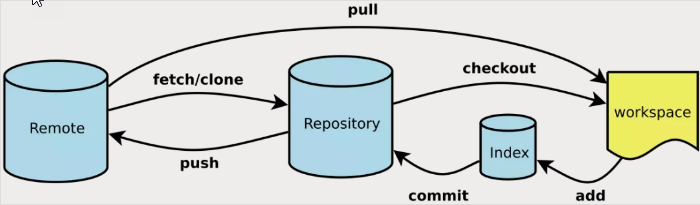
# Git简介

Git是目前世界上最先进的分布式版本控制系统。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。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。

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。

## 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）

## Git安装

**一、准备工作**

1.Git官网（<https://git-scm.com/>）下载Git。

2.GitHub官网（<https://github.com/>）注册账号。

**二、安装Git**

1.点击安装，选择目录为根目录，文件夹为Git，然后Next->默认安装。

2.安装完成后打开根目录，找到Git文件夹，右键git-bash.exe可执行文件将其固定的工具栏，点击git-bash.exe运行。

3.输入指令 ls 查看文件目录。

此为运行正常安装完成！

**三、运行Git**

1.设置用户名和邮箱。

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

设置邮箱： $ git config --globaluser,email "email@example.com"。

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

2.输入 $ git init 启动Git。

**四、在GitHub上设置公钥**

***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。***

1.新建/连接主存储库。

2.输入 $ ssh-keygen -t rsa -C "[email@example.com](mailto:%2213963639551@163.com%22)" 生成公钥，在C:\Users\Administrator\.ssh下找到id\_rsa.pub文件，记事本打开复制公钥。

3.在GitHub上新建密钥，自定义标题（title）将公钥粘贴保存。

**五、克隆远程存储库到本地**

新建一个文件夹作为本地仓库，右击点击 Git Bash Here，把远程github的项目clone到本地，远程仓库的地址<https://github.com/liuxinfengabc/cultivate.git>

命令：$ git clone<https://github.com/liuxinfengabc/cultivate.git>

Git clone相当于以下四个步骤

>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)

**六、添加并推送文件**

1.把项目文件复制到本地的 D:\webroot\Git\gitskills 存储库中。

2.将项目文件添加到工作区中。

命令：git add <file>

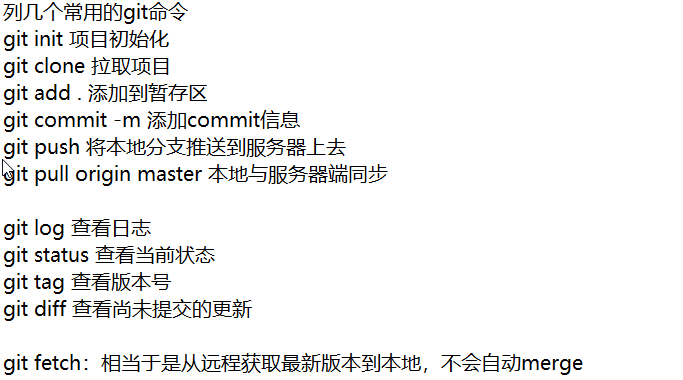
3.将工作区中的项目文件存入master分支中。

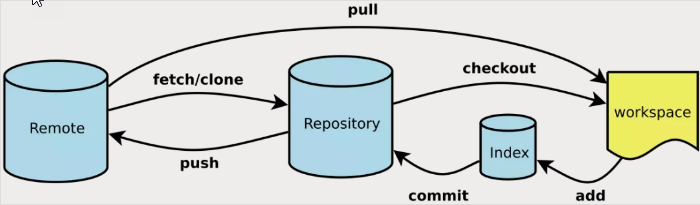
命令：git commit -m "add <file>"

4.将本地的项目文件推送到GitHub服务器上。

命令：git puch github 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

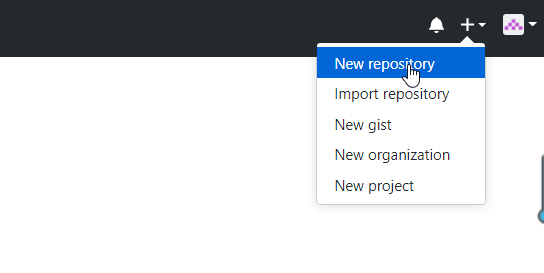
场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时， 用命令git checkout -- file。

