使用Nexus2.0搭建Maven私服

# 复习：

Maven

普通项目

Web项目 app

父类项目（统一版本号） 子类项目（继承父类，继承jar的版本）

依赖的项目（项目打包 安装到本地仓库 ） 其他的项目 依赖打包后的项目（mysql.jar）

聚合项目 SSM

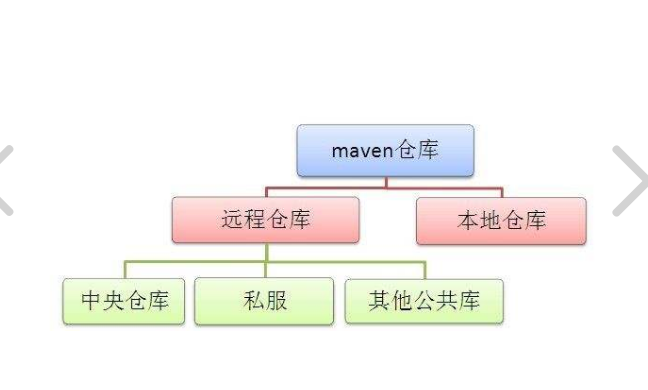
父类项目 统一jar包版本 ，pojo,mapper ,service ，web (子类项目)

# 目标

1 Linux 安装配置maven 私服

2 IDEA连接私服

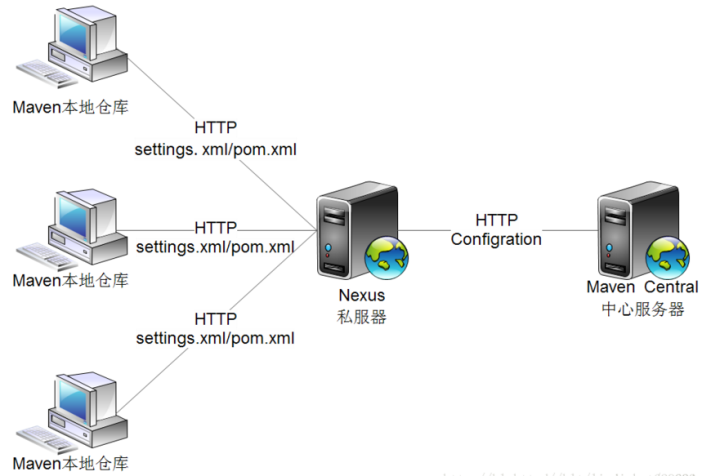
# 1 Maven私服



中央仓库是 官网

其他仓库 比如阿里云。。

私服公司内部 仓库



# 2 Maven搭建

## 2.1 安装配置Nexus环境

1. 安装环境准备
2. 服务器环境
   1. CentOS 6.4 64位
   2. JDK 1.8
3. nexus-2.12.0-01-bundle.tar.gz 复制到/usr/local/
4. 安装配置
5. 解压tar

tar -zxvf  nexus-2.12.0-01-bundle.tar.gz

1. 配置文件：conf/nexus.properties
2. 开放8081端口

vim /etc/sysconfig/iptables

service iptables restart  #最后重启防火墙使配置生效

1. 启动Nexus

注意：配置bin/nexus (RUN\_AS\_USER=root)

命令：cd /usr/local/nexus-2.11.4-01-bundle/nexus-2.11.4-01/bin

. /nexus start

1. 访问Nexus

http://服务器IP:8081/nexus/index.html

默认端口：8081

默认用户名/密码：admin/admin123

## 2.2 仓库管理

因为新搭建的Neuxs环境只是一个空的仓库，那么首先就需要与远程中心库进行同步，进行私有仓库索引更新。更新的方式有两种：自动方式、手动方式。

1. 自动更新方式

开启远程索引自动更新下载（Central Repository），设置Download Remote Indexes：True（远程下载Central仓库的索引）。Nexus会自动从远程中央仓库下载索引文件, 在Central上右键点击Update/RapairIndex，下载index文件。一般远程仓库都比较大，构件较多，索引文件会很大，比如：从http://repo1.maven.org/maven2下载后 就有2G，因此需要的时间就比较长。可以进入Scheduled Tasks查看任务的执行情况，当执行完成时，远程仓库的索引就已经建立完毕了。更新完索引后可在Browse Index页签点击刷新按钮就可显示所有更新后的索引。那么在这里由于网络原因，我们不推荐自动方式，强烈推荐手动方式。

