普通GIT命令

# Git init

初始化，在需要git管理的文件夹用此命令

# Git add

使用命令git add <file>，注意，可反复多次使用，添加多个文件；

# Git commit

使用命令git commit -m <message>，完成提交。

# Git status

要随时掌握工作区的状态，使用git status命令。

# Git diff

用git diff可以查看修改内容。

# Git reset

* HEAD指向的版本就是当前版本，因此，Git允许我们在版本的历史之间穿梭，使用命令git reset --hard commit\_id。
* 当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃暂存区中的修改，用命令git reset HEAD <file>。

# Git log

1. 穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。
2. 图形化显示：可以看出各个分支的排布

git log *--graph --pretty=oneline --abbrev-commit*

# Git reflog

要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

# Git checkout

* 当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout – file，此时会用缓存区内的内容替代。
* git checkout <name>:切换分支
* 如果加上-b参数，表示先创建再切换

# Git rm

命令git rm用于删除一个文件。如果一个文件已经被提交到版本库，那么你永远不用担心误删，但是要小心，你只能恢复文件到最新版本，你会丢失最近一次提交后你修改的内容。

# Git restore <file>

恢复工作区内的file至未修改状态

# Git branch

查看分支

# git branch <name>

创建分支

# git branch -d <name>

删除分支

# git merge <name>

(1)合并某分支到当前分支

(2) git merge --no-ff -m "新建的commit的名字" 合并分支的名字

--no-ff表示采用普通合并（非快速合并），即合并之前会先生成一个新的commit。

# git stash

可以把当前工作现场“储藏”起来，等以后恢复现场后继续工作

Github相关命令

# 创建本地和远程的连接

（1）

由于你的本地Git仓库和GitHub仓库之间的传输是通过SSH加密的，所以，需要一点设置：

创建SSH Key。在用户主目录下，看看有没有.ssh目录，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，如果已经有了，可直接跳到下一步。如果没有，打开Shell（Windows下打开Git Bash），创建SSH Key：

$ ssh-keygen.exe -t rsa -C "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。

（2）

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：点“Add Key”，你就应该看到已经添加的Key。

# 添加远程库

（1）

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。

在Repository name填入learngit，其他保持默认设置，点击“Create repository”按钮，就成功地创建了一个新的Git仓库。

（2）

现在，我们根据GitHub的提示，需要把一个已有的本地仓库与之关联，在本地的仓库下运行命令：

$ git remote add origin 网址

“origin”是默认的远程库的名字，也可以取其它喜欢的名字。自此，origin即和远程网址仓库在一起。

网址：例如：<https://github.com/kangchi0225/learngit.git>

（3）

接下来，需要将本地内容推送到远程：

$ 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，origin是默认的远程仓库名称。

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。

# 使用GitHub

* 在GitHub上，可以任意Fork开源仓库；
* 自己拥有Fork后的仓库的读写权限；
* 可以推送pull request给官方仓库来贡献代码。

# 一些命令

## git remote –v

查看远程库的信息

## git clone <uri>

克隆网址uri到本地。默认远程库名字取为origin。

## git remote remove <name>

删除已经存在的远程库。例如git remote remove origin

## git remote add <name> <uri>

添加新的远程库。Name可任意自己取名字，添加后，以后就可以用改名字代表远程库uri了。例如：

git remote add origin https://github.com/kangchi0225/learning-notes

## git push <remote\_name> <local\_name>

格式：git push 远程仓库名字 本地分支名字

例如：git push origin master

## git pull <name>

抓取远程的新提交的内容。

Name：(1)远程网址例如：<https://github.com/kangchi0225/learngit.git>

(2)远程库的名字，例如origin(这个名字是在git remote add 中取得名字)

## git branch --set-upstream <branch-name> <remote-name>/<branch-name>

建立本地分支和远程分支的关联

## git checkout -b branch-name origin/branch-name

在本地创建和远程分支对应的分支，使用此命令，本地和远程分支的名称最好一致。

一些问题

# SSL certificate problem: unable to get local issuer certificate

完整错误提示：

错误：fatal: unable to access 'https://xxx.xxx.com/xxx/xxx.git/': SSL certificate problem: unable to get local issuer certificate

原因：服务器的SSL证书未经过第三方机构签署  
解决方法：可以忽略SSL验证解决该问题

git config --global http.sslVerify false