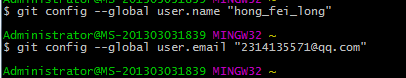
**Window7 下MsysGit工具使用**

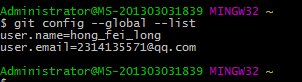
**一、配置用户名和密码:**

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

git config --global user.email  "Your Email used to register"

检查配置是否成功：git config --global --list  查看设置的用户名和Email

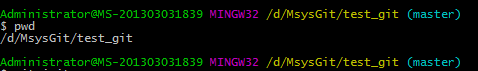




使用命令git help <command>查看command的用法

IMG_256

使用命令pwd //查看当前目录



使用命令 touch license.txt 增加一个文件license.txt

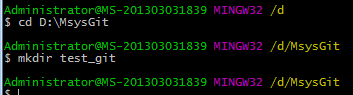
IMG_256

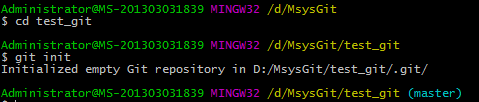
**二、创建版本库：**

mkdir test\_git        //新建目录test\_git

      cd test\_git            //进入目录test\_git

      git init                  //初始化git，现在git会管理这个目录中的所有内容了





**三、添加文件到版本库中**

touch readme.txt   //创建文件readme.txt，在windows下直接到目录新建文件即可

readme.txt内容：

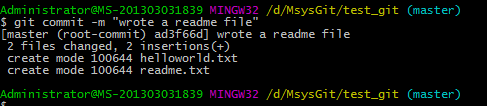
Git is a version control system

Git is free software

git add readme.txt //告诉git，把文件添加到版本库中



git commit -m "wrote a readme file" //提交，及说明注释



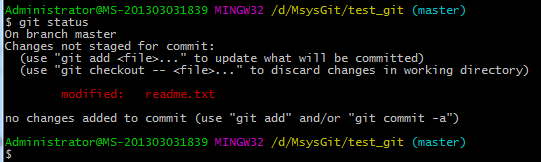
**四、穿梭时光机**

修改readme.txt文件内容

Git is a distributed version control system.

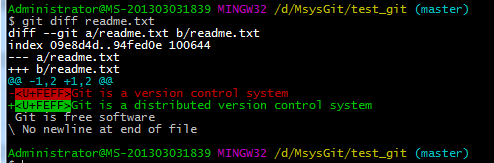
Git is free software.

Git status //查看版本库当前状态



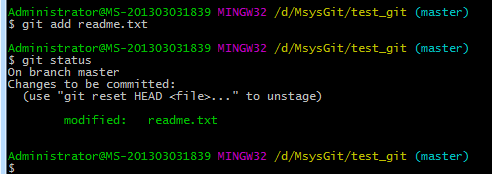
上面的提示告诉我们，readme.txt被修改过了，但还没有准备提交的修改。

Git diff readme.txt //查看修改前、后的具体内容



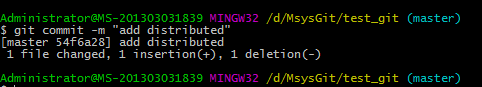
执行git add readme.txt 后

查看状态 git status



上面的提示告诉我们，将要被提交的修改有readme.txt文件，下一步可放心提交

提交git commit -m “add distributed”



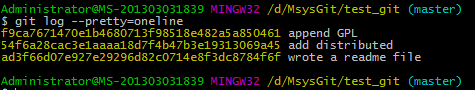
**五、版本退回**

git log //查看修改记录

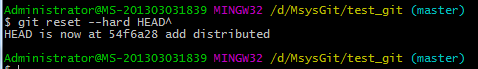


该信息显示：版本数（最近到最远的提交日志），修改者，修改时间

git log --pretty=oneline //版本单行显示

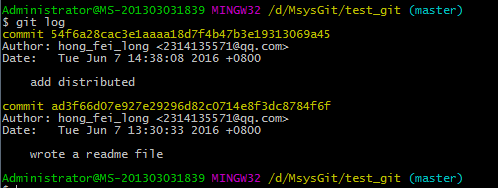


git reset --hard HEAD^ //回到上次版本



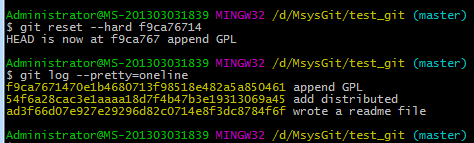
上一个版本就是HEAD^，上上一个版本就是HEAD^^，当然往上100个版本写100个^比较容易数不过来，所以写成HEAD~100

