

# Jenkins 与 GitLab 的自动化构建之旅

## 1 Gitlab 的安装及仓库创建

### 1.1 下载 Gitlab 安装包

1).官网下载速度较慢 建议先行下载

国内的源里面可以找到最新的版本，[请单击这里查看](#)。

2).安装依赖

```
sudo apt-get install curl openssh-server ca-certificates postfix
```

3).配置 postfix 邮箱



选择 Internet Site ( F12 ) Enter 下一步

## Postfix Configuration

The "mail name" is the domain name used to "qualify" \_ALL\_ mail addresses without a domain name. This includes mail to and from <root>: please do not make your machine send out mail from root@example.org unless root@example.org has told you to.

This name will also be used by other programs. It should be the single, fully qualified domain name (FQDN).

Thus, if a mail address on the local host is foo@example.org, the correct value for this option would be example.org.

System mail name:

neil-vm

<Ok>

<Cancel>

这里设置FQDN 使用默认即可。

## 1.2 安装 Gitlab

在终端执行：sudo dpkg -i gitlab-ce\_9.5.4ce.0\_amd64.deb 进行安装。

```
gavin@gavin:~/下载$
gavin@gavin:~/下载$
gavin@gavin:~/下载$ sudo dpkg -i gitlab-ce_9.5.4-ce.0_amd64.deb
[sudo] gavin 的密码:
对不起，请重试。
[sudo] gavin 的密码:
正在选中未选择的软件包 gitlab-ce。
(正在读取数据库 ... 系统当前共安装有 254911 个文件和目录。)
正准备解包 gitlab-ce_9.5.4-ce.0_amd64.deb ...
正在解包 gitlab-ce (9.5.4-ce.0) ...
```

出现 It looks like... 表示安装成功！

## 1.3 安装 Git 工具

```
gavin@gavin:/opt/share$ sudo apt-get install git
[sudo] gavin 的密码:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
```

## 1.4 生成密钥文件

使用 ssh-keygen 生成密钥文件 .ssh/id\_rsa.pub。

```

gavin@gavin:~/.ssh$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/gavin/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/gavin/.ssh/id_rsa
Your public key has been saved in /home/gavin/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:yxARDhTv4YL20xLAGsKHT1v16Up3SDFW2m2ToRt3VAo gavin@gavin
The key's randomart image is:
+---[RSA 2048]---+
|.O=.O. ..E. .O|
|.O.O.= O +O O.+|
|.OO. B O.++ *.|
|.. = * + = O|
| = O S ..|
| O . * = .|
| . = .|
|_|
+-----[SHA256]-----+
gavin@gavin:~/.ssh$

```

生成秘钥

生成的私钥

生成的公钥

```

gavin@gavin:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDBwML7x2avFAVp/oXI0lXNmfgGJawSTMNqgeS5Ng0pl/Z1YctkSioX0E9nmtMN1FPp6mrMBX
jZ3IjY/EpCaGViGefSypCvscOoFXemD2pvPPkQyH7T6mdablctI4iZ2MwKPZTv9eAYP/voq9z+NuwWCRPvM55VOXuD6JLqz7DQ3Vuf3JtVaTC2
xNmmAMD2M8EUGvgB/H/pTHKaw3 gavin@gavin
gavin@gavin:~$ gedit .ssh/id_rsa.pub

```

这里生成的两个秘钥很重要，会在后面 Gitlab 的仓库配置与 Jenkins 的构建免密连接时候用到。

## 2 GitLab 简单配置及项目新建

### 2.1 配置 Gitlab

这一步在官方的文档里面没有，但是如果如果没有配置的话，直接启动 GitLab，会出现不正确的 FQDN 错误，导致无法正常启动。因此必须做配置。

```
sudo gedit /etc/gitlab/gitlab.rb
```

把 external\_url 改成部署机器的域名或者IP地址。

```

8
9 ## GitLab URL
10 ##! URL on which GitLab will be reachable.
11 ##! For more details on configuring external_url see:
12 ##! https://docs.gitlab.com/omnibus/settings/configuration.html#configuring-the-external-url-for-gitlab
13 external_url 'http://gavin'
14 external_url 'http://192.168.0.122'
15 ## Legend
16 ##! The following notations at the beginning of each line may be used to
17 ##! differentiate between components of this file and to easily select them using
18 ##! a regex.

```

然后对 GitLab 进行重配置（这一步也是启动 GitLab）：

```
sudo gitlab-ctl reconfigure
```

查看启动状态：

```
sudo gitlab-ctl status
```

```
gavin@gavin:~$  
gavin@gavin:~$  
gavin@gavin:~$ sudo gitlab-ctl status  
[sudo] gavin 的密码:  
run: postgresql: (pid 10413) 57s; run: log: (pid 10412) 57s  
run: redis: (pid 10335) 63s; run: log: (pid 10334) 63s  
gavin@gavin:~$
```

在浏览器的地址栏中输入服务器的公网 IP 即可登录 GitLab 的界面，第一次登录使用的用户名和密码为 root 和 5iveL!fe。



Your password has been changed successfully.

