**Git跟着廖雪峰老师的git教程步骤学习，简要记下来防止自己忘记**。

**1：安装git（Windows下**）

**2：安装完成后，在开始菜单里找到“Git”->“Git Bash”**，蹦出一个类似命令行窗口的东西，就说明Git安装成功！

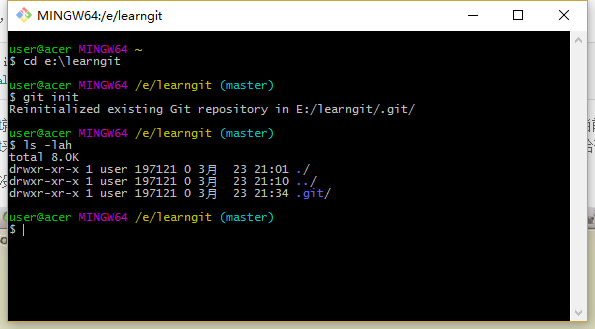
安装完成后，还需要最后一步设置，在命令行输入：

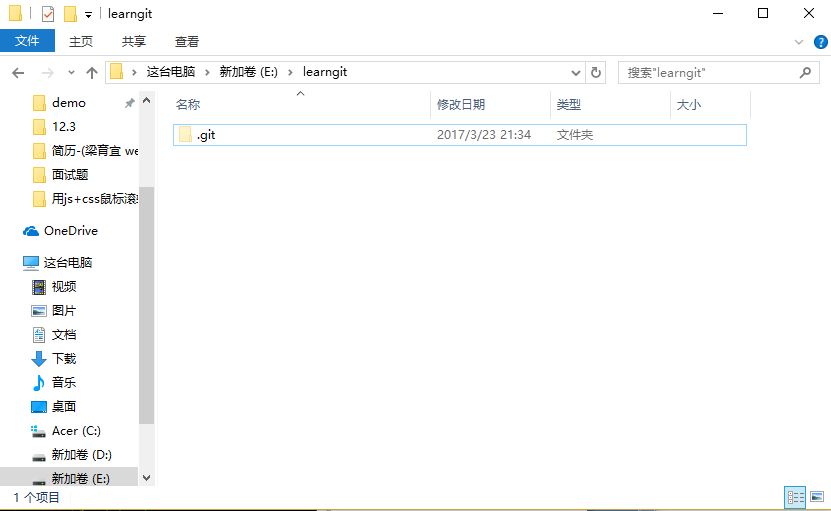
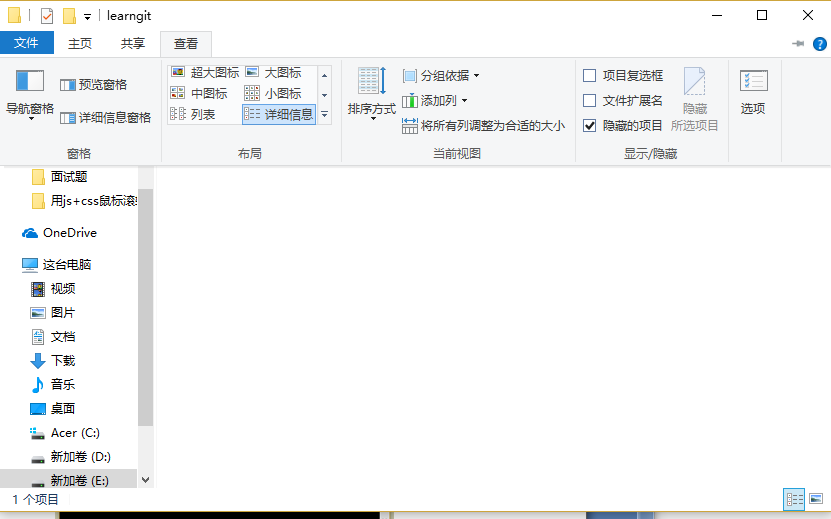
$ git config --global user.name "Your Name"

$ git config --global user.email "email@example.com"

**3：创建版本库。**

1. 在E盘的根目录下创建一个learngit的文件夹（位置随意，完全凭自己的喜好），这是一个空的文件夹，然后打开git bash 进入创建好的文件夹并进行git init命令把这个目录变成Git可以管理的仓库，如果没看见可以用键入ls –lah命令（.git文件夹默认是隐藏的），还是看不见，在窗口顶端有个查看，勾选隐藏的文件，就可以看到了。.git目录是Git来跟踪管理版本库的，没事千万不要手动修改这个目录里面的文件，不然改乱了，就把Git仓库给破坏了。





1. 也可以在git bash下创建

$ mkdir learngit

$ cd learngit

$ pwd

/Users/michael/learngit

 mkdir: 新建一个目录,就是新建一个文件夹. 如mkdir src 新建src 文件夹。pwd命令用于显示当前目录。这个仓库位于/Users/michael/learngit

