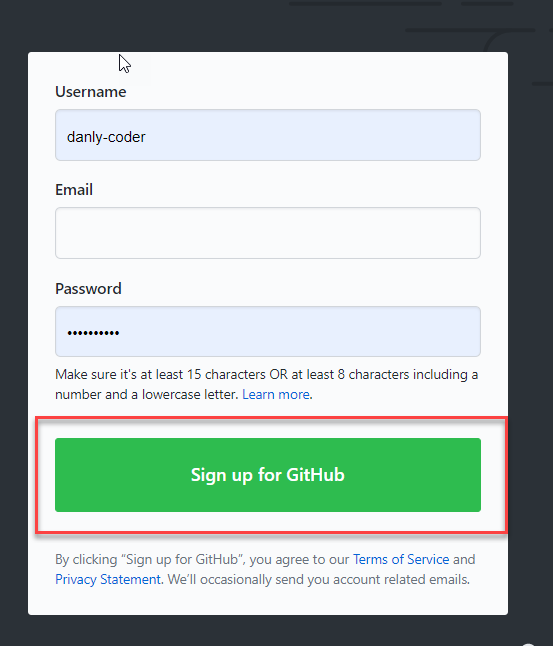
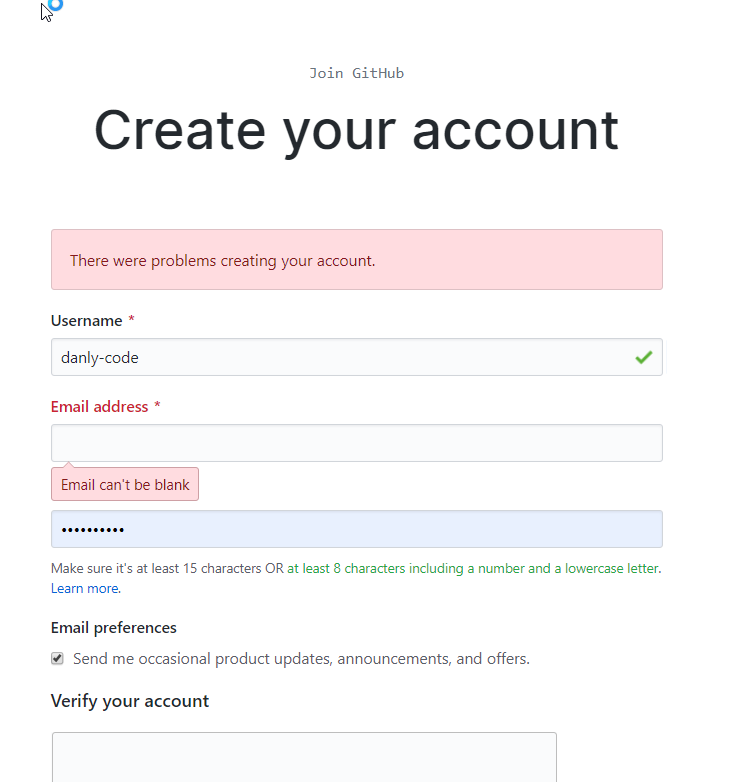
github是通过Git进行版本控制的软件源代码托管服务平台

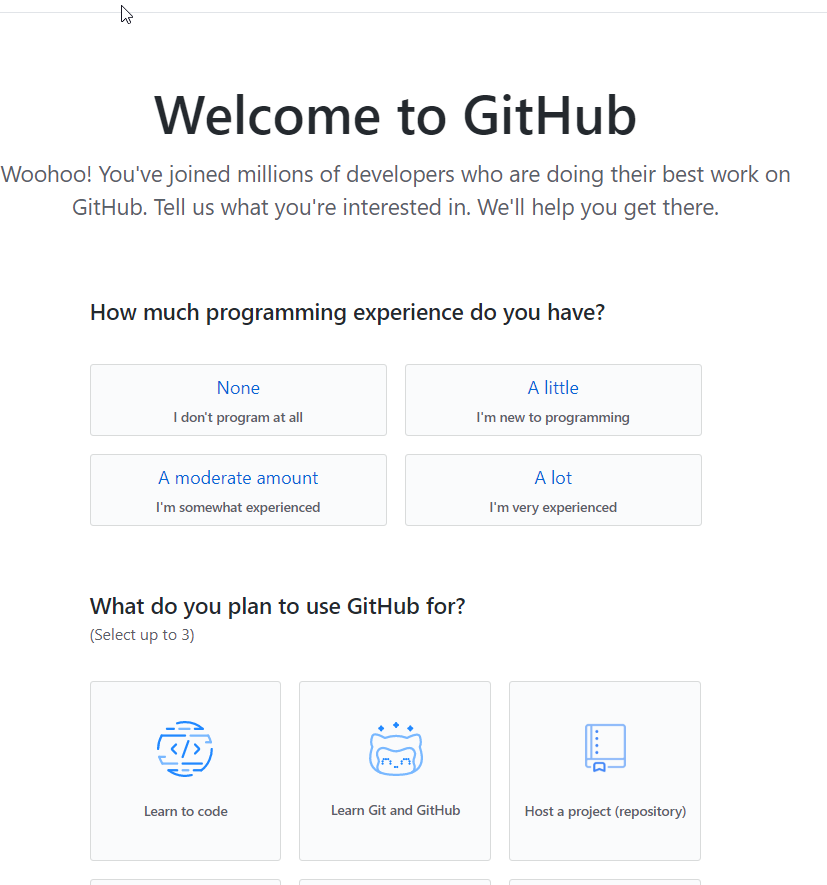
github除了Git代码仓库托管及基本的Web管理界面以外，它还提供了一些方便社会化共同软件开发的功能，即一般人口中的社群功能，包括允许用户追踪其他用户、组织、软件库的动态，对[软件代码](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=4196027&ss_c=ssc.citiao.link)的改动和bug提出评论等。

