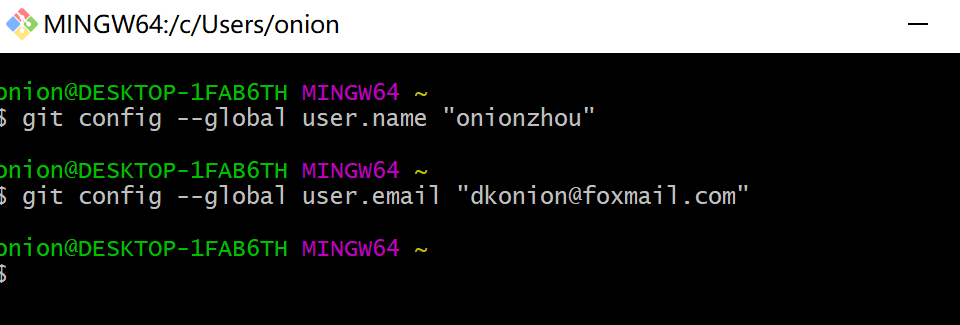
# Git 初始化

第一步配置你的名字和邮箱

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

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



# Git 提交代码

# git add “xxxx.file”

# git commit -m “xxxxx”

# git push

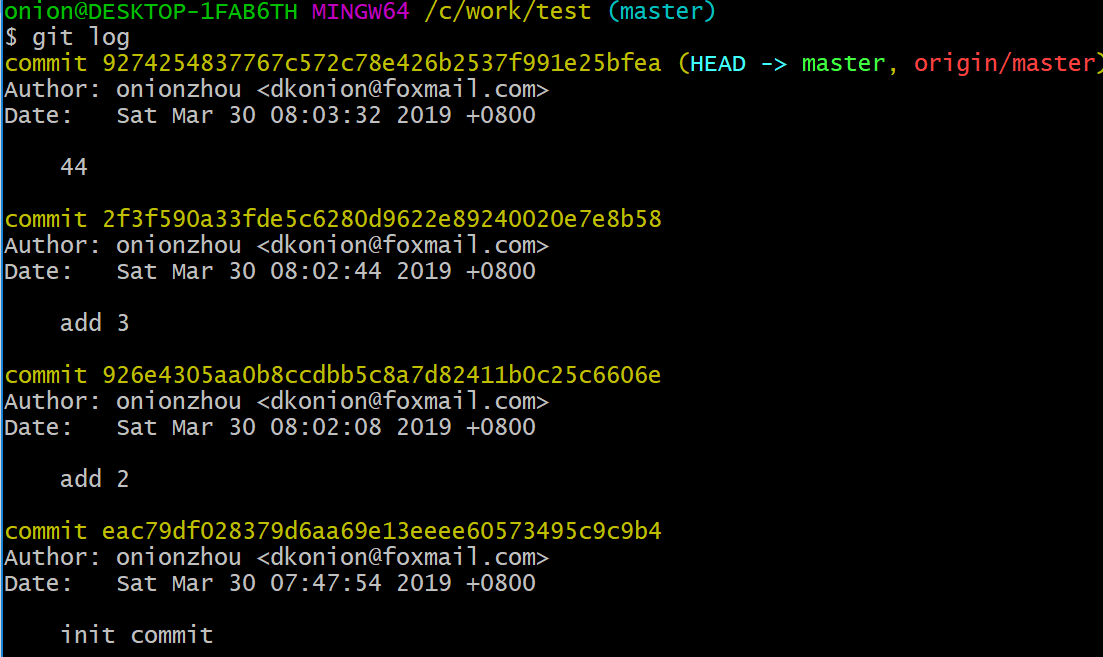
# Git 删除add 文件

**git rm --cached "文件路径"**，不删除物理文件，仅将该文件从缓存中删除；

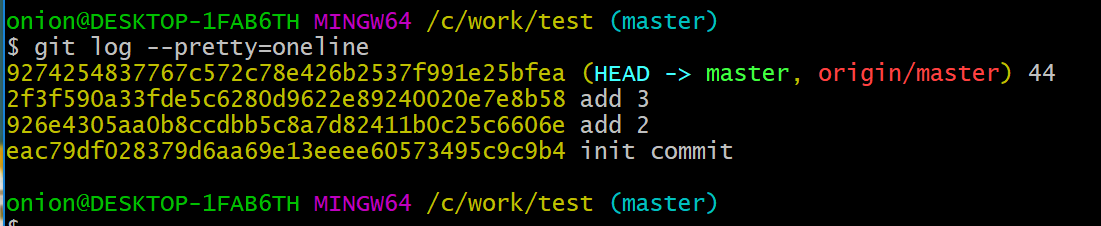
**git rm --f "文件路径"**，不仅将该文件从缓存中删除，还会将物理文件删除（不会回收到垃圾桶）

