

# Docker Jenkins + SVN| GitLab CI/CD

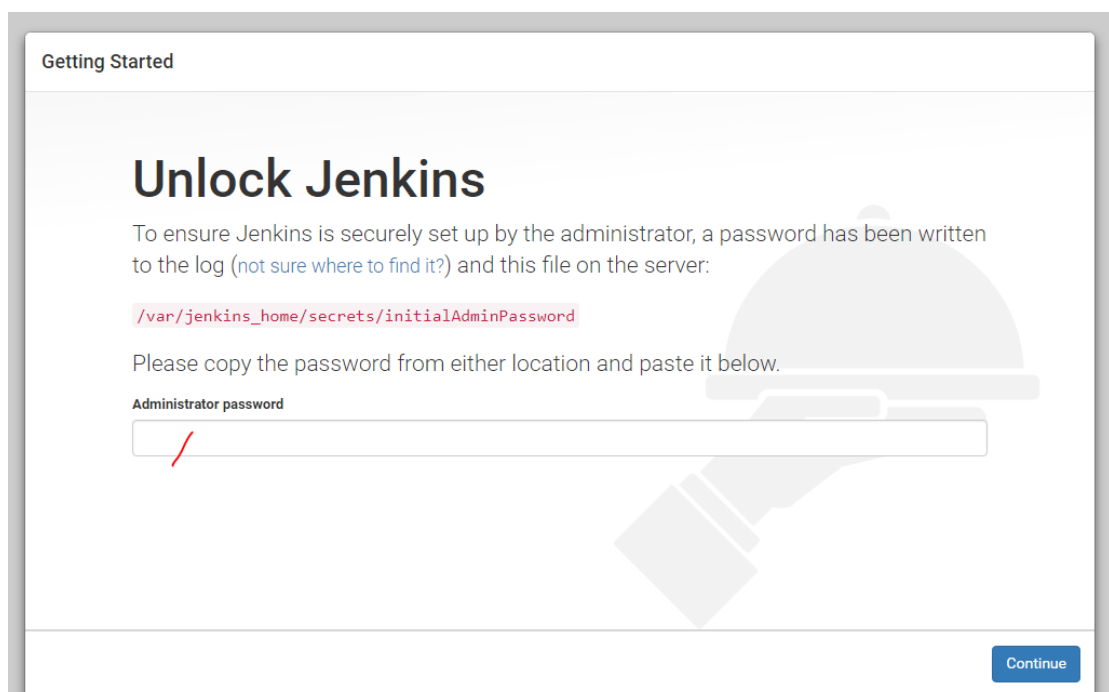
1 Docker 容器中启动jenkins	2
2 选择插件安装	3
3 设置超级管理员帐号	4
4 依赖环境	4
5 SVN+Jenkins自动构建	4
5.1 创建一个自由风格的软件项目	4
5.2 源码管理选择Subversion	5
5.3 配置构建触发器	5
6 Git+Jenkins自动构建	8
附录:	16

作者	日期	版本
姜鹏	2019/03/25	V0.1

# 1 Docker 容器中启动jenkins

版本: Jenkins 2.60.3

```
$docker run -u root --rm -d --name jenkins -p 8099:8080 -p 50001:50000 -v /data/jenkins:/var/jenkins_home/ -v /usr/home/softwares/apache-maven-3.6.0:/var/lib/maven -v /var/run/docker.sock:/var/run/docker.sock jenkinsci/blueocean
```



Password: 0c15823f28a240378e3480bb3ba402d1\$docker

进入容器:

```
$exec -it Jenkins sh
```

查看密码:

```
$ cat /var/jenkins_home/secrets/initialAdminPassword
```

若容器启动jenkins失败, 可以尝试:

清理此容器的网络占用

```
$docker network disconnect --force bridge jenkins
```

查看是否还有同名容器占用

```
$docker network inspect jenkins
```

```
systemctl start docker
```

```
systemctl stop docker
```

## 2 选择插件安装

Getting Started

### Installation Failures

Some plugins failed to install properly, you may retry installing them or continue with the failed plugins

