# Nexus简介

Nexus用于搭建远程仓库。

## Nexus的下载与安装

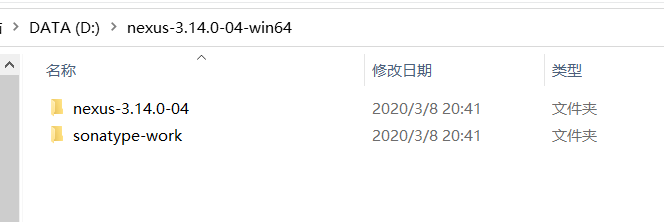
通过以下地址能够下载Nexus

<https://www.sonatype.com/appscan>

将下载的压缩包解压后，即表示安装。你可以将它放在合适的位置。

## Nexus目录结构

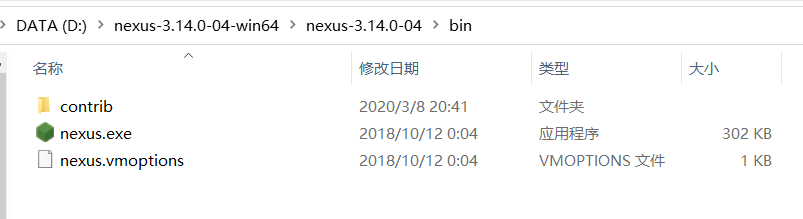
Nexus安装目录结构如下：



nexus目录为Nexus的安装目录；sonatype-work为仓库目录。进入到nexus目录中，如下：



我们只需关注bin目录和etc目录。进入bin目录中，如下：



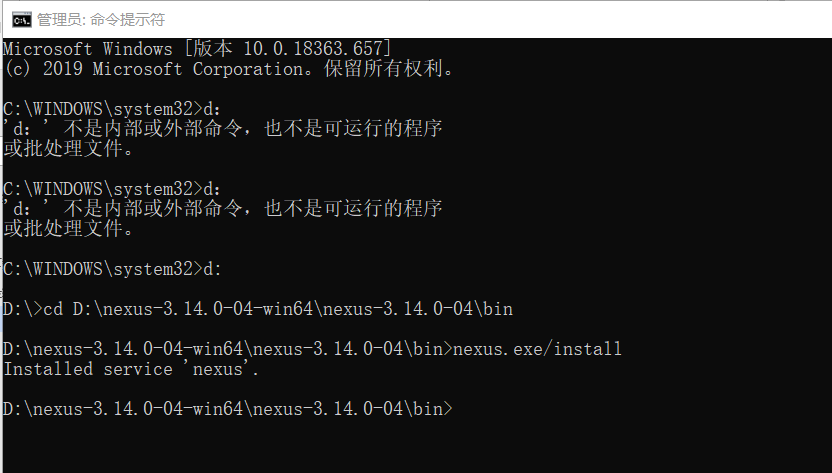
该目录中存放一些命令。nexus.vmoptions是一个配置文件，通过该文件我们可以修改一些虚拟机的配置，如内存等。一般默认即可。

etc目录如下：

该目录是存放配置文件的目录。如通过nexus-default.properties资源文件，我们可以配置Nexus的端口号，ip地址，上下文路径等。

## 注册与启动服务

使用管理员身份运行cmd，进入Nexus的bin目录，并执行**nexus.exe/install**命令注册服务。如下：

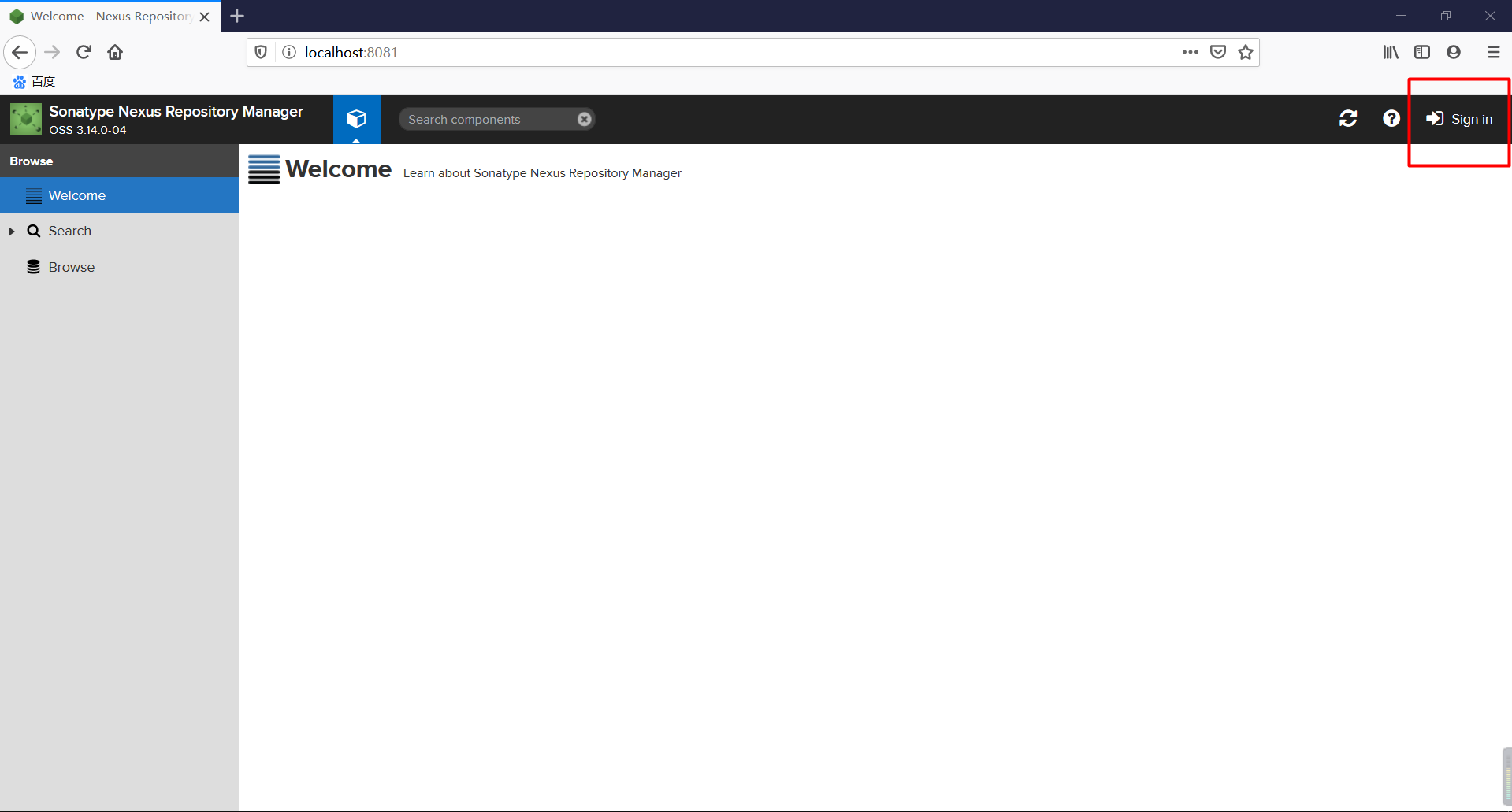


注册服务完成后，将其启动。如下：

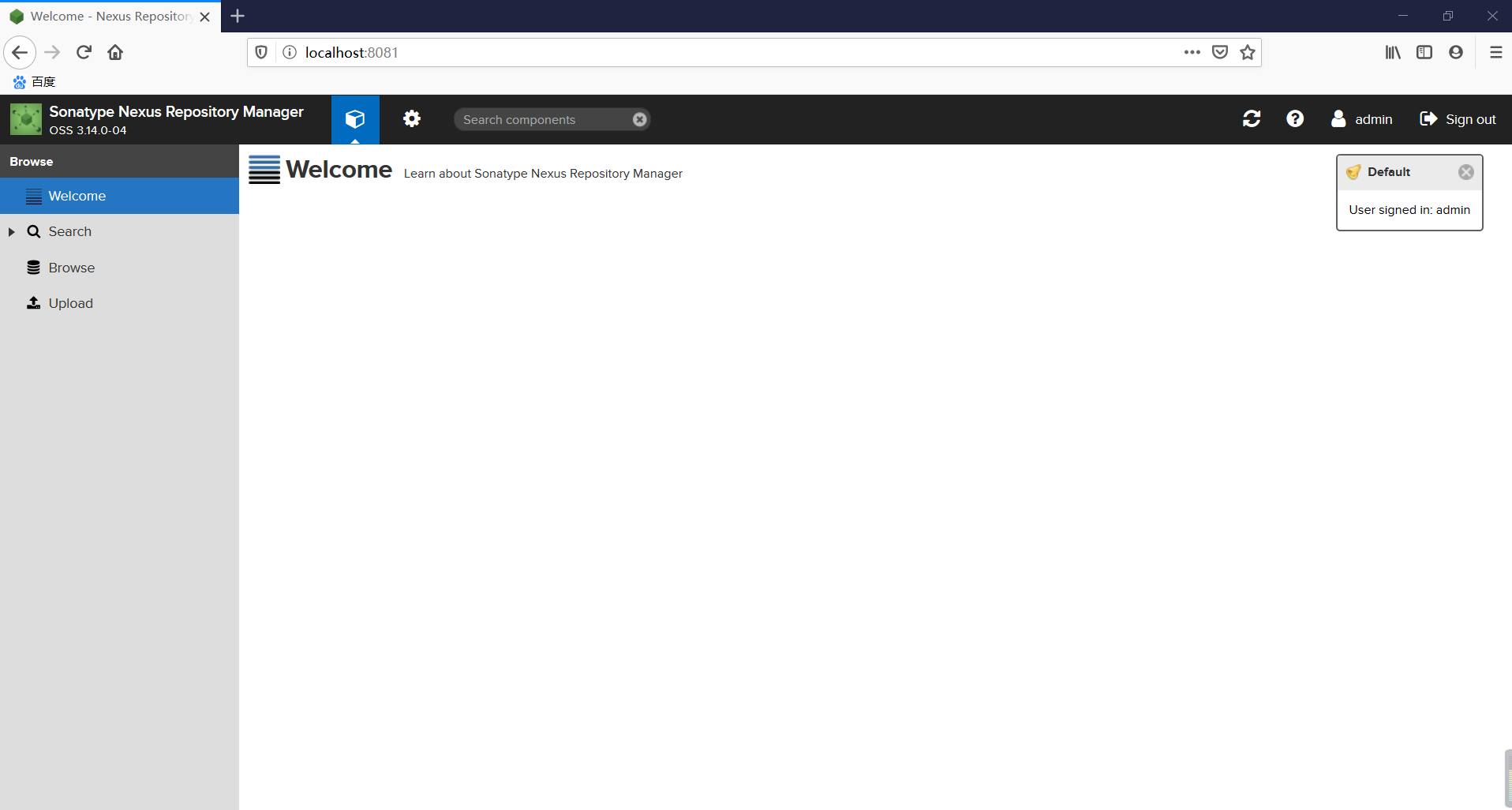


## 进入管理后台

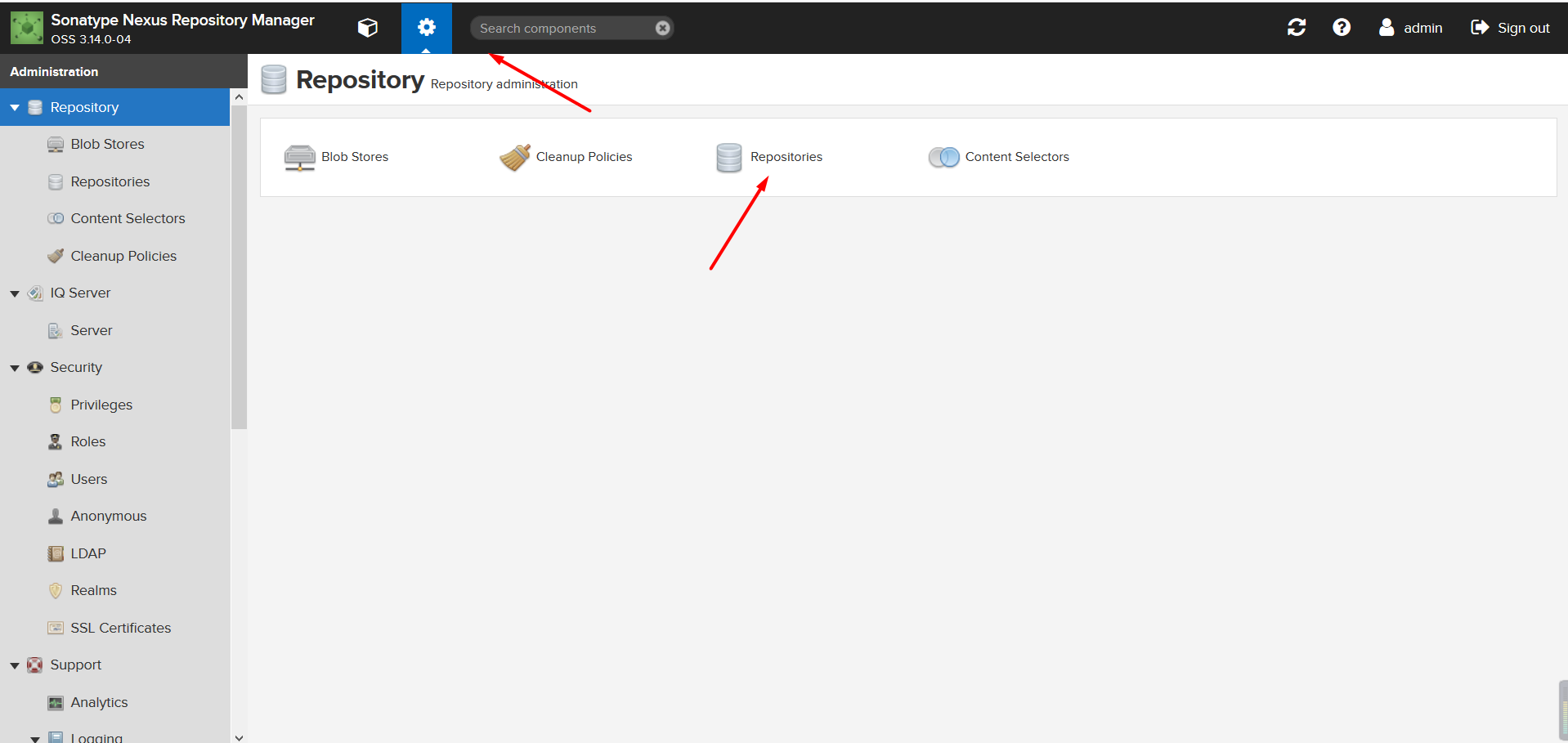
启动nexus后服务后，进入浏览器，输入localhost:8081（注意端口号与etc目录中nexus-default.properties配置的一致）。如下：

