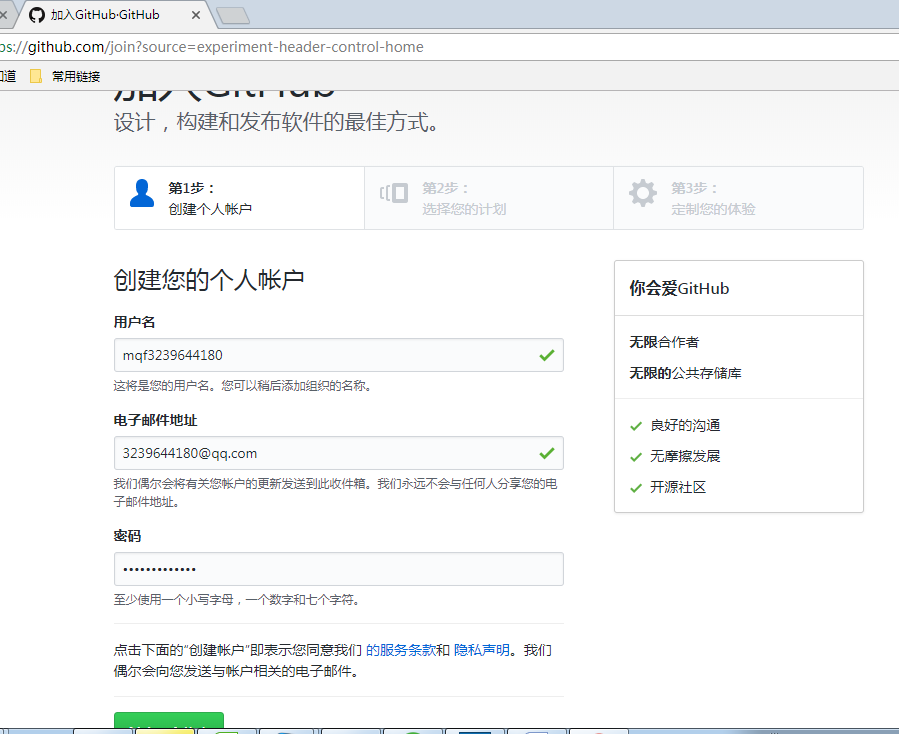
GIT的注册：



密码：3239644180mqf



或者跳过：



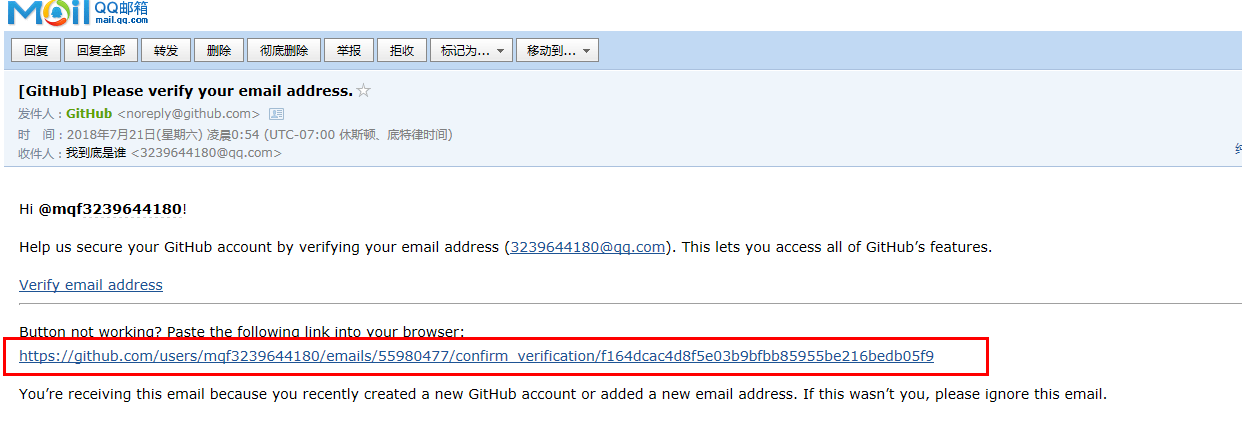
最后：



点击启动项目：

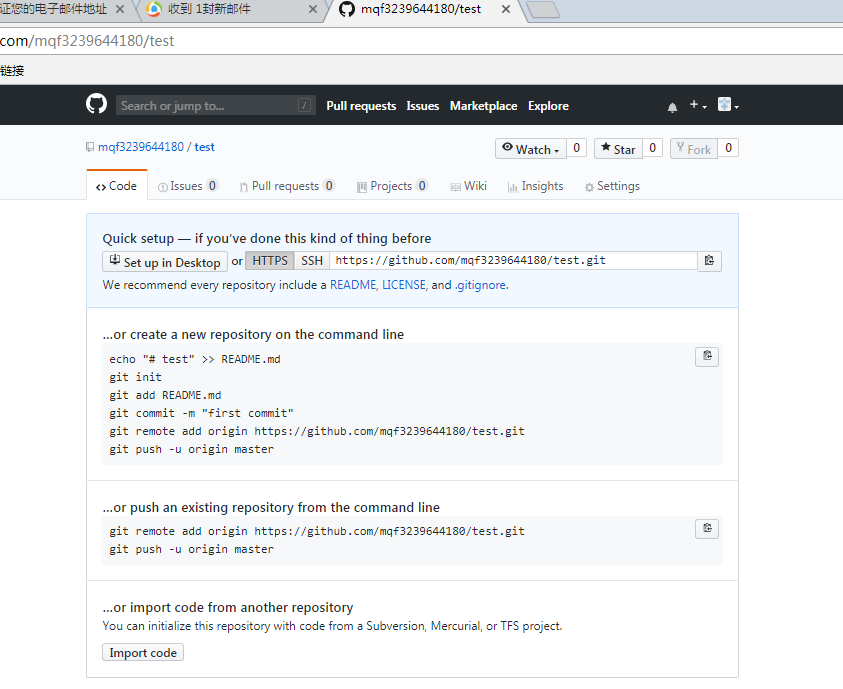


登录邮箱进行验证：





完毕



# 基本使用：

1. 常见版本控制  
   - svn   
     - 版本控制工具  
     - 中心式版本控制工具  
   - git  
     - 版本控制工具  
     - 分布式版本工具  
2. 常用Git代码托管平台  
   - GitHub  
     - 全球最大的代码托管平台  
     - 各种程序，各种语言的代码  
     - 比较干净的社区  
     - 免费，公开  
     - 收费，私有  
   - 码云  
   - 免费，私有  
   - 淘宝代码

1. 依托于GitHub使用

   - 注册一个GitHub账号，注意就是需要在邮箱中激活一下

进入GitHub官网注册

## 2. git下载：

## （1）进入网站：https://git-scm.com/downloads

（2）添加下载源：



