设置全局的用户名

$git config --global user.name FuckJava

设置全局邮箱，也就是本地签名，作用是区分不同操作者的身份，Git在首次启动时必须设置签名，否则无法commit(提交)

$git config --global user.email 13214212383@163.com

查看配置信息

$ cat ~/.gitconfig

[user]

name = FuckJava

email = [13214212383@163.com](mailto:13214212383@163.com)

初始化

$ git init

Initialized empty Git repository in D:/GitFile/.git/

初始化成功后会在D:/GitFile目录下多了一个隐藏文件

查看当前目录下所有文件和文件夹，隐藏文件也可以查看得到

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ ll -a

total 20

drwxr-xr-x 1 Fuck\_Java 197121 0 Apr 23 14:43 ./

drwxr-xr-x 1 Fuck\_Java 197121 0 Apr 23 14:30 ../

drwxr-xr-x 1 Fuck\_Java 197121 0 Apr 23 14:43 .git/

查看文件的名称，隐藏文件查看不到

$ ls

查看本地库的状态

$ git status

On branch master

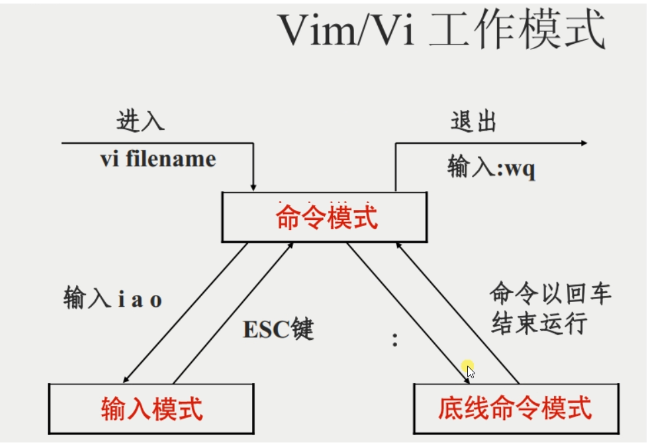
No commits yet

现在的状态是什么都没有

nothing to commit (create/copy files and use "git add" to track)

创建一个文件，在当前目录下

$ vim hello.java



不进入insert模式下输入 yy是复制 p是粘贴

查看文件中的内容

$ cat hello.java

二次查看本地库状态

$ git status

On branch master

No commits yet

Untracked files:

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

hello.java

检测到了未追踪的文件

nothing added to commit but untracked files present (use "git add" to track)

添加到暂存区

$ git add hello.java

warning: LF will be replaced by CRLF in hello.java.

The file will have its original line endings in your working directory

第三次查看本地库状态

$ git status

On branch master

No commits yet

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

检测到暂存区存在了一个新文件

new file: hello.java

第一次提交

$ git commit -m "first commit" hello.java

warning: LF will be replaced by CRLF in hello.java.

The file will have its original line endings in your working directory

[master (root-commit) bee2ceb] first commit

1 file changed, 31 insertions(+)

create mode 100644 hello.java

再次查看本地库状态

$ git status

On branch master

没有东西需要提交

nothing to commit, working tree clean

重新编辑文件

$ vim hello.java

查看状态

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: hello.java

发现工作区的文件被修改

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

将文件添加到暂存区

$ git add hello.java

warning: LF will be replaced by CRLF in hello.java.

The file will have its original line endings in your working directory

查看状态

$ git status

On branch master

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: hello.java

提交

$ git commit -m "second commnit" hello.java

warning: LF will be replaced by CRLF in hello.java.

The file will have its original line endings in your working directory

[master d41862e] second commnit

1 file changed, 1 insertion(+), 5 deletions(-)

查看状态

$ git status

On branch master

没有在暂存区发现要提交或者要修改的文件

nothing to commit, working tree clean

查看版本信息，发现提交了两次，下面两个黄色的称为版本信息（部分）

HEAD ->指向谁，说明版本库中用的是谁的节点，下面这个说明版本库中用的是第二次提交的节点

$ git reflog

d41862e (HEAD -> master) HEAD@{0}: commit: second commnit

bee2ceb HEAD@{1}: commit (initial): first commit

查看详细信息

$ git log

commit d41862eb02ecb9d92c53d4e4fd2755795eb667f0 (HEAD -> master)