上面操作之后，在查看日志，git log



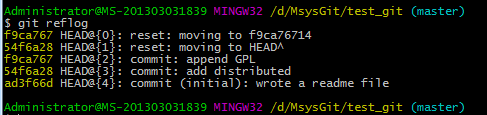
至此最新版本消失

（1）若要找回最新版本，可找到那个append GPL的commit id，然后git reset --hard id即可



（2）找不到commit id 时，使用如下命令即可找到commit id：

git reflog //记录的每次操作命令



从该显示可知道，版本append GPL 的id是f9ca767

（1）工作区和暂存区

@版本库：在工作区有一个隐藏目录 .git

@工作区：在电脑里能看到的目录，比如我的 test\_git 文件夹就是一个工作区

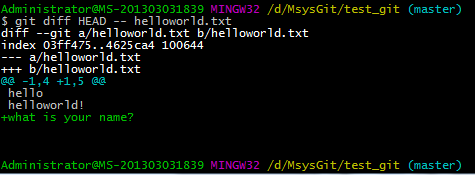
@暂存区：在.git 目录下有stage(或 index) 就是暂存区

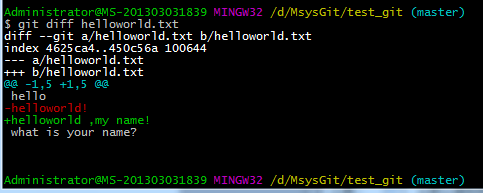
@git为我们自动创建的第一个分支master，以及指向master的一个指针叫HEAD

|  |  |
| --- | --- |
| 指令操作 | 解释 |
| Git add readme.txt | 把文件readme.txt修改添加到暂存区 |
| Git commit -m “message” | 把暂存区的所有内容提交到当前分支master |
| Git rm readme.txt | 把要删除的文件readme.txt添加到暂存区 |

git diff HEAD -- readme.txt命令可以查看工作区和版本库的区别

git diff readme.txt //查看工作区和存储区的区别



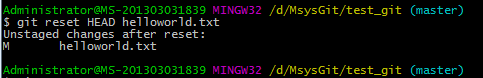


git checkout -- helloworld.txt //撤销工作区文件

IMG_256

说明：把在工作区的helloworld.txt文件修改回到最近一次git commit或git add时的状态。

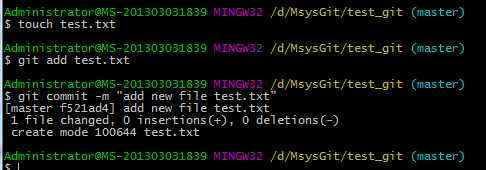
git reset HEAD helloworld.txt //撤销暂存区的helloworld.txt文件



说明：可以把暂存区的修改撤销掉（unstage），且重新放回工作区，即更新工作区文件

（2）删除文件

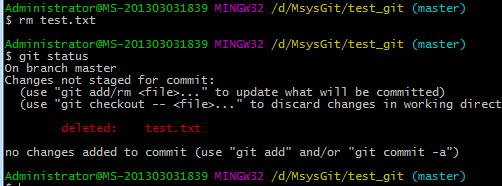
新建添加文件test.txt



rm test.txt //删除工作区文件

git rm test.txt //选中要删除的文件

删了工作区的test.txt文件后的状态

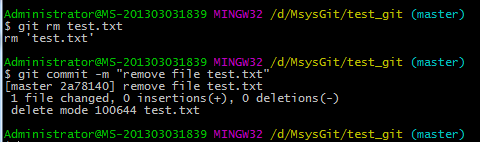


两种选择：

1. 删除错了，想恢复工作区文件



2.确实要删除，还要删除版本库里的记录（注意要两部，和添加一样）



**六、远程管理**

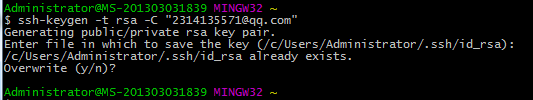
第1步：创建SSH Key

在用户主目录（C:\Users\Administrator\.ssh）下，看看有没有.ssh目录

1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，如果有则下一步

2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"](mailto:\"youremail@example.com\")

