# Nexus3私服搭建及maven库的使用

## [安装nexus3.x](http://blog.csdn.net/sinat_34017357/article/details/51776641)

[环境说明](http://blog.csdn.net/sinat_34017357/article/details/51776641" \l "t0)

[安装jdk](http://blog.csdn.net/sinat_34017357/article/details/51776641" \l "t1)

[安装nexus](http://blog.csdn.net/sinat_34017357/article/details/51776641" \l "t2)

### 环境说明

安装环境：

操作系统：linux 64位。

JDK：jdk1.8 64位

nexus：nexus3.2.0

### 安装jdk

nexus3.x需要JDK1.8支持，所以我们首先在[Linux](http://lib.csdn.net/base/linux" \t "_blank" \o "Linux知识库)下面安装JDK1.8.   
JDK下载地址：[http://www.oracle.com/technetwork/cn/java/javase/downloads/jdk7-downloads-1880260.html](http://www.oracle.com/technetwork/cn/java/javase/downloads/jdk7-downloads-1880260.html" \t "_blank)

第一步：将下载的包解压到自己的安装目录

tar -xvzf jdk-8u74-linux-x64.tar

第二步：配置系统环境变量   
使用vi编辑/etc/profile文件：

vi /etc/profile

将下面的代码添加到profile文件末尾：

export JAVA\_HOME=/opt/program/jdk1.8.0\_74

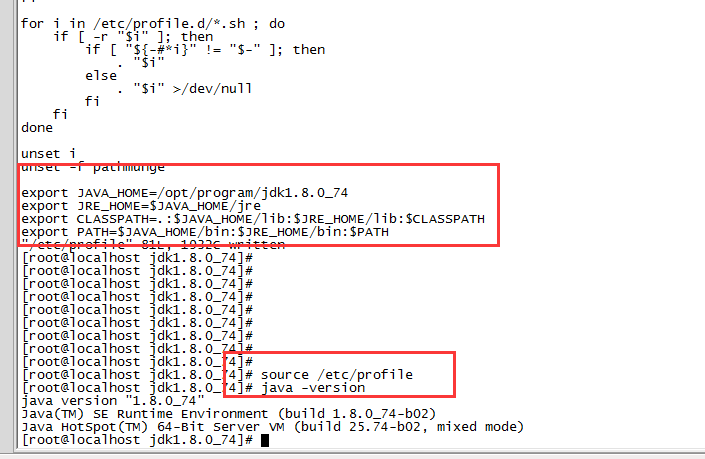
export JRE\_HOME=$JAVA\_HOME/jre

export CLASSPATH=.:$JAVA\_HOME/lib:$JRE\_HOME/lib:$CLASSPATH

export PATH=$JAVA\_HOME/bin:$JRE\_HOME/bin:$PATH

然后保存退出：wq   
生效profile文件，命令如下：

source /etc/profile

第三步：验证   
输入[Java](http://lib.csdn.net/base/javase" \t "_blank" \o "Java SE知识库) -version命令，如果得到如下信息表示安装成功：   


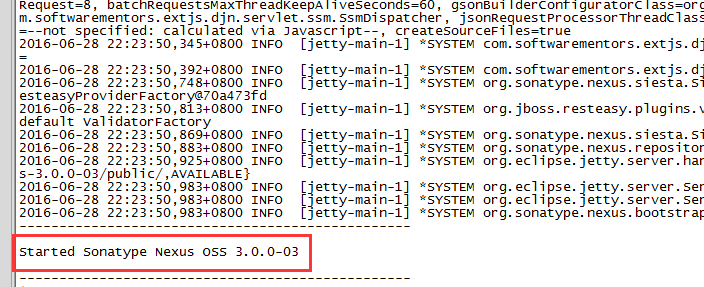
### 安装nexus

nexus下载地址：[http://www.sonatype.com/download-oss-sonatype](http://www.sonatype.com/download-oss-sonatype" \t "_blank)

第一步：将下载的文件放到安装目录下，解压   
tar -xvzf nexus-3.0.0-03-unix.tar

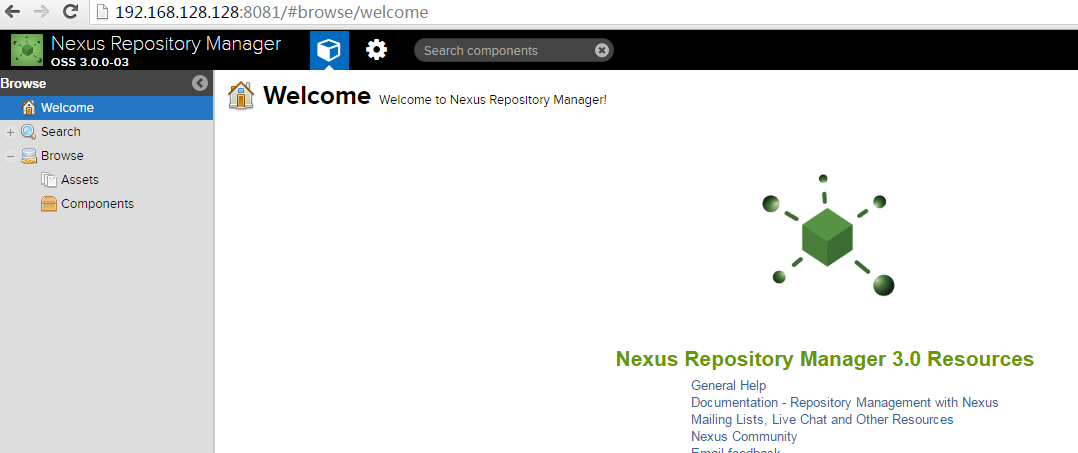
第二步：启动nexus   
进入bin目录，运行如下命令启动（&符号表示后台启动）：

