[Jenkins](http://www.51testing.com/html/26/javascript:;" \t "http://www.51testing.com/html/26/_self) 是一个开源软件项目，旨在提供一个开放易用的软件平台，使软件的持续集成变得可能。现在[软件开发](http://www.51testing.com/html/26/javascript:;" \t "http://www.51testing.com/html/26/_self)追求的是效率以及质量，Jenkins使得自动化成为可能！

**环境**

　　Ubuntu 14.10 (GNU/Linux 3.16.0-33-generic x86\_64)

**准备工作**

　　●Git版本控制服务器

　　●Tomcat发布服务器

　　●Jenkins服务器(提前安装好[Maven](http://www.51testing.com/html/26/javascript:;" \t "http://www.51testing.com/html/26/_self),Git,Jdk)

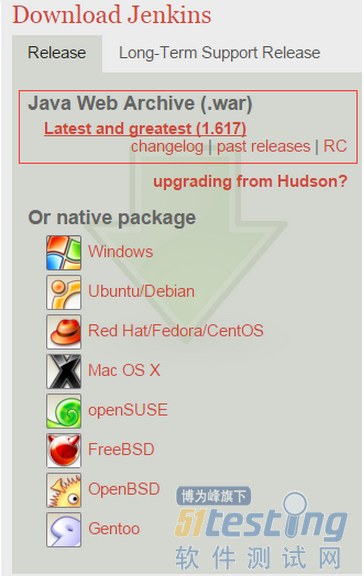
　　实验时可以在同一台机器配置，但是生产不建议，一台机器挂了，所有服务器都挂了。

　　废话不多说，直接上干货！

**步骤**

**安装Jenkins**

　　下载Jenkins War包，Jenkins官网 。

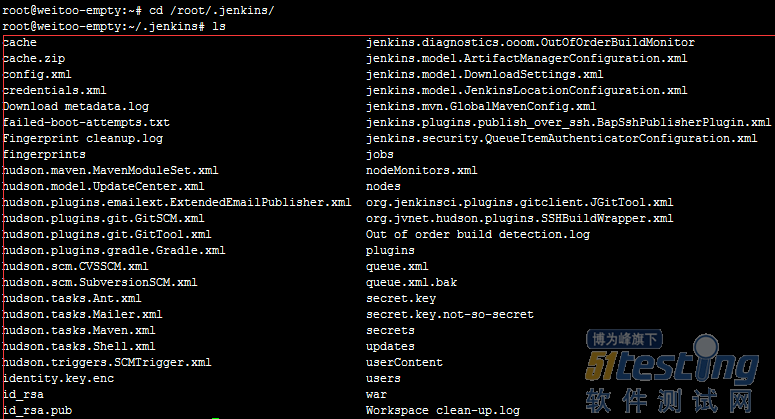
[](http://www.51testing.com/batch.download.php?aid=79633)

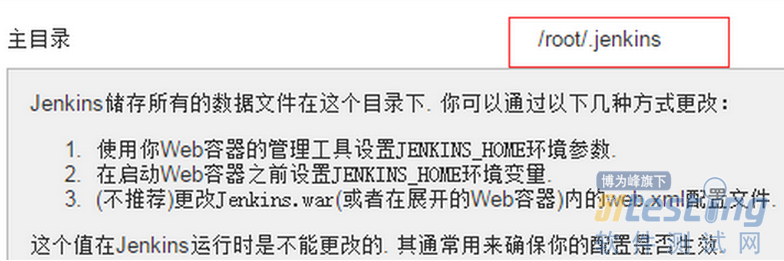
　　启动Jenkins ，将War包放入Tomcat容器里，启动Tomcat。

[](http://www.51testing.com/batch.download.php?aid=79634)

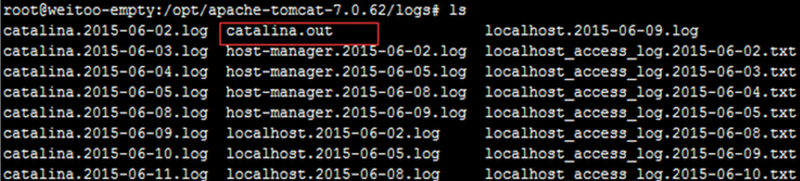
　　提示：

　　此时Jenkins在初始化配置目录，其默认配置目录路径为当前用户下的.jenkins目录，用户也可以自定义目录，Jenkins默认是把配置文件中的数据读到内存中，如果你替换了之前的配置文件，此时需要点击Jenkins的读取设置或者重启Tomcat,如果此时Jenkins页面无响应，则应该查看Tomcat的Catalina.out,多半是由于内存溢出造成(解决方法增大Tomcat调用Java虚拟机时内存大小,本文不做重点)，运行Jenkins的服务器配置最好内存1G以上，因为后续会加入一些Jenkins插件，有一些会比较占用内存，导致Jenkins启动不了。

[](http://www.51testing.com/batch.download.php?aid=79635)