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Sign in

Register

Username or email

root

Password

••••••••

☐ Remember me

[Forgot your password?](#)

Sign in

Didn't receive a confirmation email? [Request a new one.](#)

首次登录会强制用户修改密码。密码修改成功后，输入新密码进行登录。

## 2.2 Gitlab 项目新建

在 GitLab 的主页中新建一个 Project：

Projects

Search

Customize your experience

Change syntax themes, default project pages, and more in preferences.

Check it out

Your projects

Starred projects

Explore projects

Filter by name...

Last updated

New Project

T Administrator / test

★ 0

192.168.0.122/projects/new 140% 搜索

Projects Search

http://192.168.0.122/ root 用户组 test 仓库名

Want to house several dependent projects under the same namespace? [Create a group](#)

**Project description (optional)**

Description format

**Visibility Level** 仓库权限

- ☒ **Private**  
Project access must be granted explicitly to each user.
- ☐ **Internal**  
The project can be accessed by any logged in user.
- ☐ **Public**  
The project can be accessed without any authentication.

Create project 创建

添加 ssh key 导入步骤2中生成的密钥文件内容（密钥前面1.4节已生成）：

User Settings Search

Profile Account Applications Chat Access Tokens Emails Password Notifications **SSH Keys** Preferences Audit Log

**SSH Keys**

SSH keys allow you to establish a secure connection between your computer and GitLab.

**Add an SSH key**

Before you can add an SSH key you need to [generate it](#).

**Key**

```
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAQEAsoobnasjfd+/zwqRINin3aSXET90S3L1499jghG/NKR82169o7GRivgWoLTV1u62NQr1GpCJ46t83TPHz1k3uqCuiCqllh15ovIG65A0cJLCSn/UlPcfQ6TEEC/QMGdJUOTofE4QmoxRce13n2SxOS87TRQ6+IE6bn+CK0GBDy9qjCVC962LPuNVjH8DF3F5ai7mtUMELyYcr9emkxhn2YgIZvLsDLjkdXjFBjbi2csgW5C/CtpW4tp/pDjRMM4AKBSKxwsodRs3RTTgoXtMpuUc+MnNvuGjV/GOKi7Uijp5Y7g99aUdPoMgZugKa9nKSchu68w==root@iZbp1h7fx16gkr9u4gk8v3Z
```

**Title**

root@iZbp1h7fx16gkr9u4gk8v3Z

Add key

Your SSH keys (0)

There are no SSH keys with access to your account.

ssh key 添加完成：

User Settings Search

Profile Account Applications Chat Access Tokens Emails Password Notifications **SSH Keys** Preferences Audit Log

**SSH Key**

Title: root@iZbp1h7fx16gkr9u4gk8v3Z

Created on: Apr 20, 2017 1:52pm

Last used on: N/A

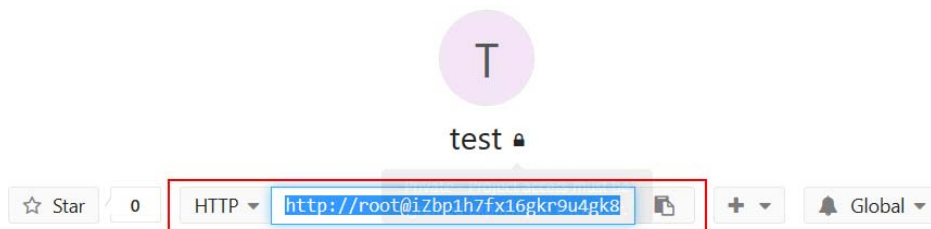
Fingerprint: 34:ac:e7:de:34:0d:7b:b6:69:72:1d:00:3e:50:04:4c

```
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAQEAsoobnasjfd+/zwqRINin3aSXET90S3L1499jghG/NKR82169o7GRivgWoLTV1u62NQr1GpCJ46t83TPHz1k3uqCuiCqllh15ovIG65A0cJLCSn/UlPcfQ6TEEC/QMGdJUOTofE4QmoxRce13n2SxOS87TRQ6+IE6bn+CK0GBDy9qjCVC962LPuNVjH8DF3F5ai7mtUMELyYcr9emkxhn2YgIZvLsDLjkdXjFBjbi2csgW5C/CtpW4tp/pDjRMM4AKBSKxwsodRs3RTTgoXtMpuUc+MnNvuGjV/GOKi7Uijp5Y7g99aUdPoMgZugKa9nKSchu68w==
```

