springboot(十六)：使用Jenkins部署Spring Boot

 2017/11/11

jenkins是devops神器，本篇文章介绍如何安装和使用jenkins部署Spring Boot项目

jenkins搭建 部署分为三个步骤；

* 第一步，jenkins安装
* 第二步，插件安装和配置
* 第三步，Push SSH
* 第四步，部署项目

**第一步 ，jenkins安装**

准备环境：

JDK:1.8  
Jenkins:2.83 Centos:7.3  
maven 3.5‘

jdk默认已经安装完成

**配置maven**

版本要求maven3.5.0

软件下载

wget http://mirror.bit.edu.cn/apache/maven/maven-3/3.5.0/binaries/apache-maven-3.5.0-bin.tar.gz

安装

*## 解压*

tar vxf apache-maven-3.5.0-bin.tar.gz

*## 移动*

mv apache-maven-3.5.0 /usr/local/maven3

修改环境变量， 在/etc/profile中添加以下几行

MAVEN\_HOME**=**/usr/local/maven3

export MAVEN\_HOME

export PATH**=${**PATH**}**:**${**MAVEN\_HOME**}**/bin

记得执行source /etc/profile使环境变量生效。

验证 最后运行mvn -v验证maven是否安装成功

**配置防护墙**

关闭防护墙

*#centos7*

systemctl stop firewalld.service

**==============================**

*#以下为：centOS 6.5关闭防火墙步骤*

*#关闭命令：*

service iptables stop

*#永久关闭防火墙：*

chkconfig iptables off

两个命令同时运行，运行完成后查看防火墙关闭状态

service iptables status

**jenkins 安装**

下载

cd /opt

wget http://mirrors.jenkins.io/war/2.83/jenkins.war

启动服务

java -jar jenkins.war &

Jenkins 就启动成功了！它的war包自带Jetty服务器

第一次启动Jenkins时，出于安全考虑，Jenkins会自动生成一个随机的按照口令。**注意控制台输出的口令，复制下来**，然后在浏览器输入密码：

INFO:

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

Jenkins initial setup is required. An admin user has been created and a password generated.

Please use the following password to proceed to installation:

0cca37389e6540c08ce6e4c96f46da0f

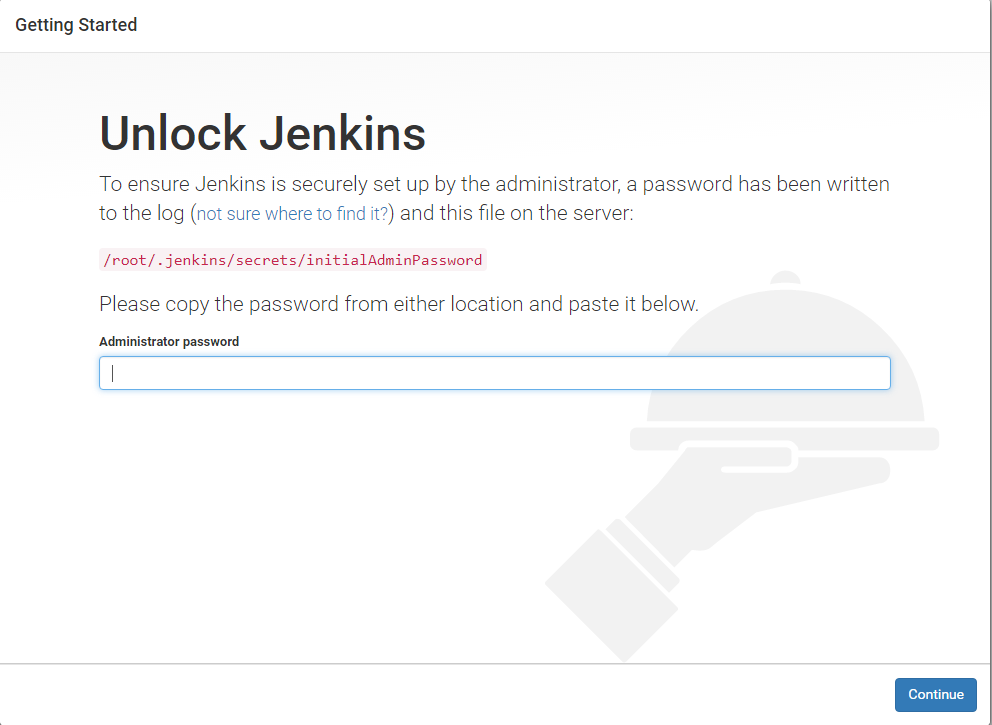
This may also be found at: /root/.jenkins/secrets/initialAdminPassword

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

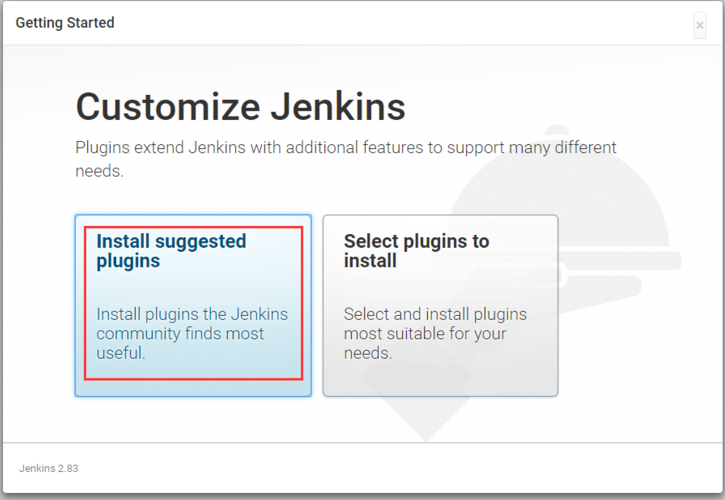
**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

