# 一、状态status+查修改diff

* 要随时掌握工作区的状态，使用git status命令。
* 如果git status告诉你有文件被修改过，用git diff可以查看修改内容

## 1.1、git diff

git diff #是工作区(work dict)和暂存区(stage)的比较

git diff --cached #是暂存区(stage)和分支(master)的比较

git diff HEAD #查看工作区和版本库里面最新版本的区别

修改后查看状态： modified: git.txt

Dit diff: 工作区与缓存区比较，有区别（修改后未add）

$ git status

On branch master

Changes not staged for commit:

(use "git add ..." to update what will be committed)

(use "git checkout -- ..." to discard changes in working directory)

modified: git.txt

Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)

$ git diff

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

index bb7fdf0..7d9fb6e 100644

--- a/git/git.txt

+++ b/git/git.txt

@@ -1,2 +1,4 @@

Add new one

add two

+

+git diff

\ No newline at end of file

将修改添加到缓存区，查看状态: modified: git.txt

Git diff: 工作区和缓存区，此时无区别(修改后，已add, 工作区=缓存区)

Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)

$ git add git.txt

Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD ..." to unstage)

modified: git.txt

|  |
| --- |
| Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)  $ git diff  查看缓存区与版本库的区别：（已add未commit -m）  Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)  $ git diff --cached  diff --git a/git/git.txt b/git/git.txt  index bb7fdf0..7d9fb6e 100644  --- a/git/git.txt  +++ b/git/git.txt  @@ -1,2 +1,4 @@  Add new one  add two  +  +git diff  \ No newline at end of file  查看工作区与版本库的区别：（已add未commit -m）  Administrator@Nancy MINGW64 /e/Nancyxiao/study/git (master)  $ git diff HEAD  diff --git a/git/git.txt b/git/git.txt  index bb7fdf0..7d9fb6e 100644  --- a/git/git.txt  +++ b/git/git.txt  @@ -1,2 +1,4 @@  Add new one  add two  +  +git diff  \ No newline at end of file |
| 补充：  d:\Documents\Tencent Files\768445005\FileRecv\MobileFile\Image\SB%`_98H1{O)%VOW`){D8%W.png |
|  |

# 二、版本回退reset –-hard

Git log：查看提交历史

Git log –pretty=oneline：简化显示提交历史

Git reflog：查看命令历史

Git reset –hard HEAD^：回退到上一个版本

Git reset –hard HEAD^^：回退到上上一个版本

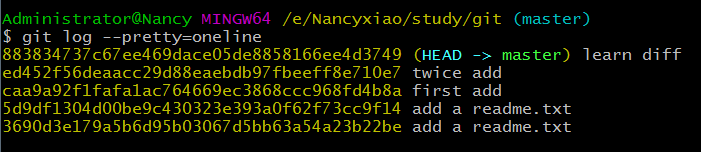
Git reset –hard HEAD~100：回退到前100个版本

Git reset –hard [commit id]：回到回退前（未来）的某个版本

git log命令显示从最近到最远的提交日志

如果嫌输出信息太多，看得眼花缭乱的，可以试试加上--pretty=oneline参数

$ git log --pretty=oneline



在Git中，用HEAD表示当前版本,上一个版本就是HEAD^，上上一个版本就是HEAD^^，当然往上100个版本写100个^比较容易数不过来，所以写成HEAD~100。

