**章鱼猫：GitHub的吉祥物（Octocat）**

2018-05-21 22:35

章鱼猫（Octocat）是GitHub的吉祥物，形象是一个下半身为章鱼的爪子，上本身为猫头的卡通形象。使用程序员社交网站GitHub的各位同学对它一定不会陌生。

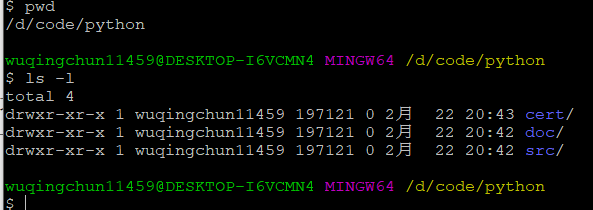


注册GitHub帐号

创建仓库 – repository

上传本地项目到 github

1. 创建一个本地项目



1. 建立本地仓库

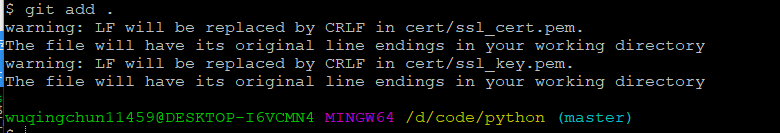
创建一个空的Git仓库

执行指令git init

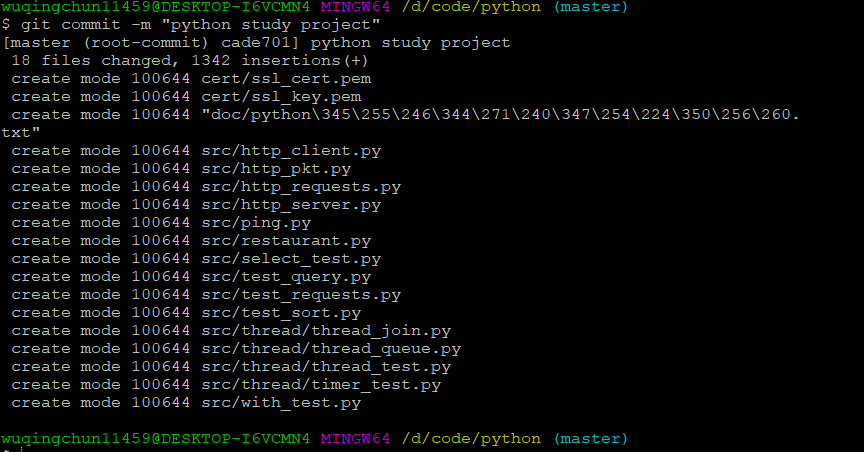
This command creates an empty Git repository - basically a .git directory with subdirectories for objects, refs/heads, refs/tags, and template files. An initial HEAD file that references the HEAD of the master branch is also created.

初始化成功后你会发现项目里多了一个隐藏文件夹.git  
这个目录是Git用来跟踪管理版本库的，没事千万不要手动修改这个目录里面的文件，不然改乱了，就把Git仓库给破坏了。

将所有文件添加到仓库  
执行指令：git add .