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ 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

Jenkins 2.60.3

[Continue](#) [Retry](#)

## 3 设置超级管理员帐号

jenkins/Jenkins

## 4 依赖环境

➤ JDK: /docker-java-home

```
$ java -version
openjdk version "1.8.0_171"
OpenJDK Runtime Environment (build 1.8.0_171-8u171-b11-1~deb9u1-b11)
OpenJDK 64-Bit Server VM (build 25.171-b11, mixed mode)
$ pwd
/docker-java-home/bin
$
```

## 5 SVN+Jenkins自动构建

### 5.1 创建一个自由风格的软件项目

Enter an item name

spring-boot-redis

» Required field

**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，因此你可以有多个相同名的内容，只要它们在不同的文件夹里即可。

OK

## 5.2 源码管理选择Subversion

源码管理

☐ 无

☐ Git

☐ Mercurial

☒ Subversion

Modules

Repository URL

svn://39.108.210.27:3691/dubbo

?

Credentials

root/\*\*\*\*\*

Ad

Local module directory

.

?

Repository depth

infinity

?

Ignore externals

☒

?

Cancel process on externals fail

☒

?

## 5.3 配置构建触发器

**构建触发器**

☒ 触发远程构建 (例如,使用脚本) ?

身份验证令牌   
Use the following URL to trigger build remotely: JENKINS\_URL/job/apache-dubbo/build?token=TOKEN\_NAME 或者 /buildWithParameters?token=TOKEN\_NAME  
Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

☐ 其他工程构建后触发 ?

☒ 定时构建 ?

日程表   
上次运行的时间 Thursday, November 29, 2018 10:47:17 PM GMT; 下次运行的时间 Thursday, November 29, 2018 10:52:17 PM GMT. ?

☐ GitHub hook trigger for GITScm polling ?

☒ 轮询 SCM ?

日程表  ?

输入：

<http://39.108.210.27:8099/job/apache-dubbo/build?token=trigger01>

可以触发一次构建

(1) 定时构建：不管SVN或Git中数据有无变化，均执行定时化的构建任务；

(2) 轮询SCM：只要SVN或Git中数据有更新，则执行构建任务；

构建语法说明：

(1)格式为：\*\*\*\*\*（五个星）；

(2) 第一个\*表示分钟，取值0~59

(3) 第二个\*表示小时，取值0~23

(4) 第三个\*表示一个月的第几天，取值1~31

(5) 第四个\*表示第几月，取值1~12

(6) 第五个\*表示一周中的第几天，取值0~7，其中0和7代表的都是周日

使用举例：

每隔10分钟构建一次： H/10 \* \* \* \*

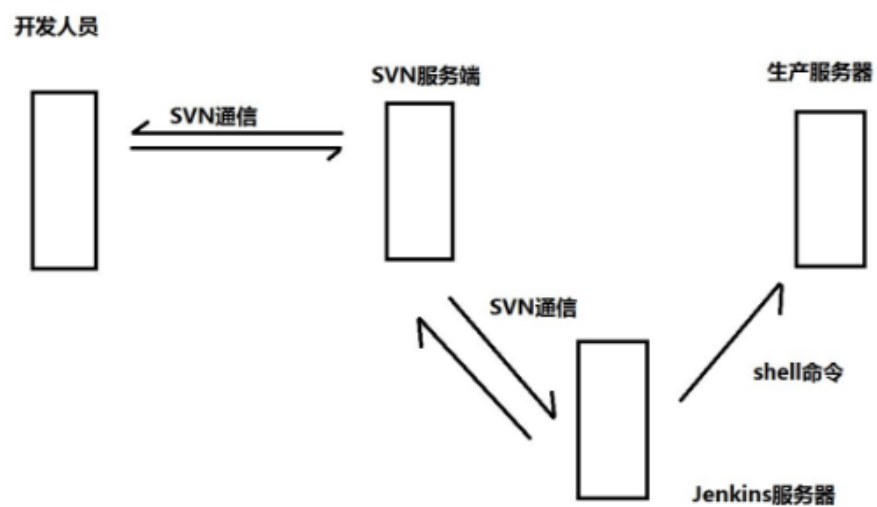
每隔1小时构建一次： H H/1 \* \* \*

每月30号构建一次： H H 30 \* \*

The screenshot shows the Jenkins configuration page for a build step. It includes the following settings:

- 注入构建变量** (Inject build variables): ☐
- 使用Maven私有仓库** (Use Maven private repository): ☐
- 配置文件** (Configuration file):
  - Dropdown menu: 文件系统中的 settings 文件 (File in the file system)
  - 文件路径 (File path): /usr/home/softwares/apache-maven-3.6.0/conf/settings.xml
- 全局配置文件** (Global configuration file):
  - Dropdown menu: 使用默认 Maven 全局设置 (Use default Maven global settings)

At the bottom, there is a button labeled **增加构建步骤** (Add build step) with a dropdown arrow.



## 6 Git+Jenkins自动构建



Git项目: <https://github.com/alanjiang/hook.git>

JDK1.8 : /usr/bin/java

Git1.8.3.1: /usr/bin/git

Maven3.6.0: /usr/home/softwares/apache-maven-3.6.0/

### • 第一步:

进入jenkins，点击“系统管理”， 点击 “系统设置”。 进入gitlab 选项:

要求填写 URL和凭据。

The screenshot shows the Jenkins configuration interface for GitLab. At the top, there is a '新增' (Add) button. The main section is titled 'Gitlab' and includes a checkbox for 'Enable authentication for 'project' end-point' which is checked. Below this, the 'GitLab connections' section is visible. It contains the following fields:

- Connection name:** A text input field containing 'spring-boot-redis'.
- Gitlab host URL:** A text input field containing 'http://39.108.210.27/root/spring-boot-redis.9'.
- Credentials:** A dropdown menu showing '- 无 -' (None) and an 'Add' button.

Below the 'Credentials' field, there is a red warning message: 'API Token for Gitlab access required' and 'API Token for accessing Gitlab'. At the bottom right, there are buttons for '高级...' (Advanced...), 'Test Connection', and '删除' (Delete). At the bottom center, there is another '新增' (Add) button.



## • 第二步：配置gitlab 公匙+私匙

例如：我的gitlab帐号为：root，对应的邮箱为：[admin@example.com](mailto:admin@example.com)

(1) 生成公、私匙放在本机

```
/Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa  
$mkdir -p /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/  
$touch /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/  
id_rsa
```

执行：

```
$ssh-keygen -t rsa -C "admin@example.com"
```

最后生成了公匙和私匙，如下：

```
zhangxiaodeMacBook-Air:~ zhangxiao$ pwd  
/Users/zhangxiao/Desktop/keys-gitlab/root/.ssh  
zhangxiaodeMacBook-Air:~ zhangxiao$ ls  
id_rsa      id_rsa.pub
```

(2) 将公匙id\_rsa.pub的内容配置在gitlab上。

User Settings > SSH Keys

### 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 one](#) or use an [existing key](#).

#### Key

```
ssh-rsa  
AAAAB3NzaC1yc2EAAAADAQABAAQCKm+jdWKYTuYRTbSDEBsLUzMbrBBpE1IRxe76fLX  
bhXTHswlxUQ9nmaBJRvTmcPwQdoieXhrLjJv5eRcENRYHncLyIXJqZSYFJA4UTd+0dz64dOclz  
0IKVy71GS7vkGMP+B5v1H3kEfpTsDaAMUJ8XzqvibGaG3rQamNosobly7tRu+H4cU94VggT  
EV78ITqUTIQQR4qX3fQwmGj2lcJ1zIFvNMmSVGf7wVkh/erwvi1FTqUsNQZLQx+q9HhWJLaqVB  
+JV+XskukiteHJbls4NBWGYdtqbLmJEQO9YXqp078oz16WTUJ8S2nL7WAH9nis4YdfhM4EoZ  
QerBHDfXDn admin@example.com
```

