GitHub

Git简介

GitHub和Git

Git是软件，GitHub是网站。

GitHub的功能

个人

储存代码，历史代码找回，从历史版本生出新的版本

多人

权限管理，分支管理

Git命令行

版本控制

查看git configuration

```
$ cd ~
$ cat .gitconfig
```

status

```
$ git status
```

显示文件到变化，比如deleted, modified等等。红色表示没加到暂存区里，绿色表示加进去了

```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
  deleted:    test.rtf

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

add

添加文件到working directory里

```
$ git add <filename>
```

或者直接添加所有文件

```
$ git add .
```

commit

```
$ git commit <filename>
```

进入vim界面，添加本次提交到注释。或者

```
$ git commit -m "内容" <filename>
```

对某个文件添加注释，或者

```
$ git commit -m "内容"
```

对所有文件添加同样的注释

log

用来查看历史版本

```
$ git log
```

显示

```
commit 51152592bfeab2b1dba050cf0a9ff10b0e9c64a5 (HEAD -> main, origin/main)
Author: HaibinZHAO <zhb950102@qq.com>
Date:   Fri Apr 16 16:22:24 2021 +0200

    update

......省略

commit 501b37b430cf2c78f57d59af937a15f61a72950b
Author: HaibinZHAO <zhb950102@qq.com>
Date:   Thu Apr 15 17:34:21 2021 +0200

    add algorithm for trajectory
```

可以用

```
$ git log --oneline
```

用一行显示一个历史记录

```
5115259 (HEAD -> main, origin/main) update
cc0994b test
3a9a95b update
501b37b add algorithm for trajectory
4679042 rename video of trajektory
73bbc3a update
0cfeb76 update
1ec5311 Initial commit
```

也可以用

```
$ git reflog
```

还显示到当前版本的步数：

```
5115259 (HEAD -> main, origin/main) HEAD@{0}: commit: update
cc0994b HEAD@{1}: commit: test
3a9a95b HEAD@{2}: commit: update
501b37b HEAD@{3}: commit: add algorithm for trajectory
4679042 HEAD@{4}: commit: rename video of trajektory
73bbc3a HEAD@{5}: commit: update
0cfeb76 HEAD@{6}: Branch: renamed refs/heads/master to refs/heads/main
0cfeb76 HEAD@{8}: commit: update
1ec5311 HEAD@{9}: initial pull
```

reset

将指针指向历史版本

```
$ git reset --hard <hash value of version>
```

哈希值就是log里第一列的东西，例如

```
$ git reset --hard 4679042
>>> HEAD is now at 4679042 rename video of trajektory
```

diff

```
$ git diff <filename>
```

比较文件和暂存区的文件到区别

```
$ git diff <历史记录> <filename>
```

历史记录比如：HEAD^, HEAD~. 不加filename就查看所有文件

分支控制 Branch

查看分支

```
$ git branch -v
```

也能看到自己所在的分支

创建分支

```
$ git branch <branch name>
```

切换branch

```
$ git checkout <branch name>
```

合并Branch

例如将branch2合并到branch1，先进入branch1，然后用merge合并

```
$ git checkout branch1
$ git merge branch2
```

冲突解决

文件在2个分支中都被修改，则会在合并时产生冲突，比如认为选择其中之一

打开文件

```
$ vim <conflict filename>
```

会有

```
<<<<<<<<<< HEAD
<内容1>
==========
<内容2>
>>>>>>>>>> branch2
```

显示冲突的内容。解决方法：删除<<<<<, =====, >>>>>>，以及把内容1和内容2修改（至于怎么修改，认为决定）。之后再运行

```
$ git add <confict filename>
$ git commit -m "内容" #<不能加文件名>
```

和远程库的交互

准备工作

下载Git

本地配置

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

生成SSH公钥

```
$ ssh-keygen -t rsa -C "email@example.com"
```

找到生成的id_rsa.pub文件并将内容复制到GitHub里面：

GitHub主页 -> Account settings -> SSH Keys -> Add SSH Key

验证是否成功

```
$ ssh -T git@github.com
```

连接远程仓库

初始化本地仓库

在目录下

```
$ git init
$ git remote add origin git@github.com:xxx/project.git
$ git pull origin master
```

第2行的origin代表的就是后面那个地址，或者说origin是那个地址的**别名**

上传本地文件

```
$ git add .
$ git commit -m 'first commit'
$ git push origin master
```

克隆远程库

```
$ git clone <地址>
```

会自动

- 远程库下载到本地
- 创建origin别名
- 自动进行git的初始化

抓取别的分支fetch

```
$ git fetch origin <branch2>
```

然后可以用

```
$ git merge origin/<branch2>
```

合并2个分支

拉取别的分支pull

```
git pull origin <branch2>
```

GitHub合作流程

甲为领导者，乙和丙共同完成一个功能。

甲创建Repository，并且写好了框架，主branch为main，给乙和丙的branch为master。

乙和丙共同往master分支push东西，他们约定乙每月双数天工作，丙单数天工作。

因此乙每天工作完后将内容push上去，第二天丙将内容pull下来之后继续写代码，然后push上去，以此类推。

有一天乙脑子进了水，将2月5号误以为是双数天，于是和丙进行了相同的工作。

由于乙的效率较高，写完后把代码push上去了，因此，丙无法push了，因为两人写的内容产生了冲突。

丙打电话给乙，说：“ 你今天不应该工作。”乙说：“抱歉，那你来解决这个冲突吧！”

丙将（乙push上去的）分支pull下来，就显示有冲突，于是丙比较了他和乙的代码，发现乙的代码比自己写的好多了，于是默默将自己的代码删掉，merge了2个分支之后，push了上去。

功能完成后，甲fetch了master分支，经过比较，发现乙丙误删了另外一个与他们的功能无关的文件，但是其他的功能都基于这个文件，于是将删除文件这个行为去掉了，并将其他的部分和main分支merge在了一起。

乙和丙失业了。