HEAD指向的版本就是当前版本，因此，Git允许我们在版本的历史之间穿梭，使用命令git reset --hard commit\_id。

穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。

要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

回退到上一版本：$ git reset --hard HEAD^

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git reset --hard HEAD^

HEAD is now at 92c4890 commit doc

查看命令历史： $ git reflog （回退前命令和回退命令都能看到）

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git reflog

92c4890 (HEAD -> master) HEAD@{0}: reset: moving to HEAD^

09eff3a HEAD@{1}: commit: reset

92c4890 (HEAD -> master) HEAD@{2}: commit: commit doc

8838347 HEAD@{3}: commit: learn diff

ed452f5 HEAD@{4}: reset: moving to ed452f56

caa9a92 HEAD@{5}: reset: moving to HEAD^

ed452f5 HEAD@{6}: commit: twice add

caa9a92 HEAD@{7}: commit: first add

5d9df13 HEAD@{8}: commit: add a readme.txt

3690d3e HEAD@{9}: commit (initial): add a readme.txt

查看提交历史： $ git log --pretty=oneline （回退前命令和回退命令看不到）

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git log --pretty=oneline

92c4890d4e4c9672da4aeae2200e69c8d346c45f (HEAD -> master) commit doc

883834737c67ee469dace05de8858166ee4d3749 learn diff

ed452f56deaacc29d88eaebdb97fbeeff8e710e7 twice add

caa9a92f1fafa1ac764669ec3868ccc968fd4b8a first add

5d9df1304d00be9c430323e393a0f62f73cc9f14 add a readme.txt

3690d3e179a5b6d95b03067d5bb63a54a23b22be add a readme.txt

回到未来（回到退回前的某个commit）：$ git reset --hard 09eff3a

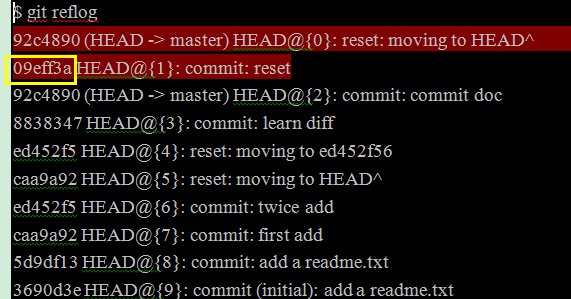
Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git reset --hard 09eff3a

Unlink of file 'git/LearnGit.docx' failed. Should I try again? (y/n) y

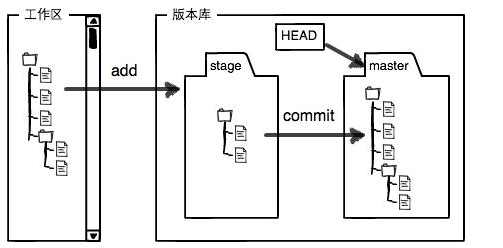
HEAD is now at 09eff3a reset

其中09eff3a为用git reflog查到的提交id



# 三、工作区、版本库

## 3.1、工作区版本库



用git diff HEAD -- readme.txt命令可以查看指定文件工作区和版本库里面最新版本的区别

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git diff HEAD -- git.txt

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

index 09e440c..767eb3e 100644

--- a/git/git.txt

+++ b/git/git.txt

@@ -5,3 +5,5 @@ git diff

learn git reset --hard HEAD^

learn git reset --hard id

+

+workpace

## 3.2、撤回修改

命令git checkout -- readme.txt意思就是，把readme.txt文件在工作区的修改全部撤销，这里有两种情况：

一种是readme.txt自修改后还没有被放到暂存区，现在，撤销修改就回到和版本库一模一样的状态；

一种是readme.txt已经添加到暂存区后，又作了修改，现在，撤销修改就回到添加到暂存区后的状态。

总之，就是让这个文件回到最近一次git commit或git add时的状态。

用命令git reset HEAD <file>可以把暂存区的修改撤销掉（unstage），重新放回工作区

git reset命令既可以回退版本，也可以把暂存区的修改回退到工作区。当我们用HEAD时，表示最新的版本

### 小结

场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout –- file。

场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD <file>，就回到了场景1，第二步按场景1操作。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考[版本回退](https://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/0013744142037508cf42e51debf49668810645e02887691000" \t "_blank)一节，不过前提是没有推送到远程库 git reset –hard HEAD^

修改：增加了workpace

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git diff HEAD -- git.txt

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

index 09e440c..767eb3e 100644

--- a/git/git.txt

+++ b/git/git.txt

@@ -5,3 +5,5 @@ git diff

learn git reset --hard HEAD^

learn git reset --hard id

+

+workpace

添加到暂存区

|  |
| --- |
| Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)  $ git add git.txt  撤回暂存区的修改 |

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git reset HEAD git.txt

Unstaged changes after reset:

M git/LearnGit.docx

M git/git.txt

丢弃工作区修改

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git checkout -- git.txt

查看文件内容，没了workpace

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ cat git.txt

Add new one

add two

git diff

learn git reset --hard HEAD^

learn git reset --hard id

## 3.3 删除文件

命令git rm用于删除一个文件。如果一个文件已经被提交到版本库，那么你永远不用担心误删，但是要小心，你只能恢复文件到最新版本，你会丢失**最近一次提交后你修改的内容**。

一般情况下，你通常直接在文件管理器中把没用的文件删了，或者用rm命令删了

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ rm test.txt

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add/rm ..." to update what will be committed)

(use "git checkout -- ..." to discard changes in working directory)

modified: LearnGit.docx

deleted: test.txt

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ ls

'~$arnGit.docx' Git-2.8.1-32-bit.exe\* 接口测试全流程.txt

'~WRL3249.tmp' github-git-cheat-sheet.pdf

git.txt LearnGit.docx

场景一：确实要从版本库中删除该文件，那就用命令git rm删掉，并且git commit

小提示：先手动删除文件，然后使用git rm <file>和git add<file>效果是一样的。

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git rm test.txt

rm 'git/test.txt'

