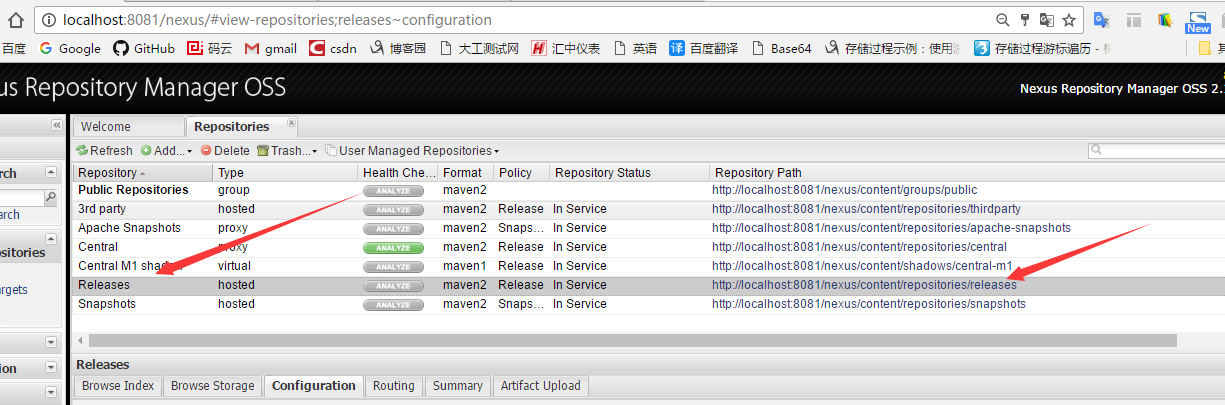
# 1、配置发布工厂

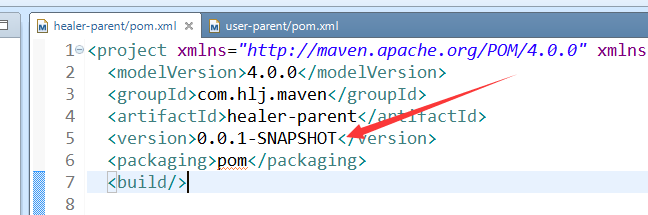
## 1、发布项目到realease工厂（url为下面的）

<http://localhost:8081/nexus/content/repositories/releases/>



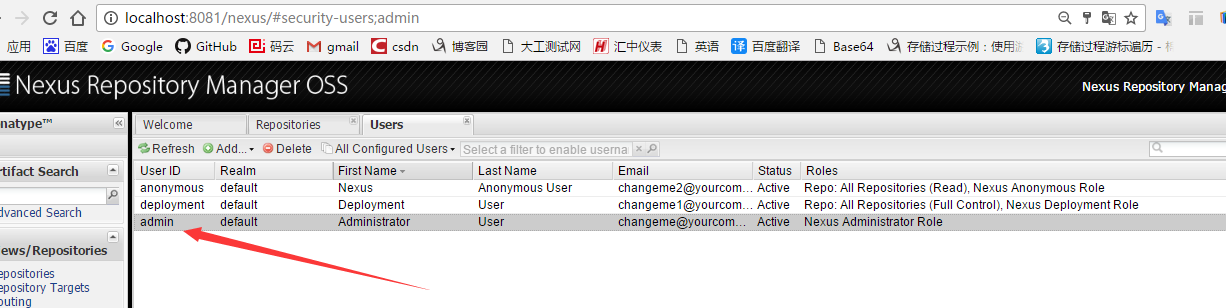
|  |
| --- |
| <distributionManagement>  <repository>  <id>healer-release</id>  <name>healer release resp</name>  <url>http://localhost:8081/nexus/content/repositories/releases/</url>  </repository> |

## 2、目前我们的项目设置的工厂为releases工厂，所以还需要添加一个snapshots工厂



|  |
| --- |
| <distributionManagement>  <repository>  <id>healer-release</id>  <name>healer release resp</name>  <url>http://localhost:8081/nexus/content/repositories/releases/</url>  </repository>  <snapshotRepository>  <id>healer-snapshots</id>  <name>healer snapshots resp</name>  <url>http://localhost:8081/nexus/content/repositories/snapshots//</url>  </snapshotRepository>  </distributionManagement> |