./nexus start

然后通过打印的日志就可以产看是否启动成功。   


第三步：访问私服   
nexus启动成功之后，我们就可以访问咱们的私服了。默认的端口是8081， RUL为：[http://serveripaddress:port](http://serveripaddress:port" \t "_blank)，例如： [http://localhost:8081/](http://localhost:8081/" \t "_blank)。

可以看到，已经可以访问了：



当然你也可以使用默认账号admin/admin123登录。

OK，到这里一个maven私服就搭建好了！

### nexus仓库类型说明

proxy代理仓库：代理仓库，通常被用来代理远程的公共仓库。

例如maven中央仓库，我们可以通过maven settings设置镜像，来帮助我们下载服务器的jar包。

hosted本地仓库：通常我们会署自部己的构件到这一类型的仓库。

例如ojdbc.jar涉及版权问题，可以将jar通过mvn deploy发布到hosted类型的仓库中，以供开发人员下载使用

group仓库组：用来组合、合并多个hosted/proxy仓库。

例如我们的构建计划中会把hosted和proxy仓库组合起来，我们只需要在maven settings中设置一个镜像，即可随意的下载hosted、proxy中的任何jar包。

## 仓库构建计划

构建maven2(proxy)仓库wonders-proxy，用来代理

构建maven2(hosted)仓库wonders-hosted，用来管理公司开发的jar和一些无法下载的jar包

构建maven2(group)仓库wonders-group，用来合并proxy和hosted仓库，方便开发人员使用

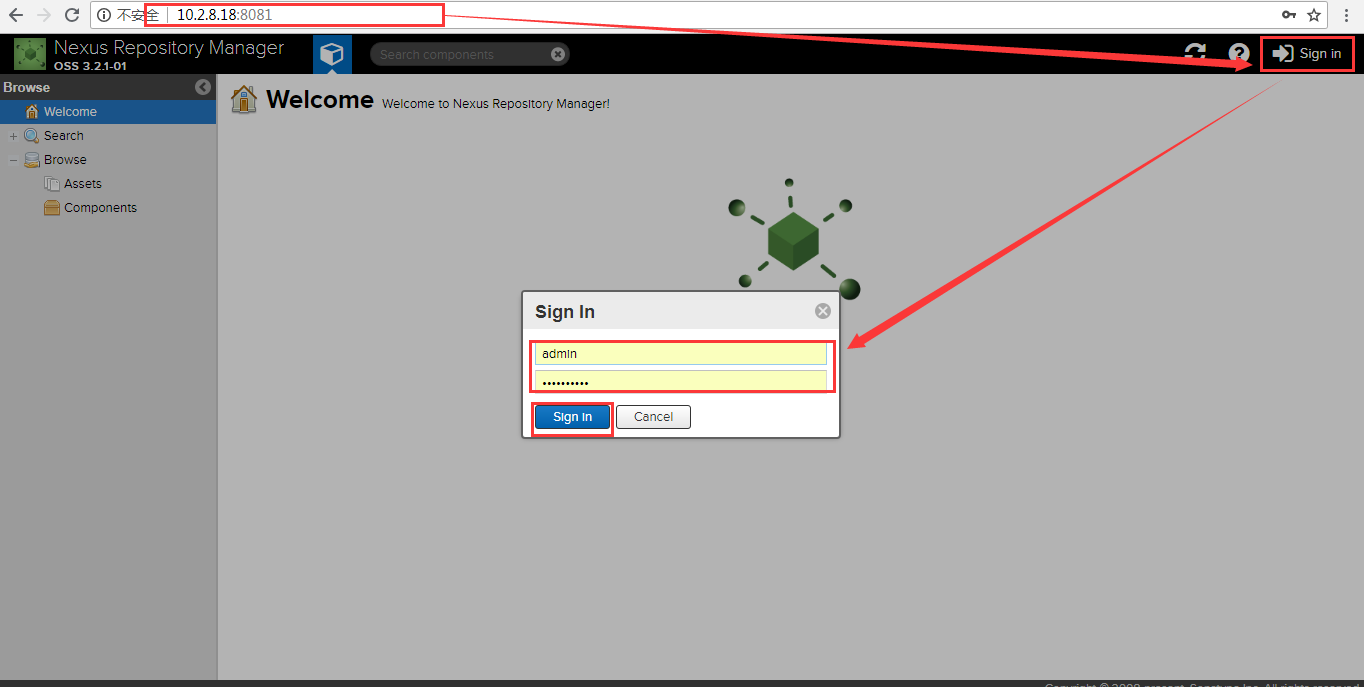
## Nexus私服搭建

### 登录

公司Nexus私服地址：<http://10.2.8.18:8081>

管理员帐号密码：admin/\*\*\*

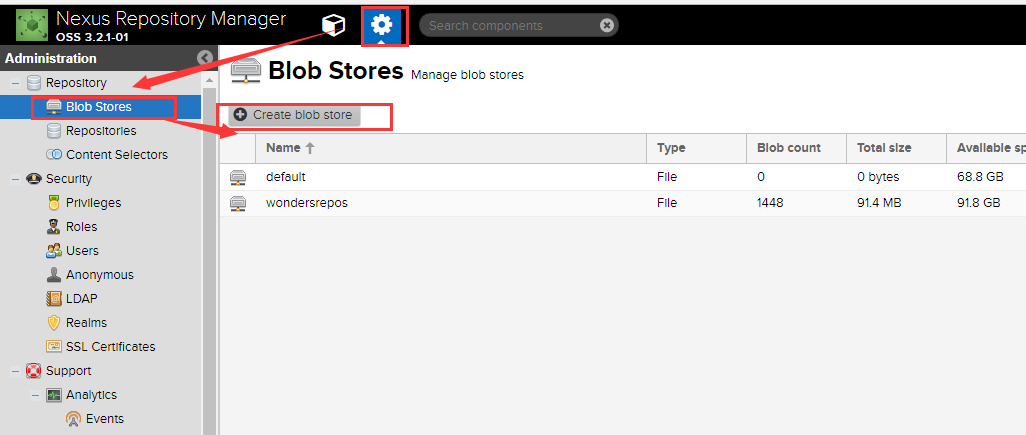
开发人员帐号密码：wonderes/\*\*



### 文件存储

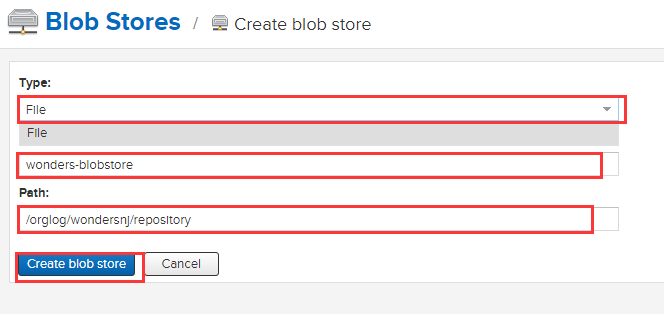