1. 把文件添加到版本库

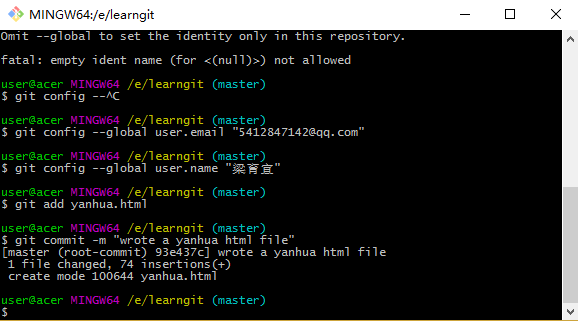
（所有的版本控制系统，只能跟踪文本文件的改动，比如TXT文件，网页，所有的程序代码等等）

在工作目录下创建一个yanhua.html文件.

用命令git add告诉Git，把文件添加到仓库：

$ git add yanhua.htm

然后，用命令git commit告诉Git，把文件提交到仓库：



-m后面输入的是本次提交的说明，可以输入任意内容，当然最好是有意义的，这样你就能从历史记录里方便地找到改动记录。

git commit命令执行成功后会告诉你，1个文件被改动（我们新添加的yanhua.html文件），插入了两行内容（yanhua.html有74行内容）

commit可以一次提交很多文件，所以你可以多次add不同的文件，即：使用命令git add <file>，可反复多次使用，添加多个文件；比如：

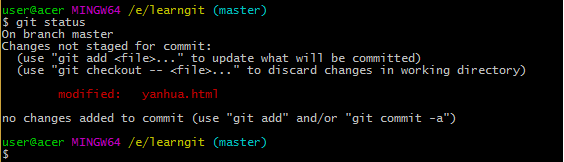
$ git add file1.txt

$ git add file2.txt file3.txt

$ git commit -m "add 3 files."

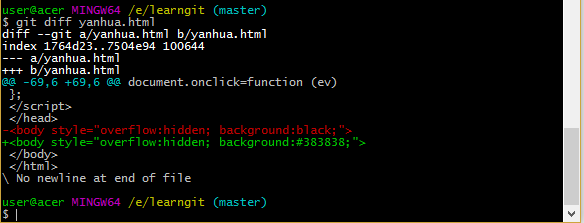
1. **修改文件**

当我修改yanhua.html 了后，可以使用git status命令查看：



git status命令可以让我们时刻掌握仓库当前的状态，上面的命令告诉我们，yanhua.html被修改过了，但还没有准备提交的修改。

用git diff这个命令可以查看具体修改了什么内容；

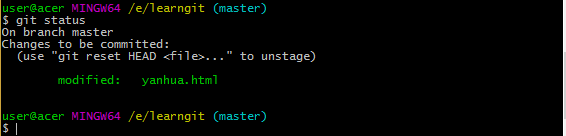


git diff顾名思义就是查看difference，显示的格式正是Unix通用的diff格式，可以从上面的命令输出看到，我把背景颜色从black改成了#383838。

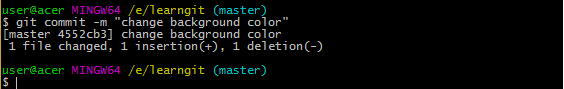
知道了对yanhua.html作了什么修改后，再把它提交到仓库就放心多了，提交修改和提交新文件是一样的两步，第一步是git add:

$ git add yanhua.html

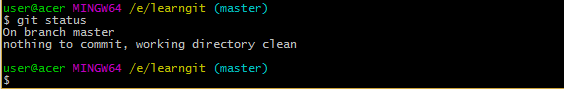
同样没有任何输出。在执行第二步git commit之前，我们再运行git status看看当前仓库的状态:



git status告诉我们，将要被提交的修改包括readme.txt，下一步，就可以放心地提交了：



提交后，我们再用git status命令看看仓库的当前状态



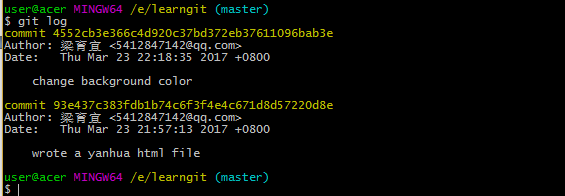
Git告诉我们当前没有需要提交的修改，而且，工作目录是干净（working directory clean）的。

(Tips:

git add -A; 添加所有改动

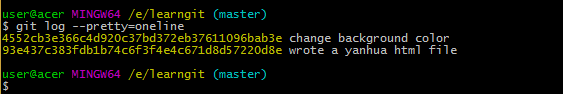
git commit -m "add all changed" 提交)

命令可以告诉我们历史记录，在Git中，我们用git log命令查看:



git log命令显示从最近到最远的提交日志，我们可以看到2次提交，最近的一次是改变背景色，最早的一次是wrote a yanhua html file.