于Debian / Ubuntu

对于Ubuntu，此PPA提供最新的稳定上游Git版本

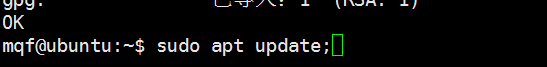
# add-apt-repository ppa:git-core/ppa # apt update; apt install git

## 安装的命令步骤：

1. sudo add-apt-repository ppa:git-core/ppa



2. sudo apt update



3.sudo apt install git



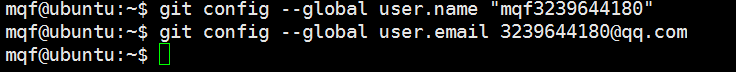
## 2.设置用户名和密码

用户信息

当安装完 Git 应该做的第一件事就是设置你的用户名称与邮件地址。 这样做很重要，因为每一个 Git 的提交都会使用这些信息，并且它会写入到你的每一次提交中，不可更改：

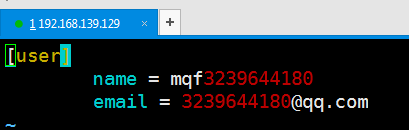
$ git config --global user.name "mqf3239644180"

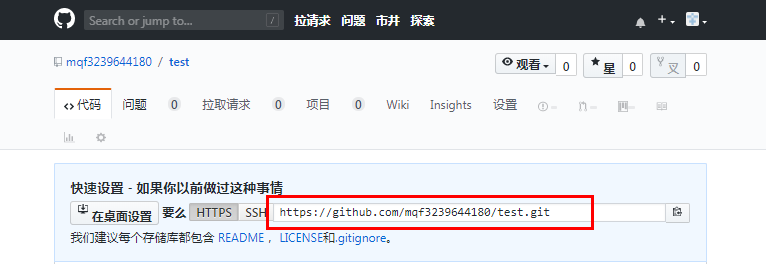
$ git config --global user.email 3239644180@qq.com