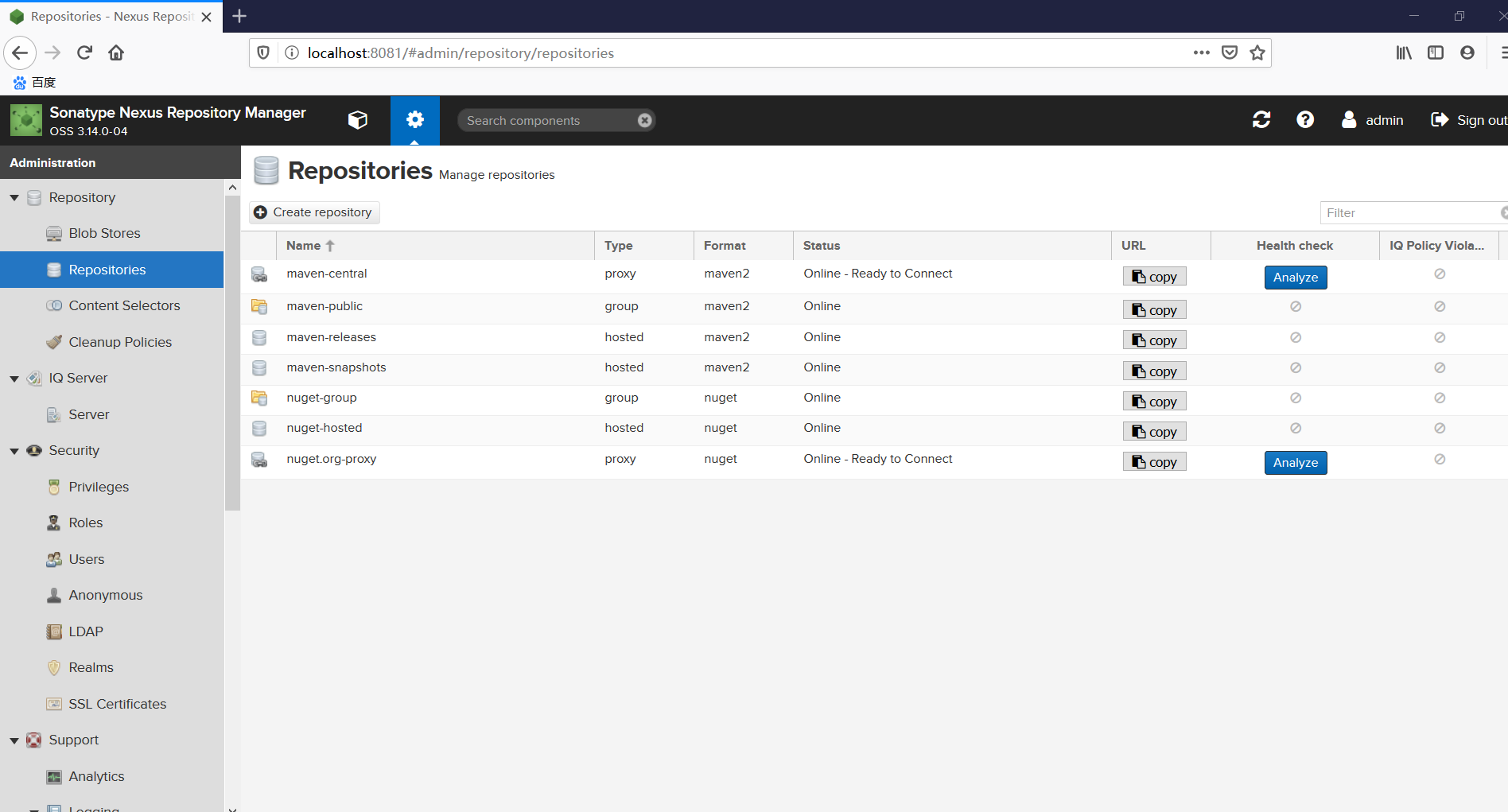


点击右上角Sign in进行登录，进入后台管理。登录名默认为admin，密码默认为admin123登录成功后。如下：



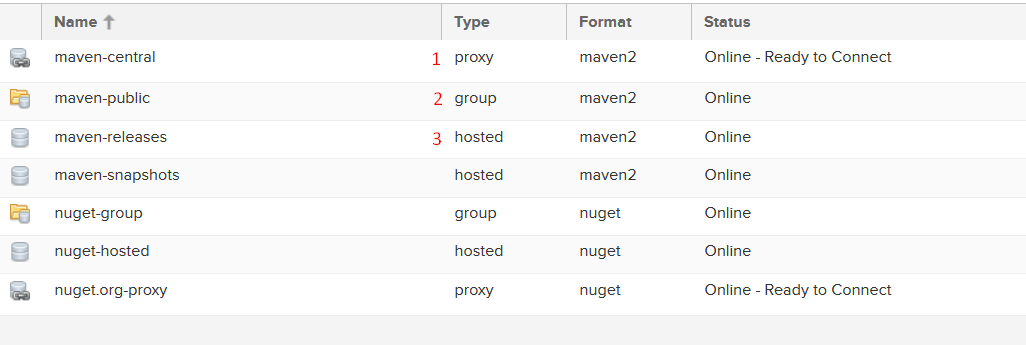
在上方找到设置，进入仓库。如下：



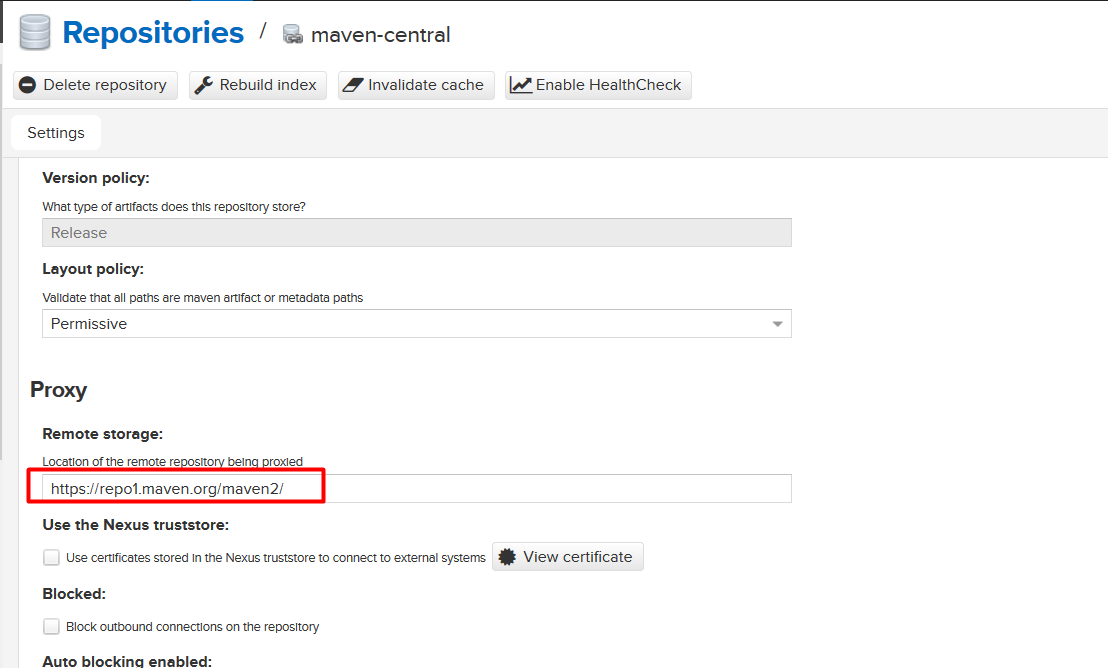


## 认识仓库类型

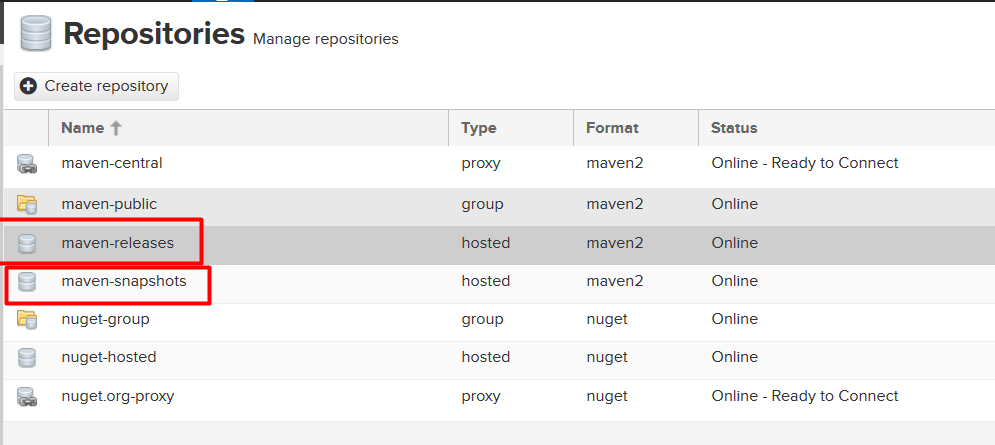
进入仓库中，可以看到分为3种，proxy（代理类型）、group（分组类型）、hoste（宿主类型）。如下：



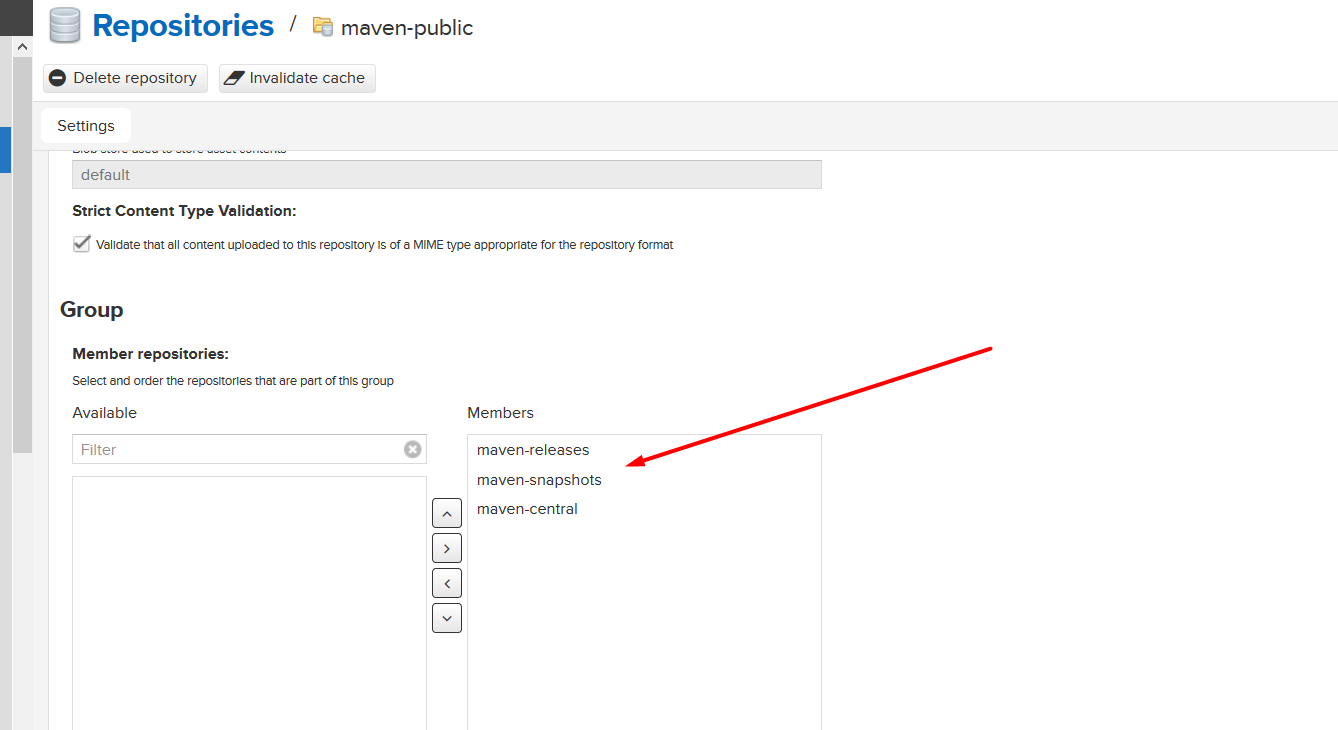
点击maven-central仓库，该仓库为代理类型的仓库，可以理解为中央仓库，代理的是Maven团队维护的中央仓库。如下：



maven-releases和mavensnapshots其实就是远程仓库。其中maven-releases存放的是公司或团队的正式版的jar包，mavensnapshots存放的是公司或团队的测试版的jar包。如下：



maven-public能够管理其他库，进入后如下：



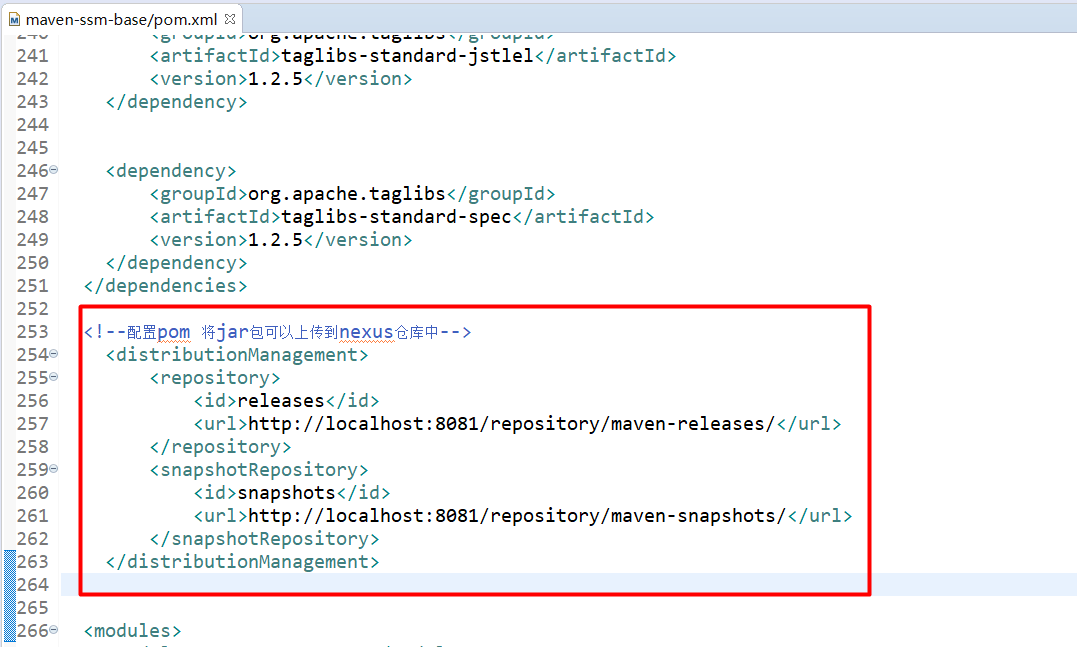
它可以将其他仓库加入到这里边统一管理。如果我们想获取或上传jar包，它可以自行判断从哪个仓库获取。