1. 手动更新方式
2. **下载索引文件**

* nexus-maven-repository-index.gz

下载地址：http://repo.maven.apache.org/maven2/.index/

* nexus-maven-repository-index.properties

下载地址：http://repo.maven.apache.org/maven2/.index/

* indexer-cli-5.1.0.jar

下载地址：http://maven.outofmemory.cn/org.apache.maven.indexer/indexer-cli/5.1.0/

1. **解压缩索引文件**

将上面三个文件（.gz & .properties & .jar）放置到同一目录下，运行命令：

进入该路径 执行   
java -jar indexer-cli-5.1.0.jar -u nexus-maven-repository-index.gz -d indexer

1. **停止Nexus**

./usr/local/nexus2.12/nexus-2.12.0-01/bin/nexus stop

1. **删除原有的索引文件**

将{nexus\_home}\sonatype-work\nexus\indexer\central-ctx下的文件全部删掉，执行命令：rm \*

1. **拷贝索引至central-ctx目录下**

将nexus-maven-repository-index.gz解压后的indexer目录中所有文件，放到sonatype-work\nexus\indexer\central-ctx下面，执行命令：

cp -r \* /usr/local/nexus2.12/sonatype-work/nexus/indexer/central-ctx/

1. **启动Nexus即自动更新索引**

更新完索引后可在Browse Index页签点击刷新按钮就可显示所有更新后的索引，展开索引树，点击到叶子节点--依赖jar包，可查看其依赖配置坐标。

## 2.3Nexus仓库及常规操作介绍

### 2.3.1Nexus仓库

Nexus 的Repositories，一共默认有8个仓库，如下图2所示：



图2

上图中Nexus的预定义仓库的type类型、policy策略、格式均不同，下面依次进行介绍。

* 四种仓库类型：

1. Hosted Repository：本地仓库，通常我们会部署自己的构件到这一类型的仓库。比如公司的一些构件库等等。简单理解就是：内部项目的发布仓库（如：执行mvn clean install命令发布项目，最终发布的jar包会放入HostedRepository）
2. Proxy Repository：代理仓库，它们被用来代理远程的公共仓库，如Maven中央仓库。从远程中央仓库中寻找数据的仓库（可以点击对应的仓库的Configuration页签下Remote Storage Location属性的值即被代理的远程仓库的路径）
3. Virtual Repository：虚拟仓库
4. Repository Group：仓库组，用来合并多个hosted/proxy仓库。组仓库用来方便开发人员进行设置的仓库, group仓库可以自由组合其他的仓库为一个组，然后在项目的pom文件的repository可以配置这个仓库。可以说仓库组是为了简化仓库的引用而提出的。

* 两种仓库格式：

1. maven2
2. maven1

* 两种仓库Policy：

1. Release（发布版本，即稳定版本）
2. Snapshot（资源快照，版本相对不稳定）

下面我们再重点介绍几个Nexus预定义仓库：

1. 3rd party

存放第三方构件，一些无法从公共仓库获得的构件，比如一些早期开源项目，没有采用Maven方式构建的，还有一些非开源的组建等等，这些构件中央仓库里没有，需要用户手动上传的构件，那么都会存放在这个仓库里。在仓库的Artifact Upload页签中可以执行上传操作。

1. Apache Snapshots: 使用代理ApacheMaven仓库快照版本的构件仓库
2. Central: 用来代理Maven中央仓库中发布版本构件的仓库，策略为Release、只会下载和缓存中央仓库中的发布版本构件。

注意:该库是私服上的中央仓库，并不是Maven的中央库。可以把常用的jar包拷进去，这样客户端就可以从这里下载了。也可作为Maven中央库的代理。

1. Releases: 存放稳定版本的构件。比如：我们完成了一个版本下的组件开发,就可以把它发布到这里
2. Snapshots:存放快照版本的构件。比如：在下一个组件版本的所有开发和测试工作完成之前，是不应该发布到release仓库的，但可能其他项目只需用到这个组件的某些接口，只要这些接口完成了开发并通过测试，就可以拿来使用, 从而实现多个项目并行开发。
3. PublicRepositories:  仓库组

如下图3所示，Configuration里面，Ordered Group Repositories，包含了三个仓库Releases，3rd party和Central。也就是说我们在pom.xml引用这个仓库组，其包含的仓库也被引用。