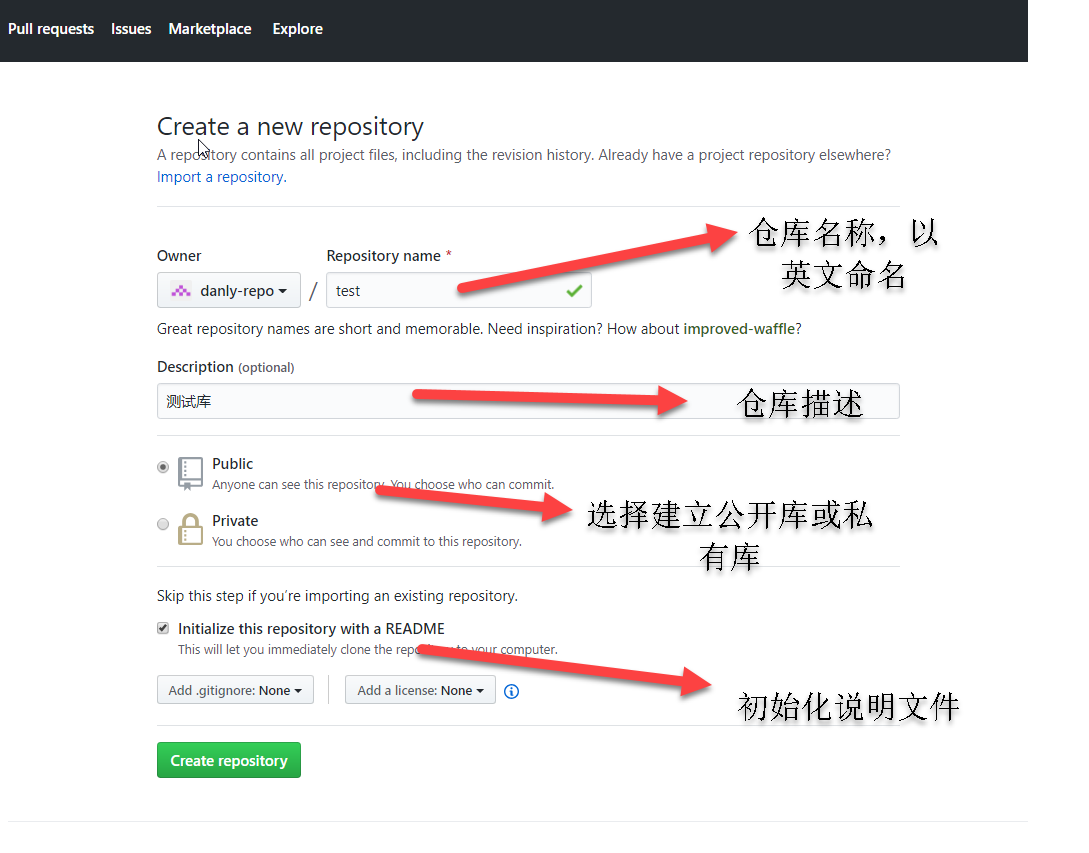
场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想 丢弃修改，分两步，第一步用命令git reset HEAD file，就回到了场景1，第 二 步按场景1操作。