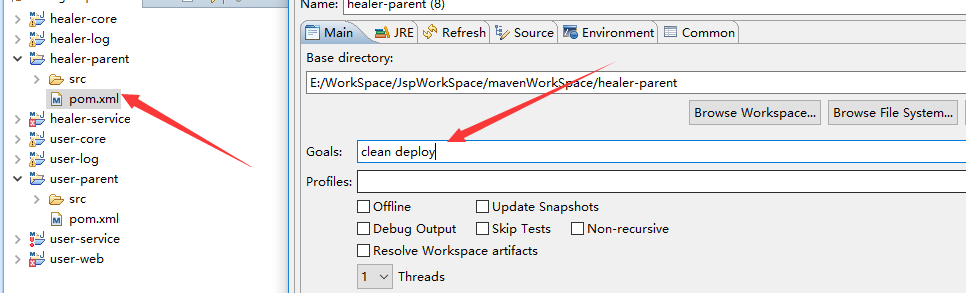
# 2、现在还不能发布，需要到nexus授权（使用admin用户）



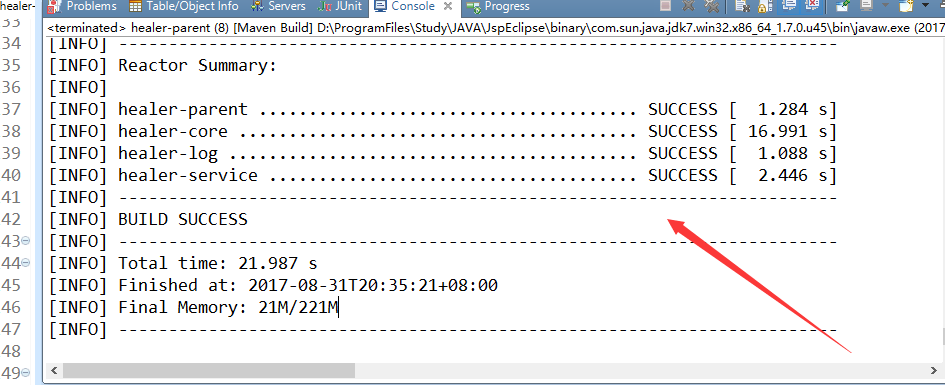
## 1、打开maven的setting.xml文件，找到server，admin默认的密码是132456,下面的id为上面发布工厂中仓库的id- healer-release

|  |
| --- |
| <server>  <id>healer-release</id>  <username>admin</username>  <password>123456</password>  </server>  <server>  <id>healer-snapshots</id>  <username>admin</username>  <password>123456</password>  </server> |

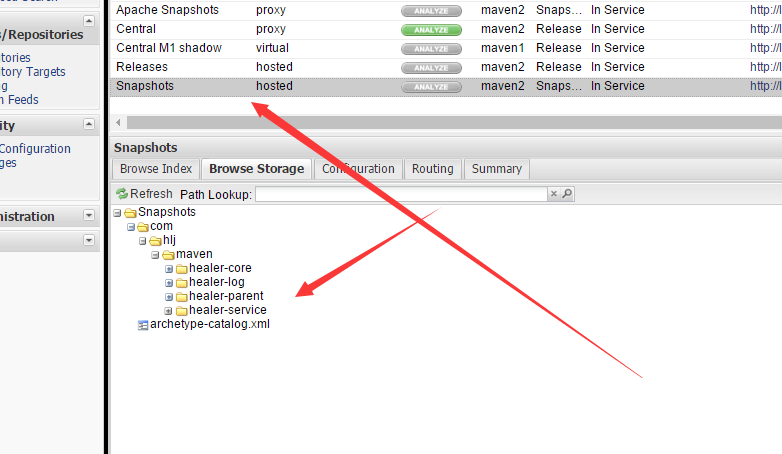
## 2、运行 clean deploy 发布工厂



### 1、运行成功



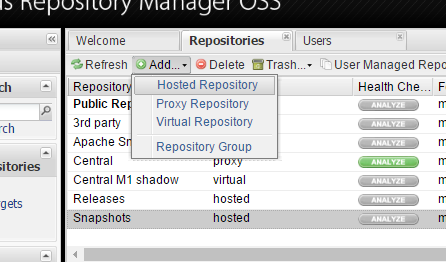
### 2、nexus上观察snapshots（因为我们的版本是snapshot）



