

# jenkins+maven+tomcat+git

已测 : jenkins+maven+tomcat+git

git服务器---阿里云服务器centos7.4 -----ip:47.75.158.XX

jenkins服务器---阿里云服务器centos7.4-----ip:47.89.23.X

=====

测试环境:

部署git服务器:

安装git

```
[root@www ~]# yum install -y git
```

```
[root@www ~]# git config --global user.name 'tigerfive'
```

```
[root@www ~]# git config --global user.email 'tigerfive@aliyun.com'
```

```
[root@www ~]# useradd git ---创建git用户
```

```
[root@www ~]# passwd git 设置git用户密码
```

Changing password for user git.

New password:1

BAD PASSWORD: The password is a palindrome

Retype new password:1

passwd: all authentication tokens updated successfully.

创建git本地仓库:

```
[root@www ~]# mkdir /git-root
```

```
[root@www ~]# cd /git-root
```

```
[root@www git-root]# git init --bare test.git
```

```
[root@www git-root]# chown -R git.git . -----创建的哪个用户修改为哪个
```

```
[root@www ~]$ ssh-keygen
```

```
[root@www ~]$ ssh-copy-id -i username@47.89.23.X ---git客户端ip
```

=====

jenkins用户如下操作:

```
[root@www ~]# useradd jenkins
```

```
[root@www ~]# passwd jenkins
```

```
[root@www ~]# chown jenkins.jenkins . -R
```

```
[root@www ~]$ ssh-keygen
```

```
[root@www ~]$ ssh-copy-id -i jenkins@47.89.23.23
```

=====

下载测试源码包:

```
[root@www ~]# cd /git-root/
```

```
[root@www git-root]# git clone https://github.com/jenkins-docs/simple-java-maven-app.git
```

```
[root@www git-root]# ls
easy-springmvc-maven  shell.git  simple-java-maven-app  test2.git  test.git
```

=====

在jenkins服务器上面:

jenkins服务器使用root用户clone代码如下:

安装git作为客户端:

```
[root@xuan ~]# yum install -y git
```

```
[root@xuan ~]# useradd git
```

```
[root@xuan ~]# passwd git
```

Changing password for user git.

New password:1

BAD PASSWORD: The password is a palindrome

Retype new password:1

passwd: all authentication tokens updated successfully.

在root下:

```
[root@xuan ~]# ssh-keygen
```

```
[root@xuan ~]# ssh-copy-id -i git@47.75.158.102 ---git服务端ip
```

这一步可做可不做

```
[root@xuan ~]# su - git
```

```
[git@xuan ~]$ ssh-keygen
[git@xuan ~]$ ssh-copy-id -i git@47.75.158.102 ----git服务器端ip
如果运行git是jenkins用户:
在root下:
[root@xuan ~]# ssh-keygen
[root@xuan ~]# ssh-copy-id -i jenkins@47.75.158.102 ---git服务端ip
```

=====

部署java环境:

安装jdk:1.8

```
[root@xuan ~]# wget --no-check-certificate --no-cookies --header "Cookie: oraclelicense=accept-securebackup-cookie" 'http://download.oracle.com/otn-pub/java/jdk/8u171-b11/512cd62ec5174c3487ac17c61aaa89e8/jdk-8u171-linux-x64.tar.gz'
```

新连接:

```
[root@xuan ~]# wget --no-check-certificate --no-cookies --header "Cookie: oraclelicense=accept-securebackup-cookie" 'http://download.oracle.com/otn-pub/java/jdk/8u181-b13/96a7b8442fe848ef90c96a2fad6ed6d1/jdk-8u181-linux-x64.tar.gz'
```

```
wget https://download.oracle.com/otn/java/jdk/8u211-b12/478a62b7d4e34b78b671c754eaaf38ab/jdk-8u211-linux-x64.tar.gz?AuthParam=1556463153_45b09804abbea475138af618423dae3c
```

```
[root@xuan ~]# tar xzf jdk-8u171-linux-x64.tar.gz -C /usr/local/
[root@xuan ~]# cd /usr/local/
[root@xuan local]# mv jdk1.8.0_171/ java
```