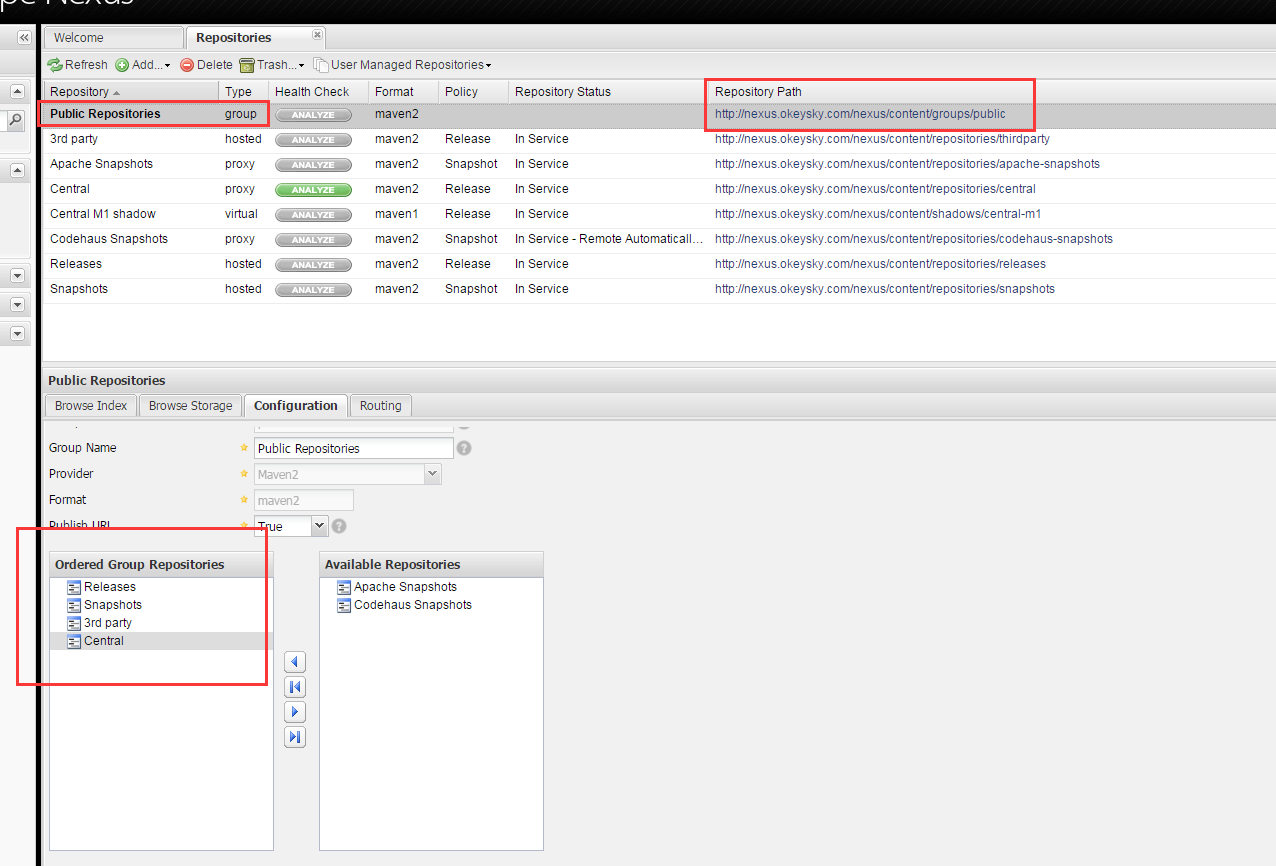


图3

### 2.3.2常规操作

1. 修改admin用户密码

点击左侧菜单栏的Security——>Users，可看到系统默认的用户。点击set pwd，就可进行新密码的设置，如下图4所示。(需要注意：在发布构件一般需要帐号和密码)