# 使用Nexus

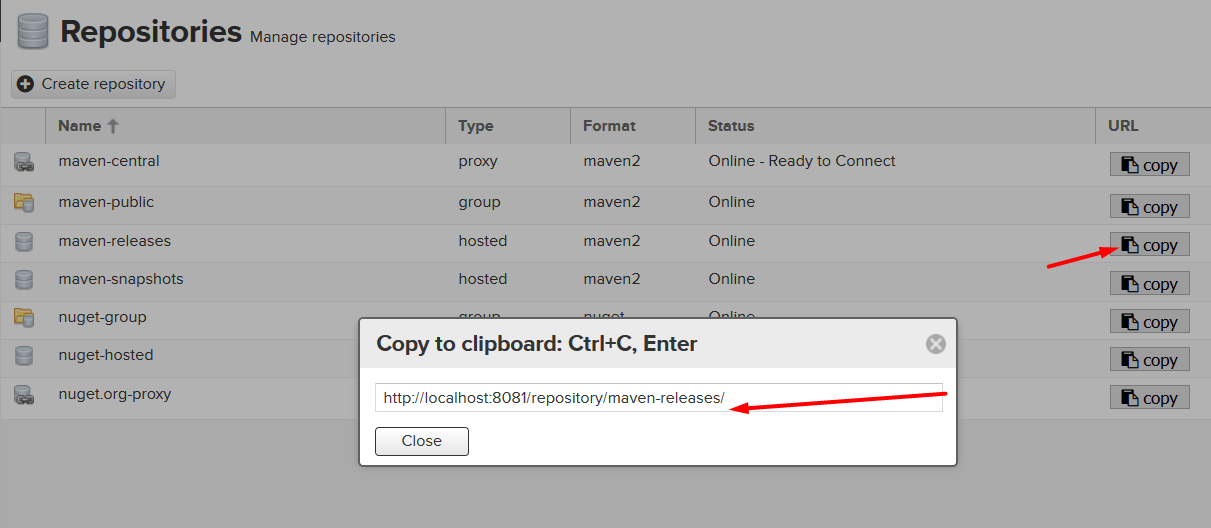
## 上传jar包到远程仓库

需求：项目模块分工后，其他模块需要依赖父项目，所以需要将父项目打成jar包安装到远程仓库中，service模块的开发依赖mapper模块，所以mapper模块的开发者需要将mapper模块打成jar包上传到远程仓库，供service模块依赖引用。

1. 在父项目pom.xml文件中配置nexus仓库地址。如下：



其中<id>标签中的内容可以随便定义，但一般将其定义为仓库名后面的部分；<url>标签定义远程仓库的地址。需要与仓库url相对应。如下：



1. 打开maven安装目录conf/settings.xml文件，配置仓库权限。如下：

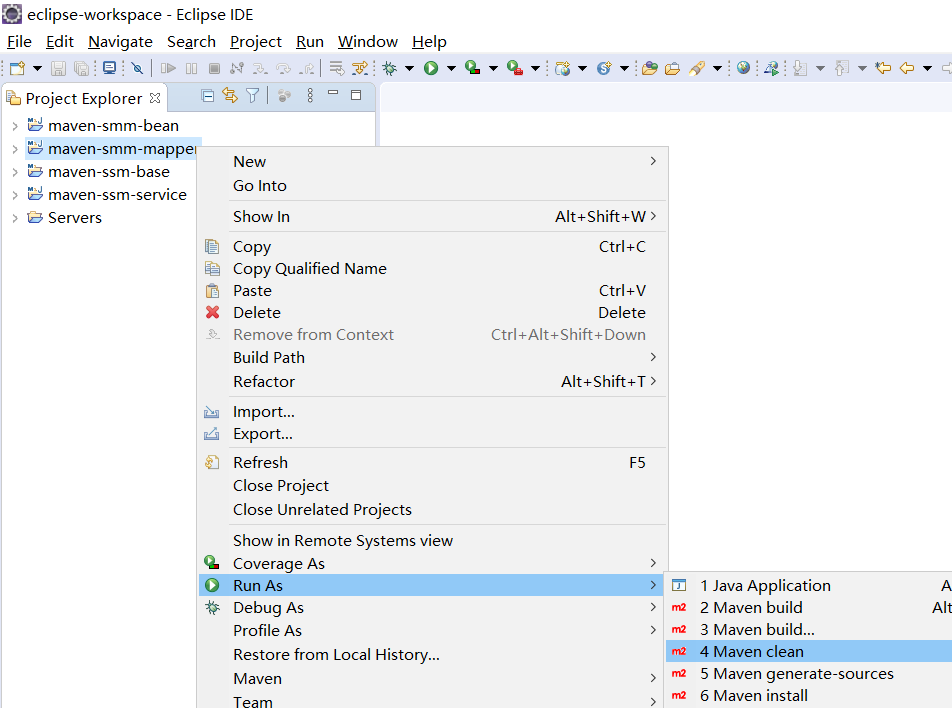


<id>标签中的内容与父项目pom.xml中配置的远程仓库的<id>两者要一致。（pom.xml看齐setting.xml），<username>定义用户名、<password>定义密码。

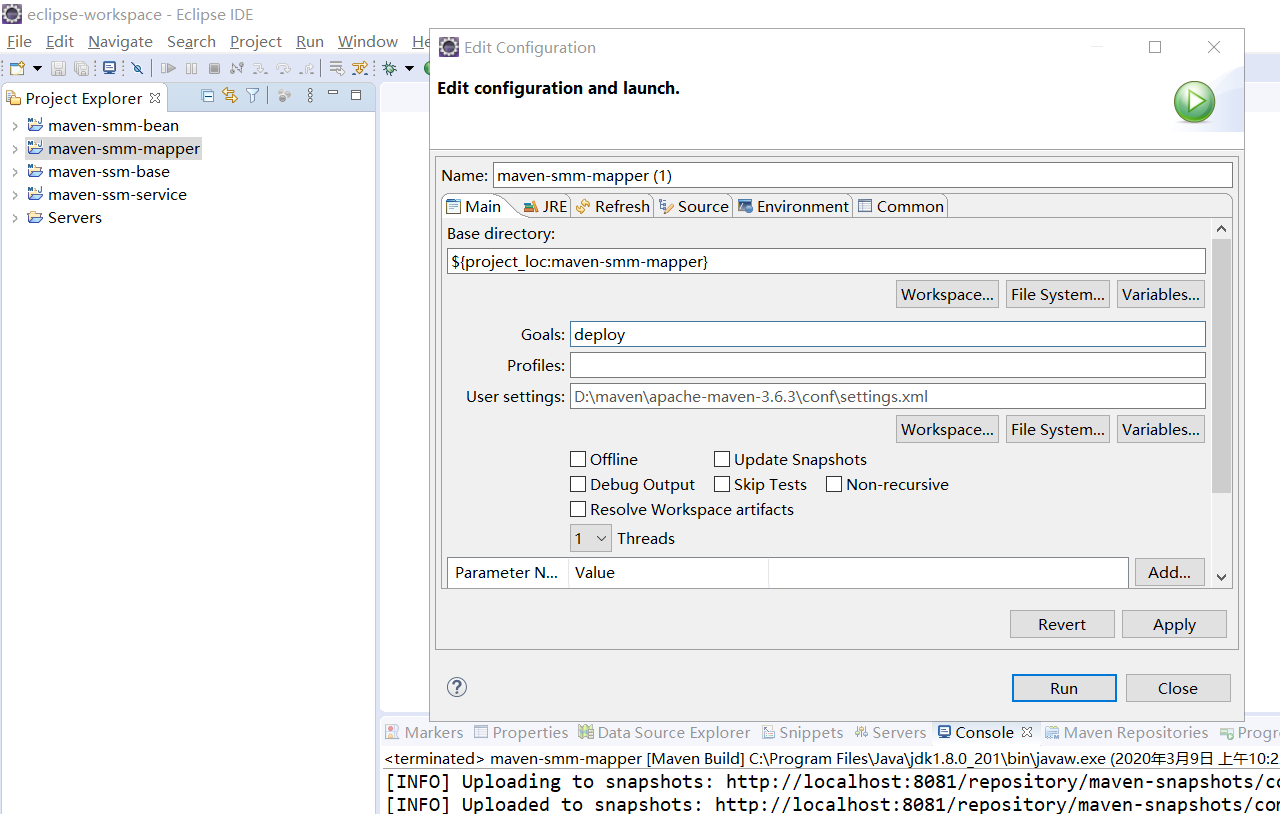
表示admin用户对这两个仓库具有权限。

1. 远程仓库和仓库权限配置好后，mapper模块开发者就可以将mapper模块上传到仓库中了。如下：

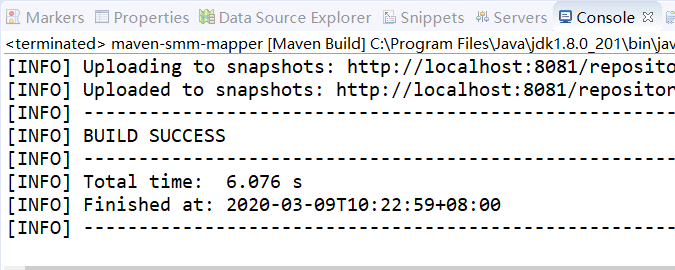
先清理一下：



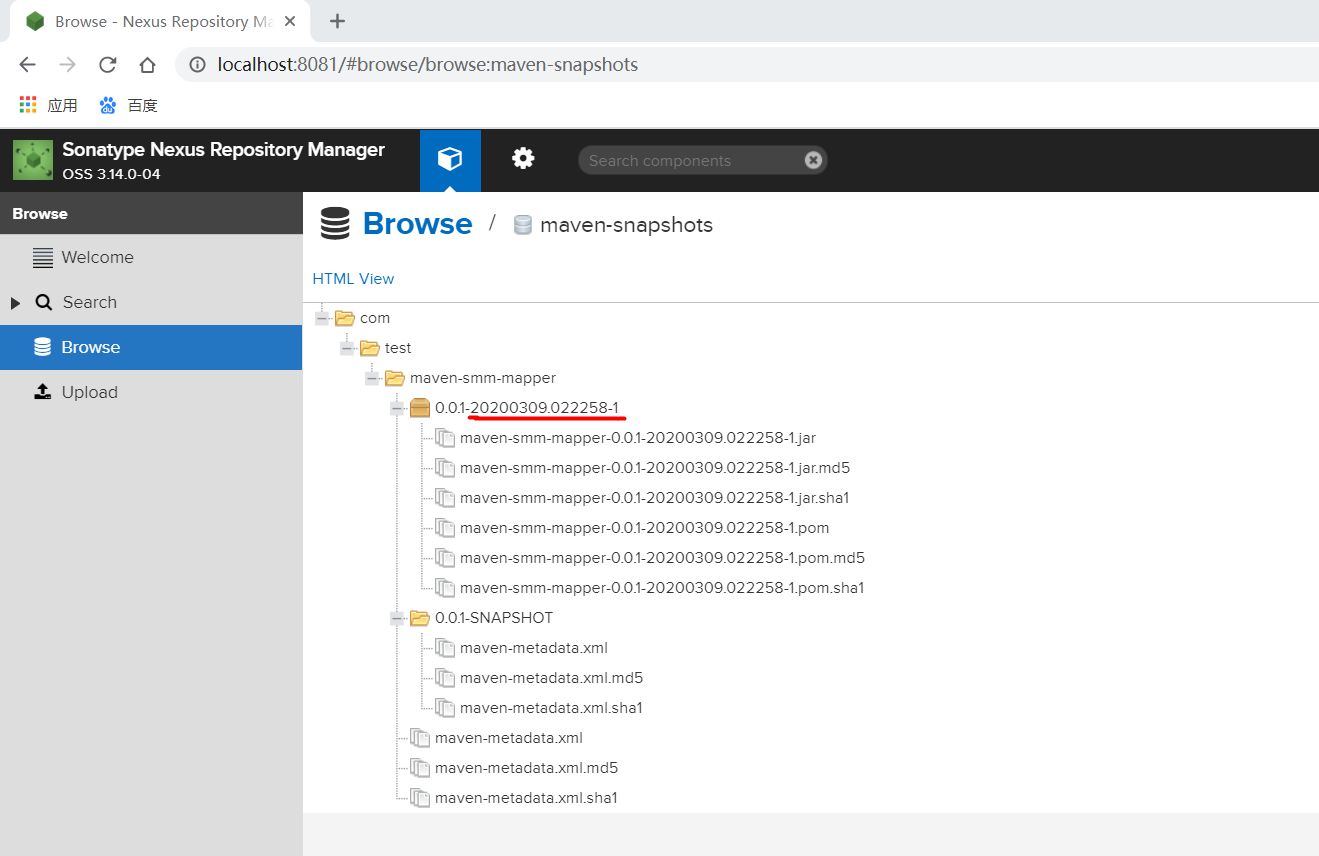
上传：



上传成功：



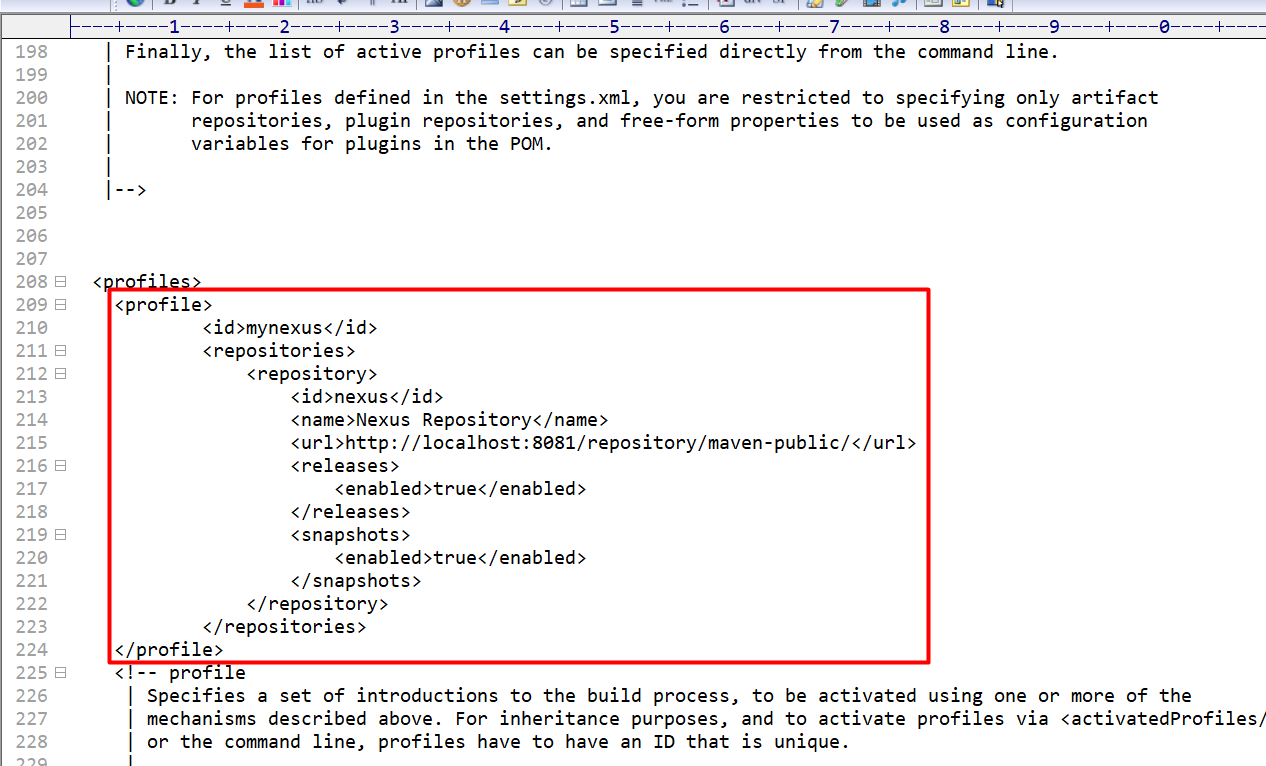
1. 打开Nexus管理后台，登录后在主界面打开【Blowse】-->【maven-snapshots】仓库，即可看到上传的jar包。



