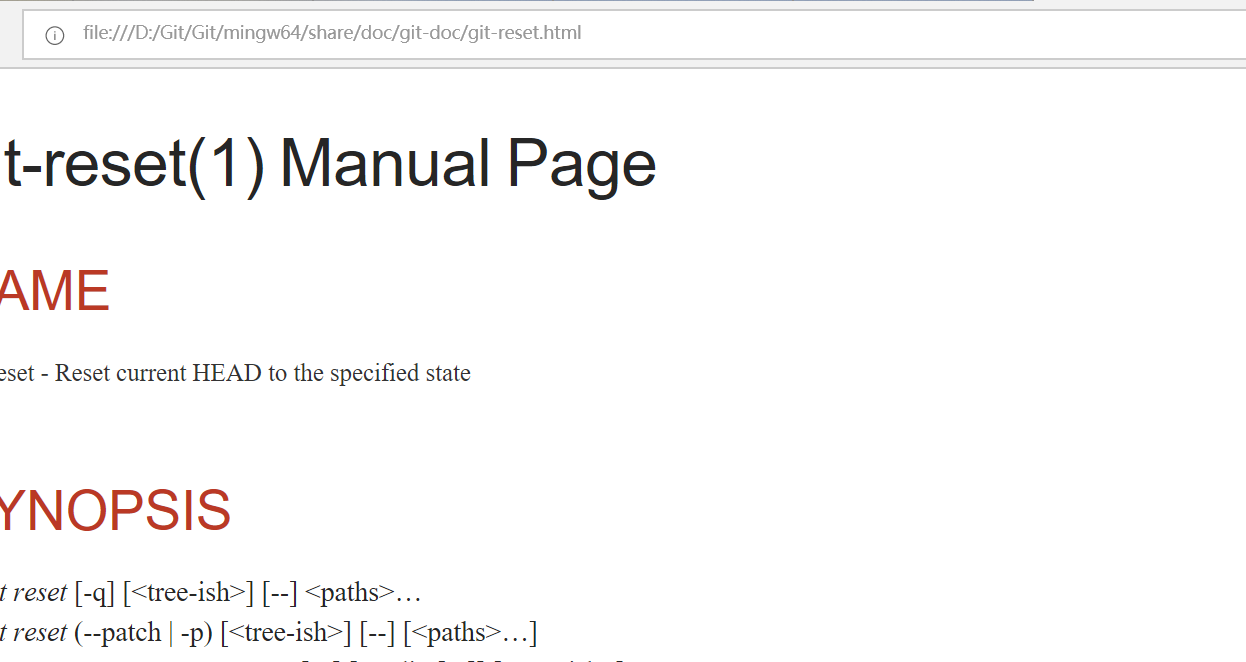
0.查看本地文档

Git help [指令]





输入指令会弹出一个html界面

1. git init

这个指令会产生一个.git文件，.git文件里面是各种配置信息，所以说不要瞎动

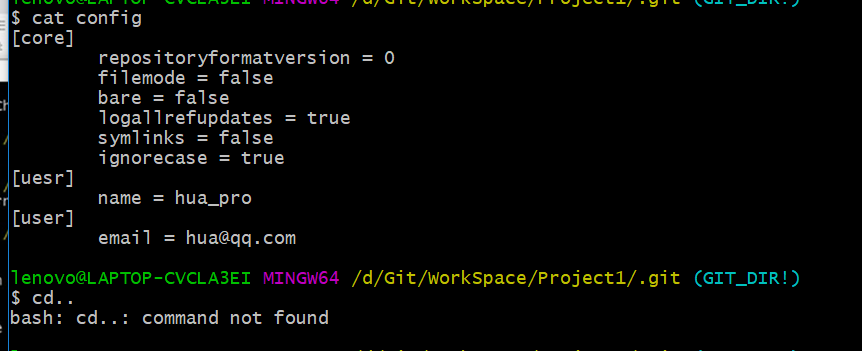
2.设置签名

设置成本地可用

git config user.name hua\_pro

git config user.email [Hua@qq.com](mailto:Hua@qq.com)

设置到.git目录下面的config文件里面

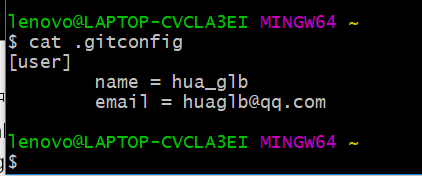


设置成本系统用户可用

git config --global user.name hua\_pro

git config --global user.email [Hua@qq.com](mailto:Hua@qq.com)