Author: FuckJava <13214212383@163.com>

Date: Sat Apr 23 15:34:03 2022 +0800

second commnit

commit bee2cebba9ac5800473111ef03a1ec6553b330c3

Author: FuckJava <13214212383@163.com>

Date: Sat Apr 23 15:21:46 2022 +0800

first commit

穿梭版本 bee2ceb为穿梭过去的版本信息

$ git reset --hard bee2ceb

HEAD is now at bee2ceb first commit

通过再次查看版本信息来判断穿梭是否成功

$ git reflog

bee2ceb (HEAD -> master) HEAD@{0}: reset: moving to bee2ceb

d41862e HEAD@{1}: commit: second commnit

bee2ceb (HEAD -> master) HEAD@{2}: commit (initial): first commit

查看文件信息，回到了第一次提交的时间节点

$ cat hello.java

再次穿梭到第二次提交时间节点

$ git reset --hard d41862e

HEAD is now at d41862e second commnit

查看文件信息，回到了第二次提交的时间节点

$ cat hello.java

查看分支，下面的\*表示当前工作区所在的分支

$ git branch -v

\* master d41862e second commnit

创建名称为FuckJava-pay的分支

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git branch FuckJava-pay

查看分支，多出来一个FuckJava-pay的分支，说明成功创建分支

$ git branch -v

FuckJava-pay d41862e second commnit

\* master d41862e second commnit

切换到FuckJava-pay分支

$ git checkout FuckJava-pay

Switched to branch 'FuckJava-pay'

查看文件信息，发现在分支里也有一份第二次提交的文件

$ cat hello.java

切换回主分支，在文件中修改代码

$ git checkout master

Switched to branch 'master'

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ vim hello.java

再切换到FuckJava-pay分支

$ git checkout FuckJava-pay

Switched to branch 'FuckJava-pay'

M hello.java

在FuckJava-pay分支下查看文件的内容，发现，该分支的文件内容也被修改了

$ cat hello.java

回到主分支，并查看状态

$ git checkout master

Switched to branch 'master'

M hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: hello.java

需要再次被提交

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

提交到缓存区

$ git add hello.java

提交

$ git commit -m "分支修改第一次" hello.java

[master 5fda50f] 分支修改第一次

1 file changed, 4 insertions(+), 17 deletions(-)

查看状态，没有东西需要提交

$ git status

On branch master

nothing to commit, working tree clean

切换分支

$ git checkout FuckJava-pay

Switched to branch 'FuckJava-pay'

查看状态

$ git status

On branch FuckJava-pay

nothing to commit, working tree clean

在FuckJava-pay分支中修改文件

$ vim hello.java

查看修改后的文件内容，确定文件已经被修改

Fuck\_Java@FuckJava MINGW64 /d/GitFile (FuckJava-pay)

$ cat hello.java

修改后不提交不能切换分支

$ git checkout master

error: Your local changes to the following files would be overwritten by checkout:

hello.java

Please commit your changes or stash them before you switch branches.

Aborting

提交

$ git add hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (FuckJava-pay)

$ git commit -m "4444" hello.java

[FuckJava-pay decfb79] 4444

1 file changed, 1 insertion(+), 17 deletions(-)

切换分支

$ git checkout master

Switched to branch 'master'

在主分支上打开文件，发现没被修改

$ vim hello.java

合并分支，也就是要回到主分支上，将(FuckJava-pay)合并到主分支

$ git checkout master

Switched to branch 'master'

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git merge FuckJava-pay

Auto-merging hello.java

CONFLICT (content): Merge conflict in hello.java

Automatic merge failed; fix conflicts and then commit the result.