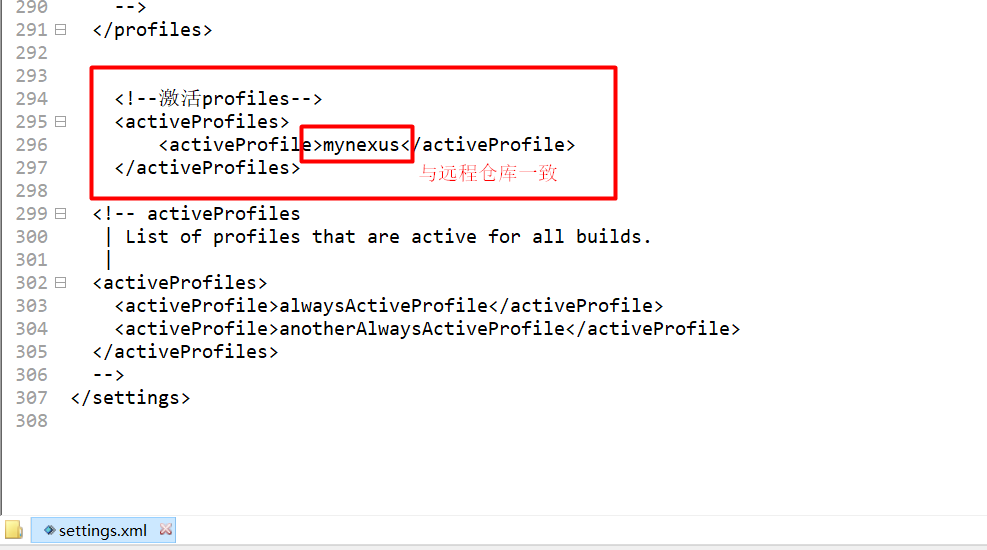
其中提交的版本后面会加上时间戳表示何时提交的，以及-1，表示第一个版本，如果再mapper模块开发者再提交一次，则为-2。

## 从远程仓库下载jar包

1. 打开maven安装目录下config/settings.xml文件。配置远程仓库。如下：



1. 对配置的远程仓库进行激活。如下：

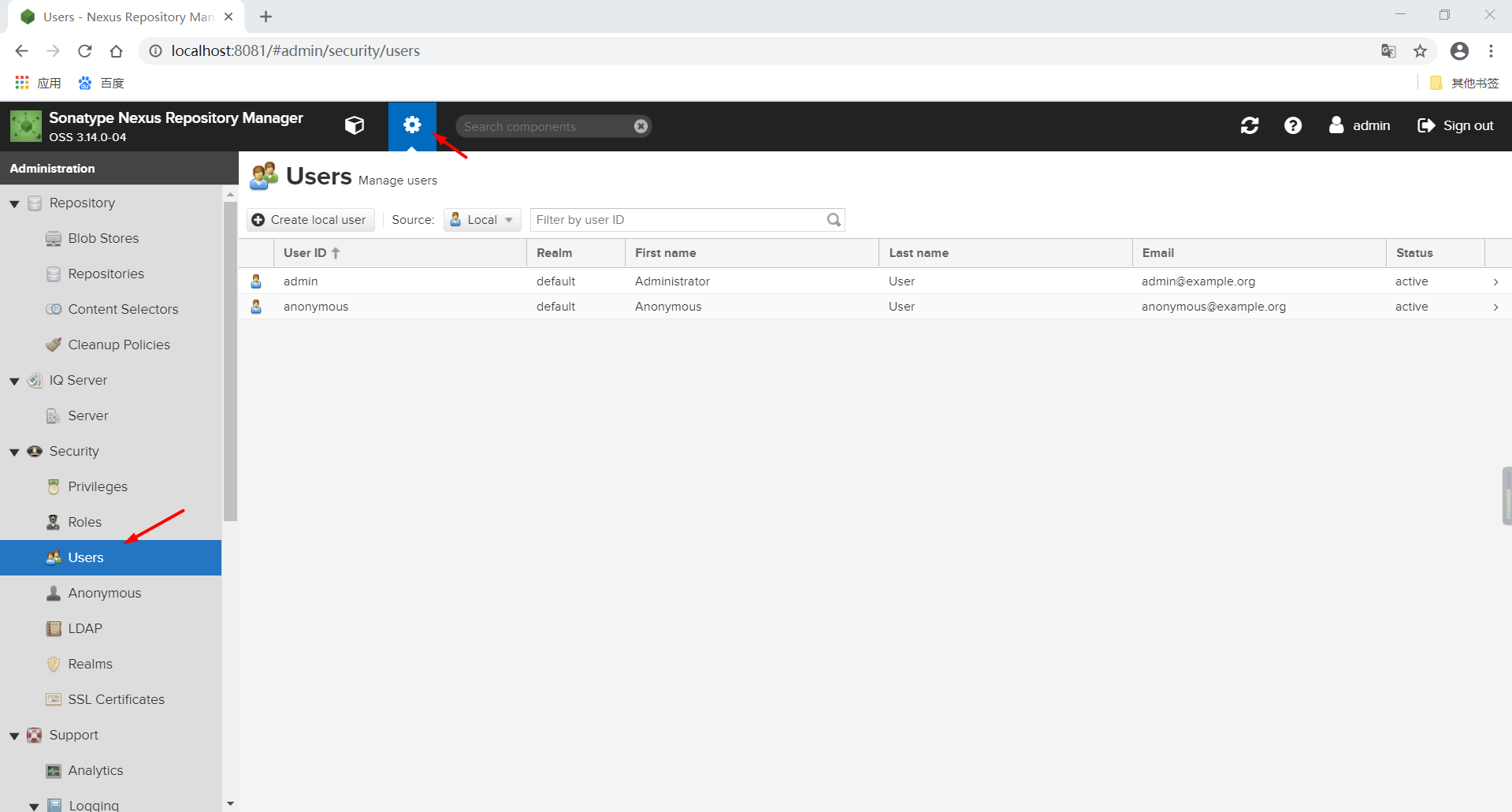


配置好远程仓库并激活后，service模块就可以在pom.xml文件中声明依赖引入mapper模块打成的jar包。（注意：在前面学习为了多模块分工开发时，都是在一个电脑上模拟，并且将mapper模块打成jar包放入本地仓库了，所以需要先把它删除）

# Nexus日常管理

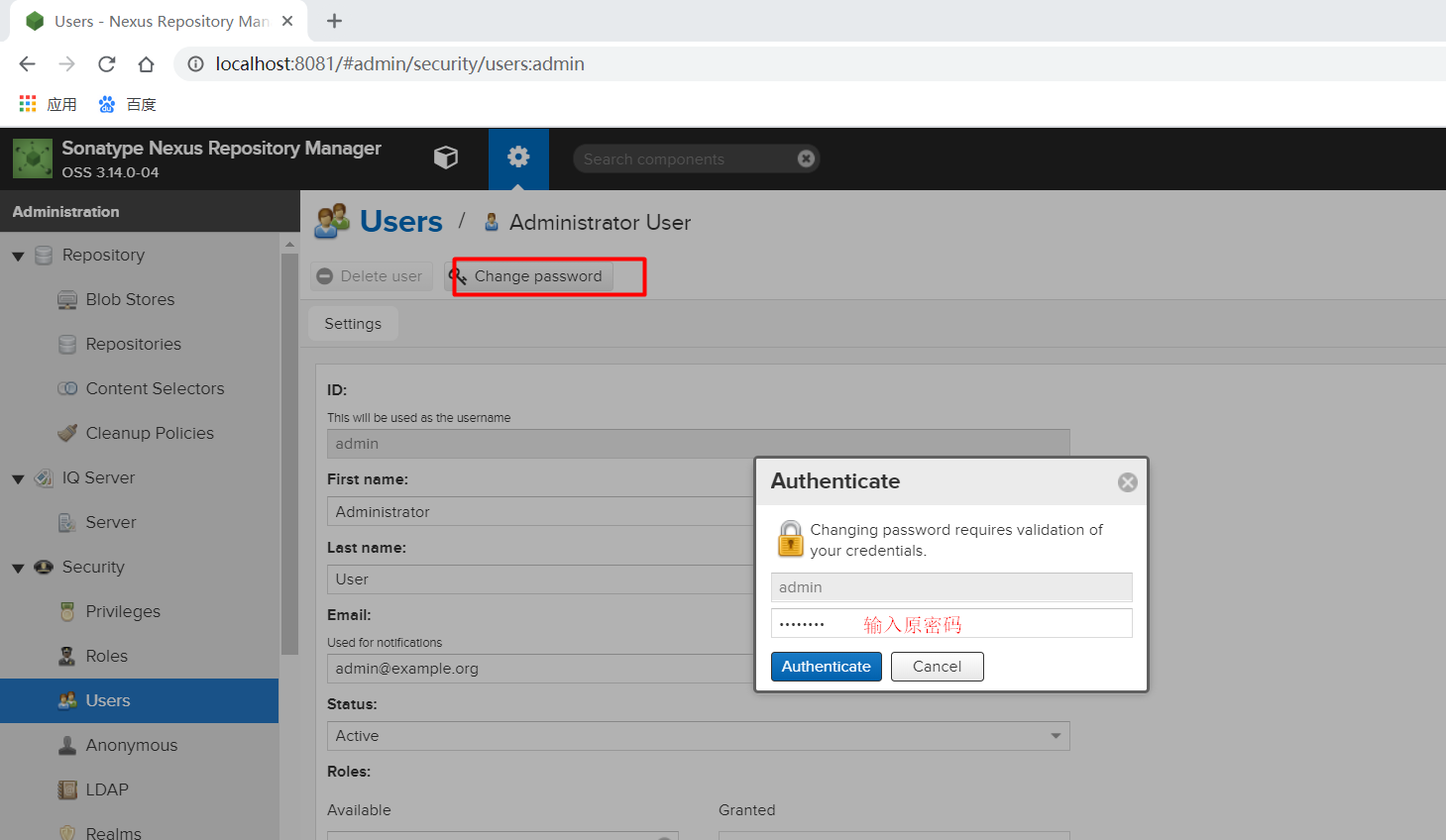
## 修改管理员密码

Nexus默认的超级管理员用户为admin，密码为admin123。为了安全起见，我们需要对密码进行修改。

1. 进入Nexus管理后台，打开【设置】下的【Users】。如下：

在右边可以看到有两个默认用户。admin用户为超级管理员，anonymous表示匿名用户。

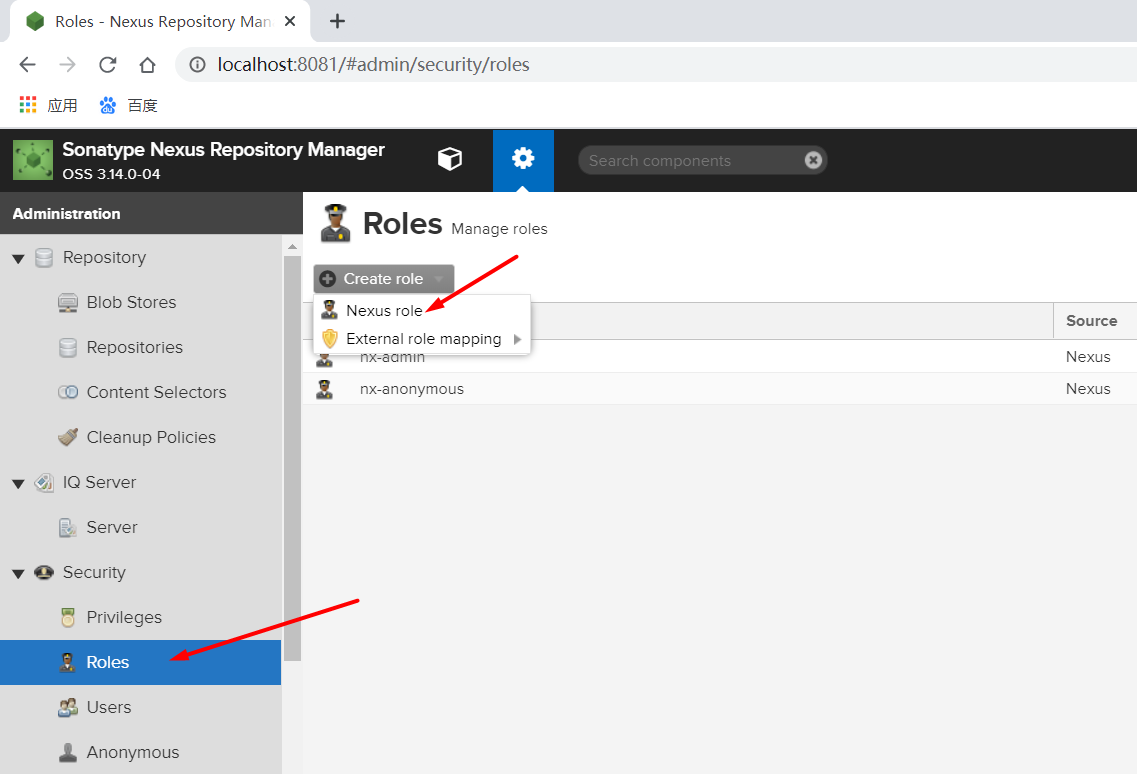
1. 点击第一个admin用户，然后在进入的页面中点击上方的【Change password】。先输入原密码，再输入新密码。如下：