#### 标题

admin@example.com

Add key

点击“Add Key” 完成配置。

### (3) 配置全局用户名变量

```
git config --global user.name "root"
```

```
git config --global user.name "admin@example.com"
```

### (4) 添加sshkey至ssh-agent

```
$ssh-add /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa
```

```
[zhangxiaodeMacBook-Air:gitlab-workspace zhangxiao$ ssh-add /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa
Enter passphrase for /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa:
Identity added: /Users/zhangxiao/Desktop/keys-gitlab/root/.ssh/id_rsa (admin@example.com)
```

测试ssh-agent是否生效，执行：

```
$ssh-agent
```

测试一下使用ssh协议git clone 一个gitlab上的项目：

```
zhangxiaodeMacBook-Air:gitlab-workspace zhangxiao$ git clone git@39.108.210.27:root/spring-boot-redis.git
Cloning into 'spring-boot-redis'...
remote: Counting objects: 568, done.
remote: Compressing objects: 100% (384/384), done.
remote: Total 568 (delta 145), reused 535 (delta 117)
Receiving objects: 100% (568/568), 246.26 KiB | 153.00 KiB/s, done.
Resolving deltas: 100% (145/145), done.
```

### (5)gitlab 上生成一个 “个人访问的Token”。

请记住这个gitlab的API Token：

g97g3\_wRyVcDRc9ELa\_6

Rnner Token: wLKrAnSAYzFzPbDQcedm

## Active Personal Access Tokens (1)

Name	Created	Expires	Scopes	
spring-boot-redis	Mar 15, 2019	Never	api, read_user, sudo, read_repository	Revoke

Account

Applications

Chat

Access Tokens

Emails

Password

Notifications

SSH Keys

GPG Keys

Preferences

Active Sessions

Authentication log

Pipeline quota

each application you use that needs access to the GitLab API.

You can also use personal access tokens to authenticate against Git over HTTP. They are the only accepted password when you have Two-Factor Authentication (2FA) enabled.

Name

spring-boot-redis

Expires at

Scopes

☐ api Access the authenticated user's API

Full access to GitLab as the user, including read/write on all their groups and projects

☐ read\_user Read the authenticated user's personal information

Read-only access to the user's profile information, like username, public email and full name

☐ sudo Perform API actions as any user in the system (if the authenticated user is an admin)

Access to the Sudo feature, to perform API actions as any user in the system (only available for admins)

☐ read\_repository Read Repository

Read Repository

Create personal access token

Active Personal Access Tokens (0)

This user has no active Personal Access Tokens.

(6) 在jenkins上配置凭据：选择 Gitlab API token，填写上一步的API Token。

新增

Jenkins 凭据提供者: Jenkins

添加凭据

Domain

全局凭据 (unrestricted)

类型

GitLab API token

范围

全局 (Jenkins, nodes, items, all child items, etc)

API token

.....

ID

描述

添加

取消

也可以创建一个基于SSH公、私匙的凭据：如下图：

范围

全局 (Jenkins, nodes, items, all child items, etc)