### 3、成功

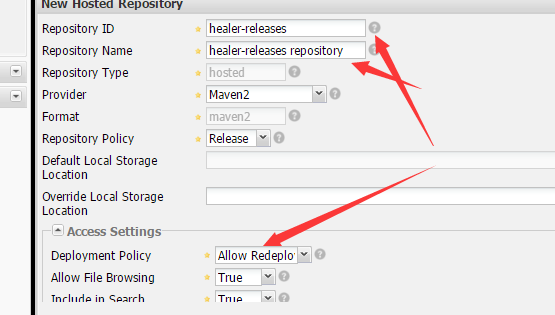
# 3、每个项目用独立的工厂

## 1、添加两个工厂healer-releases ,healer-snapshots

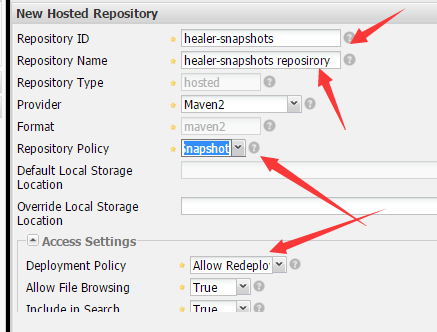


## 2、填写工厂名称（自定义）、描述、允许发布

### 1、healer-release工厂

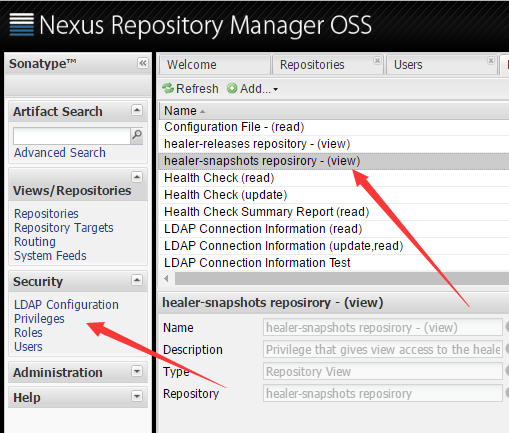


### 2、healer-snapshot工厂

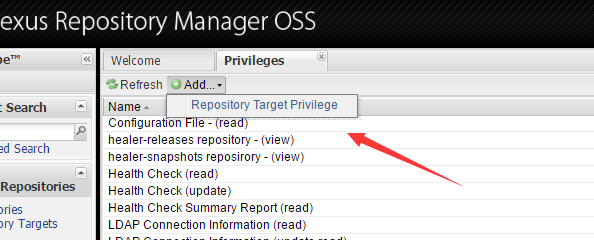


## 3、分配工厂的权限

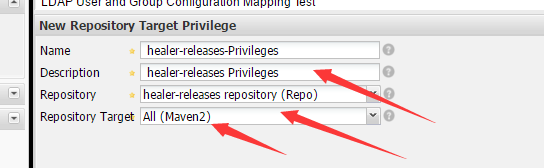
### 1、默认view，预览的权限

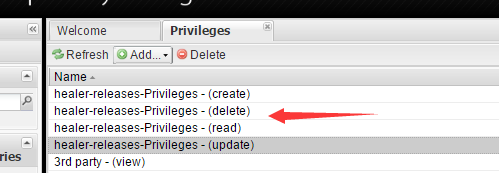


### 2、给我们添加的工厂授予权限

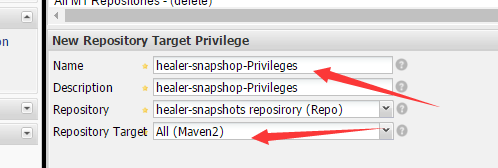


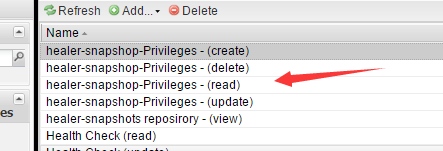
#### 1、给healer-release添加权限名称为healer-releases-Privileges，授予maven2所有的权限