安装tomcat:8.0

```
[root@xuan ~]# wget http://mirrors.shu.edu.cn/apache/tomcat/tomcat-8/v8.0.53/bin/apache-tomcat-8.0.53.tar.gz
wget https://archive.apache.org/dist/tomcat/tomcat-8/v8.0.53/bin/apache-tomcat-8.0.53.tar.gz
```

```
[root@xuan ~]# tar xzf apache-tomcat-8.0.52.tar.gz -C /usr/local/
[root@xuan ~]# cd /usr/local/
[root@xuan local]# mv apache-tomcat-8.0.52/ tomcat
```

安装maven:3.5.4

```
[root@xuan ~]# wget http://mirrors.tuna.tsinghua.edu.cn/apache/maven/maven-3/3.5.4/binaries/apache-maven-3.5.4-bin.tar.gz
[root@xuan ~]# tar xzf apache-maven-3.5.3-bin.tar.gz -C /usr/local/java/
[root@xuan ~]# cd /usr/local/java/
[root@xuan java]# mv apache-maven-3.5.3/ maven
```

设置环境变量 ;

```
[root@xuan ~]# vim /etc/profile
JAVA_HOME=/usr/local/java
JRE_HOME=/usr/local/java
MAVEN_HOME=/usr/local/java/maven
PATH=$PATH:$JAVA_HOME/bin:$MAVEN_HOME/bin
export PATH USER LOGNAME MAIL HOSTNAME HISTSIZE HISTCONTROL JAVA_HOME MAVEN_HOME
export JENKINS_HOME=/opt/jenkins --设置jenkins的主目录，最后的war包会存放在这里。
```

```
[root@xuan ~]# source /etc/profile
```

测试:

```
[root@xuan ~]# java -version
'java version "1.8.0_171"
```

```
[root@xuan ~]# mvn -v
Apache Maven 3.5.3 (3383c37e1f9e9b3bc3df5050c29c8aff9f295297; 2018-02-25T03:49:05+08:00)
```

安装jenkins:2.129 ----通过官网直接下载war包。

官网:<http://updates.jenkins-ci.org/download/war/>

```
[root@xuan ~]# wget http://updates.jenkins-ci.org/download/war/2.129/jenkins.war
```

将war包拷贝到tomcat的发布网站目录:

```
[root@xuan ~]# cp jenkins.war /usr/local/tomcat/webapps/
```

启动tomcat:

```
[root@xuan ~]# cd /usr/local/tomcat/bin/
```

```
[root@xuan bin]# ./startup.sh
```

```
[root@xuan bin]# ./shutdown.sh ---关闭
```

测试:

<http://47.89.23.X:8080/jenkins>



jenkins安装成功

根据提示获取密码并输入:

```
[root@xuan ~]# cat /root/.jenkins/secrets/initialAdminPassword
```

51bd5b1ac6ed4792862abc3b0679ebc5 ---复制这个密码粘贴到上面提示的空格里面。

添加插件:

# 自定义Jenkins

插件通过附加特性来扩展Jenkins以满足不同的需求。

## 安装推荐的插件

安装Jenkins社区推荐的插件。

## 选择插件来安装

选择并安装最适合的插件。



等待安装:

# 新手入门

✓ Folders	OWASP Markup Formatter	Build Timeout	Credentials Binding	** JIK Tool ** Script Security ** Command Agent Launcher Folders ** bouncycastle API
Timestampers	Workspace Cleanup	Ant	Gradle	
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View	
Git	Subversion	SSH Slaves	Matrix Authorization Strategy	
PAM Authentication	LDAP	Email Extension	Mailer	

创建一个管理员账户；

## 新手入门

# 创建第一个管理员用户

用户名:

admin

密码:

.....

确认密码:

.....

全名:

电子邮件地址:

@qq.com

## 实例配置

Jenkins URL:

<http://47.89.23.23:8080/jenkins/>

Jenkins URL 用于给各种Jenkins资源提供绝对路径链接的根地址。这意味着对于很多Jenkins特色是需要正确设置的，例如：邮件通知、PR状态更新以及提供给构建步骤的BUILD\_URL环境变量。

推荐的默认值显示在尚未保存，如果可能的话这是根据当前请求生成的。最佳实践是要设置这个值，用户可能会需要用到。这将会避免在分享或者查看链接时的困惑。

## 新手入门

# Jenkins已就绪！

Jenkins安装已完成。

开始使用Jenkins



← → ×

47.89.23.23:8080/jenkins/



# Jenkins

Jenkins

 新建任务

 用户

 构建历史

 系统管理

 我的视图

 凭据

 新建视图

构建队列

队列中没有构建任务

构建执行状态

1 空闲  
2 空闲

## 欢迎使用Jenkins!

开始 **创建一个新任务**

开始安装所需插件:

新建任务

用户

构建历史

 系统管理

我的视图

凭据

新建视图

构建队列

队列中没有构建任务

构建执行状态

1 空闲  
2 空闲

## 管理Jenkins

反向代理设置有误

 **系统设置**  
全局设置和路径

 **全局安全配置**  
Jenkins安全, 定义谁可以访问或使用系统。

 **凭据配置**  
配置凭据的提供者和类型

 **全局工具配置**  
工具配置, 包括它们的位置和自动安装器

 **读取设置**  
放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件时重新读取

 **插件管理**  
添加、删除、禁用或启用Jenkins功能扩展插件。

所需的插件:

- Maven插件 Maven Integration plugin

- 发布插件 Deploy to container Plugin

安装插件Deploy to container ---支持自动化代码部署到tomcat容器

GIT pligin 可能已经安装

Maven Integration :jenkins利用Maven编译，打包所需插件

Publish Over SSH :通过ssh连接

Ansible ----调用ansible命令

- Gitlab插件：GitLab Plugin 和 Gitlab Hook Plugin

安装过程:

系统管理--->插件管理---->可选插件--->过滤Deploy to container---->勾选--->直接安装

 返回到工作台

 系统管理



1

2

3

4

返回到工作台

系统管理

可更新 可选插件 已安装 高级

安装 ↓


	名称	版本
<input checked="" type="checkbox"/>	Deploy to container	1.13

直接安装 下载待重启后安装 1 小时 28 分 之前获取了更新信息 立即获取



 返回

 系统管理


 插件管理


继续安装


## 安装/更新 插件中

准备

- Checking internet connectivity
- Checking update center connectivity
- Success

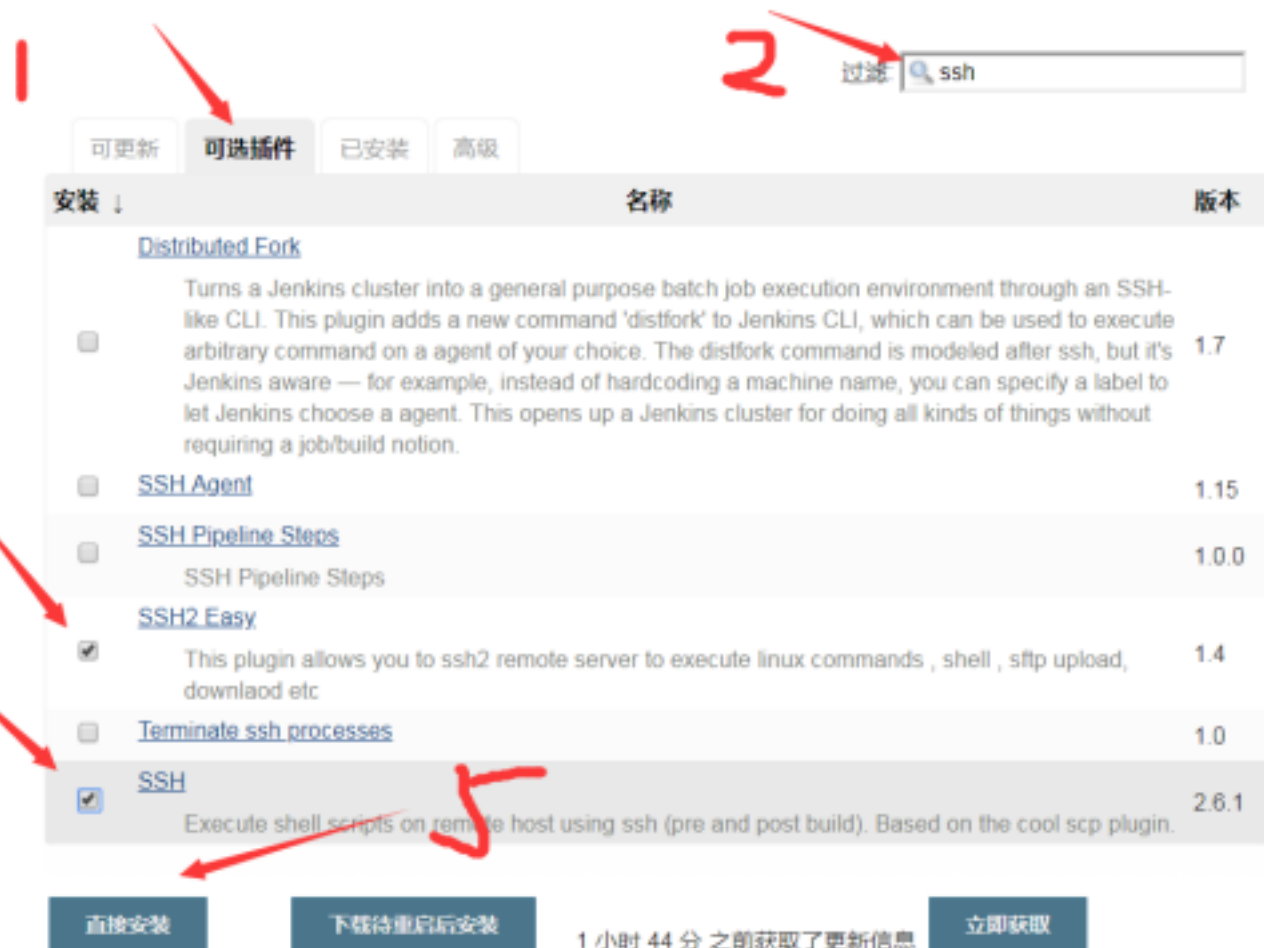
Deploy to container  完成

 [返回首页](#)  
(返回首页使用已经安装好的插件)

 ☐ 安装完成后重启Jenkins(空闲时)

可以不用

以上的插件安装步骤同第一个安装一样。出上面的插件还需要安装一个插件如下：



在安装一个插件ansible



插件目前安装完毕，如果需要用到gitlab，需要在百度一下了。

## Jenkins系统设置

注意：这里没有强调的都设置为默认即可  
插件已经基本准备好了,下面还要再做一些基础配置

要配置的有：

jdk maven 和git

系统管理->Global Tool Configuration,配置jdk,git,maven的根目录

- 用户
- 构建历史