访问 浏览器访问：http://localhost:8080/

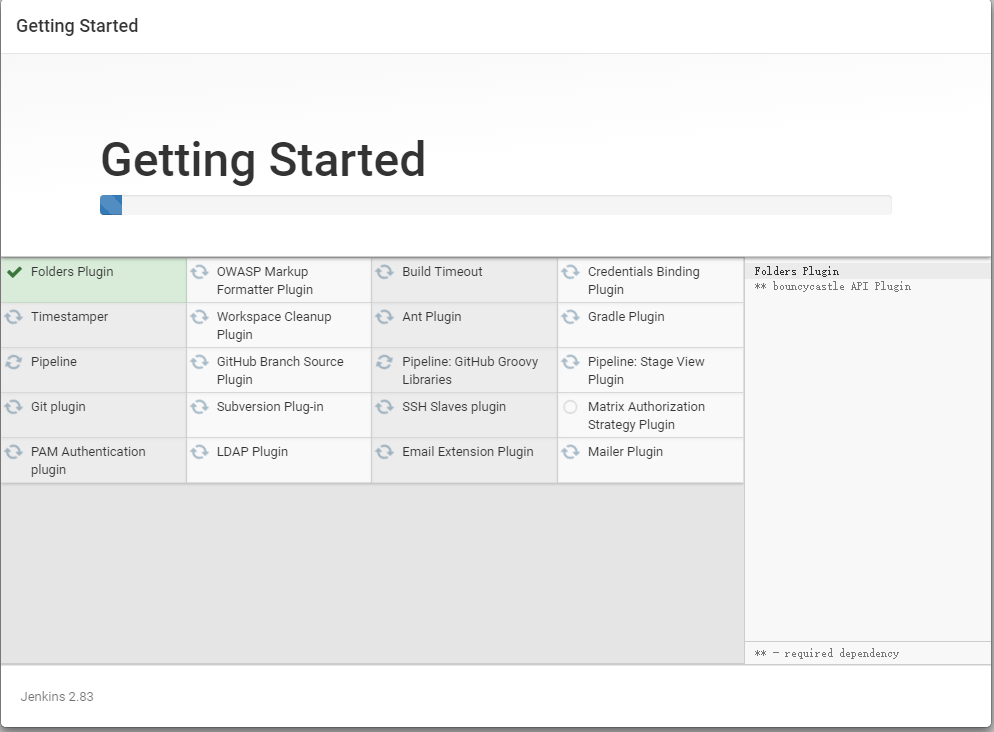


输入：0cca37389e6540c08ce6e4c96f46da0f

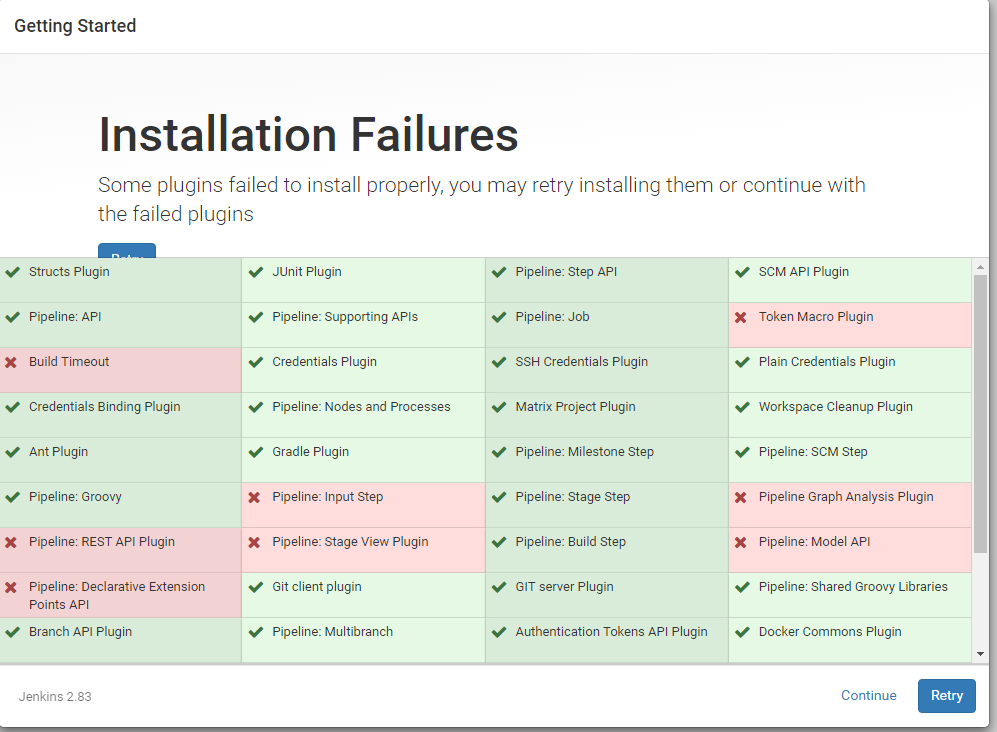
进入用户自定义插件界面，建议选择安装官方推荐插件，因为安装后自己也得安装:



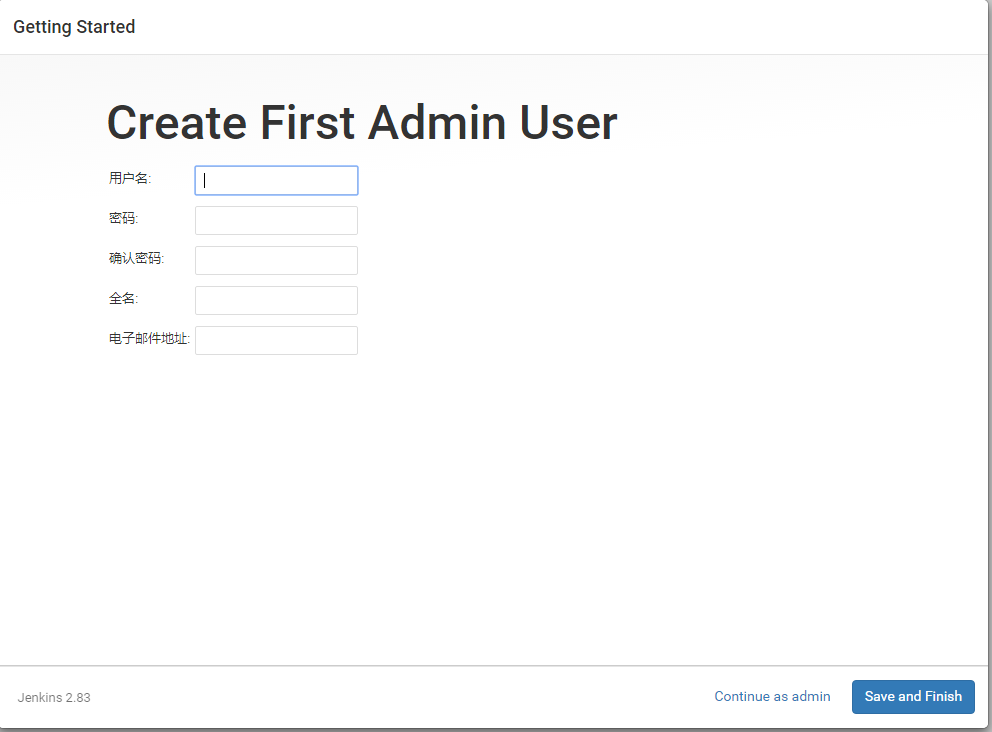
接下来是进入插件安装进度界面:



插件一次可能不会完全安装成功，可以点击Retry再次安装。直到全部安装成功

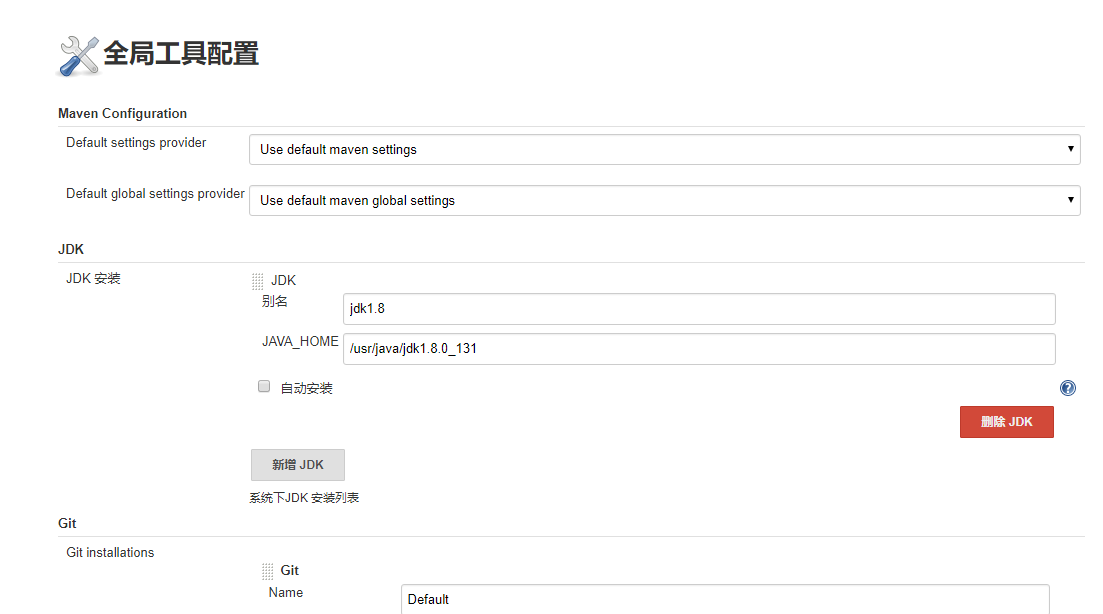


等待一段时间之后，插件安装完成，配置用户名密码:



输入：admin/admin

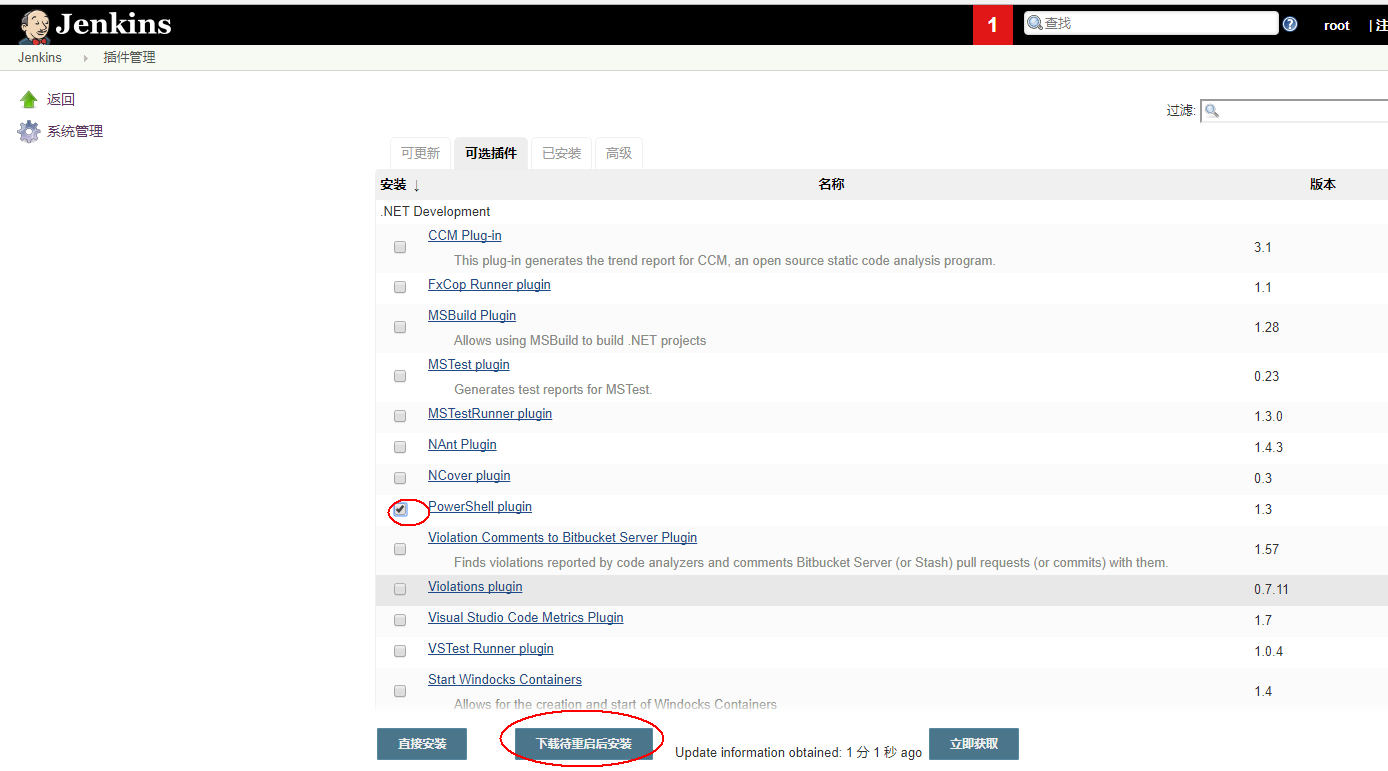
系统管理-》全局工具配置 jdk路径，



**第二步，插件安装和配置**

有很多插件都是选择的默认的安装的，所以现在需要我们安装的插件不多，Git plugin和Maven Integration plugin，publish over SSH。

