Git命令行学习

创建版本管理库并且添加文件：

Git init: 创建当前的版本管理库
Git add:添加当前的文件
Git commit -m “XXXX” : 把当前的文件添加到版本管理库里面

回溯之前的文件，只要把head pointer指向想要被回溯的那个文件

Git reset – -hard commit_id : commit id指的是之前的某次快照提交
Git log:查看提交的历史
Git reflog: 查看命令的历史

暂存区和master branch: git的workflow:

先把工作区的文件用git add添加到暂存区（stage）,在暂存区，再用git commit -m,一下子把所有文件都推送到master branch

撤销修改，分为两种情况，一种是在working directory直接修改，第二种是已经commit,在stage的时候，撤销修改会回到第一次添加的到stage时候的状态。用的命令行如下：

Git restore +文件名

删除某个文件：

创建某个文件，git add ,git commit,之后到版本库中，如果working directory想要删除这个文件，直接rm + file_name就可以了。

1. 如果想在版本库中删除这个文件，变得和working directory一致，那么需要先删除，再commit

Git rm readme.txt
Git commit -m “Remove the readme file from git”

2. 如果想撤回working directory的操作，用git restore + file_name

和github远程仓库做关联

1. 在github上建立一个远程仓库： github create repo
2. 把本地仓库和远程仓库关联
   本地运行： git remote add origin git@XXXXXX
3. 把本地库的内容推送到远程库上
   本地运行：git push -u origin master

//github上面的repo默认名字是origin

之后本地做了提交，就只需要通过命令：git push origin master提交到github

4. 如果想要删除远程库，就用命令：
   Git remote rm origin

从远程clone一个repo

Git clone http:XXXX (This link is from github)

创建并且合并分支

背后逻辑：master / main分支就是我们一直track并且和remote sync的分支，当我们想创建新的分支的时候，可以一直在新的分支里面进行修改，修改完成之后，上传到新的分支，然后再用master的指针指向新的分支，这个时候新的分支就可以删除了。

命令行逻辑：
Git branch dev : 创建dev这个分支
Git checkout dev:将指针事项dev这个分支
Git branch:查看现在的分支，有*号的，代表目前指针只在这个分支。这个命令会列出所有分支
在dev分支进行一些操作，通过git add, git commit进行最终的提交
Git checkout master 提交完成之后，切换到master到分支：
这个时候我们查看master分支，发现刚才的改变并没有显现。那是因为我们一开始是在dev分支做改变的。我们只需要把dev的工作成果合并到master分支就可以了
Git merge dev, 这句话的意思是把master指针指向dev
Git branch -d dev：放心地删除dev分支
Git branch:这个时候我们查看分支，发现只剩下master分支了

这个方法的逻辑和debug很像，可以现在一个新分支上面debug，然后提交到当前分支，再和master merge

无法合并分支的情况：

背后逻辑：如果在新的分支上更改了一个文件，在master上对同一个文件做了修改，这个时候这个文件在两个不同的分支上作出修改，直接merge会出错。对应的方法应该是，打开这个文件，让两个更改做出统一，然后再git add, git commit更新到版本库，再删除刚才新的分支。注意更改同一份文件的时候，用vim editor打开可以看到两次不同更改内容，用>>>>> ===== <<<<<等符号分隔开。

如果要设计一个新feature,最好建立一个新的feature分支，如果要丢弃一个没有被合并过的分支，需要用到如下命令：

Git branch -D <name>

多人协作：

1. 如果要查看远程库的信息，用git remote -v
2. 推送分支：
    a. 把当前分支上所有内容推送到远程： git push origin master
    b. 推送其他分支dev到origin: git push origin dev

实际工程中，master是主分支，要和远程保持一致。dev分支是开发分支，所有的成员都在上面工作，也要推送到远程。bug分支只要在local修改就可以。feature分支是否推送到远程，取决于你和小伙伴们的开发进度。