- 系统管理
- 我的视图
- 凭据
- 新建视图

## 管理Jenkins

反向代理设置有误

更多信息

不再显示



### 系统设置

全局设置和路径



### 全局安全配置

Jenkins安全，定义谁可以访问或使用系统。



### 凭据配置

配置凭据的提供者和类型



### 全局工具配置

工具配置，包括它们的位置和自动安装器



### 读取设置

放弃当前内存中所有的设置信息并从配置文件中重新读取。仅用于当您手动修改配置文件时重新读取设置。



### 插件管理

添加、删除、禁用或启用Jenkins功能扩展插件。

## JDK

### JDK 安装

新增 JDK

JDK

别名

java

JAVA\_HOME

/usr/local/java

☐ 自动安装

删除 JDK

新增 JDK

系统下JDK 安装列表

## Git

### Git installations

Git

Name

Default

Path to Git executable

/usr/bin/git

☐ 自动安装

Delete Git

Add Git

这里选择默认就行，直接用yum安装的git1.8版本的如果是编译安装请把git的环境变量设置到/etc/profile这个文件里面

Maven

Maven 安装

新增 Maven

Maven

Name

maven

MAVEN\_HOME

/usr/local/java/maven

☐ 自动安装

记得把自动安装取消

删除 Maven

新增 Maven

系统下Maven 安装列表

Ansible

Ansible 安装

新增 Ansible

Ansible

Name

ansible

☒ 自动安装

新增安装

删除 Ansible

新增 Ansible


系统下Ansible 安装列表

在最下面点击save保存

Save

Apply

创建一个任务进行测试；

 新建任务 用户 构建历史 系统管理 我的视图 凭据 新建视图

构建队列



队列中没有构建任务

构建执行状态



1 空闲

## 管理Jenkins

反向代理设置有误



系统设置

全局设置和路径



全局安全配置

Jenkins安全，定义谁可以访问或使用系统。



凭据配置

配置凭据的提供者和类型

## 输入一个任务名称

test

» 必填项



### 构建一个自由风格的软件项目

这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目,甚至可以构建软件以外的系统.



### 构建一个maven项目

构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻构建配置.



### 流水线

精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线（更加正式地应当称为工作流），增加或者组织难型。



### 构建一个多配置项目

适用于多配置项目,例如多环境测试,平台指定构建,等等.