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。

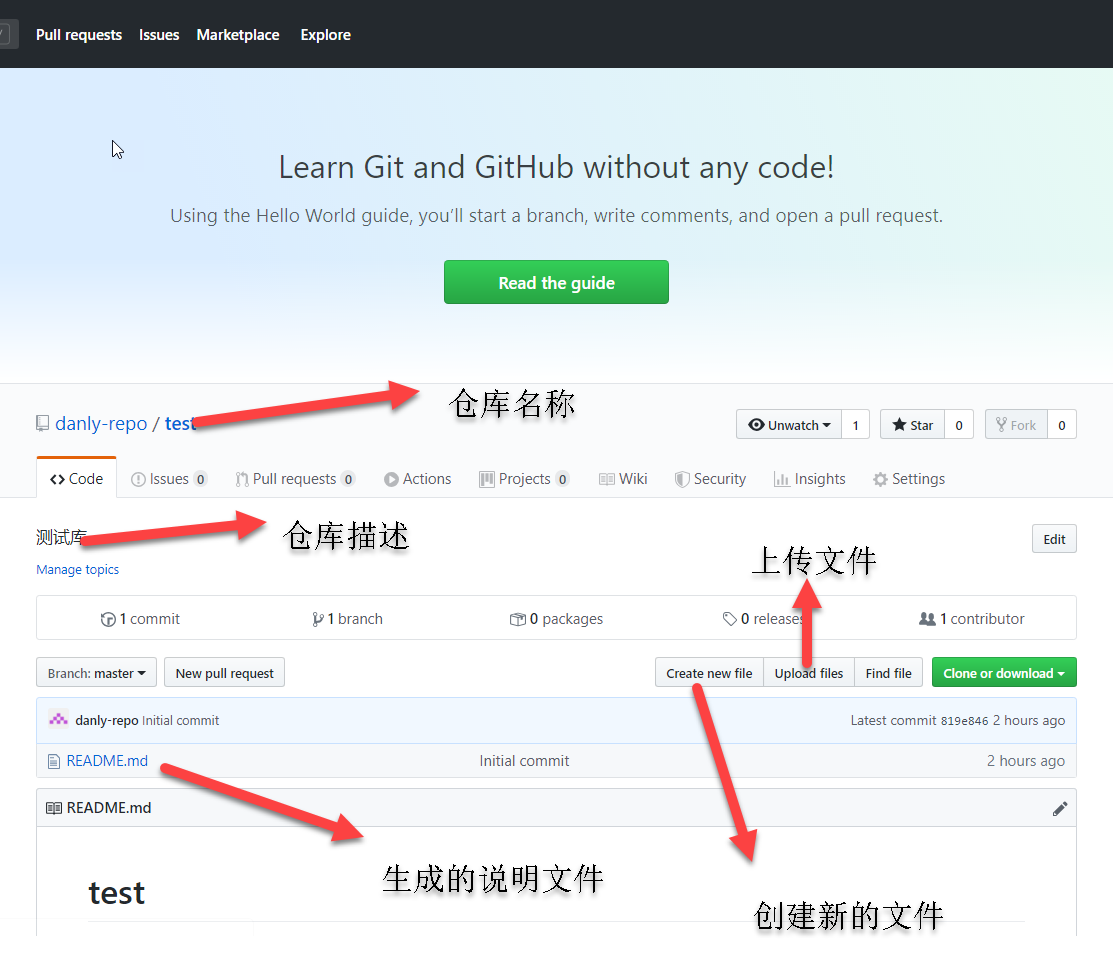
# GitHub操作

## 创建仓库



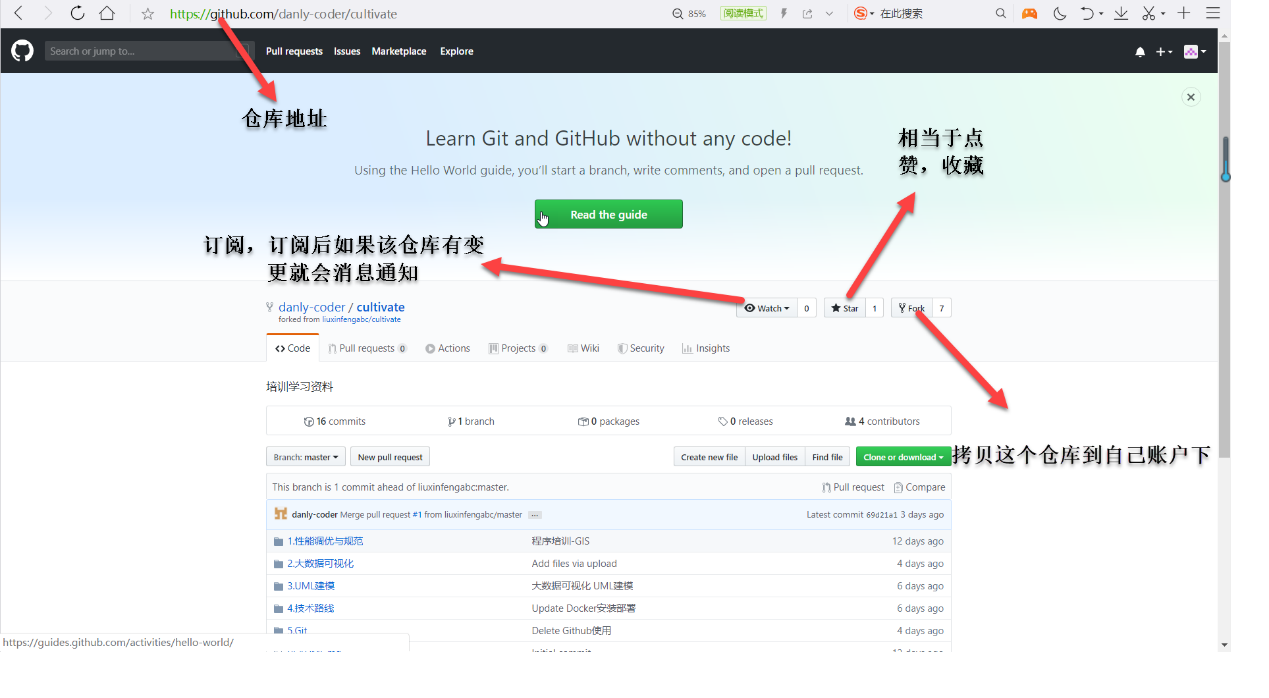


生成的仓库如下

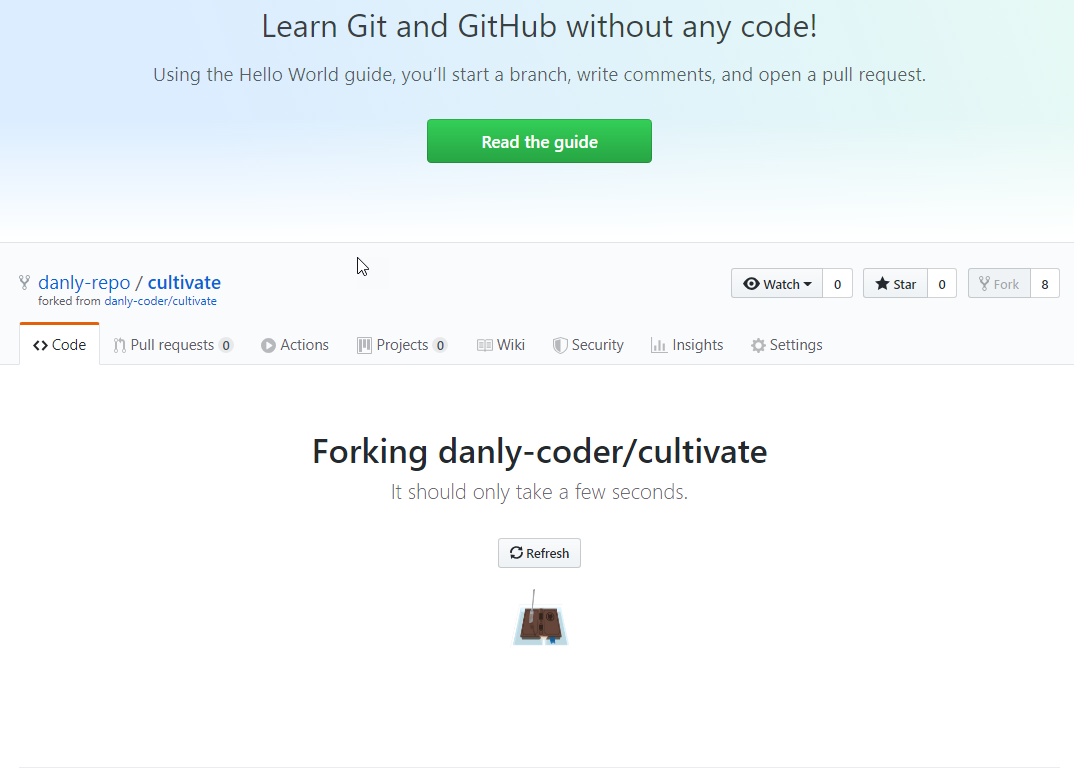