1. GitHub
2. 注册账号

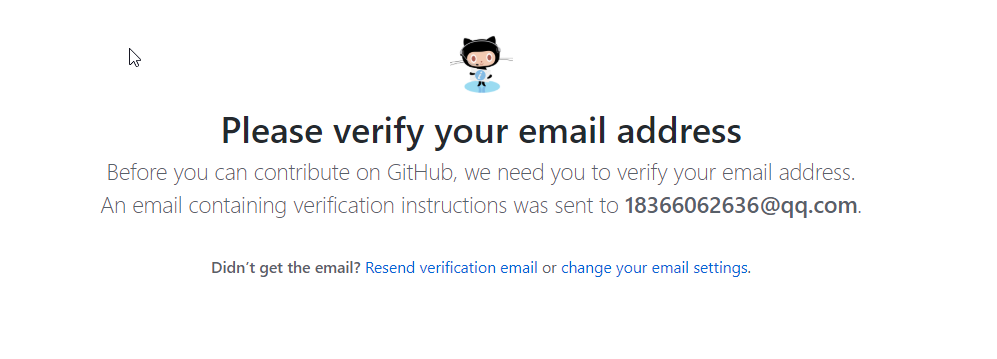




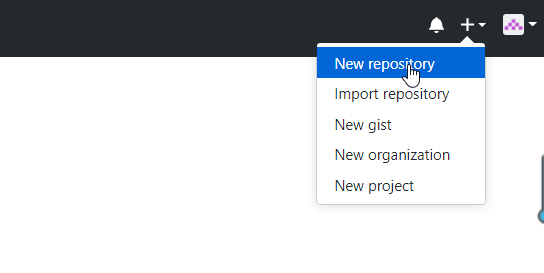
2.根据个人情况进行选择

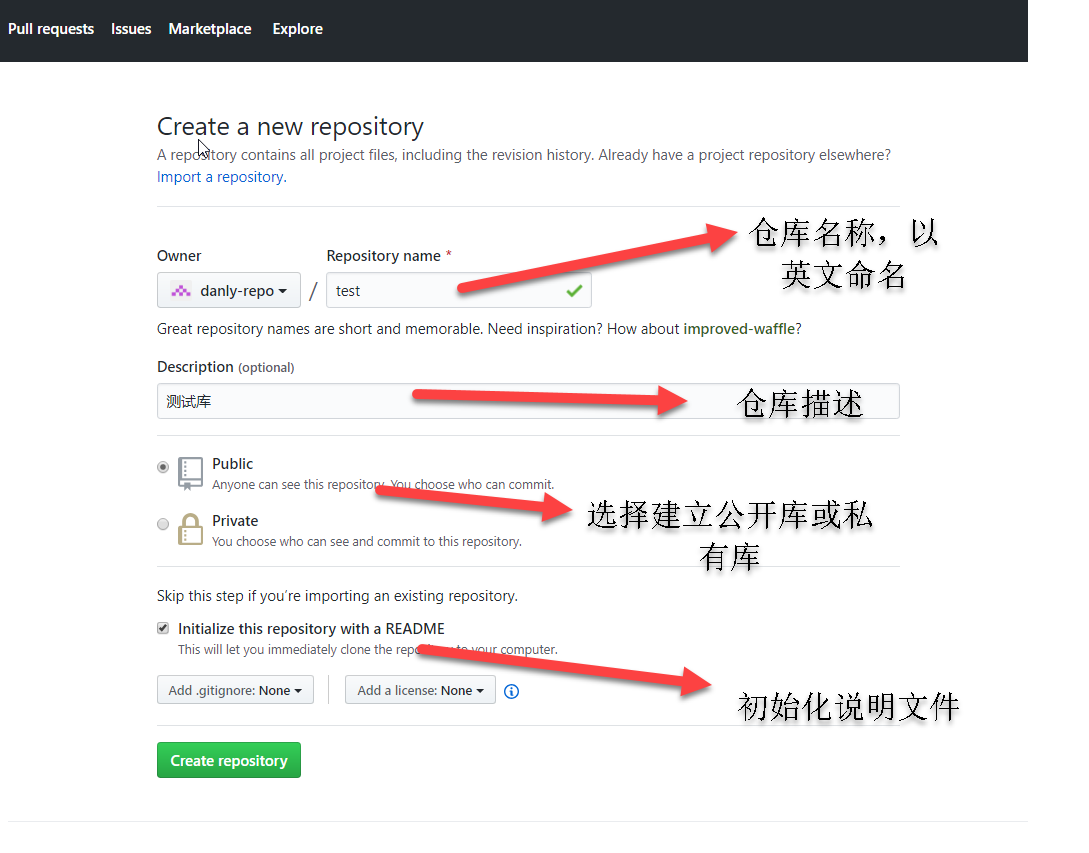


3.验证邮箱，必须验证后才能创建仓库：

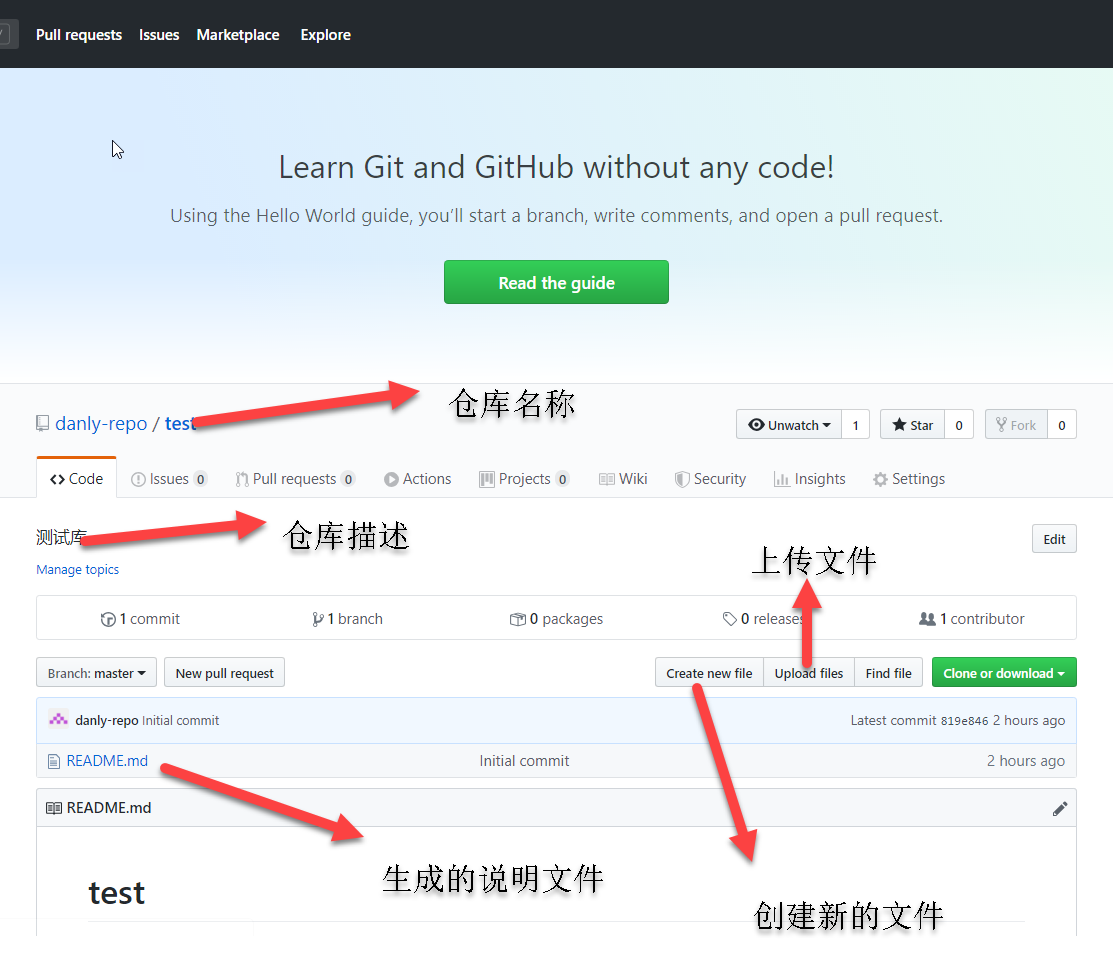


4.创建仓库



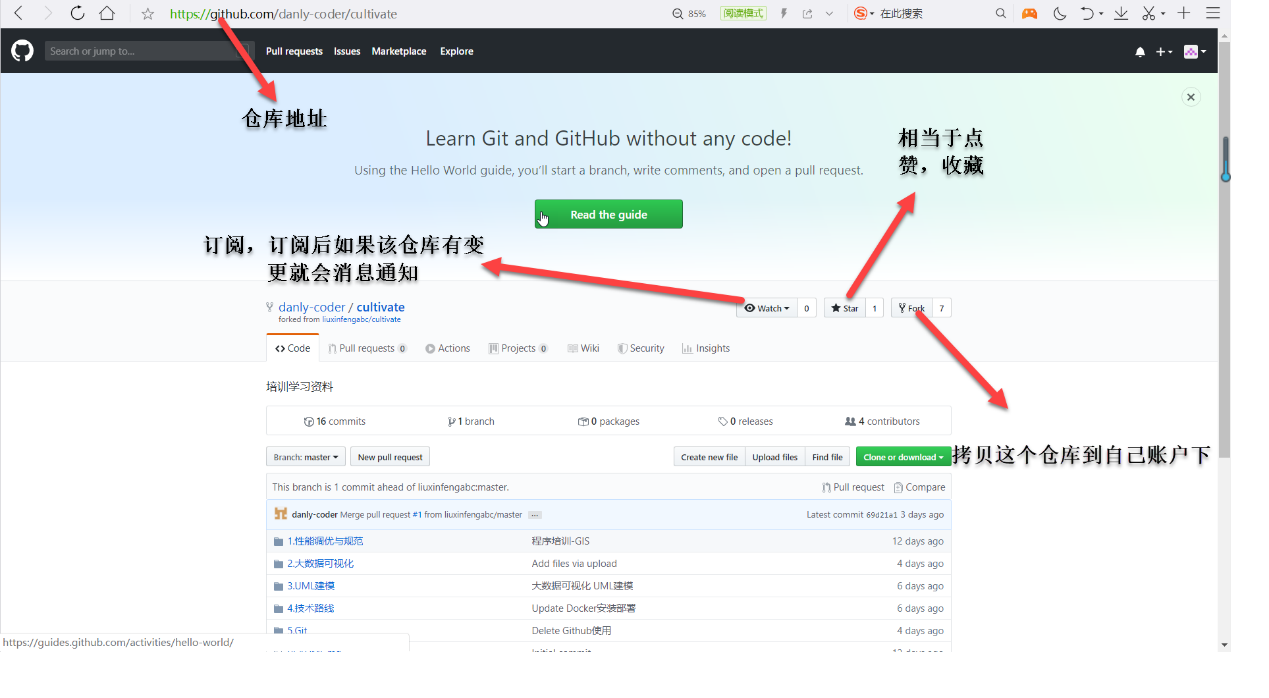