即仓库中的jar包实际存储的位置

#### 创建自定义文件存储位置



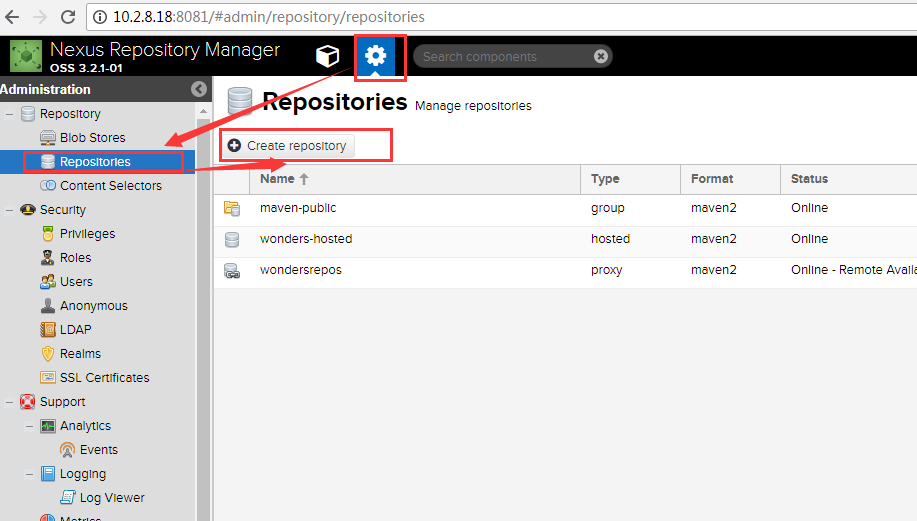
检查磁盘空间情况，尽量使用所在磁盘空间充足的文件夹。

常用命令：df -h

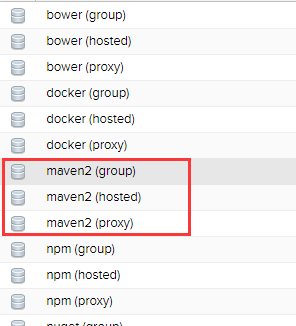


填写名称和存放路径后，创建即可。

### 创建仓库

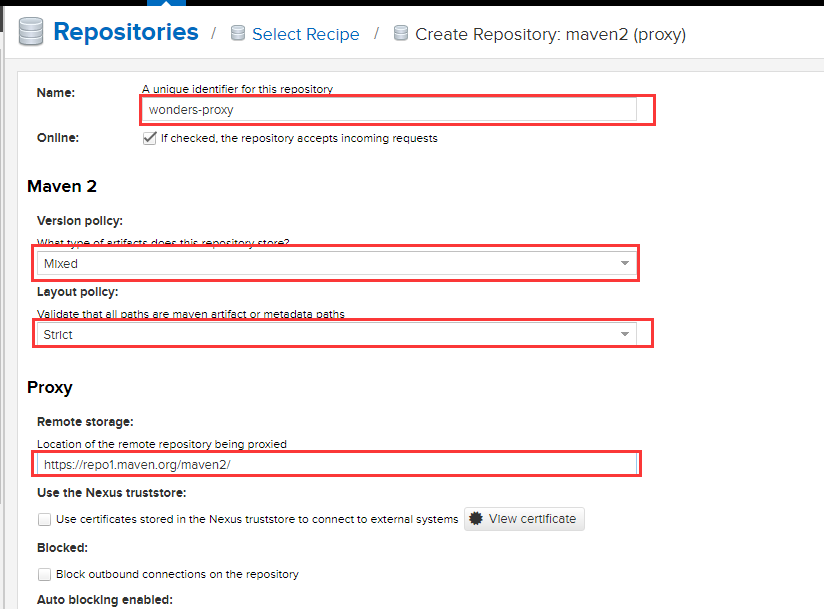


选择仓库类型



#### proxy仓库

上一步骤选择仓库类型maven2(proxy)



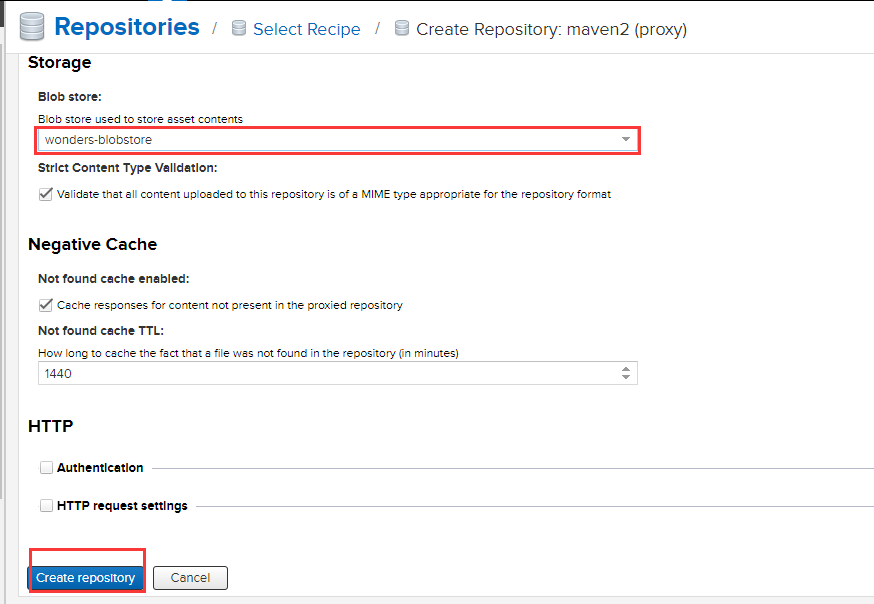
Name：就是为代理起个名字

Remote Storage: 代理的地址，

常用地址：

Maven中央仓库的地址为: https://repo1.maven.org/maven2/

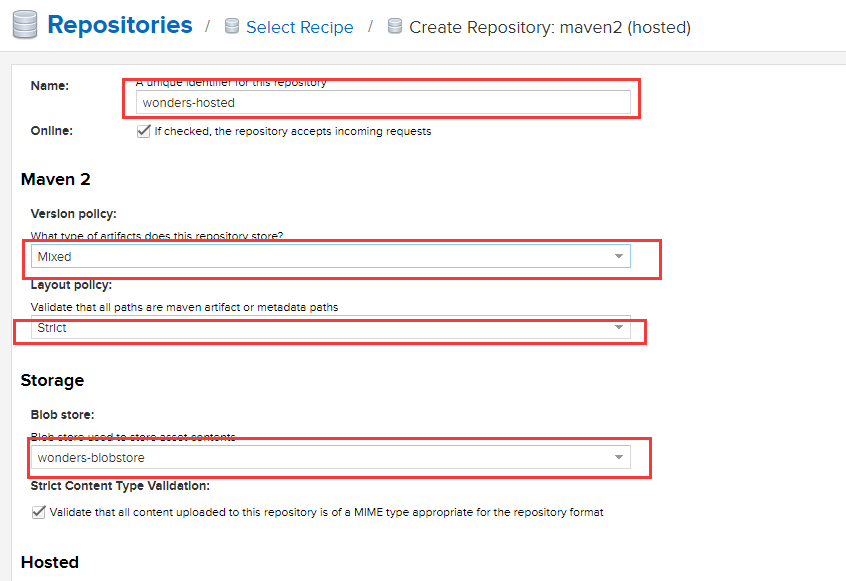
开源中国仓库地址：http://maven.oschina.net/content/groups/public/

