|  |
| --- |
| shell编程第一周 |
| **作者：王有坤**  **归档：学习笔记**  **更新yum源**    **2018/6/7** |
| **快捷键：**  Ctrl + 1 标题1  Ctrl + 2 标题2  Ctrl + 3 标题3  Ctrl + 4 实例  Ctrl + 5 程序代码  Ctrl + 6 正文 |
| **格式说明：**  蓝色字体：注释  黄色背景：重要  绿色背景：注意 |

****

**老男孩教育教学核心思想6重：重目标、重思路、重方法、重实践、重习惯、重总结**

**学无止境，老男孩教育成就你人生的起点！**

**联系方式:**

|  |  |
| --- | --- |
| **网站运维QQ交流群：** | |
| Linux 385168604 | 架构师 390642196 |
| Python 29215534 | 大数据 421358633 |
| **官方网站:** | |
| [**http://www.oldboyedu.com**](http://www.oldboyedu.com) |  |

目 录

# git和github的学习

## git命令

第一步:git的基础设置,这一步是必须的,否则后面的所有操作都会提示你是谁,你没有权限操作.

git config --global [user.name](qq://txfile/) wangyoukun #设置用户

git config --global user.email [1064043323@qq.com](mailto:1064043323@qq.com) #设置邮箱

git config –list #查看全局设置的结果,以下是我设置的结果情况

user.name=wangyoukun

user.email=1064043323@qq.com

core.repositoryformatversion=0

core.filemode=true

core.bare=false

core.logallrefupdates=true

第二步:创建git管理的目录

在家目录下面创建git管理的目录

mkdir git\_data

第三步:参考git2.0讲义

git init

git status 查看当前的状态

[root@node1 git\_data]# git status

On branch dev

nothing to commit, working tree clean #这是默认的没有问题的提交结果情况

touch a.txt #创建一个文件

git add a.txt #将文件提交到暂存区域

git rm --cached a.txt

[root@node1 git\_data]# ll -a

total 16

drwxr-xr-x 3 root root 70 Jun 9 14:56 .

dr-xr-x---. 8 root root 264 Jun 11 09:11 ..

-rw-r--r-- 1 root root 8 Jun 9 11:46 a.txt

-rw-r--r-- 1 root root 6 Jun 9 11:39 b.txt

-rw-r--r-- 1 root root 12 Jun 9 14:56 c.txt

-rw-r--r-- 1 root root 4 Jun 9 15:08 d.txt

drwxr-xr-x 8 root root 183 Jun 11 09:18 .git #.git默认就是本地仓库

[root@node1 git\_data]# ll .git

total 28

drwxr-xr-x 2 root root 6 Jun 9 09:52 branches

-rw-r--r-- 1 root root 20 Jun 9 15:10 COMMIT\_EDITMSG

-rw-r--r-- 1 root root 92 Jun 11 09:16 config

-rw-r--r-- 1 root root 73 Jun 9 09:52 description

-rw-r--r-- 1 root root 20 Jun 9 15:07 HEAD

drwxr-xr-x 2 root root 242 Jun 9 09:52 hooks

-rw-r--r-- 1 root root 353 Jun 9 15:10 index #默认就是本地仓库的暂存区域

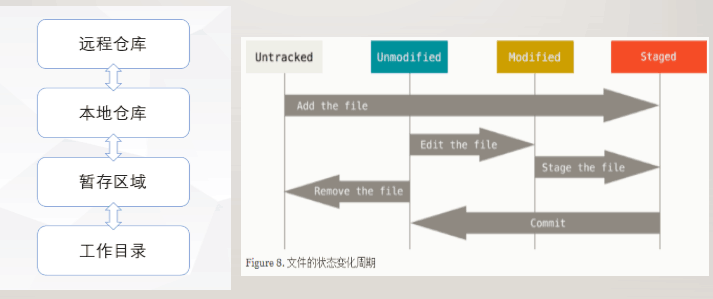
drwxr-xr-x 2 root root 21 Jun 9 09:52 info

drwxr-xr-x 3 root root 30 Jun 9 10:10 logs

drwxr-xr-x 40 root root 4096 Jun 9 15:10 objects

-rw-r--r-- 1 root root 41 Jun 9 14:41 ORIG\_HEAD

drwxr-xr-x 4 root root 31 Jun 9 09:52 refs





git checkout -- a.txt 恢复文件在暂存区的状态

#这个命令什么时候用呢?

第一步:当文件a.txt文件被创建,并且追加过文件内容,后来执行了git add a.txt命令

第二步:此时git status会发现提示a.txt文件是暂存状态

第三步:如果此时再重新编辑文件,添加新的文件内容的话,查看文件状态git status就会发现同时存在着暂存状态和提示未提交的更改

第四步:这个时候我发现文件被我修改错了,我想回到最近的一次状态,即暂存状态时候的文件,执行git ckeckout – a.txt 命令.执行完这个命令之后,就会发现文件撤销成功了,就是之前自己想要的哪个状态.

git rm -f c.txt 删除文件在暂存区的状态

这个命令什么时候用呢?