设置到c盘的.gitconfig目录下面



此处是为了演示，设置了两个，通常开发的时候只设置系统的就好

3.基本的操作

a.状态查看

Git status

查看工作区，暂存区状态

b.添加操作

Git add [file name]

将工作区的新建/修改添加到暂存区

c.提交操作

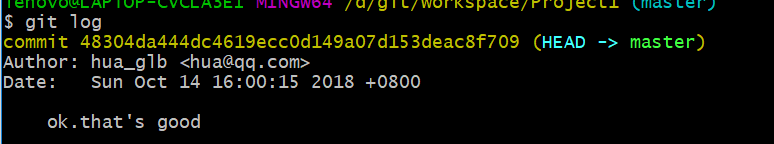
Git commit [file name]

Git commit -m “message” [file name]

将暂存区的内容提交到本地库

4.查看历史记录

a. git log 最完整的显示日志



多屏显示控制方式

空格向下翻页

B向上翻页

b. git log --pretty=oneline



这个比上一个显示的简单，少了些日期用户什么的

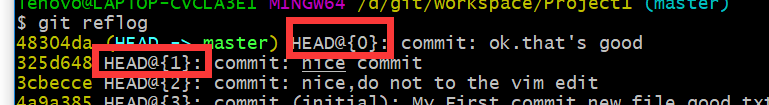
c. git log --oneline



这个相对于上一个指令是hash值变短了

只显示当前版本后面的版本

d. it log --reflog



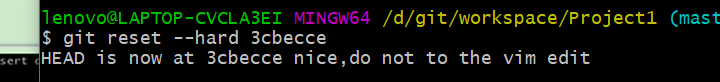
红框圈起来的部分是指移动到这个版本时需要几步

全部都显示

5.版本的前进后退

a.基于索引值操作（推荐）

git reset –hard [索引值],



这个就不用管前进还是后退了，选择对应的版本号然后过去就可以了

注意后面那个—hard还可以是—soft还可以是—mixed

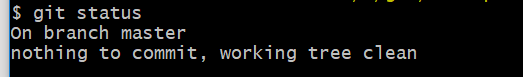
区别是:git有三个区域，本地库，暂存区，工作区

Soft:只会把本地库的版本进行移动

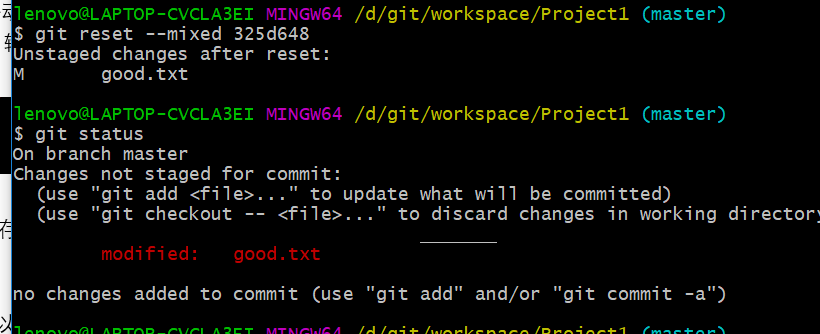
Mixed：会把本地库和暂存区的版本进行移动

Hard：会把本地库，暂存区，工作区都进行移动

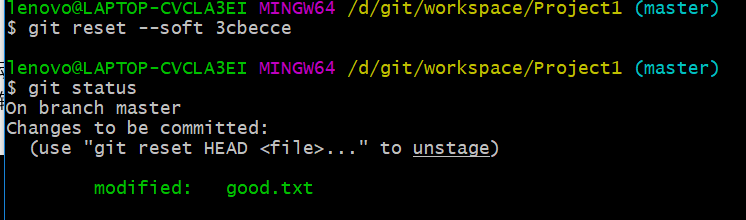
所以说，使用hard操作完之后，都进行移动，输入git status会发现没有需要添加到缓存区和需要提交到本地库的内容



在使用mixed操作完之后，移动了本地库和暂存区，所以说显得工作区做了改变，如下所示



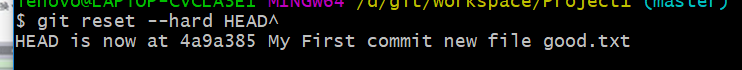
在使用soft操作完之后，只移动了本地库，所以说，本地库移动就显得暂存区和工作区都发生了改变，所以说如下所示