或者打开gitconfig



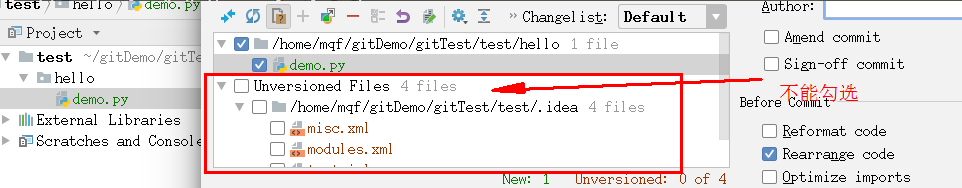
vim .gitconfig

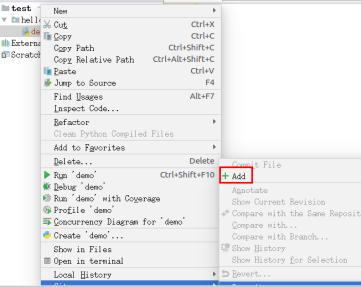
  
3. 获取Git仓库  
   - git clone xxx  
     - 将xxx中对应的git仓库进行检索下载

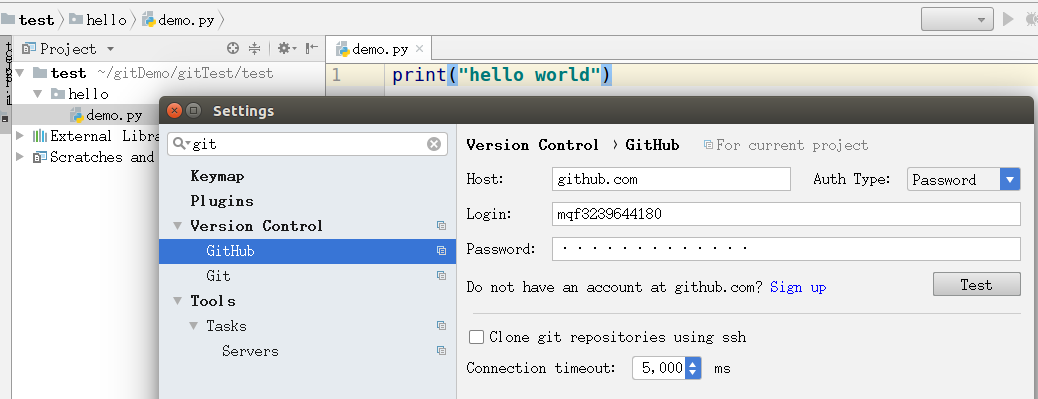


 git clone https://github.com/mqf3239644180/test.git

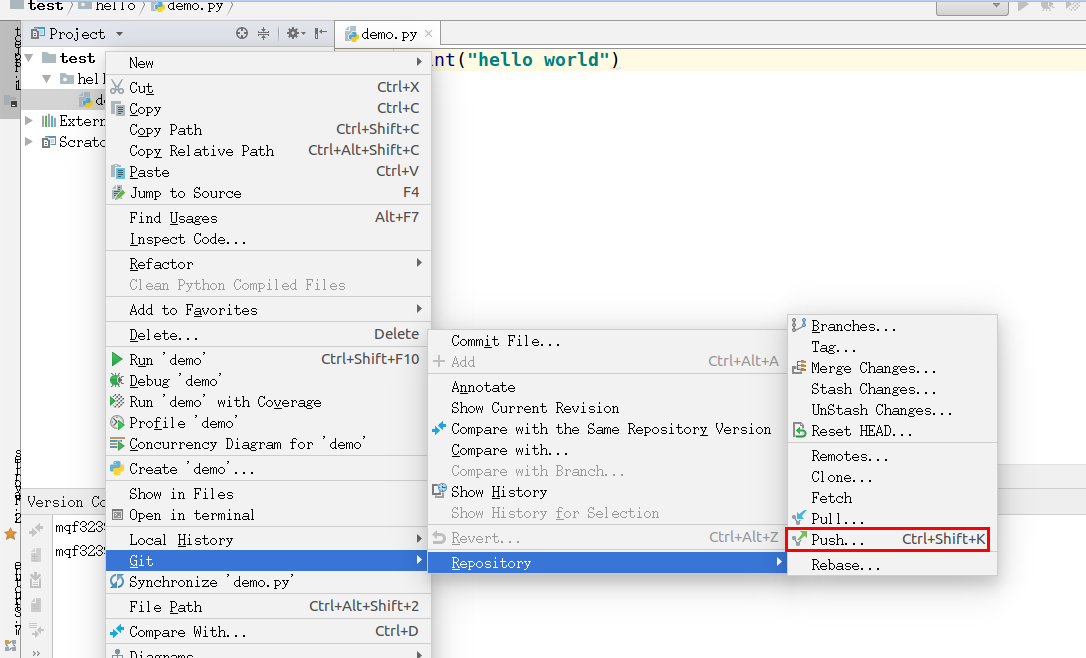
  
4. 使用PyCharm打开Git仓库  