第一步;当我不想要这个文件,但是这个文件又被我执行了git add c.txt命令了

第二步:执行git rm –f c.txt命令,此时查看状态git status发现c.txt文件没有被暂存,只是提示文件没有被追踪

第三步:此时执行ll命令发现,文件没有被真正删除,只是文件在暂存区域被删除

git mv a1.txt a.txt 暂存区修改文件名并不影响文件的状态

第一步:执行修改文件名的命令git mv a1.txt a.txt

第二步:git status查看文件没有发生状态变化,即修改之后的文件还是暂存的状态

git log 查看commit提交的日志,这个命令显示结果最为详细但是过于累赘

git log –oneline #这条命令显示的只有hash值的前几位数值,和commit时编辑的描述信息 `–m “描述”`

[root@node1 git\_data]# git log --oneline

0802f09 modify on big

a4db9c8 modify first on big branch

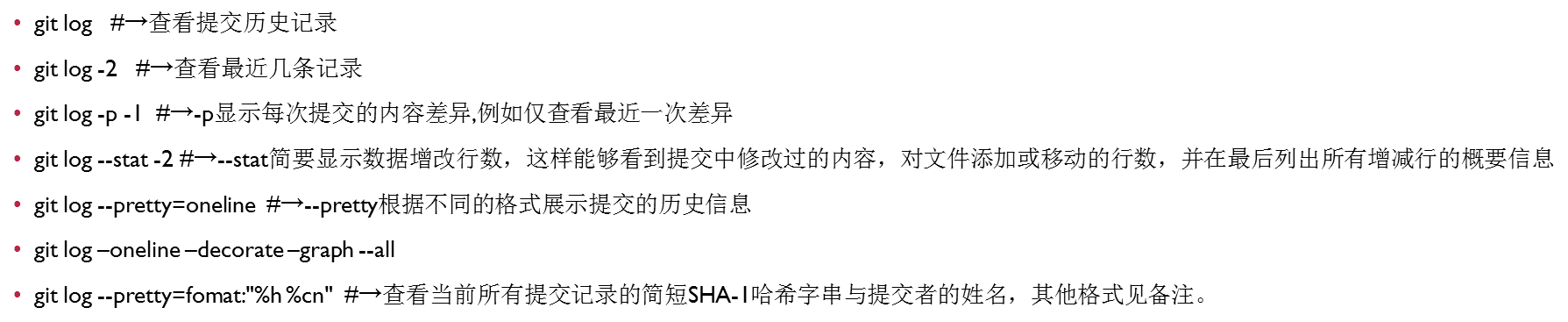
91d093e merge dev c.txt to master

c965f54 modify c.txt on master

eb9b1f8 modify c.txt

3942076 Merge branch 'dev'

7a54252 modify c.txt on dev branch



git reset --hard HEAD #和git reflog命令一般是结合使用的

git reflog

什么时候用到这个命令?

当你执行了某个操作,并且已经提交了这个操作,此时是不是无法回到之前的那个状态了?

答案当然是否定的.那就是我们为什么使用git管理版本控制系统?

我们执行这个命令之后可以查看所有的hash值和对应的head值,checkout和描述信息

我们想回到某个状态只需要切换到某个状态就可以了

执行的命令就是Git reset –hard HEAD

其中HEAD就是产生的hash值,例如下面的命令结果

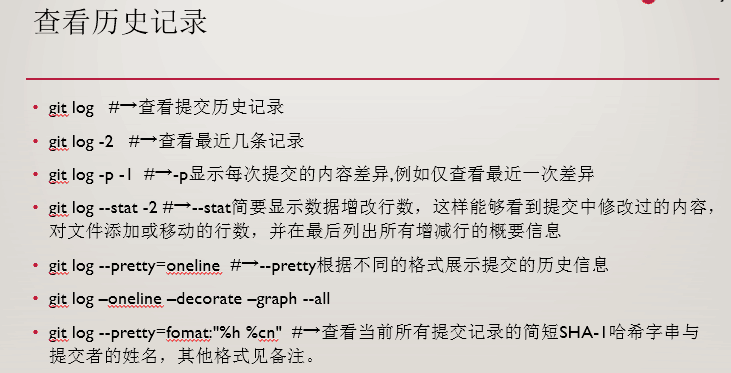
[root@node1 git\_data]# git reset --hard ccd192d

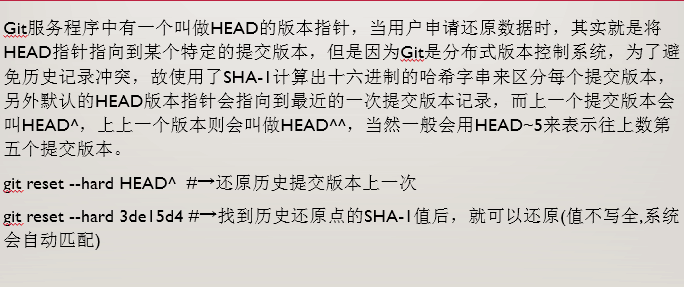
此时就有一个疑问了,我们为什么不直接使用git log –oneline获取hash值再去执行呢?

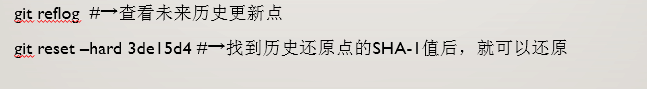
原因只有一个:那就是这条命令根本无法满足我们的使用.git log –oneline 只能满足查看之前的所有commit的hash值,无法查看执行了git reset –hard HEAD之后的所有commit的hash值结果



按照系统的git add提示  -------🡪段大哥提供







git diff 区别文件修改

git diff –cached

[root@node1 git\_data]# echo "123">>nihao.txt

[root@node1 git\_data]# git diff

diff --git a/nihao.txt b/nihao.txt

index 5c0ac06..692d3f9 100644

--- a/nihao.txt

+++ b/nihao.txt

@@ -1 +1,2 @@

00000

+123

git tag v1.0 打标签

[root@node1 git\_data]# git tag v3.0 0802f09

[root@node1 git\_data]# git tag #查看所有的标签

V2.0

V3.0

V4.0

V5.11

v3.0

[root@node1 git\_data]# git show v3.0 #查看v3.0的状态

commit 0802f096cba7f97454adbce82f3b8741ab94afc3

Author: wangyoukun <1064043323@qq.com>

Date: Mon Jun 11 09:41:03 2018 +0800

modify on big

diff --git a/e.txt b/e.txt

index b10664a..a7ac935 100644

--- a/e.txt

+++ b/e.txt

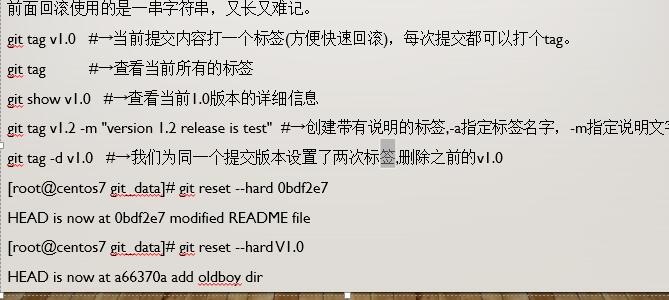
@@ -1 +1,2 @@

[root@node1 git\_data]# git tag -d V5.11 #删除版本

Deleted tag 'V5.11' (was b3220d9)