Username

root

Private Key

Enter directly

Key

-----BEGIN OPENSSH PRIVATE KEY-----  
b3BlnNzaC1rZXktZjEAAAAACmFlczl1Ni1jdHIAAAAGYmNyeXB0AAAAAGAAAABB4b/A073  
g6e936K8AsU4zzAAAAEAAAAEAAAEAXAAAB3NzaC1yc2EAAAADAQABAAQCKm+jdWKYT  
uYRTbSDEBsLUzMBRBBpE1iRxe76fLXbhXTHswlxUQ9nmaBjRvTmcPwQdoieXhrLjJv5eRc  
ENRYHncLYiXJqZSYFJA4UTd+0dz64dOclz0IKVy71GS7vkGMP+B5v1H3kEtpaTsDaAMUJ8  
XzqvjbGaG3rQamNosobLy7iRu+H4cU94VqgTEV78jTqUTJQOqr4qX3fQwmGj2lcJ1zIFyNM  
mSVGF7wVkhH/prwyj1FTqUsNQZLQx+q9HhWLaqVB+JV+XskukiteHJbIs4NBWGYdtqbLmJ  
EqO9YXqp078oz16WTUJ8S2nL7WAH9nis4YdfhM4EoZQerBHDfXDnAAD0BghNfJWRvtBOP  
6THZKayi0yuqAtwraTKI37PtcxOAfP09rC7IjKXvfbJtwsXtLpmI3UX99BHUWACY/oUZvE  
Mr5TXn4M1RI5hCQ7bi/qvmQTvK9LxB9JrRmSs85m5RWzqxqF9xylMa+2NUalgVRL9Yh9t2  
imAozv7XHdbYKdZIFPOTMtk0EMv/oXLay4hP8VQQALSNWIAJcLNVZtqg1JHtViiNb/AHF  
oiyIt5ZGmFJAeRc3GuVGrZnMqd16MgR9sIBCgzwoKlGu2ociyhXS0IEF5OMdiHbBblgW11  
DarWPVAZ85GXNlmoAvjdekTJ+p1I/Tz6DvnK14l89ATjldyIVc7WrRTOysX6KmsMtyQzR  
4xc/7SXIPIcXsJdz7YG3H4sohuCmGG9J2k8xhCsS0WuhrPsBb98LCEfIfpkvFShCe523Ci  
V7/Tq4g9KcV+JNSSnyEfnr7F+s/xbhtxqEwlqRrX3jby/MWG0OZB92r8xCH1/VnWSe1Hz0  
/Hl1/3i8TYba2n1UA/2qnQSBROWtl49pf5iif6JkHx0NK1ucdRnoi3/LcXAOSQRv1jszl  
UTWPJMF8K8zlogFTKbRp4nlXeMbrDsR+ISrGb9xqsUkWhG5iO2TZKmY/gYABOtKnROtaPO  
EnsfXG1fErJi5EqG8MxawOq48GNEsgxngAbgQL/JONXXe8IO6aBwfpukJOnV20rq+wVw1  
7OyKZA+DT4ss4QDkEfoXI9szls7JoOBl7sCjkDTKLueZSzy9mVZwzYO/i/Gl1WHb7W4oLh  
/W8e0ATLC+qVpAiS9+ozK11HPMT7QJHHeRQDmeExsb9nCNikA5jsb8F2SVY4JTrB2y+xs  
LUFZ0UxXTHGvAsrOR6KgoConlfoCuGCxRDHcAgEpF517kMIMSiY7YV9WPY82Bs80ShlIp  
itRUaP1pkqW3NvmM14XDOlJTUOH9GZHumiJ5aW9uQf50+SH0emPoAI7VD0EaA9UI9Y5FIf  
rm1J+wKKLqv+2D5JWe5lwQNB0WiVzx0xR7T6kFNNDW7QUskZHCgMlad2haN5ewhrOd7JOd  
rDURmp/aMRsKYTCuRZacS98IU9iwwi2APqD5cNoaU05la6Z0lnb+XgoQ5C3Tq4nsh1OLwD  
fWLx3lZxx0iJeYkKhFa02Xu68RZA1p0pS3FXWJAGVmOsnZlU/Sz035zhQ+vCSJfaTMcS  
wGqJLKK7zGT22KUncxBXj4PIChzoJq/plCnCPirqi2F03MuA7YwQ4yP3E9K1oM/LjgTRrT  
mkGcDmE/J3hfireCdUIGWTRT8tFsg=  
-----END OPENSSH PRIVATE KEY-----

Passphrase

.....