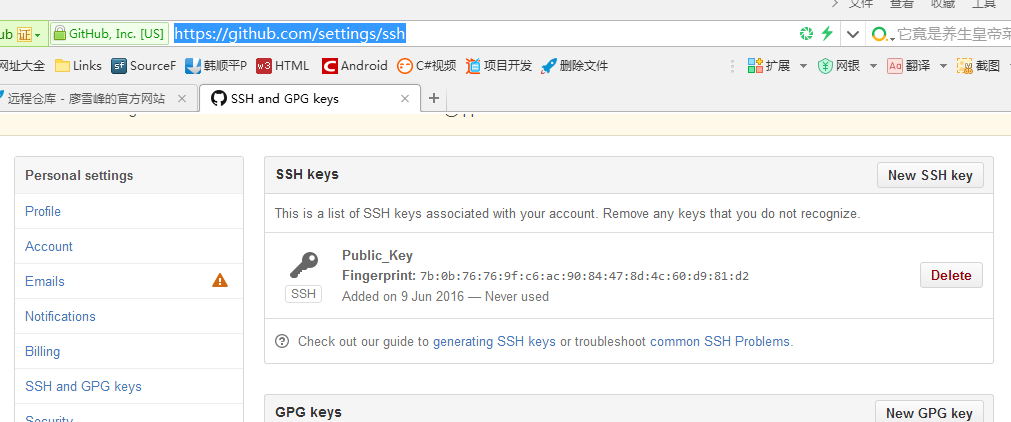


你需要把邮件地址换成你自己的邮件地址，然后一路回车，使用默认值即可，由于这个Key也不是用于军事目的，所以也无需设置密码。

如果一切顺利的话，可以在用户主目录里找到.ssh目录，里面有id\_rsa和id\_rsa.pub两个文件，这两个就是SSH Key的秘钥对，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。

第2步：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：



第3步：添加添加远程库

（1）第一步：登陆GitHub，然后，在右上角找到“Create a new repo”按钮，创建一个新的仓库

（2）第二步：在本地的 test\_git仓库下运行命令

$ git remote add origin [git@github.com:michaelliao/learngit.git](mailto:git@github.com:michaelliao/learngit.git)

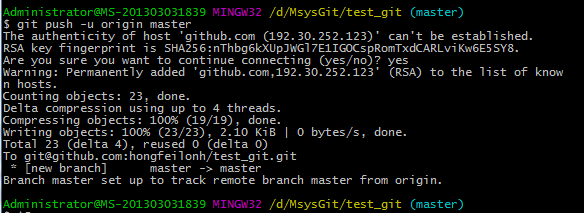


(注意：把上面的michaelliao替换成你自己的GitHub账户名，否则，你在本地关联的就是我的远程库，关联 没有问题，但是你以后推送是推不上去的，因为你的SSH Key公钥不在我的账户列表中。

添加后，远程库的名字就是origin，这是Git默认的叫法，也可以改成别的，但是origin这个名字一看就知道是远程库。)

（3）第三步，把本地库的所有内容推送到远程库上：

$ git push -u origin master



把本地库的内容推送到远程，用git push命令，实际上是把当前分支master推送到远程。

（注意：由于远程库是空的，我们第一次推送master分支时，加上了-u参数，Git不但把本地的master分支内容推送的远程新的master分支，而且把本地的master分支和远程的master分支关联起来，在以后的推送或者拉取时就可以简化命令。

推送成功后，可以立刻在GitHub页面中看到远程库的内容已经和本地一模一样）

（4）第四步：从现在起，只要本地作了提交，就可以通过命令：

$ git push origin master //把本地master分支的最新修改推送至GitHub

(说明：当你第一次使用Git的clone或者push命令连接GitHub时，会得到一个警告：

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

RSA key fingerprint is xx.xx.xx.xx.xx.

