之前我们操作过git，使用git 一般就几个步骤。

1. git clone [SSH地址]
2. git add .
3. git commit -m “提交信息”
4. git push origin master

接下来我们要介绍以下几种新操作：

拉取 pull

之前提到用git clone 来拷贝远程仓库里的文件，这个只使用于我刚创建出一个新的远程仓库的情况，当然这一次我也会新创建一个仓库，仍然可以用git clone拷贝下来，但是以后这个远程仓库就不再变了，也就是说以后每一次练习都是在同一个远程仓库下了。

那么以后就有可能出现这种情况了：

比如你先从远程仓库拷贝下我的代码文件，并修改完又推送回远程仓库了，接下来在你不知道的情况下，我又对远程仓库你的代码作了修改，我已经同步到远程仓库了，但是请注意你的本地电脑里还没有我修改过的痕迹，这时候你再修改就本地代码就推送不上了，因为这时候你本地的代码和远程的不同步。

那么怎么才能同步呢？

这时候就需要使用git pull 命令了

为了简单起见，我们接下来以后都统一使用一个分支master（分支的内容以后再说）

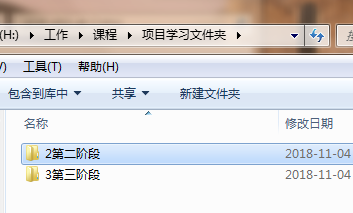
git pull

或者

git pull origin master

即可同步本地和远程的代码。

以后的学习项目目录文件管理如下：



我会把每一个阶段学习的内容统一推送到远程去，以后都是采用拉取来同步代码文件啦~~~~

也就是说以后就变为

git pull

git add .

git commit -m “提交信息”

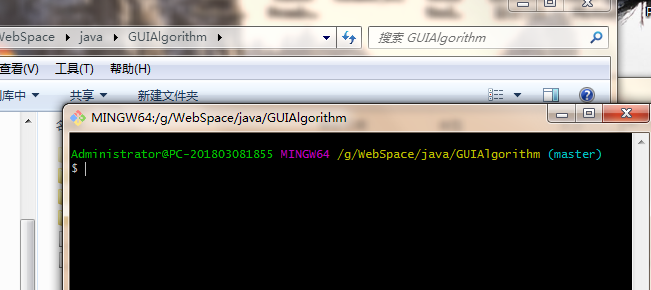
git push origin master

git 版本穿梭

在git 中有一个很重要的功能就是版本穿梭，那什么是版本呢？就像一个文档，修改来修改去，最后被要求说还是用最初的版本吧，怎么办，都修改成不知道最初的版本怎么样了！

在git 中实际上每一次提交，也就是使用git commit -m “提交信息”，都会产生一个版本修改记录，根据这个修改记录就可以随意撤回任意一个版本啦，那具体怎么来操作呢？

首先我们在文件夹中调取那个熟悉的黑框框

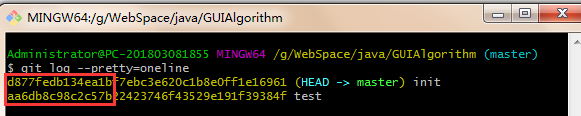


首先在这个项目中我已经提交了两次

可是我怎么拿到这两次的提交记录呢

一般我们简单点采用

git log --pretty=oneline



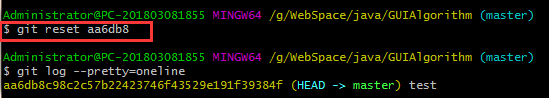
我们就可以看到这两次记录啦

其中黄色部分是修改的commit id 白色的是提交信息（绿色代表分支，蓝色代表当前版本），commit id 就是用于接下来版本穿梭的依据了

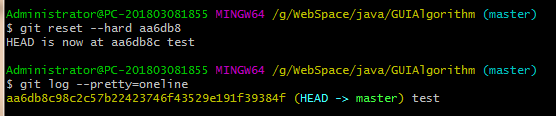
版本穿梭使用的是

git reset 【commit\_id】 其中commit id 一般只需要前6位即可

比如我要回到最初的test版本，就采用

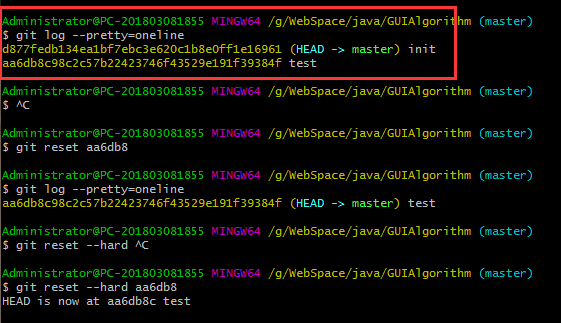


当然上述方法是软回退，我们还有另外一种回退方式是硬回退，只需要加上 --hard即可



软回退和硬回退有什么区别呢？可以先自己实践一下，我会在测评的区提问这个问题哦~~

那么回到最初的版本了，回得去最新版本吗？当然，只要你还保留有最新版本的commit id，（没有怎么办，也可以找回来，参考上节课的资料，但是非不得已别丢失吧）



（只要还有最初的那条记录即可）