   - 在界面多了Version Control 这个模块  
   - 在右击上也多了 Git这样一个菜单  
  
5. 写的代码基本都是需要添加版本控制中的，.idea 文件夹中的所有文件都是不需要上传的（这个PyCharm根据电脑相关信息生成的），每台电脑都是不一样的

  
6. 默认情况下，没有添加到Git中的文件是  暗红色  
7. 添加到Git中，文件会变成绿色

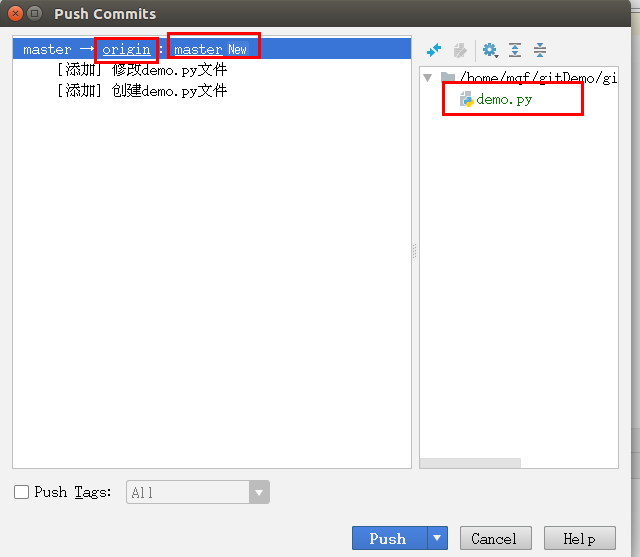
  
8. 文件commit会回归正常颜色   
9. 当文件发生修改的时候，会变成蓝色的  
### 文件提交  
1. 先add  
2. 再commit  
   - 需要检查提交  
   - 添加提交日志  
3. 将代码推送到远端