插件安装：系统管理 > 插件管理 > 可选插件,勾选需要安装的插件，点击直接安装或者下载重启后安装

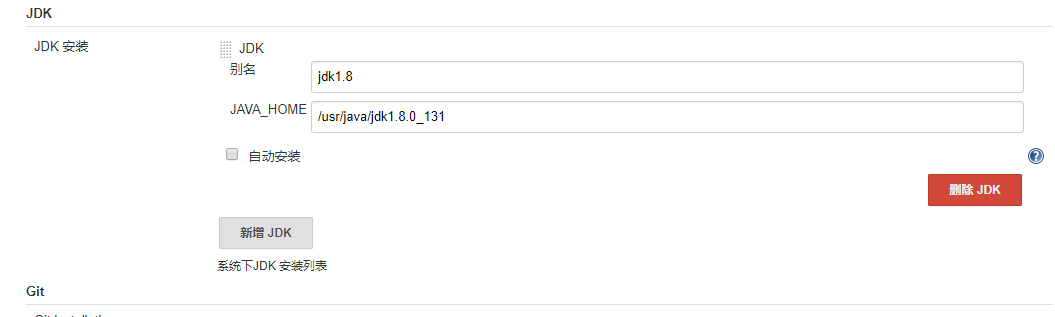


**配置全局变量**

系统管理 > 全局工具配置

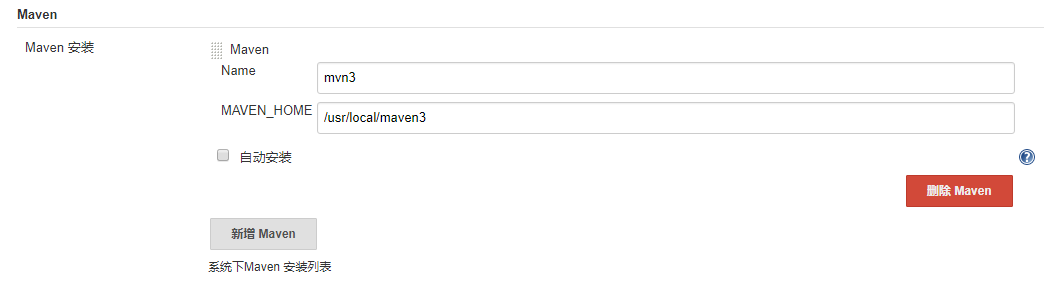
**JDK**

配置本地JDK的路径，去掉勾选自动安装



**Maven**

配置本地maven的路径，去掉勾选自动安装



其它内容可以根据自己的情况选择安装。

**配置 SSH免登陆**

ssh的配置可使用密钥，也可以使用密码，这里我们使用密钥来配置，在配置之前先配置好jenkins服务器和应用服务器的密钥认证 **jenkins服务器**上生成密钥对，使用ssh-keygen -t rsa命令

输入下面命令 一直回车，一个矩形图形出现就说明成功，在~/.ssh/下会有私钥id\_rsa和公钥id\_rsa.pub

ssh-keygen -t rsa

将**jenkins服务器**的公钥id\_rsa.pub中的内容复制到**应用服务器** 的~/.ssh/下的 authorized\_keys文件

ssh-copy-id -i id\_rsa.pub 192.168.0.xx

chmod 644 authorized\_keys

在**应用服务器**上重启ssh服务，service sshd restart现在jenkins服务器可免密码直接登陆应用服务器

之后在用ssh B尝试能否免密登录B服务器，如果还是提示需要输入密码，则有以下原因

* a. 非root账户可能不支持ssh公钥认证（看服务器是否有限制）
* b. 传过来的公钥文件权限不够，可以给这个文件授权下 chmod 644 authorized\_keys
* c. 使用root账户执行ssh-copy-id -i ~/.ssh/id\_rsa.pub 这个指令的时候如果需要输入密码则要配置sshd\_config

vi /etc/ssh/sshd\_config

*#内容*

PermitRootLogin no

修改完后要重启sshd服务

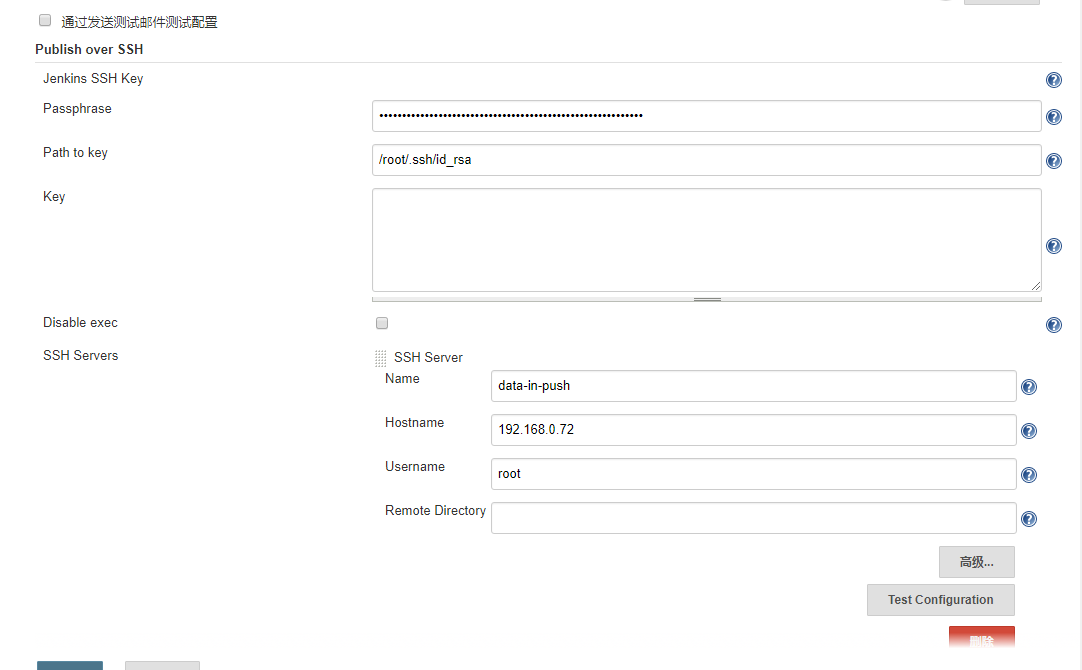
service sshd restart

最后，如果可以SSH IP 免密登录成功说明SSH公钥认证成功。

**第三步，Push SSH**

系统管理 > 系统设置

选择 Publish over SSH



Passphrase 不用设置 Path to key 写上生成的ssh路径：/root/.ssh/id\_rsa

下面的SSH Servers是重点

Name 随意起名代表这个服务，待会要根据它来选择 Hostname 配置应用服务器的地址 Username 配置linux登陆用户名 Remote Directory 不填

