## 一、jenkins简介

Jenkins是基于Java开发的一种[持续集成](http://baike.baidu.com/view/5253255.htm)工具，用于监控持续重复的工作，主要特点是

开源免费；跨平台，几乎支持所有的平台；

master/slave支持分布式的build；

web形式的可视化的管理页面；

安装配置超级简单；

tips及时快速的帮助；

已有的200多个插件；

前面安装非常简单，后面功能实现部分遇到一些问题。本文只在原来的基础上做了些改动，此文档后续完善！欢迎大家一起讨论。

## 二、jenkins安装部署

安装要求：

服务端系统 CentOS 6.6 jdk 1.8 tomcat8

这里主要讲两种安装方式

### 1、对于RedHat 系列的系统，使用rpm包安装

设置yum源

wget -O /etc/yum.repos.d/jenkins.repo<http://jenkins-ci.org/redhat/jenkins.repo>

rpm --import http://pkg.jenkins-ci.org/redhat/jenkins-ci.org.key

安装jenkins

yum -y install jenkins

安装最新版到/usr/lib/jenkins目录下，默认的jenkins家目录是/var/lib/jenkins

启动jenkins

servicejenkins start

jenkins默认运行在8080端口

2、tomcat中运行war包

war包下载路径：

<http://mirrors.jenkins-ci.org/war/latest/jenkins.war>

创建jenkins的主目录，有关jenkins的文件会放置到此目录下

mkdir -p /data/jenkins/home

创建运行tomcat的普通账号jenkins

groupaddjenkins

useradd -g jenkinsjenkins

解压tomcat ，清空webapps目录下的其他文件，放置war包到webapps目录下

tarzxvf apache-tomcat-8.0.24.tar.gz -C /data/jenkins

cd /data/jenkins/

mv apache-tomcat-8.0.24/ tomcat\_jenkins

rm -rftomcat\_jenkins/webapps/\*

cdtomcat\_jenkisn/webapps

wget<http://mirrors.jenkins-ci.org/war/latest/jenkins.war>