生成的仓库如下

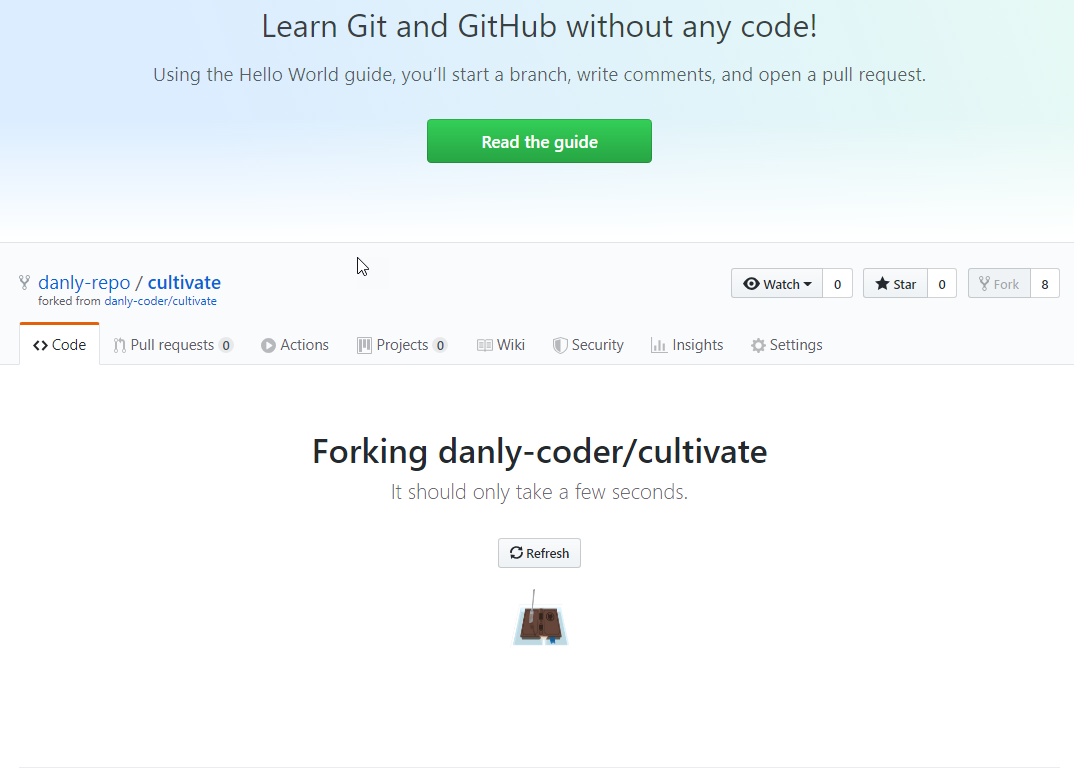


打开另外一个仓库的网址<https://github.com/danly-coder/cultivate>

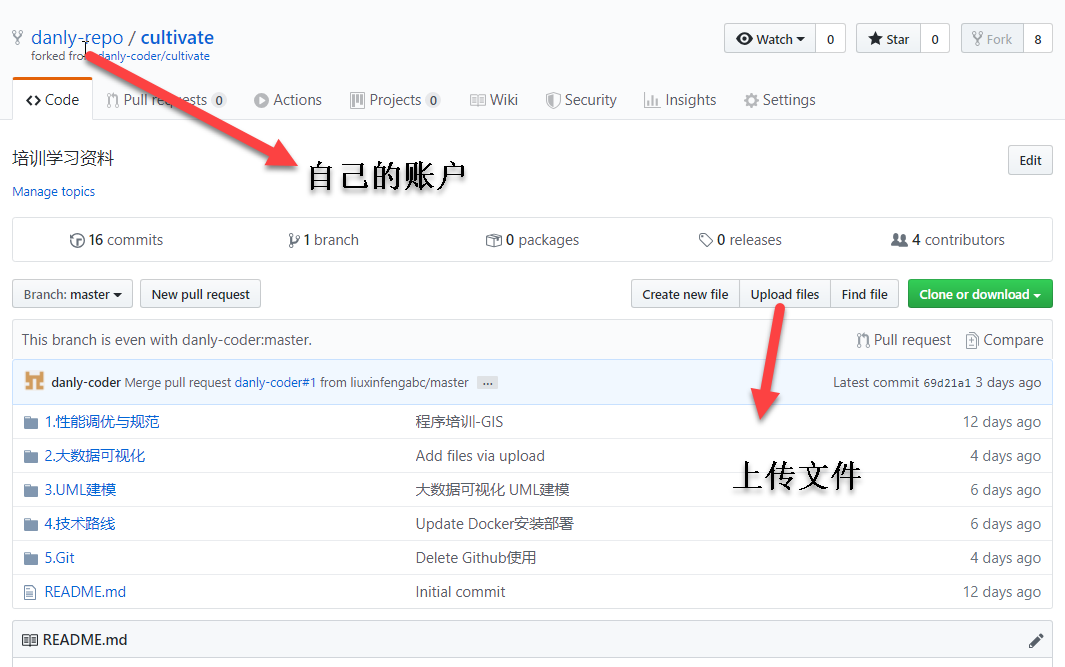
这是danly-coder用户下cultivate库地址



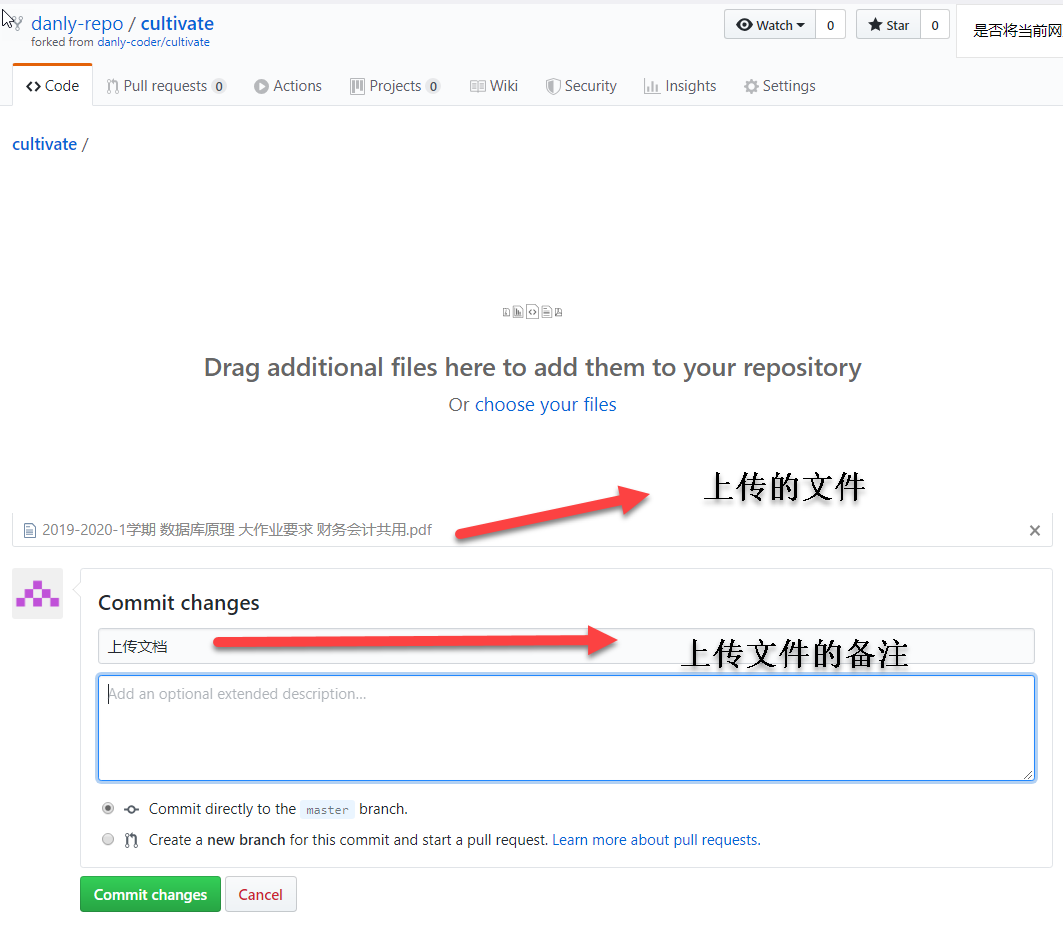
Fork该仓库



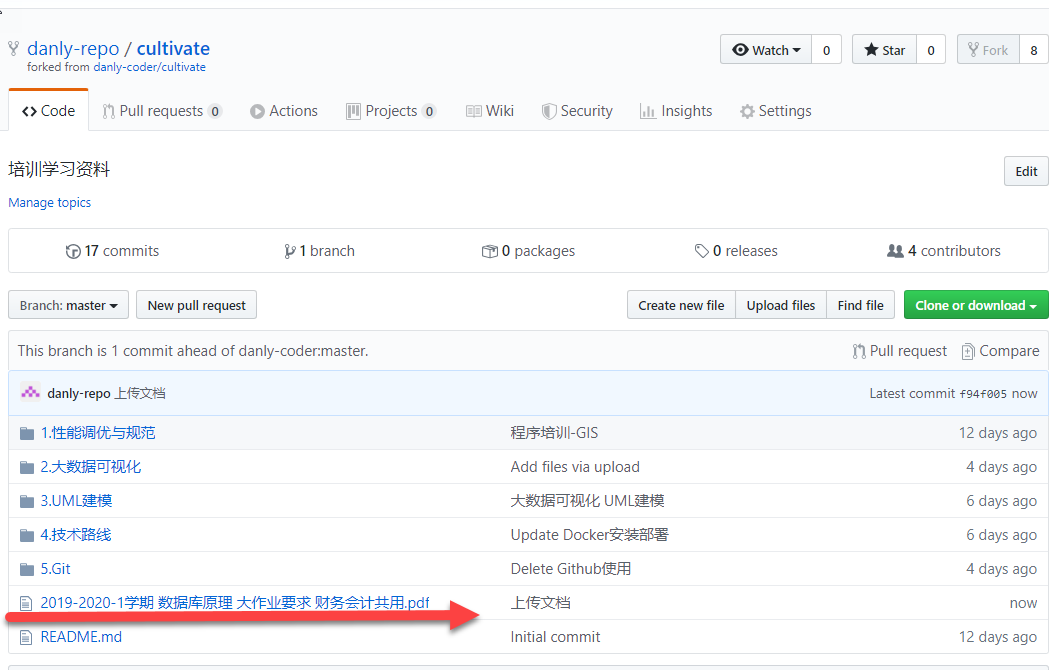