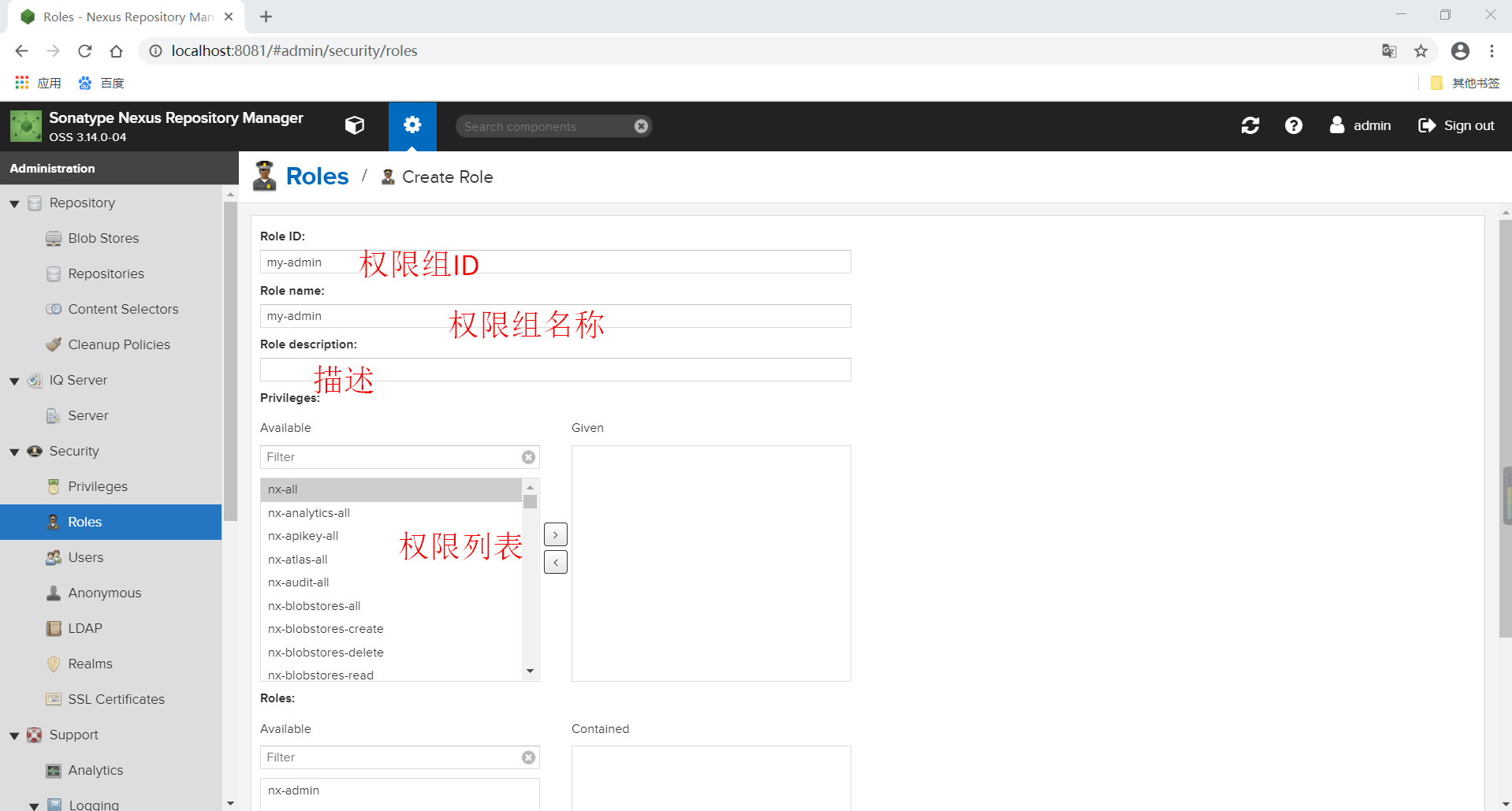
密码修改完成后，点击右上方的【Sing Out】登出一下，然后再使用新密码登录。

## 创建权限组

1. 进入【Roles】页面，点击上方【Create roles】下的第一个选项。如下：

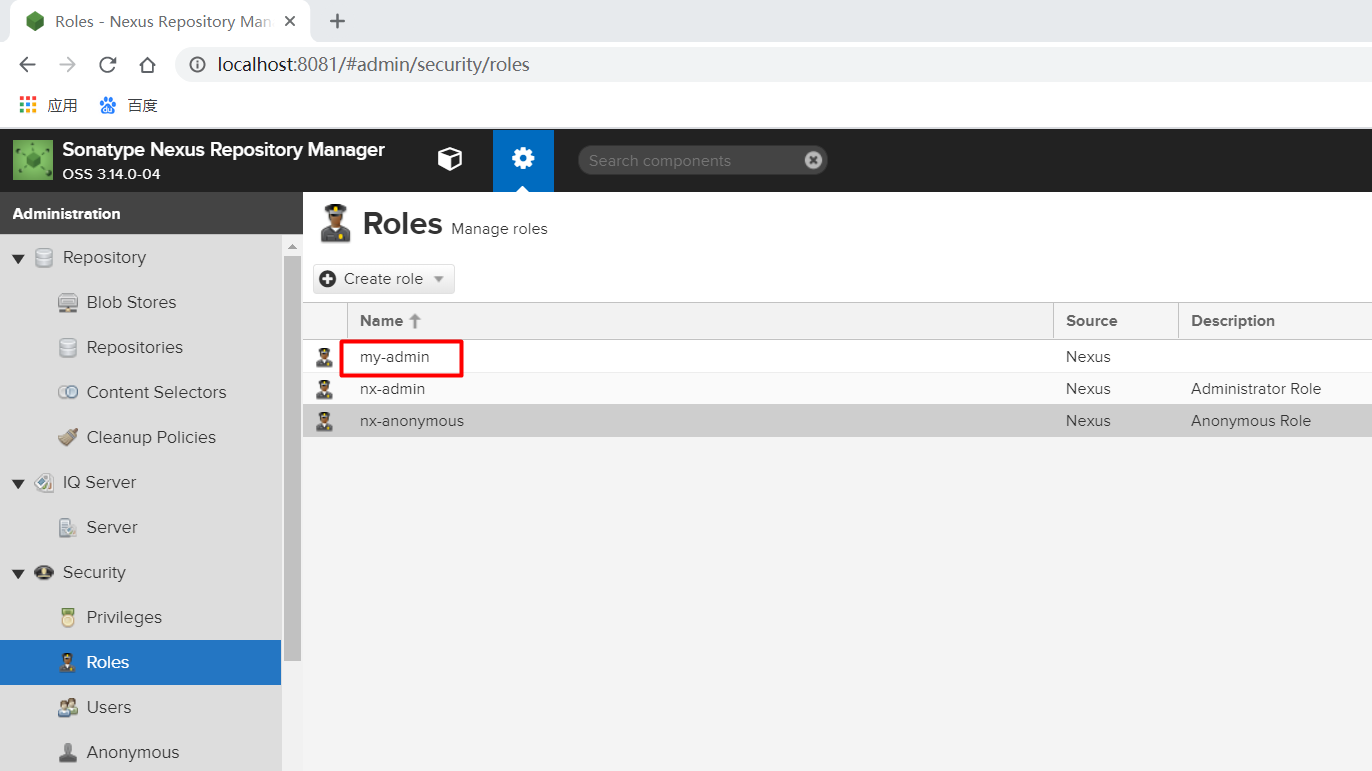


1. 进入权限组创建页面；填写权限组ID、名称，为权限组添加权限等。如下：

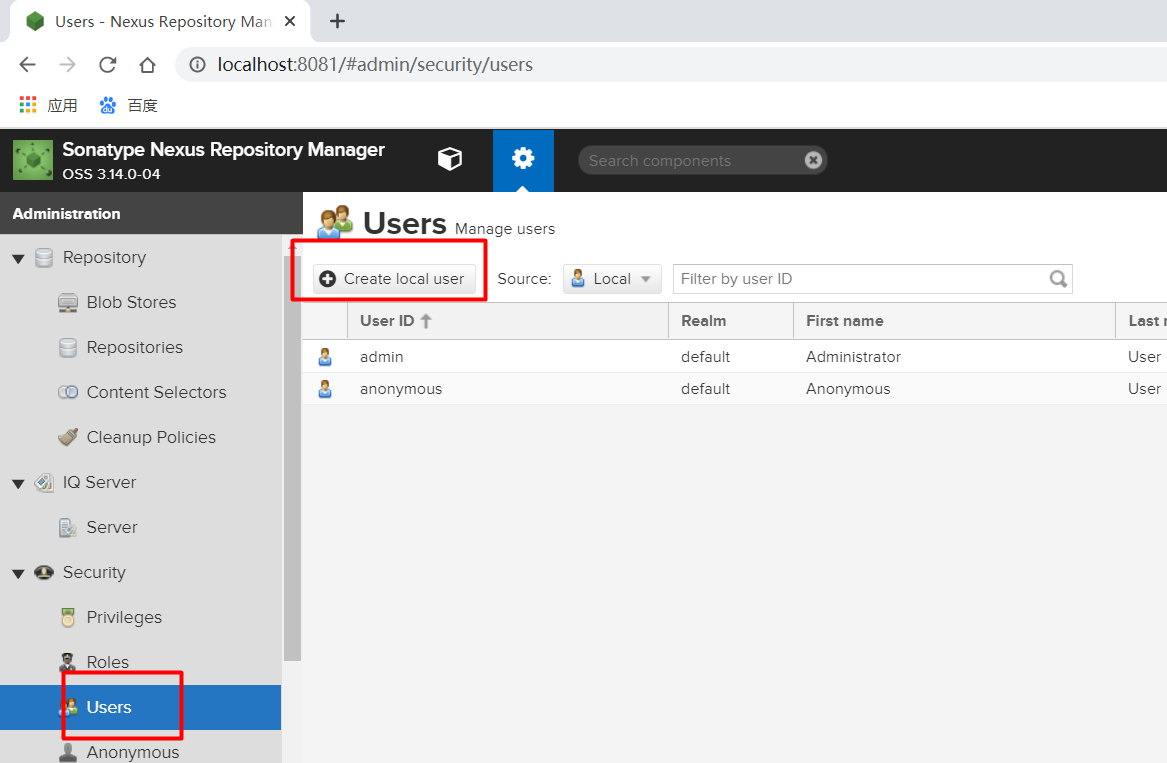


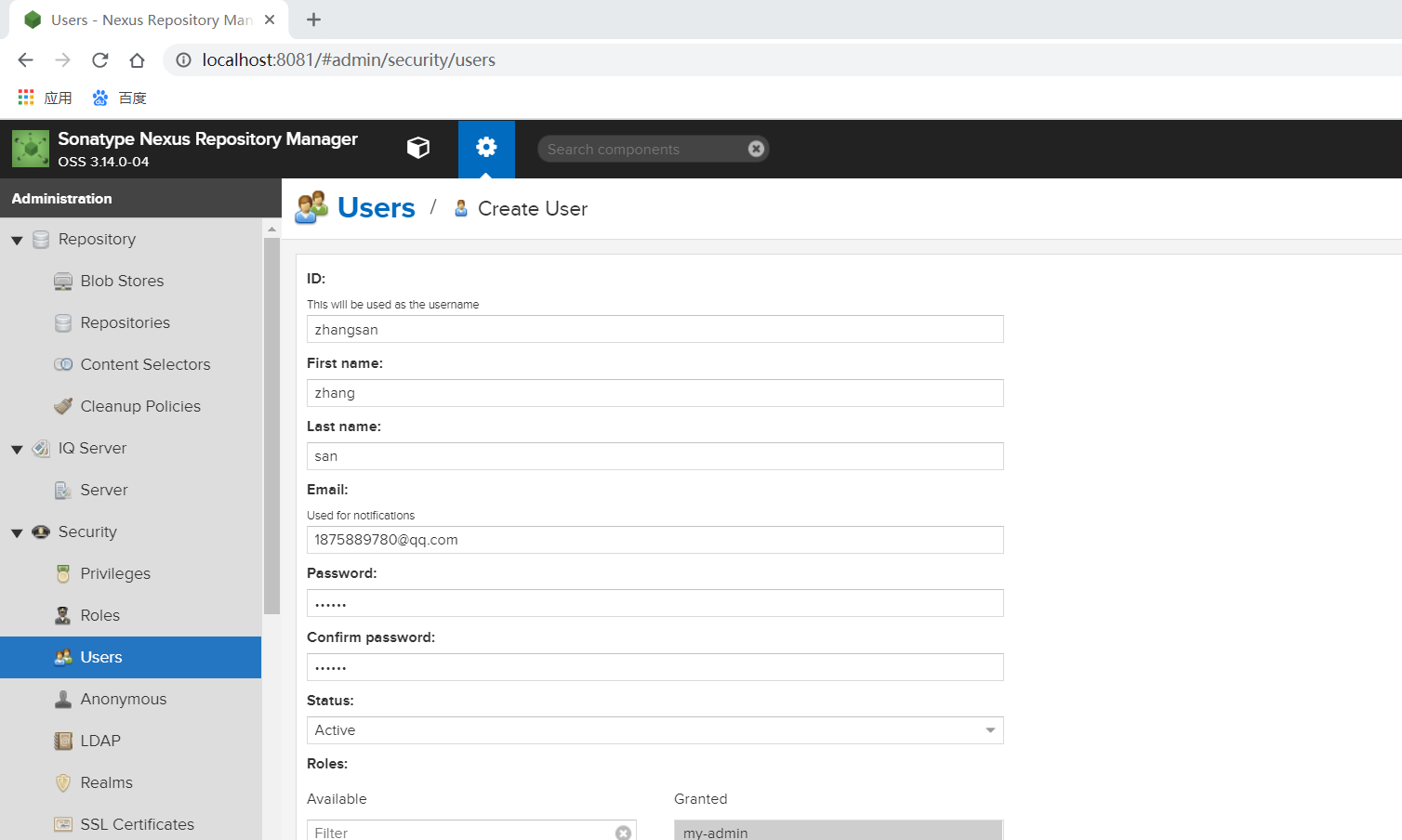
在权限列表中，nx-all表示所有权限。

1. 权限组配置好后，点击下方的【Create Role】选项完成对权限组的创建。如下：



## 创建本地用户

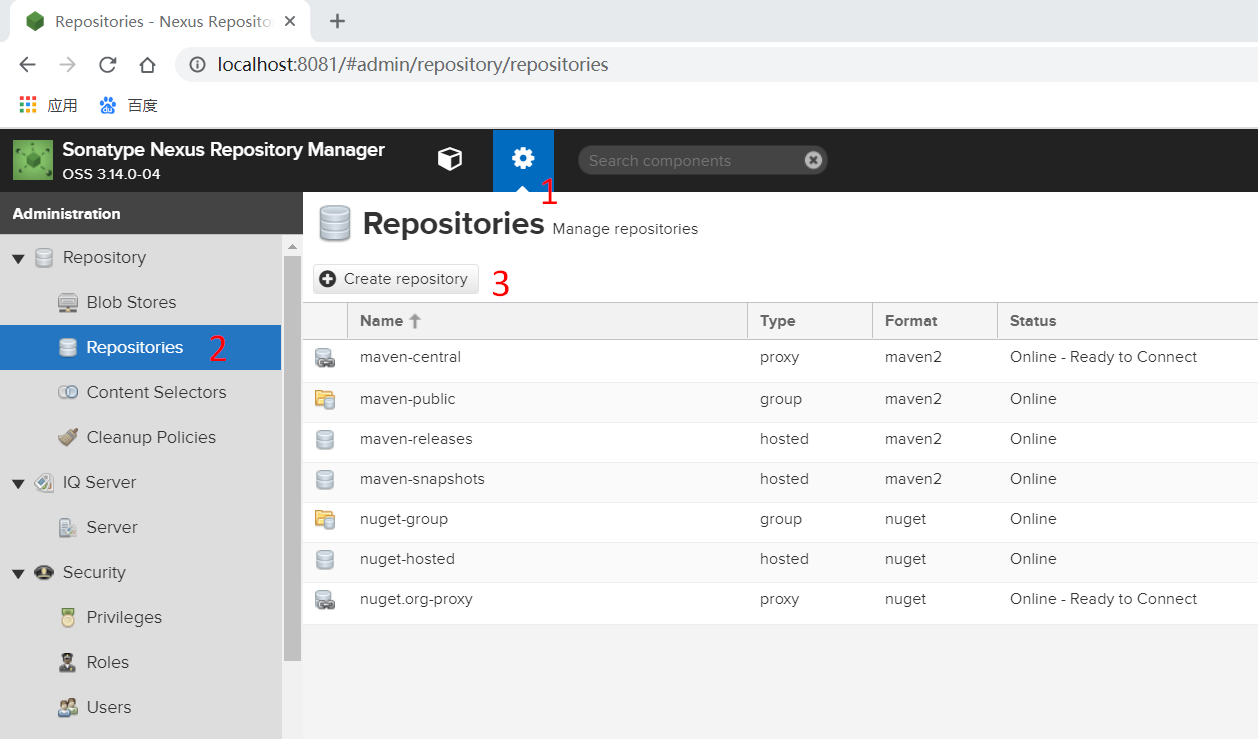
1. 进入【Users】页面，点击上方的【Create local User】。如下：
2. 填写相关信息，用户ID（用户名）、用户姓与名、用户邮件、密码、确认密码、状态等，还有所属权限组。如下：