[](http://www.51testing.com/batch.download.php?aid=79636)

[](http://www.51testing.com/batch.download.php?aid=79637)

[](http://www.51testing.com/batch.download.php?aid=79638)

**安装Jenkins插件**

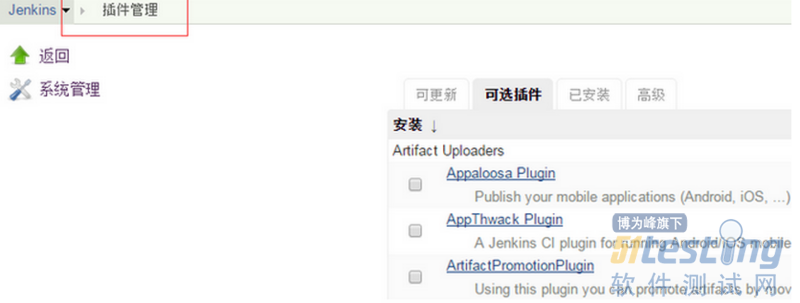
　　Email Extension Plugin (邮件通知)

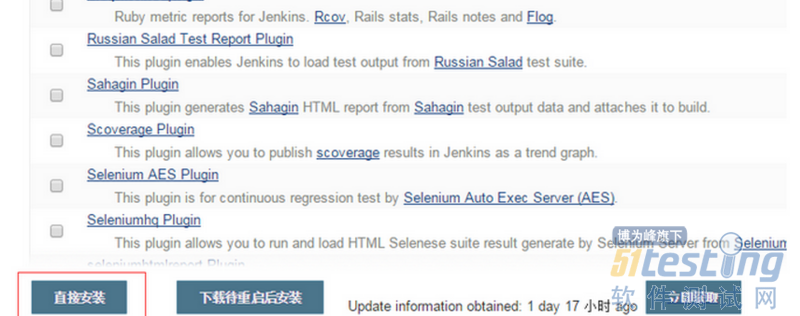
　　GIT plugin (可能已经默认安装了)

　　Publish Over SSH (远程Shell)

　　安装方法：

　　首页->系统管理->管理插件->可选插件->过滤(搜索插件名)->勾选->点击最下面直接安装即可(需要等待一段时间,详情可以看catalina.out日志变化)

[](http://www.51testing.com/batch.download.php?aid=79639)

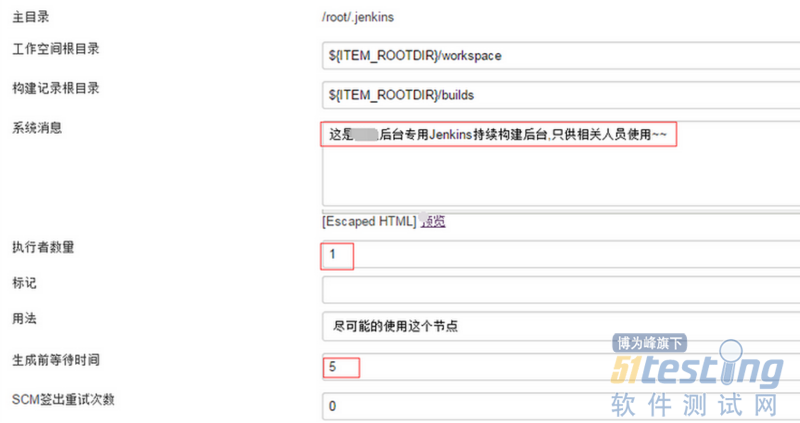
[](http://www.51testing.com/batch.download.php?aid=79640)

**配置Jenkins**

**配置基本信息**

　　每个选项后都有个问号解释当前含义(此步新手可以略过，默认不填即可)

　　配置方法：首页->系统管理->系统设置

[](http://www.51testing.com/batch.download.php?aid=79641)

**配置邮件**

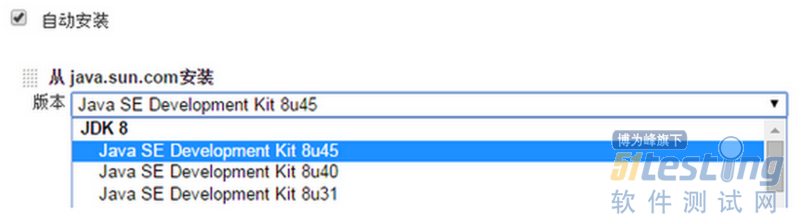
　　管理员邮件地址就是邮件的发件人地址(必须和后面邮件配置发件人邮箱一致，否则发不成功邮件)

[](http://www.51testing.com/batch.download.php?aid=79642)

**配置Jdk**

　　JAVA\_HOME为Jdk路径 其中Jdk也可以从这里下载安装解压(不推荐,需要填写oracle account)