在自己的账户就是danly-repo下就可看到cultivate库



点击上传文件后进行上传，添加备注后提交改变。

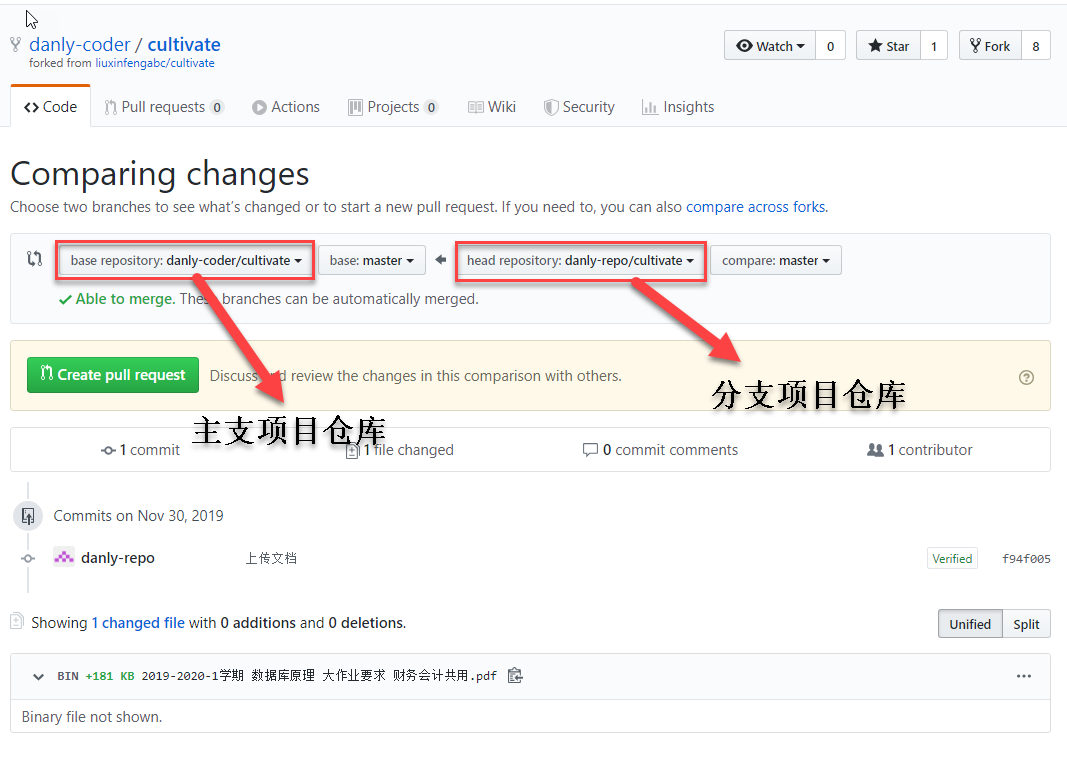


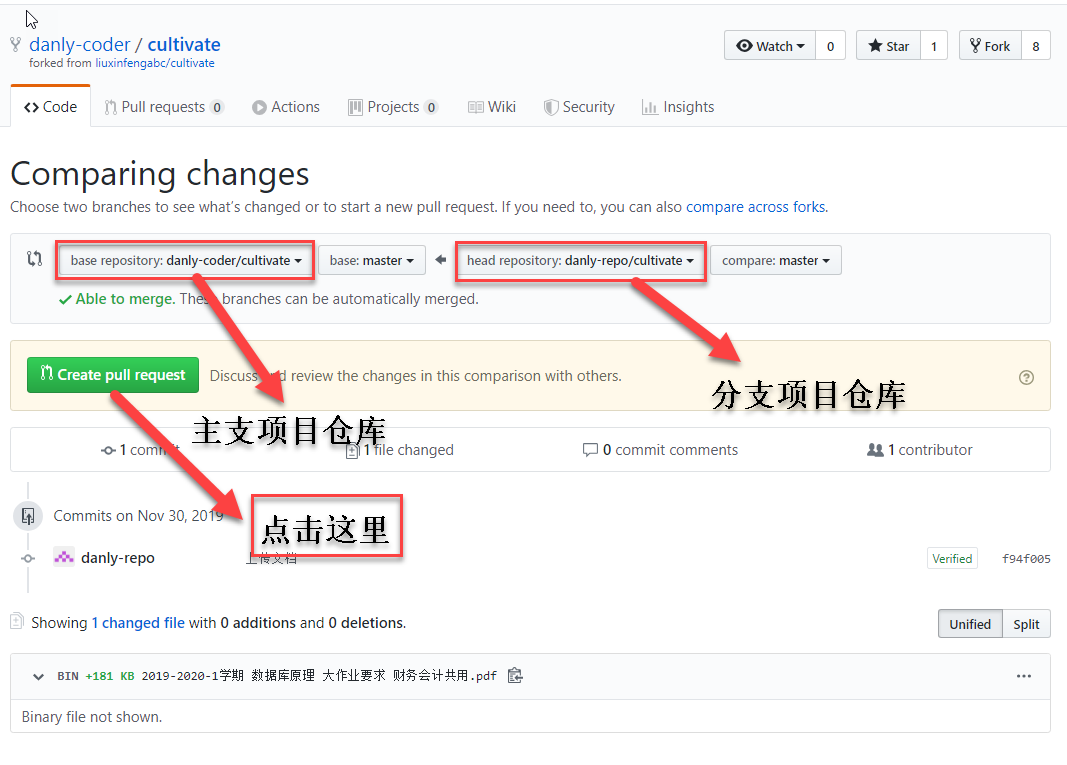
可以看到这个账号下cultivate库增加了文件



Repo账号的cultivate是从coder账户下拷贝过来，相当于coder账户的一个分支，现在分支里的内容发生了改变，主支没有改变，我们可以改变后的内容new pull request,这样就能够把修改后的文件提交给主支。