[root@node1 git\_data]# git reset --hard V3.0 #切换状态更加便捷

HEAD is now at c5d19ad modify a.txt second



git branch #查看当前在哪个分支上面

[root@node1 git\_data]# git branch

\* big

dev

master

git branch dev #创建dev分支

[root@node1 git\_data]# git branch small #创建small分支

[root@node1 git\_data]# git branch #查看当前在那个分支上面

\* big

dev

master

small

git checkout master #切换分支到master分支上面

[root@node1 git\_data]# git checkout small

Switched to branch 'small'

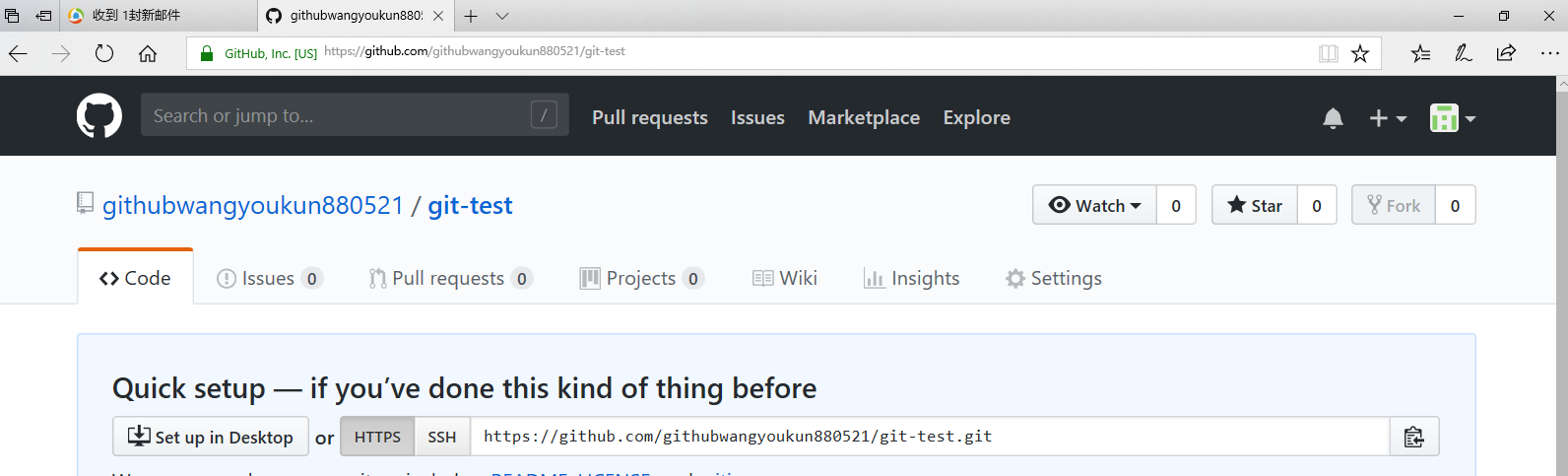
## 本地仓库的数据到远程仓库GitHub的部署过程

第一步:注册,并进行登录

<https://github.com/join/plan>

访问页面进行注册,系统会发送一个邮件到自己的邮箱,到邮箱进行登录https://github.com/githubwangyoukun880521/git-test

第二步:正常登录之后会出现如下页面,创建git-test





第三步:linux服务器上创建远程登陆仓库

[root@node1 git\_data]# git remote add origin [git@github.com:githubwangyoukun880521/git-test.git](mailto:git@github.com:githubwangyoukun880521/git-test.git)

[root@node1 git\_data]# git remote -v

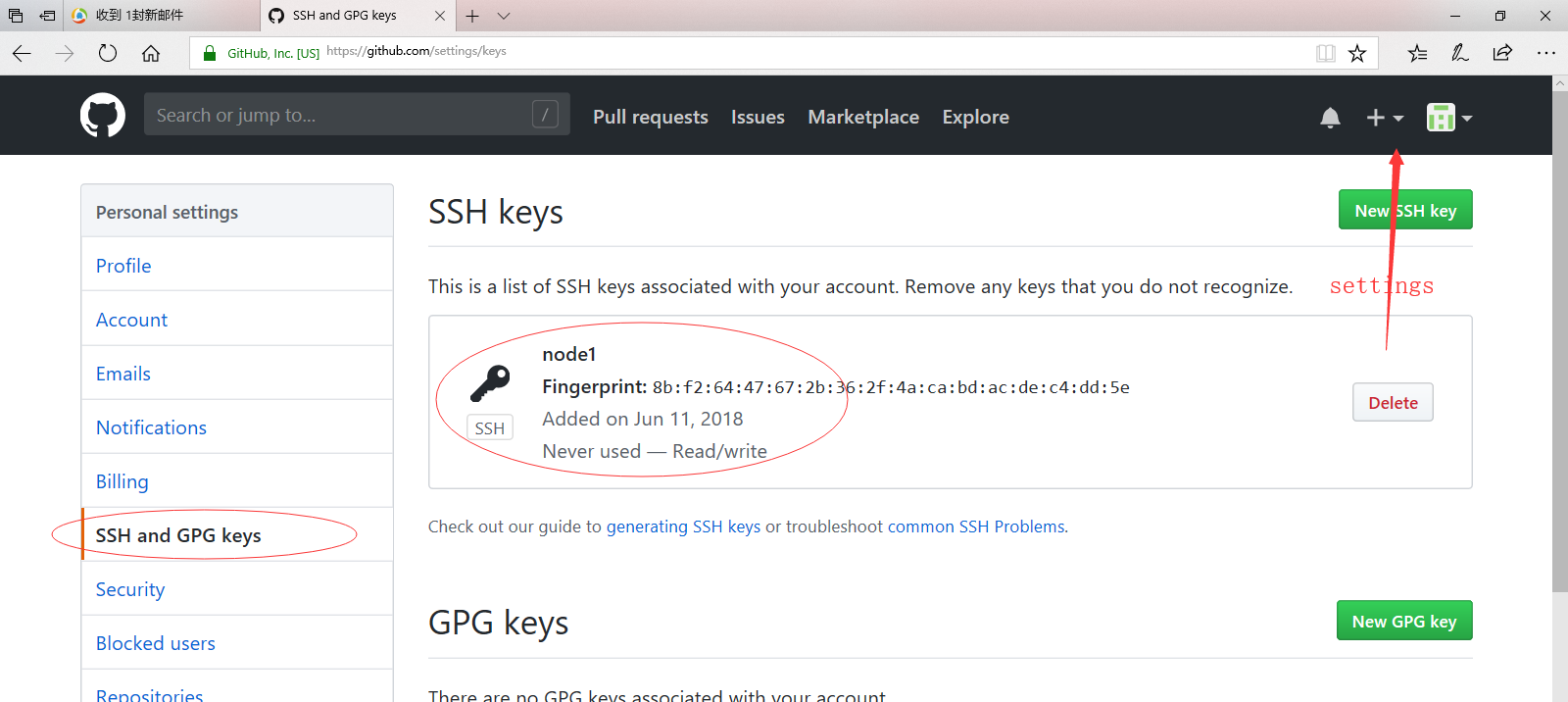
origin git@github.com:githubwangyoukun880521/git-test.git (fetch)

origin git@github.com:githubwangyoukun880521/git-test.git (push)

