在这之前先了解github网站上可以进行的git命令操作。

Github工具

安装好后，在命令行用git –version来验证安装是否成功。

打开方式：在想要新建工程的文件夹下右键点击就行。

Github工具的配置：（分为全局的和项目的配置）

查看配置信息：git config –list

设置用户名：git config –global user.name “name” name为自己在github上面创建的用户名

设置用户邮箱: git config –global user.email [1757207196@qq.com](mailto:1757207196@qq.com) //这个是邮箱

设置这两个的原因，因为需要将本地的厂库提交到github上面。用于区别哪些代码是那个人贡献的

Git工具光标，中文等的设置。。。。。

上面的配置信息保存在主文件夹下的名为.gitconfig的隐藏文件中

另外：git config –list列出来的其他信息分别还有什么用。

《二》命令行点的操作

命令行的换行：用符号 \

命令行的终止：ctrl + c

命令行的移动操作等：和vim的命令差不多。如用git log进入log文本，相当于是进入vim操作了

Linux bash的快捷键

**Ctrl-E 相当于End键，即将光标移动到本行末尾**

**Ctrl-D 相当于Del键，即删除光标所在处的字符  
Ctrl-K 用于删除从光标处开始到结尾处的所有字符**

**Ctrl-L 清屏，相当于clear命令  
Ctrl-R 进入历史命令查找状态，然后你输入几个关键字符，就可以找到你使用过的命令  
Ctrl-U 用于删除从光标开始到行首的所有字符。一般在密码或命令输入错误时常用**

**《三》**

**Git工具命令的好处：git在github网站上进行的操作不能进行同时多个提交、创建文件夹等操作**

**所以推荐使用本地客户端的git命令工具。同时在git命令行下可以使用bash脚本文件进行批处理。同时可以在**

**本地端使用其他的编译器，编辑器进行编写好程序、文档然后在上传到github网站上，这样有格式比较便捷。**

在git工具中，工作区、暂存区、仓库之间的关系见链接<http://marklodato.github.io/visual-git-guide/index-zh-cn.html>

具体的命令：

Git init //在当前目录新建一个代码库（注意目录是在你电脑上右键点击的目录？？），还是自己在命令行下通过cd命令修改的目录(为红色正确)。

Git clone [URL] # [https://github.com/[user.name]/[user.reposname](https://github.com/%5buser.name%5d/%5buser.reposname)] #即从github网站上克隆有一个已有的仓库（包括仓库的历史）到本地。

Git add files1 files2 //添加指定文件到暂存区中

Git rm files1 //删除本地文件files1，同时将该删除的文件添加到暂存区中

Git mv [file-name1] [file-name2] //将原文件名为file-name1的文件更改为file-name2，同时将该文件放到暂存区中

Git commit –m ‘’message’’ //将暂存区中的所有文件提交到仓库中，message为这次提交时添加的说明信息。后面会有说提交的格式

Git commit –a –m ‘’message’’ //将本地文件直接提交到仓库中

查看每一次的变动信息

git status

git log (--online)

怎么在本地端删除一个已经创建的仓库？直接将该仓库所在的文件夹删除吗？？

例子：

基本操作的流程。

第一步：通过bash命令进入到/g/git\_re/,然后在该文件夹下创建demo1文件夹

第二步：使用git init 命令初始化一个Git代码空间，查看git status

$ git status

On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

第三步：使用vi命令编辑创建一个README.md文件，并在里面添加the first github project in local

第四步：使用cat README.md（可以使用bash的命令行补全，table键）。查看README.md文件是否正确创建

第五步：使用git status 命令查看仓库状况。

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

README.md

nothing added to commit but untracked files present (use "git add" to track)

创建的本地文件没有任何提交，变成为红色

第六步：使用git add README.md将本地文件添加到暂存区。

$ git add README.md

warning: LF will be replaced by CRLF in README.md.

The file will have its original line endings in your working directory.