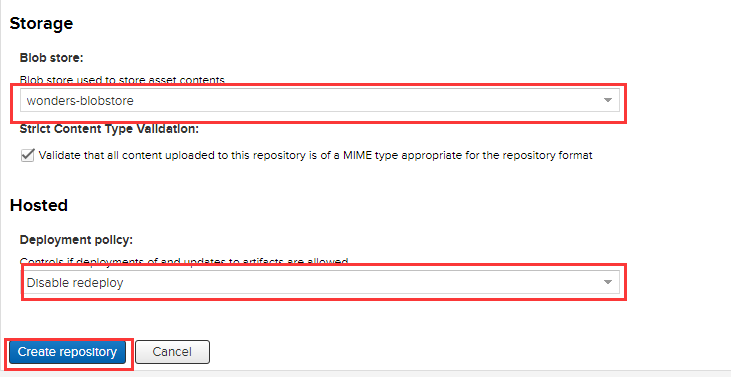


选择自己的文件存储，创建即可。

#### hosted仓库

上一步骤选择仓库类型maven2(hosted)

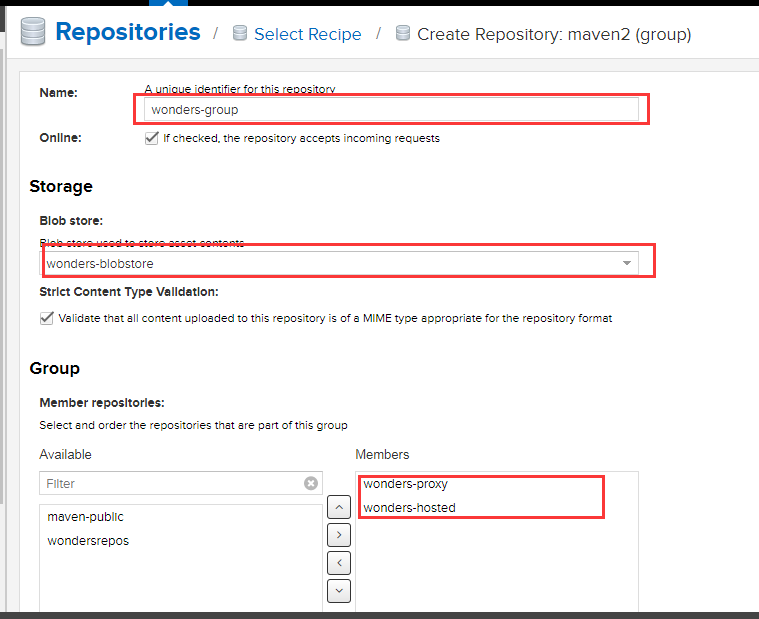


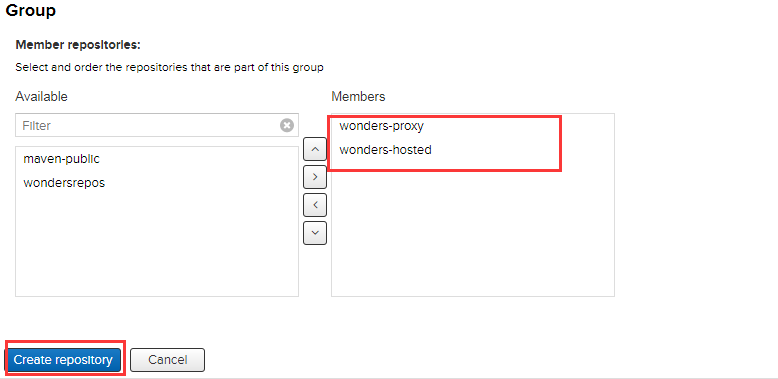


较为简单，只要选择允许部署和文件存储，保存即可

#### group仓库

上一步骤选择仓库类型maven2(group)