第四步:linux服务器上创建秘钥对,并添加到远程仓库上

[root@node1 .ssh]# ssh-keygen -t rsa

[root@node1 .ssh]# cat id\_rsa.pub



验证能否正常连接远程仓库,出现下面的黄色区域的内容,表明能够正常的访问远程仓库了

[root@node1 git-test]# ssh -T git@github.com

The authenticity of host 'github.com (52.74.223.119)' can't be established.

RSA key fingerprint is SHA256:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8.

RSA key fingerprint is MD5:16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'github.com,52.74.223.119' (RSA) to the list of known hosts.

Hi githubwangyoukun880521! You've successfully authenticated, but GitHub does not provide shell access.

第五步:准备发送文件到远程仓库,查看最终结果

[root@node1 git-test]# for name in `seq 10`;do touch oldboy${name}.txt ;done

[root@node1 git-test]# for name in oldboy{1..10}.txt;do git add .;done

[root@node1 git-test]# git commit -m "commit oldboy ok"

参照ssh下面的提示信息输入到命令行

[root@node1 git-test]# git remote add origin git@github.com:githubwangyoukun880521/git-test.git

[root@node1 git-test]# git push -u origin master

Counting objects: 3, done.

Compressing objects: 100% (2/2), done.

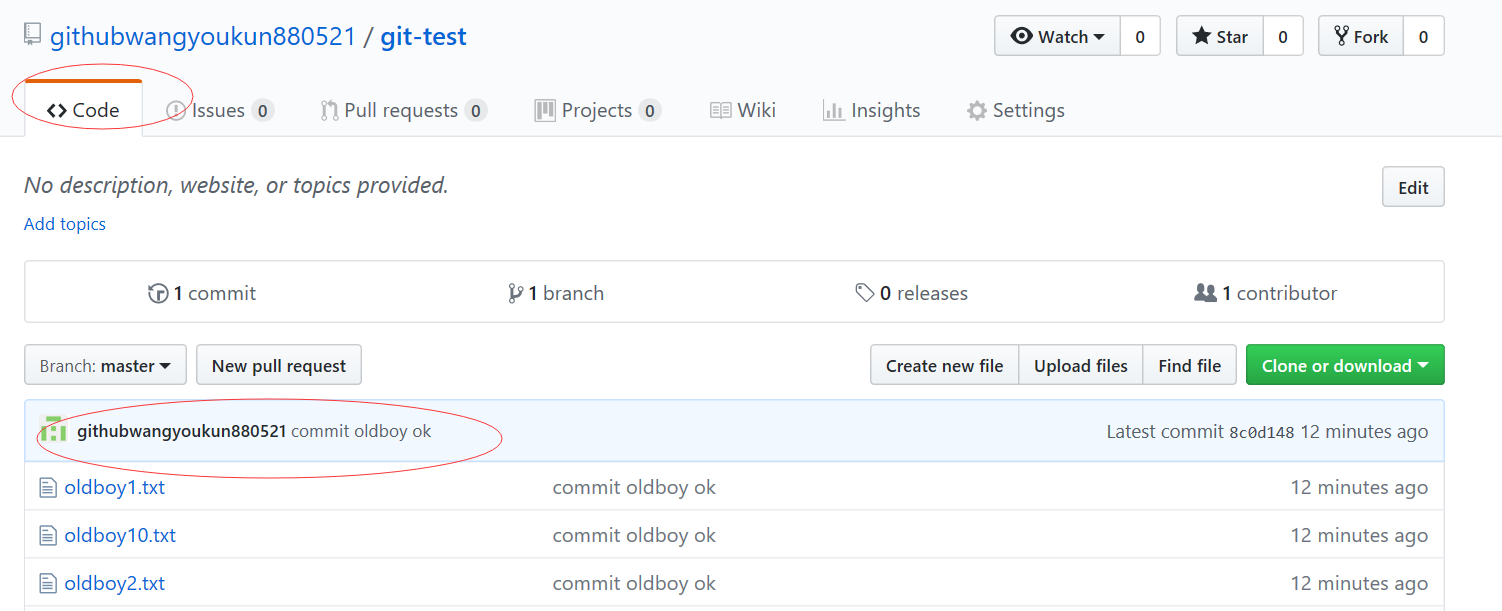
Writing objects: 100% (3/3), 249 bytes | 0 bytes/s, done.

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

To github.com:githubwangyoukun880521/git-test.git

\* [new branch] master -> master

Branch master set up to track remote branch master from origin.



