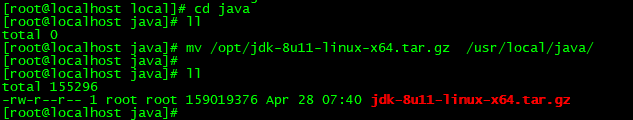
# 安装jdk

### 利用命令mkdir -p /usr/local/java创建文件夹，然后移动文件到该文件夹里。

移动jdk到此目录下



### 解压并打开profile文件

位置:/etc/profile

在文件的最后面添加下面配置：

JAVA\_HOME=/usr/local/java/jdk1.8.0\_11

JRE\_HOME=/usr/local/java/jdk1.8.0\_11/jre

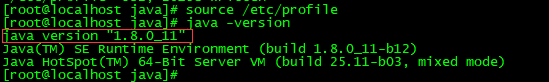
CLASS\_PATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar:$JRE\_HOME/lib

PATH=$PATH:$JAVA\_HOME/bin:$JRE\_HOME/bin

export JAVA\_HOME JRE\_HOME CLASS\_PATH PATH

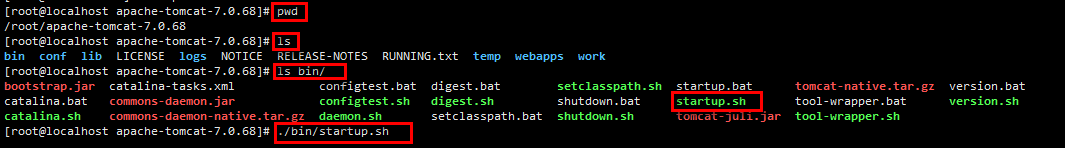
### 重新加载文件:

# source /etc/profile



# 安装tomcat

上传、解压运行





### Tomcat8及以上配置Manager访问权限

1.修改$TOMCAT\_HOME/webapps/manager/META-INF/context.xml文件，允许哪些IP可以访问Manager服务

<?xml version="1.0" encoding="UTF-8"?>

<!--

Licensed to the Apache Software Foundation (ASF) under one or more

contributor license agreements. See the NOTICE file distributed with

this work for additional information regarding copyright ownership.

The ASF licenses this file to You under the Apache License, Version 2.0

(the "License"); you may not use this file except in compliance with

the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<Context antiResourceLocking="false" privileged="true" >

<Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="^.\*$" />

</Context>

2.修改$TOMCAT\_HOME/conf/tomcat-users.xml文件，配置用户名及密码

<?xml version="1.0" encoding="UTF-8"?>

<!--

Licensed to the Apache Software Foundation (ASF) under one or more

contributor license agreements. See the NOTICE file distributed with

this work for additional information regarding copyright ownership.

The ASF licenses this file to You under the Apache License, Version 2.0

(the "License"); you may not use this file except in compliance with

the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<tomcat-users xmlns="http://tomcat.apache.org/xml"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"

version="1.0">

<!--

NOTE: By default, no user is included in the "manager-gui" role required

to operate the "/manager/html" web application. If you wish to use this app,

you must define such a user - the username and password are arbitrary. It is

strongly recommended that you do NOT use one of the users in the commented out

section below since they are intended for use with the examples web

application.

-->

<!--

NOTE: The sample user and role entries below are intended for use with the

examples web application. They are wrapped in a comment and thus are ignored

when reading this file. If you wish to configure these users for use with the

examples web application, do not forget to remove the <!.. ..> that surrounds

them. You will also need to set the passwords to something appropriate.