Remove

项目地址，该地址在进行 clone 操作时需要用到：





## 2.3 代码上传

克隆项目，在本地生成同名目录，并且目录中会有所有的项目文件

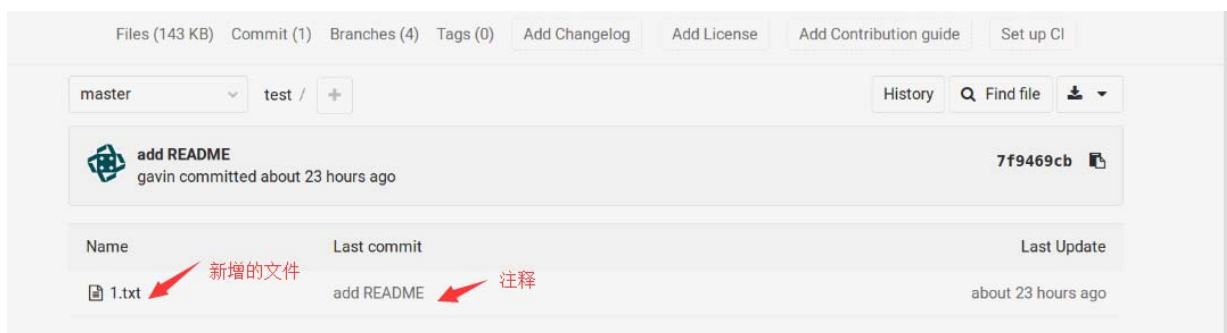
git clone git@192.168.0.122:gavin/test.git :

```
gavin@gavin:/opt/share$ git clone git@192.168.0.122:gavin/test.git
正克隆到 'test'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
接收对象中: 100% (3/3), 完成.
检查连接... 完成.
gavin@gavin:/opt/share$
```

进入到项目目录，拷贝自己的项目文件到此目录上传：

```
cd test/
cp -rf 自己项目路径/* .
git add .
git commit -m "add README" #将代码提交到本地仓库
git push -u origin master #将文件同步到GitLab服务器上
```

在网页中查看上传的文件已经同步到 GitLab 中：



## 3 Jenkins 安装与配置

### 3.1 Java 环境配置

Jenkins 基于 Java，Linux 下安装 Java 只要配置 Java 环境变量即可。

首先，解压java到相应目录，我一般习惯把安装的软件放到目录/usr/local下。

```

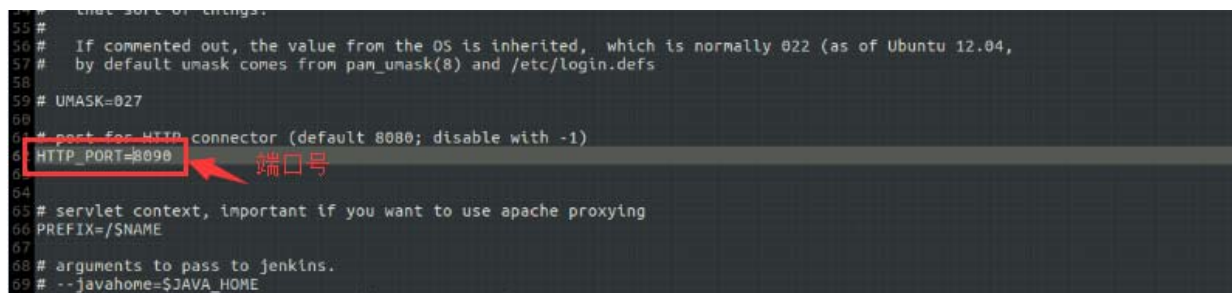
javin@javin:~$ ps -ef | grep jenkins
jenkins 3288 1 0 17:38 ? 00:00:08 /lib/systemd/systemd --user
jenkins 3289 3288 0 17:38 ? 00:00:00 [sd-pam]
jenkins 3290 3288 0 17:38 ? 00:00:08 /usr/sbin/dmccore -name=jenkins --libert --env=JENKINS_HOME=/var/lib/jenkins --socket=/var/log/jenkins/jenkins.log --gliffle=/var/run/jenkins/jenkins.pid --jars/lib/jenkins-0.jar --webroot=/var/cache/jenkins --httpport=8088
jenkins 3290 3288 0 17:38 ? 00:00:12 /usr/bin/java -Djava.net.useSystemProxies=true -jar /usr/share/jenkins/jenkins.war --webroot=/var/cache/jenkins --httpport=8088
javin 5400 25730 0 17:11 pts/22 00:00:06 grep --color=auto jenkins
javin@javin:~$