设置目录权限

chown -R jenkins:jenkins /data/jenkins/

java的配置

tarzxvfjdk-8u74-linux-x64.tar.gz -C /usr/local

mv /usr/local/jdk1.8.0\_74/ /usr/local/java

添加环境变量vim /etc/profile

JAVA\_HOME=/usr/local/java

PATH=$JAVA\_HOME/bin:$PATH

export PATH

以jenkins用户身份启动tomcat,并设置jenkins home 目录

su - jenkins

vim /home/jenkins/.bash\_profile # 添加一行

export JENKINS\_HOME=/data/jenkins/home/

source /home/jenkins/.bash\_profile

cd /data/jenkins/tomcat\_jenkins/bin/

[jenkins@proxy bin]$ ./startup.sh

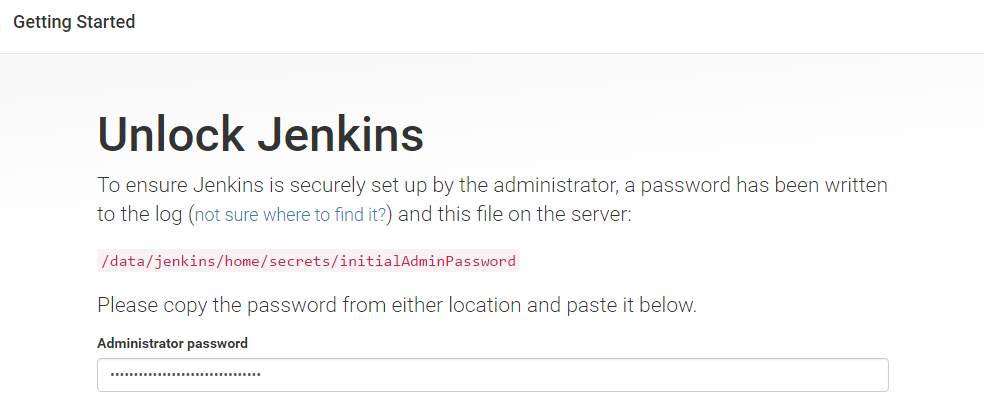
确认tomcat日志中无报错

访问web界面,继续完成jenkins的安装

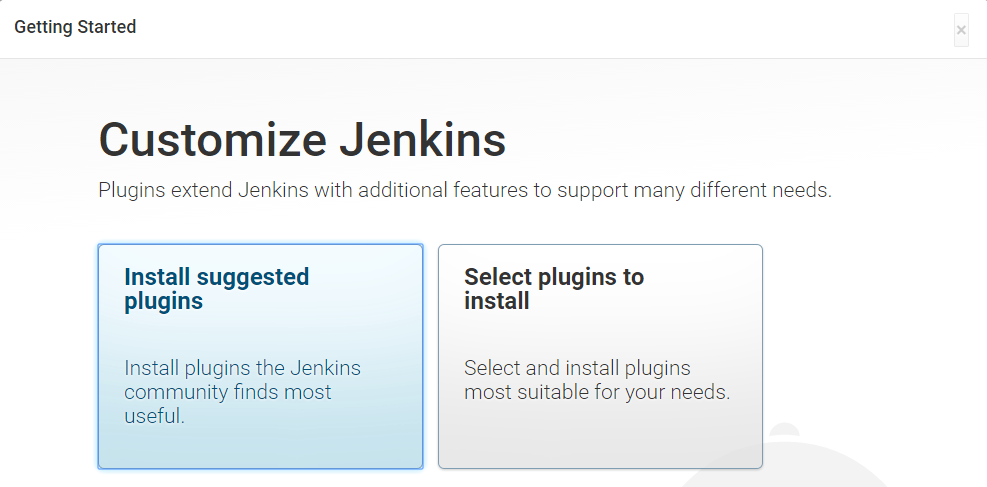
[http://ip:8080/jenkins](http://192.168.30.87:8080/jenkins)

在下面这个文件中找到初始化管理员密码，并输入下图方框中，此密码在catalina.out 中也可以找到，输好密码之后 点击右下角的"Continue"

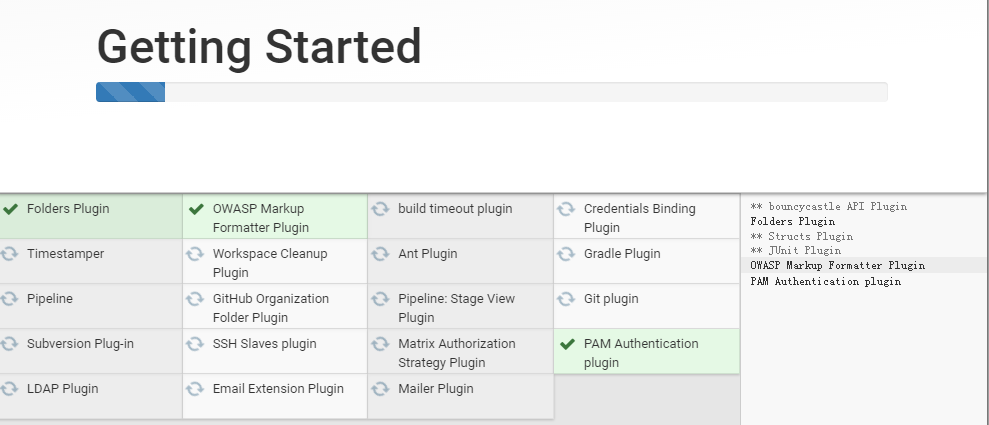
/data/jenkins/home/secrets/initialAdminPassword



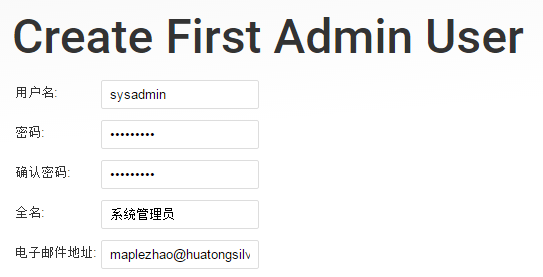
选择安装的插件，插件可以拓展jenkins的功能，这边选第一个，安装建议的插件



等待插件安装完成

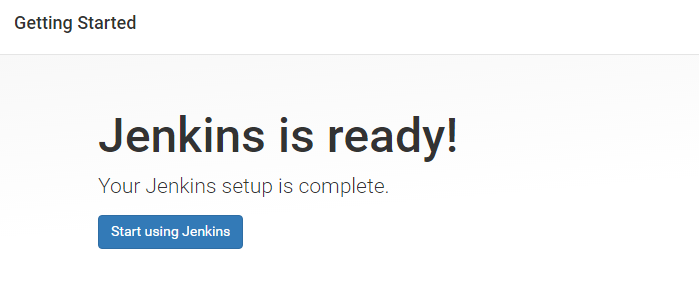


设置管理员账号的相关信息，然后点击保存和完成



密码是Jks#30D87

安装完成



### 3、jenkins配置

（1）、maven配置

下载

Maven 下载链接 <http://mirror.bit.edu.cn/apache/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.tar.gz>