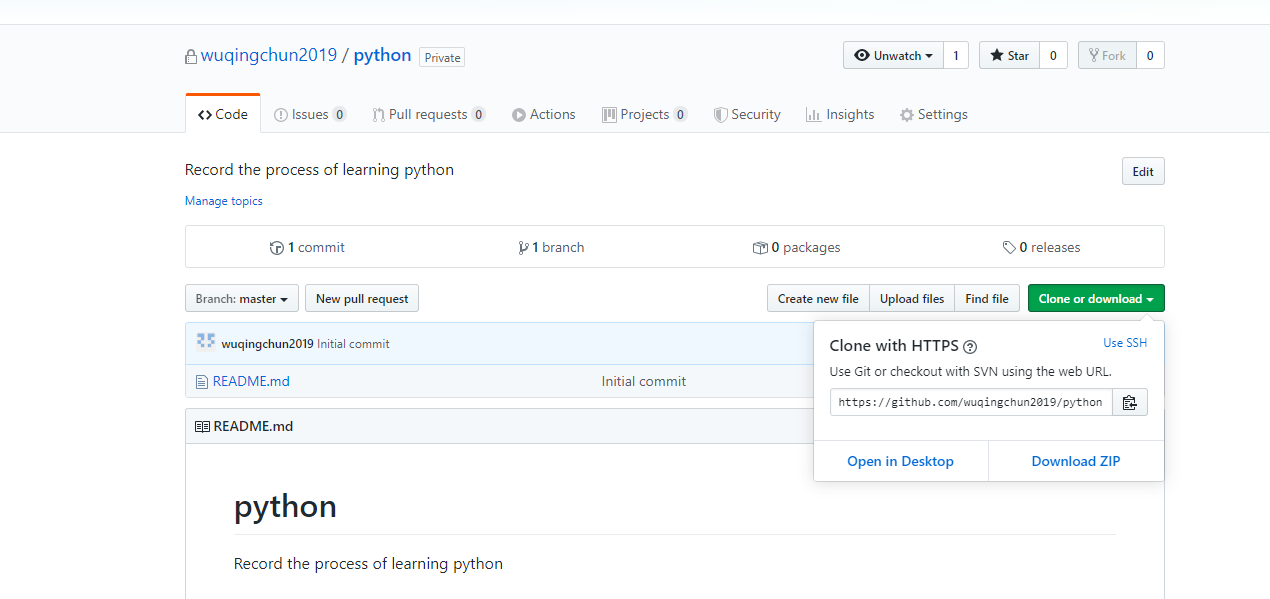
然后把文件提交到仓库，双引号内是提交注释  
执行指令：git commit -m "提交文件"