```

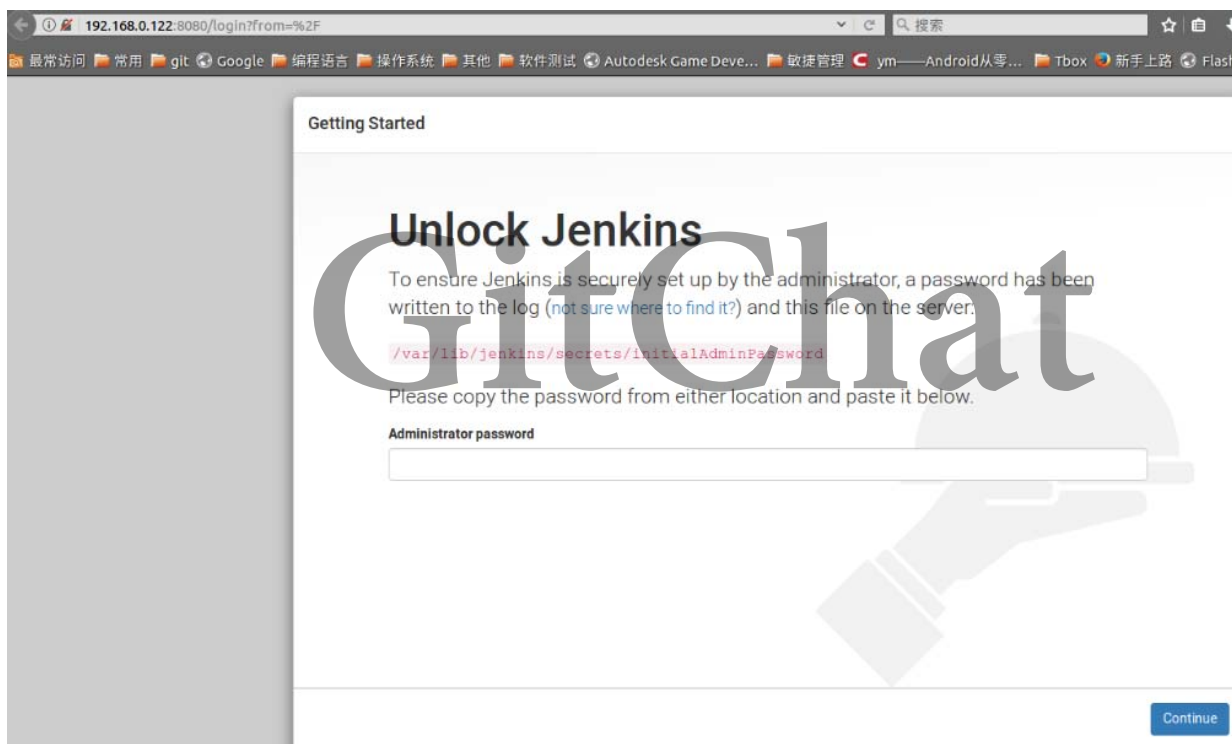
### 3.3 Jenkins 配置

上面只是安装完成了 Jenkins，还需要进行一些配置才可以。  
在这个系统端口中，8080已经在使用中了。所以在 /etc/default/jenkins.修改 Jenkins 默认端口设置：

```
gavin@gavin:~$ gedit /etc/default/jenkins
```



修改默认端口为 HTTP\_PORT=8090，这时通过浏览器就可以访问 Jenkins 了。比如我的地址：<http://192.168.0.122:8090/>



可以看到提示，为了确保 Jenkins 的安全，将管理员的密码写入文件，需要复制到下面的文本框做验证。

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

然后把输出的内容复制到上面密码框处。

然后，到了选择插件的界面，通过附加功能扩展 Jenkins 可以支持许多不同的需求。



# Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

## Install suggested plugins

Install plugins the Jenkins community finds most useful.





















## Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.60.3

## Getting Started

# Getting Started

 Folders Plugin	 OWASP Markup Formatter Plugin	 build timeout plugin	 Credentials Binding Plugin	<b>**</b> bouncycastle API Plugin
 Timestamper	 Workspace Cleanup Plugin	 Ant Plugin	 Gradle Plugin	
 Pipeline	 GitHub Branch Source Plugin	 Pipeline: GitHub Groovy Libraries	 Pipeline: Stage View Plugin	
 Git plugin	 Subversion Plug-in	 SSH Slaves plugin	 Matrix Authorization Strategy Plugin	
 PAM Authentication plugin	 LDAP Plugin	 Email Extension Plugin	 Mailer Plugin	

**\*\*** - required dependency

Jenkins 2.60.3

# Getting Started

✓ Folders Plugin	✓ OWASP Markup Formatter Plugin	⚙ build timeout plugin	⚙ Credentials Binding Plugin	** bouncycastle API Plugin Folders Plugin ** Structs Plugin ** JUnit Plugin OWASP Markup Formatter Plugin PAM Authentication plugin ** Windows Slaves Plugin ** Display URL API Jenkins Mailer Plugin LDAP Plugin ** Pipeline: Step API
⚙ Timestampers	⚙ Workspace Cleanup Plugin	⚙ Ant Plugin	⚙ Gradle Plugin	
⚙ Pipeline	⚙ GitHub Branch Source Plugin	⚙ Pipeline: GitHub Groovy Libraries	⚙ Pipeline: Stage View Plugin	
⚙ Git plugin	⚙ Subversion Plug-in	⚙ SSH Slaves plugin	⚙ Matrix Authorization Strategy Plugin	
✓ PAM Authentication plugin	✓ LDAP Plugin	⚙ Email Extension Plugin	✓ Mailer Plugin	
				** - required dependency

Jenkins 2.60.3

插件安装完成，就到了创建用户的界面，这里可以创建一个 Jenkins 用户。

Getting Started

Create First Admin User

用户名:

密码:

确认密码:

全名:

电子邮件地址:

Jenkins 2.60.3

Continue as admin

Save and Finish

到这里，基本配置就完成了。

# Jenkins is ready!

Your Jenkins setup is complete.