打开另外一个仓库的网址<https://github.com/danly-coder/cultivate>

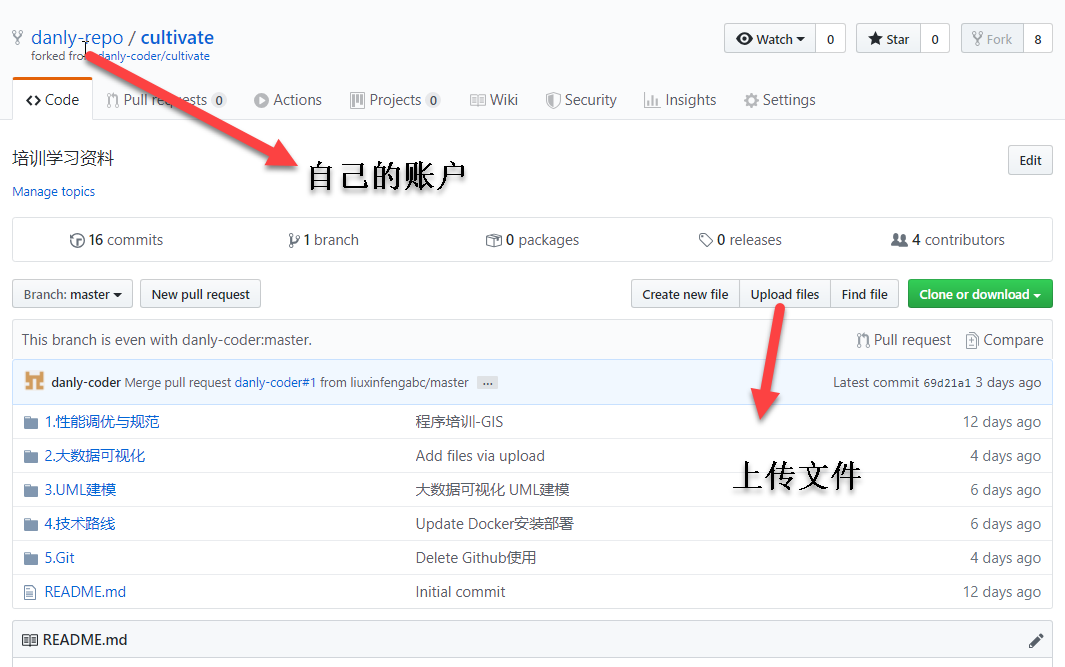
这是danly-coder用户下cultivate库地址



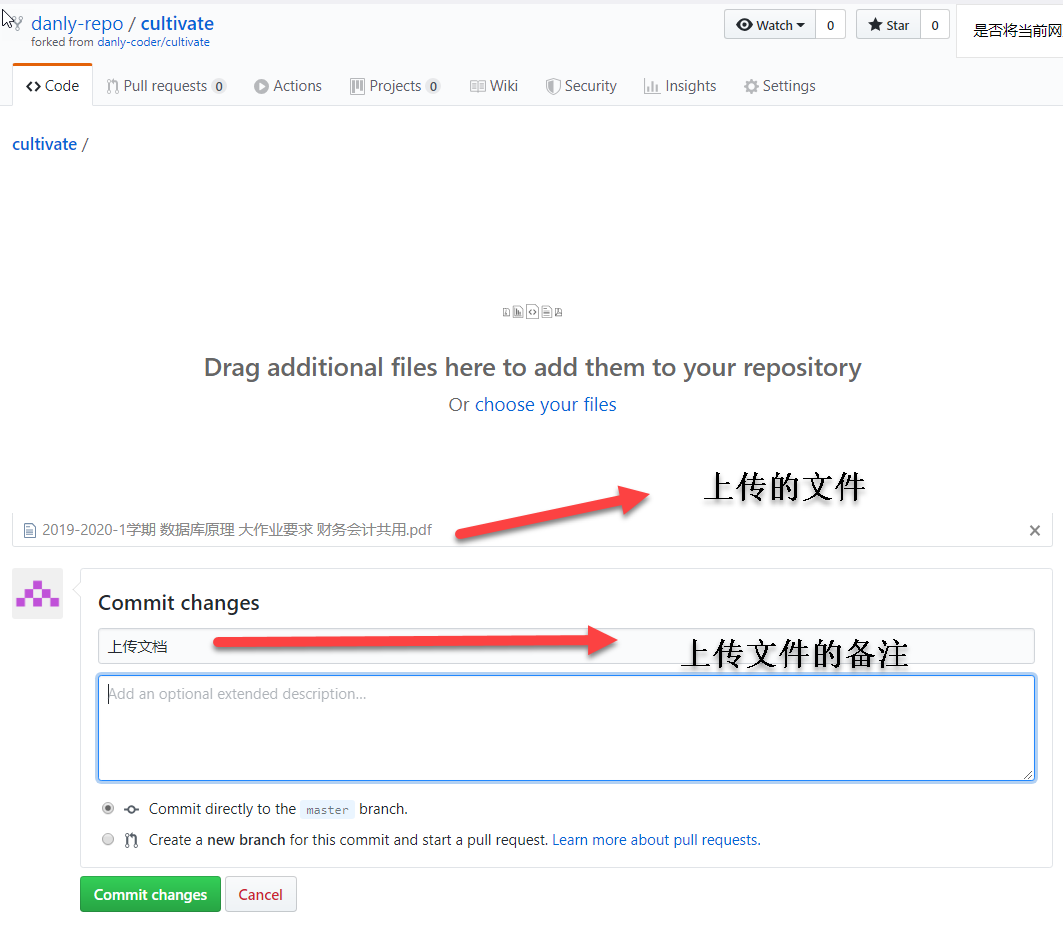
Fork该仓库



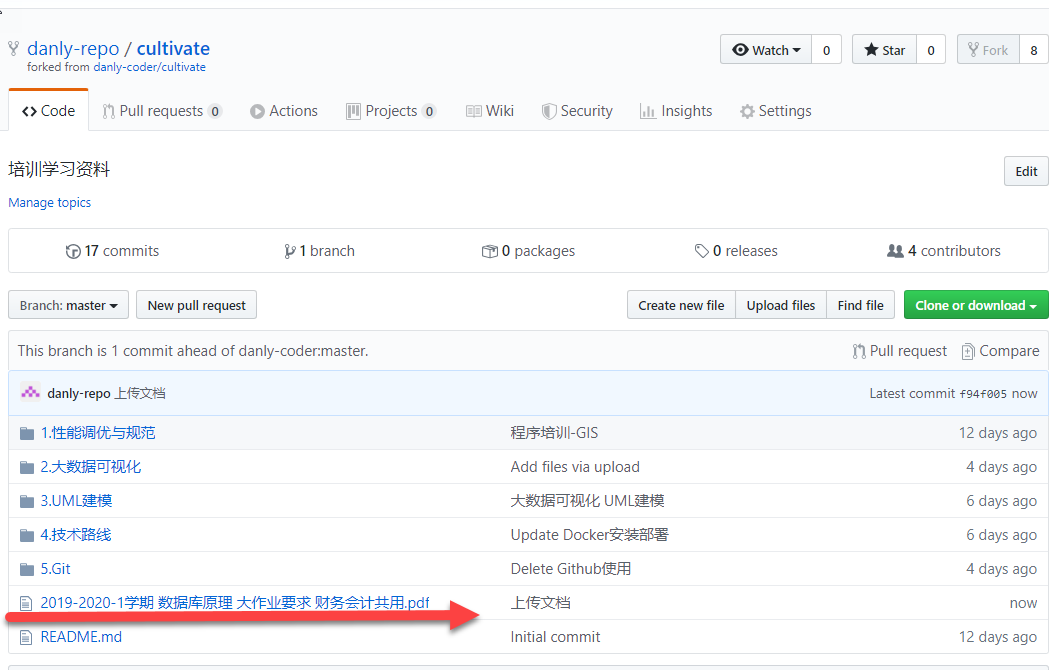
