# Git笔记

## 命令

### git init目录变成Git可以管理的仓库,初始化

### git add test.txt 添加到仓库

### git commit –m “第一次提交” 提交

-m后面是备注

### git status 查看当前仓库的状态

是否有文件修改，是否有未提交等

### git log 版本历史记录

### git reset --hard head^^往后退两个版本

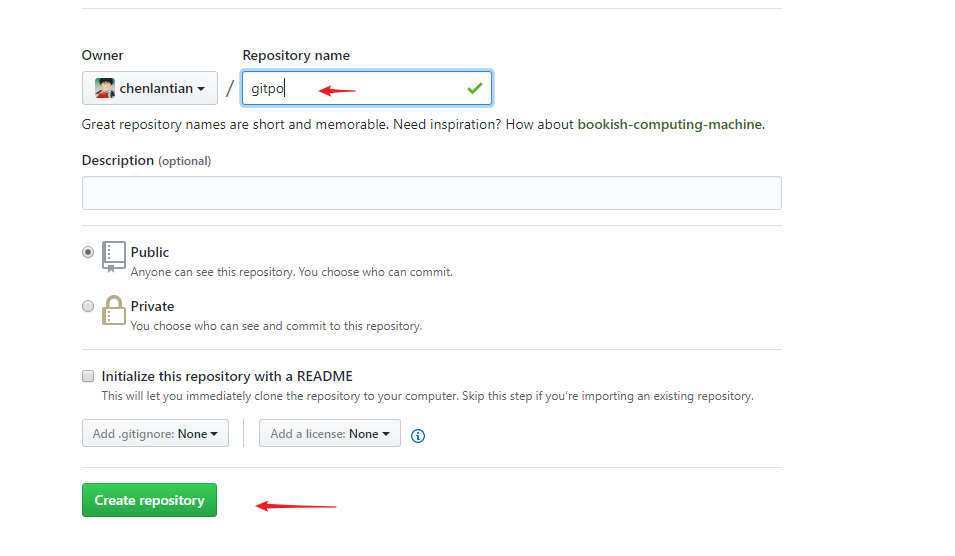
上一个版本就是HEAD^，上上一个版本就是HEAD^^，当然往上100个版本写100个^比较容易数不过来，所以写成HEAD~100。

### git reflog 查看操作记录

### git reset – hard comit\_id 回到制定ID的版本

### 添加远程仓

先new gitpo



git git remote add origin <https://github.com/chenlantian/gitpo.git>

或者

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

推送到远程

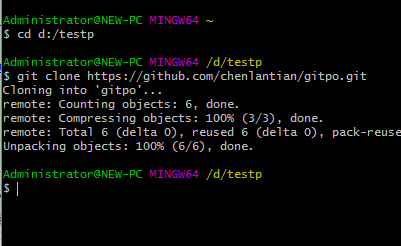
$ git push -u origin master

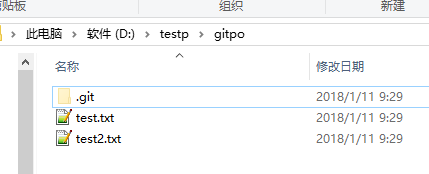
把本地库的内容推送到远程，用git push命令，实际上是把当前分支master推送到远程。

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不但会把本地的master分支内容推送的远程新的master分支，还会把本地的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。

### 克隆远程仓

$ git clone <https://github.com/chenlantian/gitpo.git>





### 创建与合并分支

* 查看分支：git branch
* 创建分支：git branch <name>
* 切换分支：git checkout <name>
* 创建+切换分支：git checkout -b <name>
* 合并某分支到当前分支：git merge <name>
* 删除分支：git branch -d <name>
* 强制删除未合并的分支git branch -D <name>

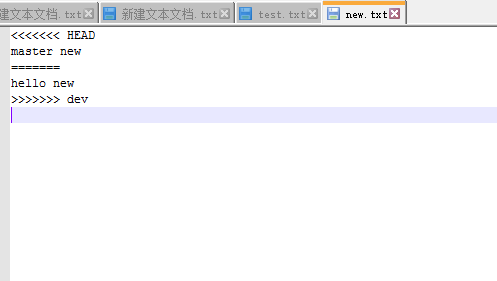
### 解决冲突

**产生冲突原因:**

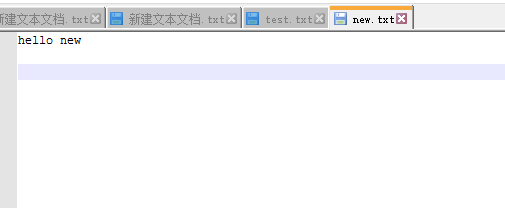
当分支dev修改并提交了new.txt文件,此时切换到master分支,也修改并提交new.txt文件.最后使用合并命令,此时产生冲突.

解决:

找到冲突合并产生冲突的文件



修改为:



提交

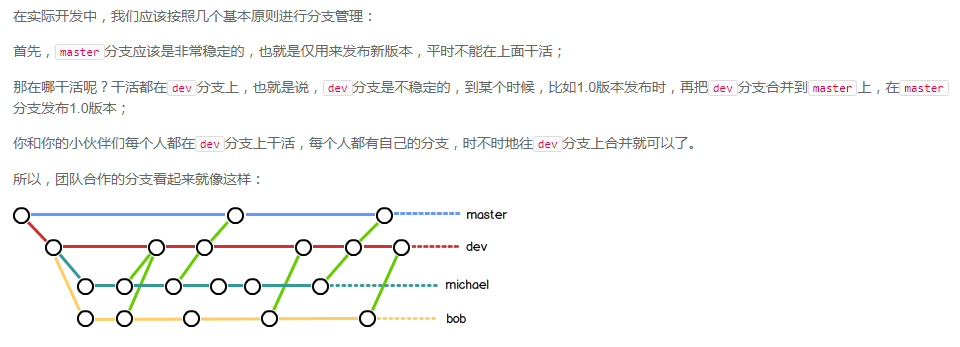
再删除dev分支

**小结**

当Git无法自动合并分支时，就必须首先解决冲突。解决冲突后，再提交，合并完成。

用git log --graph命令可以看到分支合并图。

### 分支管理策略



### Bug分支以及工作现场保留

**场景:**

正在dev分支开发某个功能,开发到了一半,此时接到来自master分支存在一个bug需要修复,此时使用 stash命令保留dev现场工作,切换到master,创建一个fixbug分支,修复完成合并fixbug分支,删除fixbug,切换到dev, gti stash pop恢复现场工作,继续接着开发

* 查看现场:git stash list
* 恢复现场:

1. git stash apply恢复后，stash内容并不删除
2. git stash pop恢复后，stash删除