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：

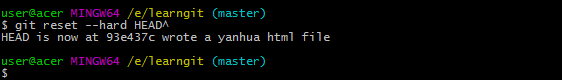


看到的一大串类似4552...1096bab3e的是commit id（版本号）

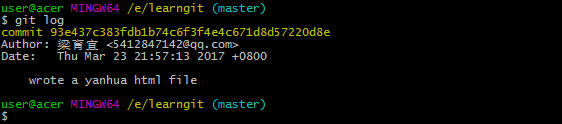
1. **回退到上一个版本**

首先，Git必须知道当前版本是哪个版本，在Git中，用HEAD表示当前版本，也就是最新的提交4452...1096bab3e（注意我的提交ID和你的肯定不一样），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。

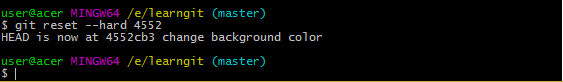
现在，我们要把当前版本“change background color”回退到上一个版本“wrote a yanhua html file”，就可以使用git reset命令：



用git log再看看现在版本库的状态：最新的那个版本已经看不到了.

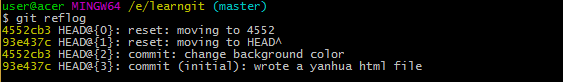


如果还想回到第二次修改的文件，只要上面的命令行窗口还没有被关掉，你就可以顺着往上找啊找啊，找到那个change background color的commit id是4552...，于是就可以指定回到未来的某个版本：



再去看yanhua.html的内容，又回来了。

Git提供了一个命令git reflog用来记录你的每一次命令：、



第三第四行前面的数字就是是我两次命令的commit id。

在退回以前版本时，最好用git log命令查看提交历史，以便确定要回退到哪个版本。用git reflog查看命令历史，以便确定要回到未来的哪个版本。

用git diff HEAD -- yanhua.html命令可以查看工作区和版本库里面最新版本的区别：

1. **撤销修改**

git checkout -- file可以丢弃工作区的修改：

$ git checkout -- yanhua.html

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

一种是yanhua.html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yanhua.html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。

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 re**set** HEAD yanhua.html

Unstaged changes after reset:

M yanhua.html

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。

再用git status查看一下，现在暂存区是干净的，工作区有修改：

$ git status

*# On branch master*

*# Changes not staged for commit:*

*# (use "git add <file>..." to update what will be committed)*

*# (use "git checkout -- <file>..." to discard changes in working directory)*

*#*

*# modified:* yanhua.html

*#*

no changes added to commit (use "git add" **and**/**or** "git commit -a")

还记得如何丢弃工作区的修改吗？

$ git checkout -- yanhua.html

$ git status

*# On branch master*

nothing to commit (working directory clean)

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](http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/0013744142037508cf42e51debf49668810645e02887691000)一节，不过前提是没有推送到远程库。

1. **删除文件**

先添加一个新文件test.txt到Git并且提交：

$ git add test.txt

$ git **commit** -m "add test.txt"

[master 94cdc44] **add** test.txt

1 file changed, 1 insertion(+)

**create** mode 100644 test.txt

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：

$ rm test.txt

这个时候，Git知道你删除了文件，因此，工作区和版本库就不一致了，git status命令会立刻告诉你哪些文件被删除了：

$ git status

*# On branch master*

*# Changes not staged for commit:*

*# (use "git add/rm <file>..." to update what will be committed)*

*# (use "git checkout -- <file>..." to discard changes in working directory)*

*#*

*# deleted: test.txt*

*#*

no changes added to commit (use "git add" **and**/**or** "git commit -a")

现在你有两个选择，一是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：

$ git rm test.txt

rm 'test.txt'

$ git **commit** -m "remove test.txt"

[master d17efd8] remove test.txt

1 file changed, 1 deletion(-)

**delete** mode 100644 test.txt

现在，文件就从版本库中被删除了。

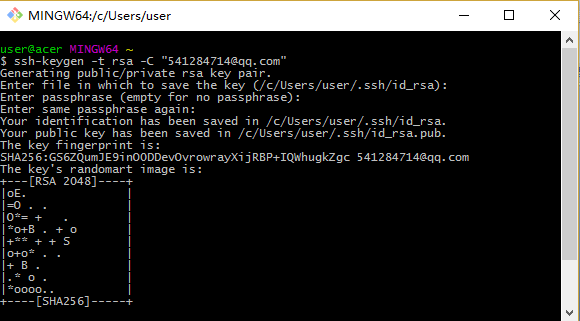
另一种情况是删错了，因为版本库里还有呢，所以可以很轻松地把误删的文件恢复到最新版本：