-->

<!--

<role rolename="tomcat"/>

<role rolename="role1"/>

<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>

<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>

<user username="role1" password="<must-be-changed>" roles="role1"/>

-->

<role rolename="manager-gui"/>

<role rolename="manager-status"/>

<role rolename="manager-script"/>

<user username="admin" password="123456" roles="manager-gui, mananger-status"/>

</tomcat-users>

# 安装maven

1:前提：先安装jdk

2:获取**maven**安装包  
wget [http://apache.fayea.com/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz](https://link.jianshu.com?t=http:/apache.fayea.com/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz)

3:解压 tar -zxvf [apache-maven-3.3.9-bin.tar.gz](https://link.jianshu.com?t=http:/apache.fayea.com/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz)

4:移动maven到部署目录  
mv apache-maven-3.3.9 /usr/local/maven

5:配置环境变量  
vi /etc/profile

在尾部追加行：  
export MAVEN\_HOME=/usr/local/maven

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

6:最后，使更改生效  
source /etc/profile

7:看下是否安装成功  
mvn -v

如果正常显示版本，说明部署成功～

Over～～～～

# 安装git

2 移除旧版本git

centos自带Git，7.x版本自带git 1.8.3.1（应该是，也可能不是），  
安装新版本之前需要使用yum remove git卸载（安装后卸载也可以）。

[root@Git ~]# git --version    ## 查看自带的版本git version 1.8.3.1

[root@Git ~]# yum remove git   ## 移除原来的版本

3 安装所需软件包

[root@Git ~]# yum install curl-devel expat-devel gettext-devel openssl-devel zlib-devel

[root@Git ~]# yum install gcc-c++ perl-ExtUtils-MakeMaker

下载&安装

[root@Git ~]# cd /usr/src

[root@Git ~]# wget https://www.kernel.org/pub/software/scm/git/git-2.7.3.tar.gz

5 解压

[root@Git ~]# tar xf git-2.7.3.tar.gz

6 配置编译安装

[root@Git ~]# cd git-2.7.3

[root@Git ~]# make configure

[root@Git ~]# ./configure --prefix=/usr/local/git ##配置目录

[root@Git ~]# make profix=/usr/local/git

[root@Git ~]# make install

7 加入环境变量

[root@Git ~]# echo "export PATH=$PATH:/usr/local/git/bin" >> /etc/profile

[root@Git ~]# source /etc/profile

8 检查版本

[root@Git git-2.7.3]# git --version

git version 2.7.3

|  |
| --- |
| git init //把这个目录变成Git可以管理的仓库  git add README.md //文件添加到仓库  git add . //不但可以跟单一文件，还可以跟通配符，更可以跟目录。一个点就把当前目录下所有未追踪的文件全部add了  git commit -m "first commit" //把文件提交到仓库  git remote add origin git@github.com:wangjiax9/practice.git //关联远程仓库  git push -u origin master //把本地库的所有内容推送到远程库上 |

走一遍

systemctl stop firewalld.service #停止firewall

systemctl disable firewalld.service #禁止firewall开机启动

firewall-cmd --state #查看默认防火墙状态（关闭后显示notrunning，开启后显示running）

# 安装jenkins

yum install wget

wget https://pkg.jenkins.io/redhat/jenkins-2.140-1.1.noarch.rpm

rpm -ivh jenkins-2.140-1.1.noarch.rpm

彻底卸载

service jenkins stop

yum clean all

yum -y remove jenkins

rm -rf /var/cache/jenkins

rm -rf /var/lib/jenkins/

配置jenkins端口

vi /etc/sysconfig/jenkins

|  |
| --- |
| ## Type: integer(0:65535)  ## Default: 8080  ## ServiceRestart: jenkins  #  # Port Jenkins is listening on.  # Set to -1 to disable  #  JENKINS\_PORT="8099" |

启动

sudo service jenkins start/stop/restart

如果启动失败:

Starting jenkins (via systemctl): Job for jenkins.service failed because the control process exited with error code. See "systemctl status jenkins.service" and "journalctl -xe" for details.