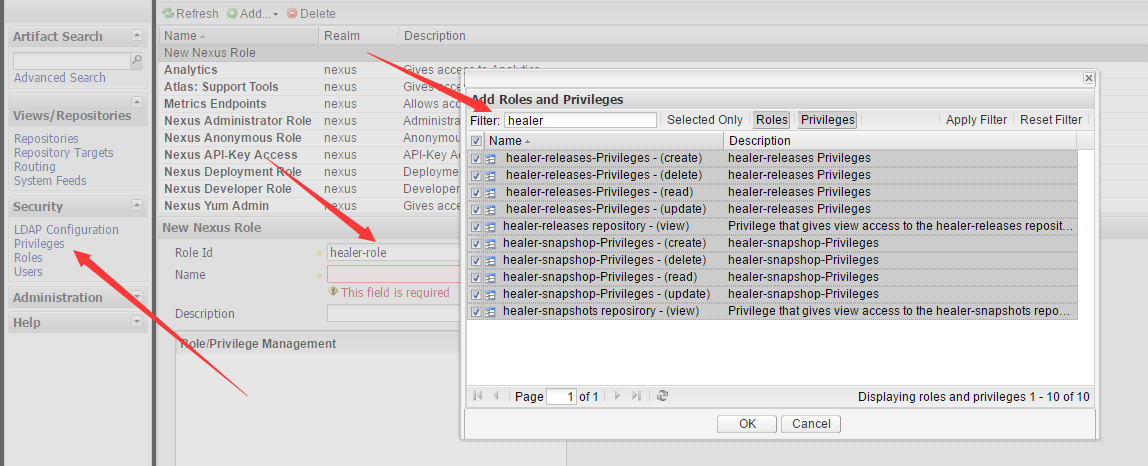
#### 2、给healer-snapshop权限

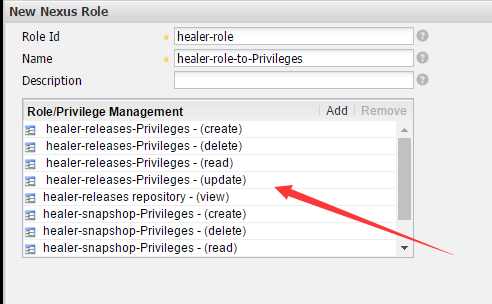




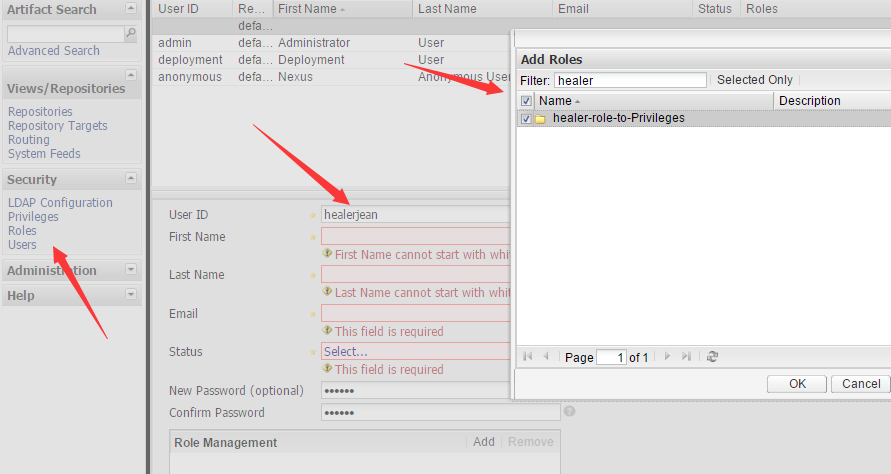
### 3、将权限分配给角色

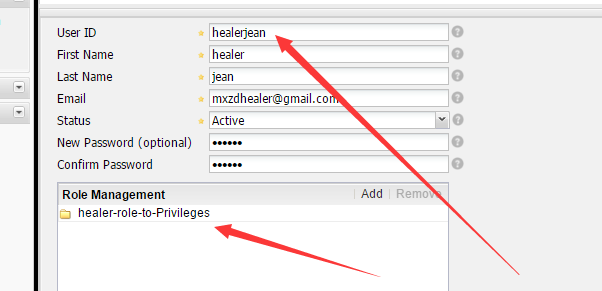
#### 1、添加一个haeler-role的角色，将有关这个项目的权限全部选中



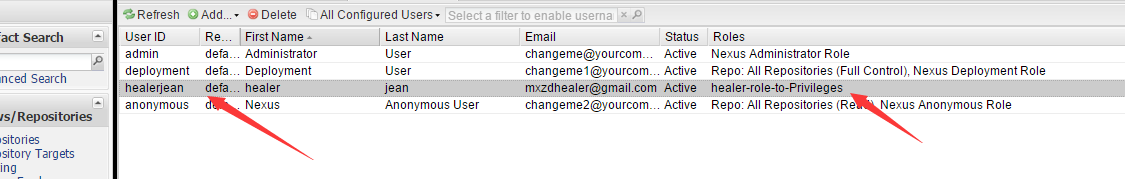


### 4、添加一个用户，将角色给用户（healerjean 147094）



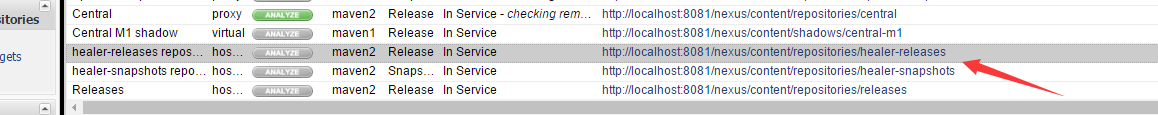


### 5、用户拥有角色，角色具备权限



## 4、使用我们自己的工厂进行发布

### 1、修改pom.xml中发布工厂相关的url，设置为我添加的用户所具备权限的所在的工厂

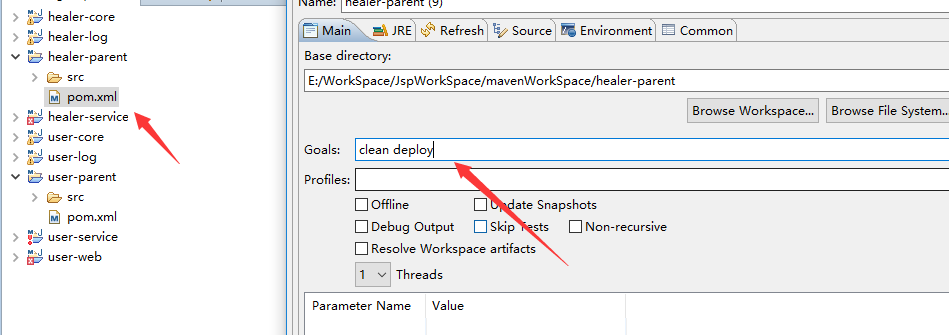


|  |
| --- |
| <distributionManagement>  <repository>  <id>healer-release</id>  <name>healer-snapshots reposirory</name>  <url>http://localhost:8081/nexus/content/repositories/healer-releases/</url>  </repository>  <snapshotRepository>  <id>healer-snapshots</id>  <name>healer-snapshots reposirory</name>  <url>http://localhost:8081/nexus/content/repositories/healer-snapshots/</url>  </snapshotRepository>  </distributionManagement> |