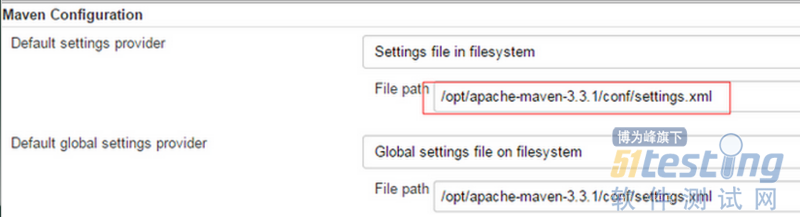
[](http://www.51testing.com/batch.download.php?aid=79643)

[](http://www.51testing.com/batch.download.php?aid=79644)

**配置 Maven**

　　配置 Maven Configuration

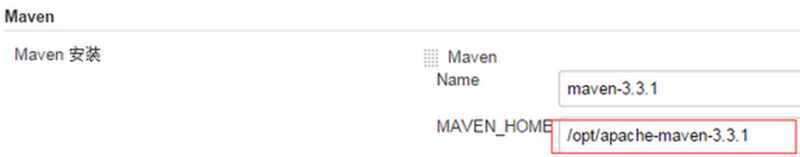
　　路径为maven的setting.xml路径(Maven安装略)

[](http://www.51testing.com/batch.download.php?aid=79645)

　　配置Maven项目

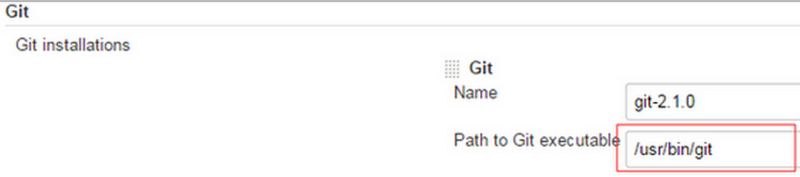
[](http://www.51testing.com/batch.download.php?aid=79646)

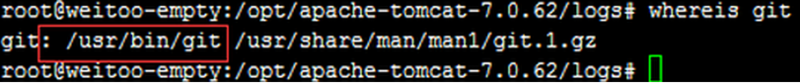
　　配置Maven安装目录

[](http://www.51testing.com/batch.download.php?aid=79647)

**配置 Git**

　　其中Path to Git executable为你git执行的路径 一般默认是/usr/bin/git ,如有差异，可以whereis git

[](http://www.51testing.com/batch.download.php?aid=79648)

[](http://www.51testing.com/batch.download.php?aid=79649)

**配置邮件**

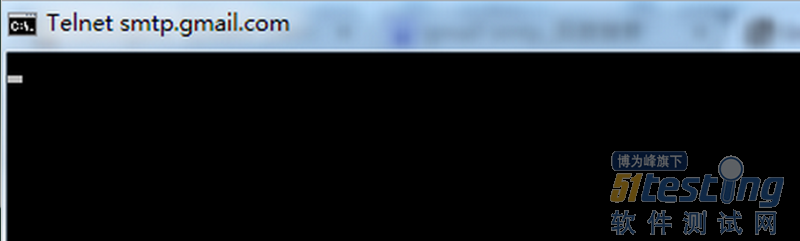
　　邮件模板配置

　　配置好邮件的模板(可自定义html编写) User Name为用户名 Password为密码 SMTP不同邮箱不同,请自行google(另外gmail邮件如无代理翻墙，请勿用，推荐163比较好配置)

**未翻墙**

[](http://www.51testing.com/batch.download.php?aid=79650)

**翻墙后**

[](http://www.51testing.com/batch.download.php?aid=79651)

**模板效果图**

[](http://www.51testing.com/batch.download.php?aid=79652)

　　Default Subject 代码：

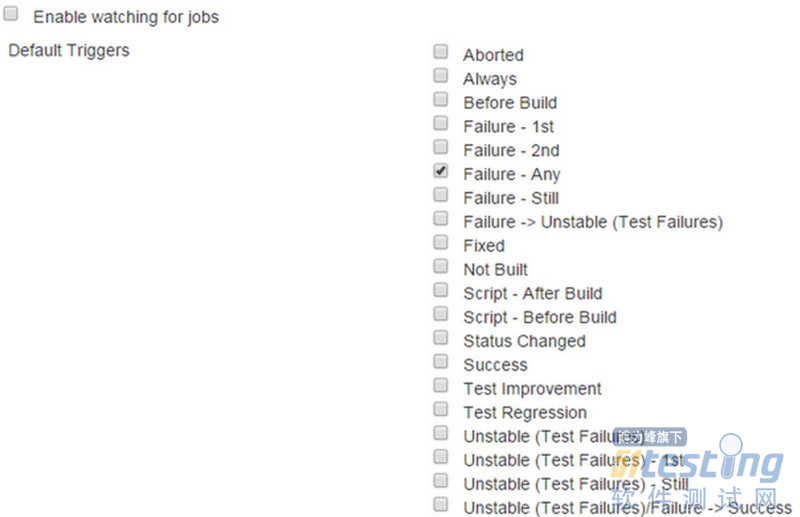
　　构建通知:$PROJECT\_NAME - Build # $BUILD\_NUMBER - $BUILD\_STATUS!