说明发生了冲突MERGIN就是冲突的信息

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master|MERGING)

冲突原因是两个文件的同一个位置的数据不同，Git无法决定使用哪一个，因此需要手动修改

手动解决代码冲突，需要进vim编辑器

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master|MERGING)

$ cat hello.java

$ cat hello.java

hello.java

hello.java

<<<<<<< HEAD

ma de daidfaoifandfaldfnalnfiandbcjaodfnakdnfka去你奶奶的

fuckjava

fuckjava

fuckjava

======= 等号上面表示当前分支的代码，等号下面表示合并过来的代码

hello.java

hello.java

hello.java

hello.java

hello.java

hello.java

hello.java

ma de dhaidfaoifandfaldfnalnfiandbcjaodfnakdnfka去你奶奶的

今天睡到11点半，啊啊哈哈哈

>>>>>>> FuckJava-pay

如果两个都保留的，进vim删掉下面三行即可

>>>>>>> FuckJava-pay

=======

<<<<<<< HEAD

提交到本地库 ，注意此次提交不再需要文件名也就是不写hello.java

$ git commit -m"merge commit"

fatal: cannot do a partial commit during a merge.

如果在$ git commit -m"merge commit"时加上了文件名，就需要再重新添加到缓冲区

然后再提交（提交时仍然不加文件名）

$ git add hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master|MERGING)

$ git commit -m "6666"

[master 3eb7e8e] 6666

在创建一个分支，验证文件是否修改成功

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git branch git-fix

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git status

On branch master

nothing to commit, working tree clean

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git branch -v

FuckJava-pay decfb79 4444

git-fix 3eb7e8e 6666

\* master 3eb7e8e 6666

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git checkout git-fix

Switched to branch 'git-fix'

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ cat hello.java

hello.java

hello.java

ma de daidfaoifandfaldfnalnfiandbcjaodfnakdnfka去你奶奶的

fuckjava

fuckjava

fuckjava

hello.java

hello.java

hello.java

hello.java

hello.java

ma de dhaidfaoifandfaldfnalnfiandbcjaodfnakdnfka去你奶奶的

今天睡到11点半，啊啊哈哈哈

在新分支中修改文件内容，并添加到缓冲区和提交到本地库

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ vim hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ vim hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ git add hello.java

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ git commit -m"git-fix commit" hello.java

[git-fix 1f667c2] git-fix commit

1 file changed, 3 insertions(+), 1 deletion(-)

切换分支合并代码

Fuck\_Java@FuckJava MINGW64 /d/GitFile (git-fix)

$ git checkout master

Switched to branch 'master'

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$ git merge git-fix

Updating 3eb7e8e..1f667c2

Fast-forward

hello.java | 4 +++-

1 file changed, 3 insertions(+), 1 deletion(-)

Fuck\_Java@FuckJava MINGW64 /d/GitFile (master)

$

Git团队协作开发

查看远程仓库的别名

$ git remote -v

远程仓库亲起别名 fuckjava是别名，后面跟的是远程仓库的地址

Fuck\_Java@FuckJava MINGW64 /d/Git/GitFile (mast

$ git remote add fuckjava https://github.com/no1Bossli/first-git.git

查看远程仓库的别名

Fuck\_Java@FuckJava MINGW64 /d/Git/GitFile (master)

$ git remote -v

fuckjava https://github.com/no1Bossli/first-git.git (fetch)

fuckjava https://github.com/no1Bossli/first-git.git (push)

因为远程仓库啥都没有，所以需要从本地仓库推送到远程仓库一份项目

查看(master) 下有啥东西

Fuck\_Java@FuckJava MINGW64 /d/Git/GitFile (master)

$ ll

total 1

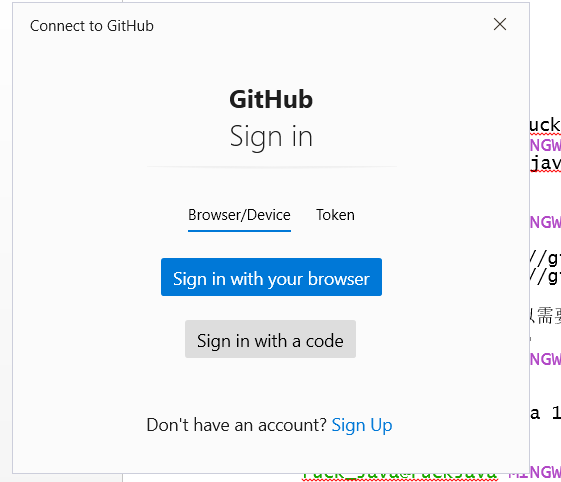
-rw-r--r-- 1 Fuck\_Java 197121 320 Apr 23 17:40 hello.java

