# maven私服

正式开发，不同的项目组开发不同的工程。

dao工程开发完毕，发布到私服。

service从私服下载dao

## 需求

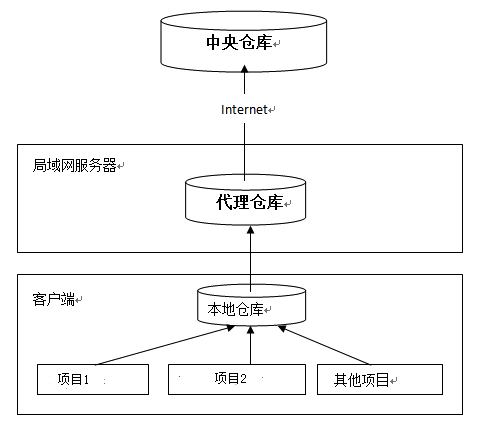
* 问题：项目组编写了一个通用的工具类，其它项目组将类拷贝过去使用，当工具类修改bug后通过邮件发送给各各项目组，这种分发机制不规范可能导致工具类版本不统一。

解决方案：项目组将写的工具类通过maven构建，打成jar，将jar包发布到公司的maven仓库中，公司其它项目通过maven依赖管理从仓库自动下载jar包。

## 分析

公司在自己的局域网内搭建自己的远程仓库服务器，称为私服，私服服务器即是公司内部的maven远程仓库，每个员工的电脑上安装maven软件并且连接私服服务器，员工将自己开发的项目打成jar并发布到私服服务器，其它项目组从私服服务器下载所依赖的构件（jar）。

私服还充当一个代理服务器，当私服上没有jar包会从互联网中央仓库自动下载，如下图：



## 搭建私服环境

### 下载nexus

Nexus 是Maven仓库管理器，通过nexus可以搭建maven仓库，同时nexus还提供强大的仓库管理功能，构件搜索功能等。

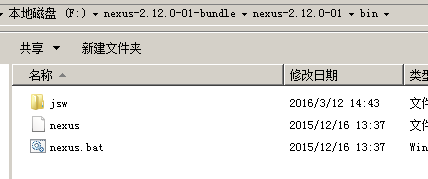
下载Nexus， 下载地址：http://www.sonatype.org/nexus/archived/



下载：nexus-2.12.0-01-bundle.zip

### 安装nexus

解压nexus-2.12.0-01-bundle.zip，本教程将它解压在F盘，进入bin目录：



cmd进入bin目录，执行nexus.bat install



安装成功在服务中查看有nexus服务：



### 卸载nexus

cmd进入nexus的bin目录，执行：nexus.bat uninstall



查看window服务列表nexus已被删除。

### 启动nexus

方法1：

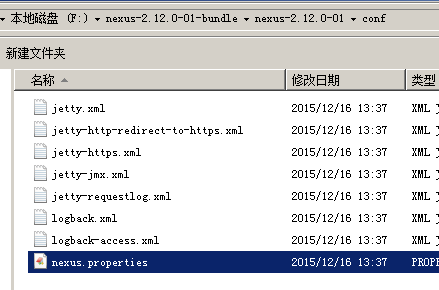
cmd进入bin目录，执行nexus.bat start

方法2：

直接启动nexus服务



查看nexus的配置文件conf/nexus.properties



# Jetty section

application-port=8081 # nexus的访问端口配置

application-host=0.0.0.0 # nexus主机监听配置(不用修改)