内网下载地址 http://192.168.30.114:8800/software/apache/apache-maven-3.3.9-bin.tar.gz

解压

tarzxvf apache-maven-3.3.9-bin.tar.gz -C /data/jenkins/

cd /data/jenkins/

mv apache-maven-3.3.9/ maven

maven仓库的配置

<http://blog.csdn.net/wanghantong/article/details/36427433>

maven 中央仓库默认从外国的服务器下载依赖包，速度比较慢，也有可能出现无法下载的问题，所以配置为aliyun镜像源

vim /data/jenkins/maven/conf/settings.xml

配置下面一段时，确认配置文件中只有一个<mirrors>

<mirrors>

    <mirror>

      <id>alimaven</id>

      <name>aliyun maven</name>

      <url><http://maven.aliyun.com/nexus/content/groups/public/></url>

      <mirrorOf>central</mirrorOf>

    </mirror>

  </mirrors>

id,name 该镜像的唯一定义符。id用来区分不同的mirror元素

url 该镜像的URL，构建系统会优先考虑使用该url

mirrorof 被镜像服务器的id

修改maven本地仓库的路径

新建目录

mkdir /data/jenkins/maven\_repo

添加环境变量

vim /home/jenkins/.bash\_profile （添加一行）

export M2\_REPO=/data/jenkins/maven\_repo

source /home/jenkins/.bash\_profile

修改配置文件

vim /data/jenkins/maven/conf/settings.xml （root添加一行）

<localRepository>/data/jenkins/maven\_repo</localRepository>

复制已经下载好的jar包到新的路径下

cp -rf /home/jenkins/.m2/repository/\* /data/jenkins/maven\_repo/

### (2)、jenkins 全局工具配置

主要配置一些工具的路径

系统管理--> Global Tool Configuration

设置jdk ，新增JDK，如下图，输入一个别名和java主目录



设置maven可执行文件环境变量

vim /etc/profile.d/maven.sh

export PATH=$PATH:/data/jenkins/maven/bin

source /etc/profile.d/maven.sh

确认设置生效

mvn -v

Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5; 2015-11-11T00:41:47+08:00)

Maven home: /data/jenkins/maven

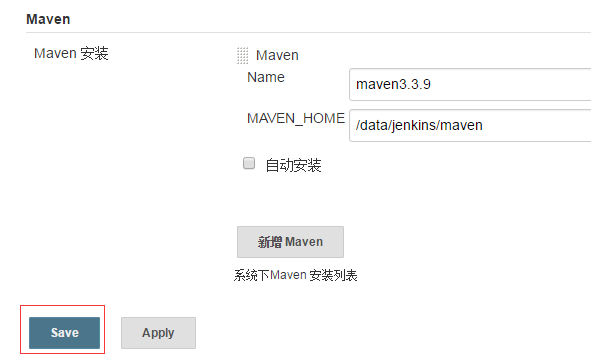
Java version: 1.8.0\_74, vendor: Oracle Corporation

Java home: /usr/local/java/jre

Default locale: en\_US, platform encoding: UTF-8

OS name: "linux", version: "2.6.32-504.el6.x86\_64", arch: "amd64", family: "unix"

同样的方式配置maven工具，最后点击Save

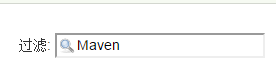


### （3）、安装插件编译项目需要的插件

Maven Integration plugin

这个插件提供一个高级集成为Maven2/3的项目

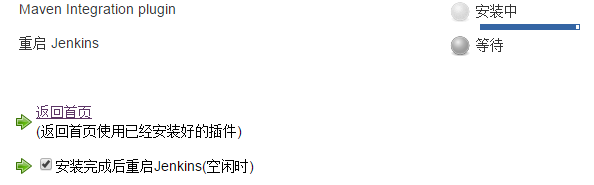
系统管理--> 管理插件--> 可选插件，在右上角输入Maven,过滤插件



从过滤出的结果中找到 Maven Integration plugin，在方框中点击鼠标左键，点击下面的“直接安装”按钮

clipboard.png

下面会自动切换到 安装/更新 插件的窗口，在“安装完成后重启Jenkins（空闲时）”前面点击鼠标左键，如下图，安装插件之后，会自动重启tomcat



[Workspace Cleanup Plugin](http://wiki.jenkins-ci.org/display/JENKINS/Workspace+Cleanup+Plugin) （已经安装）

This plugin deletes the project workspace after a build is finished.

此插件在构建完成之后删除工作空间。

[Publish Over SSH](https://wiki.jenkins-ci.org/display/JENKINS/Publish+Over+SSH+Plugin)

Publish files and/or execute commands over SSH (SCP using SFTP)

发布文件和/或者执行命令 通过SSH（SCP使用sftp方式）

注意：如果等待一会，网页界面还是没有变化，这个时候可以看下catalina.out 的日志，看下插件是否安装完成。出现这个问题的原因是右上角“允许自动刷新”需要点击下