# Git 版本回退

# git log

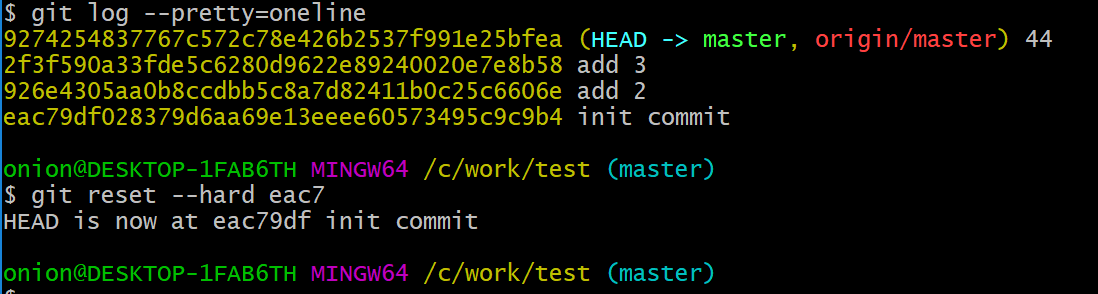


# git log –pretty=oneline



Head 指向的是当前版本

# git reset –-hard commit id



#git log 查看提交历史

# git reflog 查看命令历史

# GIt管理修改

# git checkout -- <file> 丢弃工作区的内容， 如果暂存区有就恢复暂存区的，如果没有就是版本库的

# git reset HEAD -- <file> 丢弃暂存区的内容

# Git 分支管理

## 创建合并分支

# git checkout -b dev # 创建一个dev 分支，在切换到dev分支

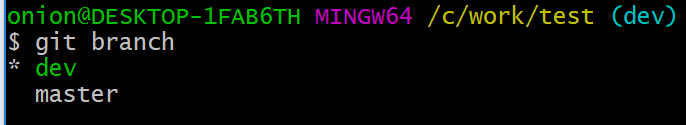
上面那条指令相当于下面两条指令

#git branch dev 创建分支

# git checkout dev 切换到dev分支

查看当前分支

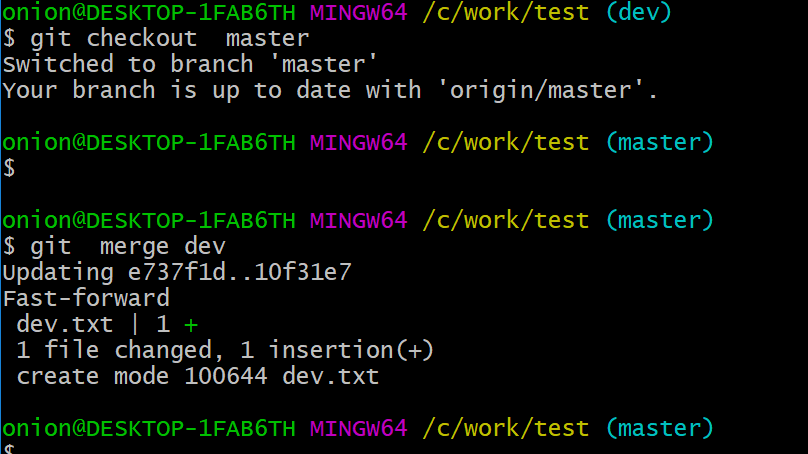
#git branch



合并分支 到主分支

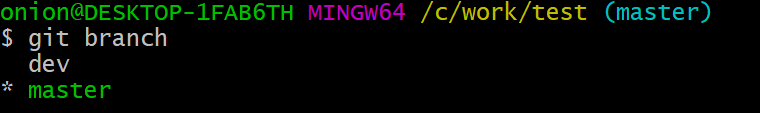
#git checkout master

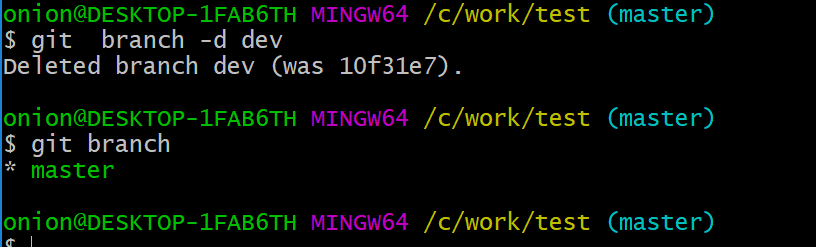
# git merge dev



删除分支

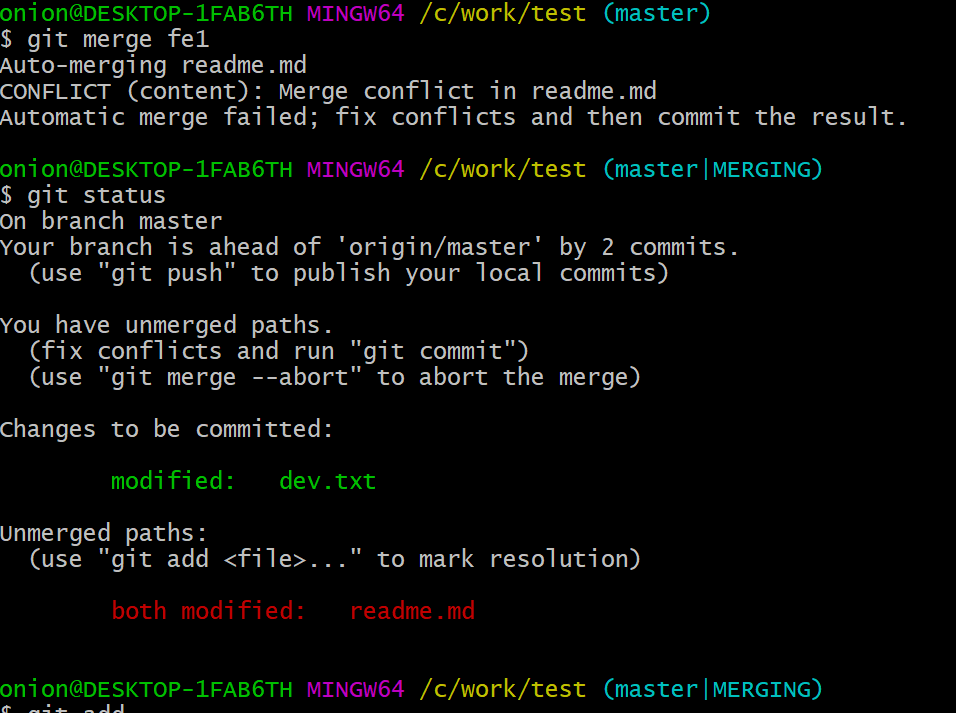