[FAILED]

加一个软连接即可

[root@localhost java]# ln -s /usr/local/java/jdk1.8.0\_11/bin/java /usr/bin/java

[root@localhost java]# sudo service jenkins start

Starting jenkins (via systemctl): [ OK ]

解锁密码

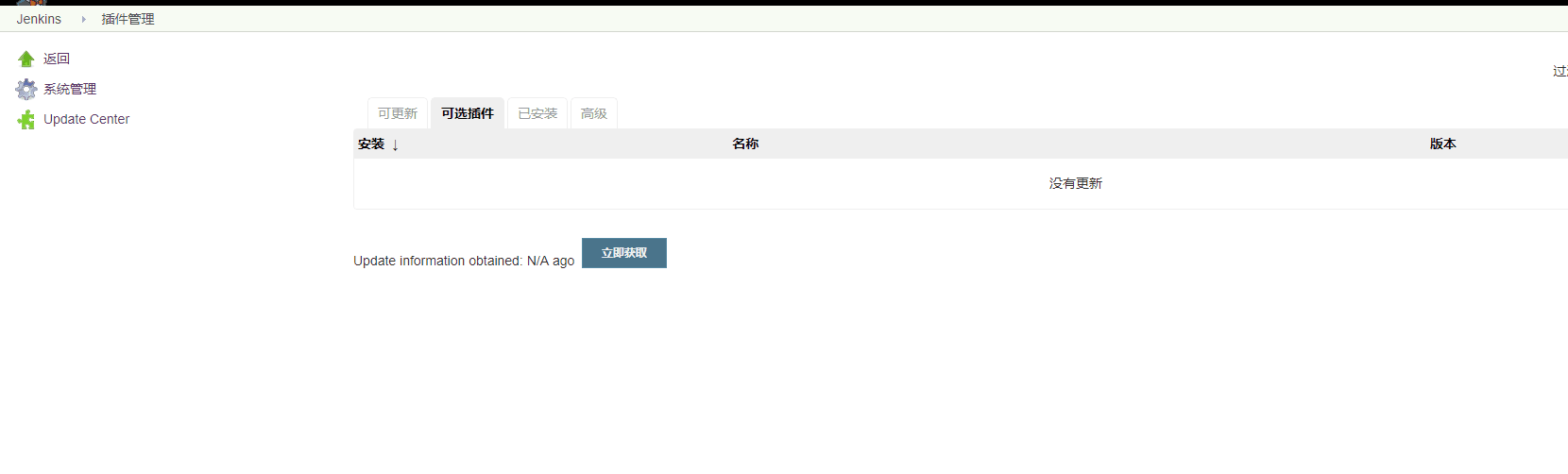
/var/lib/jenkins/secrets/initialAdminPassword

请从本地复制密码并粘贴到下面。

[root@localhost java]# cat /var/lib/jenkins/secrets/initialAdminPassword

b768931f804241449866b23857cda03b

如果插件页面为空白报错,则



**然后再打开一个新的窗口，输入网址**http://192.168.106.37:8099/pluginManager/advanced**，输入网址打开后滑动到页面下方，最底下有个【升级站点】，把其中的链接改成这个http的链接**

<http://updates.jenkins.io/update-center.json>

**然后在服务列表中关闭jenkins，再启动，可选插件里面就不是空的了，就有了各种插件，可以下载自己需要的插件了**

# jdk,tomcat,maven,jenkins,git构建ci

<https://www.cnblogs.com/xuxiaojian/p/9079132.html>

### 构建前提

由于构建的时候是基于Maven、git（GitHub）以及JDK1.8和Tomcat8来的，所以这些都需要和jenkins安装在同一台服务器上（Linux，centos7）。