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git commit -m "remove test.txt"

[master 4675109] remove test.txt

1 file changed, 0 insertions(+), 0 deletions(-)

delete mode 100644 git/test.txt

场景二：删错了，因为版本库里还有呢，所以可以很轻松地把误删的文件恢复到最新版本

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ git checkout -- test.txt

Administrator@Nancy MINGW64 /e/NancyXiao/study/git (master)

$ ls

'~$arnGit.docx' Git-2.8.1-32-bit.exe\* test.txt

'~WRL3249.tmp' github-git-cheat-sheet.pdf 接口测试全流程.txt

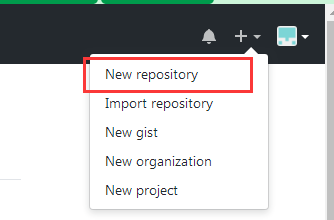
git.txt LearnGit.docx

# 四、远程仓库

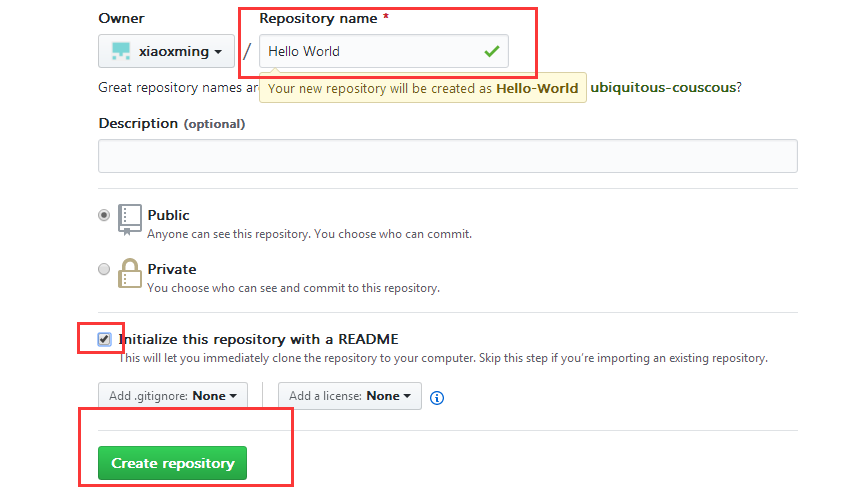
48206703+xiaoxming@users.noreply.github.com