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

这是因为Git使用SSH连接，而SSH连接在第一次验证GitHub服务器的Key时，需要你确认GitHub的Key的指纹信息是否真的来自GitHub的服务器，输入yes回车即可。

Git会输出一个警告，告诉你已经把GitHub的Key添加到本机的一个信任列表里了：

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

这个警告只会出现一次，后面的操作就不会有任何警告了。

如果你实在担心有人冒充GitHub服务器，输入yes前可以对照[GitHub的RSA Key的指纹信息](https://help.github.com/articles/what-are-github-s-ssh-key-fingerprints/" \t "http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/_blank)是否与SSH连接给出的一致)

第四步：从远程库克隆

目标：先创建远程库，然后，从远程库克隆

（1）首先，登陆GitHub，创建一个新的仓库，名字叫gitskills：

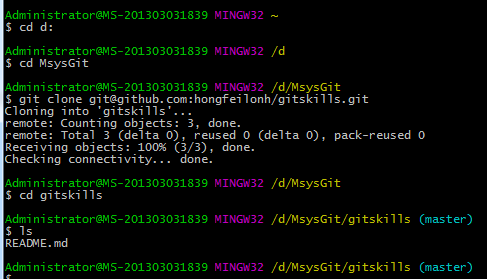
我们勾选Initialize this repository with a README，这样GitHub会自动为我们创建一个README.md文件。创建完毕后，可以看到README.md文件。

（2）现在，远程库已经准备好了，下一步是用命令git clone克隆一个本地库：

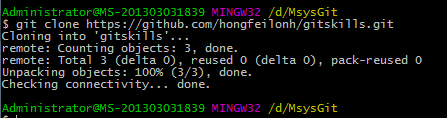
Cd 命令，先找好要放克隆版本的位置

$ git clone [git@github.com:hongfeilonh/gitskills.git](mailto:git@github.com:hongfeilonh/gitskills.git) 命令，选取目标文件进行克隆

$ ls 命令，查看文件目录，（没有.git目录）



或者



你也许还注意到，GitHub给出的地址不止一个，还可以用https://github.com/michaelliao/gitskills.git这样的地址。实际上，Git支持多种协议，默认的git://使用ssh，但也可以使用https等其他协议。

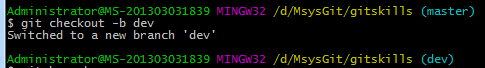
使用https协议缺点：

1.速度慢

2.每次推送都必须输入口令

**七、分支的创建和合并**

git checkout -b dev //创建dev分支，然后切换到dev分支

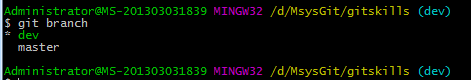


git checkout命令加上-b参数表示创建并切换，相当于以下两条命令：

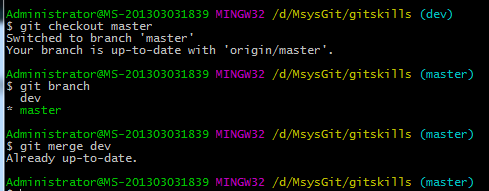
$ git branch dev //创建分支dev

$ git checkout dev //切换到分支dev

git branch //列出所有分支，当前分支前面会标一个\*号



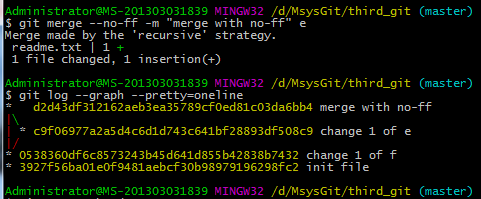
git merge dev //命令用于快速合并指定分支到当前分支



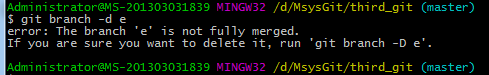
git branch -d dev //删除分支



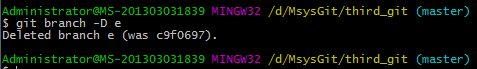
Git merge --no-ff -m “message” e //非快速合并