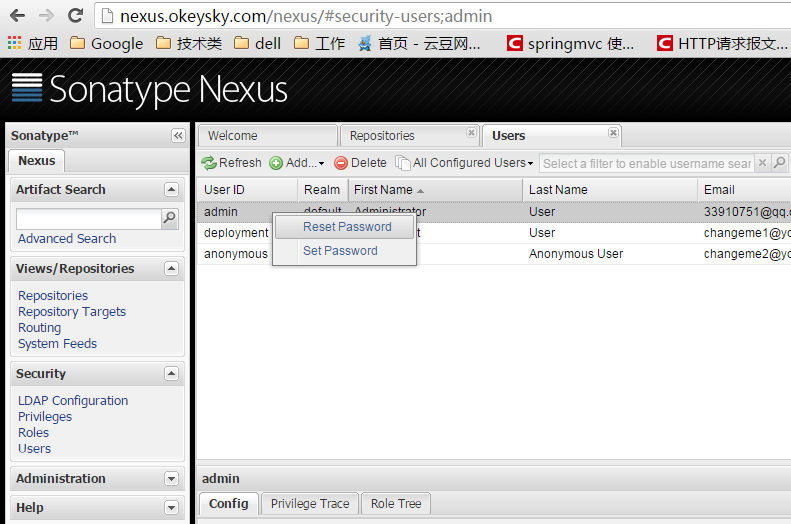


图4

1. 新增用户

点击上方的add按钮，在弹出的界面中，设置必填项：角色，状态为激活。如下图5所示。

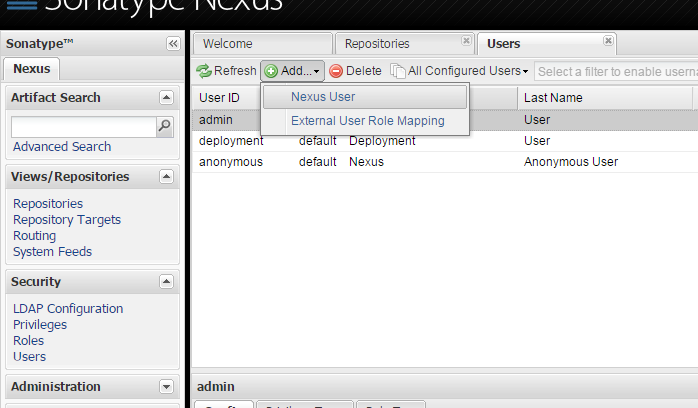


图5

# 3 IDEA 配置maven

## 3.1自动发布构件到Nexus仓库

在工程的pom.xml中添加：distributionManagement

发布仓库（distributionManagement）:负责管理构件的发布包和其他编译生成的支撑文件，那么我们自主开发的jar需要上传到Nexus的时候需要在pom中增加distributionManagement节点进行自动打包上传。