（4）、htwhr项目配置

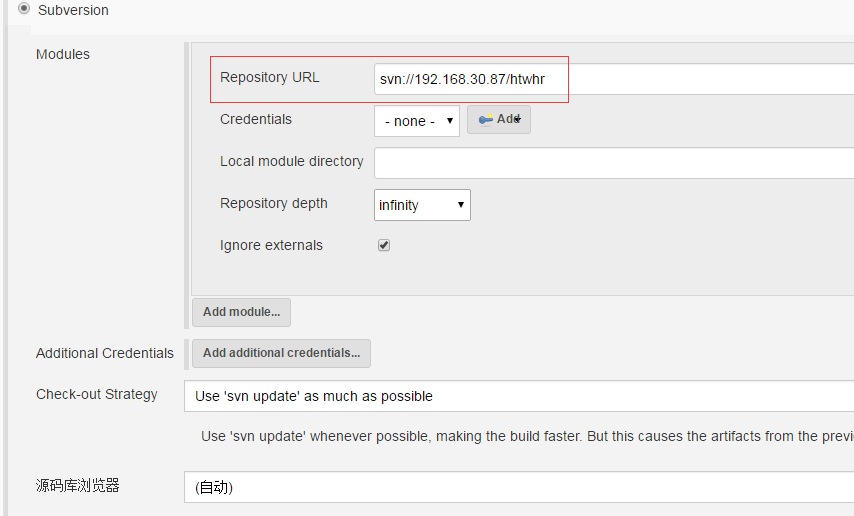
新建--> 输入项目名称htwhr-->构建一个maven项目-->OK

通用配置 --> 描述 可以输入项目的描述，比如我们输入测试环境htwhr项目

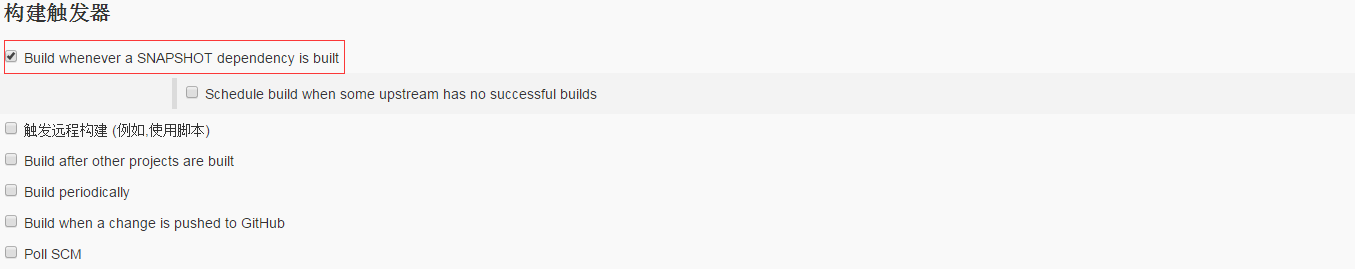


源码管理 --> Subversion

输入仓库的URL，其他几项保持默认即可，这个仓库匿名可以读取，但是不可以写入



构建触发器



点击最下面的保存按钮

在新的窗口右边点击“立即构建”



点击#1 ，选择Console Output 查看输出信息



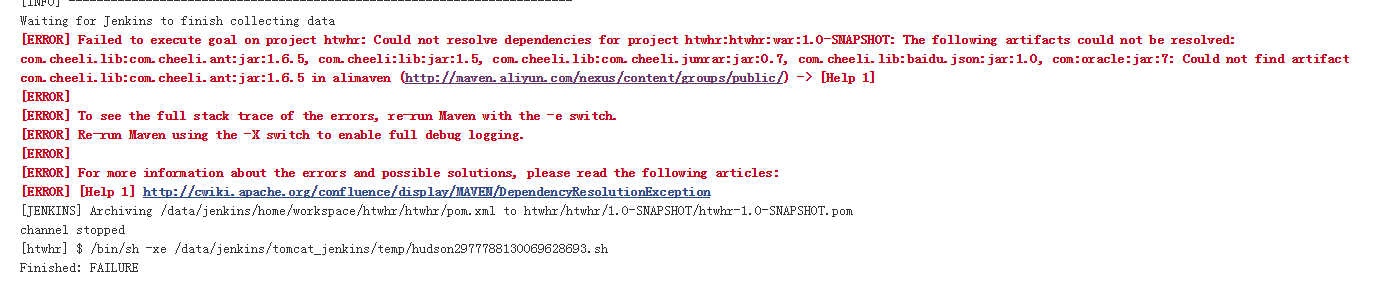
第一次构建会输出很多日志



主要是从svn检出源码到workspace，

从仓库下载公共jar包

下图报错的原因是，私有jar包无法从公共仓库上下载，手动将jar包放到相应的位置



[root@proxycheeli]# pwd

/data/jenkins/maven\_repo/com/cheeli

[root@proxycheeli]# rm -rf \*

[root@elkcheeli]#scp -r lib/ root@192.168.30.87:/data/jenkins/maven\_repo/com/cheeli

修改文件权限

chown -R jenkins:jenkins /data/jenkins/

确认jar包已存在

[root@proxy lib]# tree

.