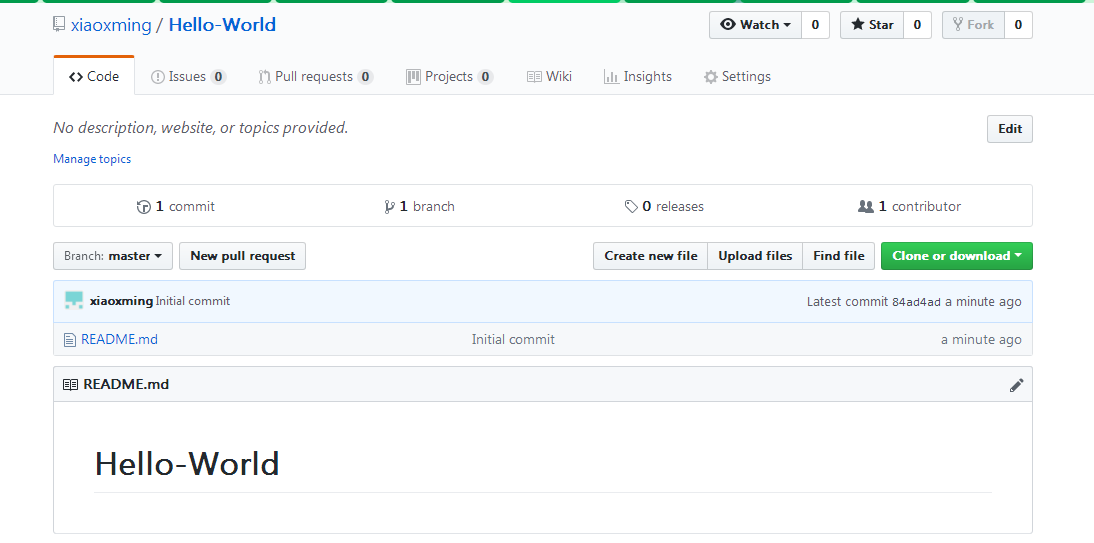
## 4.1注册GitHub

## 4.2创建仓库 New repository



### 4.2.1、创建仓库:





### 4.3 关联远程仓库

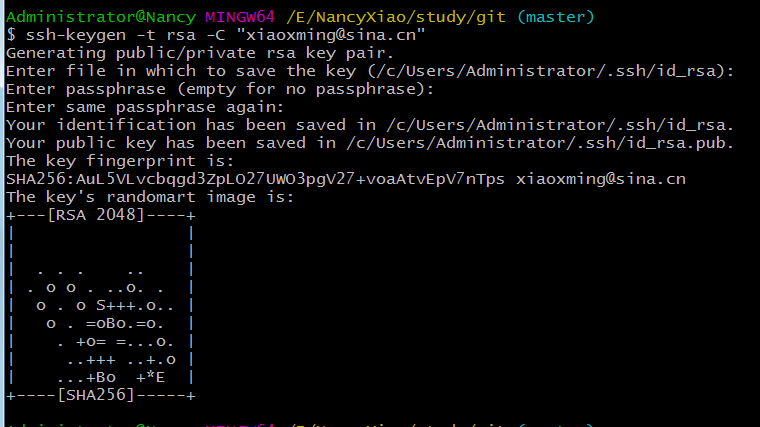
#### 4.3.1设置SSH

1）$ ssh-keygen -t rsa -C "xiaoxming@sina.cn"

2）直接回车

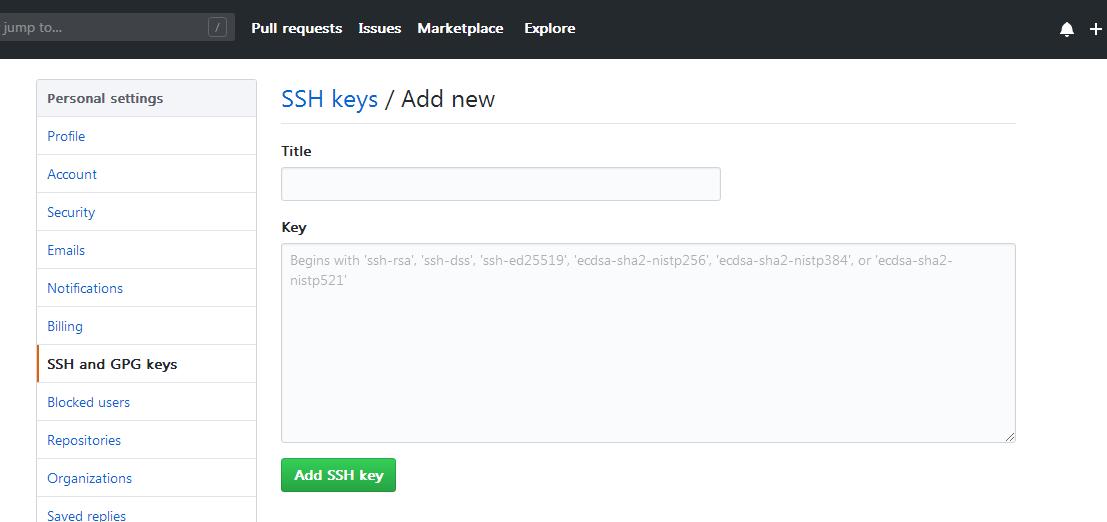
3）设置密码、确认密码

4）设置成功



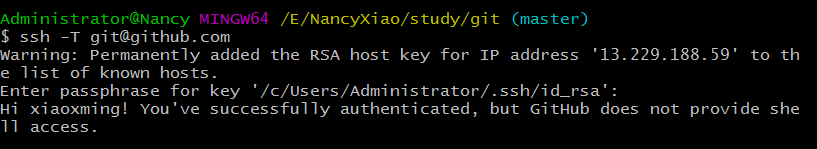
5）打开 /c/Users/Administrator/.ssh/id\_rsa.pub，全部复制

6）到github->setting->SSH and GPG keys->New SSH key, 粘贴复制的内容到key,标题随意，点击Add SSH key



