## 待解决的问题

项目中同一开发小组的同事在本地maven中手动引入了工程使用的jar包，直接将pom文件提交的svn，其它人员无法下载导致项目报错，并且在进行自动化构建时也会报jar找不到的错误。

## 解决方案

* 项目组成员手动在本地安装，运维人员在服务器进行maven安装，很显然这种方案相当麻烦，同时更换电脑等需要重新配置开发环境时就会丢失原有一些手动安装的jar包，再次使用时，需要重新安装带来重复性无用操作。
* 创建maven私服，该方案可一劳永逸的解决问题

## Nexus私服的搭建与maven整合

nexus能带来以下几点好处：

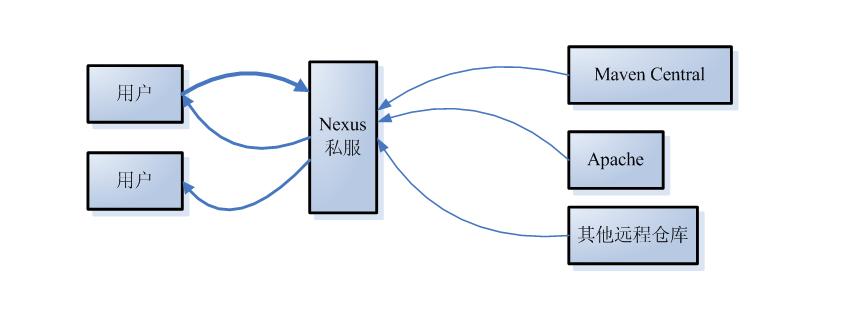
首先在服务器上安装jar包后同项目组成员只需要集中配置maven下载仓库即可；

其次节省公司带宽，私服能够缓存jar，首先在服务器查找文件，找不到才会到远程仓库下载。

最后nexus提供了管理页面，提供管理、检索jar的功能，当需要一些maven中心仓库不存在的jar时，应该首先到nexus页面搜索一下，可能就会包含你需要的jar包。

私服的原理：

使用一张网上比较经典的原理图：



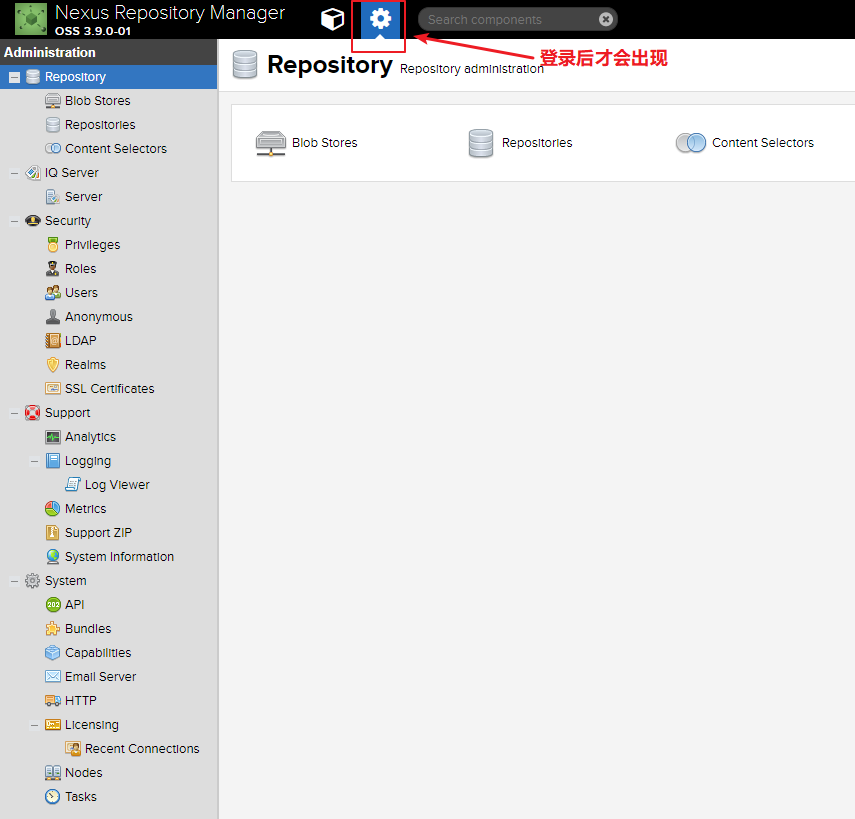
Nexus的安装：

官网找到对应的压缩包。

Linux ：解压tar -zvxf nexus-2.12.0-01-bundle.tar.gz 启动 ./bin/nexus start

Window: nexus/bin/jws 根据系统版本找到对应脚本执行发布window服务

由于nexus自带了jetty容器，默认的端口是8081，若想修改Nexus的服务端口，可以在nexus-3.9.0-01-win64\sonatype-work\nexus3\etc\ nexus.properties配置中修改 application-port。启动后访问localhost:8081/nexus即可进入管理页，默认的用户名是 admin 密码是 admin123，登录后即可出现设置选项，如图：