### 全局工具配置



### 配置JDK

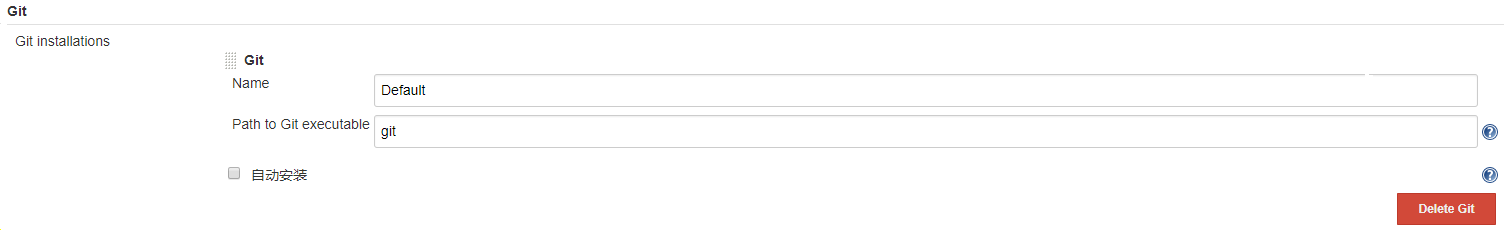


/usr/local/java/jdk1.8.0\_11

如何查看JAVA\_HOME？

执行：echo $JAVA\_HOME

### 配置Git



此处配置的是git的命令程序，如果你将git配置到了环境变量则直接可以写git，否则还是填写git命令路径

如何查看Git命令程序路径？

执行：whereis git

### 配置Maven



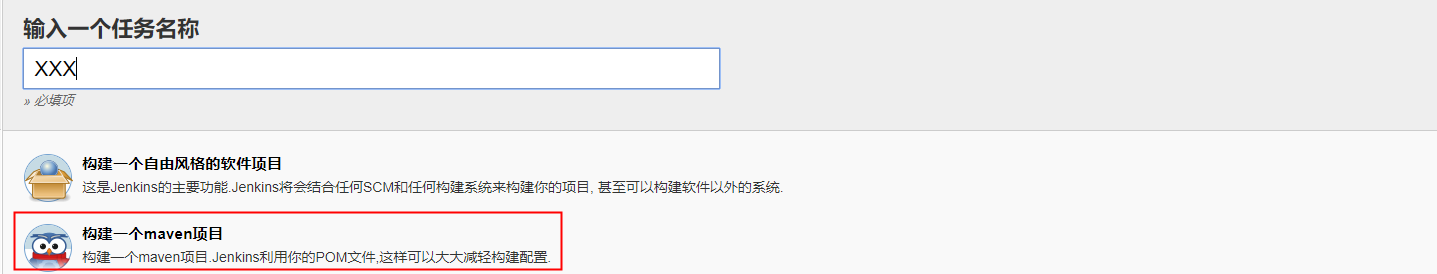
和jdk一样这里配置maven的安装路径。

如何查看maven安装路径？

执行：mvn --version

在输出结果中找到Maven home那一项就是要的路径，填上即可。

### 新建任务



https://images2018.cnblogs.com/blog/1205022/201805/1205022-20180528205938705-93244982.png

说明：下面没提到的配置都暂时先不用管。

### 【源码管理】

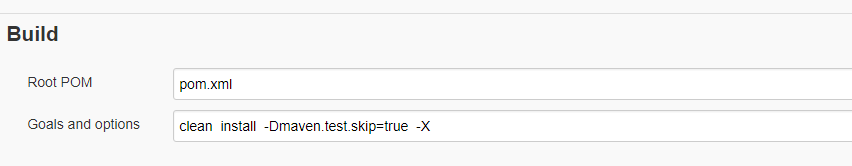
我用的GitHub。



 这里填写相应信息以便于jenkins从github上拉取指定分支代码。

### Build





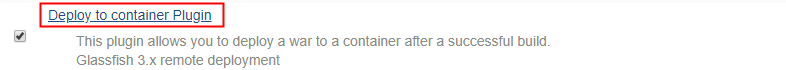
Goals and options填写运行mvn命令的参数选项，按照本地打包命令填写“mvn”之外的内容，比如我本地打包的时候用的