├── 1.5

│   └── lib-1.5.jar

├── baidu.json

│   └── 1.0

│   └── baidu.json-1.0.jar

├── com.cheeli.ant

│   └── 1.6.5

│   └── com.cheeli.ant-1.6.5.jar

├── com.cheeli.junrar

│   └── 0.7

│   └── com.cheeli.junrar-0.7.jar

└── com.cheeli.lib

└── 1.0

└── com.cheeli.lib-1.0.jar

重新编译项目



上传oracle-7.jar到/data/jenkins/maven\_repo/com/oracle/7目录下

确认文件已经存在，删除多余的两个文件

[root@proxy 7]# ls -l

total 3624

-rw-r--r-- 1 jenkinsjenkins 3698892 Jan 10 01:33 oracle-7.jar

-rw-r--r-- 1 jenkinsjenkins 274 Jan 10 01:18 oracle-7.jar.lastUpdated

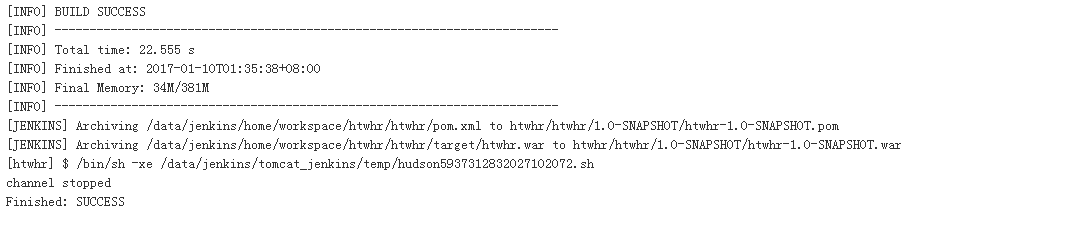
-rw-r--r-- 1 jenkinsjenkins 274 Jan 10 01:18 oracle-7.pom.lastUpdated

[root@proxy 7]# rm -rf oracle-7.jar.lastUpdated

[root@proxy 7]# rm -rf oracle-7.pom.lastUpdated

重新编译

编译成功



查看编译后的war包

[root@proxy target]# pwd

/data/jenkins/home/workspace/htwhr/htwhr/target

[root@proxy target]# ls -lhhtwhr.war

-rw-rw-r-- 1 jenkinsjenkins 44M Jan 10 01:35 htwhr.war

每次打包完成之后，将war包放到同一个目录下，并通过scp命令复制到远程服务器

mkdir -p /data/jenkins/wars/htwhr

点击post steps--> add post-build step-->Execute shell

在方框中输入下面三行命令

time=`date +%F-%T`

mv /data/jenkins/home/workspace/htwhr/htwhr/target/htwhr.war /data/jenkins/wars/htwhr/htwhr-$time.war

scp -P 22 /data/jenkins/wars/htwhr/htwhr-$time.war htwhr@192.168.30.89:/data/htwhr/wars/