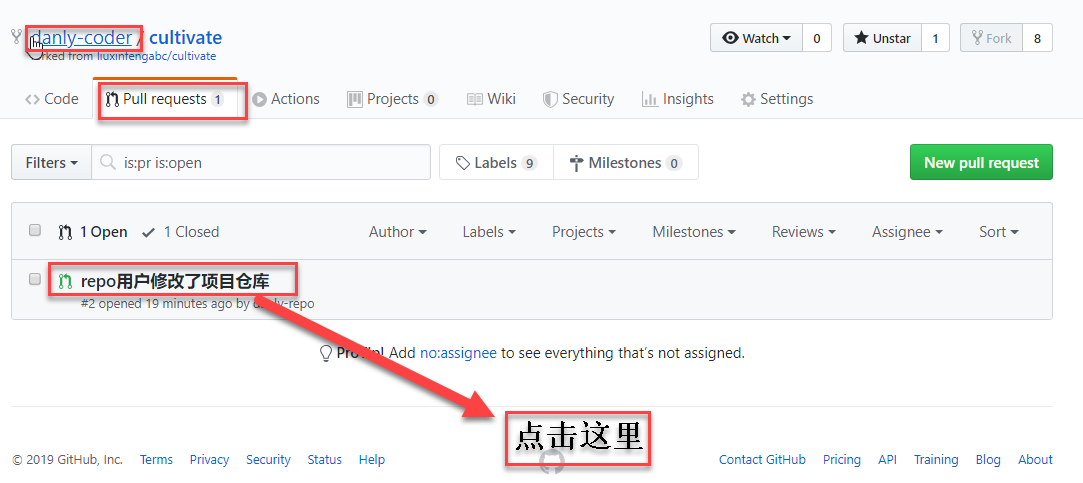


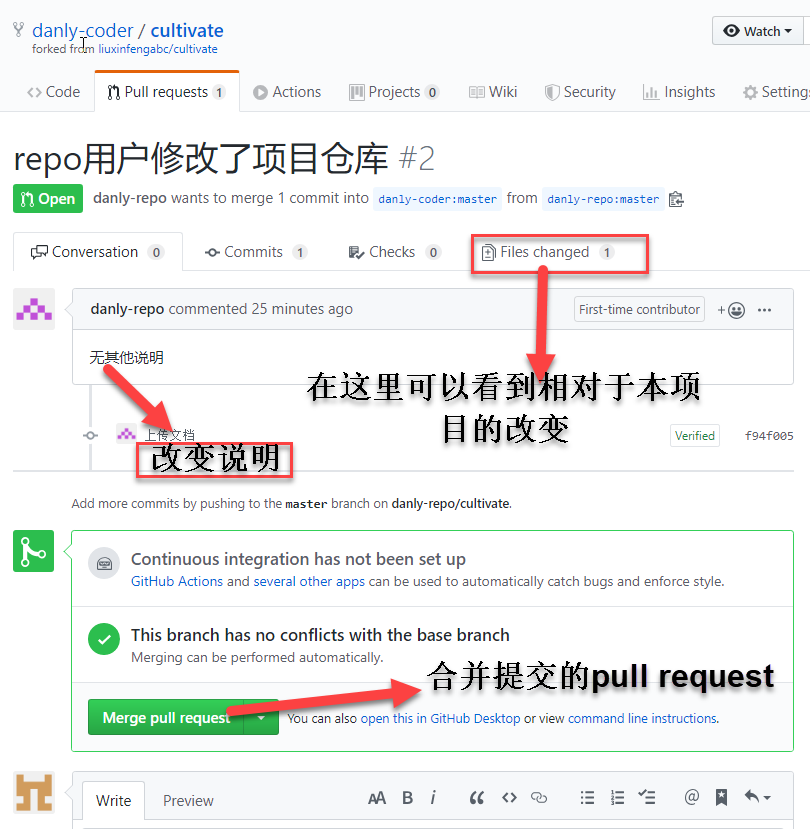




这样就完成了pull request，向主支仓库提交了修改后的项目。

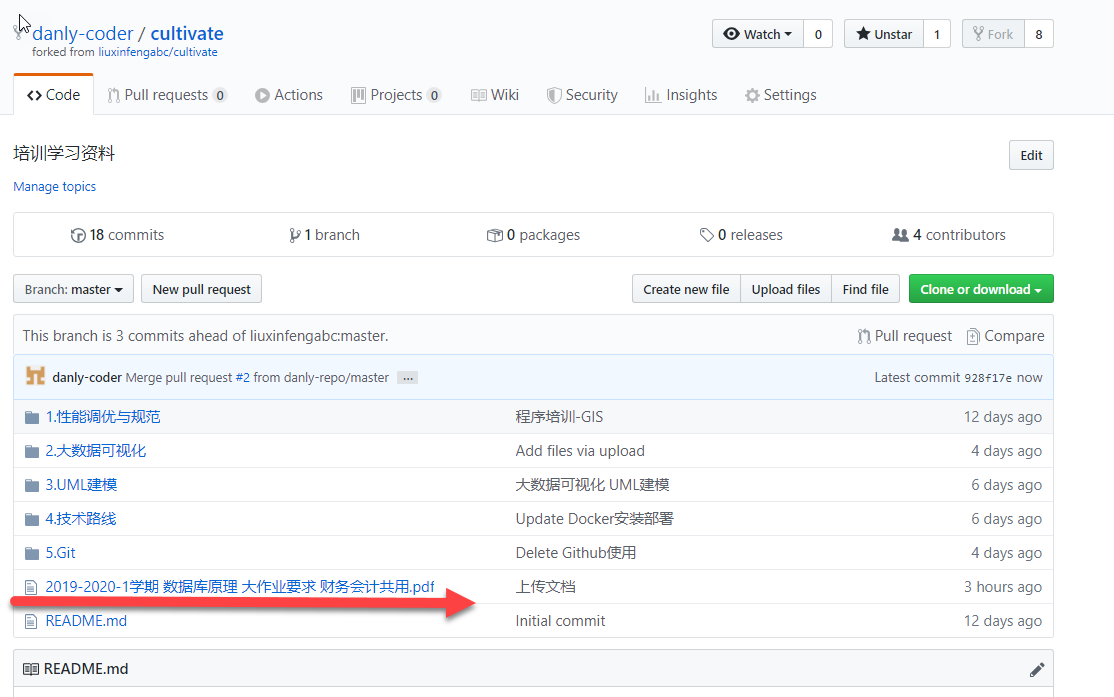
登录coder账户，点击Pull request，可以看到repo用户修改了项目仓库。





Coder账户如果觉的pull request的项目仓库可以，就可以Merge pull request（合并到主项目项目），这样主支项目仓库和分支项目仓库达成一致。如果觉的修改的不合适，就不用理睬。

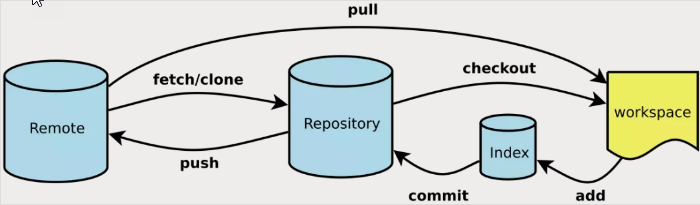
可以看到coder项目仓库下就产生了PDF文件。



1. Git

**1.Git是什么？**

Git是目前世界上最先进的分布式版本控制系统。  
工作原理 / 流程：



* Workspace：工作区
* Index / Stage：暂存区
* Repository：仓库区（或本地仓库）
* Remote：远程仓库（Github上建的仓库就相当于远程仓库）

1. SVN与Git的最主要的区别？

SVN是集中式版本控制系统，版本库是集中放在中央服务器的，而干活的时候，用的都是自己的电脑，所以首先要从中央服务器哪里得到最新的版本，然后干活，干完后，需要把自己做完的活推送到中央服务器。集中式版本控制系统是必须联网才能工作，如果在局域网还可以，带宽够大，速度够快，如果在互联网下，如果网速慢的话，就纳闷了。

Git是分布式版本控制系统，那么它就没有中央服务器的，每个人的电脑就是一个完整的版本库，这样，工作的时候就不需要联网了，因为版本都是在自己的电脑上。既然每个人的电脑都有一个完整的版本库，那多个人如何协作呢？比如说自己在电脑上改了文件A，其他人也在电脑上改了文件A，这时，你们两之间只需把各自的修改推送给对方，就可以互相看到对方的修改了。