　　模板Default Content分享：

|  |
| --- |
| (本邮件是程序自动下发的，请勿回复，<span style="color:red">请相关人员fix it,重新提交到git 构建</span>)<br/><hr/>  　　项目名称：$PROJECT\_NAME<br/><hr/>  　　构建编号：$BUILD\_NUMBER<br/><hr/>  　　GIT版本号：${GIT\_REVISION}<br/><hr/>  　　构建状态：$BUILD\_STATUS<br/><hr/>  　　触发原因：${CAUSE}<br/><hr/>  　　构建日志地址：<a href="${BUILD\_URL}console">${BUILD\_URL}console</a><br/><hr/>  　　构建地址：<a href="$BUILD\_URL">$BUILD\_URL</a><br/><hr/>  　　变更集:${JELLY\_SCRIPT,template="html"}<br/><hr/> |

　　配置邮件触发器

　　当失败的时候，会触发邮件通知，这个功能比较实用。

[](http://www.51testing.com/batch.download.php?aid=79653)

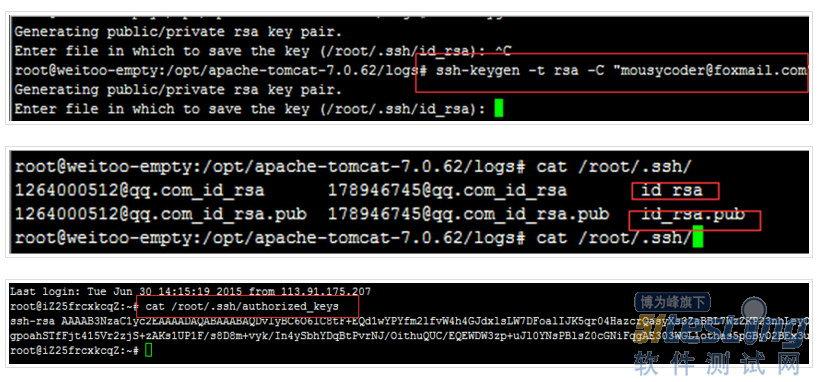
**配置 Publish over SSH**

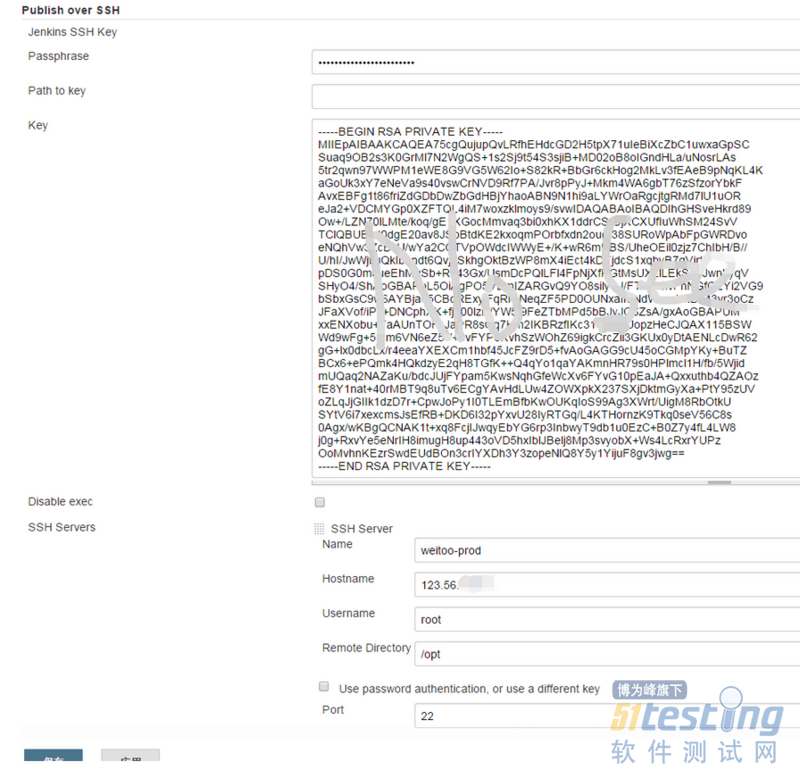
　　远程执行shell脚本 采用公钥私钥连接 其中Key里贴的是私钥 远程被管理的主机里贴的是公钥,这2台主机就是相互信任，这样scp等操作就不需要输入用户名和密码。

　　公钥私钥生成方法：

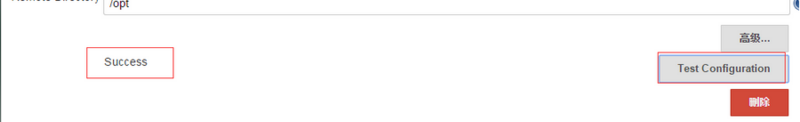
　　1.管理主机linux 上 ssh-keygen -t rsa -C "mousycoder@foxmail.com 一路回车 会在/root/.ssh下生成id\_rsa(私钥) id\_rsa.pub(公钥)。