在自己的账户就是danly-repo下就可看到cultivate库



点击上传文件后进行上传，添加备注后提交改变。

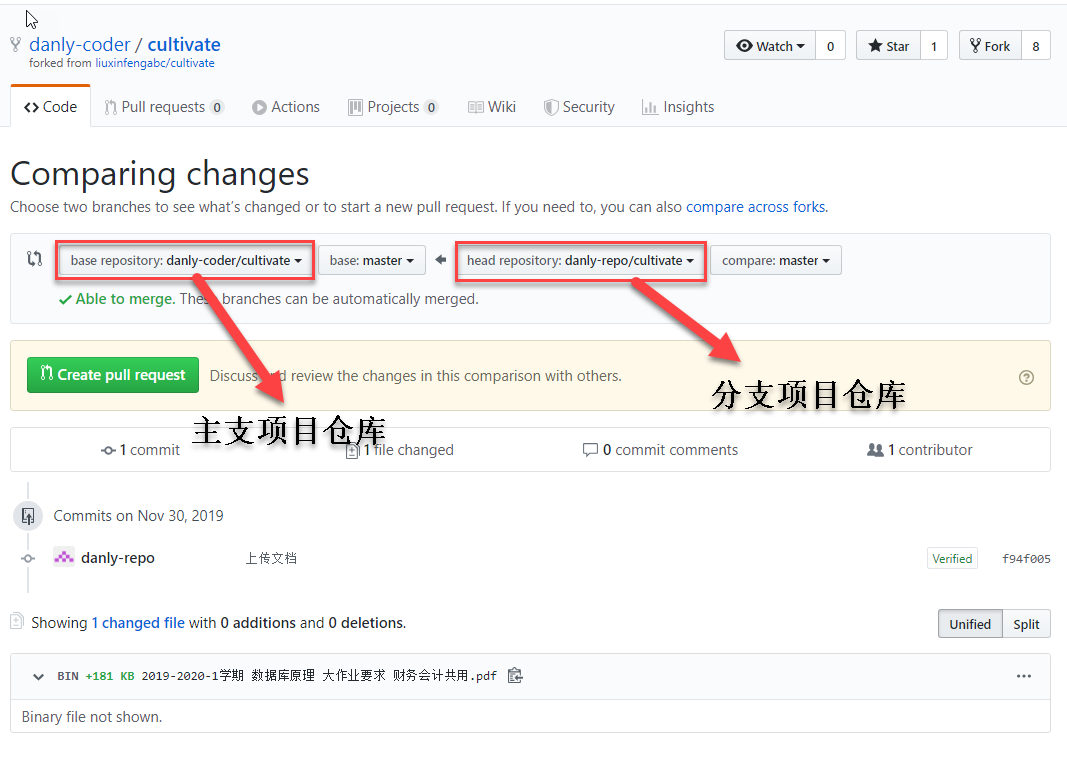


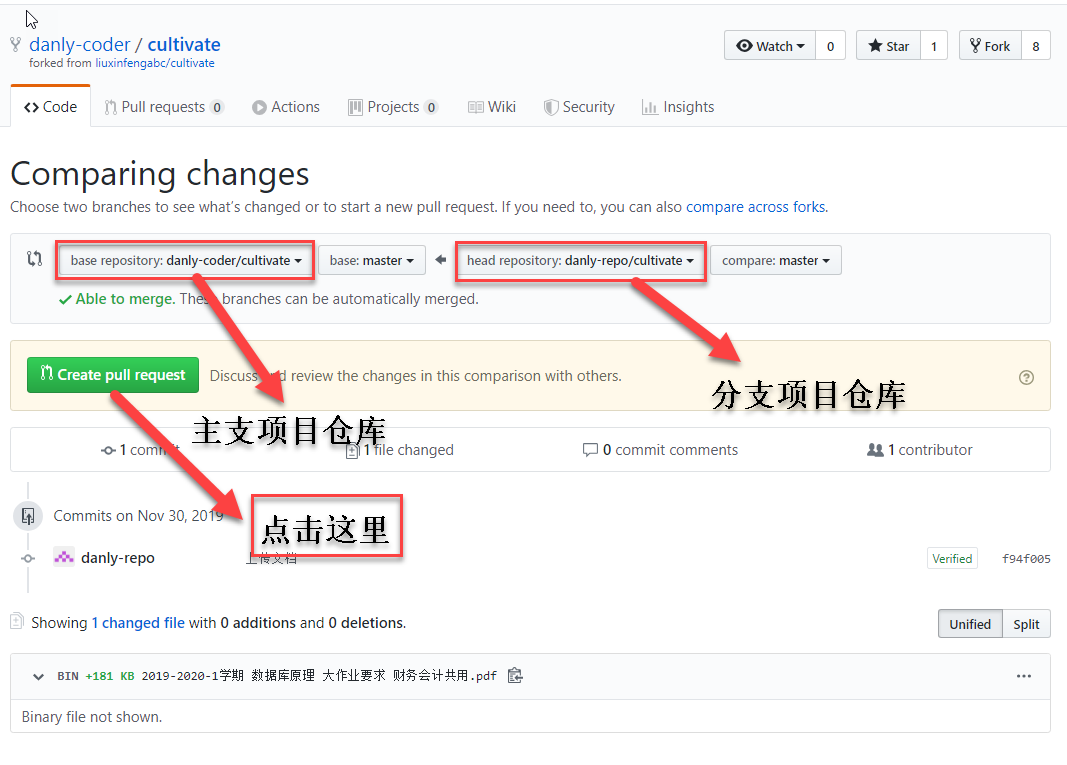
可以看到这个账号下cultivate库增加了文件



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。