ID

aa35ead0\_6bae44f0-b0a6-65a2468a1d2a

注意：这里的私匙内容与gitlab中设置的公匙是成对的。

(7) 再次回到 “系统管理” -> “系统设置”：

(8) 点“配置”

“源码管理”选项卡中填写 git项目的URL， credentials 中选取上步骤中 (6)的设置的SSH凭据，也可以是API TOKEN， 这儿演示SSH凭据。

(9) 构建选项卡中，选择 “执行SHELL”

可以将MAVEN的构建、打包命令写在执行的SHELL脚本中。

可以包括：

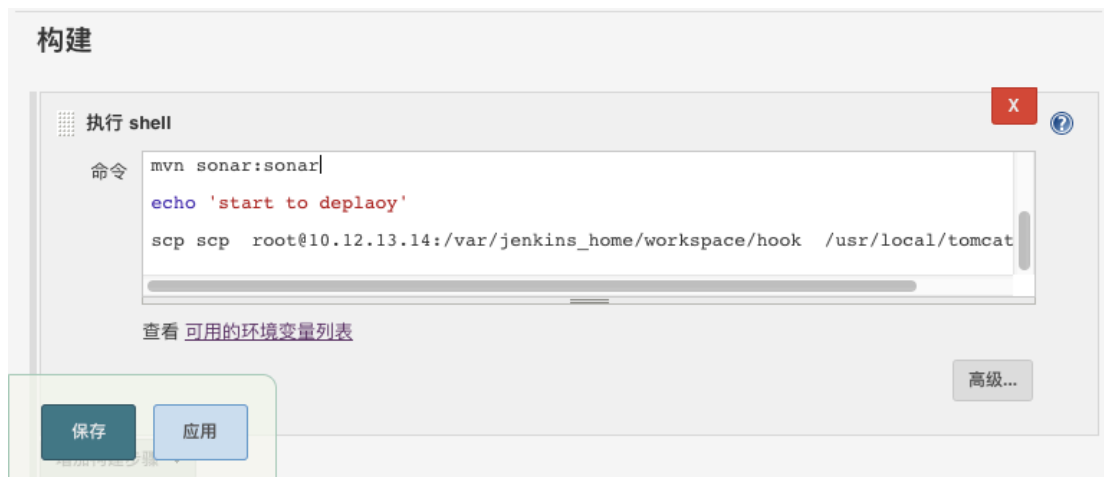
运行sonar:sonar 分析；

将WAR包部署在远程服务器上，并启动WEB应用；

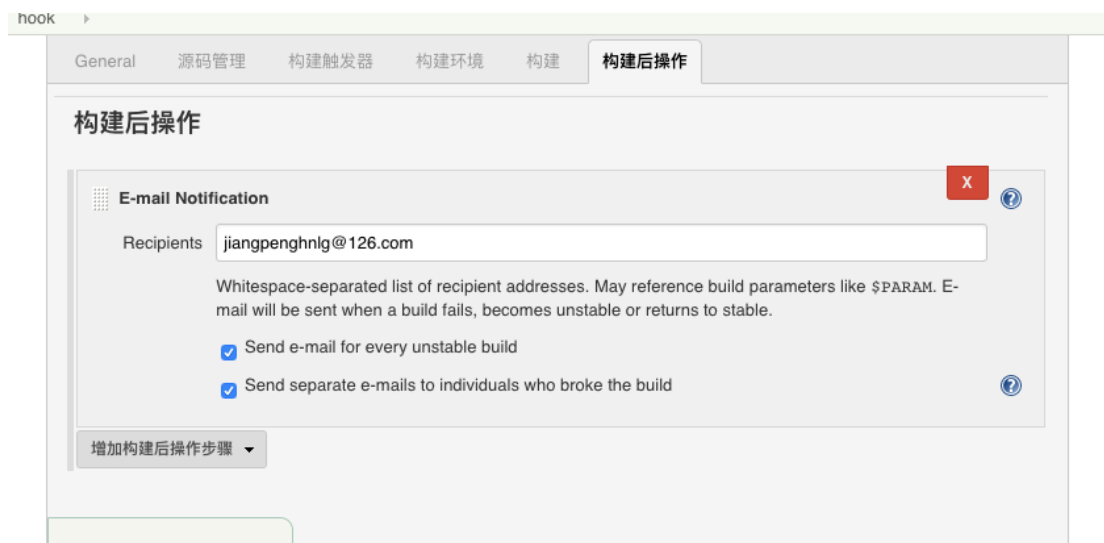
可提交一个PULL REQUEST；

GIT分支合并。。。

轻松完成CI/CD。



(10) 构建后操作，可以配置邮箱。



点击“构建历史”，可以看见所有构建历史项目。



The screenshot shows the Jenkins main dashboard. On the left is a sidebar with navigation links: 新建任务, 用户, 构建历史, 项目关系, 检查文件指纹, 系统管理, 我的视图, 打开 Blue Ocean, Lockable Resources, and 凭据. The main area displays a table of jobs. The table has columns: S, W, 名称 ↓, 上次成功, 上次失败, 上次持续时间, and 收藏. Three jobs are listed: dubbo, gitlb+jenkins, and hook. Below the table are links for 图标: S M L and RSS feeds.

S	W	名称 ↓	上次成功	上次失败	上次持续时间	收藏
		<a href="#">dubbo</a>	18 days - <a href="#">#19</a>	3 月 25 days - <a href="#">#6</a>	2 分 36 秒	
		<a href="#">gitlb+jenkins</a>	没有	8 days 18 小时 - <a href="#">#2</a>	8 分 33 秒	
		<a href="#">hook</a>	40 分 - <a href="#">#6</a>	21 分 - <a href="#">#9</a>	0.56 秒	

点击项目的名称 链接，进入构建的项目列表，可以查看日志等信息。



The screenshot shows the Jenkins job build history page for the 'hook' job. The browser address bar shows '39.108.210.27:8099/job/hook/'. The page has a sidebar with 收藏夹, 打开 Blue Ocean, and 重命名. The main content area has a 'Build History' section with a search bar and a list of builds. To the right is a '相关链接' (Related Links) section with links to various build numbers.

Build History

Build Number	Timestamp
#9	2019-3-25 上午5:39