定义时间、移动war包到htwhr目录，复制war包到远程服务器

用jenkins执行下面三条命令

生成公钥和私钥

ssh-keygen -t rsa

复制公钥到目标服务器（密码htwhr）

ssh-copy-id id\_rsa.pub htwhr@192.168.30.89

测试无密码登陆

[jenkins@proxy .ssh]$ ssh htwhr@192.168.30.89

Last login: Tue Jan 10 01:57:56 2017 from 192.168.30.114

[htwhr@db ~]$ exit

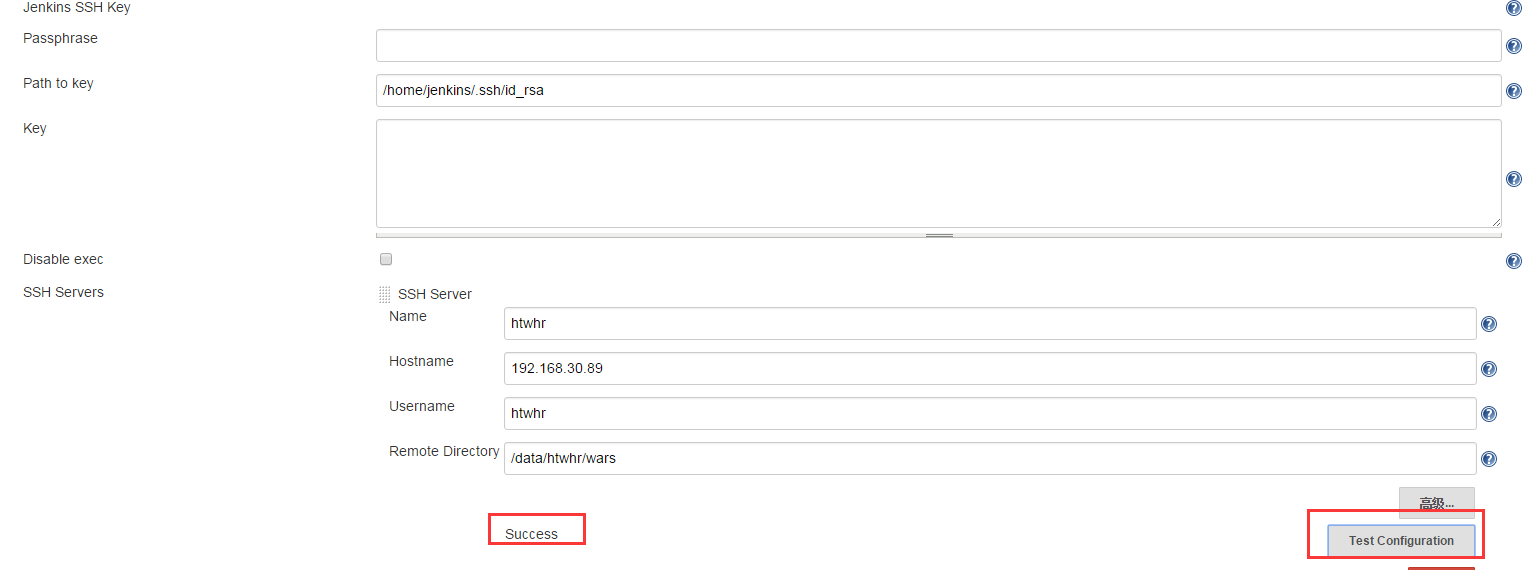
logout

增加ssh server

点击系统管理-->系统配置

找到**Publish over SSH**

输入jenkins私钥的路径，在ssh server那边输入名字，ip，账号，和复制war包到的目录，点击Test configuration，出现Success，最后点击保存



在服务端新建jenkins部署脚本

vim /data/jenkins/wars/depoly\_htwhr.sh

#!/bin/bash

#deploy tomcat\_htwhr

time1=`date +%Y%m%d`

time2=`date +%F-%T`

base=/data/htwhr/tomcat\_htwhr

export JAVA\_HOME=/data/htxh/jdk1.8.0\_74

pid1=`ps aux |greptomcat\_htwhr |grep -v grep |awk '{print $2}'`

kill -9 $pid1

if [ -d /home/htwhr/backup/$time1 ]

then

echo "dir exits"

else

mkdir -p /home/htwhr/backup/$time1

fi

cd $base

tarzcf /home/htwhr/backup/$time1/htwhr\_$time2.tar.gz webapps/\*

if [ $? -eq0 ]

then

rm -rf webapps/\*

rm -rf work/\*

else

exit 1

echo "htwhr backup failed"

fi

newwar=`ls -t /data/htwhr/wars/htwhr-\*.war | head -1`

cp -f $newwarwebapps/htwhr.war

cd $base/bin

./startup.sh

修改脚本的执行权限

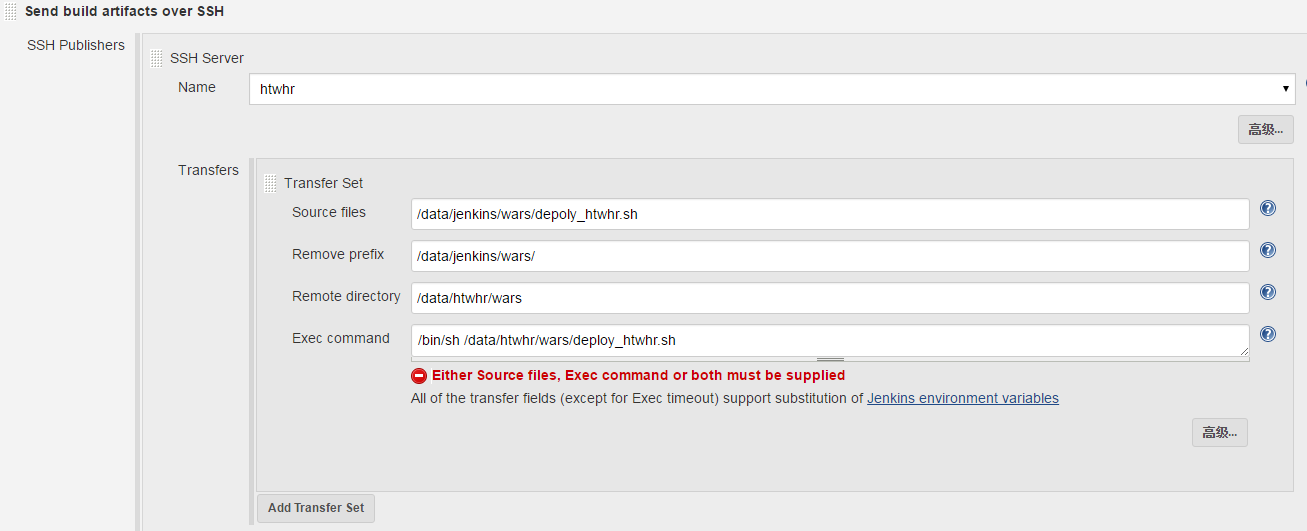
chmoda+x /data/jenkins/wars/depoly\_htwhr.sh