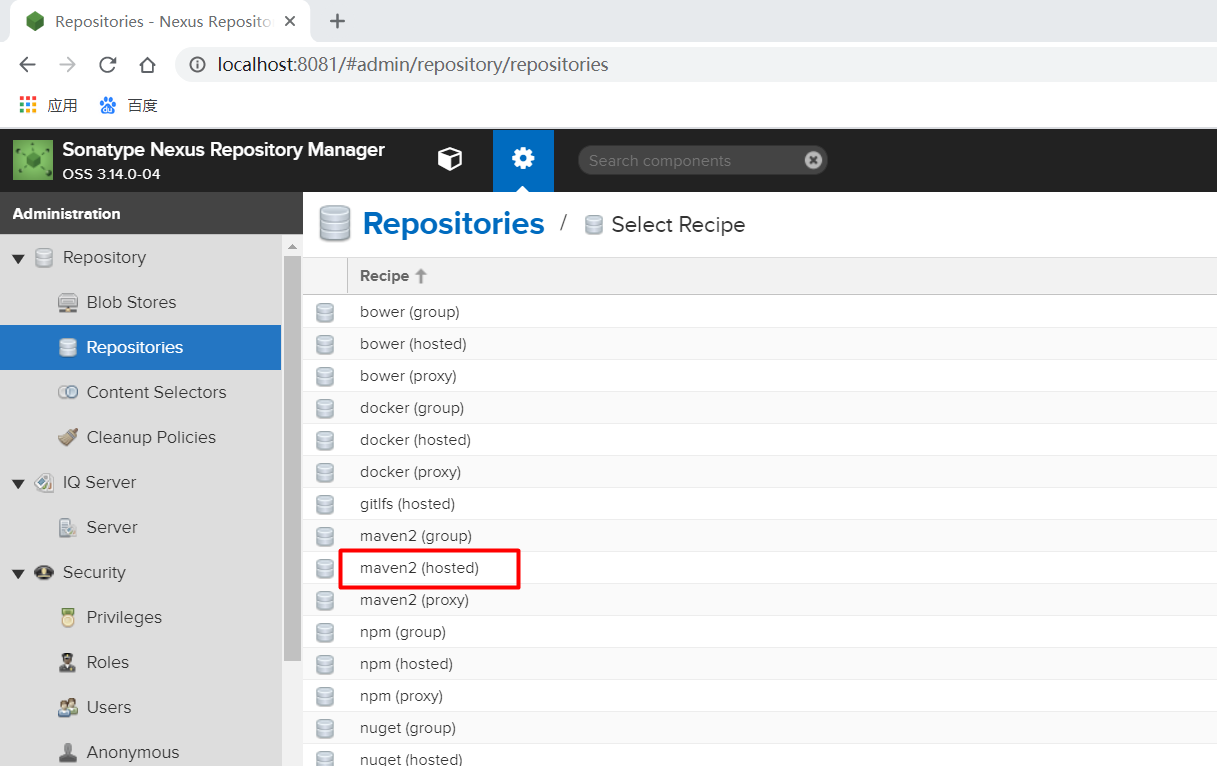
填写完成后，点击下方的【Create Local User】创建用户，即可在用户列表看到创建的新用户。

## 创建自定义仓库

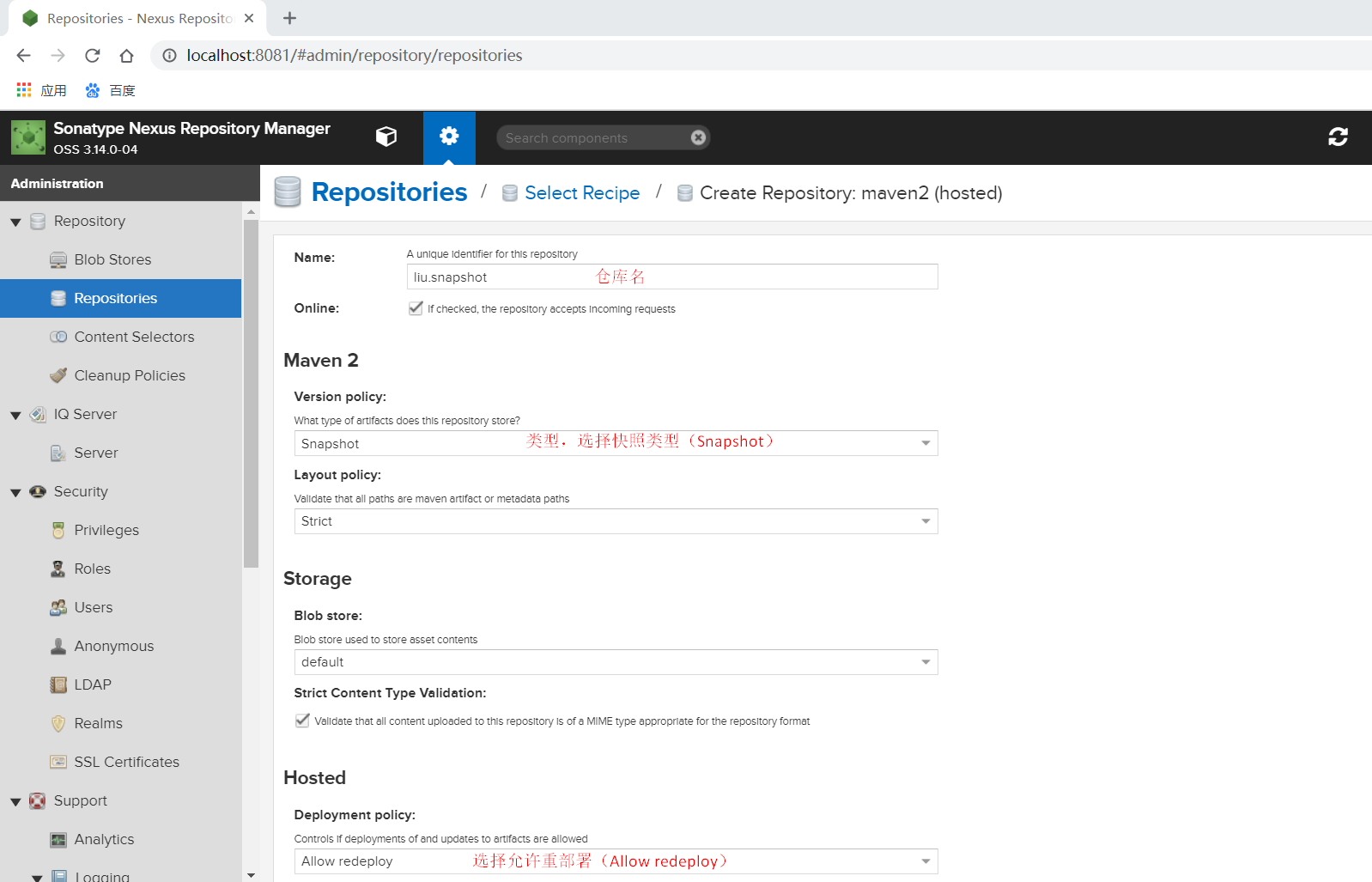
1. 进入仓库页面。如下：



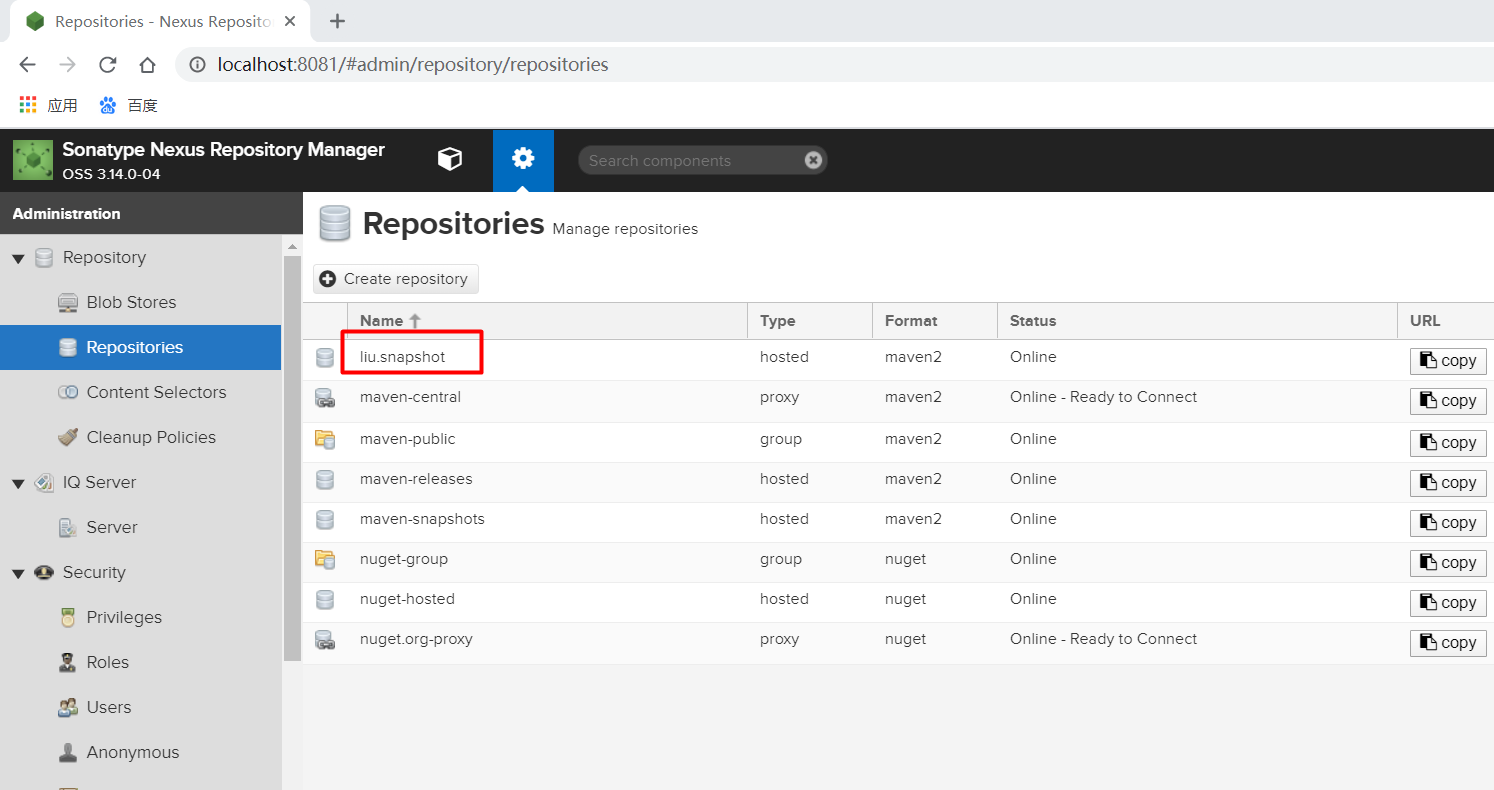
1. 进入仓库页面后，选择要创建的仓库类型。选择【Maven2 ( hosted)】。如下：



