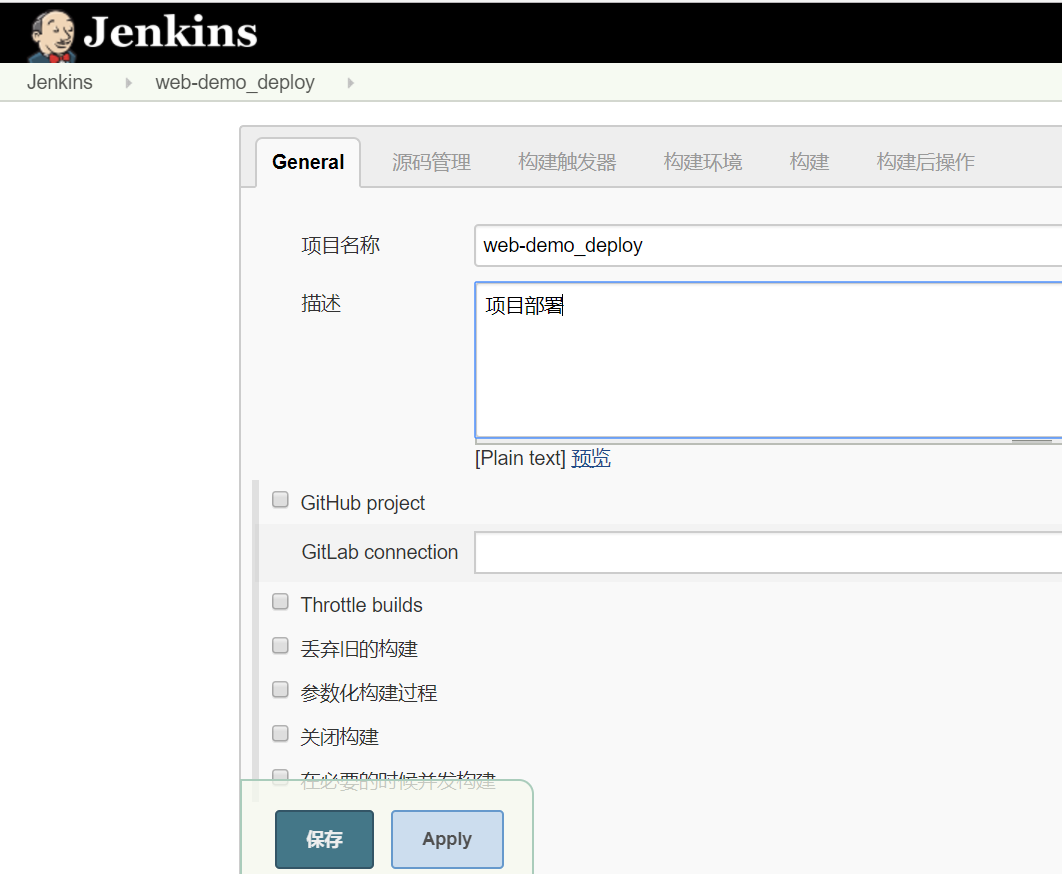


一：本文在上一篇文章的基础之上继续进行操作，上一篇实现了从git获取代码并进行代码测试，本文将在上一篇的基础之上实现将代码部署至测试环境。  
1.1：新建一个项目叫web-demo-deploy用于代码发布，上一个项目web-demo可用于代码测试，当测试阶段出现问题的时候也不会立即进行发布，只有当测试通过之后才执行发布的项目即可：



1.2：如何将代码发布到web服务器：  
1.2.1：可以通过执行命令或脚本的方式进行代码发布，在各web服务器创建一个www用户，用于启动web服务并进行代码发布：

IMG_257

# useradd www

# echo "123456" | passwd --stdin www

# su - www

$ ssh-keygen -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/home/www/.ssh/id\_rsa):

Created directory '/home/www/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/www/.ssh/id\_rsa.

Your public key has been saved in /home/www/.ssh/id\_rsa.pub.

The key fingerprint is:

bc:51:20:7d:cc:bb:de:e8:e4:11:d3:f7:1b:ec:0c:0d www@node1.chinasoft.com

The key's randomart image is:

+--[ RSA 2048]----+

| ...o |

| ...+ |

| ... |

| . ... |

| S o..E. |

| o.o .+. |

| ..oo . +.|

| oo.. + o|

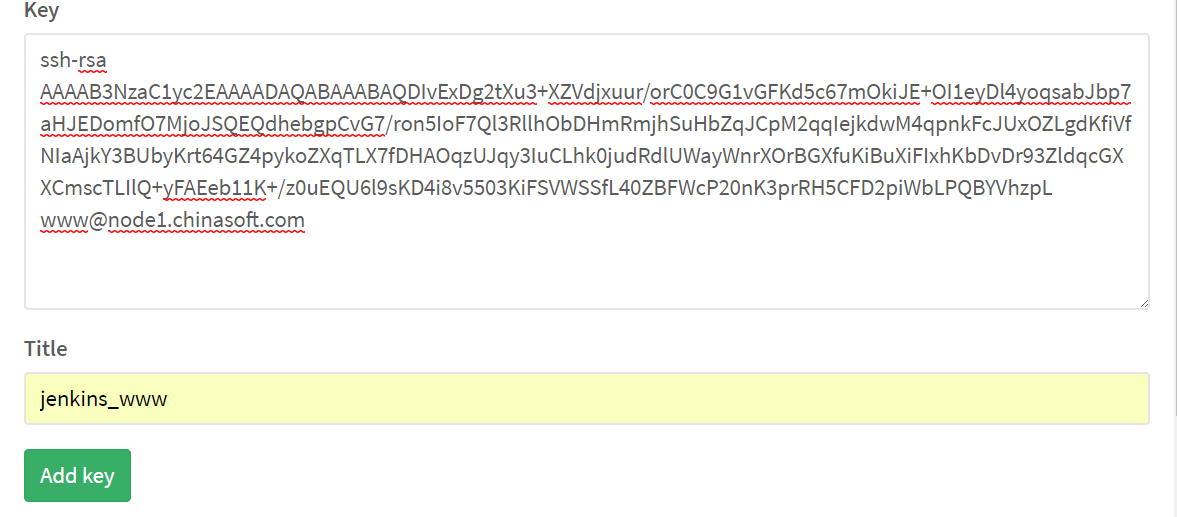
| .o + |

+-----------------+

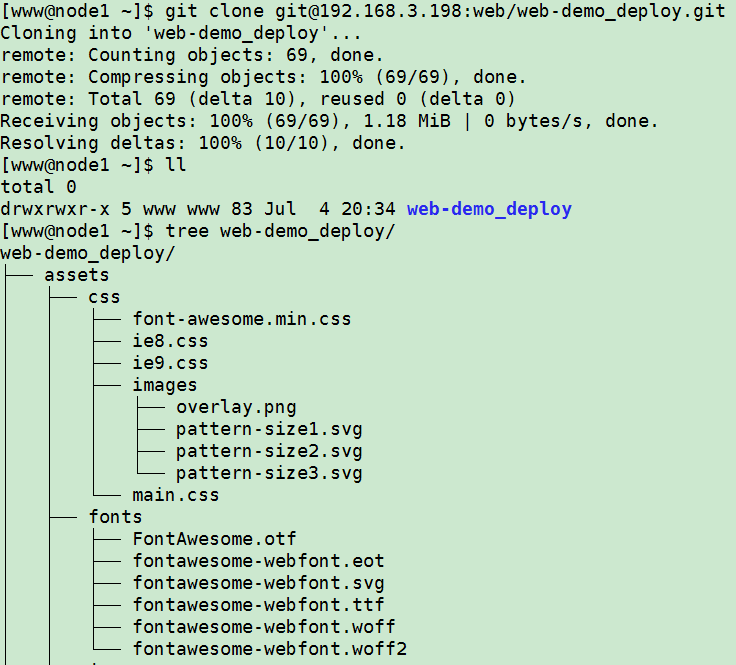
[www@node1 ~]$ cat .ssh/id\_rsa.pub ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDIvExDg2tXu3+XZVdjxuur/orC0C9G1vGFKd5c67mOkiJE+OI1eyDl4yoqsabJbp7aHJEDomfO7MjoJSQEQdhebgpCvG7/ron5IoF7Ql3RllhObDHmRmjhSuHbZqJCpM2qqIejkdwM4qpnkFcJUxOZLgdKfiVfNIaAjkY3BUbyKrt64GZ4pykoZXqTLX7fDHAOqzUJqy3IuCLhk0judRdlUWayWnrXOrBGXfuKiBuXiFIxhKbDvDr93ZldqcGXXCmscTLIlQ+yFAEeb11K+/z0uEQU6l9sKD4i8v5503KiFSVWSSfL40ZBFWcP20nK3prRH5CFD2piWbLPQBYVhzpL www@node1.chinasoft.com

IMG_258

1.2.2：在git服务器将www用户的公钥添加至部署key，将root的公钥添加至ssh key，以让www用户有获取代码权限，让root用户有提交代码的权限：

  
ssh keys和deploy keys区别：  
github账户的SSH keys，相当于这个账号的最高级key，只要是这个账号有的权限（任何项目），都能进行操作。  
仓库的Deploy keys，顾名思义就是这个仓库的专有key，用这个key，只能操作这个项目，其他项目都没有权限。  
说白了就相当于你有一所大别墅，SSH key能开别墅中的任何一个房间。而Deploy key只能开进别墅中的一个单间。

1.2.4：确认www用户有拉取代码权限：



1.3：关于shell脚本执行权限：

#稍后会通过jenkins执行一个脚本，从而完成代码的发布，但是默认执行的用户是jenkins，需要赋予jenkins一定的权限，另外发布的脚本可能在本机也可能不在本机，本次设想不在本机保存脚本，则设置如下：