Start using Jenkins

Jenkins 2.60.3



如果在后续使用中，有插件需要安装，通过在已运行的 Jenkins 主页中，点击左侧的系统管理—>管理插件进入如下界面搜索安装：

更新 **可选插件** 已安装 高级 → 1. 选择可选插件

安装	名称	版本
Artifact Uploaders		
<input type="checkbox"/>	<a href="#">Appaloosa Plugin</a> Publish your mobile applications (Android, iOS, ...) to the <a href="#">appaloosa-store.com</a> platform.	1.4.0
<input type="checkbox"/>	<a href="#">Artifactory Plugin</a> This plugin allows deploying Maven 2, Maven 3, Ivy and Gradle artifacts and build info to the Artifactory artifacts manager.	2.1.7
<input type="checkbox"/>	<a href="#">Backlog Plugin</a> This plugin integrates <a href="#">Backlog</a> to Jenkins.	1.9
<input type="checkbox"/>	<a href="#">Build Publisher Plugin</a> This plugin allows records from one Jenkins to be published on another Jenkins.	1.17
<input type="checkbox"/>	<a href="#">Confluence Publisher Plugin</a> This plugin allows you to publish build artifacts as attachments to an <a href="#">Atlassian Confluence</a> wiki page.	1.8
<input type="checkbox"/>	<a href="#">Cittericism dSYM Plugin</a> A Jenkins CI plugin for uploading dSYM files to Cittericism.	1.1
<input checked="" type="checkbox"/>	<b>Deploy Plugin</b> → 2. 选择Deploy Plugin This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build	1.9
<input type="checkbox"/>	<a href="#">Deploy WebSphere Plugin</a> This plugin is an extension of the <a href="#">Deploy Plugin</a> . It takes a war/ear file and deploys that to a running remote WebSphere Application Server at the end of a build.	1.0
<input type="checkbox"/>	<a href="#">Dimensions Plugin</a> This plugin integrates Hudson with <a href="#">Dimensions</a> , the Serena CM solution. *{_}Please note - the maintainer for this plugin is no longer myself (TPayne), but is being changed. Until it has been updated, please forward any issues to Paul Caruana (pcaruana@serena.com)._*	0.8.1
<input type="checkbox"/>	<a href="#">FTP-Publisher Plugin</a> This plugin can be used to upload project artifacts and whole directories to an ftp server.	1.2

3. 点击Install without restart

Install without restart Download now and install after restart

## 4 Android 项目构建

### 4.1 SDK 环境变量的设置

在“系统管理”→“系统设置”→“全局属性”设置 SDK 的环境变量名称与本地 SDK 的路径。

Jenkins

新建  
用户  
任务历史  
项目关系  
检查文件指纹  
**系统管理** → 1  
My Views  
Credentials

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

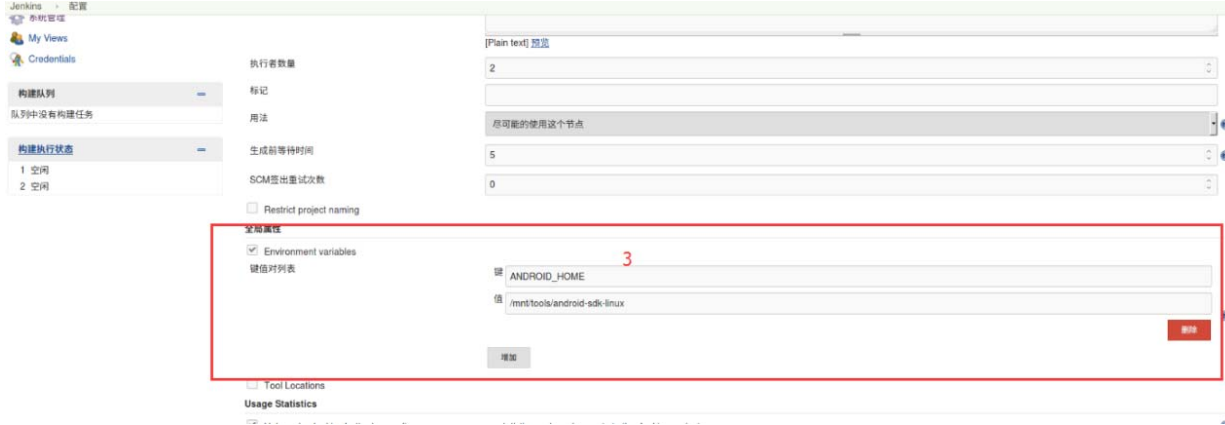
⚠ Jenkins新版本 (2.89.2)可点击 [download](#) ([变更说明](#))下载。  
Warnings have been published for the following currently installed components:

- Jenkins 2.73.3 core and libraries:
  - Multiple security vulnerabilities in Jenkins 2.94 and earlier, and LTS 2.89.1 and earlier
  - Script Security Plugin 1.35:
    - Arbitrary file read vulnerability