#git branch -d dev





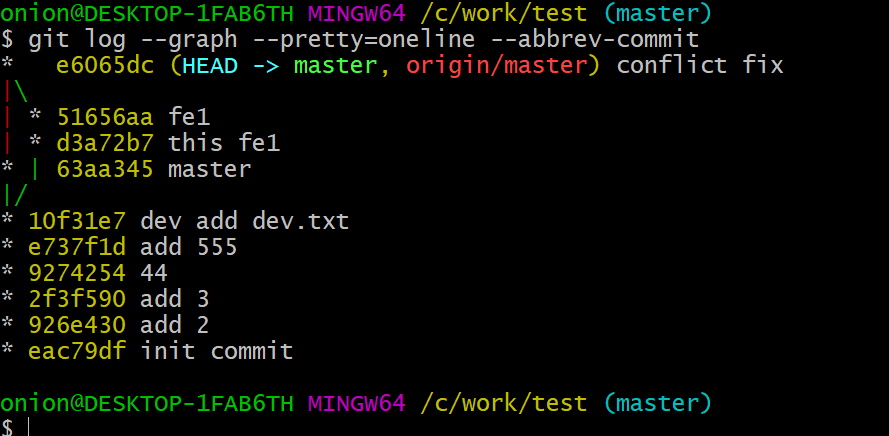
## 解决冲突

当 merge 分支fe1 到master 时，因为都修改了同一个文件，就需要自己合并一下代码



Git用<<<<<<<，=======，>>>>>>>标记出不同分支的内容，我们修改如下后保存

#$ git log --graph --pretty=oneline --abbrev-commit 查看 分支情况



#--no -ff #表示禁用fast forward ,禁用能看出历史有做过合并， 不禁用是看不出以前做个合并的

# git merge --no-ff -m "merge with no-ff" dev

## Bug 管理 (git stash)

场景，正在开发一个新feature 开发了一半，需要修复一个紧急bug，开发一半的功能无法提交，怎么解决

# git stash 把当前的工作存储起来

拉bug分支，修复，提交，合并到主分支，删除bug分支（）

完成后恢复当前工作区

#git stash list

#git stash pop ,恢复并删除stash 里存储的内容

#1.git stash apply ; 恢复stash 里的内容到工作区，不删除stash里的内容

#2.git stash drop 删除stash 里存储的内容

Ps 多次stash 后，可以通过 git stash apply stash@{0} 恢复指定的stash