没有合并的分支，使用 git branch -d e 删除报错

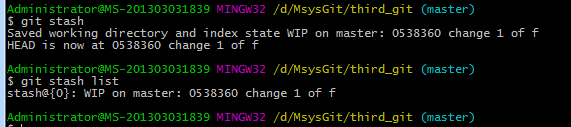


应该使用git branch -D e 命令来删除

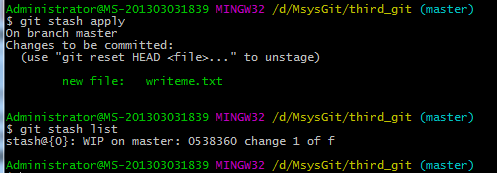


**状态保存于恢复及删除命令**

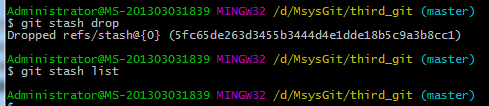
Git stash //保存当前工作状态，此刻工作区的内容为master



Git stash apply //回复保存的状态

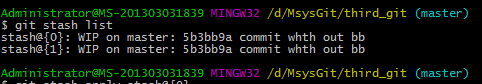


Git stash drop //删除保存的状态



git stash pop //恢复状态及删除保留的状态

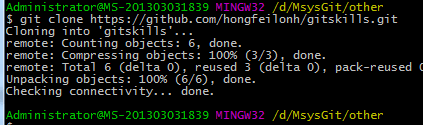
注：（保存状态的git stash 多次使用后，状态的顺序：0代表最近一次保存的）