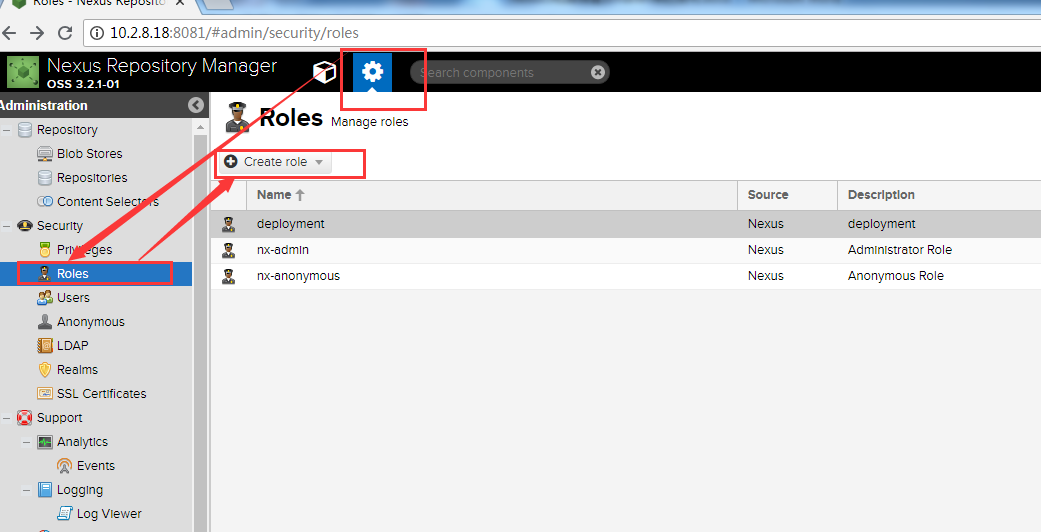


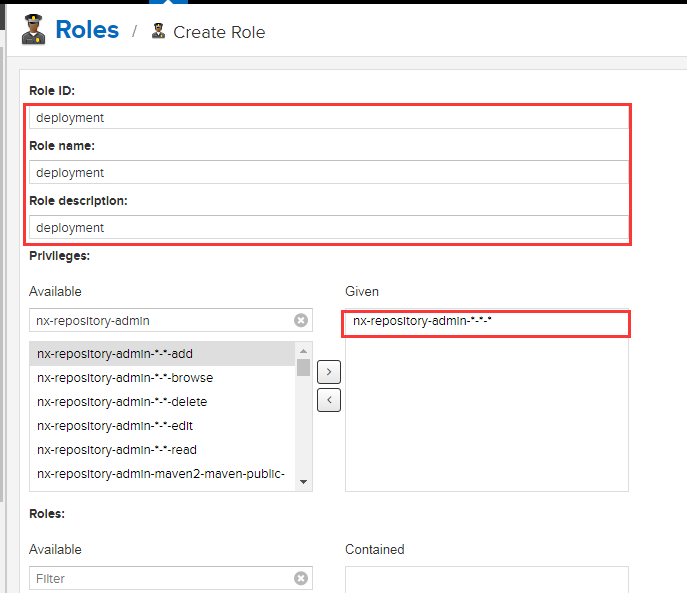


填写名称、存储位置、成员仓库（就是刚刚创建的hosted、proxy仓库），创建即可

### 用户权限管理

#### 角色

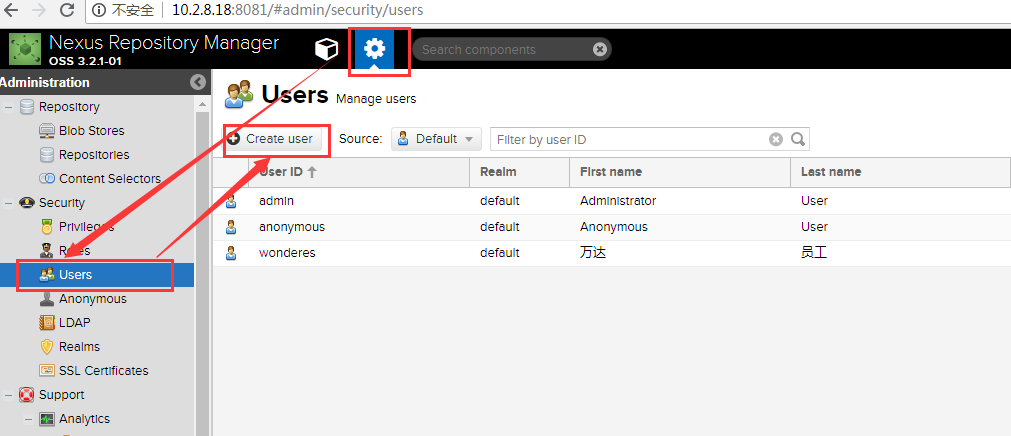


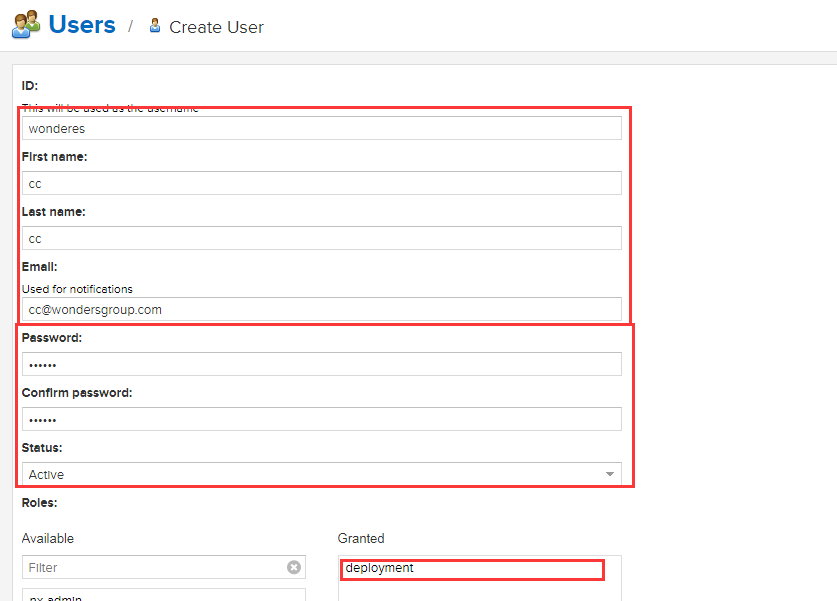


选择所需权限，点击创建即可。

图中例子为设置一个可以发布jar包的角色，因此选择nx-repository-admin-\*-\*-\*权限。

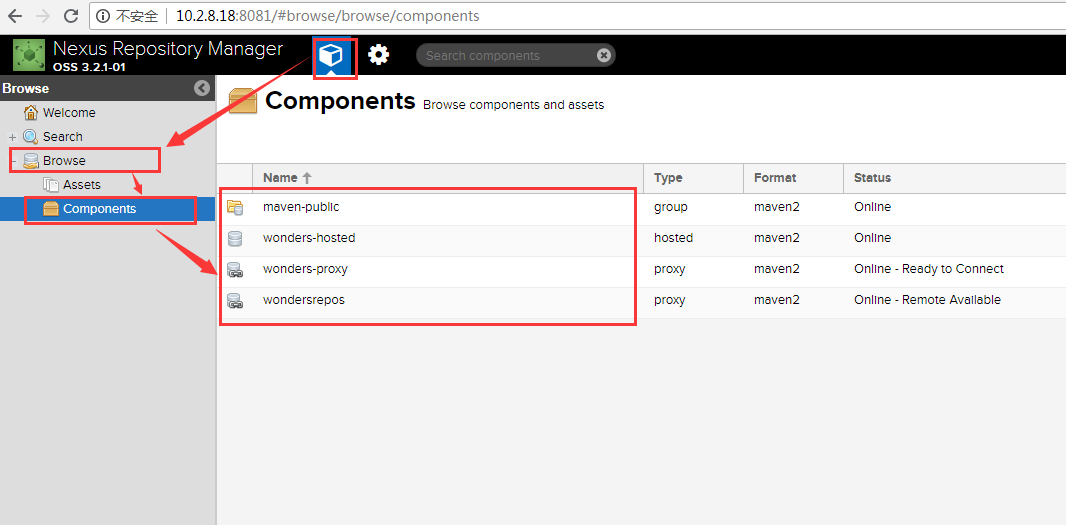
#### 用户

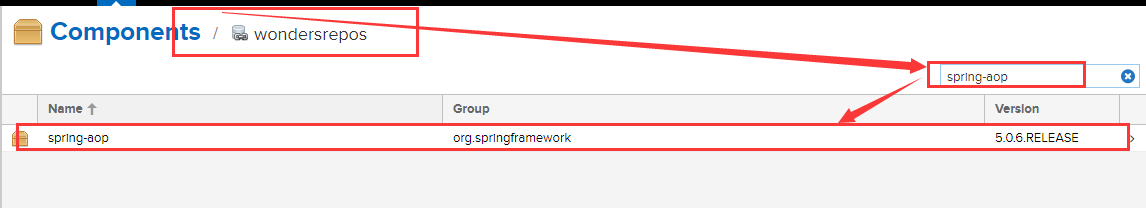




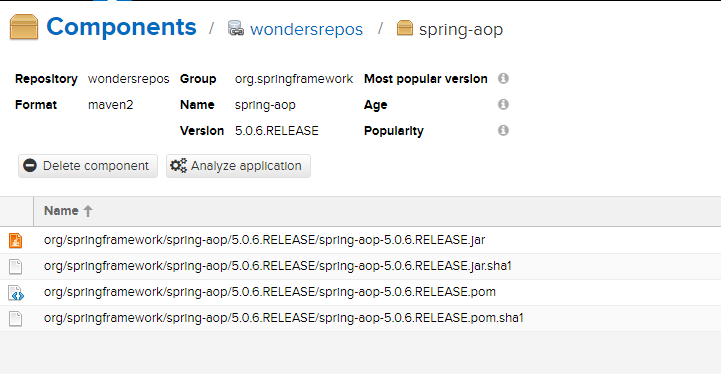
设置名称、邮箱、状态、角色信息，创建即可。

### 查询操作





选择库，输入资源关键字，即可查看是否存在和一些基本信息。



## 私服的使用

### 配置

开发人员需要在本地仓库的settings.xml

settings.xml所在的文件夹为C:\Users\${你的用户名}\.m2或者${MAVEN\_HOME}/conf中，根据实际的情况来找。

代码如下

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <settings xmlns=*"http://maven.apache.org/SETTINGS/1.0.0"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd"*>  <localRepository>e:\\repo</localRepository>  <servers>  <server>  <id>wonders-release</id>  <username>wonderes</username>  <password>123456</password>  </server>  </servers>  <mirrors>  <mirror>  <id>wonders-group</id>  <mirrorOf>\*</mirrorOf>  <url>http://10.2.8.18:8081/repository/wonders-group/</url>  </mirror>  </mirrors>  </settings> |

localRepository设置本地仓库路径，默认路径为C:\Users\${你的用户名}\.m2\repository

如果存在多个本地仓库，可以指定路径。

server用于mvn deploy上传时的默认用户密码。

mirror是代理仓库的镜像，可以下载仓库中的jar文件。

### 发布

#### Jar文件发布

命令

mvn deploy:deploy-file -DgroupId={groupid} -DartifactId={artifactid} -Dversion={version} -Dpackaging=jar -Dfile={jarfilepath} -Durl={nexus路径} -DrepositoryId={repositoryid}