　　2.copy 公钥的内容到远程需要通信(被管理)的主机 /root/.ssh/authorized\_keys 如无此目录文件则手动创建。

[](http://www.51testing.com/batch.download.php?aid=79654)

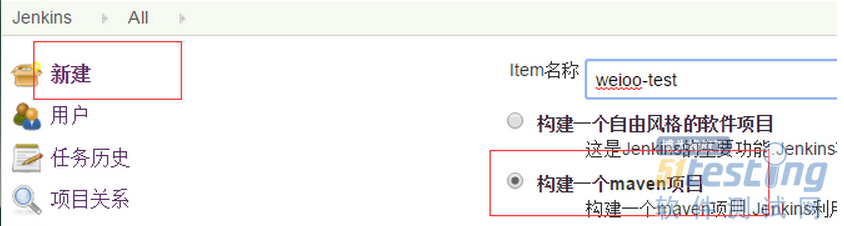
[](http://www.51testing.com/batch.download.php?aid=79655)

　　配置完之后可以Test Configuration

[](http://www.51testing.com/batch.download.php?aid=79656)

　　配置 Job

　　步骤:首页->新建->构建一个maven项目(输入item名称)->进入该项目->配置

[](http://www.51testing.com/batch.download.php?aid=79657)

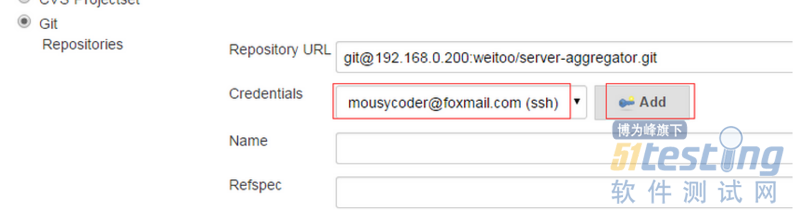
[](http://www.51testing.com/batch.download.php?aid=79658)

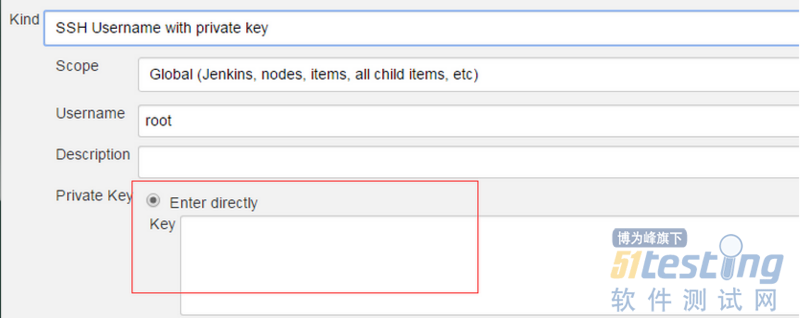
　　JOB基本信息

[](http://www.51testing.com/batch.download.php?aid=79659)

　　项目源码管理

　　Repository UR 项目地址 Credentials授权可以是SSH也可以是用户名密码(SSH方法同上)

[](http://www.51testing.com/batch.download.php?aid=79660)

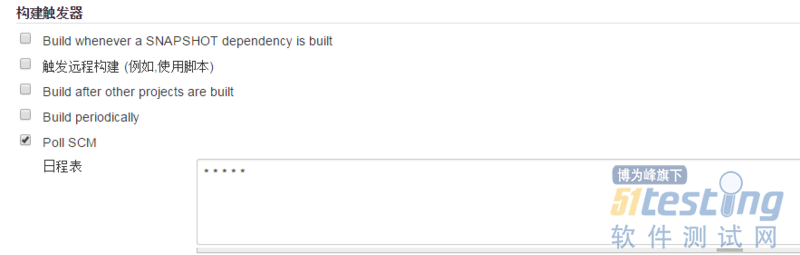
[](http://www.51testing.com/batch.download.php?aid=79661)

　　选择需要构建的分支，我们项目采用git工作流 默认master和develop 平时开发构建develop分支，正式上线构建master并且打标签(前公司git提交标准化相当复杂，分支相当多,这里大家可以根据实际情况来)。

[](http://www.51testing.com/batch.download.php?aid=79662)

　　构建触发器

　　这里我们选择poll轮询每隔1分钟去检测git仓库代码库版本,如果有更改则立刻构建，这里大家可以根据自己团队实际情况去制定，当然还有另外一个插件gitlab-hook可以主动去通知jenkins构建,不过插件所占内存比较大，需要增大tomcat虚拟机内存配置，不然会内存溢出，个人觉得如果一个团队人很多的话，选择poll更适合并且时间间隔设置长一些，避免频繁构建,gitlab-hook 适合人很少甚至一个人的情况。

[](http://www.51testing.com/batch.download.php?aid=79663)