连接：  


推送远端：

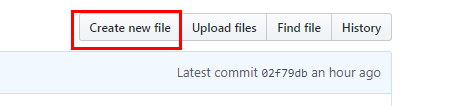


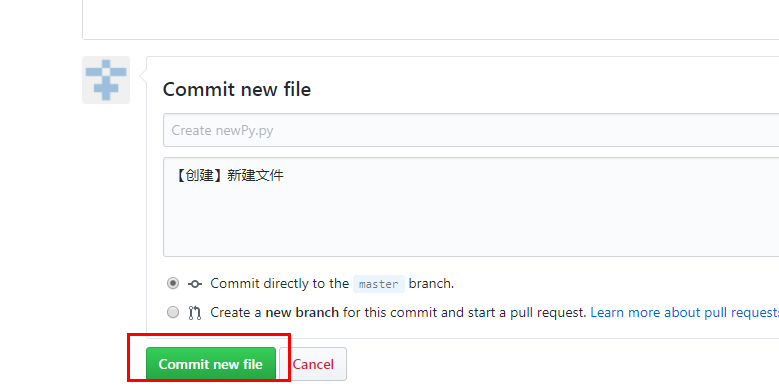
把代码从本地分支推送到远程分支：



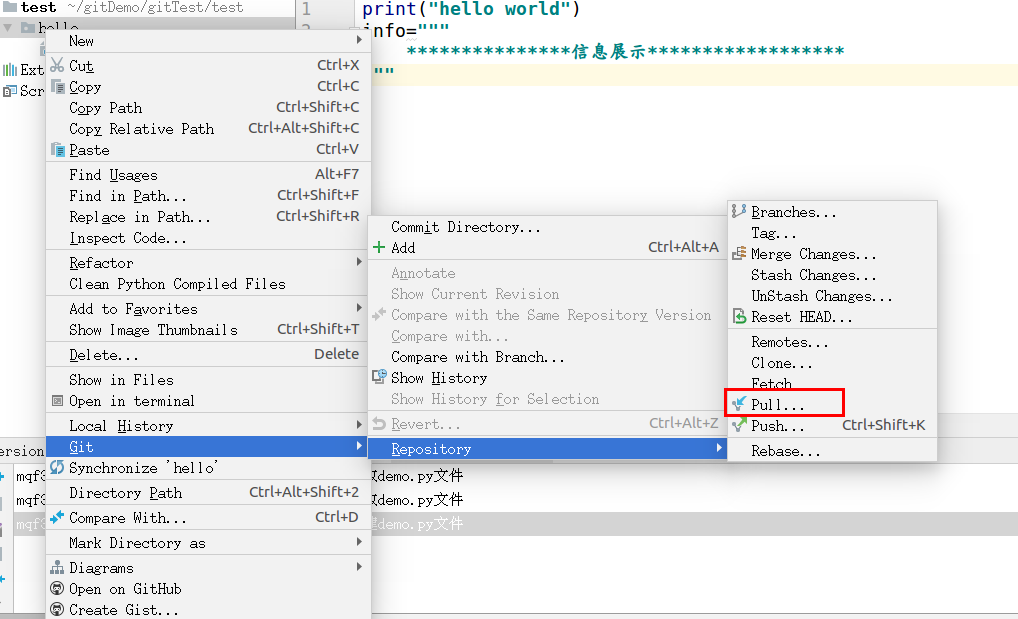
# 获取远程代码：

（1）在github网站创建一个文件：

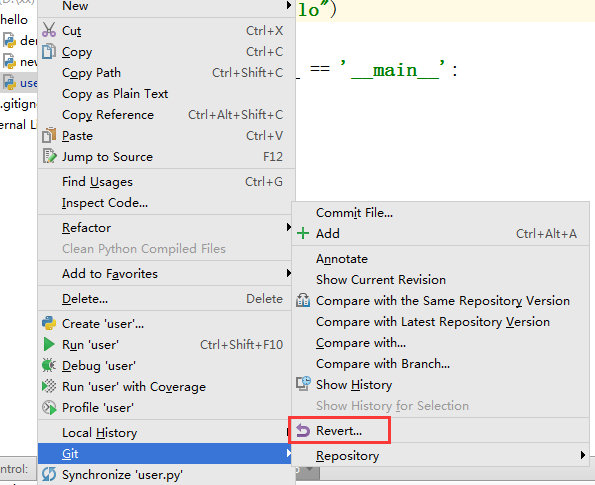




代码拉取：



代码的回退：

  
### Git指令  
  
1. git clone 代码地址  
2. git status  
   - 查看git状态  
3. git add  变更的文件名  
   - 添加文件变更追踪  
   - git add -A   
     - 添加所有变更文件  
4. git commit  
   - git commit -m "xxx"  
     - 提交到本地仓库，并且添加xxx描述  
5. git push  
   - git push origin master  
   - 默认是master主分支

如果是其他的自己定义的分支的话就是：

  
   - 推送的时候，推送到原始的主分支上  
6. git pull  
   - 更新代码，拉取代码  
7. 分支 branch  
   - git branch  BRANCH\_NAME 创建一个分支  
   - git branch -a 显示所有分支

- git branch 显示所在分支  
8. 切换分支  
   - git checkout BRANCH\_NAME  
     - 切换分支  
9. 分支中的数据是隔离开的  
10. git log  
    - 查看提交修改历史的

q键退出  
11. git merge   
    - git merge BRANCH\_NAME  
      - 合并一个分支