$ git checkout -- test.txt

git checkout其实是用版本库里的版本替换工作区的版本，无论工作区是修改还是删除，都可以“一键还原”。

1. **远程仓库**
2. 创建SSH Key。在用户主目录下，看看有没有.ssh目录，如果有，再看看这个目录下有没有id\_rsa和id\_rsa.pub这两个文件，如果已经有了，可直接跳到下一步。如果没有，打开Shell（Windows下打开Git Bash），创建SSH Key：

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



id\_rsa是私钥，不能泄露出去，id\_rsa.pub是公钥，可以放心地告诉任何人。

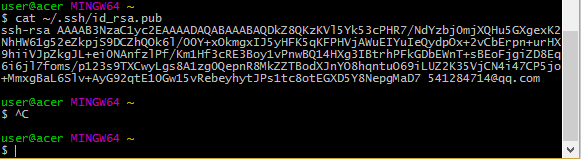
（2）：登陆GitHub，打开“Account settings”，“SSH Keys”页面：

然后，点“Add SSH Key”，填上任意Title，在Key文本框里粘贴id\_rsa.pub文件的内容：

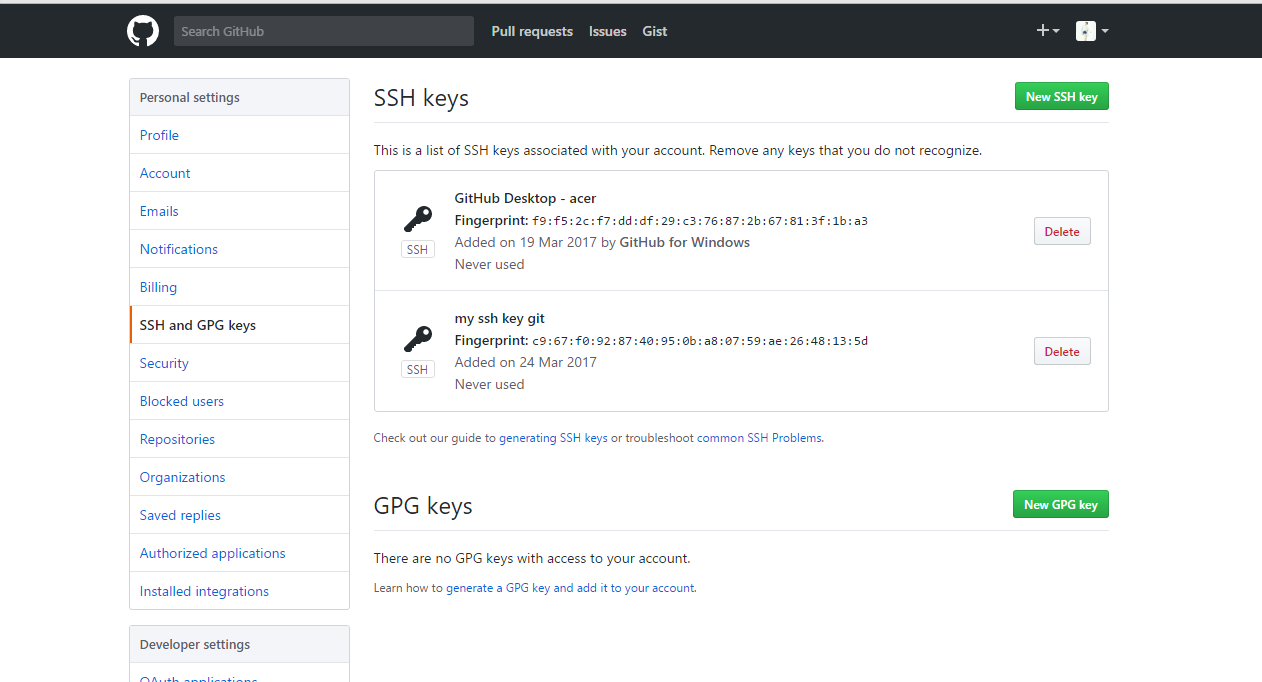
我使用clip命令没有将公钥复制到剪切板，不知道是没问题。

clip < ~/.ssh/id\_rsa.pub

然后我使用cat ~/.ssh/id\_rsa.pub 命令，成功了，公钥以ssh- 开头，邮箱地址结尾。



GitHub上ssh key添加成功



1. **创建GitHub仓库。（创建了一个**learngit的仓库**）**

根据GitHub的提示，在本地的learngit仓库下运行命令：

$ git remote add origin https://github.com/lyuyi/learngit.git

****

把本地库的所有内容推送到远程库上：

使用命令git push -u origin master第一次推送master分支的所有内容；

此后，每次本地提交后，就可以使用命令git push origin master推送最新修改；

（可能第一次push弹出ssh验证，输入GitHub用户名及密码即可）