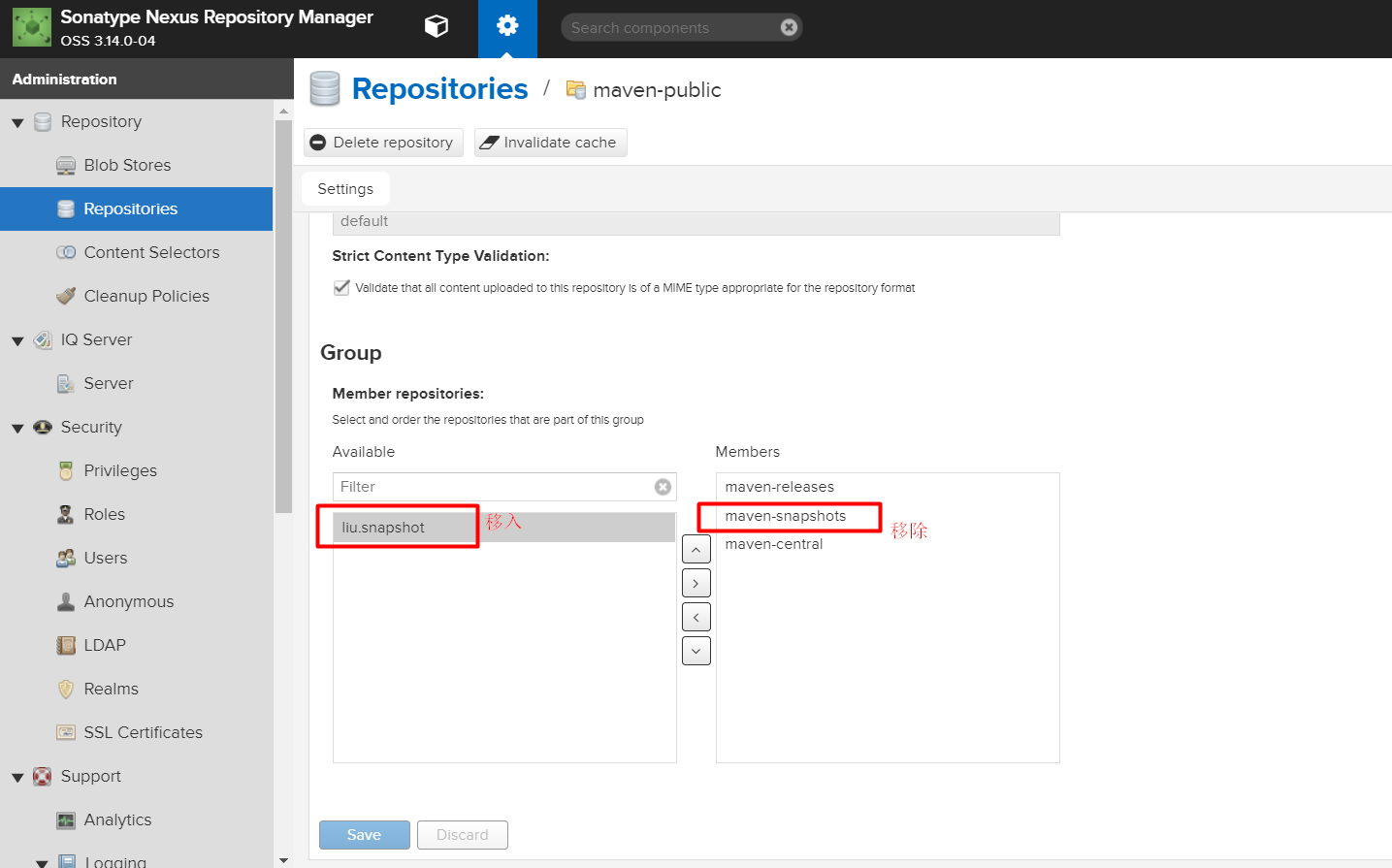
1. 进行仓库创建页面。现在我们创建一个快照仓库来替代默认的快照仓库。如下：



1. 填写完成后，点击下方的【Create repository】创建仓库。在仓库列表中即可看到自定义的仓库。如下：

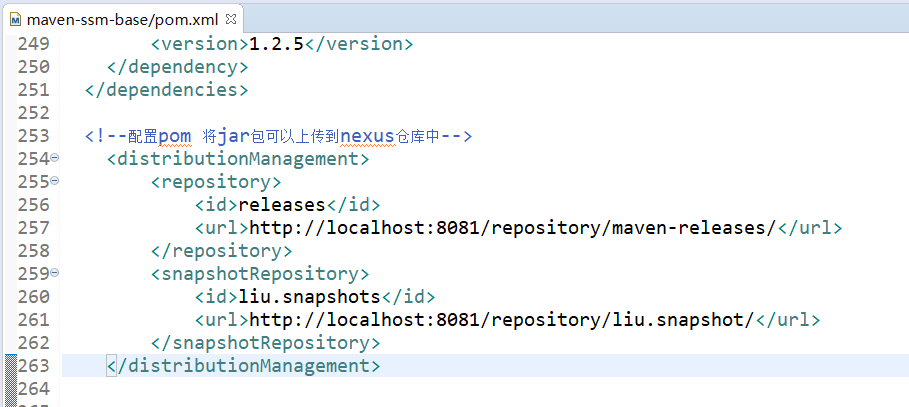


1. 进入仓库列表，进入【mavne-public】仓库，将刚刚创建的仓库添加到maven-public中，并将默认的快照仓库移除。如下：



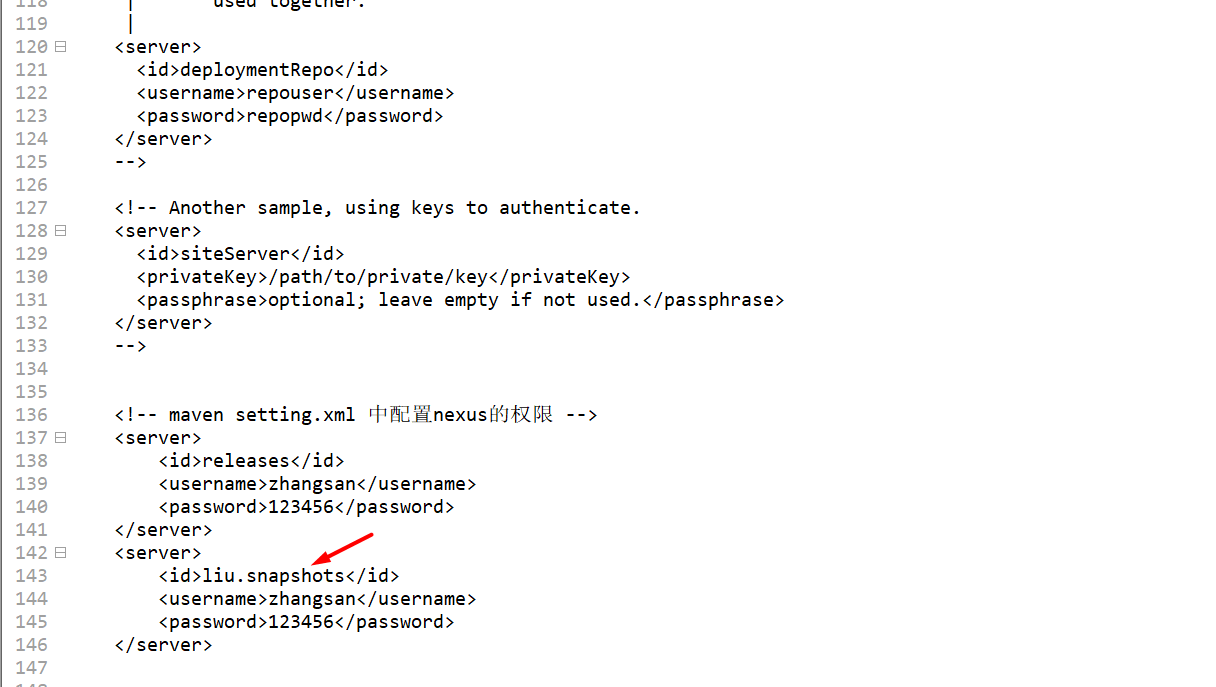
点击保存。这样，这个仓库就可以用来使用了。

1. 复制刚刚新建仓库的url地址，进入到父项目的pom.xml文件，对快照仓库信息进行修改。如下：



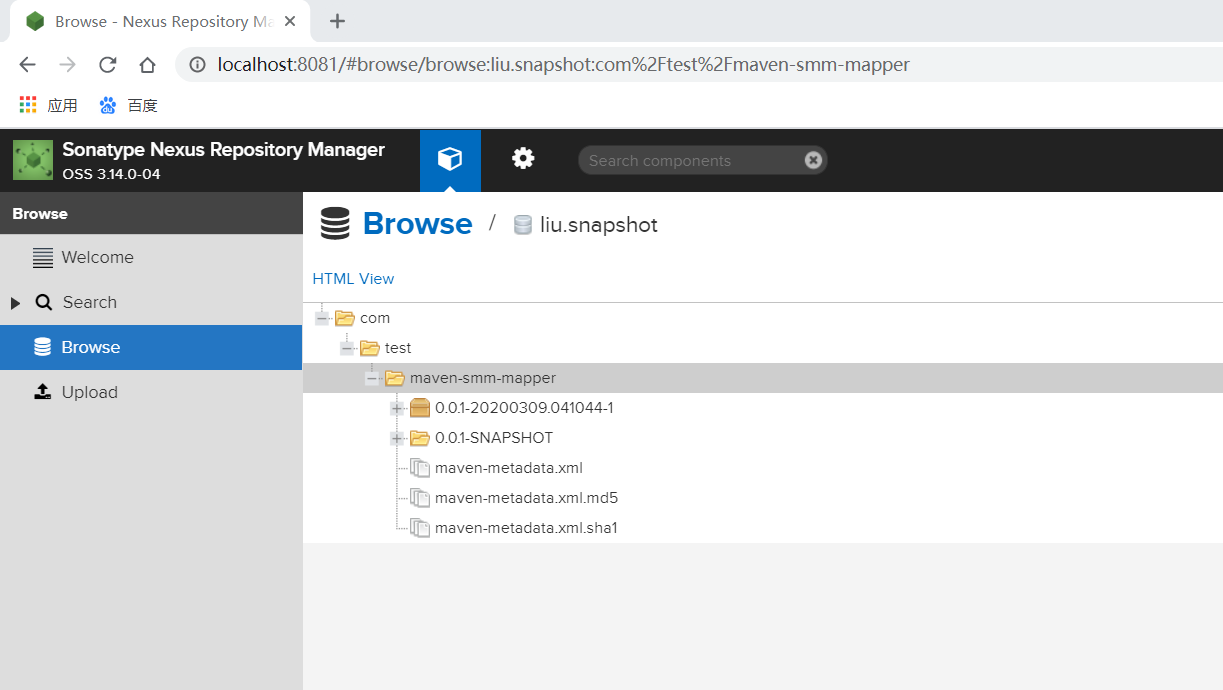
<id>标签中一般设为自定义仓库的名称，<url>为自定义仓库的url地址。

1. 修改了父项目的pom.xml文件，别忘了修改maven根目录/conf/settings.xml文件，因为pom.xml文件是向settings.xml文件看齐的。注意两者<id>标签中的值要相同。如下：