Go to plugin manager Configure which of these warnings are shown

**系统设置** → 2  
全局设置&路径

- [Configure Global Security](#)  
Secure Jenkins, define who is allowed to access/use the system.
- [Configure Credentials](#)  
Configure the credential providers and types
- [Global Tool Configuration](#)  
Configure tools, their locations and automatic installers.
- [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。
- [管理插件](#)  
添加、删除、禁用或启用 Jenkins 功能扩展插件。 ([禁用更新](#))



## 4.2新建 Android 项目

开始创建一个 AndroidDemo 项目进行演示:



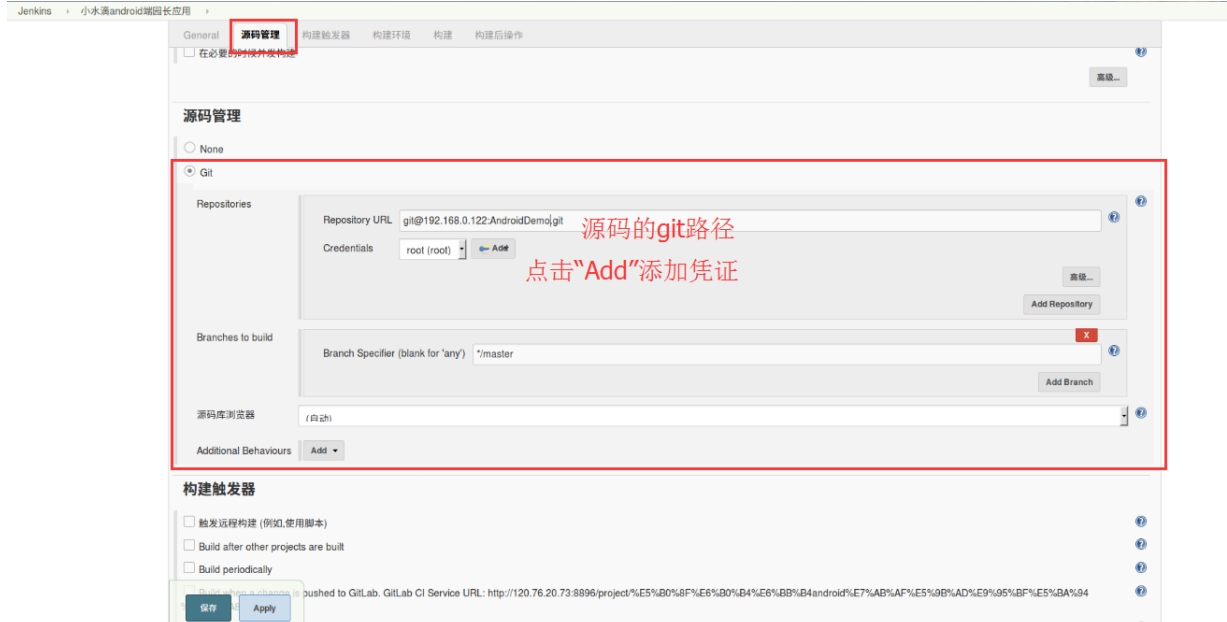
构建一个自由风格的软件项目，然后填写项目名称。



源码管理，这里可以根据自己的实际选择 Git 或者 SVN 服务器。先设置 Git 的源码路径：

然后设置免密凭证。如果是第一次需要通过“Add”添加。





添加凭证：

单击“add”按钮进行添加：



此处的私钥既是1.4章节所产生。

设置完成上面的步骤，直接按左下角保存，项目创建完成。

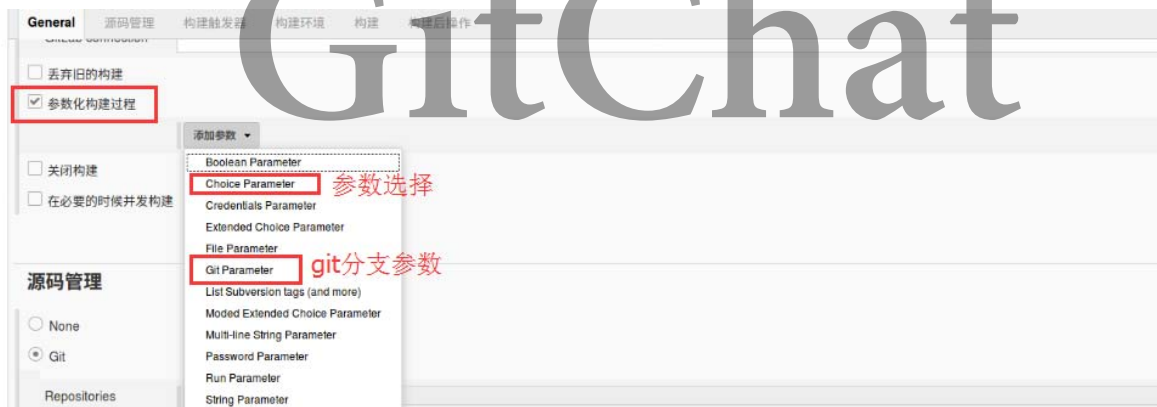
## 5 参数化项目构建

### 5.1 参数设置

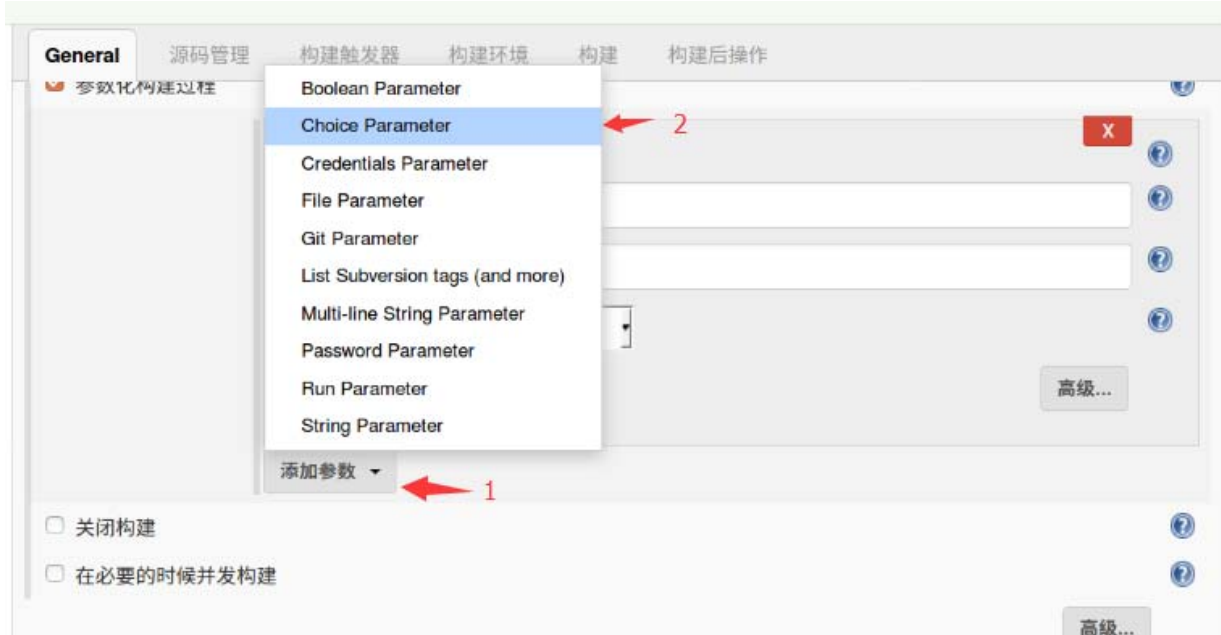
建好的项目，相应修改构建参数等配置，直接通过“配置”进行修改：