通过以上步骤，本地git仓库就已经建立好了

关联 github 仓库

到github仓库复制仓库路径

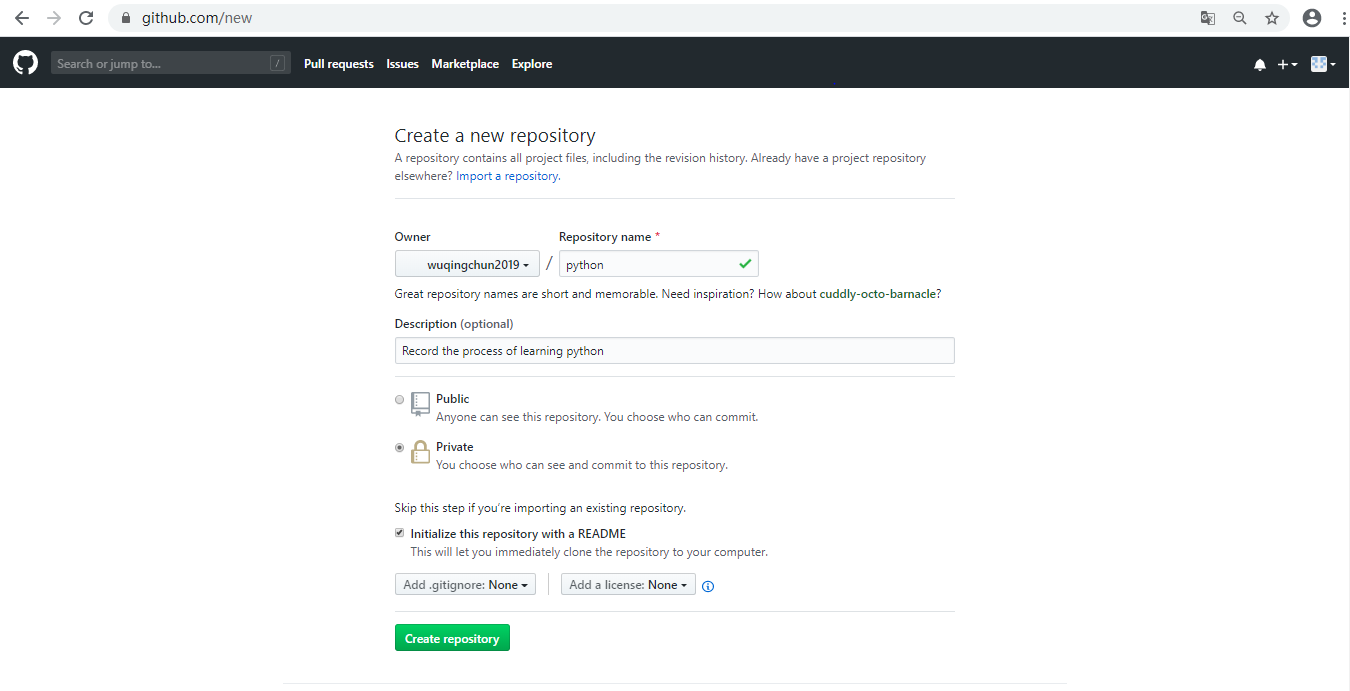


然后执行命令

然后执行指令：git remote add origin https://github.com/wuqingchun2019/python.git



在github上创建仓库

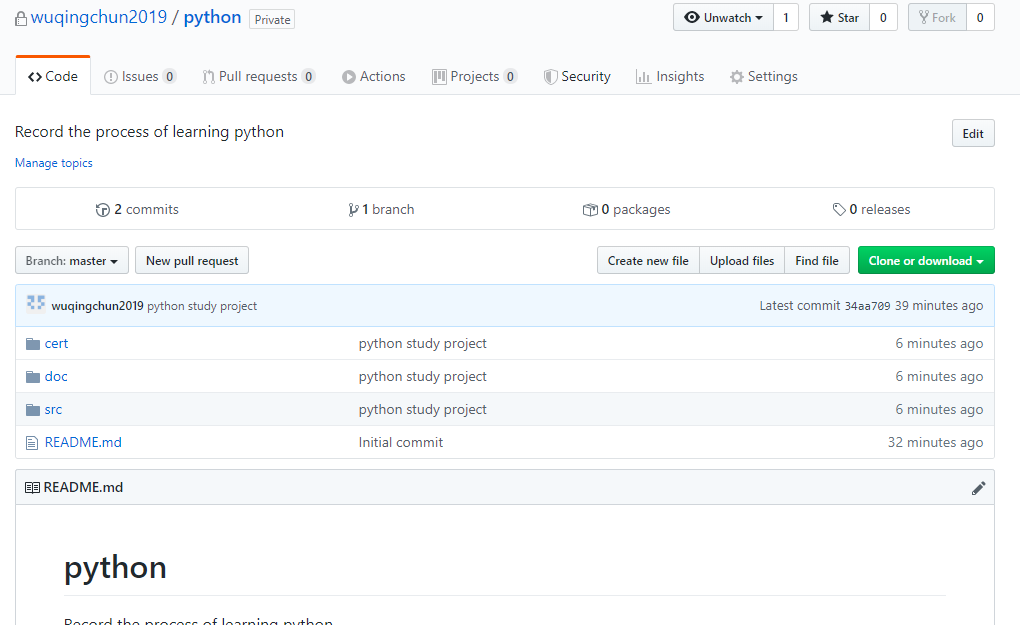


上传本地代码

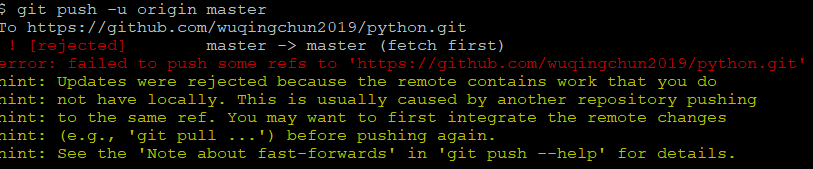
执行指令：git push -u origin master

敲一个：yes， 然后回车

注意咯：git是不能管理空的文件夹的，文件夹里必须有文件才能add



错误



出现错误的主要原因是github中的README.md文件不在本地代码目录中,

命令行中输入:

git pull --rebase origin master

$ ls -la

total 21

drwxr-xr-x 1 wuqingchun11459 197121 0 2月 22 21:29 ./

drwxr-xr-x 1 wuqingchun11459 197121 0 1月 12 18:52 ../

drwxr-xr-x 1 wuqingchun11459 197121 0 2月 22 21:29 .git/

drwxr-xr-x 1 wuqingchun11459 197121 0 2月 22 21:29 cert/

drwxr-xr-x 1 wuqingchun11459 197121 0 2月 22 21:29 doc/

-rw-r--r-- 1 wuqingchun11459 197121 49 2月 22 21:29 README.md

drwxr-xr-x 1 wuqingchun11459 197121 0 2月 22 21:29 src/

git push -u origin master

$ git push -u origin master

fatal: HttpRequestException encountered.

▒▒▒▒▒▒▒▒ʱ▒▒▒▒

Username for 'https://github.com': wuqingchun2019

Enumerating objects: 25, done.

Counting objects: 100% (25/25), done.