1.3.1：解决脚本运行问题：

#将脚本放在www用户家目录/home/www，git代码也放在家目录，因此需要jenkins服务器远程到代码发布服务器执行远程命令，需要做免登陆认证，将jenkins服务器root和www用户的公钥放在代码部署服务器的www用户家目录.ssh/authorized\_keys文件中，使jenkins服务器能够不输入密码就可以调用部署服务器的脚本：

jenkins服务器：192.168.3.199  
deploy部署服务器：192.168.3.12

IMG_261

$ chmod 600 authorized\_keys

$ cat authorized\_keys

[www@192.168.3.12 ~]$ cat .ssh/authorized\_keysssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDIvExDg2tXu3+XZVdjxuur/orC0C9G1vGFKd5c67mOkiJE+OI1eyDl4yoqsabJbp7aHJEDomfO7MjoJSQEQdhebgpCvG7/ron5IoF7Ql3RllhObDHmRmjhSuHbZqJCpM2qqIejkdwM4qpnkFcJUxOZLgdKfiVfNIaAjkY3BUbyKrt64GZ4pykoZXqTLX7fDHAOqzUJqy3IuCLhk0judRdlUWayWnrXOrBGXfuKiBuXiFIxhKbDvDr93ZldqcGXXCmscTLIlQ+yFAEeb11K+/z0uEQU6l9sKD4i8v5503KiFSVWSSfL40ZBFWcP20nK3prRH5CFD2piWbLPQBYVhzpL www@node1.chinasoft.comssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCsYf1pLYFBUhThXz5pqPMl9TVJxzKEkB/6vImEcDnBqDhrWZe+OqIWp+GTbkHNcXDejD1pBvvQScPIuxlz/r7OEBRTpTjmZOAaLCRMljhx2iMsgTdyjSqZFXMAXRI+F/ZPKKypDW2ZLMLjyqB6ZHK+9/SIMVGwzw/Ey3kqAQovI7UQMoL/59xjah+9zNGboTpZI613LX5vrgCghWUS5NHxU/DNUWjaxFuYJqr7ELKVrG/vZJcbtwmGpRDcCq03Kl2Mz0lHhkGZVDHWqhIPcyRjKrDh0/WqaTlPuIYZ3bZu33aQSxXV5GMGv6VqfIkYVU0uFewL4znPKFPa1z4mAJpR root@node1.chinasoft.com

IMG_262

1.3.2：确认可以免密码远程登陆：  
使用root和www用户测试一下是否可以免秘钥登陆，以便让部署服务器将用户的key添加到know\_keys，否则报错Host key verification failed

ssh www@192.168.3.12ssh www@192.168.3.13

1.3.3：解决jenkins没有权限的问题，在jenkins服务器192.168.3.199上操作：

# vim /etc/sudoers

#Defaults requiretty #注释掉，不需要tty

jenkins ALL=(ALL) NOPASSWD: /usr/bin/ssh

1.3.3：配置jenkins项目执行shell脚本：  
1.3.3.1:脚本内容(需要放在部署服务器192.168.3.12的/home/www目录下):

IMG_263

www@192.168.3.12 $ vim dep.sh

#!/bin/bash

cd /home/www/web-demo\_deploy/ #进入到本地代码库

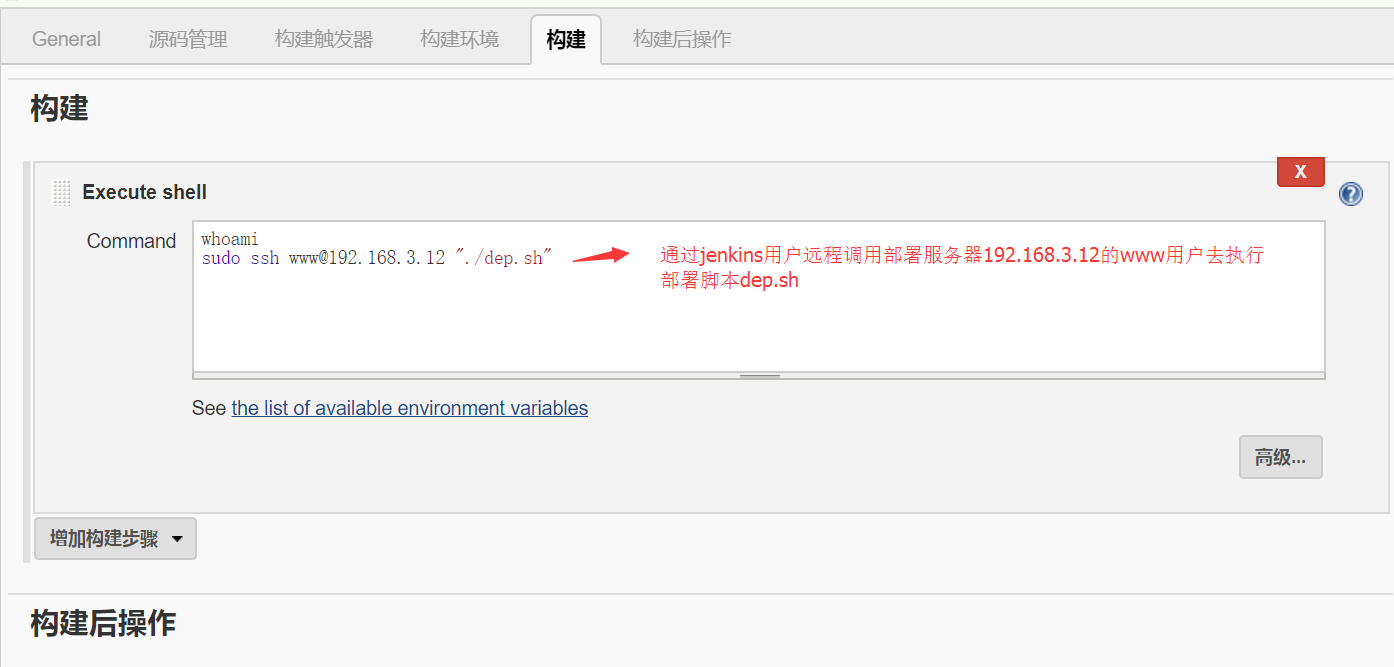
git pull #从git服务器更新代码scp -r ./\* www@192.168.3.12:/webroot/web\_www #将代码部署至web服务器