jenkins web端配置

找到htwhr项目，点击配置

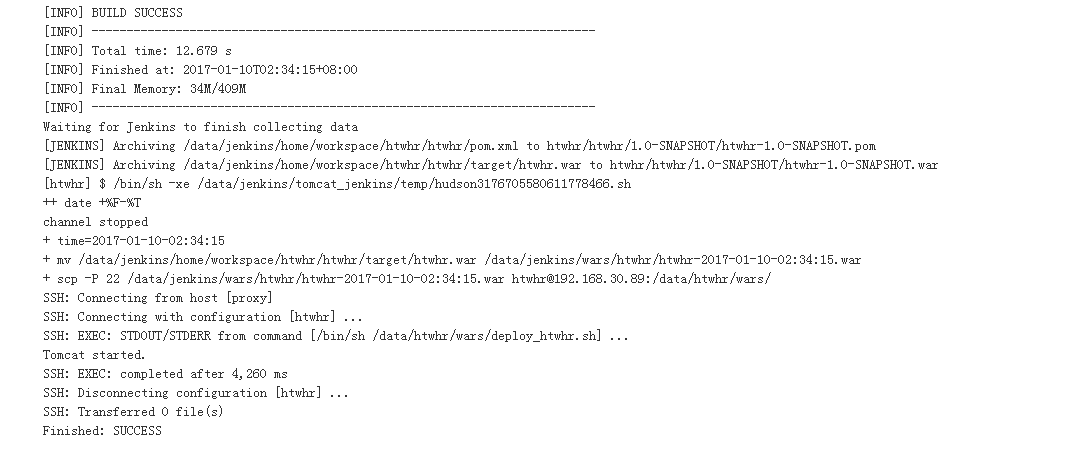
构建后操作，

选择服务器和设置脚本的路径，远程目录，执行的脚本命令，最后点击保存



点击立即构建

目标服务器tomcat需要是运行状态，否则会报错



执行成功

确认目标机器上tomcat进程正常

[htwhr@db 20170110]$ ps aux |grep tomcat

htwhr 18503 15.2 7.8 4706380 632584 ? Sl 02:34 0:14 /data/htxh/jdk1.8.0\_74/bin/java -Djava.util.logging.config.file=/data/htwhr/tomcat\_htwhr/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.endorsed.dirs=/data/htwhr/tomcat\_htwhr/endorsed -classpath /data/htwhr/tomcat\_htwhr/bin/bootstrap.jar:/data/htwhr/tomcat\_htwhr/bin/tomcat-juli.jar -Dcatalina.base=/data/htwhr/tomcat\_htwhr -Dcatalina.home=/data/htwhr/tomcat\_htwhr -Djava.io.tmpdir=/data/htwhr/tomcat\_htwhr/temp org.apache.catalina.startup.Bootstrap start

htwhr 18645 0.0 0.0 103256 868 pts/0 S+ 02:36 0:00 grep tomcat

root用户重新加载htschedule

supervisorctl reload

查看状态

supervisorctl status

htschedule RUNNING pid 19170, uptime 0:00:06

访问网页



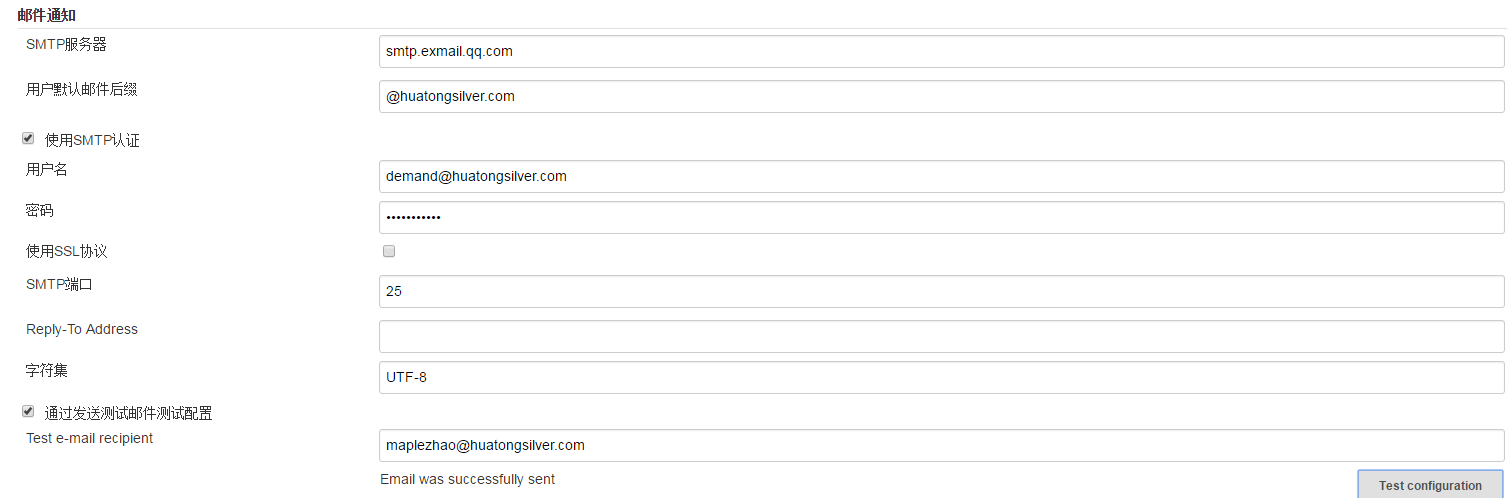
（5）邮件通知

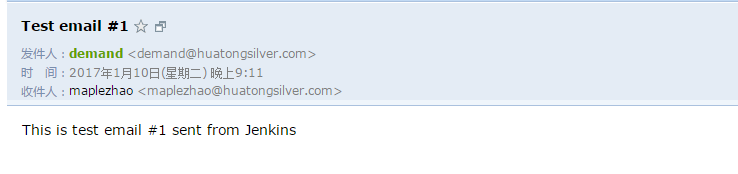
配置发件人地址：[demand@huatongsilver.com](mailto:demand@huatongsilver.com)