向远程仓库推送 git push 加要推送的别名（fuckjava），加本地的分支（master）此处用的主分支

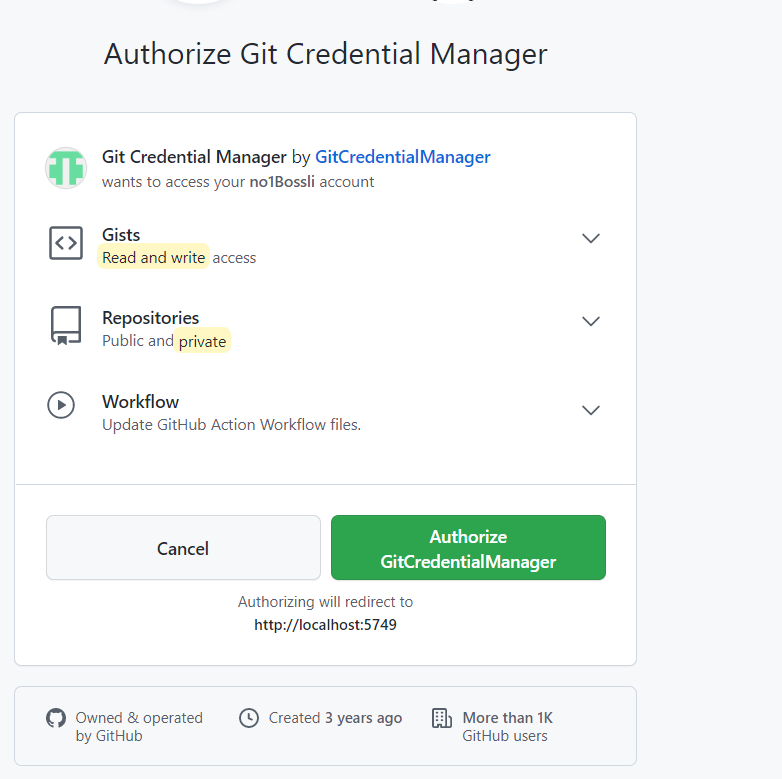
Fuck\_Java@FuckJava MINGW64 /d/Git/GitFile (master)

$ git push fuckjava master

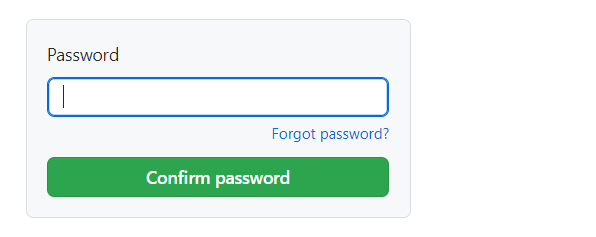
推送成功后会弹出来下面的截图，点击蓝色按钮即可，如果是403，则说明网络链接有问题，如果是10054则需要去控制面板中——凭证管理器——windows凭据——编辑或者删除，即可，删除当前授信用户信息（此种情况通常不是出现在第一次推送的情况，是因为当前用户和github上的登录用户不统一所导致的）

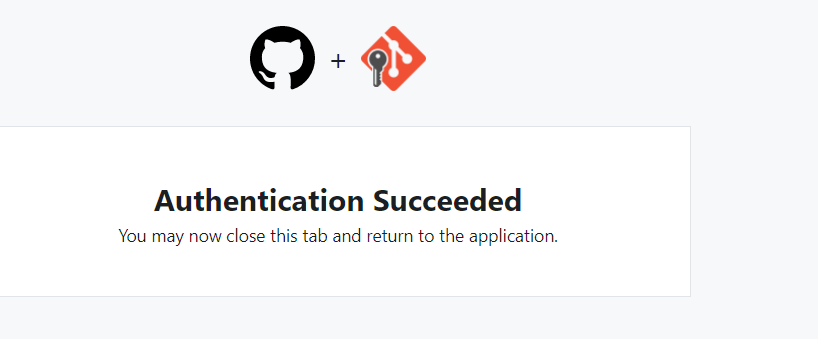
点击下面的绿色按钮进行授权



输入密码



出现下面界面表示成功



推送成功（虽然显示不安全）

$ git push fuckjava1 master

warning: ----------------- SECURITY WARNING ----------------

warning: | TLS certificate verification has been disabled! |

warning: ---------------------------------------------------

warning: HTTPS connections may not be secure. See https://aka.ms/gcmor more information.

warning: ----------------- SECURITY WARNING ----------------

warning: | TLS certificate verification has been disabled! |

warning: ---------------------------------------------------

warning: HTTPS connections may not be secure. See https://aka.ms/gcmor more information.

