**跟着廖雪峰老师练习的git使用笔记**

使用前：

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：

第1步：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 -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。

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：

**1.Git安装完成后设置用户和邮箱：**

git config –global user.name “ryan1943”

git config –global user.email [chenwd\_135@qq.com](mailto:chenwd_135@qq.com)

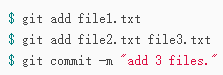
**2.初始化**

新建目录，在里面初始化一个Git仓库，使用git init命令。

添加文件到Git仓库，分两步：

第一步，使用命令git add <file>添加修改，可以添加多次

第二步，使用命令git commit提交修改



**3. 要随时掌握工作区的状态，使用git status命令**。

如果git status告诉你有文件被修改过，用git diff可以查看修改内容。

**4. 版本回退**

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。

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。

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。



**5. 工作区和暂存区**

git add命令实际上就是把要提交的所有修改放到暂存区（Stage），然后，执行git commit就可以一次性把暂存区的所有修改提交到分支。

每次修改，如果不add到暂存区，那就不会加入到commit中。

**6. 撤销修改**

场景1：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。例如：git checkout – test.txt

场景2：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，想丢弃修改，分两步，第一步用命令git reset HEAD file-name，就回到了场景1，第二步按场景1操作。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考版本回退一节，不过前提是没有推送到远程库。

**7.文件删除**

工作区删除但没有提交到暂存区：rm test.txt 用git checkout – test.txt恢复

工作区删除且提交到暂存区：git rm test.txt 用git checkout HEAD – test.txt

**8.添加远程仓库和推送**

登录GitHub，把公钥添加到ssh里面，就可以确认只有你自己才能推送。

在GitHub新建一个仓库learngit

要关联一个远程库，使用命令git remote add origin git@github.com:ryan1943/learngit.git

关联后，使用命令git push -u origin master第一次推送master分支的所有内容；

此后，每次本地提交后，只要有必要，就可以使用命令git push origin master推送最新修改；

**9.从远程仓库克隆**

使用命令：git clone git@github.com:ryan1943/learngit.git

即可在当前目录下生产一个远程仓库名字一样的文件夹

**10.分支管理**

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>

查看分支合并情况：git log --graph --pretty=oneline --abbrev-commit

删除分支：git branch -d <name>

丢弃一个没有被合并过的分支，可以通过git branch -D <name>强行删除

**11.分支管理策略**

合并分支时，如果可能，Git会用Fast forward模式，但这种模式下，删除分支后，会丢掉分支信息。如果要强制禁用Fast forward模式，Git就会在merge时生成一个新的commit，这样，从分支历史上就可以看出分支信息。下面我们实战一下--no-ff方式的git merge：

git merge --no-ff -m "merge with no-ff" dev

**12.保护工作现场**

存储当前工作现场：git stash

查看被保护起来的现场：git stash list

恢复并删除：git stash pop

恢复指定的现场但不删除：git stash apply <name>

删除指定的现场：git stash drop <name>

**13在GitHub上建立分支**

在本地新建一个分支： git branch Branch1

切换到你的新分支: git checkout Branch1

将新分支发布在github上： git push origin Branch1

在本地删除一个分支： git branch -d Branch1

在github远程端删除一个分支： git push origin :Branch1 (分支名前的冒号代表删除)

**14.多人协作**

查看远程库信息：git remote -v

在本地创建和远程分支对应的分支：git checkout -b branch-name origin/branch-name

从本地推送分支，使用git push origin branch-name，如果推送失败，先用git pull抓取远程的新提交

建立本地分支和远程分支的关联，使用git branch --set-upstream branch-name origin/branch-name；

从远程抓取分支，使用git pull，如果有冲突，要先处理冲突。

**15.标签**

命令git tag <name>用于新建一个标签，默认为HEAD，也可以指定一个commit id；

git tag -a <tagname> -m "blablabla..."可以指定标签信息；

git tag -s <tagname> -m "blablabla..."可以用PGP签名标签；

命令git tag可以查看所有标签。

命令git push origin <tagname>可以推送一个本地标签；

命令git push origin --tags可以推送全部未推送过的本地标签；

命令git tag -d <tagname>可以删除一个本地标签；

命令git push origin :refs/tags/<tagname>可以删除一个远程标签

**16.码云的使用**

关联远程仓库：git remote add origin [git@gitee.com:chenwandu/learngit.git](mailto:git@gitee.com:chenwandu/learngit.git)（报错，因为仓库名origin与GitHub的冲突）

删除远程仓库(如果有需要)：git remote rm origin

关联码云的远程仓库：git remote add gitee [git@gitee.com:chenwandu/learngit.git](mailto:git@gitee.com:chenwandu/learngit.git)(码云的远程仓库名是gittee)

查看远程库信息：git remote –v可看到可以看到两个远程库

推送到码云：git push gitee master git push gitee dev

**17.忽略特殊文件**

忽略某些文件时，需要编写.gitignore；

.gitignore文件本身要放到版本库里，并且可以对.gitignore做版本管理

强制添加被忽略的文件：git add -f App.class

查看问题：git check-ignore -v App.class

.gitignore的各种配置：

[**https://github.com/github/gitignore**](https://github.com/github/gitignore)

.gitignore规则不生效的解决办法

<http://blog.csdn.net/liunianerge/article/details/75308884>

[**https://segmentfault.com/q/1010000009420444**](https://segmentfault.com/q/1010000009420444)

代码合并：Merge、Rebase 的选择

<https://github.com/geeeeeeeeek/git-recipes/wiki/5.1-%E4%BB%A3%E7%A0%81%E5%90%88%E5%B9%B6%EF%BC%9AMerge%E3%80%81Rebase-%E7%9A%84%E9%80%89%E6%8B%A9>