**远程推送工作**

**克隆到本地**

http克隆

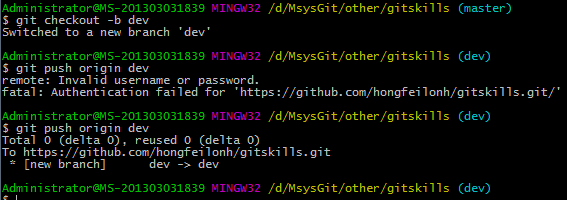


从远程仓库克隆时，本地的master分支和远程的master分支对应，并且，远程仓库的默认名称是origin。

**分情况**

1. **远程只有master分支**

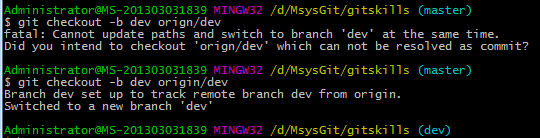
先创建dev分支，在远程推送



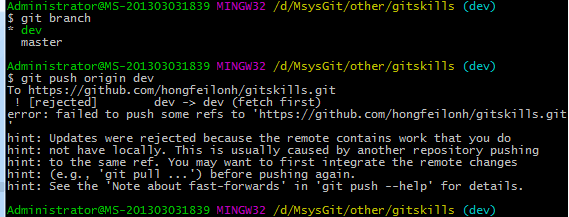
要输入登录名hongfeilonh 密码hong12345

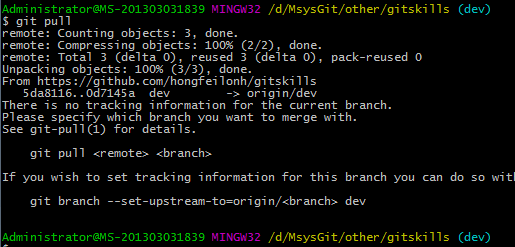
1. **远程有master，dev分支**

**克隆后，得到dev分支下的工作区**



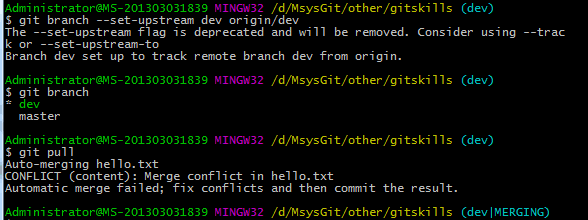
**Dev分支上修改，在推送**



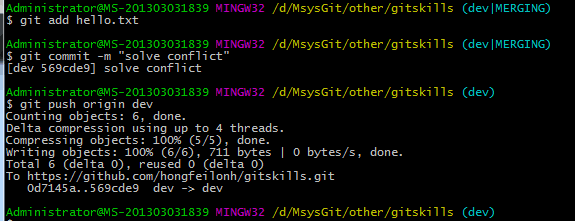


git pull也失败了，原因是没有指定本地dev分支与远程origin/dev分支的链接，根据提示，设置dev和origin/dev的链接：

**先指定dev，在pull**



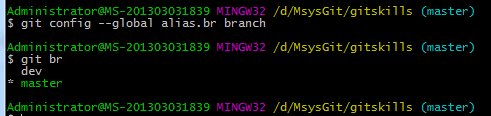
**这是有冲突，先解决，在commit，最后推送**



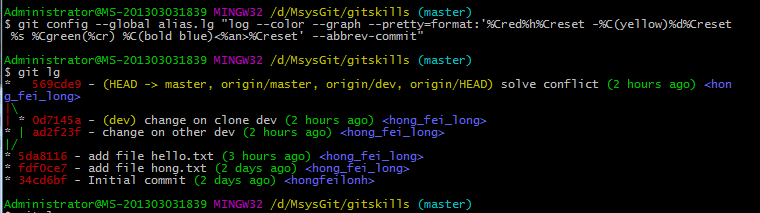
**查看远程分支得个数**



**配置别名**



配置Git的时候，加上--global是针对当前用户起作用的，如果不加，那只针对当前的仓库起作用



**//命令行处理远程分支**

1. **删除远程分支 ：git push origin :remote\_name**
2. **查看远程分支列表：git branch -r**
3. **将远程跟新到本地：git remote update**
4. **将本地分支切换到目标分支：git checkout -b local\_name origin/remote\_name**

**总结MsysGit**

|  |  |  |
| --- | --- | --- |
| **命令** | **作用** | **表现** |
| git config --global user.name "name" |  |  |
| git config --global user.email "email" |  |  |
| git config --global --list | 查看设置的name，email |  |
| git help |  |  |
| pwd | 查看当前路径 |  |
| touch license.txt | 增加文件 |  |
| mkdir test\_git | 增加文件夹 |  |
| cd test\_git | 进入路径 |  |
| git init | 对当前路径进行管理 |  |
| git add readme.txt | 将文件添加到存储区 |  |
| git commit -m "message" | 将暂存区提交到版本库中 |  |
| git status | 1. 比较工作区和存储区，不同提示 2. 比较暂存区和版本区，不同提示 |  |
| git diff readme.txt | 比较工作区和暂存区的异同 |  |
| git diff HEAD -- readme.txt | 比较工作区和版本库的异同 |  |
| git log | 当前分支下的提交版本 |  |
| git log --pretty=oneline | 当前分支下提交版本单行显示 |  |
| git reset --hard HEAD^ | 回到当前分支的上次版本 |  |
| git reflog | 记录commit，reset，  merge，checkout， |  |
| git reset --hard e8f9833 | 可以使当前分支回到任意分支的版本上 |  |
| git reset HEAD helloworld.txt | 撤销暂存区的文件 |  |
| git checkout -- helloworld.txt | 撤销工作区文件，回到上次add或commit状态（误删、改） |  |
| rm test.txt 或 git rm test.txt | 删除工作区文件 |  |
|  |  |  |
|  |  |  |

**操作**

**git add readme.txt**

**git commit -m "wrote a readme file"**

**Git rm readme.txt**

**帮助**

**ls**

**git status**

**git diff readme.txt**

**git log**

**git log --pretty=oneline**

**操作**

**git reset --hard HEAD^**

**git reset --hard f9ca767**

**帮助**

**git reflog**

**git diff HEAD -- readme.txt命令可以查看工作区和版本库的区别**

**git diff readme.txt //查看工作区和存储区的区别**

**操作**

**git checkout -- helloworld.txt //撤销工作区文件**

**git reset HEAD helloworld.txt //撤销暂存区的helloworld.txt文件**

**rm test.txt //删除工作区文件**

**git rm test.txt //选中要删除的文件**

**git push -u origin master**

**git clone git@github.com:hongfeilonh/gitskills.git**

**git branch dev //创建分支dev**

**git checkout dev //切换到分支dev**

**git checkout -b dev //创建dev分支，然后切换到dev分支**

**git branch**

**git merge dev**

**git branch -d dev //删除分支**

**git branch -D dev**

**Git merge --no-ff -m “message” e //非快速合并**

**//命令行处理远程分支**

**1.删除远程分支 ：git push origin :remote\_name**

**2.查看远程分支列表：git branch -r**

**3.将远程跟新到本地：git remote update**

**4.将本地分支切换到目标分支：git checkout -b local\_name origin/remote\_name**

**5拉取 git pull**