Enumerating objects: 21, done.

Counting objects: 100% (21/21), done.

Delta compression using up to 6 threads

Compressing objects: 100% (14/14), done.

Writing objects: 100% (21/21), 1.66 KiB | 1.66 MiB/s, done.

Total 21 (delta 7), reused 0 (delta 0), pack-reused 0

remote: Resolving deltas: 100% (7/7), done.

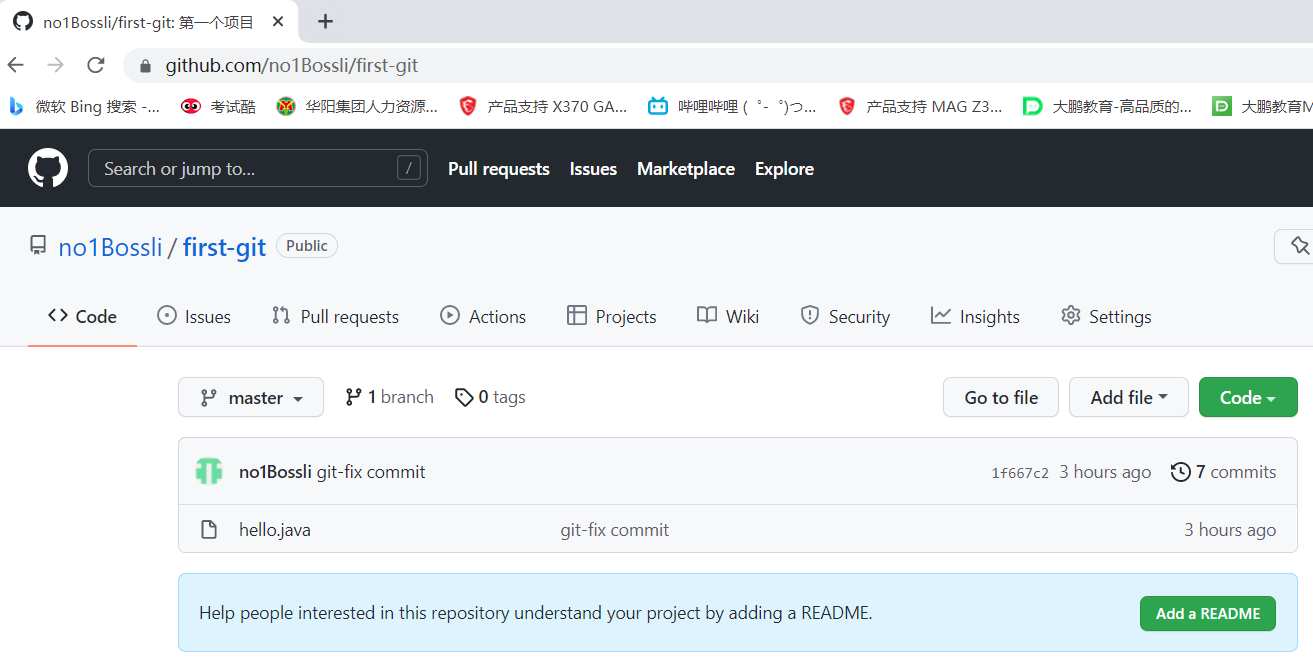
To https://github.com/no1Bossli/first-git.git

\* [new branch] master -> master

Fuck\_Java@FuckJava MINGW64 /d/Git/GitFile (master)