## 多人协作

1.尝试推送代码到远端

# git push origin <branch\_name>

2.推送失败，说明你的远端代码比你本地代码新，尝试拉取最新的代码

#git pull

如果合并有冲突，就解决冲突，并在本地提交

3.解决冲突后，重新git push origin <branch\_name>

注：git pull 提示 no tracking information ，说明本地分支和远程分支链接没有创建，

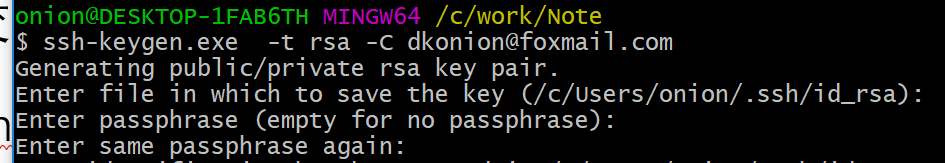
#git branch --set-upstream-to <branch-name> origin/<branch-name>

4.在本地创建和远程分支对应关系是

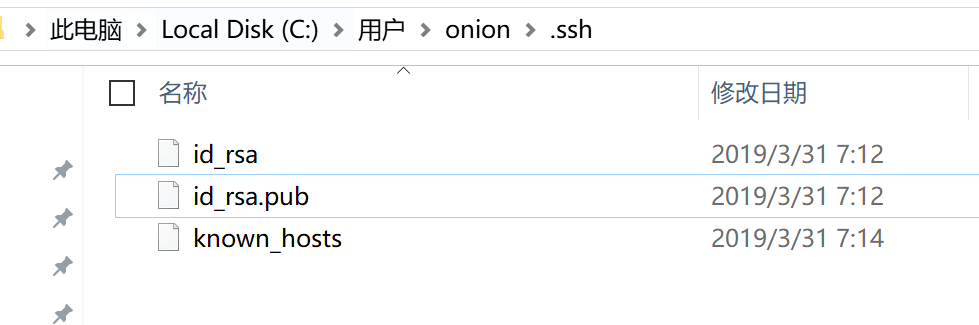
#git checkout -b branch-name origin/branch-name 分支名最好相同

# GitHub 添加公钥

* ssh-keygen.exe -t rsa -C dkonion@foxmail.com



* 在/c/users/onion/.ssh/id\_rsa/ 目录里把id\_rsa.pub里的密钥复制到github上



* Github 账户 ---settings---ssh and GPS key ---new ssh key 添加公钥
* # ssh -T [git@github.com](mailto:git@github.com) 可以测试连接情况