选择实际需要用到的参数，比如发布的版本类型，Git 分支参数等。



1) 选择参数的设置：



设置打包的类型是 debug 或者 release。

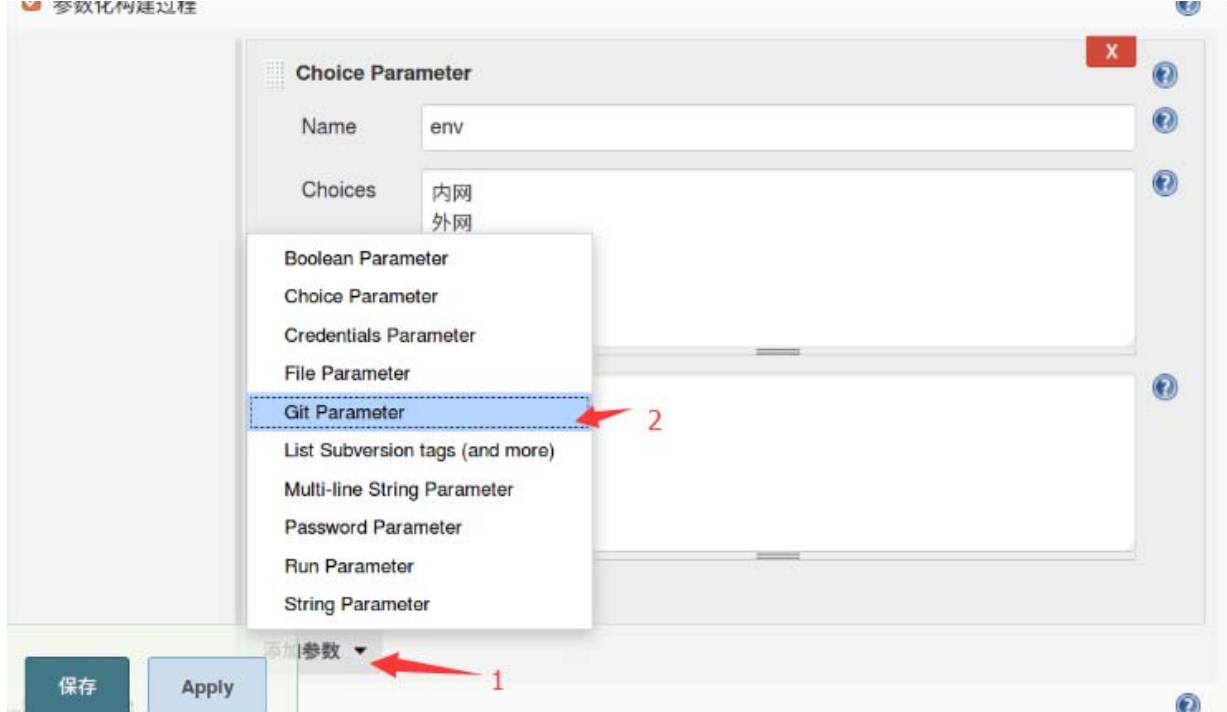


打包的 App 针对的发布平台：

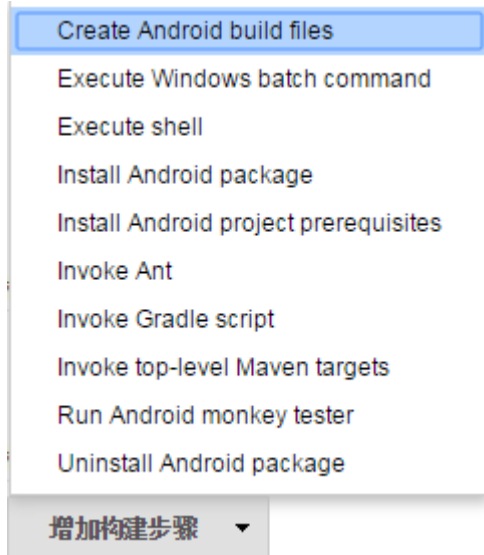


2) Git 分支选择：

想在构建的时候，自动获取 Git 仓库的分支并选择构建，可以设置如下：



3) gradle 脚本命令的配置：



选择“Invoke Gradle script”添加 gradle 命令脚本。



设置完这些参数保存，就可以退出到项目列表界面。

## 5.2 项目构建

进入项目开始构建，选择相应的参数。



构建成功的话，那么结果如下图：



**Jenkins**

Jenkins > AndroidDemo > #10

 返回到工程

 状态集

 变更记录

 Console Output

 编辑编译信息

 删除本次生成

 Git Build Data

 No Tags

 前一次构建

**构建 #10 (2015-12-11 18:36:35)**

**构建产生文件**

demo-debug-unaligned.apk1.70 MB [view](#)

demo-debug.apk1.70 MB [view](#)

demo-release-unsigned.apk1.68 MB [view](#)

No changes.

启动用户[helen](#)

**Revision:** 88a3378242549565620fa8c bae69bdf1e55f7239

- refs/remotes/origin/master

构建完会在“Build History” 部分显示构建结果是成功还是失败并可以查看相应的构建日志，方便分析：

Jenkins > test >

 返回面板

 状态

 修改记录

变更记录