### 文件夹

创建一个可以嵌套存储的容器。利用它可以进行分组。视图仅仅是一个过滤器，而文件夹则是一个独立的命名空间，因此的内容，只要它们在不同的文件夹里即可。



### GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.



### Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

确定

开始配置:

**General** 源码管理 构建触发器 构建环境 Pre Steps Build Post Steps 构建设置 构建后操作

描述

[纯文本] 预览 隐藏预览

☐ GitHub project

☐ Throttle builds

☒ 丢弃旧的构建

策略 Log Rotation

保持构建的天数 5

如果非空，构建记录将保存此天数 可以根据实际情况填写数量

保持构建的最大个数 10

如果非空，最多此数目的构建记录将被保存

高级...

**源码管理**

☐ 无

☒ Git

Repositories

Repository URL git@47.75.158.102:/git-root/test.git

Credentials git Add

选择add点击jenkins添加一个用户连接的凭证

客户端的git用户，jenkins通过那个用户去下载上传代码

Branches to build

Branch Specifier (blank for 'any') \*/master

看开发的创建的分支，最后用那个分支

源码库浏览器 (自动)

Additional Behaviours 新增

☐ Subversion

这个图片就是点击add之后的，



## 添加凭据

Domain

类型

范围

Username  git 客户端的git用户或者其它用户，就是用那个用户从git服务器上下载代码的用户

Private Key ☒ Enter directly git客户端用户的秘钥

Key

Passphrase

ID

描述

## Build

Root POM

Goals and options

## Post Steps

☐ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☒ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

注意:

扩展:

**maven**命令小结

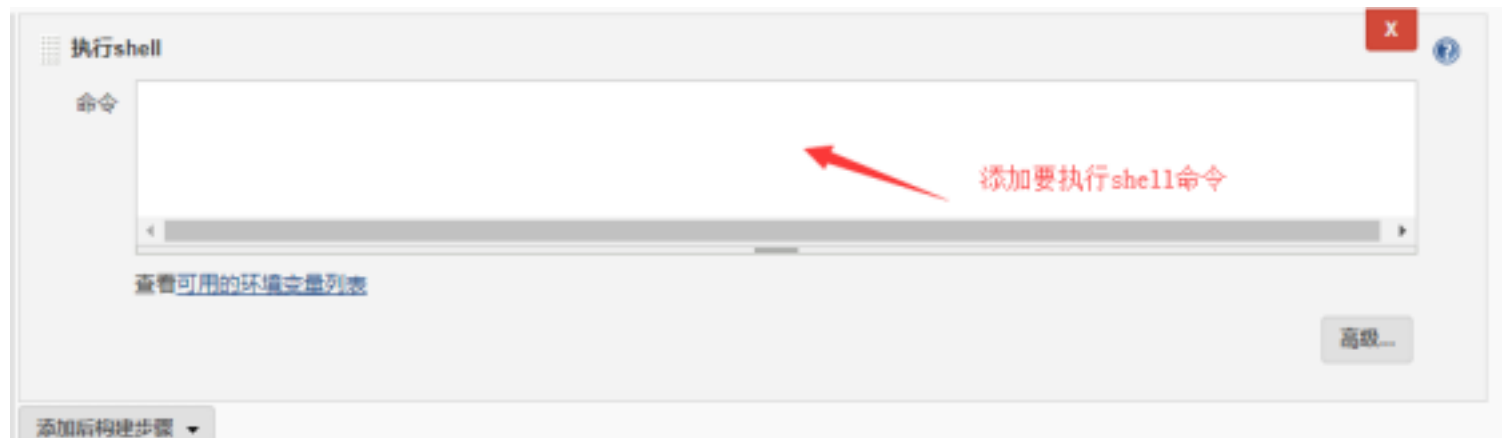
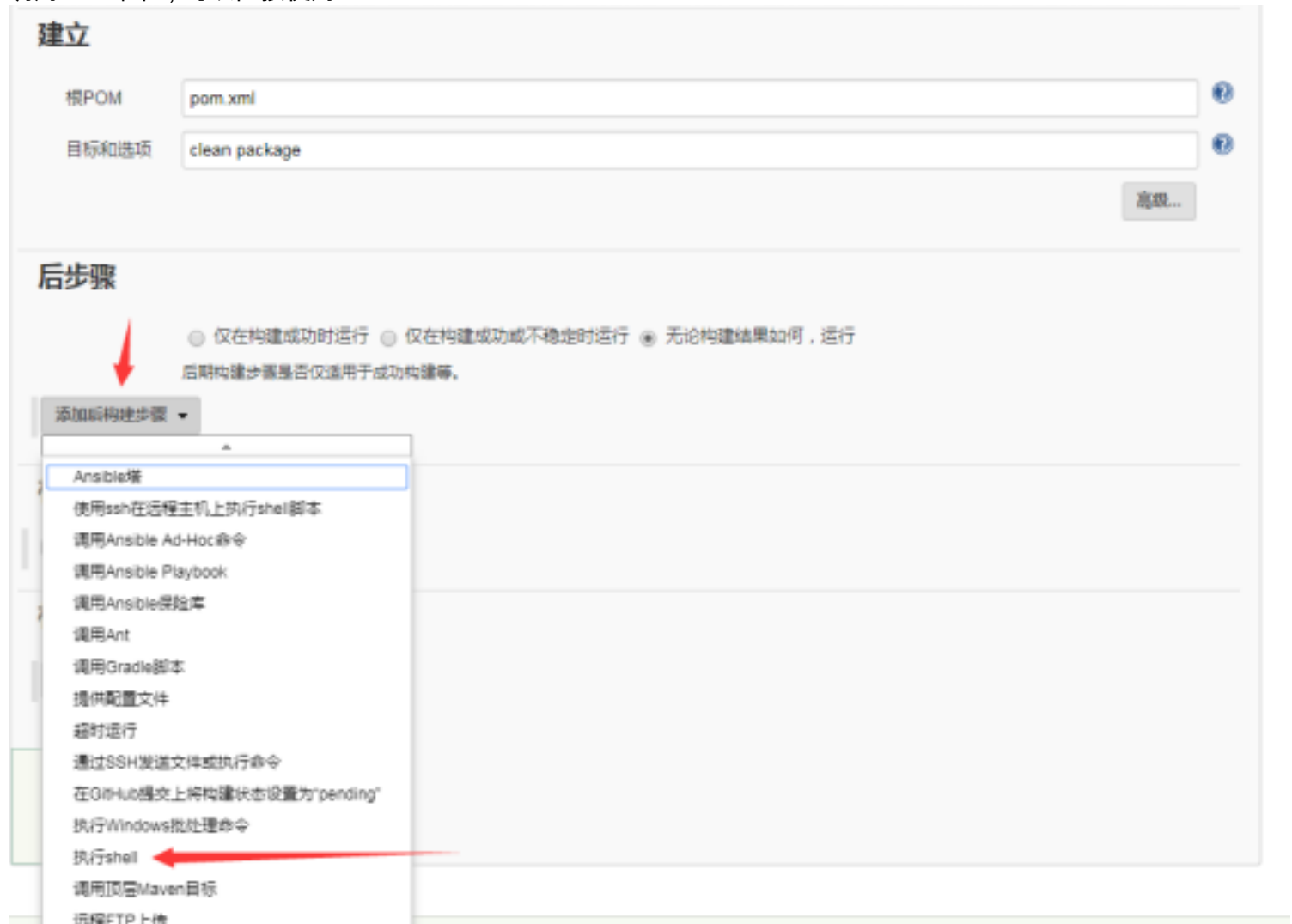
mvn compile 命令会根据pom.xml 中定义的dependencies 依赖，去maven 中心下载相关的包并进行编译，将编译后