　　构建命令

　　我们采用最简单的clean install 当然这里可以根据各自需求

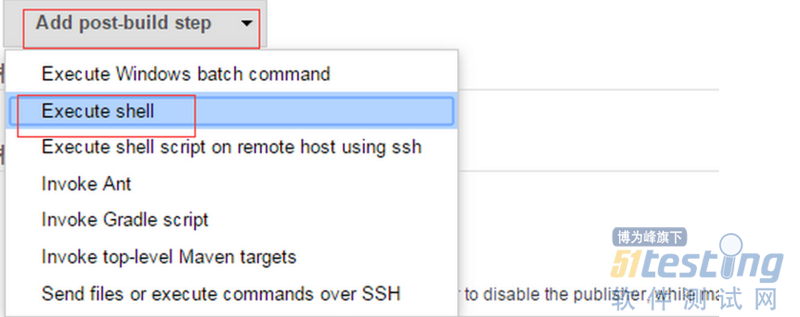
　　例如 部署后的产物上传到nexus等,详情参考 Maven命令

　　clean install deploy:deploy-file -DgroupId=com.weitoo -DartifactId=common -Dversion=0.1-SNAPSHOT -Dpackaging=jar -Dfile=D:\workspace\server-aggregator\common\target\common-0.1-SNAPSHOT.jar -Durl=http://192.168.0.200:8081/nexus/content/repositories/thirdparty/ -DrepositoryId=thirdparty

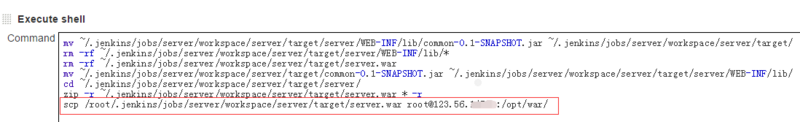
[](http://www.51testing.com/batch.download.php?aid=79664)

　　Add post-build step

　　构建成功后执行shell命令

[](http://www.51testing.com/batch.download.php?aid=79665)

　　该shell的目的是取出war包lib中其他所有lib包 只留下common-0.1-SNAPSHOT.jar 大大减少war包大小(完整war包30M 传包到阿里云服务器需要2分多,精简后2M，10秒多,大大提高构建速度)。

[](http://www.51testing.com/batch.download.php?aid=79666)

　　分享我的Shell

　　mv ~/.jenkins/jobs/server/workspace/server/target/server/WEB-INF/lib/common-0.1-SNAPSHOT.jar ~/.jenkins/jobs/server/workspace/server/target/

　　rm -rf ~/.jenkins/jobs/server/workspace/server/target/server/WEB-INF/lib/\*

　　rm -rf ~/.jenkins/jobs/server/workspace/server/target/server.war

　　mv ~/.jenkins/jobs/server/workspace/server/target/common-0.1-SNAPSHOT.jar ~/.jenkins/jobs/server/workspace/server/target/server/WEB-INF/lib/

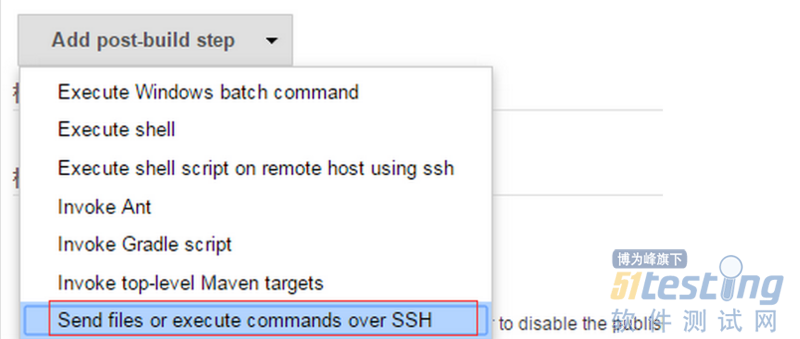
　　cd ~/.jenkins/jobs/server/workspace/server/target/server/

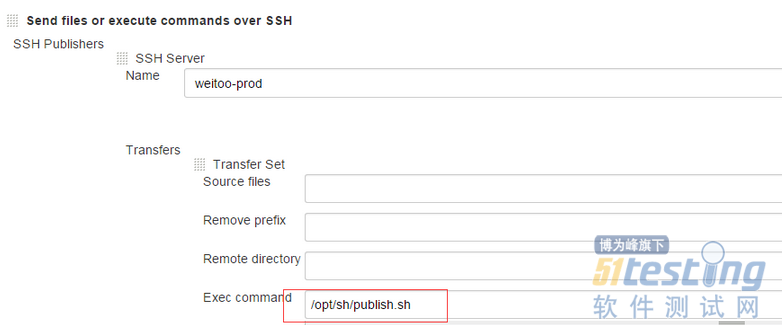
　　zip -r ~/.jenkins/jobs/server/workspace/server/target/server.war \* -r

　　scp /root/.jenkins/jobs/server/workspace/server/target/server.war root@123.56.xxx.xx:/opt/war/

　　构建成功远程执行shell脚本

　　exec command 是远程sh的路径

[](http://www.51testing.com/batch.download.php?aid=79667)