Console Output

编辑编译信息

删除本次生成

Parameters

Git Build Data

No Tags

 构建历史

**Project test**

工作区

最新修改记录

**相关连接**

[Last build\(#2\),19 秒之前](#)

[Last stable build\(#2\),19 秒之前](#)

[Last successful build\(#2\),19 秒之前](#)

[Last completed build\(#2\),19 秒之前](#)

#2

2017-9-12 下午4:48

 **构建结果**

#1

2017-9-12 下午4:29



 RSS 全部

 RSS 失败

构建结果是红色代表构建失败，上图颜色表示构建成功。

构建状态：下图中分级符号概述了一个 Job 新近一次构建会产生四种可能的状态。

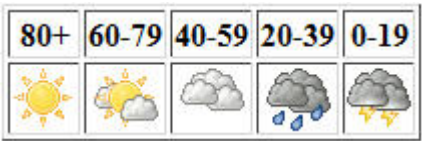
- Successful：完成构建且被认为是稳定的。
- Unstable：完成构建，但被认为不稳定。
- Failed：构建失败。
- Disabled：构建已禁用。

Successful	Unstable	Broken	Disabled
			

在主界面则是通过构建稳定性评分等级进行表示。



构建稳定性：当一个 Job 中构建已完成并生成了一个未发布的目标构件，如果您准备评估此次构建的稳定性，Jenkins 会基于一些后处理器任务为构建发布一个稳健指数（从0-100），这些任务一般以插件的方式实现。它们可能包括单元测试（JUnit）、覆盖率（Cobertura）和静态代码分析（FindBugs）。分数越高，表明构建越稳定。下图中分级符号概述了稳定性的评分范围。任何构建作业的状态（总分100）低于80分就是不稳定的。



还有很多的参数配置，如触发器配置、邮箱配置，自动化发布等的参数很多，这里就不一一介绍，感兴趣的朋友可以上网或者留言交流。

# GitChat