Delta compression using up to 4 threads

Compressing objects: 100% (23/23), done.

Writing objects: 100% (24/24), 15.59 KiB | 939.00 KiB/s, done.

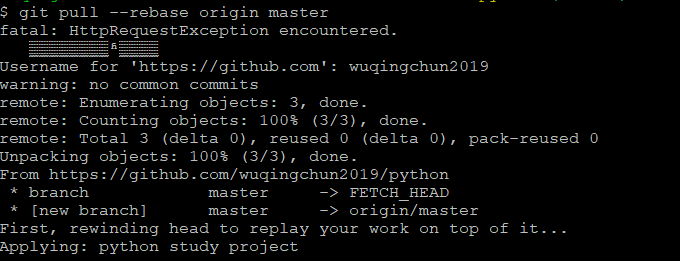
Total 24 (delta 0), reused 0 (delta 0)

To https://github.com/wuqingchun2019/python.git

532ee0d..34aa709 master -> master

Branch 'master' set up to track remote branch 'master' from 'origin'.

然后又会出现



**git push origin与git push -u origin master的区别**

$ git push origin

上面命令表示，将当前分支推送到origin主机的对应分支。

如果当前分支只有一个追踪分支，那么主机名都可以省略。

$ git push 如果当前分支与多个主机存在追踪关系，那么这个时候-u选项会指定一个默认主机，这样后面就可以不加任何参数使用git push。

$ git push -u origin master 上面命令将本地的master分支推送到origin主机，同时指定origin为默认主机，后面就可以不加任何参数使用git push了。

不带任何参数的git push，默认只推送当前分支，这叫做simple方式。此外，还有一种matching方式，会推送所有有对应的远程分支的本地分支。Git 2.0版本之前，默认采用matching方法，现在改为默认采用simple方式。

作者：51cf68e144ad  
链接：https://www.jianshu.com/p/dd864fcee643  
来源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。

修改代码并提交到远程github仓库

$ git diff HEAD

**diff --git a/src/select\_test.py b/src/select\_test.py**

**index d999a85..f2b0d53 100644**

**--- a/src/select\_test.py**

**+++ b/src/select\_test.py**

@@ -2,63 +2,14 @@

import select

import socket

-def show\_event\_debug(sock, event\_string):

- local\_addr = sock.getsockname()

- peer\_addr = sock.getpeername()

- print("Connection received an {} event. Local={}, "

- "Peer={}.".format(event\_string, local\_addr, peer\_addr))

-

-def conn\_recv(sock, wlist):

- response = sock.recv(1024)

- print("Received message:\n{}".format(response))

- wlist.append(sock)

-

-def conn\_send(sock):

- sock.send(b"Hello")

-

-def accept\_new\_conn(sock, rlist):

- accept\_sock, client\_addr = sock.accept()

- print("Succeed to accept a new connection {}.".format(client\_addr))

- rlist.append(accept\_sock)

-

-def schedule():

-

- sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM, 0)

- sock.bind(('', 12000))

- sock.listen(5)

- sock.setblocking(False)

-

- rlist = [sock]

- wlist = []

- xlist = []

-

+def event\_schedule(handler\_list):

while True:

- rlist\_ret, wlist\_ret, xlist\_ret = select.select(rlist, wlist, xlist)

- for item in rlist\_ret:

- show\_event\_debug(item, "input")

- if item == sock:

- accept\_new\_conn(sock, rlist)

- else:

- show\_event\_debug(item, "input")

- conn\_recv(item, wlist)

+ wants\_recv = [h for h in handler\_list if h.wants\_to\_receive()]

+ wants\_send = [h for h in handler\_list if h.wants\_to\_send()]

- for item in wlist\_ret:

- show\_event\_debug(item, "output")

- conn\_send(item)

-

- for item in xlist\_ret:

- show\_event\_debug(item, "error")

- rlist.remove(item)

- wlist.remove(item)

- item.close()

- print("Connection closed.")