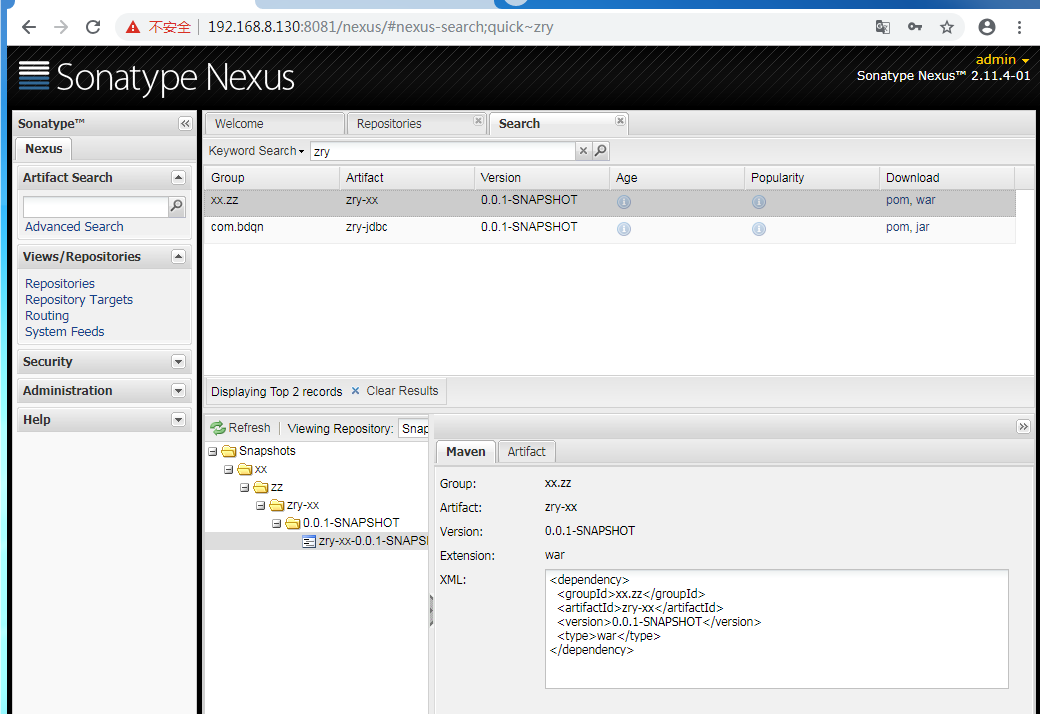
pom 加入 代码 关键配置代码如下：

|  |
| --- |
| <distributionManagement>  <repository>  <id>releases</id><!--ID需要与你的release仓库的Repository ID一致-->  <url>http://192.168.74.163:8081/nexus/content/repositories/releases</url>  </repository>  <snapshotRepository>  <id>snapshots</id><!-- ID需要与你的snapshots仓库的Repository ID一致-->  <url>http://192.168.74.163:8081/nexus/content/repositories/snapshots</url>  </snapshotRepository>  </distributionManagement> |

1. 修改本地maven的settings.xml文件，配置用户名/密码，关键配置代码如下：

|  |
| --- |
| <servers>  <server>  <id>releases</id>  <username>admin</username>  <password>admin123</password>  </server>  <server>  <id>snapshots</id>  <username>admin</username>  <password>admin123</password>  </server>  </servers> |

3 deploy时会自动发布到快照版本库中



<dependency>

<groupId>com.bdqn</groupId>

<artifactId>zry-jdbc</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

当然若你要上传构件到其他仓库，可依照如下代码进行配置：

|  |
| --- |
| <server>    <id>thirdparty</id>    <username>admin</username> <password>admin123</password>  </server>  <server>    <id>central</id>    <username>admin</username>    <password>admin123</password>  </server> |

## 3.2配置Maven从Nexus下载构件

1. 首先配置Maven镜像

配置镜像目的就是让Maven只使用私服，修改Maven的setting.xml文件，具体配置如下。

|  |
| --- |
| <mirror>  <id>nexus</id>  <mirrorOf>\*</mirrorOf>  <name>local nexus repository</name>  <url>http://192.168.74.163:8081/nexus/content/groups/public/</url>  </mirror> |

在pom文件不做任何配置的情况下，默认是使用id为central的Maven中央库进行配置的。那么以上配置只取代了pom中的下载仓库，覆盖了maven默认的配置。mirrorOf配置为\*，表示这个镜像配置适用于所有的仓库，是所有仓库的一个镜像，maven无论要去那个仓库下载构件都会跳转到镜像URL（此时配置的是nexus的group类型的仓库地址，该组仓库包含了其他几个类型的仓库）去下载，不会到中央资源库下载。当然mirrorOf也可以配置一个或多个仓库，多个仓库配置值要与对应的仓库id一致，以逗号分隔。

1. 修改项目的pom文件，设置下载仓库（repostories）

|  |
| --- |
| <repositories>  <repository>  <id>public</id>  <name>Team Maven Repository</name>  <url>http://192.168.74.163:8081/nexus/content/groups/public/</url>  <releases>  <enabled>true</enabled><!—是否支持更新—>  </releases>  <snapshots>  <enabled>true</enabled>  </snapshots>  </repository>  </repositories> |

注：上述配置只对当前项目有效，若需让本机所有Maven项目均使用该Mavne私服，应该在setting.xml中进行配置。

1. 通过在settings.xml中配置profile元素，可以让本机所有的Maven项目都使用Maven私服。

|  |
| --- |
| <profiles>                 <profile>                     <id>public</id>                     <repositories>                         <repository>                           <id>public</id>                           <name>Nexus</name>                           <url>http://192.168.64.128:8081/nexus/content/groups/public/<url>                           <release><enabled>true</enabled></release>                           <snapshots><enabled>true></enabled></snapshots>                         </repository>                      </repositories>  </profile>  <activeProfiles><!-- activeProfiles用来激活—>                <activeProfile>public</activeProfiles>   </activaProfiles> |

<dependency>

<groupId>com.bdqn</groupId>

<artifactId>bdqn-test</artifactId>

<version>0.0.1-SNAPSHOT</version>

<type>war</type>

</dependency>