mvn clean package  -Dmaven.test.skip=true

此处就应该填写clean package  -Dmaven.test.skip=true即可

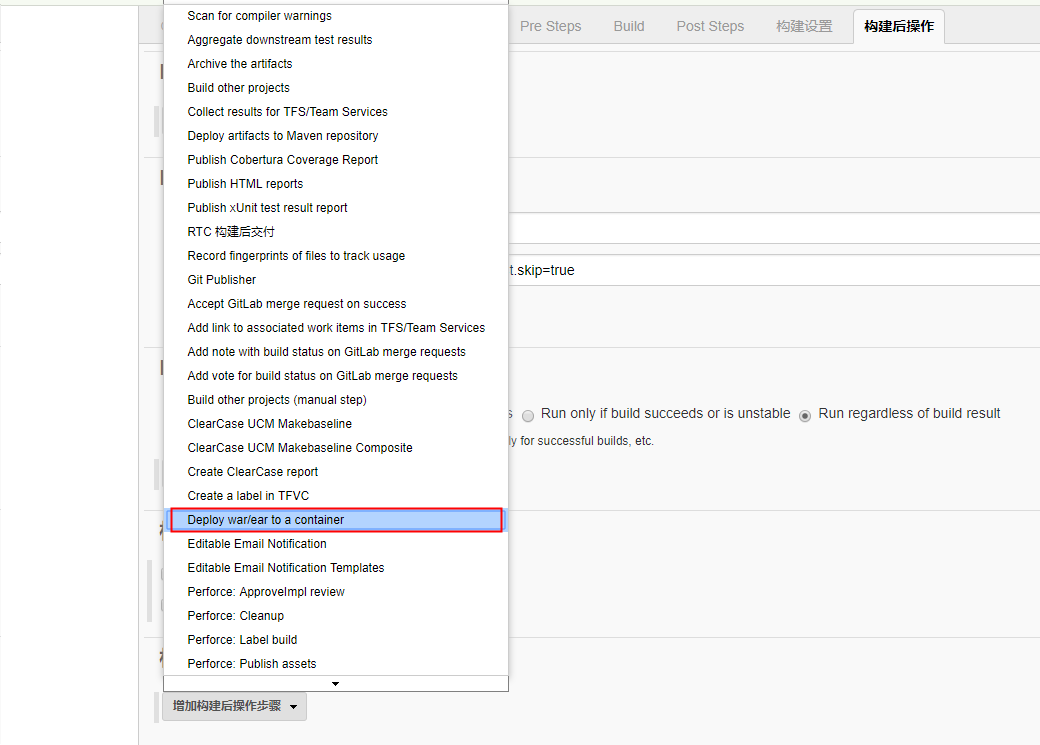
### 构建后操作

先将之前的配置保存，进入【系统配置】-【插件管理】-【可选插件】找到如下插件下载安装。



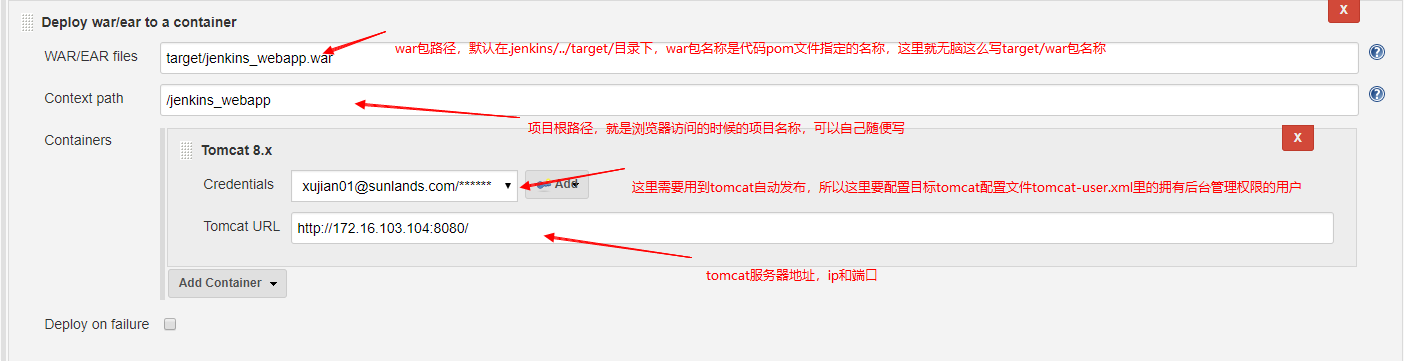
该插件是用来将war包发布到服务器上。

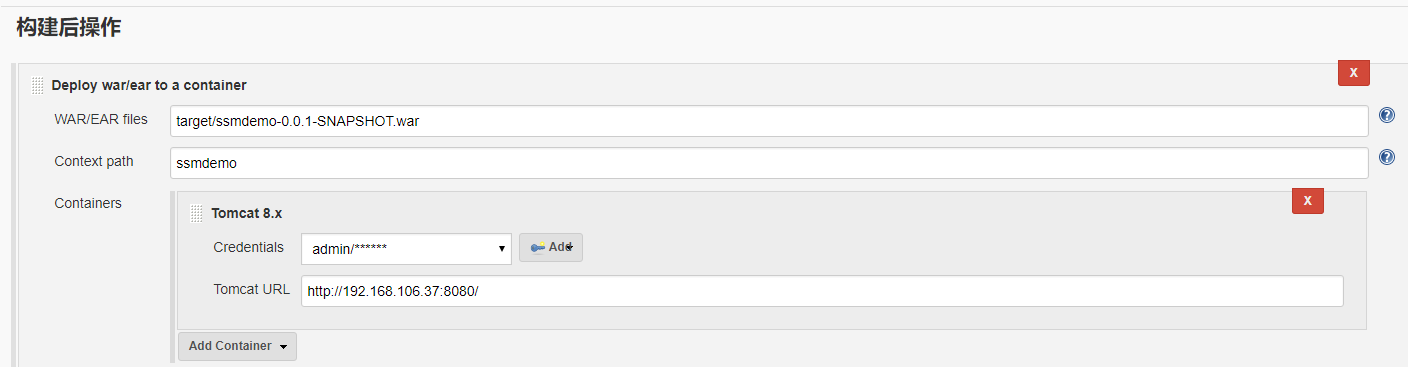
再回到刚才项目的配置【构建后操作】，点击【增加构建后操作步骤】选择下图所示插件



选择之后即可填写发布应用的信息。

配置tomcat之前先确保目标tomcat已经启动





### 修改tomcat-user.xml的权限:

<tomcat-users xmlns="http://tomcat.apache.org/xml"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"

version="1.0">

<role rolename="manager-gui"/>

<role rolename="admin-gui"/>

<role rolename="manager-script"/>

<user username="admin" password="admin" roles="manager-gui,admin-gui,manager-script"/>

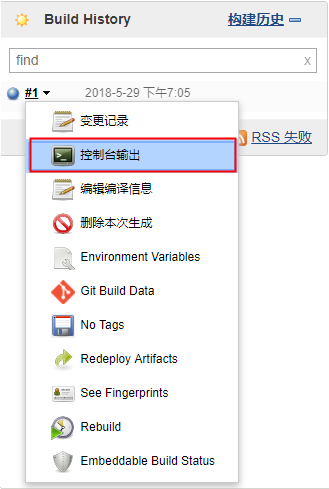
</tomcat-users>

到此配置完成！

### 构建项目



点击【立即构建】会出现如下试图



点击【控制台输出】可以看到构建日志，日志最终打印SUCCESS即表示构建成功。

### 验证

进入目标服务器tomcat/bin目录下，使用

tail -f ../logs/catalina.out

即可看到war包已经自动部署到tomcat上了。

浏览器输入项目访问地址就可以看到响应了