nexus路径对应hosted的仓库路径

repositoryid对应settings.xml中的server id

如下是ojdbc8的发布命令

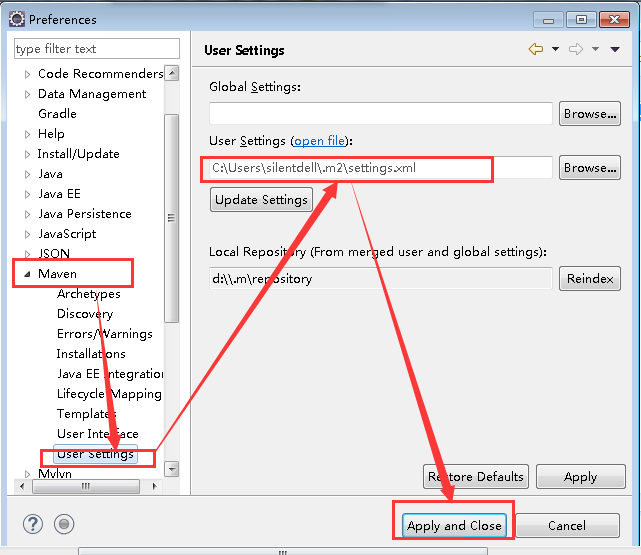
mvn deploy:deploy-file -DgroupId=com.oracle -DartifactId=ojdbc6 -Dversion=12.1.0.1 -Dpackaging=jar -Dfile=C:\Users\silent\Downloads\ojdbc6-12.1.0.1.jar -Durl=http://10.2.8.18:8081/repository/wonders-hosted/ -DrepositoryId=wonders-release

#### 项目自动发布

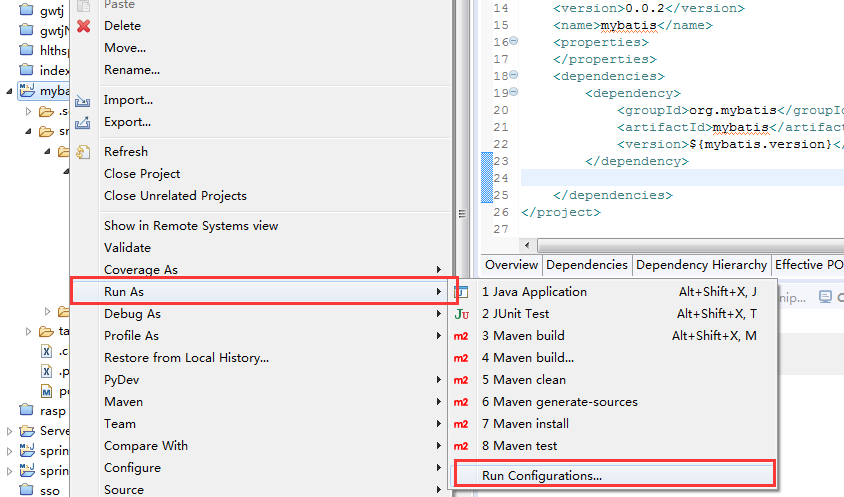
pom.xml中增加配置

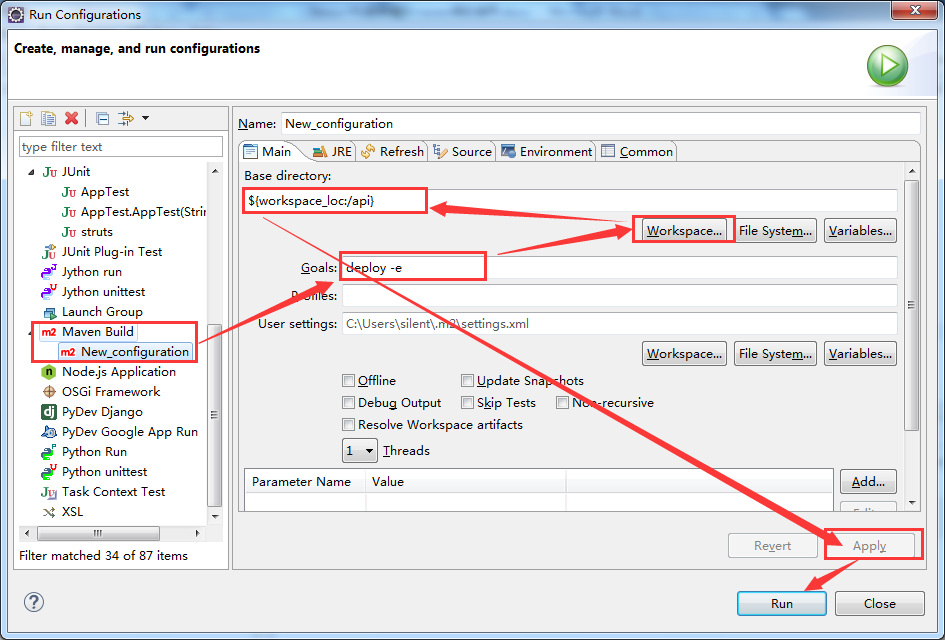
|  |
| --- |
| <distributionManagement>  <repository>  <id>wonders-release</id>  <url>http://10.2.8.18:8081/repository/wonders-hosted/</url>  </repository>  </distributionManagement> |

Eclipse中window-preferences设置一下settings.xml



Eclipse中配置mvn命令启动项





选择部署的项目workspace(工作空间)，run即可

### 下载

Eclipse中window-preferences配置一下即可