1. **Git安装**

Git 各平台安装包下载地址为：<http://git-scm.com/downloads>

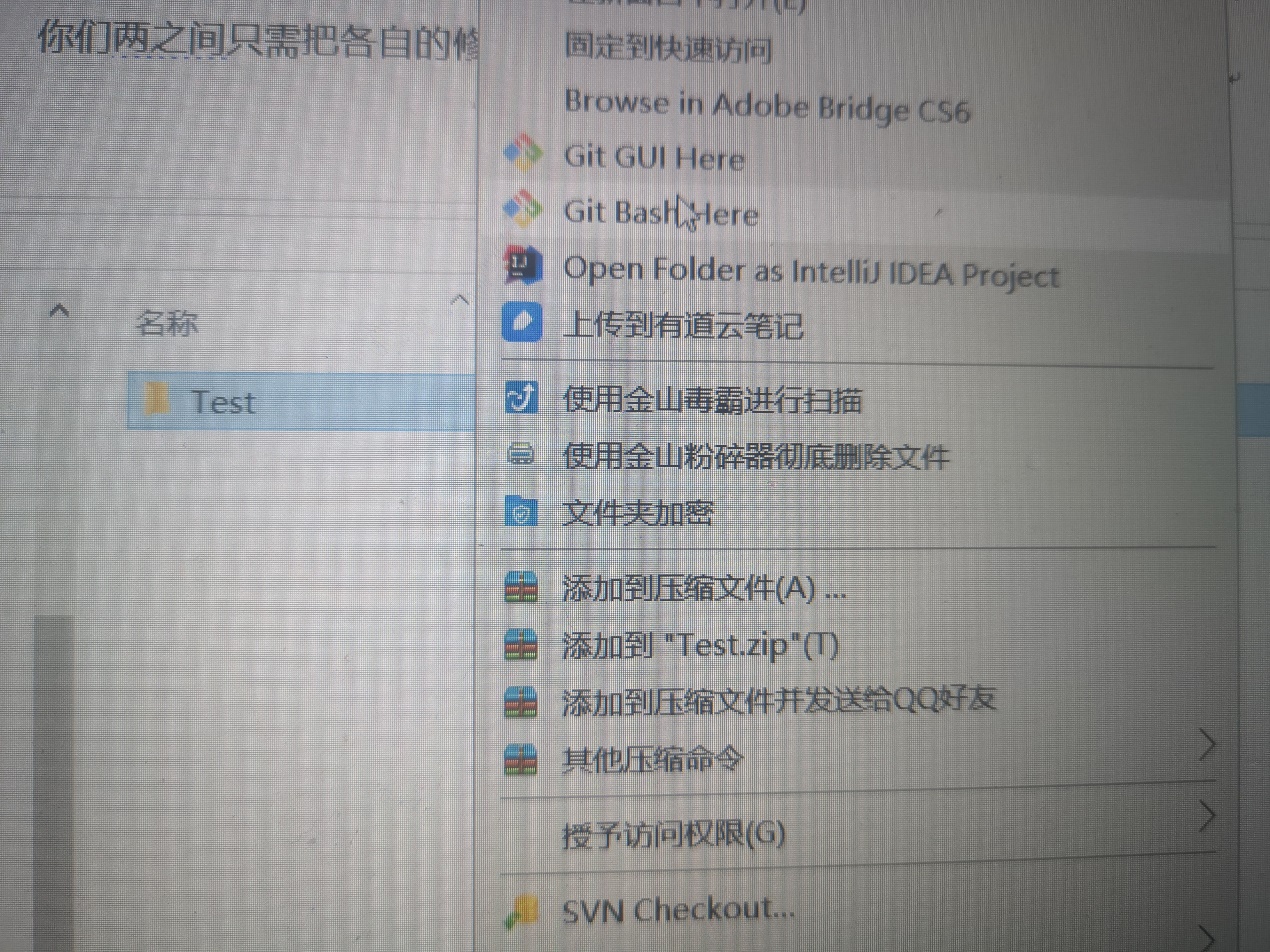
然后进行默认安装即可。安装完成后，在开始菜单里面找到 "Git --> Git Bash",如下：



新建一个文件夹作为我们的本地仓库

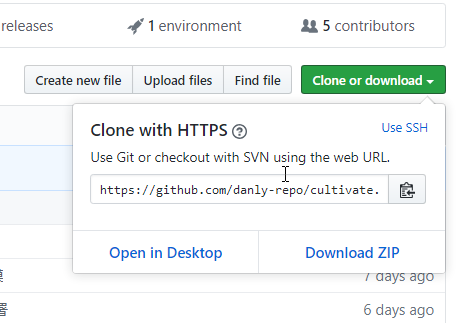


右击点击 Git Bash Here



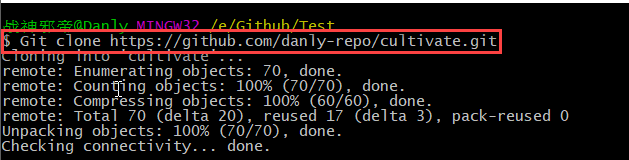
现在我们想把远程就是github的项目clone到本地

这是远程仓库的地址https://github.com/danly-repo/cultivate.git



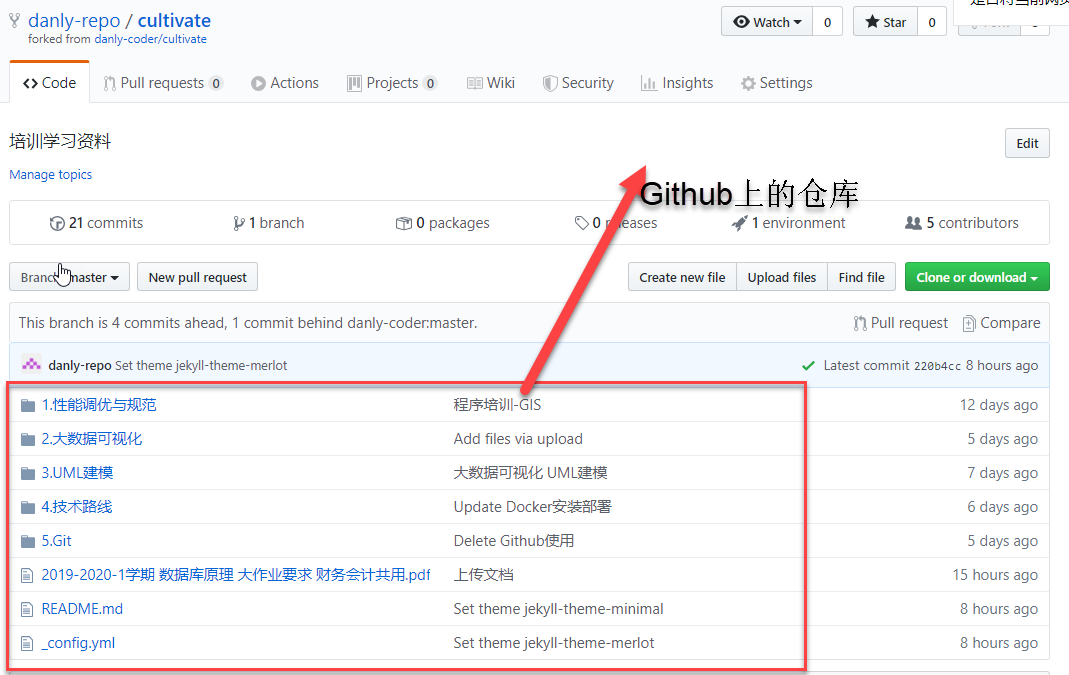
Git clone相当于以下四个步骤，所以不用Git init

>git init(创建本地存储库)  
> git remote add(将URL添加到该存储库)  
>git fetch(从该URL中获取所有分支到本地存储库)  
>git checkout(创建工作树中主分支的所有文件)



可以看到已经把Github上的远程仓库克隆到本地仓库

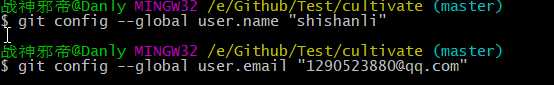




设置用户名和邮箱，修改后提交到远程仓库能够看到是谁提交的

设置用户名：$ git config --global user.name "Your Name"。

设置邮箱：$ git config –global user,email "email@example.com"。



首先touch（新建）一个文件

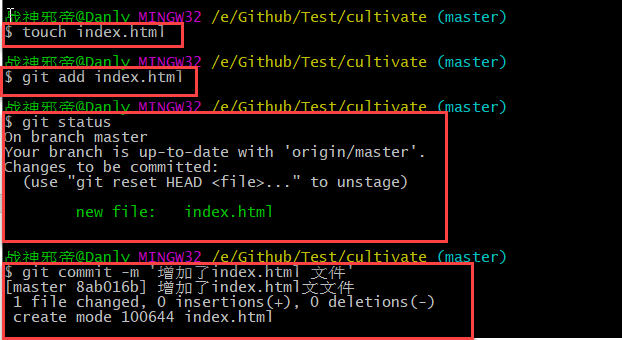
第一步：用命令git add .(.代表所有的文件，也可以只写文件名)告诉Git，把文件添加到仓库（实际上就是把文件修改添加到暂存区）

第二步：用命令git commit -m ‘你提交的信息’告诉Git，把文件提交到仓库（实际上 就是把暂 存区的所有内容提交到当前分支）

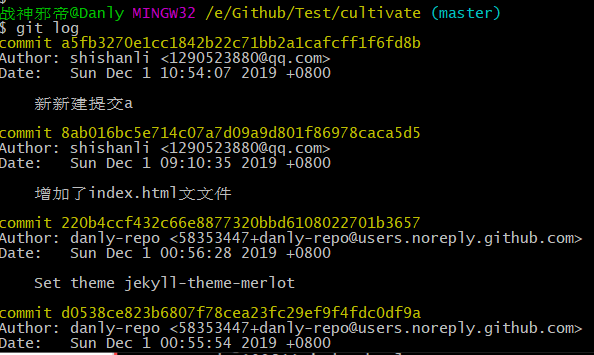
第三步：git status命令可以让我们时刻掌握仓库当前的状态

第四步：git pull 拉取服务器上更新文件

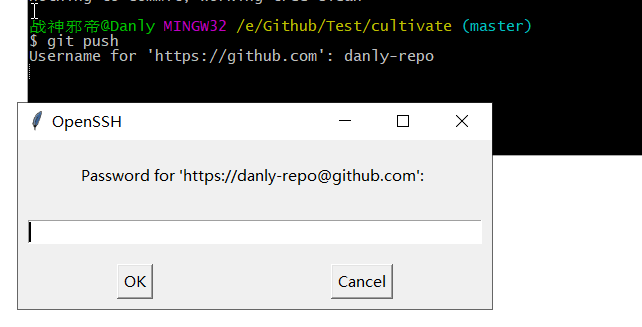
第六步： git push 把文件提交到服务器(要push前一定要先pull下来看下，然后在git status看下有没有冲突)