验证是否成功

ssh -T git@github.com



#### 4.3.2关联-推送

在本地仓库git下执行命令

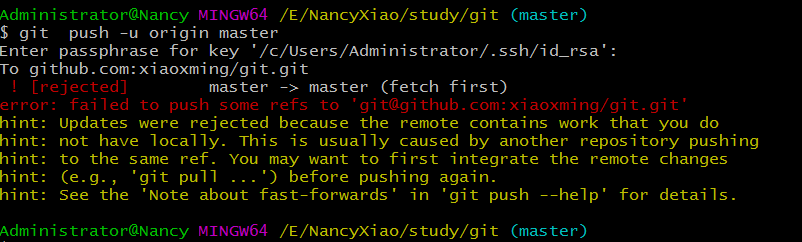
git remote add origin git@github.com:xiaoxming/git.git

git.git 是远程上的仓库

将本地所有内容推送到远程仓库

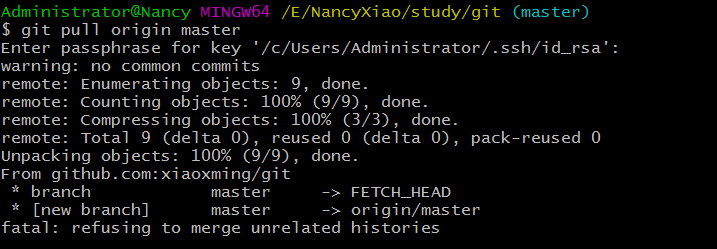
git push -u origin master

##### 推送过程报错1：

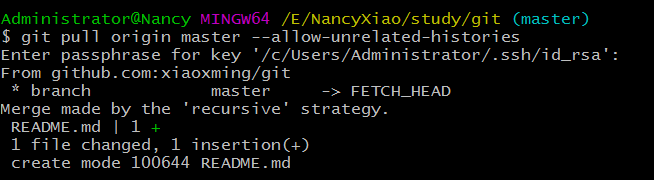


解决办法： $ git pull origin master

##### 继续报错2：

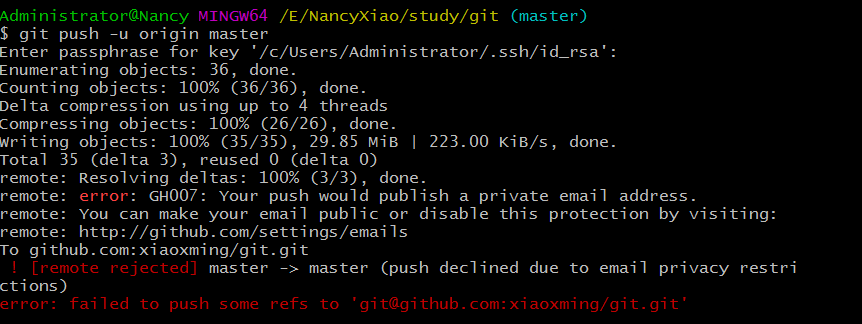


解决办法：git pull origin master --allow-unrelated-histories



##### 再次推送报错3

：$git push -u origin master



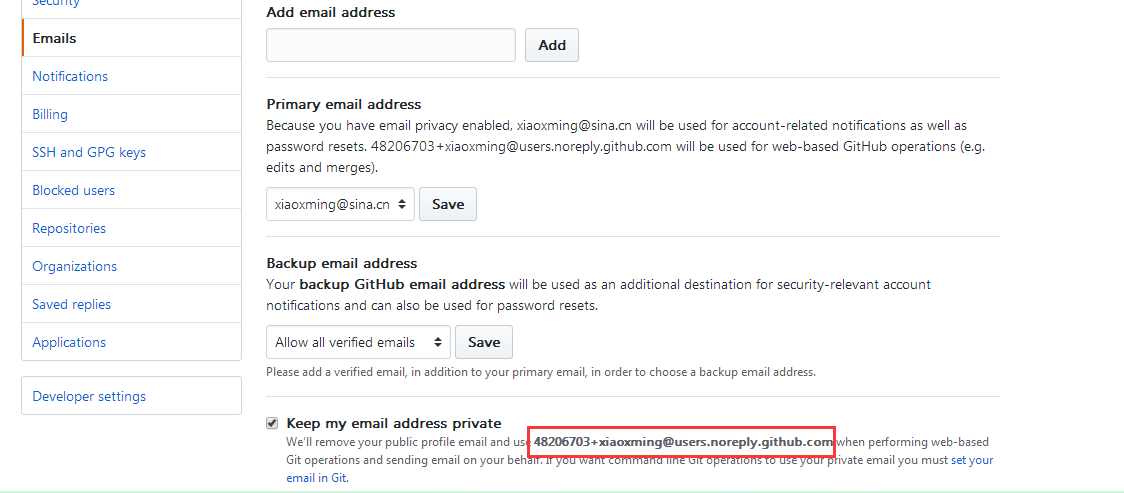
解决办法：

参考：https://blog.csdn.net/qq\_26819733/article/details/78374129

1）git config --global user.email

查看当前的用户

2）查看推荐用户



3）设置推荐用户

git config --global user.email 48206703+xiaoxming@users.noreply.github.com

4）再次推送，成功

git push -u origin master