结果发现文件能够正常发送到远程仓库中去了

第六步:重新写一个文件并发送到远程仓库中去,查看最终结果

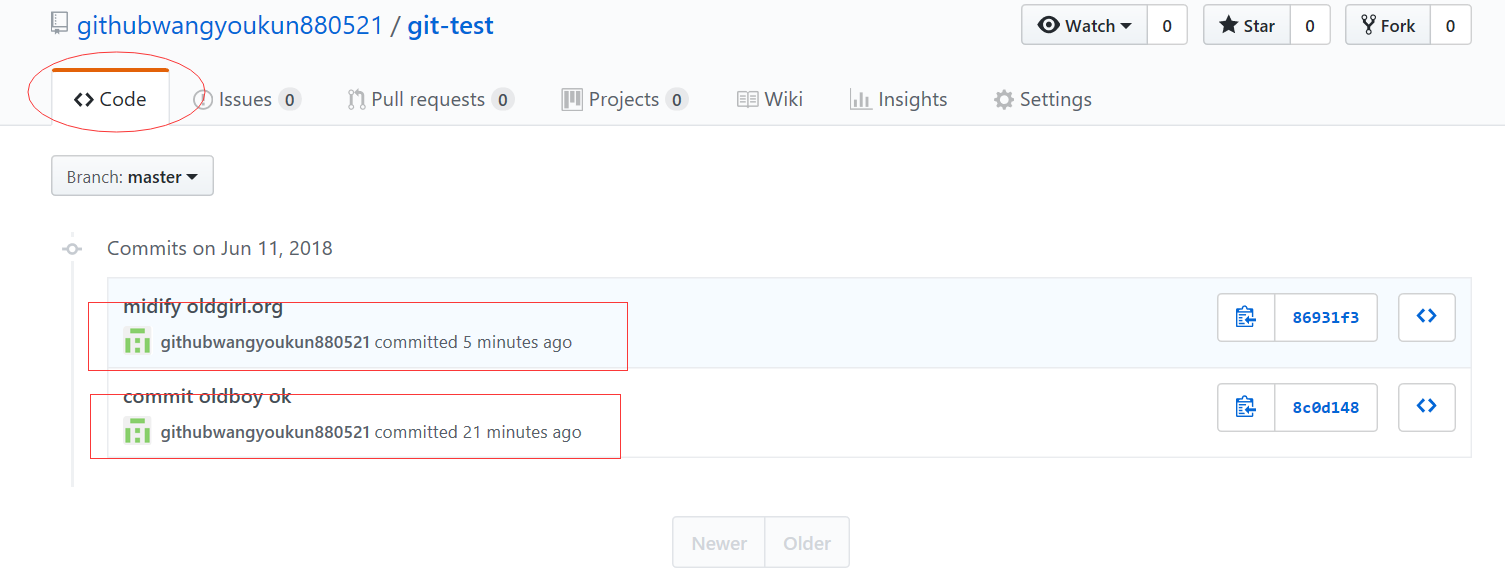
[root@node1 git-test]# touch oldgirl.org

[root@node1 git-test]# git add oldgirl.org

[root@node1 git-test]# echo "your name is oldboy my name is oldgirl ,we are friends" >oldgirl.org

[root@node1 git-test]# git commit -a -m "midify oldgirl.org"

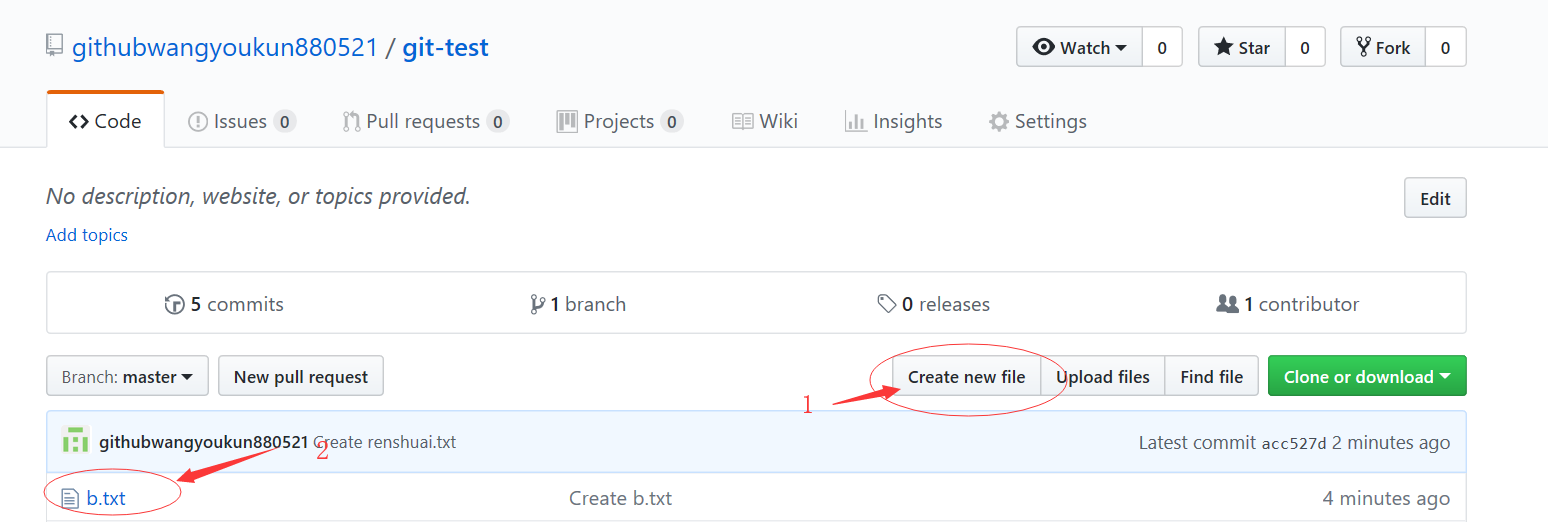
[root@node1 git-test]# git push -u origin master #将本地数据文件发送到远程仓库中去