scp -r ./\* www@192.168.3.13:/webroot/web\_www

www@192.168.3.12 $ chmod +x dep.sh

IMG_264

1.3.3.2:在jenkins调用脚本:  
#在项目的构建步骤调用，项目-配置-构建-增加构建步骤-Execute shell





访问web页面测试：



在git仓库创建代码并更新至git服务器：

IMG_268

[www@master web-demo\_deploy]$ vim index.html # 添加www.chinasoft.com

[www@master web-demo\_deploy]$ git add index.html

[www@master web-demo\_deploy]$ git commit -m 'edit index.html add www.chinasoft.com'

[master 51f8f11] edit index.html add www.chinasoft.com1 file changed, 1 insertion(+), 1 deletion(-)

[www@master web-demo\_deploy]$ git push origin master

Counting objects: 5, done.

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 313 bytes | 0 bytes/s, done.

Total 3 (delta 2), reused 0 (delta 0)

To git@192.168.3.198:web/web-demo\_deploy.git

ac41e81..51f8f11 master -> master

IMG_269

再次在jenkins执行项目构建

可以看到刚刚更新的代码获取成功



1.4：让代码测试项目管理代码发布项目，当代码测试的项目执行成功之后自动调用代码发布的项目完成代码部署：  
1.4.1：安装插件，jenkins的插件默认安装路径