nexus-webapp=${bundleBasedir}/nexus # nexus工程目录

nexus-webapp-context-path=/nexus # nexus的web访问路径

# Nexus section

nexus-work=${bundleBasedir}/../sonatype-work/nexus # nexus仓库目录

runtime=${bundleBasedir}/nexus/WEB-INF # nexus运行程序目录

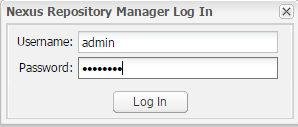
访问：

http://localhost:8081/nexus/

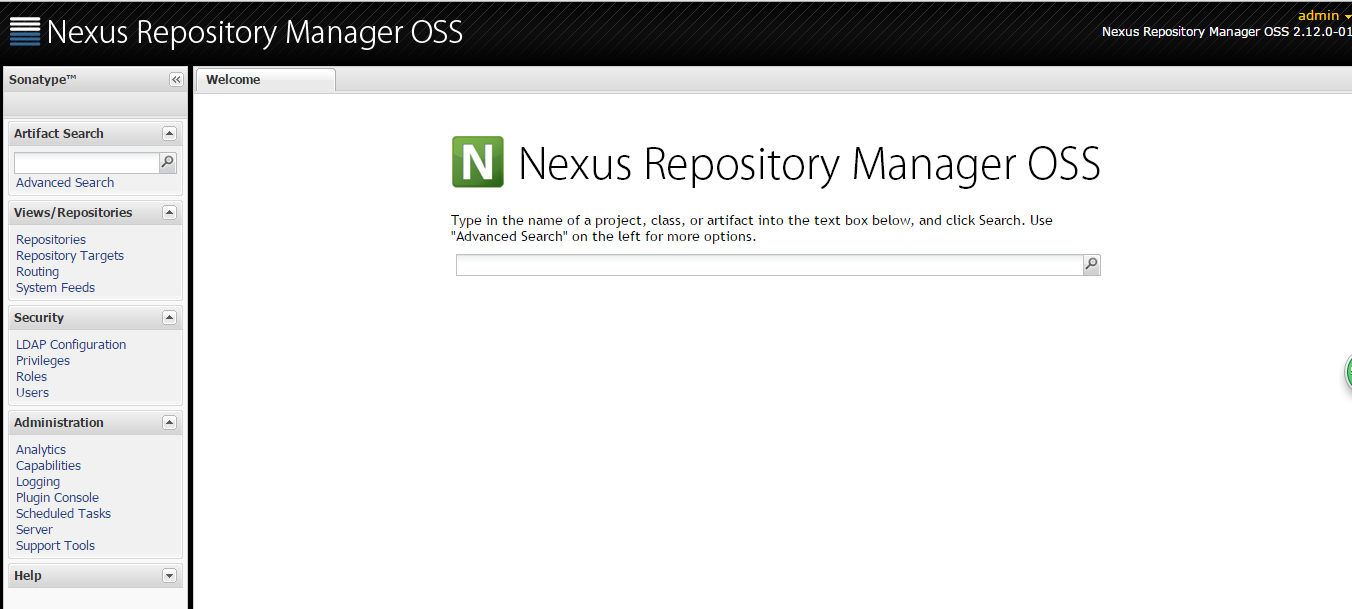


使用Nexus 内置账户admin/admin123登陆：

点击右上角的Log in，输入账号和密码 登陆



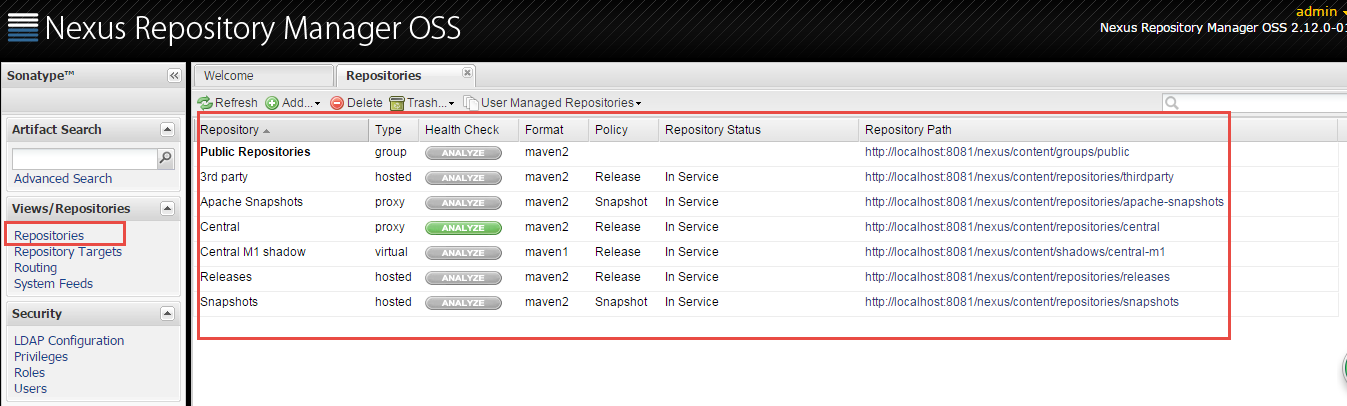
登陆成功：



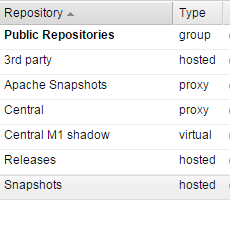
### 仓库类型

nexus

查看nexus的仓库：

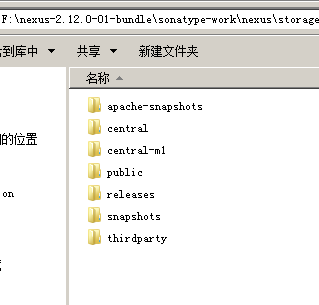


nexus的仓库有4种类型：



1. hosted，宿主仓库，部署自己的jar到这个类型的仓库，包括releases和snapshot两部分，Releases公司内部发布版本仓库、 Snapshots 公司内部测试版本仓库
2. proxy，代理仓库，用于代理远程的公共仓库，如maven中央仓库，用户连接私服，私服自动去中央仓库下载jar包或者插件。
3. group，仓库组，用来合并多个hosted/proxy仓库，通常我们配置自己的maven连接仓库组。
4. virtual(虚拟)：兼容Maven1 版本的jar或者插件