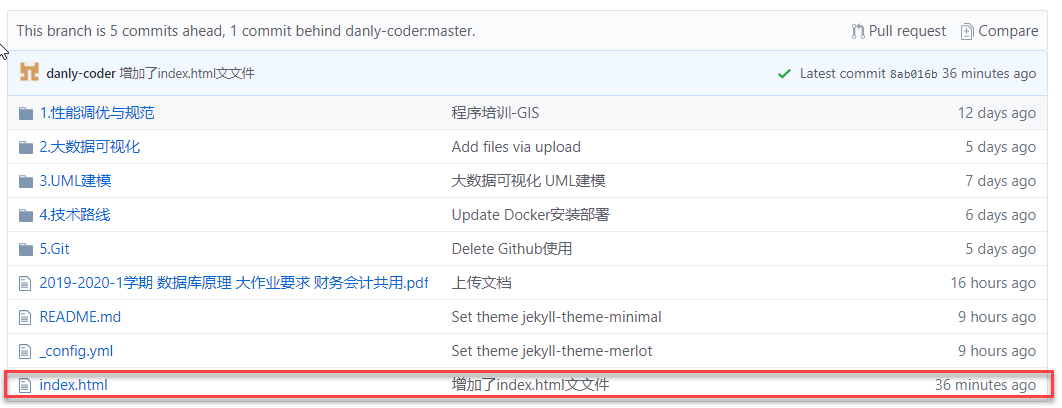
Git log可以查看更改日志



Git push时需要输入，远程仓库的账户和密码



在github远程仓库上就可以看到有了改变



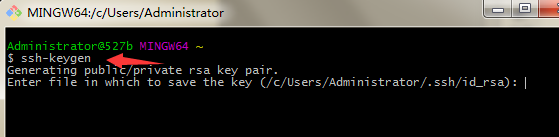
https 和 SSH 的区别：

1、前者可以随意克隆github上的项目，而不管是谁的；而后者则是你必须是你要克隆的项目的拥有者或管理员，且需要先添加 SSH key ，否则无法克隆。

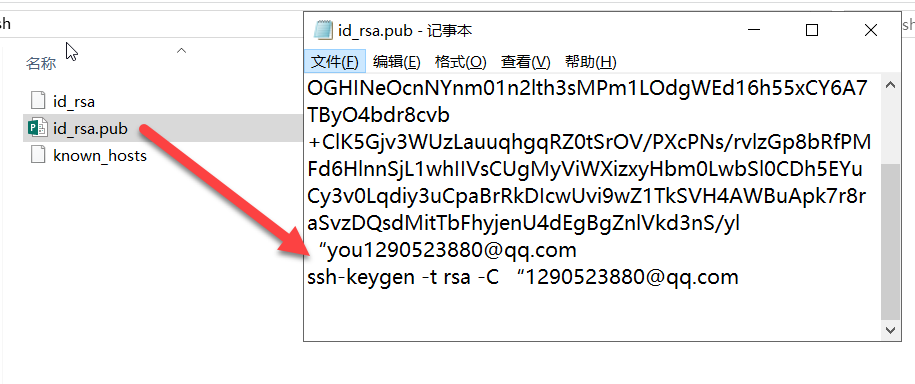
2、https url 在push的时候是需要验证用户名和密码的；而 SSH 在push的时候，是不需要输入用户名的，如果配置SSH key的时候设置了密码，则需要输入密码的，否则直接是不需要输入密码的。

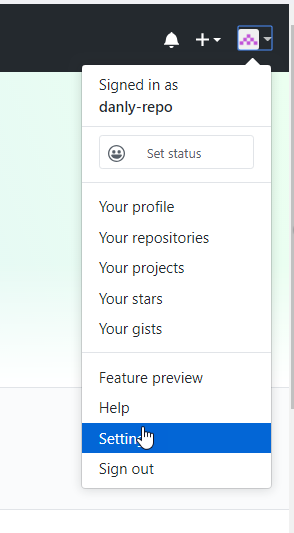
## 创建SSH Key

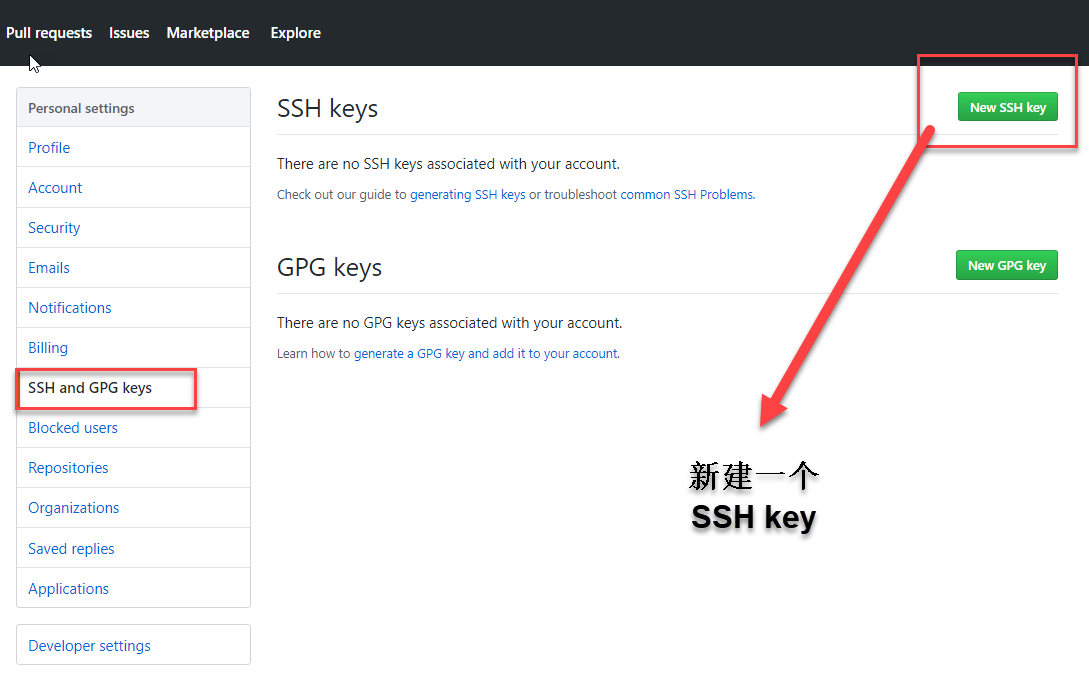
打开Git Bash,然后输入ssh-keygen生成公钥和私钥：  
http://images2015.cnblogs.com/blog/939417/201610/939417-20161012212745921-1496971112.png