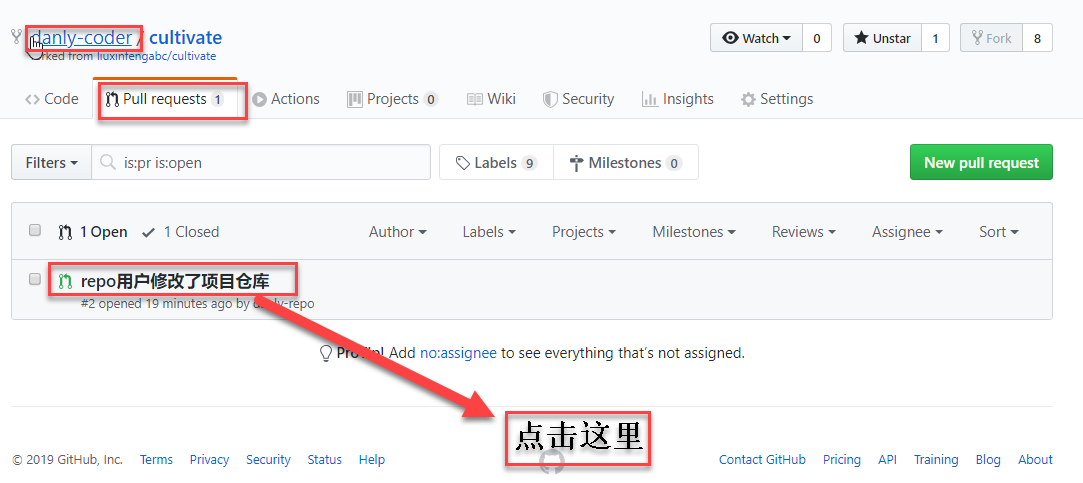


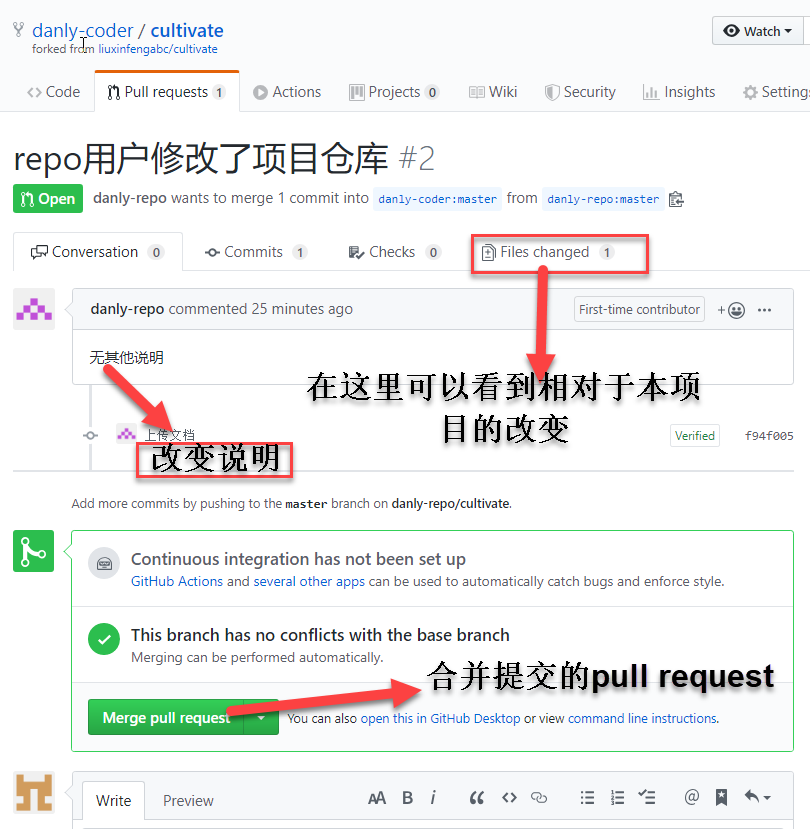




这样就完成了pull request，向主支仓库提交了修改后的项目。

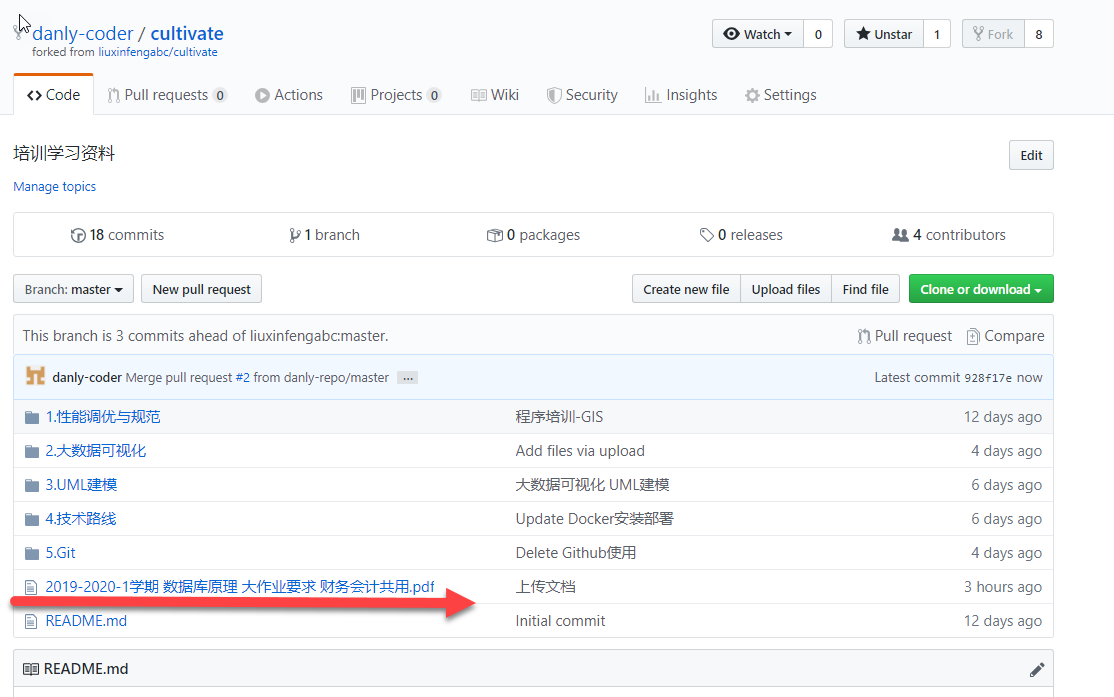
登录coder账户，点击Pull request，可以看到repo用户修改了项目仓库。