nexus仓库默认在sonatype-work目录中：



* **central：代理仓库，代理中央仓库**



* **apache-snapshots：代理仓库**

存储snapshots构件，代理地址https://repository.apache.org/snapshots/

* **central-m1：**virtual类型仓库，兼容Maven1 版本的jar或者插件
* **releases：本地仓库，存储releases构件。**
* **snapshots：本地仓库，存储snapshots构件。**
* **thirdparty：第三方仓库**
* **public：仓库组**

## 将项目发布到私服

### 需求

企业中多个团队协作开发通常会将一些公用的组件、开发模块等发布到私服供其它团队或模块开发人员使用。

本例子假设多团队分别开发dao、service、web，某个团队开发完在dao会将dao发布到私服供service团队使用，本例子会将dao工程打成jar包发布到私服。

私服(服务端)

nexus

dao工程

(客户端)

service工程

（客户端）

上传

上传

### 配置

第一步： 需要在客户端即部署dao工程的电脑上配置 maven环境，并修改 settings.xml 文件，配置连接私服的用户和密码 。

此用户名和密码用于私服校验，因为私服需要知道上传都 的账号和密码 是否和私服中的账号和密码 一致。

<server>

<id>releases</id>

<username>admin</username>

<password>admin123</password>

</server>

<server>

<id>snapshots</id>

<username>admin</username>

<password>admin123</password>

</server>

releases 连接发布版本项目仓库

snapshots 连接测试版本项目仓库



第二步： 配置项目pom.xml

配置私服仓库的地址，本公司的自己的jar包会上传到私服的宿主仓库，根据工程的版本号决定上传到哪个宿主仓库，如果版本为release则上传到私服的release仓库，如果版本为snapshot则上传到私服的snapshot仓库

<distributionManagement>

<repository>

<id>releases</id>

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

</repository>

<snapshotRepository>

<id>snapshots</id>

<url>http://localhost:8081/nexus/content/repositories/snapshots/</url>

</snapshotRepository>

</distributionManagement>

注意：pom.xml这里<id> 和 settings.xml 配置 <id> 对应！

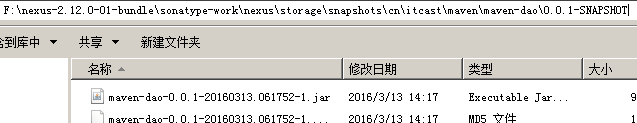
### 测试

将项目dao工程打成jar包发布到私服：

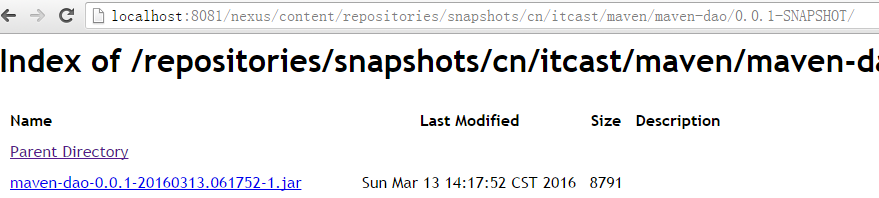
1、首先启动nexus

2、对dao工程执行deploy命令

根据本项目pom.xml中version定义决定发布到哪个仓库，如果version定义为snapshot，执行deploy后查看nexus的snapshot仓库，如果version定义为release则项目将发布到nexus的release仓库，本项目将发布到snapshot仓库：



也可以通过http方式查看：



## 从私服下载jar包

### 需求

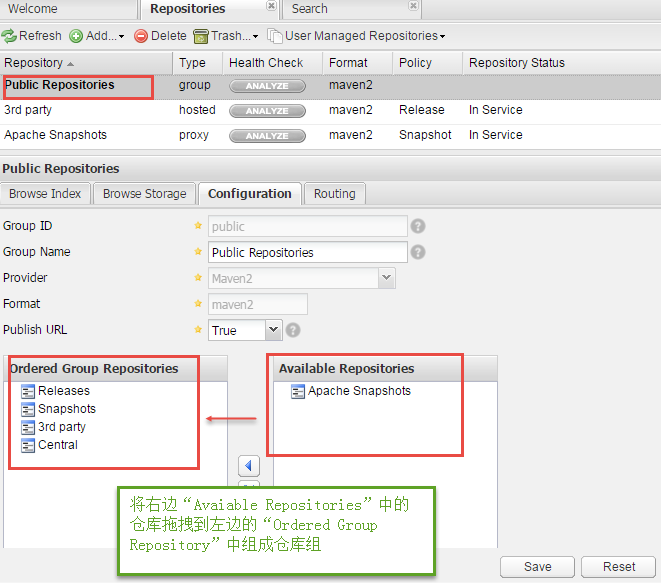
没有配置nexus之前，如果本地仓库没有，去中央仓库下载，通常在企业中会在局域网内部署一台私服服务器，有了私服本地项目首先去本地仓库找jar，如果没有找到则连接私服从私服下载jar包，如果私服没有jar包私服同时作为代理服务器从中央仓库下载jar包，这样做的好处是一方面由私服对公司项目的依赖jar包统一管理，一方面提高下载速度，项目连接私服下载jar包的速度要比项目连接中央仓库的速度快的多。

