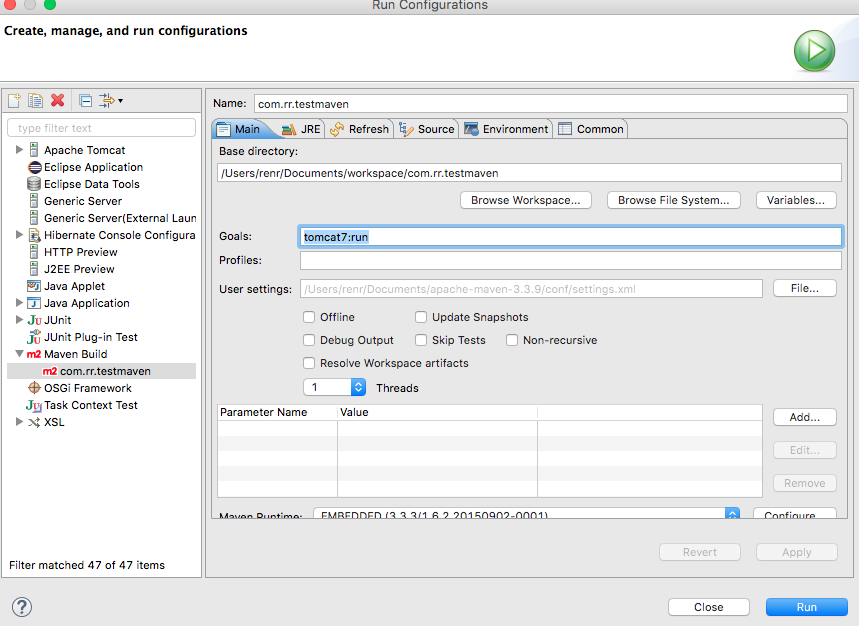
不使用archetye方式：

勾选“create a simple project”