顺便把用户信息也修改一下。

此后，用户上传jar包时将上传到新建的仓库中。



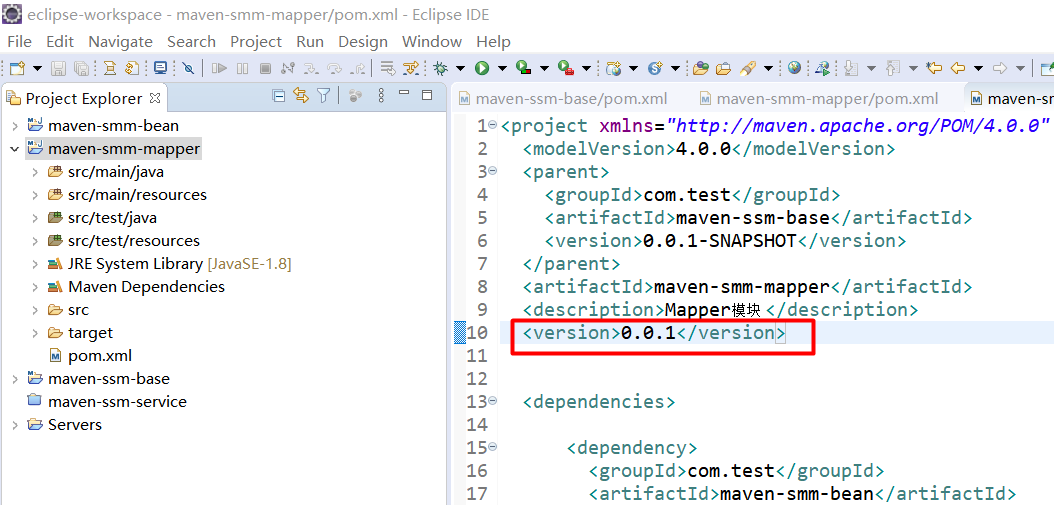
# 正式版jar包的管理

## 正式版jar包的上传与引用

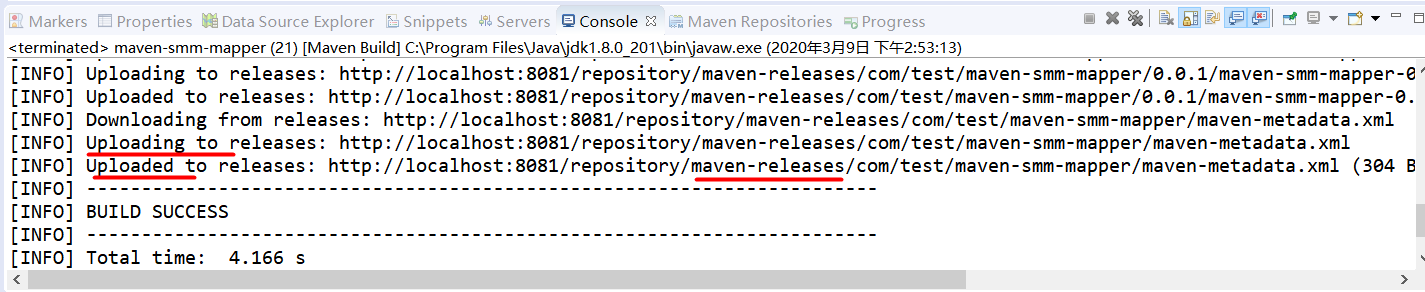
其他模块开发者在下载远程仓库jar包时，会下载最新版的jar包到其本地仓库中。

前面的学习都是将jar包发布到快照仓库中，下面将介绍如何将jar包发布到正式仓库中（maven-releases仓库）。

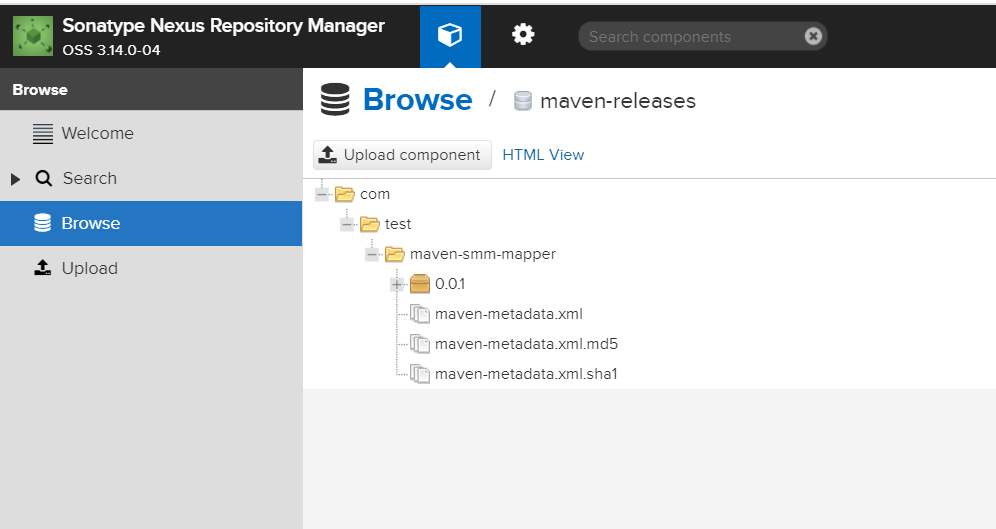
1. 打开mapper模块的pom.xml文件，添加版本号。如下：



1. 选中mapper模块鼠标右击，使用deploy命令进行上传。上传结果如下：

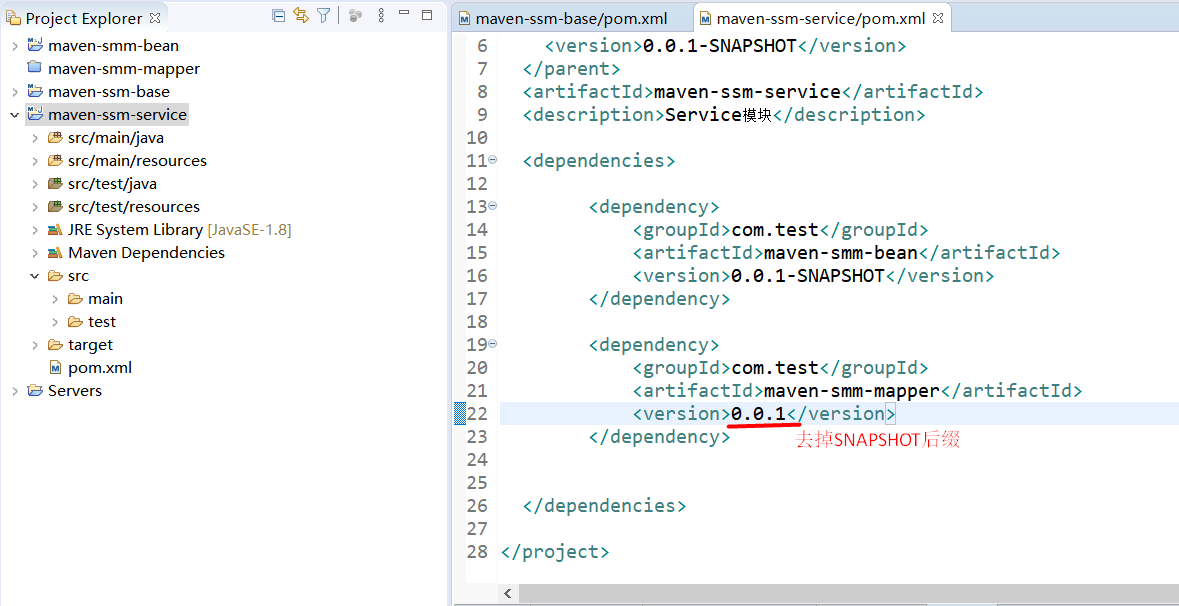


进入后台查看是否上传。如下：



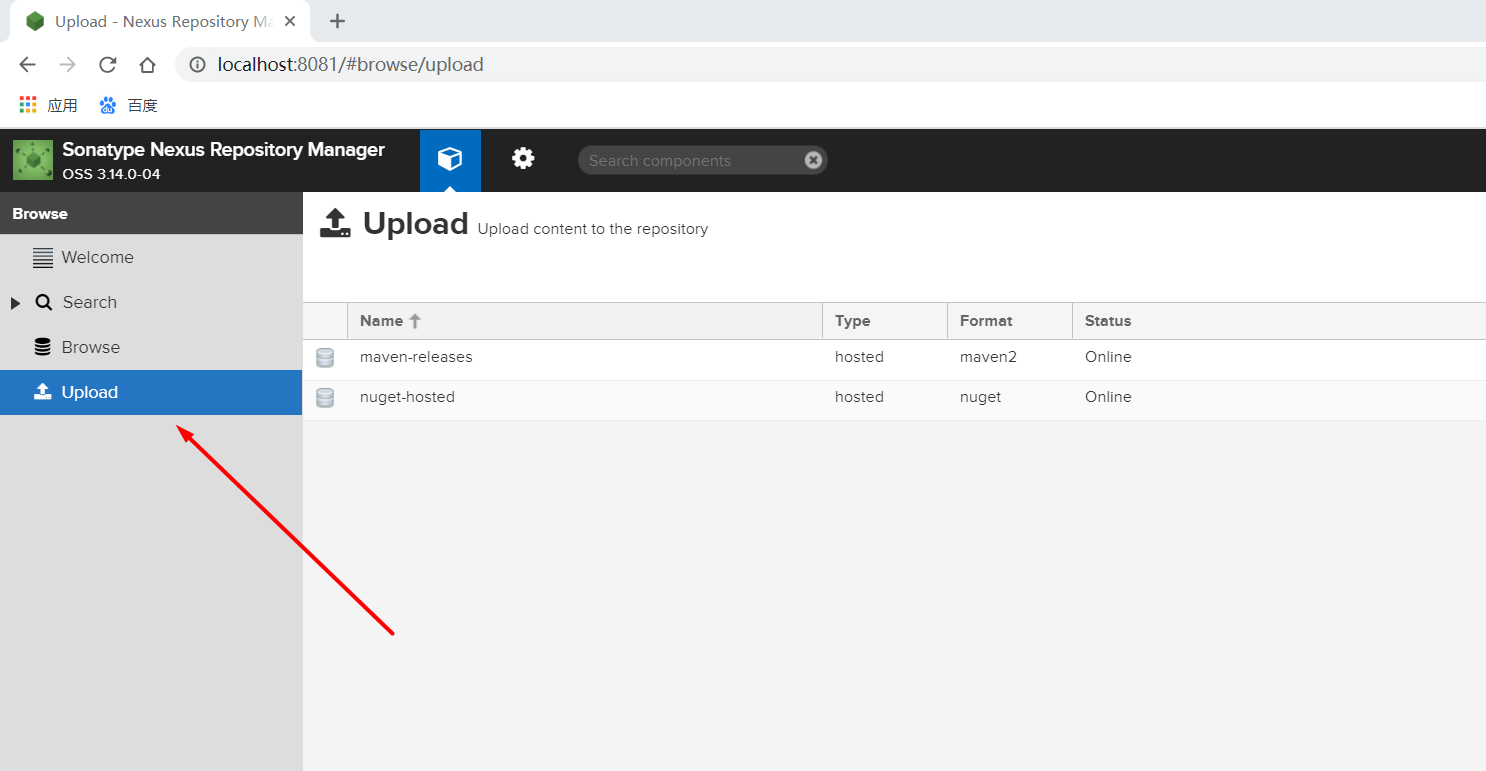
可以看出，已经成功的上传到了正式版仓库中。

1. 上传到正式版仓库中，此时service模块也必须在其pom.xml修改依赖，引入正式版的jar包。如下：



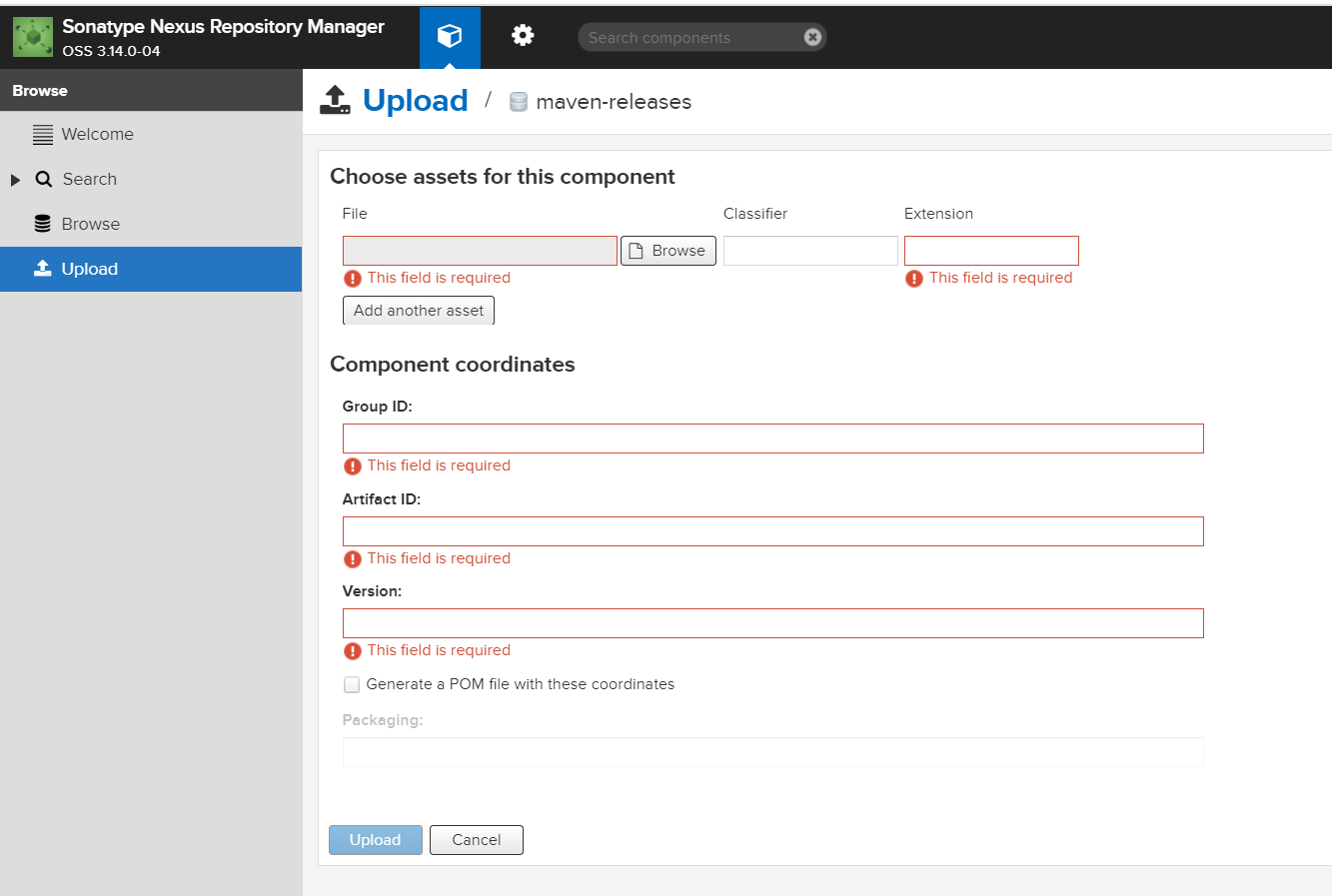
## 上传第三方jar包到远程仓库

1. 进入Nexus管理后台，进入Upload页面。如下：

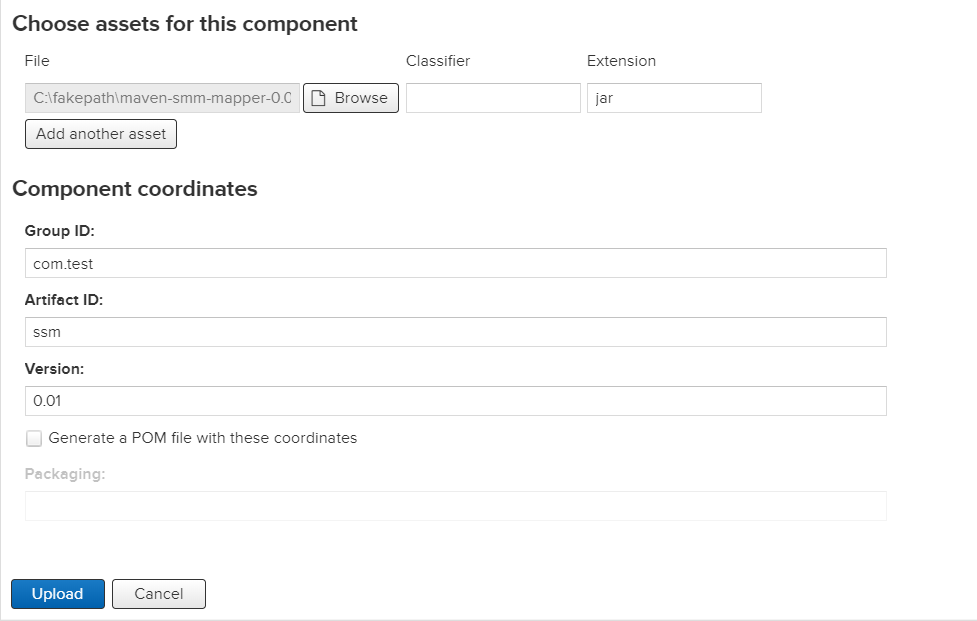


在这里，我们可以上传第三方jar包到本地仓库，将其保存在maven-releases仓库中。

1. 点击maven-releases仓库，进入以下界面：

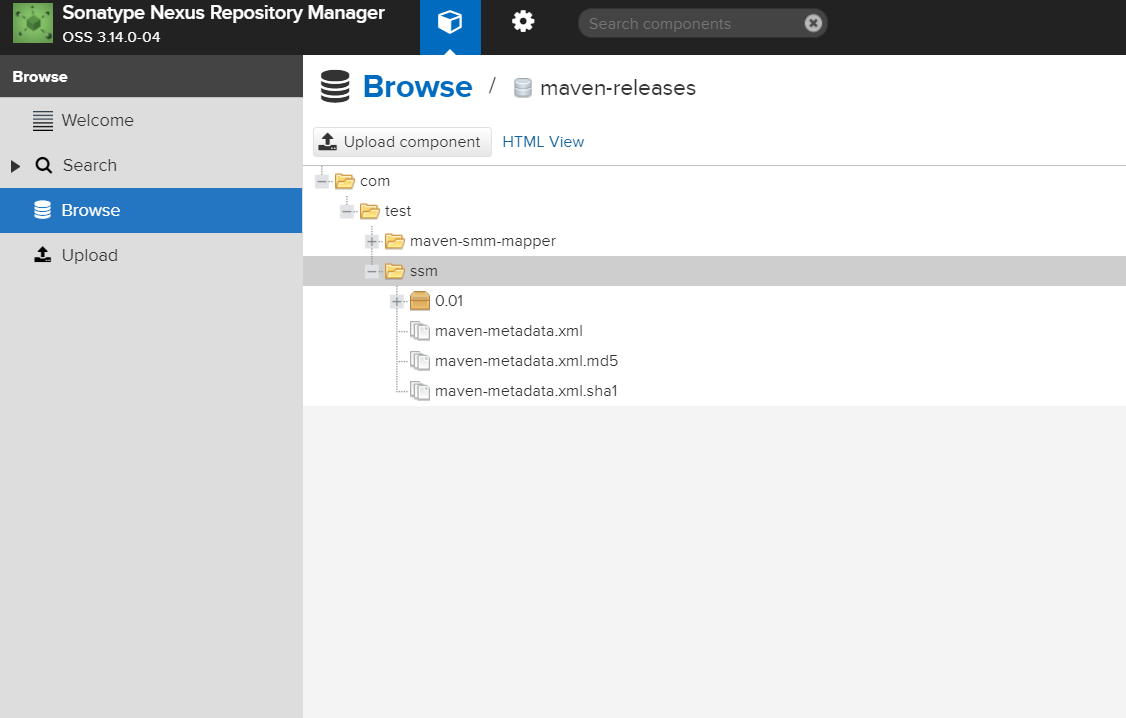


在这里，选择你要上传的第三方jar包。在选择要上传的jar包时需要注意，jar名称带有SNAPSHOT字样的是上传不上去的。



填写完毕后，点击【Upload】上传。

1. 上传成功后，进入【maven-releases】仓库即可看到刚刚上传的jar包。如下：



这样，我们就完成了将本地的第三方jar上传到远程仓库了。