本例子测试从私服下载dao 工程jar包。

### 管理仓库组

nexus中包括很多仓库，hosted中存放的是企业自己发布的jar包及第三方公司的jar包，proxy中存放的是中央仓库的jar，为了方便从私服下载jar包可以将多个仓库组成一个仓库组，每个工程需要连接私服的仓库组下载jar包。

打开nexus配置仓库组，如下图：



上图中仓库组包括了本地仓库、代理仓库等。

### 在setting.xml中配置仓库

在客户端的setting.xml中配置私服的仓库，由于setting.xml中没有repositories的配置标签需要使用profile定义仓库。

<profile>

<!--profile的id-->

<id>dev</id>

<repositories>

<repository>

<!--仓库id，repositories可以配置多个仓库，保证id不重复-->

<id>nexus</id>

<!--仓库地址，即nexus仓库组的地址-->

<url>http://localhost:8081/nexus/content/groups/public/</url>

<!--是否下载releases构件-->

<releases>

<enabled>true</enabled>

</releases>

<!--是否下载snapshots构件-->

<snapshots>

<enabled>true</enabled>

</snapshots>

</repository>

</repositories>

<pluginRepositories>

<!-- 插件仓库，maven的运行依赖插件，也需要从私服下载插件 -->

<pluginRepository>

<!-- 插件仓库的id不允许重复，如果重复后边配置会覆盖前边 -->

<id>public</id>

<name>Public Repositories</name>

<url>http://localhost:8081/nexus/content/groups/public/</url>

</pluginRepository>

</pluginRepositories>

</profile>

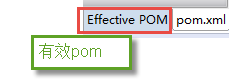
使用profile定义仓库需要激活才可生效。

<activeProfiles>

<activeProfile>dev</activeProfile>

</activeProfiles>

配置成功后通过eclipse查看有效pom，有效pom是maven软件最终使用的pom内容，程序员不直接编辑有效pom，打开有效pom



有效pom内容如下：

下边的pom内容中有两个仓库地址，maven会先从前边的仓库的找，如果找不到jar包再从下边的找，从而就实现了从私服下载jar包。

<repositories>

<repository>

<releases>

<enabled>true</enabled>

</releases>

<snapshots>

<enabled>true</enabled>

</snapshots>

<id>public</id>

<name>Public Repositories</name>

<url>http://localhost:8081/nexus/content/groups/public/</url>

</repository>

<repository>

<snapshots>

<enabled>false</enabled>

</snapshots>

<id>central</id>

<name>Central Repository</name>

<url>https://repo.maven.apache.org/maven2</url>

</repository>

</repositories>

<pluginRepositories>

<pluginRepository>

<id>public</id>

<name>Public Repositories</name>

<url>http://localhost:8081/nexus/content/groups/public/</url>

</pluginRepository>

<pluginRepository>

<releases>

<updatePolicy>never</updatePolicy>

</releases>

<snapshots>

<enabled>false</enabled>

</snapshots>

<id>central</id>

<name>Central Repository</name>

<url>https://repo.maven.apache.org/maven2</url>

</pluginRepository>

</pluginRepositories>

### 测试从私服下载jar包

测试1：局域网环境或本地网络即可

在service工程中添加以上配置后，添加dao工程的依赖，删除本地仓库中dao工程，同时在eclipse中关闭dao工程。

观察控制台：

Downloading http://localhost:8081/nexus/content/groups/public/cn/itcast/ssh/ssh-dao0423/0.0.1-SNAPSHOT/ssh-dao0423-0.0.1-20160702.123702-1.jar

项目先从本地仓库找dao，找不到从私服找，由于之前执行deploy将dao部署到私服中，所以成功从私服下载dao并在本地仓库保存一份。

如果此时删除私服中的dao，执行update project之后是否正常？

如果将本地仓库的dao和私服的dao全部删除是否正常？

测试2：需要互联网环境

在项目的pom.xml添加一个依赖，此依赖在本地仓库和私服都不存在，maven会先从本地仓库找，本地仓库没有再从私服找，私服没有再去中央仓库下载，jar包下载成功在私服、本地仓库分别存储一份。