# 五 运行项目

run as-》maven build

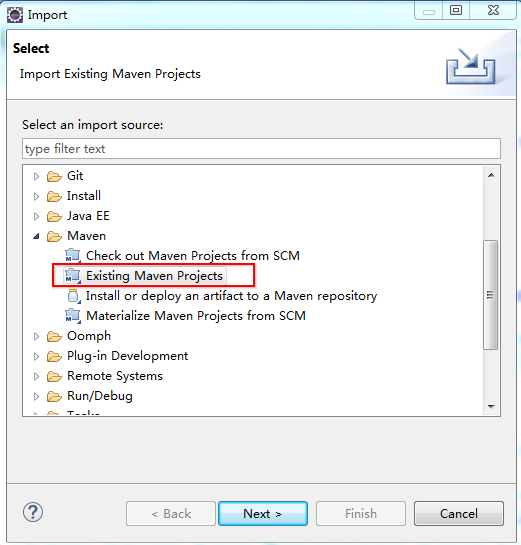
会启动tomcat插件



设置运行命令，点击run按钮执行

# 六 导入maven工程

选择导入，会出现导入存在的maven项目选项



# 七 发布项目到tomcat

1 修改tomcat的tomcat-user.xml配置

<tomcat-users>

<role rolename="admin-gui"/>

<role rolename="admin-script"/>

<role rolename="manager-gui"/>

<role rolename="manager-script"/>

<role rolename="manager-jmx"/>

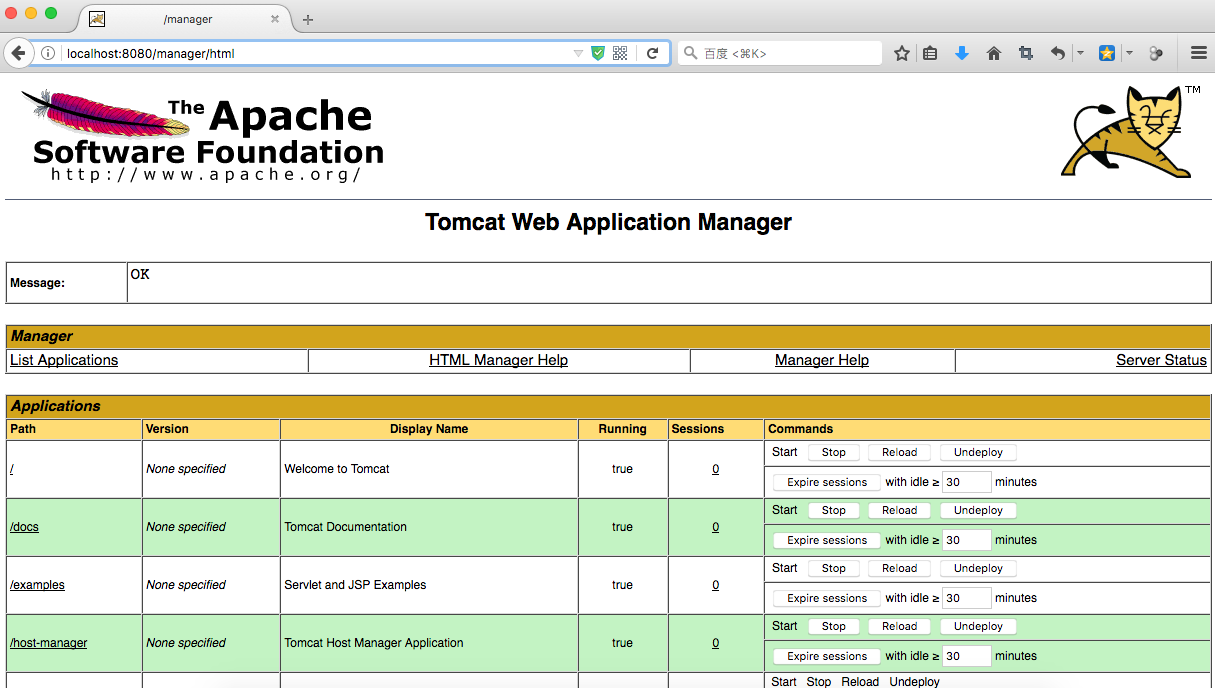
<role rolename="manager-status"/>

<user username="admin" password="admin" roles="manager-gui,manager-script,manager-jmx,manager-status,admin-script,admin-gui"/>

</tomcat-users>

保存后，启动tomcat服务器

访问<http://localhost:8080/manager/>



2 修改maven的settings.xml配置

id可以随便写，用户名和密码必须和tomcat-users.xml中配置一致

<servers>

<server>

<id>tomcat7</id>

<username>admin</username>

<password>admin</password>

</server>

</servers>

3 修改pom文件

<build>

<plugins>

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat7-maven-plugin</artifactId>

<version>2.1</version>

<configuration>

<!-- 注意此处的url 如果是远程服务器可以改成对应的地址，实现自动部署 -->

<url>http://localhost:8080/manager/text</url>

<server>tomcat7</server> <!-- 此处的名字必须和setting.xml中配置的ID一致 -->

<path>/hellosys</path> <!-- 此处的名字是项目发布的工程名 -->

</configuration>

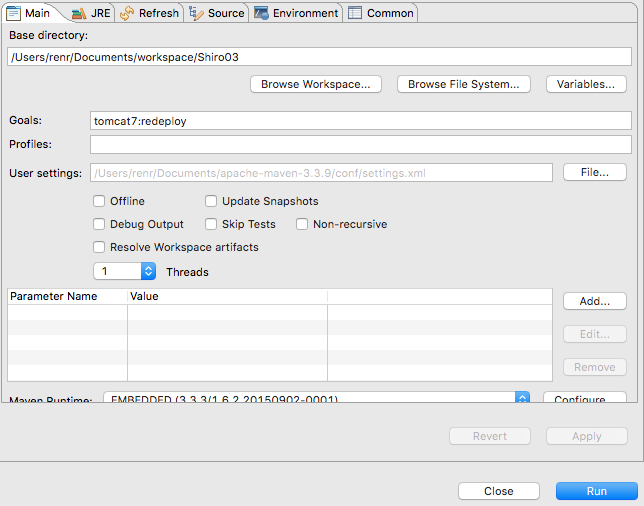
</plugin>

</plugins>

</build>

4 发布项目

命令：tomcat7:deploy（第一次使用，发布后改为redeploy）或者tomcat7:redeploy



额外：

对于中央仓库中没有的jar，可以先将该jar包下载下来，然后通过如下命令将其加入本地仓库

mvn install:install-file -Dfile=ValidateCode.jar -Dpackaging=jar -DgroupId=cn.dsna -DartifactId=ValidateCode -Dversion=1.0.0