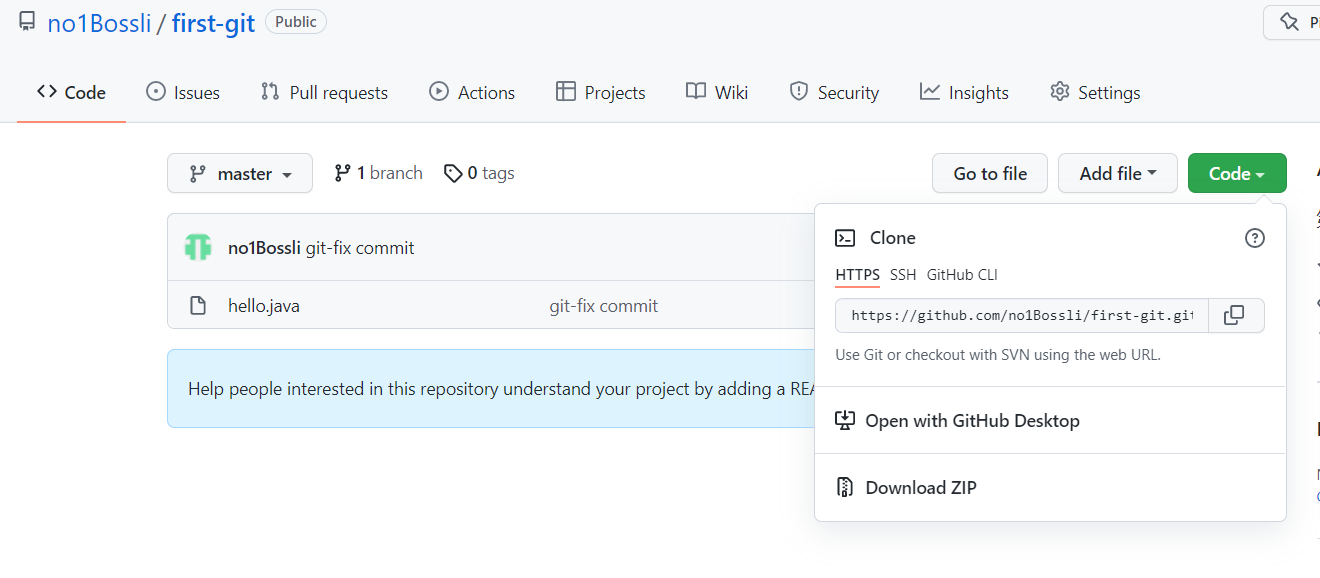
$

去Github上查看，如下图表示push成功



切换用户去Github克隆一份到本地

先去复制一下地址



然后去另一个用户的 gitBash里面粘贴地址，

Fuck\_Java@FuckJava MINGW64 /d/Git/GitHub

$ git clone https://github.com/no1Bossli/first-git.git

Cloning into 'first-git'...

remote: Enumerating objects: 21, done.

remote: Counting objects: 100% (21/21), done.

remote: Compressing objects: 100% (7/7), done.

remote: Total 21 (delta 7), reused 21 (delta 7), pack-reused 0

Receiving objects: 100% (21/21), done.

Resolving deltas: 100% (7/7), done.

上面表示克隆成功，同时在该路径下也有了新的文件夹 frist-git



进入到first-git文件夹下，查看别名，下面origin是默认的别名

Fuck\_Java@FuckJava MINGW64 /d/Git/GitHub/first-git (master)

$ git remote -v

origin https://github.com/no1Bossli/first-git.git (fetch)

origin https://github.com/no1Bossli/first-git.git (push)

也就是克隆下来做了三件事，1拉取代码，2初始化，3获取别名

去远程仓库修改一下文件

再去从本地仓库去拉取远程仓库的一份文件（此时两份是不一样的）

$ git pull origin master

fatal: unable to access 'https://github.com/no1Bossli/first-git.git/': OpenSSL SSL\_read: Connection was reset, errno 10054

报错了，原因是远程仓库没有授权

在本地仓库进行修改

Fuck\_Java@FuckJava MINGW64 /d/Git/GitHub/first-git (master)

$ vim hello.java

放入本地暂存区

$ git commit -m"bendicommit" hello.java

[master 6b5b207] bendicommit

1 file changed, 2 insertions(+), 1 deletion(-)

推到远程仓库

Fuck\_Java@FuckJava MINGW64 /d/Git/GitHub/first-git (master)

$ git push origin master

跳出登录界面

点击登录即可