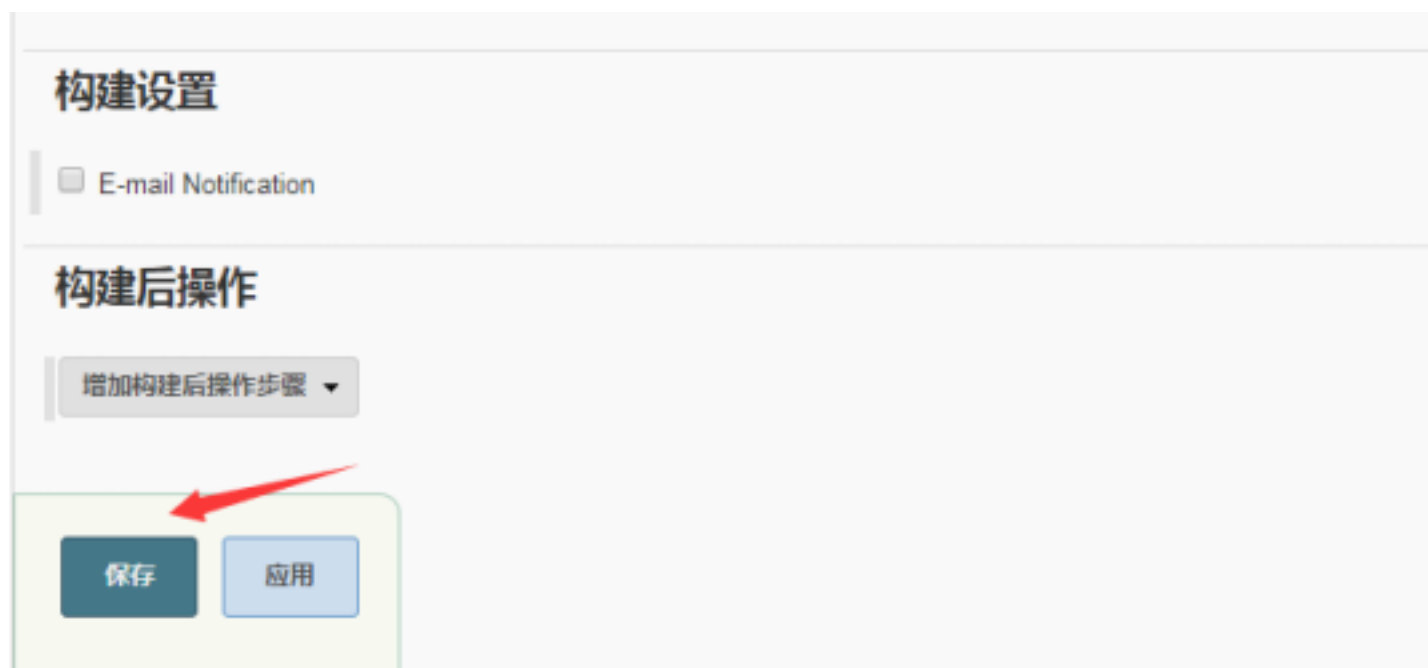
的文件放在target/classes/目录中；  
mvn test 命令会根据test目录中定义的测试文件对类进行编译测试，并把生成的测试报告存放在target/surefire-reports/目录中；  
mvn clean 命令清除target 目录  
mvn package 命令生成相关的jar包存放在target目录中  
mvn install 命令将生成的\*.jar包复制到本地库中(~/.m2/repository/)

---

调用shell命令；可以直接使用ansible



点击保存:



选择立即构建:



成功输出:



控制台部分输出：

```
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.24/plexus-utils-3.0.24.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.24/plexus-utils-3.0.24.jar (200 kB at 101 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/1.11/plexus-compiler-javac-1.11.jar (428 kB at 201 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/1.11/plexus-compiler-javac-1.11.jar (428 kB at 201 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/1.11/plexus-compiler-javac-1.11.jar (428 kB at 201 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/1.11/plexus-compiler-javac-1.11.jar (428 kB at 201 kB/s)
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-compiler-javac/1.11/plexus-compiler-javac-1.11.jar (428 kB at 201 kB/s)
[INFO] Building jar: /opt/jenkins/workspace/app/target/app-1.0-SNAPSHOT.jar
[WARNING] Attempt to (de-)serialize anonymous class hudson.maven.reporters.MavenArtifactArchiver$2; see: https://jenkins.io/redirect/serialization-of-anonymous-classes/
[WARNING] Attempt to (de-)serialize anonymous class hudson.maven.reporters.MavenFingerprint$1; see: https://jenkins.io/redirect/serialization-of-anonymous-classes/
[INFO] BUILD SUCCESS
[INFO] Total time: 01:30 min
[INFO] Finished at: 2018-07-02T20:48:50+08:00
[INFO]
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /opt/jenkins/workspace/app/pom.xml to com.mycompany.app/app-1.0-SNAPSHOT/pom
[JENKINS] Archiving /opt/jenkins/workspace/app/target/app-1.0-SNAPSHOT.jar to com.mycompany.app/app-1.0-SNAPSHOT/app-1.0-SNAPSHOT.jar
Finished: SUCCESS
```

修改jenkins主目录：可以定义通过jenkins+maven打包之后war包存放的位置





修改前先把jenkins关闭，修改之后重新启动  
修改方式；



修改后



修改之后需要重新初始化。从第一部安装插件开始。建议提前将环境变量写入/etc/profile文件