# ll /var/lib/jenkins/plugins/

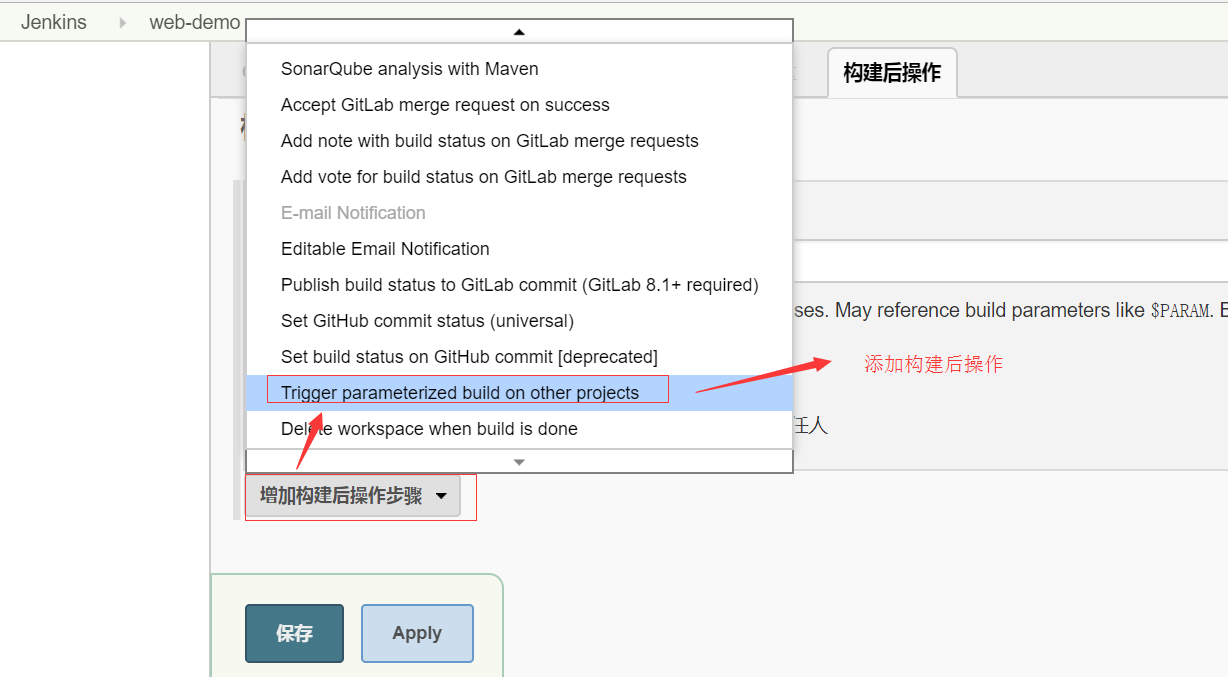
tomcat版本的安装路径：

/usr/local/tomcat/webapps/jenkins/WEB-INF/detached-plugins/

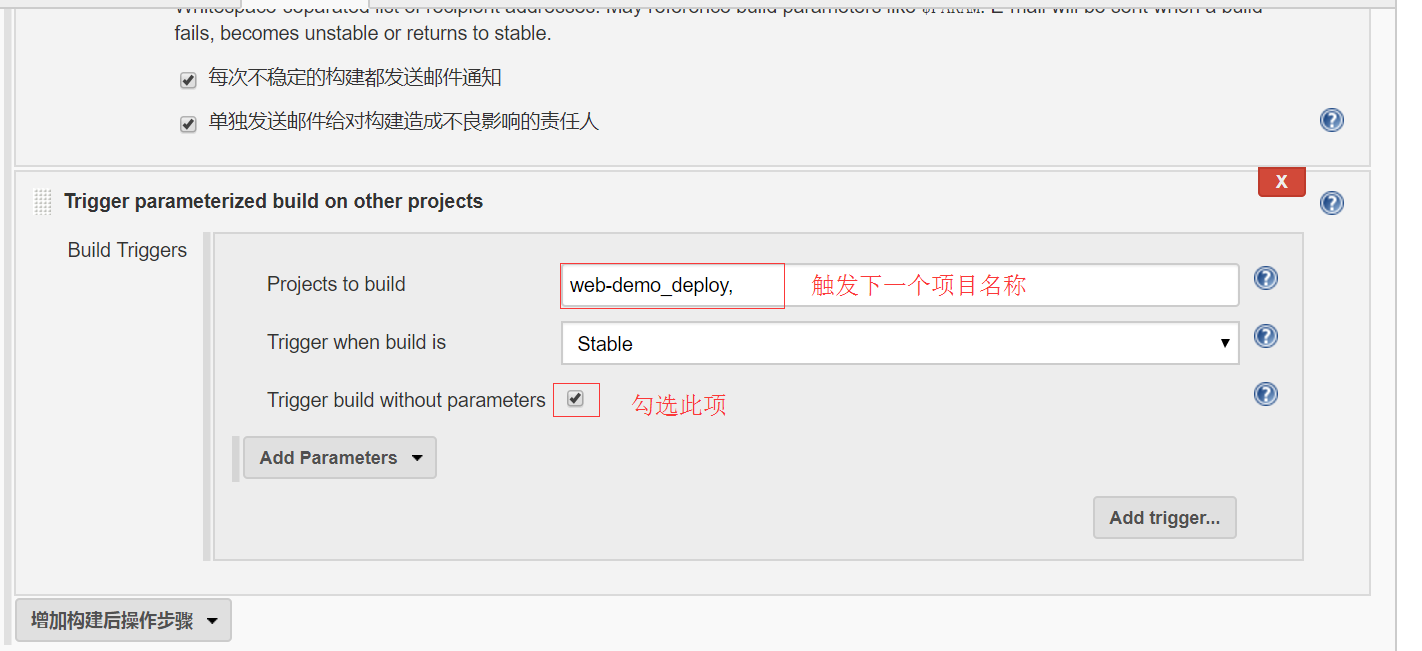
如果插件在线安装不成功可以下载插件到此目录然后把属主属组改成jenkins再重启jenkins服务即可完成安装：  
#系统管理-管理插件-可选插件，搜索Parameterized：



1.4.2：配置项目demo的构建后操作，demo构建完成后自动构建demp-deploy项目：  
#jenkins-->web-demo-->配置-->构建后操作：



1.4.3：配置如下：



1.5：测试，执行代码测试项目成功之后是否会自动执行代码部署项目：

控制台输出

IMG_274

Started by user admin

Building in workspace /home/jenkins/.jenkins/workspace/web-demo> git rev-parse --is-inside-work-tree # timeout=10

Fetching changes from the remote Git repository> git config remote.origin.url git@192.168.3.198:web/web-demo.git # timeout=10

Fetching upstream changes from git@192.168.3.198:web/web-demo.git> git --version # timeout=10

using GIT\_SSH to set credentials gitlab\_web-demo> git fetch --tags --progress git@192.168.3.198:web/web-demo.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 b8f3be4385efdf64606158c23f9f1992bb2da1d3 (refs/remotes/origin/master)

Commit message: "add www.chinasoft.com"

> git config core.sparsecheckout # timeout=10

> git checkout -f b8f3be4385efdf64606158c23f9f1992bb2da1d3

> git rev-list b8f3be4385efdf64606158c23f9f1992bb2da1d3 # timeout=10

[web-demo] $ /usr/local/sonar-scanner/bin/sonar-scanner -e -Dsonar.host.url=http://192.168.3.199:9000/ -Dsonar.language=php -Dsonar.projectName=web-demo -Dsonar.projectVersion=1.0 -Dsonar.sourceEncoding=UTF-8 -Dsonar.projectKey=web-demo -Dsonar.sources=./ -Dsonar.projectBaseDir=/home/jenkins/.jenkins/workspace/web-demo

INFO: Scanner configuration file: /usr/local/sonar-scanner/conf/sonar-scanner.properties

INFO: Project root configuration file: NONE

INFO: SonarQube Scanner 2.6.1

INFO: Java 1.8.0\_111 Oracle Corporation (64-bit)

INFO: Linux 3.10.0-514.el7.x86\_64 amd64

INFO: Error stacktraces are turned on.