[](http://www.51testing.com/batch.download.php?aid=79668)

**分享我的publish.sh文件**

|  |
| --- |
| 作用是备份每次上传的war包 重启Tomcat。  　　export JAVA\_HOME=/opt/software/jdk1.7.0\_25  　　TOMCAT\_HOME="/opt/software/apache-tomcat-7.0.59"  　　TOMCAT\_PORT=80  　　PROJECT="server"  　　BAK\_DIR=/opt/war/bak/$PROJECT/`date +%Y%m%d`  　　mkdir -p "${BAK\_DIR}"  　　cp /opt/war/"${PROJECT}".war "${BAK\_DIR}"/"${PROJECT}"\_`date +%Y%m%d%H%M%S`.war  　　#shutdown tomcat  　　/opt/sh/kill-tomcat-force.sh  　　#publish project  　　rm -rf "${TOMCAT\_HOME}"/webapps/${PROJECT}  　　cp /opt/war/"${PROJECT}".war "${TOMCAT\_HOME}"/webapps/${PROJECT}.war  　　#remove tmp  　　rm -rf /opt/war/${PROJECT}.war  　　#unzip war  　　unzip "${TOMCAT\_HOME}"/webapps/${PROJECT}.war -d "${TOMCAT\_HOME}"/webapps/${PROJECT}  　　rm -rf "${TOMCAT\_HOME}"/webapps/${PROJECT}.war  　　##copy lib  　　cp /opt/lib/\* "${TOMCAT\_HOME}"/webapps/${PROJECT}/WEB-INF/lib/  　　## start tomcat  　　sleep 3  　　#start tomcat  　　/opt/software/apache-tomcat-7.0.59/bin/startup.sh  　　echo "tomcat is starting!" |

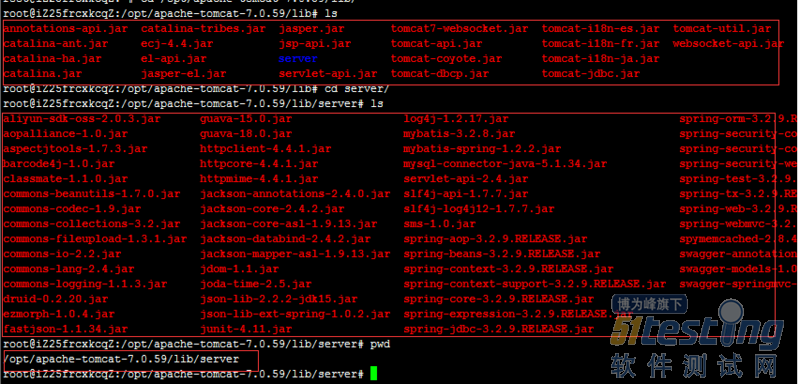
**分享我的kill-tomcat-force.sh文件**

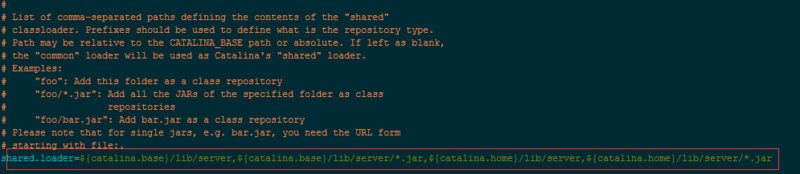
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 作用是强制关闭tomcat进程 | set fileformat=unix | path=/opt/software/apache-tomcat-7.0.59/bin | ps -ef|grep $path|grep tomcat|awk '{print $2}' | echo "exec $path/shutdown.sh" | $path/shutdown.sh | sleep 3s | #kill -9 pid | ps -ef|grep $path|grep tomcat|awk '{print $2}'|xargs kill -9 | #success msg | echo "shutdown success" | ps -ef|grep $path|grep java|awk '{print $2}' |

　　分享我的Tomcat精简方法

|  |
| --- |
| 在tomcat\_home/lib下新建自定义jar包文件，导入项目所需其他jar包(以后有新增的话，单独再导一次)  　　修改tomcat\_home/conf/catalina.properties 搜索=shared.loader加上路径  　　shared.loader=${catalina.base}/lib/server,${catalina.base}/lib/server/\*.jar,${catalina.home}/lib/server,${catalina.home}/lib/server/\*.jar |