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。

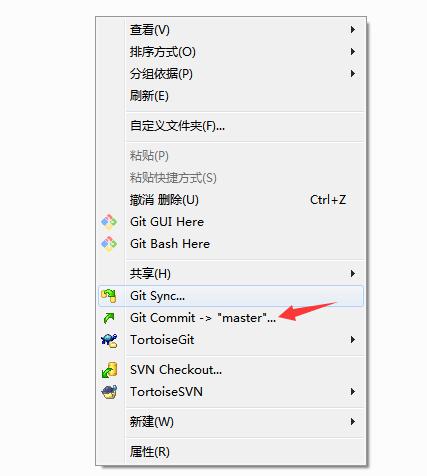


# 高级应用

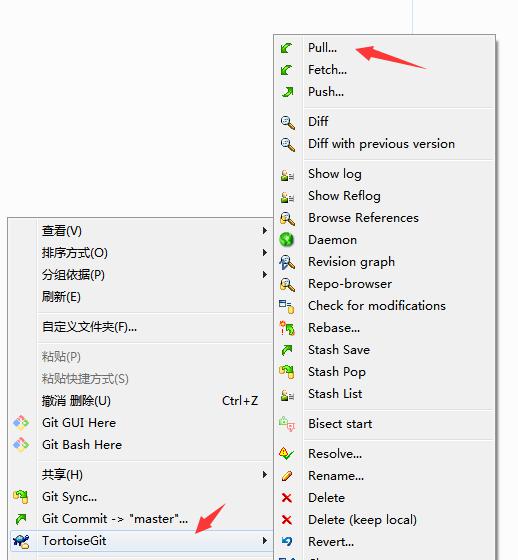
## tortoiseGit使用(windows操作）