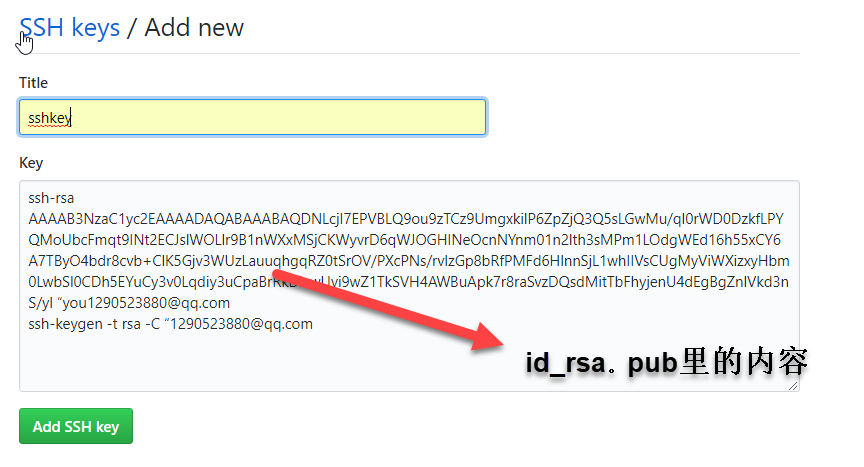
打开id\_rsa.pub，复制里面的内容

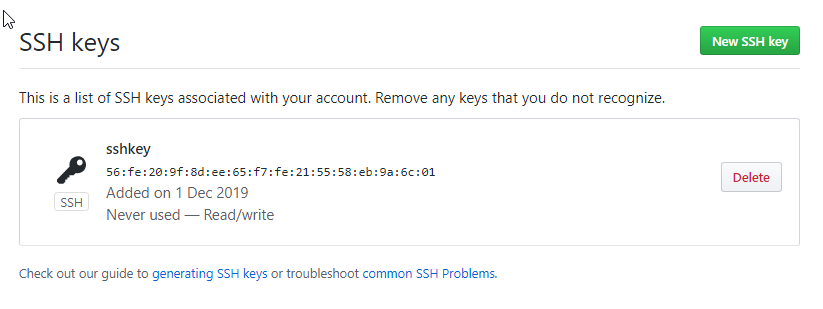






新建一个SSH keys，将复制的内容粘贴到下面的key中

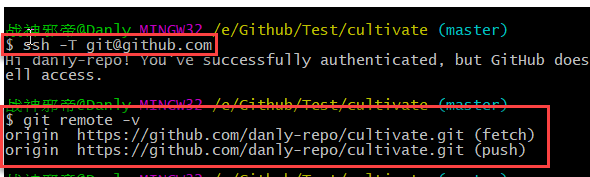




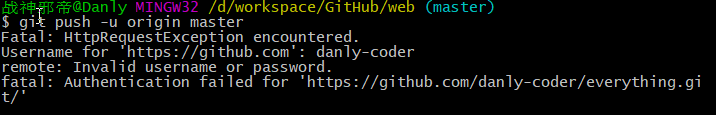
下面在git bash中输入

$ ssh -T [git@github.com](mailto:git@github.com)（通过秘钥连接）

$ git remote –v（查看已经链接的远程仓库）



$ git push origin master，将本地仓库上传到云端



如果出现上述错误

先执行

$ git config --system --unset credential.helper

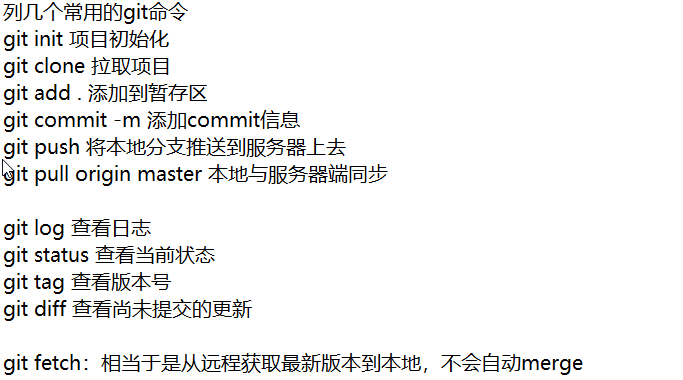
与github仓库建立连接

$ git remote add origin <https://github.com/danly-coder/everything.git>

从本地仓库上传到云端

$ git push -u origin master

**常用命令：**



## 冲突解决：

* git的回退

HEAD指向的版本就是当前版本，因此，Git允许我们在版本的历史之间穿梭， 使用命令git reset --hard commit\_id。

穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。

要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本

* git的撤销修改

场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时， 用命令git checkout -- file。

场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想 丢弃修改，分两步，第一步用命令git reset HEAD file，就回到了场景1，第 二 步按场景1操作。

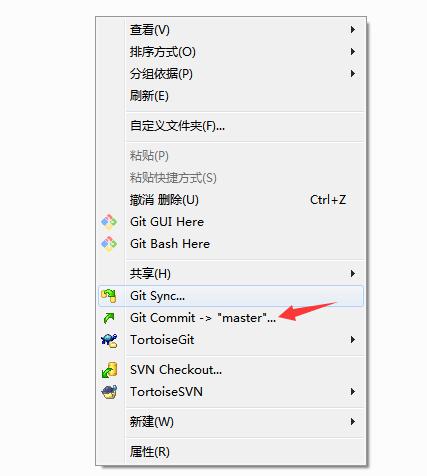
场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考版 本回退一节，不过前提是没有推送到远程库。

## git在window上直接使用tortoiseGit界面操作使用，（推荐使用window的操作）

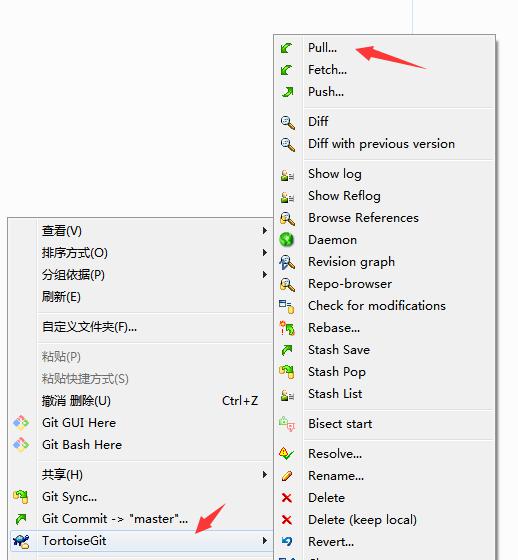
修改篇：

第一步：详细见（二、Git本地仓库创建及操作）

第二步：修改了文件后，右键选择commit，输入Message后，点Commit提交到本地仓库。



第三步：拉取服务器文件，git pull，点击OK，后点击close



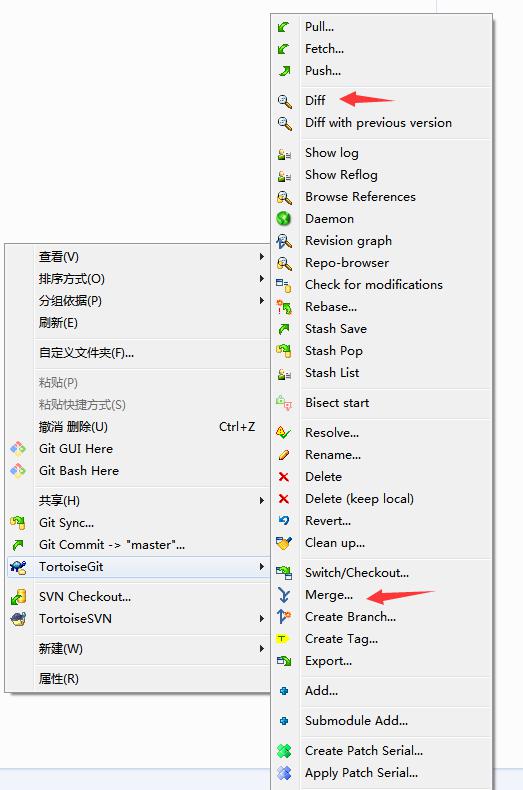
如果文件出现感叹号则表示有冲突，必须先解决冲突。

冲突解决方法：

1.在点击close后可以直接在弹出的窗口中点击yes来查看冲突，也可以通过点击Diff看本地仓库文件和服务器上面的文件哪里冲突了。

2.在弹出的窗口中双击打开冲突文件，解决冲突问题。

3.冲突解决完之后，删除在目录下生成出的多余文件，再次commit。



第四步：提交代表到服务器， git push



新增篇：

第一步：在仓库中新增文件后，先pull，然后可以在diff中查看和仓库区别

第二步：右键->commit，选择你需要提交的文件，然后点击commit

第三步：push文件

# Git本地仓库创建及操作

1.创建Git本地仓库文件夹

2.打开文件夹，右键->Git Clone...

3.输入URL，Load Putty Key选择前面存放的ppk文件，点击OK

4.对.gitignore中提到的文件进行取消跟踪（一般工程文件执行此步骤）

取消跟踪方法：

1.例如取消readme1.txt文件的跟踪，右键->git Bash Here，输入git rm --cached readme1.txt

2.例如取消.setting文件夹的跟踪，右键->git Bash Here，输入git rm -r --cached .settings/

# eclipse中使用Git导入工程

1.打开Git Repositories窗口，点击右上角Add an existing local Git Repository to this view图标

2.在Directory中选择你本地Git总仓库，在下方选择你想导入的Git仓库目录

3.打开你导入的仓库->Working Tree，选择你想导入的工程，右键->Import Projects...

4.调整选项，导入工程

5.右键已导入工程->Gradle->Refresh Gradle Project

# eclipse中使用Git同步工程

1.右键工程->Team->Synchronize Workspace进入同步窗口

2.对双向红箭头文件，打开，进行代码合并操作，操作完成后 右键->Mark asMerged

3.选择要上传的文件，右键->Add to Index/在Unstaged Changes中右键->Add to Index

4.在Staged Changes中可以查看要提交的文件，在Commit Message中输入提交内容备注，以及作者相关，点击commit至本地

5.Pull拉取服务器上更新文件

6.最后Push至服务器

# 冲突合并解决

1.首先标红的文件提交到本地库，然后pull下来，这个时候你会发现不能push上去。

2.然后你就要解决那些标红的文件的冲突，然后再commit到本地库

3.最后你再push上去