Maven的一些概念：

1）maven-central：maven中央库，该仓库代理Maven中央仓库，默认从https://repo1.maven.org/maven2/拉取jar ，其策略为Release，只会下载和缓存中央仓库中的发布版本构件。

2）maven-releases：私库发行版jar（一般为稳定版本）

3）maven-snapshots：私库快照（调试版本，可能会频繁的变动）jar

Nexus的一些概念：

1）hosted(宿主类型)：内部项目的发布仓库（内部项目，以及一些外部安装的jar包对应的仓库类型）；

2）proxy(代理类型)：从远程中央仓库中寻找数据的仓库

3）group(仓库组类型)：又叫组仓库，开发人员自己设定的仓库组，可由以上两种类型的仓库组成；

首次安装使用时可通过配置远程索引预先下载远程仓库中的数据到本地私服，避免在项目使用使用时才下载。

与maven 的整合：

配置内部开发人员本地与服务器的maven配置文件setting.xml添加一下内容：

<servers>

     <server>

          <id>nexus</id>

         <username>admin</username>

          <password>admin123</password>

      </server>

</servers>

<mirrors>

     <mirror>

        <!--This sends everything else to /public -->

        <id>nexus</id>

        <!-- \*代表所有会拦截所有远程仓库地址，替换为本地私服地址 -->

        <mirrorOf>\*</mirrorOf>

        <!--私服路径 maven-public一般为我们创建的仓库组 -->

        <url>http://localhost:8081/repository/maven-public/</url>

     </mirror>

</mirrors>

<profiles>

     <profile>

         <id>nexus</id>

         <!--Enable snapshots for the built in central repo to direct -->

         <!--all requests to nexus via the mirror -->

         <repositories>

             <repository>

                <id>central</id>

                <url>http://central</url>

                <releases>

                    <enabled>true</enabled>

                </releases>

                <snapshots>

                    <enabled>true</enabled>

                </snapshots>

            </repository>

        </repositories>

        <pluginRepositories>

            <pluginRepository>

                <id>central</id>

                <url>http://central</url>

                <releases>

                    <enabled>true</enabled>

                </releases>

                <snapshots>

                    <enabled>true</enabled>

                </snapshots>

            </pluginRepository>

        </pluginRepositories>

     </profile>

</profiles>

<activeProfiles>

     <!--make the profile active all the time 此处激活对应的profile配置 -->

     <activeProfile>nexus</activeProfile>

</activeProfiles>

</settings>

需要注意的是：

在maven配置完毕后，需要配置项目pom文件添加仓库

<repositories>

<repository>

<id>maven-public</id>

<name>maven-central</name>

<url>http://172.16.100.99:8081/repository/maven-public/</url>

<snapshots>

<enabled>true</enabled>

</snapshots>

<releases>

<enabled>true</enabled>

</releases>

</repository>

</repositories>

否则手动添加的jar包无法下载，项目报错

若想要想私服发布项目，还需要添加

<distributionManagement>

<repository>

<id>nexus1</id>

<name>Releases</name>

<!-- 从repository/maven-releases管理页面复制url地址 --> <url>http://172.16.100.99:8081/repository/maven-releases/</url>

</repository>

<snapshotRepository>

<id>nexus1</id>

<name>Snapshot</name>

<url>http://172.16.100.99:8081/repository/maven-snapshots/</url>

</snapshotRepository>

</distributionManagement>

私服手动添加jar：

*通过maven命令进行添加：mvn deploy:deploy-file -DgroupId=com.oracle -DartifactId=ojdbc6 -Dversion=11.2.0.3 -Dpackaging=jar -Dfile=D:\ojdbc6-11.2.0.3.jar -Durl=http://localhost:8081/repository/maven-releases/ -DrepositoryId=nexus*

*其中-DgroupId 为上传的jar的groupId*

*-DartifactId 为上传的jar的artifactId*

*-Dversion 为上传的jar的需要被依赖的时候的版本号*

*然后是-Dpackaging为jar，-Dfile为jar包路径*

*-Durl 为要上传的路径*

*-DrepositoryId 为repository的唯一标示，跟第二步中赋权配置的server相同*

*第二种方式：使用Nexus Repository Manager图形化界面上传*