问：这里为什么会存在警告？应该怎么解决呢？

第七步：通过git status 查看当前仓库状况。

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

new file: README.md

本地文件进入暂存区后变成绿色

第八步：通过git commit –m “the first commit”,将暂存区的内容添加到仓库中。

$ git commit -m "the first commit"

[master (root-commit) 22b3410] the first commit，这一句为添加的信息。

1 file changed, 1 insertion(+)

create mode 100644 README.md

第九步：通过git status 查看当前仓库状况。

$ git status

On branch master

nothing to commit, working tree clean

所有修改过的文件都已经提交到仓库了，所以树是干净的。

基本操作外的操作：

第十步：vi README.md 在里面添加the first modify in local。这里即修改了源文件

第十一步：git status

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: README.md

no changes added to commit (use "git add" and/or "git commit -a")

这里修改了的文件重新变成红色。

第十二步：git commit –a –m “push local file to repos directly”.直接将文件从本地提交到仓库

第十三步：git status

$ git status

On branch master

nothing to commit, working tree clean。

较高级操作：

查看不同时刻提交的版本（简要信息）

第十四步：git log （使用该命令查看所有的提交信息，每次提交的哈希值，每次提交的时间）

第十五步：使用git show，默认显示最后一次的提交时的详细信息，包括增加或删除了哪些内容。

第十六步：怎么操作查看具体的某一个版本的详细变动情况？先使用git log 得到目标版本的哈希值值

在用git show 上面得到的哈希值即可。

问怎么将当前的git仓库版本退回到之前的某一个版本（相应的本地的文件也退回到之前的版本）。在上面的步骤中进行的操作都是前进的操作，即将本地文件推向仓库。而没有进行将仓库里的文件退回到本地当中。

注意：上面的操作不管是本地文件，暂存区，仓库都是在本机电脑上，都还没有将这些东西推送到远程github网站上。

《将本地的仓库推到远程github上，将远程的github网站上的文件克隆到本地》

同步远程仓库的相关命令

增加远程仓库并命名：

git remote add [shortname] [url]

shortname就是一个变量，变量的内容为URL网址，该网址为https://github.com/user.name/user.repos

注意上面的user.repos需要在github网站上进行先创建才是，否则传入会失败。建议将本地端的仓库与github端的仓库

命名为一致。

将本地仓库添加到远程github网站上面

Git push [remote] [branch]

Remote 为上面的shortname，branch为分支，一般master为主分支。在多人的项目协作中，branch不止这一支。

将远程仓库拽到本地中

Git pull [remote] [branch]

这里的内容和上面的语句的内容也是一样的。

实例操作：将上面创建的本地仓库推送到github网站。本地到远程

第一步：在github网站上新建一个空白的仓库demo1，不要添加README.md文件

第二步：在git bash命令行下git add origin <https://github.com/larioy/demo1>

第三步：输入你在github上的用户名和密码（第一次需要，非第一次不需要）。

实例操作：在github网站上对demo1的README.md文件进行修改操作再拽回本地。

第一步：在github网站上的demo1的README.md文件添加一行，add a line to README.md in github website.并保存

第二步：在git bash工具终端输入git pull origin master.

第三步：使用git status 查看仓库的最后修改状态，使用git log查看仓库的修改版本，使用cat README.md文件确认该文件

是否是github上已经增加了add a line to README.md in github website.这一行的最新版本。

疑问：怎么从远程的github网站上面下载之前版本的仓库而不是下载当前最新的仓库呢？

简单的将本地与远程建立联的操作：

注意：使用该方式不需要再本地新建一个与远程克隆仓库名相同的文件夹，因为在克隆的时候会自动生成同名文件夹，同名文件夹在当前目录下面，这个要特别注意的，否则怕找不到。

第一步：Git clone <https://github.com/larioy/user.repos>

前提是要在github网站上新建一个仓库或者是一个已经存在的仓库。

第二步：使用git remote –v就可以查看本地端与远程段是否建立关系

第三步：使用cd命令进入到新克隆自动创建的文件夹下。

第四步：在文件夹下使用vi change.md 并写入 write in local这一行

第五步：对新创建的文件提交到本地仓库：git commit –a –m “commit new file to local repos.”

第六步：将更新后的本地仓库推送到远程：git push。这里要注意因为克隆的时候已经建立关联了，所以直接使用git push。