## 远程仓库GitHub数据到本地仓库的部署过程

### 从远程仓库拉取数据文件到本地仓库的方式一

第一步:在远程仓库上创建文件



第二步:在本地服务器上拉取文件

先拉取文件,再合并文件

[root@node1 git-test]# git fetch origin #拉取文件的命令

[root@node1 git-test]# git merge origin/master #进行合并

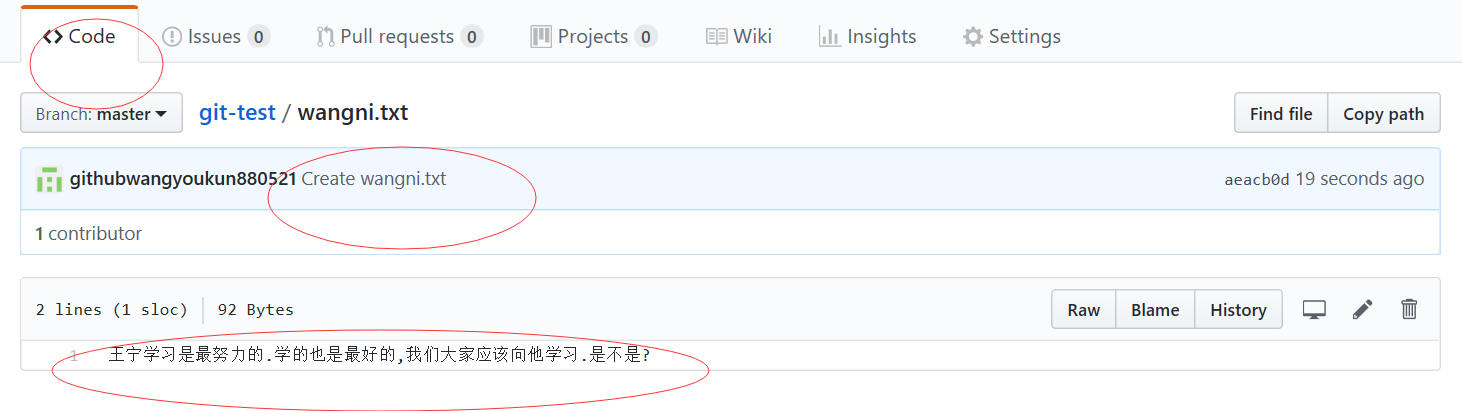
第三步:本地服务器上进行查看结果

[root@node1 git-test]# ll b.txt

-rw-r--r-- 1 root root 1 Jun 11 15:52 b.txt

### 从远程仓库拉取数据文件到本地仓库的方式二

第一步:在远程仓库上创建文件



第二步:本地服务器进行拉取文件内容,查看最终结果 直接拉取文件并且合并文件只是一个简短的命令就行了.

[root@node1 git-test]# git pull

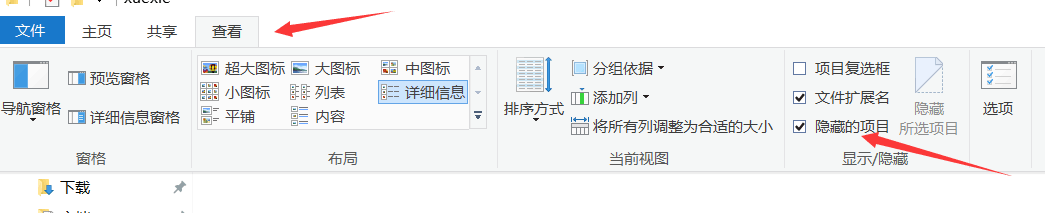
[root@node1 git-test]# ll wangni.txt

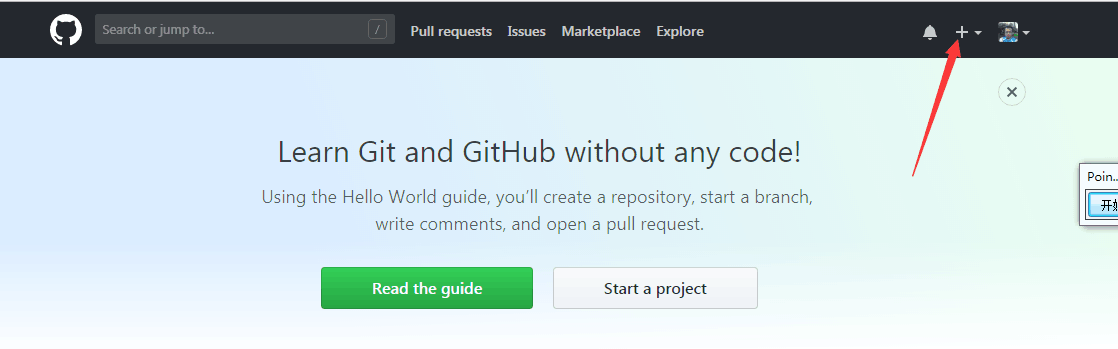
-rw-r--r-- 1 root root 92 Jun 11 16:04 wangni.txt

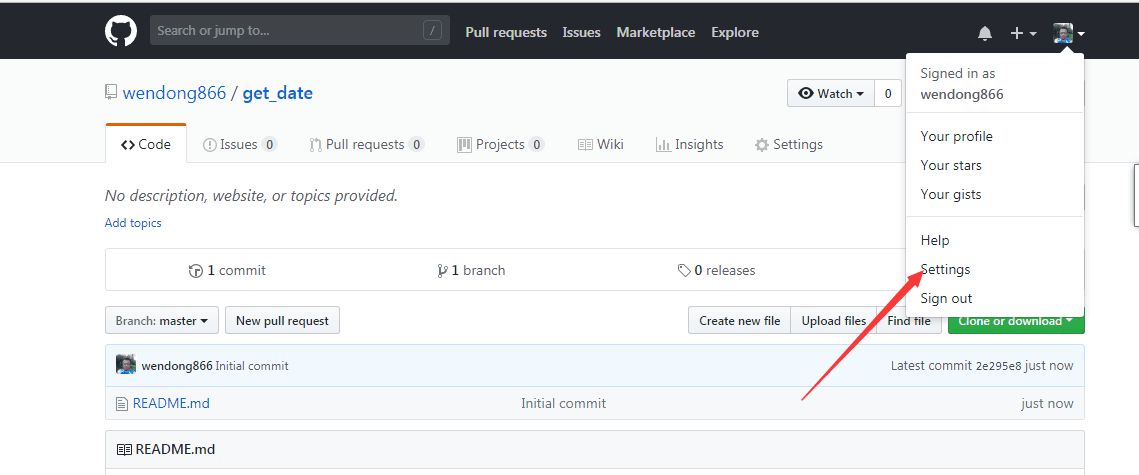
[root@node1 git-test]# cat wangni.txt

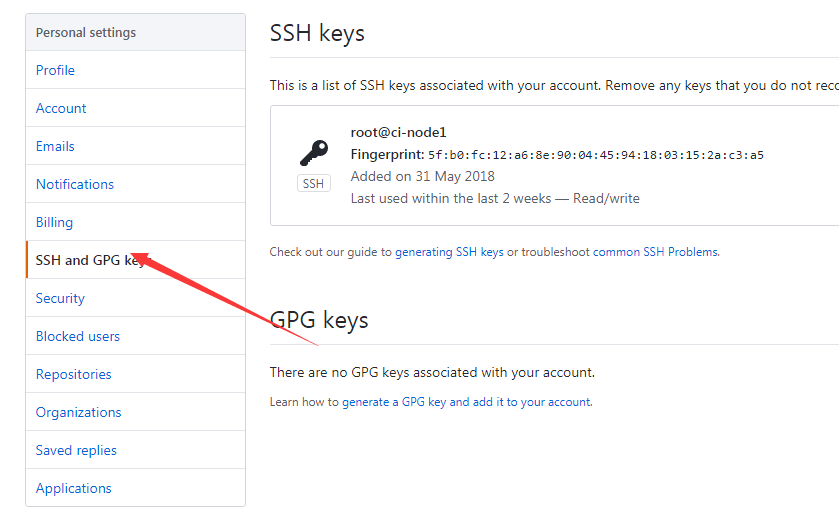
王宁学习是最努力的.学的也是最好的,我们大家应该向他学习.是不是?