INFO: User cache: /home/jenkins/.sonar/cache

INFO: Load global repositories

INFO: Load global repositories (done) | time=172ms

WARN: Property 'sonar.jdbc.url' is not supported any more. It will be ignored. There is no longer any DB connection to the SQ database.

WARN: Property 'sonar.jdbc.username' is not supported any more. It will be ignored. There is no longer any DB connection to the SQ database.

WARN: Property 'sonar.jdbc.password' is not supported any more. It will be ignored. There is no longer any DB connection to the SQ database.

INFO: User cache: /home/jenkins/.sonar/cache

INFO: Load plugins index

INFO: Load plugins index (done) | time=3ms

INFO: SonarQube server 5.6.6

INFO: Default locale: "en\_US", source code encoding: "UTF-8"

INFO: Process project properties

INFO: Load project repositories

INFO: Load project repositories (done) | time=97ms

INFO: Load quality profiles

INFO: Load quality profiles (done) | time=34ms

INFO: Load active rules

INFO: Load active rules (done) | time=380ms

WARN: SCM provider autodetection failed. No SCM provider claims to support this project. Please use sonar.scm.provider to define SCM of your project.

INFO: Publish mode

INFO: ------------- Scan web-demo

INFO: Language is forced to php

INFO: Load server rules

INFO: Load server rules (done) | time=71ms

INFO: Base dir: /home/jenkins/.jenkins/workspace/web-demo

INFO: Working dir: /home/jenkins/.jenkins/workspace/web-demo/.sonar

INFO: Source paths: .

INFO: Source encoding: UTF-8, default locale: en\_US

INFO: Index files

INFO: 0 files indexed

INFO: Quality profile for php: Sonar way

INFO: Sensor Lines Sensor

INFO: Sensor Lines Sensor (done) | time=0ms

INFO: Sensor SCM Sensor

INFO: No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.

INFO: Sensor SCM Sensor (done) | time=0ms

INFO: Sensor Analyzer for "php.ini" files

INFO: Sensor Analyzer for "php.ini" files (done) | time=3ms

INFO: Sensor SonarJavaXmlFileSensor

INFO: Sensor SonarJavaXmlFileSensor (done) | time=0ms

INFO: Sensor Zero Coverage Sensor

INFO: Sensor Zero Coverage Sensor (done) | time=0ms

INFO: Sensor Code Colorizer Sensor

INFO: Sensor Code Colorizer Sensor (done) | time=0ms

INFO: Sensor CPD Block Indexer

INFO: DefaultCpdBlockIndexer is used for php

INFO: Sensor CPD Block Indexer (done) | time=0ms

INFO: Calculating CPD for 0 files

INFO: CPD calculation finished

INFO: Analysis report generated in 47ms, dir size=8 KB

INFO: Analysis reports compressed in 7ms, zip size=3 KB

INFO: Analysis report uploaded in 47ms

INFO: ANALYSIS SUCCESSFUL, you can browse http://192.168.3.199:9000/dashboard/index/web-demo

INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report

INFO: More about the report processing at http://192.168.3.199:9000/api/ce/task?id=AV0YJcbrykzBCcoFv4Mt