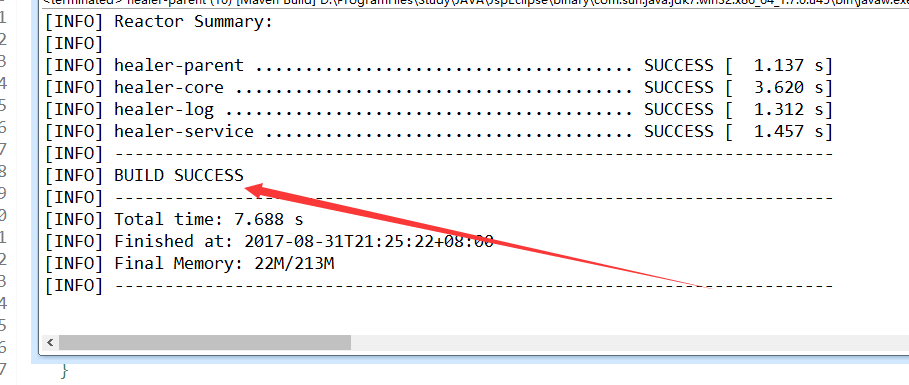
### 2、maven中setting.xml中添加以上我们自己添加工厂的所具备 权限的用户

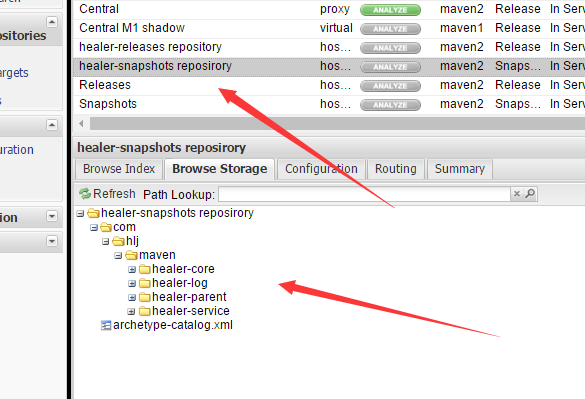
|  |
| --- |
| <server>  <id>healer-release</id>  <username>healerjean</username>  <password>147094</password>  </server>  <server>  <id>healer-snapshots</id>  <username>healerjean</username>  <password>147094</password>  </server> |

## 5、再次发布



### 1、发布成功





# 二、解释使用

<https://www.jianshu.com/p/084fd2408d9a>

# 4、解释[maven中snapshot快照库和release发布库的区别和作用](http://www.mzone.cc/article/277.html)

## 1、release和snapshot 区别

1、 snapshot 版本代表不稳定、尚处于开发中的版本，即快照版本

2、 release 版本代表功能趋于稳定、当前更新停止，可以用于发行的版本

## 2、使用场景

### 1、正在开发不停的更新 **更准确的说是不停 deploy 时**，deploy会发布到私服，则使用snapshot

* 不停更新/deploy 时，**版本号1.0不需更改**，私服中会**自动追加后缀时间**为版本名
* 其他系统使用时，会自动load**时间最近也即最新的版本**

#### 格式 <version>1.0-snapshot</version> 版本后面加一个 -snapshot

### 2、要上线一个版本，经过测试比较稳定，或者当第三方 jar 包功能确定时，可以提供一个release版本

 其他系统使用时，版本号不变，依赖包则不变，**不会自动load最新版本**

 上述有两个意思

* 假设第三方对 1.0 version 更新了，但本地有旧的 1.0 version，其他系统**不会更新引入私服中最新的1.0**，与snapshot的区别
* 第三方 升级了2.0，其他系统必须**手动更新**依赖的version为 2.0，否则不能引入最新版本，

#### 格式<version>1.0</version>，去掉-snapshot即可

## 3、说明

maven会根据模块的版本号(pom文件中的version)中是否带有-SNAPSHOT来判断是快照版本还是正式版本。

如果是快照版本，在不更改版本号的情况下，直接编译打包时，maven会自动从镜像服务器上下载最新的快照版本。

所以，我们在开发阶段，可以将公用库的版本设置为快照版本，而被依赖组件则引用快照版本进行开发，在公用库的快照版本更新后，我们也不需要修改pom文件提示版本号来下载新的版本，直接mvn执行相关编译、打包命令即可重新下载最新的快照库了，从而也方便了我们进行开发。

使用正式版本的模块，在不更改版本号的情况下，编译打包时如果本地已经存在该版本的模块则不会主动去镜像服务器上下载。