点击下方增加可以添加多个应用服务器的地址

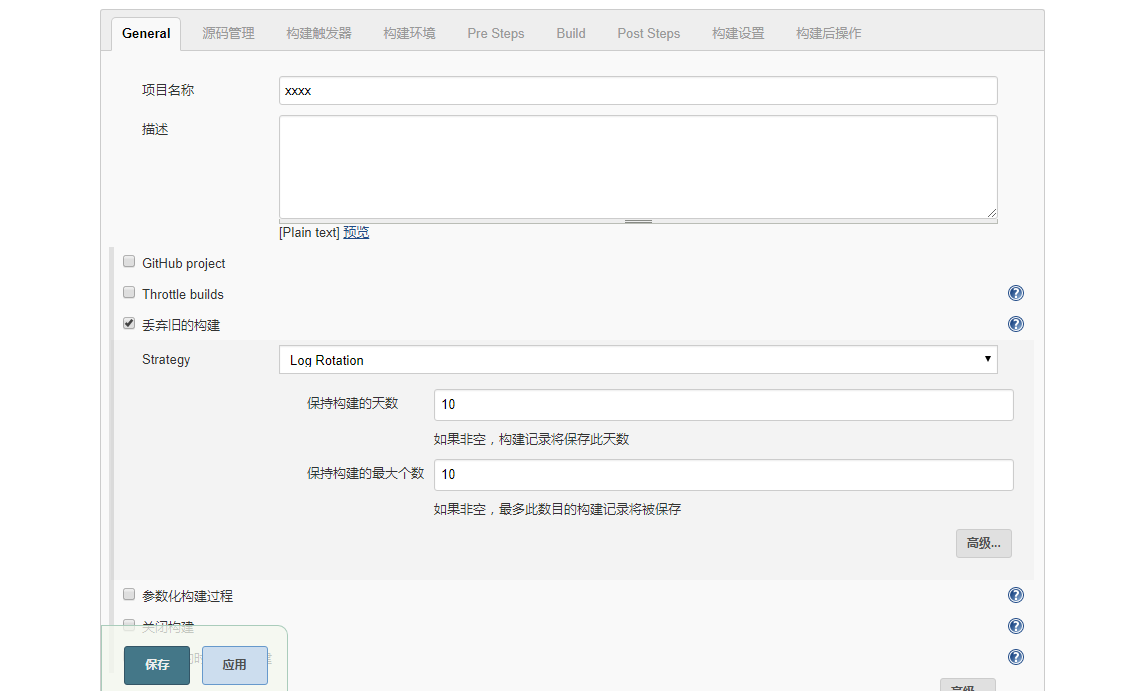
**第四步，部署项目**

首页点击**新建**：输入项目名称

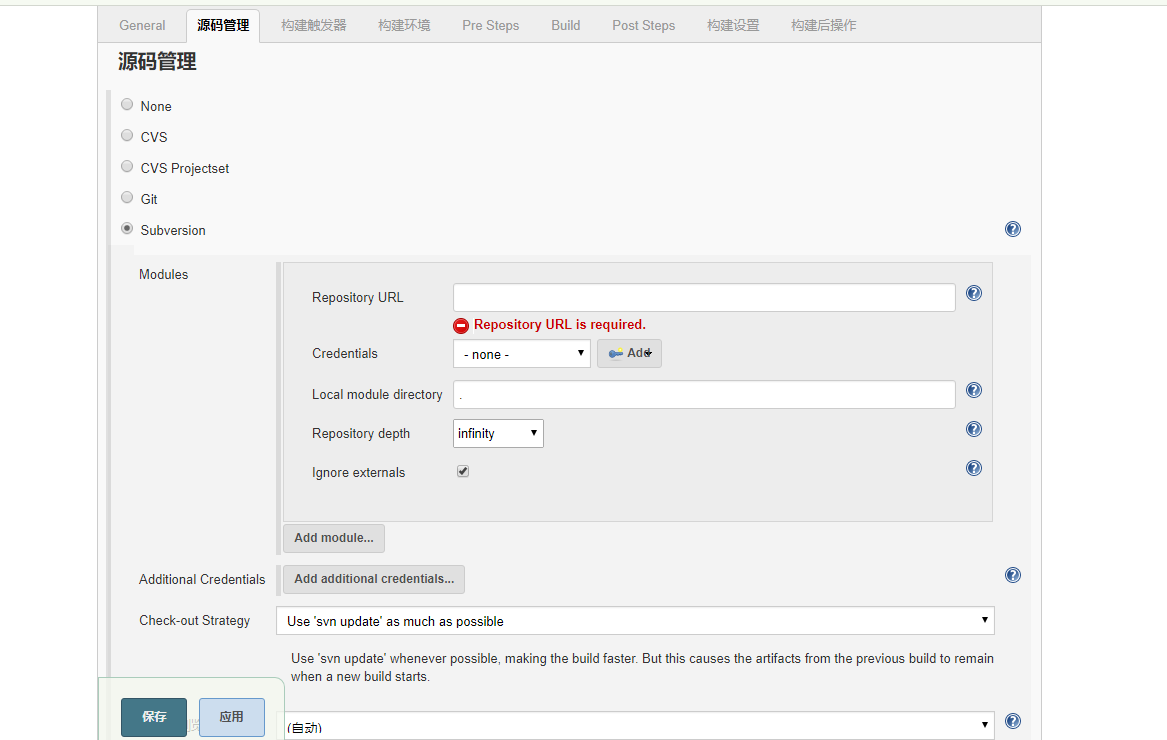


下方选择构建一个maven项目，点击确定。

勾选**丢弃旧的构建**，选择是否备份被替换的旧包。我这里选择备份最近的10个

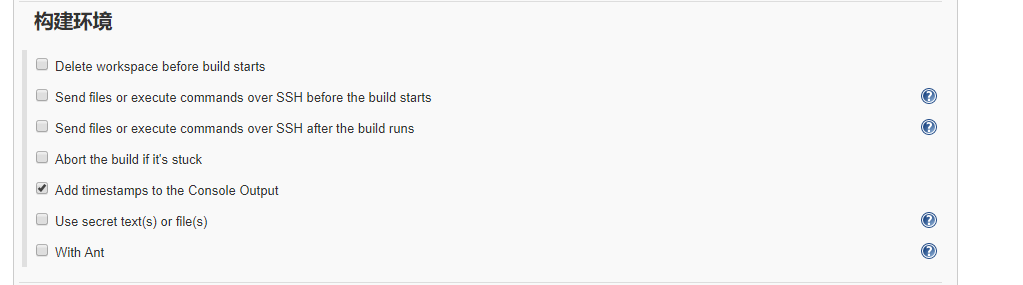


源码管理,选择svn,配置SVN相关信息，点击add可以输入svn的账户和密码



svn地址：http://192.168.0.xx/svn/xxx@HEAD,@HEAD意思取最新版本

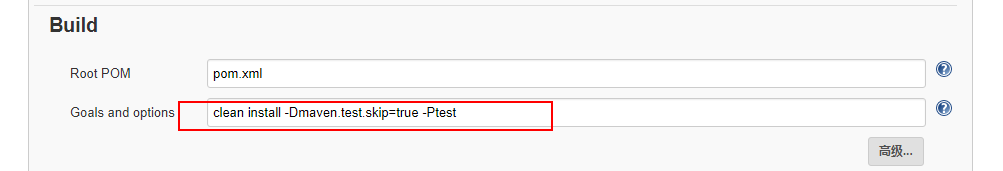
构建环境中勾选“Add timestamps to the Console Output”，代码构建的过程中会将日志打印出来