-

-def event\_schedule(handlers):

- while True:

- wants\_recv = [h for h in handlers if handlers.wants\_to\_receive()]

- wants\_send = [h for h in handlers if handlers.wants\_to\_send()]

-

- can\_recv, can\_send, except\_happen = select.select(wants\_recv, wants\_send, handlers)

+ print("1 select.select")

+ can\_recv, can\_send, except\_happen = select.select(wants\_recv, wants\_send, handler\_list)

+ print("2 select.select")

for h in can\_recv:

h.handle\_recv()

@@ -68,11 +19,100 @@ def event\_schedule(handlers):

for h in except\_happen:

h.handle\_except()

-class TCPServer():

+class EventHandler():

+ '''事件驱动IO处理类框架定义'''

+

+ def fileno(self):

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git status

On branch master

Your branch is up to date with 'origin/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: src/select\_test.py

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

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git add src/select\_test.py

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git status

On branch master

Your branch is up to date with 'origin/master'.

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: src/select\_test.py

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git commit -m "使用新的代码框架编写事件驱动IO代码"

[master 6483116] 使用新的代码框架编写事件驱动IO代码

1 file changed, 118 insertions(+), 78 deletions(-)

rewrite src/select\_test.py (81%)

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git push

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 4 threads

Compressing objects: 100% (4/4), done.

Writing objects: 100% (4/4), 1.35 KiB | 690.00 KiB/s, done.

Total 4 (delta 2), reused 0 (delta 0)

remote: Resolving deltas: 100% (2/2), completed with 2 local objects.

To https://github.com/wuqingchun2019/python.git

34aa709..6483116 master -> master

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git status

On branch master

Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

2020-02-26

1. 切换到/d/code/python目录
2. git status
3. 修改文件python学习笔记
4. 添加process目录和process\_test.py文件
5. git status

$ git status

On branch master

Your branch is up to date with 'origin/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: "doc/python\345\255\246\344\271\240\347\254\224\350\256\260.txt"

Untracked files:

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

src/process/

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

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git add .

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git commit -m "python多进程编程"

[master e8f09a0] python多进程编程

2 files changed, 54 insertions(+)

create mode 100644 src/process/process\_test.py

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

$ git push

Enumerating objects: 11, done.

Counting objects: 100% (11/11), done.

Delta compression using up to 4 threads

Compressing objects: 100% (6/6), done.

Writing objects: 100% (7/7), 1.50 KiB | 192.00 KiB/s, done.

Total 7 (delta 1), reused 0 (delta 0)

remote: Resolving deltas: 100% (1/1), completed with 1 local object.

To https://github.com/wuqingchun2019/python.git

6483116..e8f09a0 master -> master

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/code/python (master)

复制github远程仓库地址

git clone

复制仓库内容作为本地副本

$ git clone https://github.com/wuqingchun2019/python.git

Cloning into 'python'...

fatal: HttpRequestException encountered.

▒▒▒▒▒▒▒▒ʱ▒▒▒▒

Username for 'https://github.com': wuqingchun2019

remote: Enumerating objects: 31, done.

remote: Counting objects: 100% (31/31), done.

remote: Compressing objects: 100% (26/26), done.

remote: Total 31 (delta 2), reused 28 (delta 2), pack-reused 0

Unpacking objects: 100% (31/31), done.

注意：如果远程有别人已经提交了，你在push之前 先pull下来！解决冲突！再add commit push 就行了！

$ git pull

Already up to date.

$ git pull

fatal: HttpRequestException encountered.

▒▒▒▒▒▒▒▒ʱ▒▒▒▒

Username for 'https://github.com': wuqingchun2019

remote: Enumerating objects: 11, done.

remote: Counting objects: 100% (11/11), done.

remote: Compressing objects: 100% (5/5), done.

remote: Total 7 (delta 1), reused 7 (delta 1), pack-reused 0

Unpacking objects: 100% (7/7), done.

From https://github.com/wuqingchun2019/python

6483116..e8f09a0 master -> origin/master

Updating 6483116..e8f09a0

Fast-forward

...55\246\344\271\240\347\254\224\350\256\260.txt" | 20 +++++++++++++

src/process/process\_test.py | 34 ++++++++++++++++++++++

2 files changed, 54 insertions(+)

create mode 100644 src/process/process\_test.py

wuqingchun11459@DESKTOP-I6VCMN4 MINGW64 /d/mypython/python (master)

要克隆一个仓库，首先要知道仓库的地址，然后使用git clone命令进行克隆，Git支持多种协议，默认的git://使用SSH协议，也可以使用https协议，其中使用原生的git://速度更快。

使用https除了速度慢以外，还有个最大的麻烦是每次推送都必须输入口令，但是在某些只开放http端口的公司内部就无法使用ssh协议而只能用https。