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



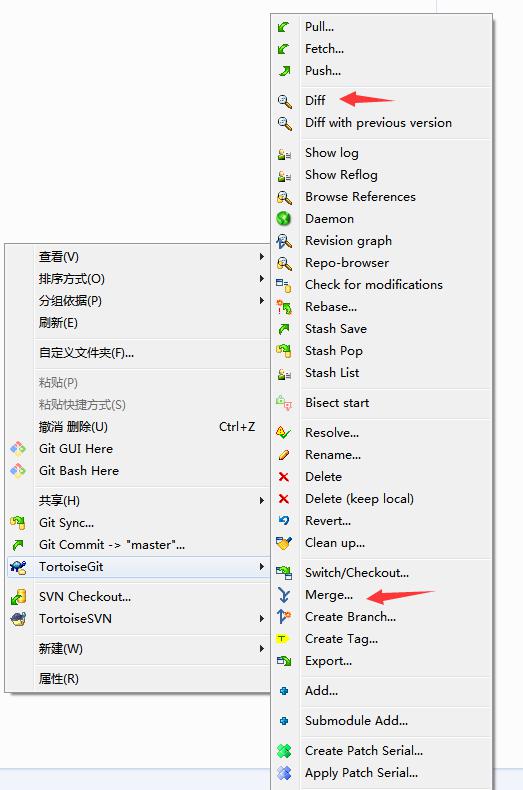
如果文件出现感叹号则表示有冲突，必须先解决冲突。

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

## 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