Git 学习

内容整理自廖雪峰老师博客https://www.liaoxuefeng.com/wiki/89604388029600

Git 简介

分布式管理系统 无需联网 C写的

分布式

每个人的电脑都是一个完整的版本库 所有的版本控制系统只能跟踪文本文件改动
多人协作时，只需把自己的修改推送给对方

MacOS 安装 Git

"Xcode" -> "Preferences" -> "Downloads" -> "Command Line Tools" ->"Install"

创建版本库 repository

版本库：一个目录，目录里所有文件都可以被Git管理、跟踪

```
$ mkdir ~/Desktop/learngit
$ cd ~/Desktop/learngit
pwd
/Users/qq/Desktop/learngit
```

使用git init把这个目录变为一个Git可管理的仓库（虽然目前是空的 empty Git repository

```
$ cd ~/Desktop/learngit
$ git init
Initialized empty Git repository in /Users/qq/Desktop/learngit/.git
```

发现此目录下多了一个.git：Git用来跟踪版本管理库

把文件添加到版本库

1. 在learngit目录下编写任意内容txt文件 e.g.learngit_0615.txt
2. 命令行输入git add file把文件提交到仓库

```
$ git add learngit_0615.txt
```

3. 命令行输入git commmit -m memo把文件提交到仓库 commit一次可以提交多个文件

```
$ git commit -m"Ttry to add file to git"
```

```
[master (root-commit) 7d15f7c] Ttry to add file to git
 Committer: qq <qq@qiyongruideMacBook-Pro.local>
 on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 6 insertions(+)
 create mode 100644 learngit_0615.txt
```

修改文件之后

运行git status查看结果

```
$ 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:   learngit_0615.txt

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

    .DS_Store

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

运行git diff查看修改了啥

```
$ git diff learngit_0615.txt
-This .txt aims to try to add file to Git
\ No newline at end of file
+This .txt aims to try to add file to Git
+
+Git is a distributed version control system.
+Git is free software.
\ No newline at end of file
```

再把learngit_0615提交到仓库

版本回退

Git内部有指向当前版本的HEAD指针，版本回退时，仅HEAD指向变化
版本控制系统中告诉我们历史记录的命令: git log

```
$ git log
commit e586d1c6c7945eb18b36993e92f076b623a30386 (HEAD -> master)
Author: qq <qq@qiyongruideMacBook-Pro.local>
Date:   Mon Jun 15 18:14:16 2020 +0800

    Add distributed and free

commit 7d15f7ce4c024b1be927d2bed05921bb2a4b1332
Author: qq <qq@qiyongruideMacBook-Pro.local>
Date:   Mon Jun 15 17:59:56 2020 +0800

    Ttry to add file to git
```

git log -pretty=oneline 只输出每一次提交的commit id:

```
$ git log --pretty=oneline
e586d1c6c7945eb18b36993e92f076b623a30386 (HEAD -> master) Add distributed and
free
7d15f7ce4c024b1be927d2bed05921bb2a4b1332 Ttry to add file to git
```

Git中，HEAD表示当前版本，HEAD^表示上一个版本，版本回退依次➕ ^ 即可

```
$ git reset --hard HEAD^
HEAD is now at 7d15f7c Ttry to add file to git
```

若想回到最新版本：从git reflog 中找到后一个版本的commit id, 将 --hard重置

```
$ git reflog
7d15f7c (HEAD -> master) HEAD@{0}: reset: moving to HEAD^
e586d1c HEAD@{1}: commit: Add distributed and free
7d15f7c (HEAD -> master) HEAD@{2}: commit (initial): Ttry to add file to git
```

```
$ git reset --hard e586d1c
HEAD is now at e586d1c Add distributed and free
```

工作区和暂存区

工作区：电脑里的目录 e.g.learngit文件夹
git add 把要提交的修改放到暂存区，git commit 一次性把暂存区中所有修改提交到master分支
（创建Git版本库时，Git自动创建master分支

管理修改

git跟踪管理的是修改而非文件
git commit提交的是暂存区的修改，未git add的修改不会被提交

撤销修改

```
$ git checkout --filename
```

- 若修改后还没有被放到暂存区（没有git add）：git checkout -- 之后回到和版本库中一样的状态
- 若添加到暂存区又做了修改：git checkout -- 之后恢复到添加到暂存区之后到状态
  git checkout 若无 --：切换到另一个分支命令
  git checkout --: 用版本库中的最新版本替换工作区的版本
  `$ git reset HEAD filename`同样可以撤销暂存区的修改
  （暂存区修改未被提交，HEAD指针无法指向新的修改，重置HEAD可回退到工作区，再用`$ git checkout --filename`丢弃工作区的修改

删除文件

在文件管理器中删除文件：`$ rm filename`

- 若要在版本库中删除该文件：先`$ git rm filename`，再用`$ git commit`提交
- 若要在文件管理器中恢复该文件：`$ git checkout --filename`
  （git checkout --: 用版本库中的最新版本替换工作区的版本

远程仓库

查看我本地仓库和GitHub仓库间的SSH加密

```
$ cat /Users/qq/.ssh/id_rsa.pub
```

若创建SSH：`$ ssh-keygen -t rsa -C email@example.com`
若要再生成一个SSH：Enter file in which to save SSH key时重新输入目录即可

添加远程仓库

```
$ git remote add summer2020 git@github.com:wojile/learning-records.git
```

添加远程仓库之前，一定要cd到learngit文件夹，然后git init❗
把本地仓库内容推送到远程仓库：

```
$ git push -u summer2020 master
```

初次使用需使用-u，之后直接输入`$ git push summer2020 master`即可

从远程仓库克隆

```
$ git clone git@github.com:user/repository.git
```

或者不使用SSH协议，使用https:

```
$ git clone https://github.com/user/repository.git
```

分支管理

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>` or `git switch <name>`
创建+切换分支：`git checkout -b <name>` or `git switch -c <name>`
合并某分支到当前分支：`git merge <name>`
删除分支：`git branch -d <name>`
Git把每次提交串成时间线，此时间线即为master分支。
Head并不是直接指向提交，而是只想master，master沿时间线移动指向不同提交。
若在B分支上工作完成，想把B合并到master：让master指向B当前的提交即完成合并。

解决冲突

当git无法执行快速合并时，会用`<<<<<<<<<<HEAD` `==========` `>>>>>>>>>>new branch`标记出冲突，需手动解决，再提交
用`git log --graph`可看到合并分支图

分支管理

在master分支发布
在dev分支干活（合并
在各自分支做自己的，每一部分完成后向dev合并

Feature分支

丢弃没有被合并过的分支：`git branch -D <name>`强行丢弃

标签管理

用简短tag 绑定一长串commit id

创建标签

切换到需要打标签的分支直接创建即可：

```
git checkout master
git tag <name>
```

标签默认打在最新提交的commit（HEAD）上
`git tag -a <tagname> -m "..."`可指定标签信息
`git tag`可查看所有标签

操作标签

删除本地标签：`git tag -d <tagname>`
删除远程标签：`git push summer2020 :ref/tags/<tagname>`
推送一个本地标签：`git push summer2020 <tagname>`
推送全部本地标签：`git push summer2020 tags`

自定义Git

忽略特殊文件

在Git工作区的根目录下创建一个特殊的.gitignore文件，然后把要忽略的文件名填进去，Git就会自动忽略这些文件。
.gitignore本身也要提交到git，也可以对gitignore做版本管理

配置别名

比如：

```
$ git config --global alias.co checkout
$ git config --global alias.ci commit
$ git config --global alias.br branch
```

SourceTree

Git图形工具界面