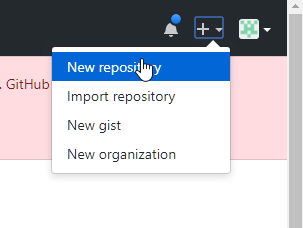
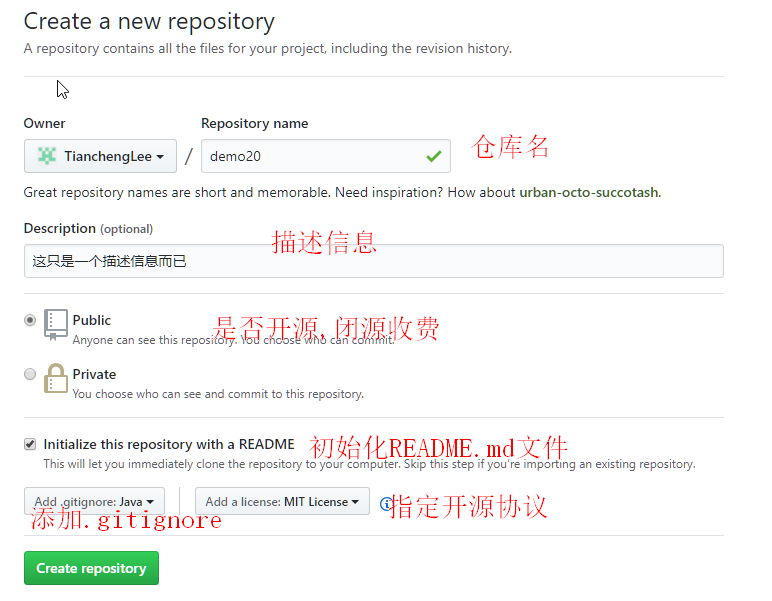
**任务1: 在Github上建立一个仓库并关联到本地,提交代码**

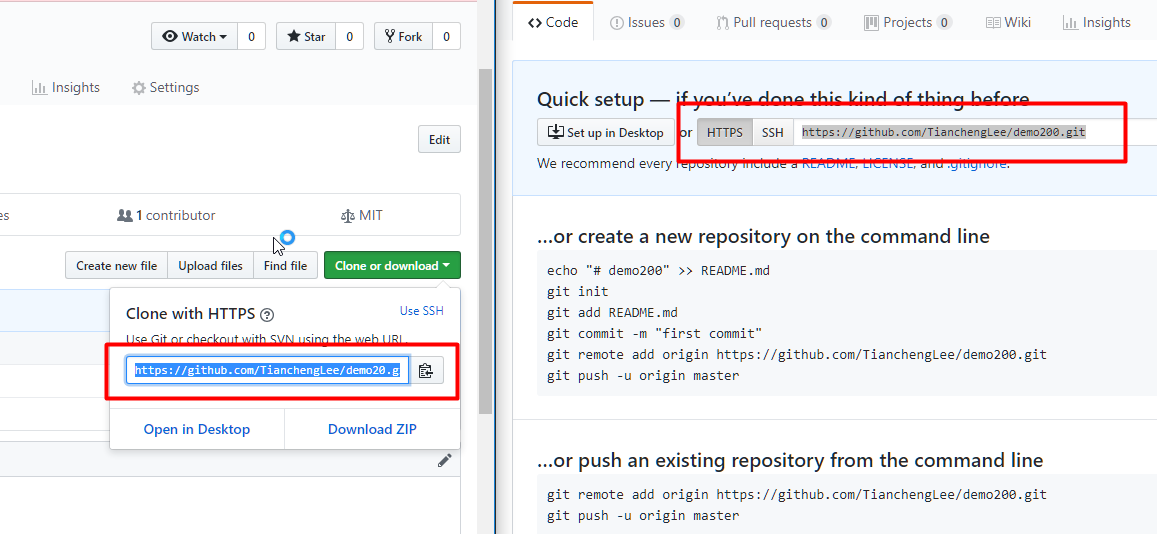
1. 在github上建立仓库 Somthing went wrong



创建仓库:



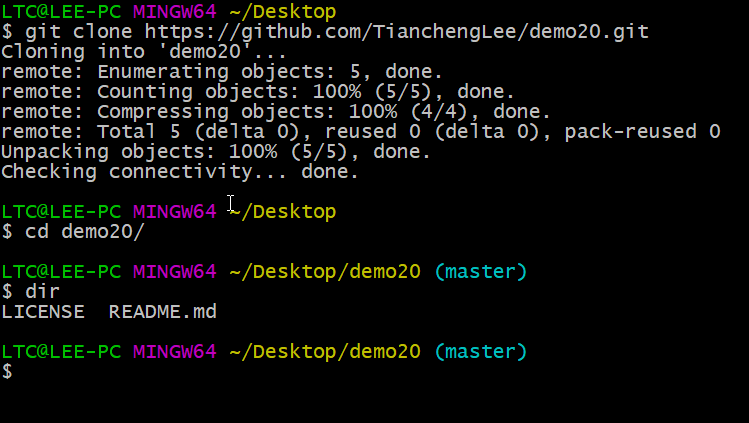
1. 将远程仓库克隆(自动关联)到本地



拷贝红框中的链接

在本地找个风水宝地打开git bash控制台:

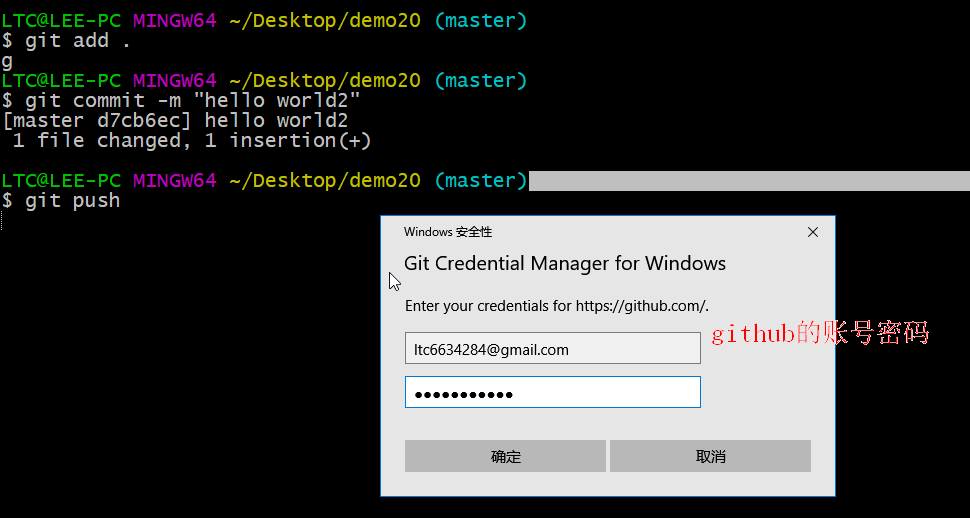
输入 git clone <https://github.com/TianchengLee/demo20.git>