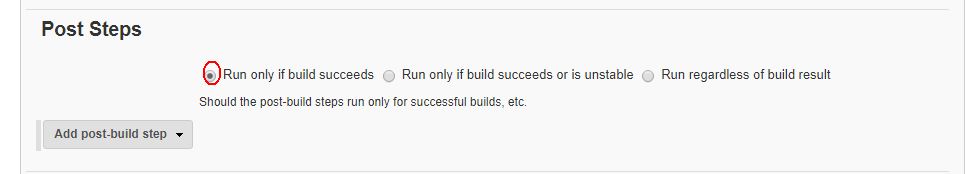
在Build中输入打包前的mvn命令，如：

clean install -Dmaven.test.skip**=**true -Ptest

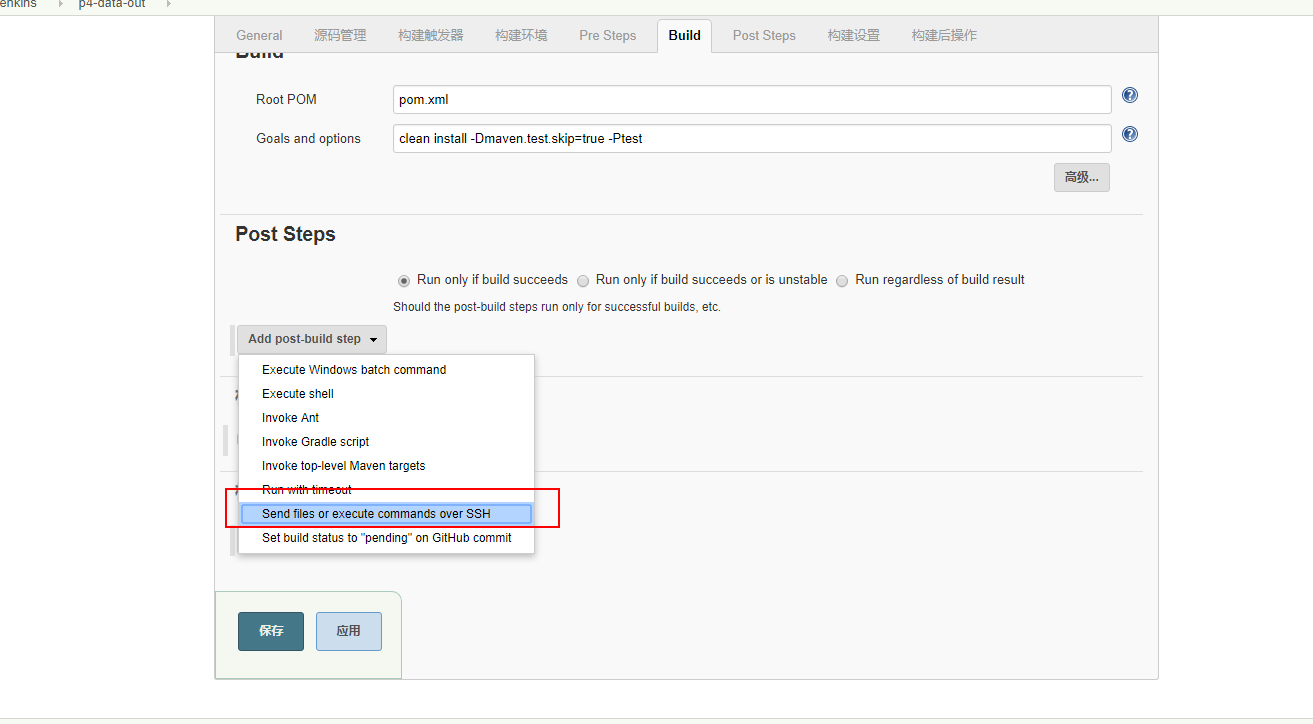
意思是：排除测试的包内容，使用后缀为test的配置文件。



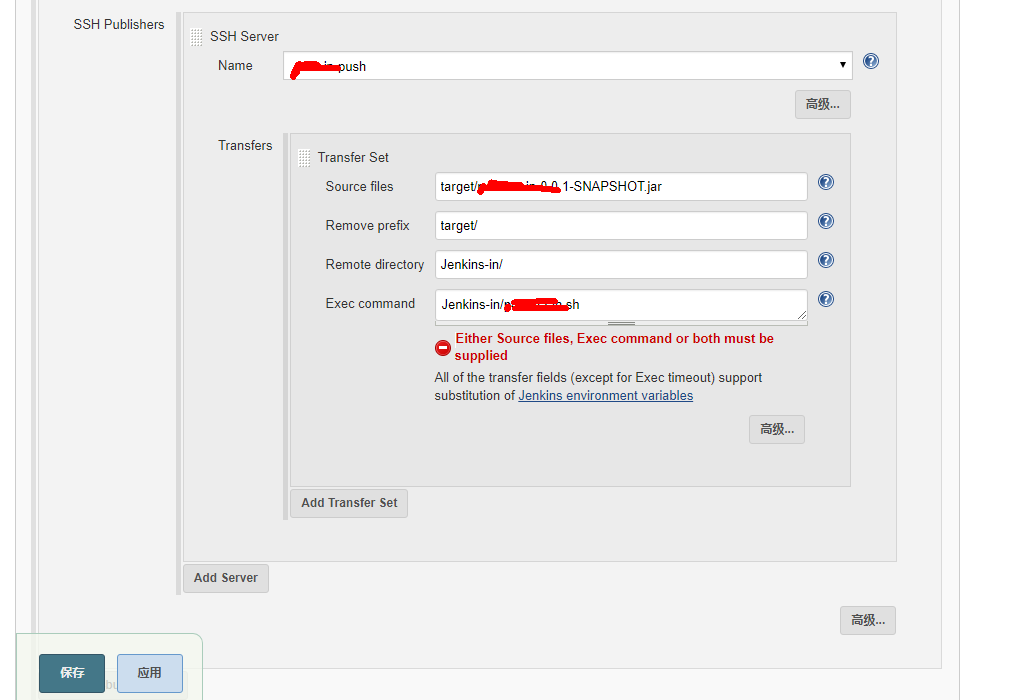
Post Steps 选择 Run only if build succeeds



点击**Add post-build step**，选择 Send files or execute commands over SSH



Name选择上面配置的Push SSH



Source files配置:target/xxx-0.0.1-SNAPSHOT.jar 项目jar包名 Remove prefix:target/ Remote directory:Jenkins-in/ 代码应用服务器的目录地址， Exec command：Jenkins-in/xxx.sh 应用服务器对应的脚本。

需要在应用服务器创建文件夹：Jenkins-in，在文件夹中复制一下脚本内容：xxx.sh

DATE**=$(**date +%Y%m%d**)**

export JAVA\_HOME PATH CLASSPATH

JAVA\_HOME**=**/usr/java/jdk1.8.0\_131

PATH**=**$JAVA\_HOME/bin:$JAVA\_HOME/jre/bin:$PATH

CLASSPATH**=**.:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib:$CLASSPATH

DIR**=**/root/xxx

JARFILE**=**xxx-0.0.1-SNAPSHOT.jar

**if** **[** **!** -d $DIR/backup **]**;**then**

mkdir -p $DIR/backup

**fi**

cd $DIR

ps -ef | grep $JARFILE | grep -v grep | awk '{print $2}' | xargs kill -9

mv $JARFILE backup/$JARFILE$DATE

mv -f /root/Jenkins-in/$JARFILE .

java -jar $JARFILE **>** out.log &

**if** **[** $? **=** 0 **]**;**then**

sleep 30

tail -n 50 out.log

**fi**

cd backup/

ls -lt|awk 'NR>5{print $NF}'|xargs rm -rf

这段脚本的意思，就是kill旧项目，删除旧项目，启动新项目，备份老项目。

全文完。