#8	2019-3-25 上午5:28
#7	2019-3-25 上午5:24
#6	2019-3-25 上午5:19
#5	2019-3-25 上午5:19
#4	2019-3-25 上午5:19
#3	2019-3-25 上午5:15
#2	2019-3-25 上午5:14

相关链接

- [最近一次构建\(#9\), 22 分之前](#)
- [最近稳定构建\(#6\), 41 分之前](#)
- [最近成功的构建\(#6\), 41 分之前](#)
- [最近失败的构建\(#9\), 22 分之前](#)
- [最近未成功的构建\(#9\), 22 分之前](#)
- [最近完成的构建\(#9\), 22 分之前](#)

查看“控制台输出”， 可以查看输出的报告。

Jenkins

2

查找

Alan | 注销

ns > hook > #9

返回到工程

状态集

变更记录

控制台输出

|| 文本方式查看

编辑编译信息

删除本次生成

Git Build Data

No Tags

打开 Blue Ocean

前一次构建

控制台输出

Started by user [Alan](#)  
Building in workspace /var/jenkins\_home/workspace/hook  
> git rev-parse --is-inside-work-tree # timeout=10  
Fetching changes from the remote Git repository  
> git config remote.origin.url git@39.108.210.27:root/spring-boot-redis.git # timeout=10  
Fetching upstream changes from git@39.108.210.27:root/spring-boot-redis.git  
> git --version # timeout=10  
using GIT\_SSH to set credentials root/secret\_pass  
> git fetch --tags --progress git@39.108.210.27:root/spring-boot-redis.git +refs/heads/\*:refs/remotes/origin/\*  
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10  
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10  
Checking out Revision 67ec65cd9e057893394a186c7ec476a4ffa6bd05  
(refs/remotes/origin/master)  
> git config core.sparsecheckout # timeout=10  
> git checkout -f 67ec65cd9e057893394a186c7ec476a4ffa6bd05  
Commit message: "add"

## 附录：

云服务中应用到的命令：

启动 停止Docker

```
$ sudo systemctl start docker
```

```
$sudo systemctl stop docker
```

```
$service docker status
```

Gitlab 服务：

```
gitlab-ctl stop|start|restart
```

参考：

<https://jenkins.io/doc/>