注意：以上内容纯属装b，显得懂的多，所以说平常基本使用hard

b.使用^符号

这个符号只能往后退



一个^符号就后退一步

c.使用~符号

只能后退

Git reset –hard HEAD~n

表示后退n步

6.删除文件并找回

前提：删除前，文件存在时的状态提交到了本地库

操作：git reset –hard 【指针位置】

指针位子：删除操作已经提交到本地库：指针位置指向历史记录

删除操作尚未提交到本地库，指针位置使用HEAD（当前状态）

7.比较文件

a. Git diff[文件名]

将工作区中的文件和暂存区进行比较

b. Git diff [本地库中历史版本] [文件名]

将工作区中的文件和本地库历史记录进行比较

c. 不带文件名比较多个文件

8.分支操作

a.创建分支

git branch [分支名]

b.查看分支

git branch -v

c.切换分支

git checkout [分支名]

d.合并分支

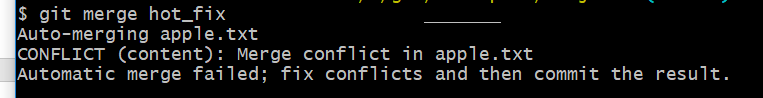
例如；想要将hot\_fix上面修改完的东西也弄到master分支上

-先将分支切换到master git checkout master git checkout【被合并的分支】

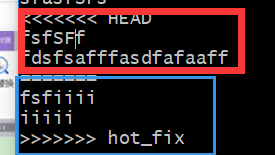
-再将hot\_fix上面的东西弄过来 git merge hot\_fix git checkout【要合并过来的新内容的分支】

e.解决冲突

因为上面分支的合并，如果两个分支都修改了同一处文件，在合并的时候会出现冲突



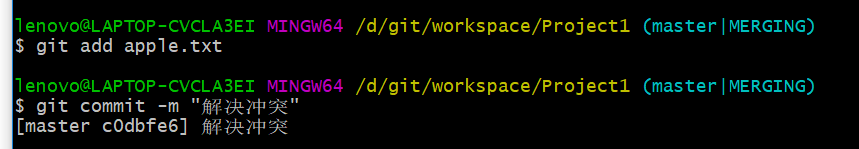
然后解决这个冲突，首先打开冲突的文件瞧一瞧



其中红框是当前分支做的修改，蓝框是hot\_fix分支做的修改

然后，将多余的信息删除，也就是<<<<<<这个东西和======这些东西删除

然后和另一个分支的人商议修改，然后修改完毕之后，添加到暂存区，然后提交到本地库

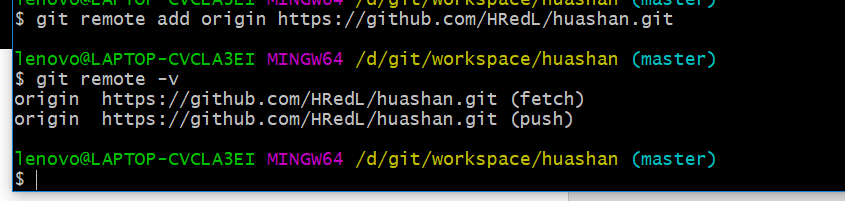


注意：解决冲突时候的提交是不带文件名

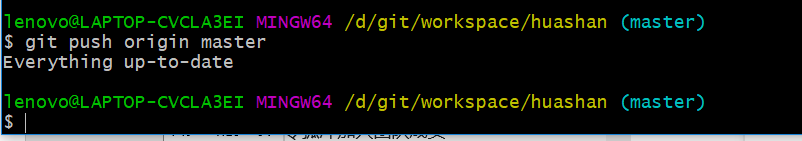
9.将某个地址起一个别名，方便以后调用

指令：git remote add origin [地址]添加

指令：git remote -v 查看



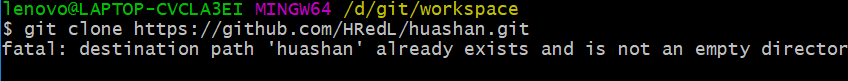
10.将本地仓库里面的东西上传到gitHub上面



11，将远程库clone到本地

直接执行这个就行，不用初始化本地化再clone

Git clone [url]



效果：完整的把远程库下载到本地

创建origin远程地址别名

初始化本地库