系统管理--> 系统设置，设置系统管理员邮箱地址

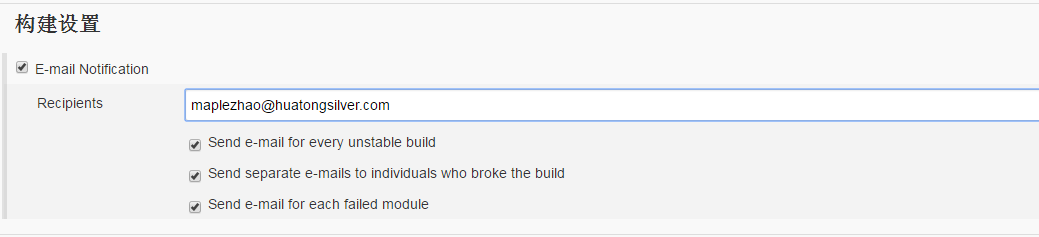


配置smtp，账，密码等，并发送测试邮件





配置单个项目的收件人



每次不稳定都会发邮件

单独发邮件给对构建造成不良影响的人

为每次失败的模块发邮件

问题：

1、tomcat catalina.out 日志报错

31-Dec-2016 15:35:31.516 WARNING [http-nio-8080-exec-1] org.apache.catalina.webresources.Cache.getResource Unable to add the resource at [/help/user/fullName.html] to the cache because there was insufficient free space available after evicting expired cache entries - consider increasing the maximum size of the cache

31-Dec-2016 15:35:31.516 WARNING [http-nio-8080-exec-1] org.apache.catalina.webresources.Cache.getResource Unable to add the resource at [/help/user/description\_ru.html] to the cache because there was insufficient free space available after evicting expired cache entries - consider increasing the maximum size of the cache

无法将[/help/user/description\_ru.html]中的资源添加到高速缓存，因为在驱逐到期的高速缓存条目后没有足够的可用空间 - 请考虑增加高速缓存的最大大小

修改tomcat 安装目录下conf目录下的context.xml添加如下行：然后重启tomcat

<Resources cachingAllowed="true" cacheMaxSize="100000" /> 单位是KB

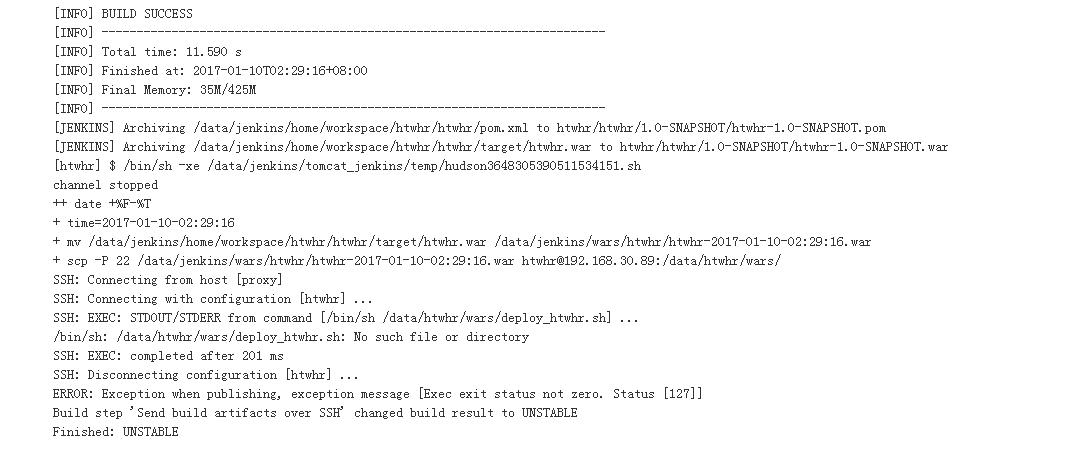
Resources 是资源定义元素

cachingAllowed：如果此标志的值是true，将用于静态资源的高速缓存。如果没有指定，该标志的默认值是true

cacheMaxSize ：以KB为单位的静态资源缓存的最大尺寸。如果未指定，则缺省值为10240 （10兆字节）。这个值可以在Web应用程序运行时进行更改（例如，通过JMX）。如果缓存正在使用的内存大于新的限制缓存将尝试规模随着时间的推移，以满足新的限制，以减少。如果需要的话，cacheObjectMaxSize将降低，以确保它是不大于 cacheMaxSize/20。

apache 官网解释的文章：<https://tomcat.apache.org/tomcat-8.0-doc/config/resources.html#Attributes>

2、jenkins服务器的脚本没有传输到目标服务器上



解决方法，在目标服务器相应的路径先放脚本

(6)待续