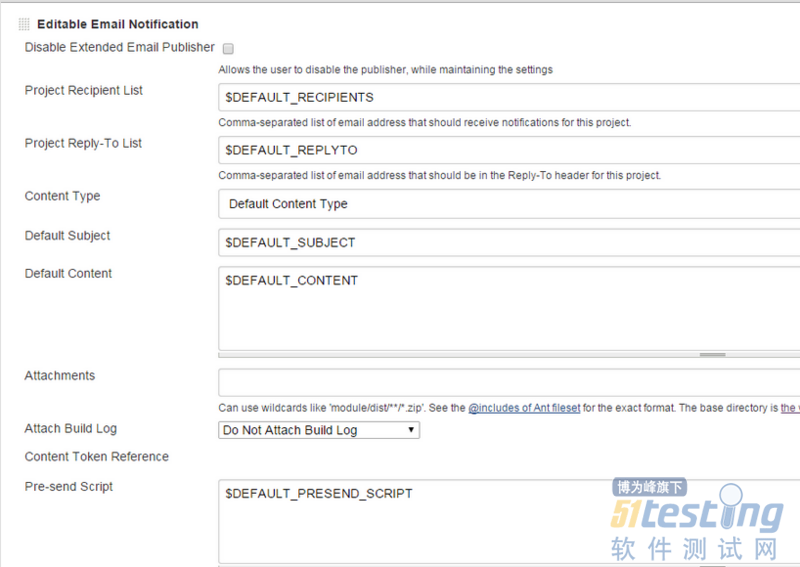
　　此时Tomcat运行前会加载server下的lib包，如果是多个项目公用一个tomcat的时候，就需要这里放公共的lib包，避免tomcat加载多余的jar包,消耗内存。

[](http://www.51testing.com/batch.download.php?aid=79669)

[](http://www.51testing.com/batch.download.php?aid=79670)

　　构建后邮件设置

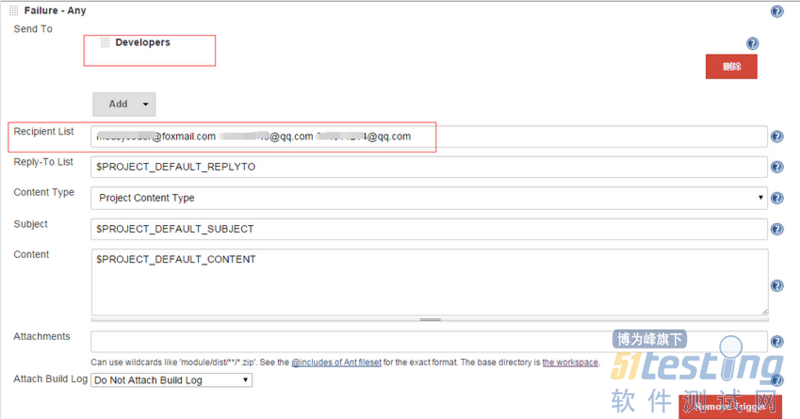
　　邮件主题收件人配置

[](http://www.51testing.com/batch.download.php?aid=79671)

　　邮件触发器

　　局部配置会覆盖掉全局配置,我们之前在全局配置里配置了构建失败邮件触发器,这里是更加精细的配置，

　　我们选择构建失败Failure-1st触发器，失败以后发邮件给开发者，(这里可以根据实际需要，配置，可以配置多个触发器)开发者的邮件在Recipient List里配置。

[](http://www.51testing.com/batch.download.php?aid=79672)

**Jenkins用户权限管理**

　　步骤：首页-> 系统管理-> Configure Global Security

　　基本配置:

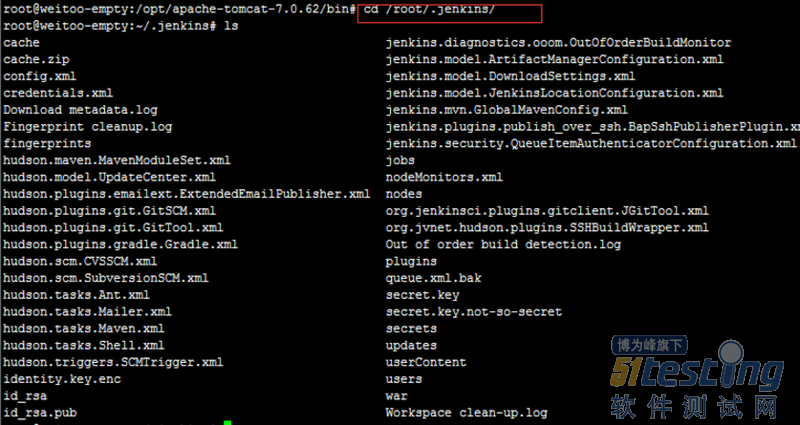
　　只有注册的用户才能操作,当然如果是大企业的话，可以采用项目矩阵授权策略,详情可以Google。

[](http://www.51testing.com/batch.download.php?aid=79673)

**Jenkins配置的备份和版本控制**

　　很多情况下稍不注意改变了Jenkins的配置，把平台弄坏了，又想去恢复，这个时候就得把Jenkins的配置文件进行配置或者版本化，只需要把/root/.jenkins/加入git版本库里即可,该目录下包含Jenkins所有信息,包括每次构建历史信息和历史jar包

　　进行全备份然后覆盖掉该文件夹的时候，重新构建JOB会出现文件夹已经存在等exception，只需要手动删掉这些目录即可，不会丢失数据。(这是Jenkins的一个bug,参考 JENKINS-21330)

[](http://www.51testing.com/batch.download.php?aid=79674)