INFO: ------------------------------------------------------------------------

INFO: EXECUTION SUCCESS

INFO: ------------------------------------------------------------------------

INFO: Total time: 43.045s

INFO: Final Memory: 42M/137M

INFO: ------------------------------------------------------------------------

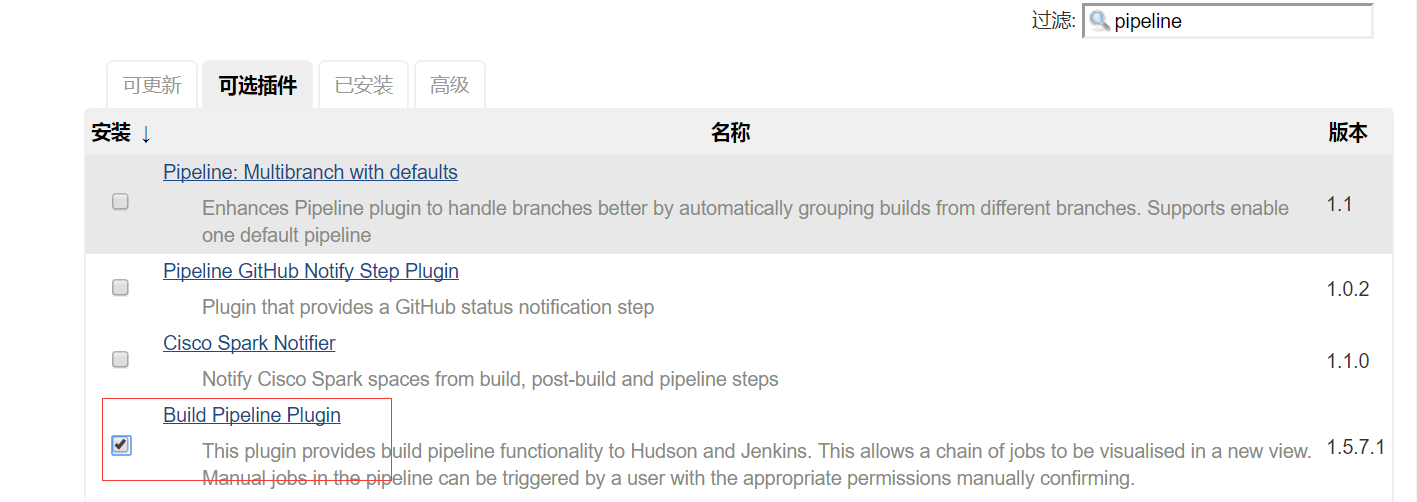
Warning: you have no plugins providing access control for builds, so falling back to legacy behavior of permitting any downstream builds to be triggered

Triggering a new build of web-demo\_deploy

Finished: SUCCESS

IMG_275

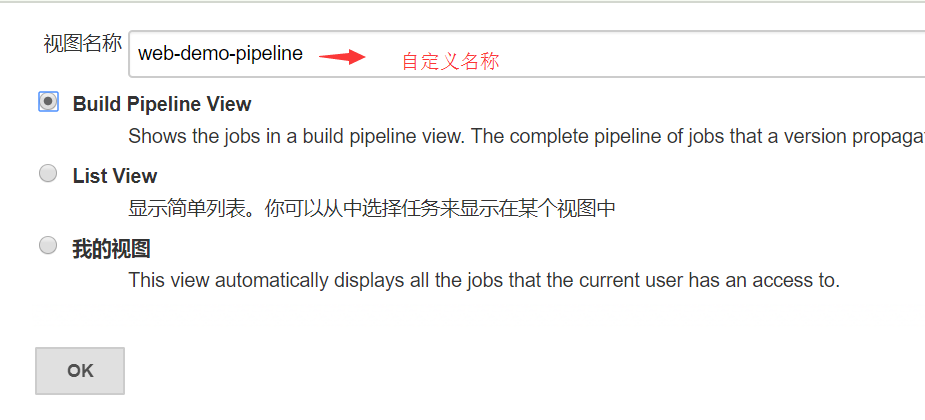
1.6：pipeline插件：  
1.6.1：#安装插件，系统管理-管理插件-可安装插件：