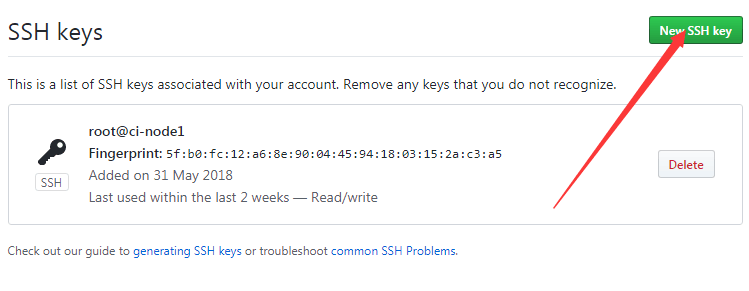
## window上的git部署过程

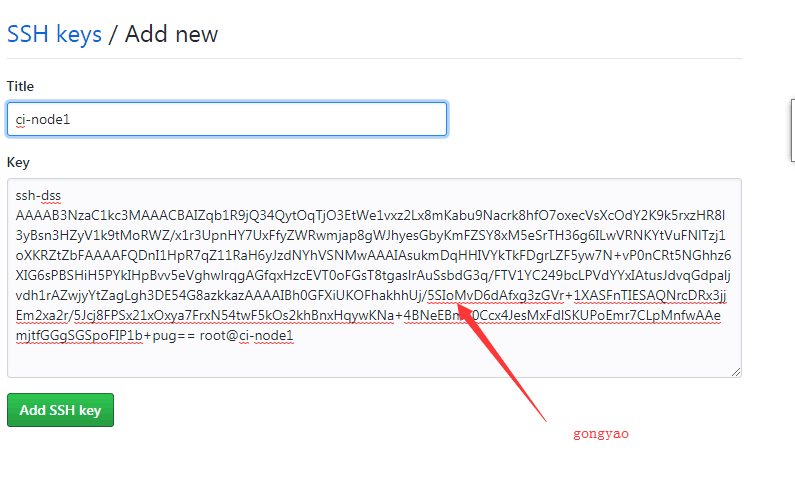












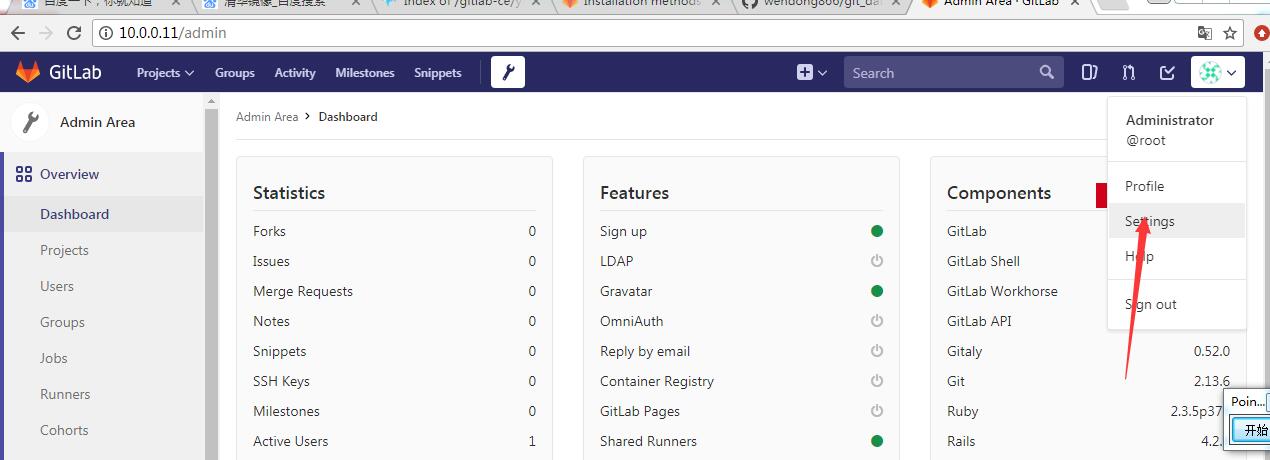
# gitlab和jenkins的学习

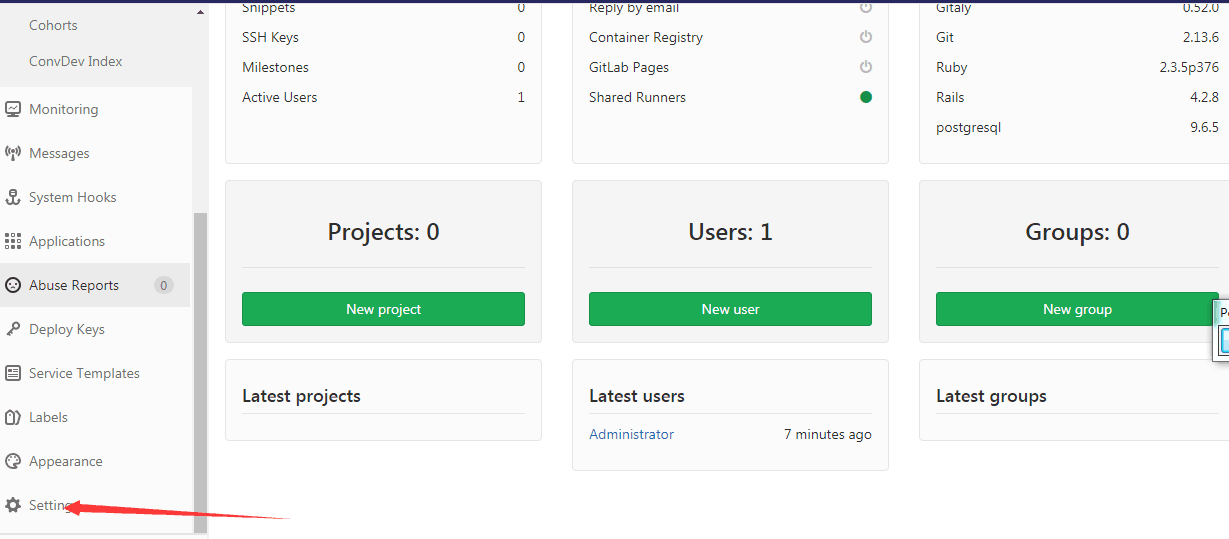
ssh-keygen -t dsa

cat .ssh/id\_dsa

git remote add origin

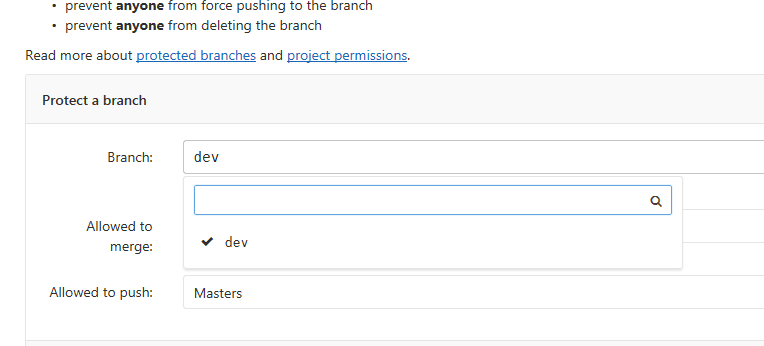
git push -u origin master

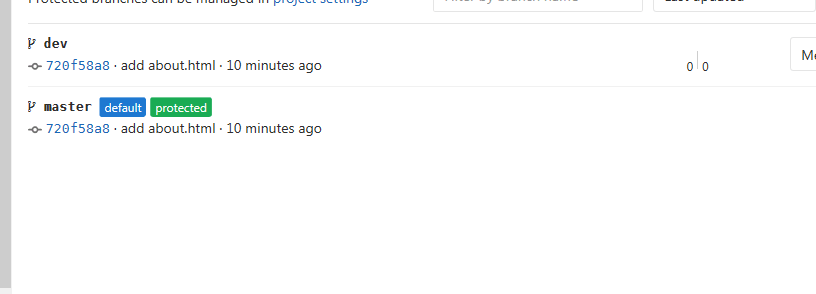


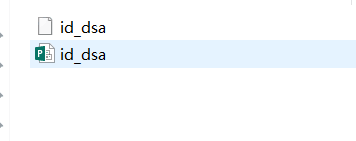


<html>  
<title>gitlab</title>  
<body>  
gitlab test  
</body>  
<html>

<html>  
<title>about</title>  
<body>  
about gitlab test  
</body>  
<html>





张晟阳提供的错误展示

gitlab\_rails['backup\_path'] = '/data/backup/gitlab'  
gitlab\_rails['backup\_keep\_time'] = 604800

gitlab-ctl reconfigure



/usr/bin/gitlab-rake gitlab:backup:create

停止数据写入服务  
gitlab-ctl stop unicorn  
gitlab-ctl stop sidekiq  
gitlab-rake gitlab:backup:restore BACKUP=1512811475\_2017\_12\_09\_10.2.2  
gitlab-ctl restart

[https://gitee.com/kangjie1209/monitor.git](qq://txfile/)

java –version

rpm -ql jenkins

#jbcrypt:$2a$10$l.8t61ivmL1m77d1369GruJbNyJeEHvOuqnR48QHRq20oeZyD  
gbvu

freestyle-job

/var/lib/jenkins/workspace

tar czf /opt/web-$(date +%F%T).tar.gz ./

ssh-copy-id -i /root/.ssh/id\_dsa.pub [10.0.0.11](qq://txfile/)

scp -P22 /opt/web-$(date +%F).tar.gz [10.0.0.11](qq://txfile/):/opt

tar czf /opt/web-$(date +%F).tar.gz ./

yum install httpd –y

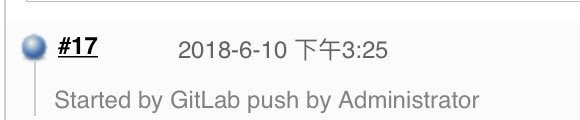
vim /etc/httpd/conf/httpd.conf 修改默认的端口