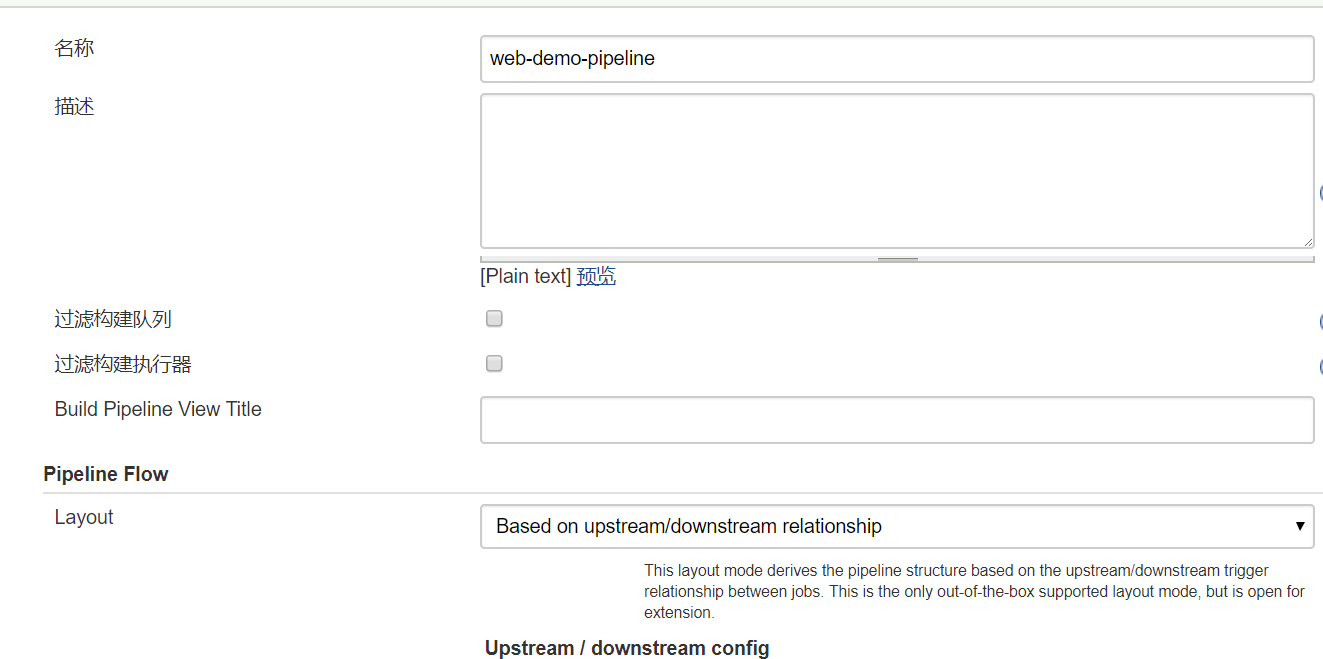


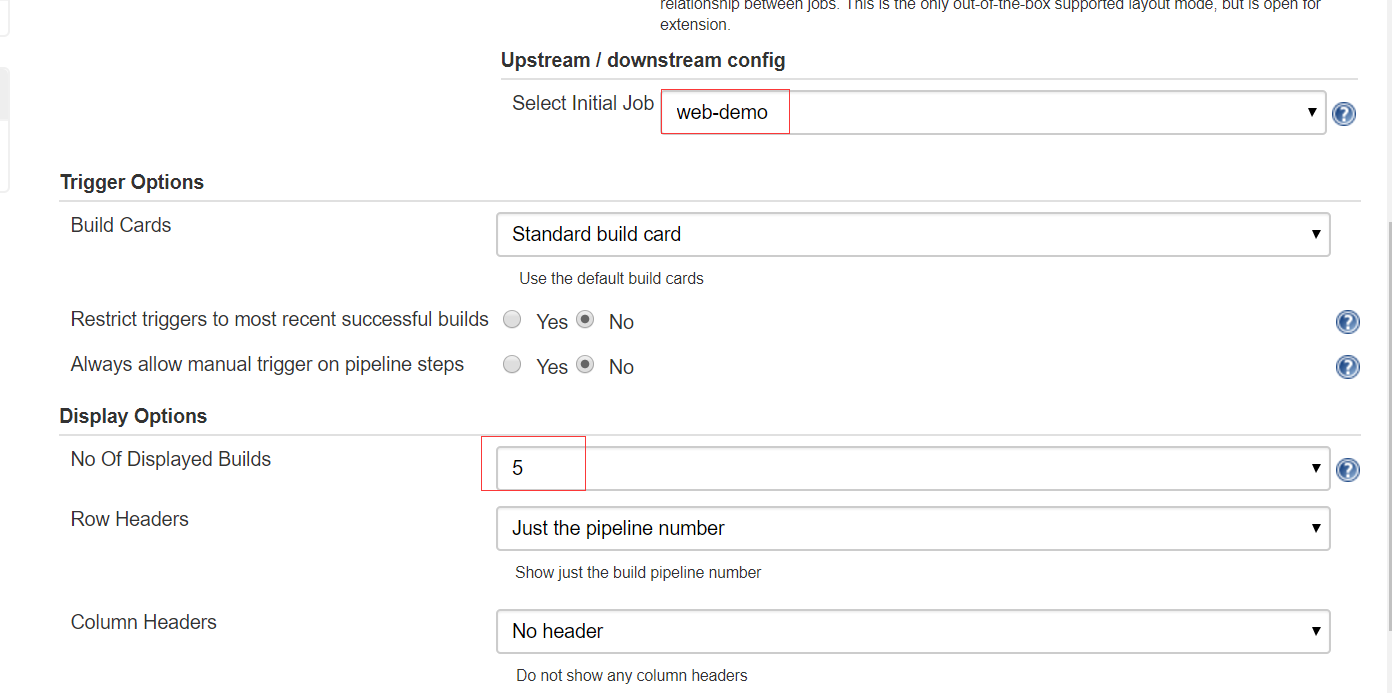
1.6.2：创建视图：

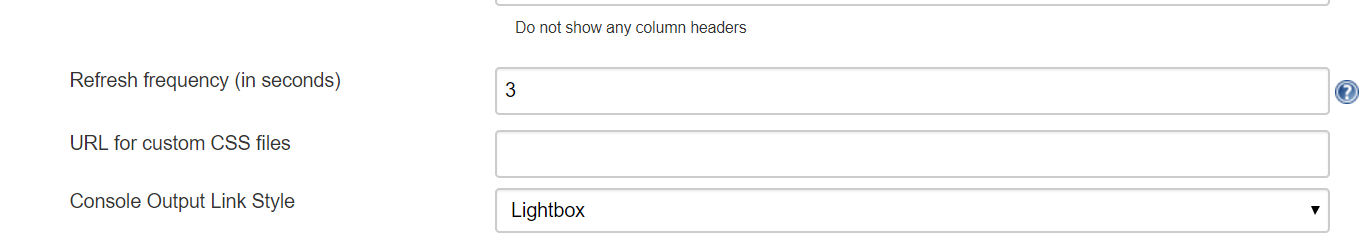


1.6.3:自定义名称：









1.6.4:配置pipeline信息,点击OK之后，弹出如下视图

点击保存之后显示的最终界面：