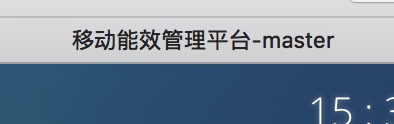
systemctl restart httpd

/var/www/html

ssh -p22 [10.0.0.11](qq://txfile/) 'cd /var/www/html && tar xf web-$(date +%F).tar.gz'

[https://gitee.com/kangjie1209/monitor.git](qq://txfile/)



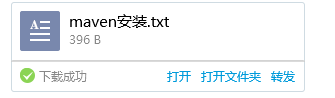




tar xf apache-maven-3.3.9-bin.tar.gz

mv apache-maven-3.3.9 ../

ln -s /usr/local/apache-maven-3.3.9 /usr/local/maven



/usr/local/maven

clean package

pipeline{

agent any

stages{

stage("get code"){

steps{

echo "get code"

}

}

stage("build"){

steps{

echo "build"

}

}

}

}

tar czf /opt/web-$(date +%F).tar.gz ./

pipeline{  
    agent any  
    stages{  
        stage("get code"){  
            steps{  
                checkout([$class: 'GitSCM', branches: [[name: '\*/master']], doGenerateSubmoduleConfigurations: false, extensions: [], submoduleCfg: [], userRemoteConfigs: [[credentialsId: 'jenkins-private', url: 'git@10.0.0.11:oldboy/monitor.git']]])  
            }  
        }  
          
        stage("package"){  
            steps{  
                sh 'tar czf /opt/web-$(date +%F).tar.gz ./'  
                  
            }  
        }  
          
        stage("deploy"){  
            steps{  
                sh 'scp -P22 /opt/web-$(date +%F).tar.gz [10.0.0.11](qq://txfile/):/var/www/html'  
                sh 'ssh -p22 [10.0.0.11](qq://txfile/) \'cd /var/www/html && tar xf web-$(date +%F).tar.gz\''  
            }  
        }  
    }  
}



mkdir /data/jenkins -p