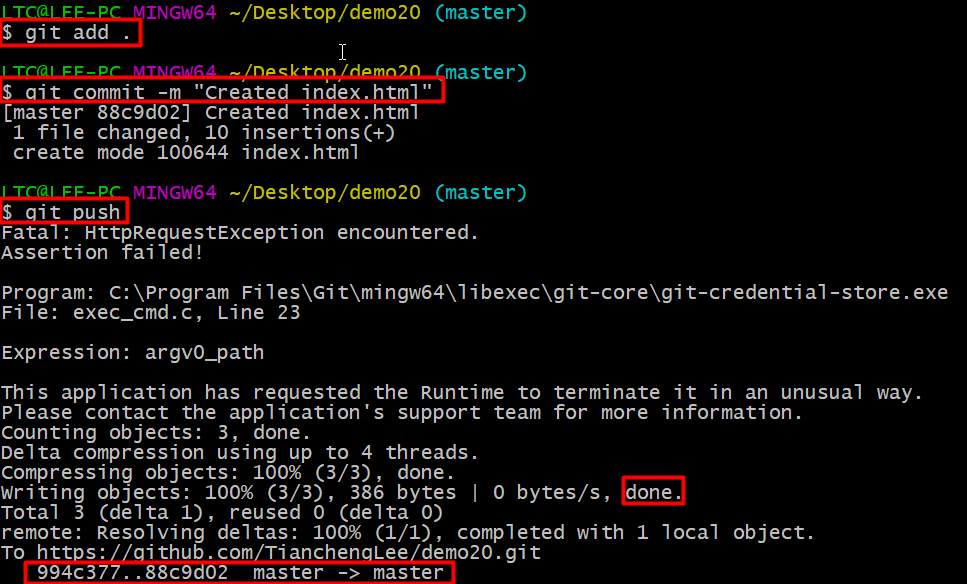


到此, 本地的demo20已经和远程的demo20仓库建立好关联了

编写代码后提交:

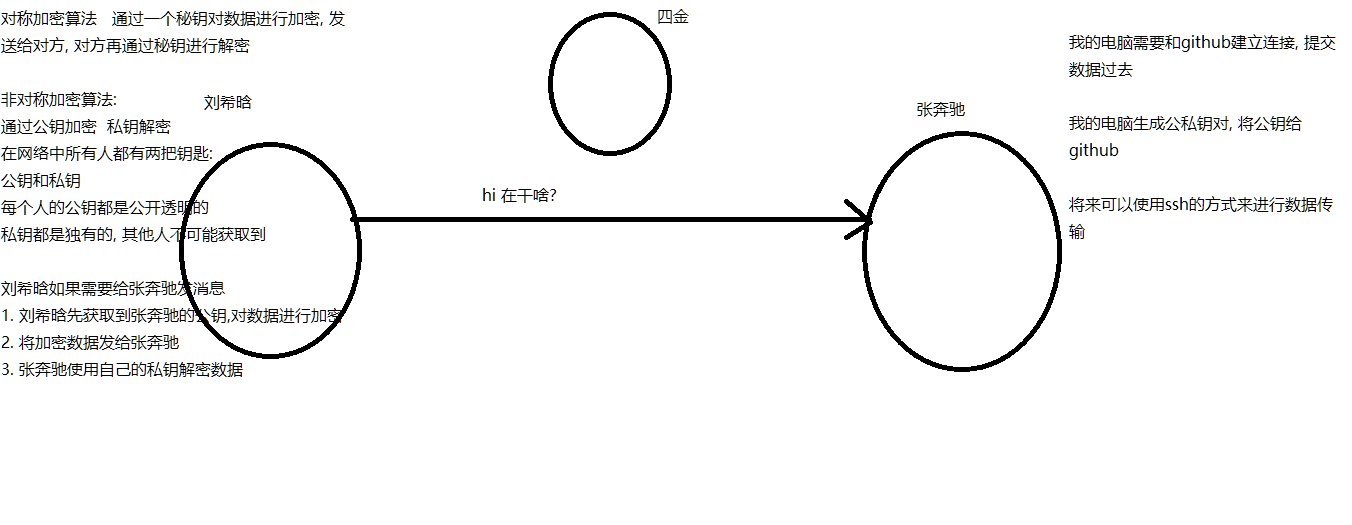
如果弹出提示框输入github的账号密码, 用邮箱作为账号





### 任务2: 配置ssh(以后提交代码可以不用输入用户名和密码)

非对称加密原理:

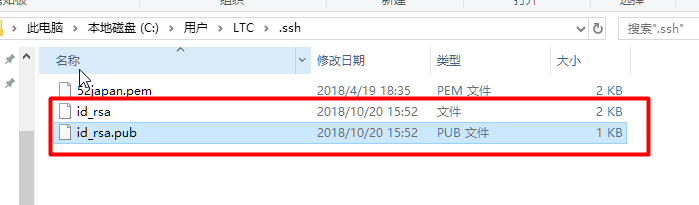


生成公私钥对:

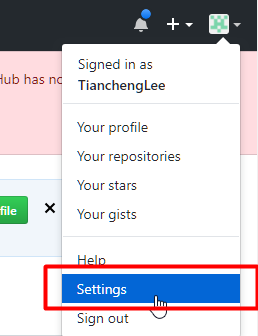


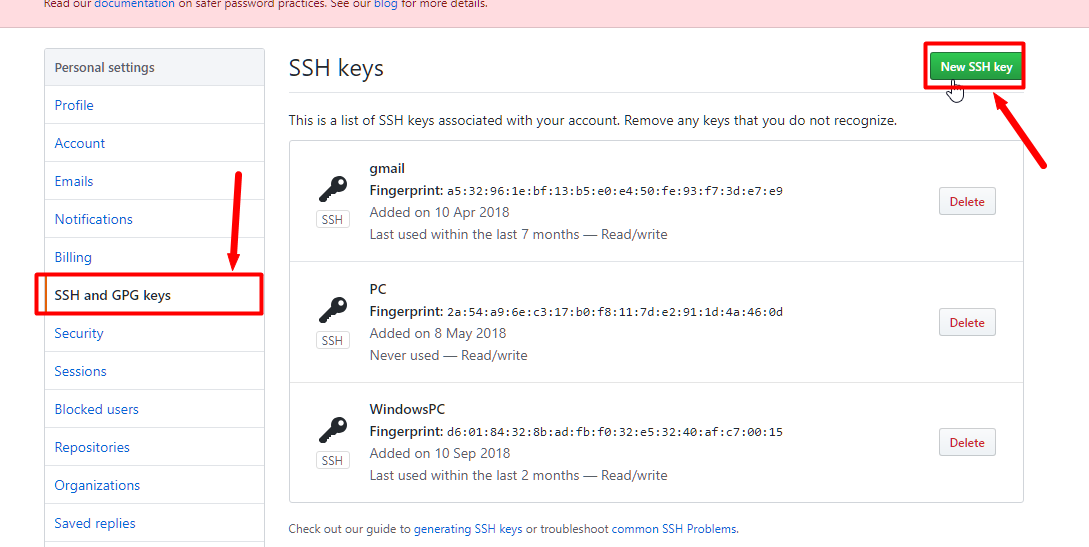
生成完毕后在

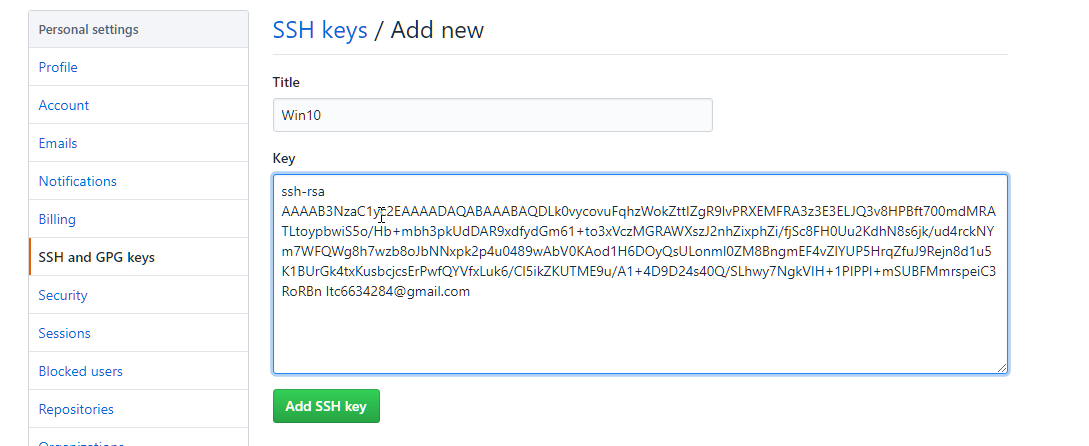
c:\users\当前用户名\.ssh\ 路径下会有一对秘钥:



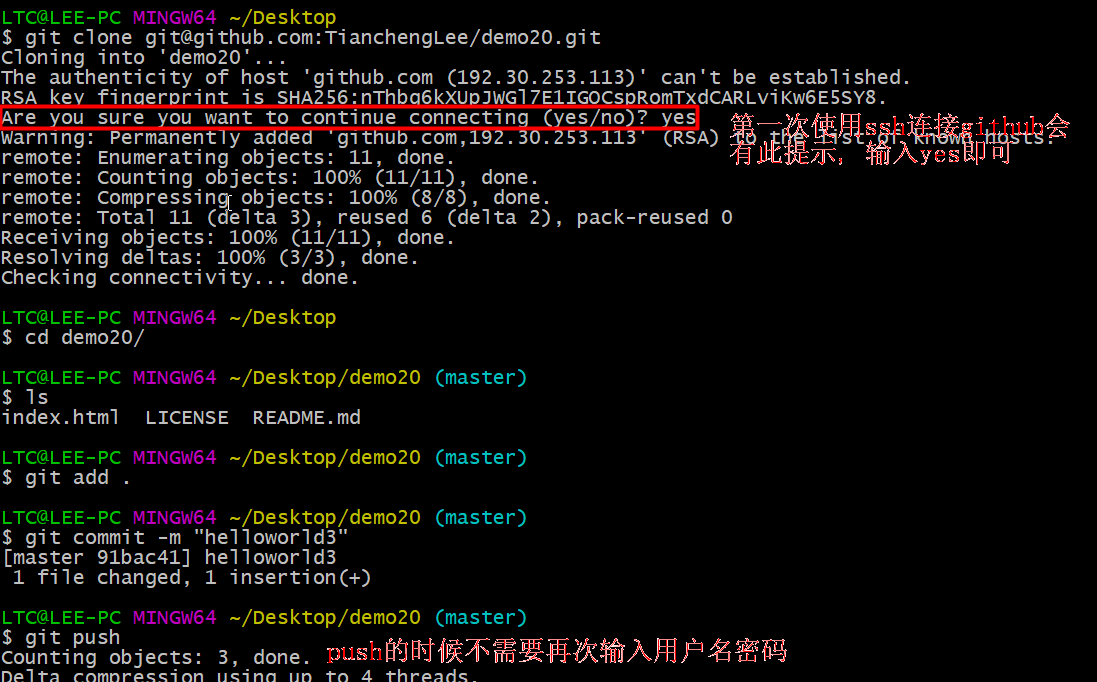
在github中添加自己的公钥:





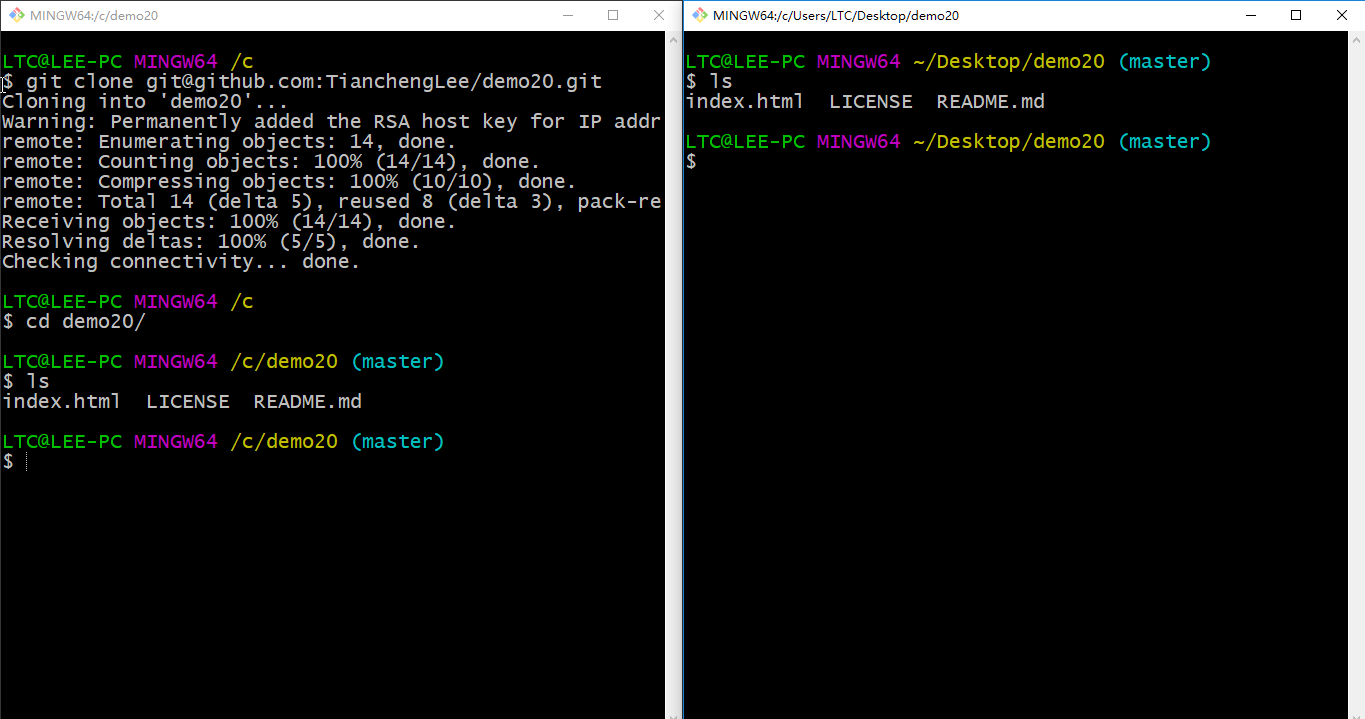


验证是否成功:



### 任务3: 多人合作开发

模拟多人合作开发 一人在C:\demo20 另一人在桌面\demo20



场景1: 两个人各自开发自己的模块, 互不干扰没有任何冲突

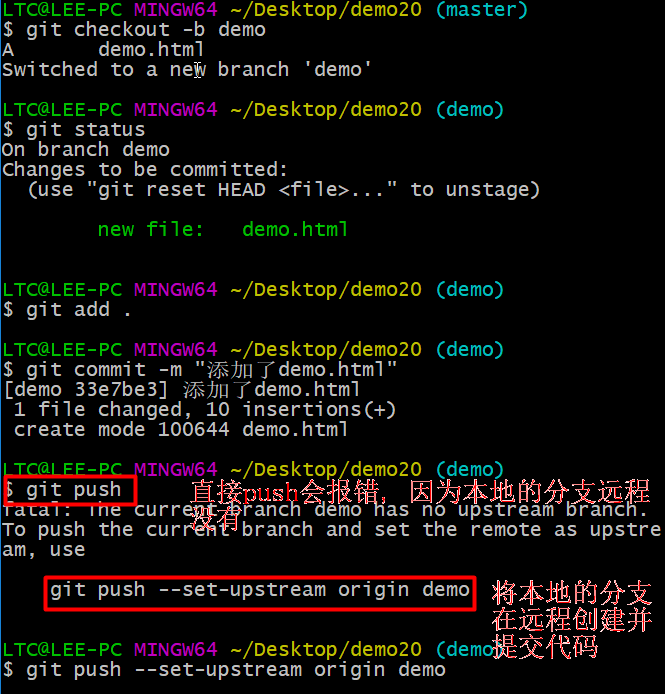
C盘的demo20写index.html

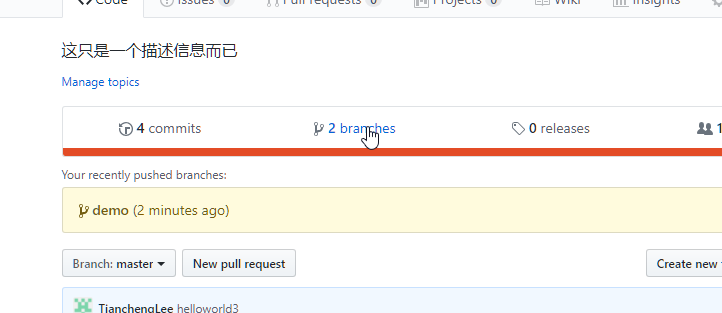
桌面的demo20写demo.html

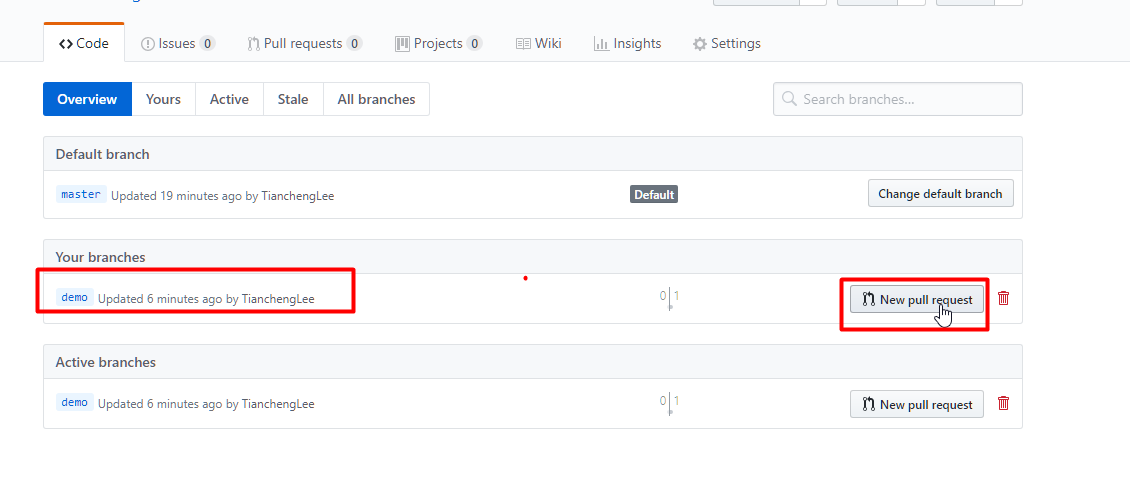
1. 新建分支
2. 将修改的代码提交到分支
3. 使用git push将本地分支提交到远程分支, 注意, 此时会报错, 根据提示输入

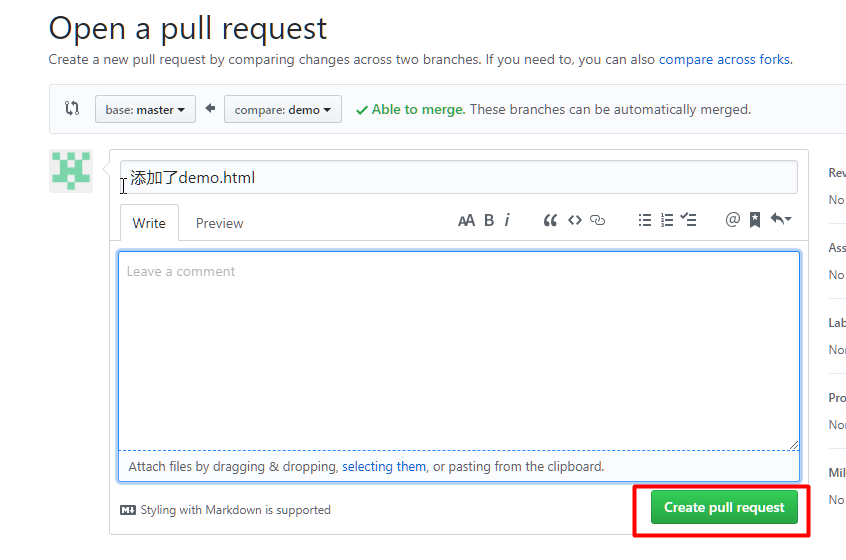
git push –set-upstream origin demo 即可完成提交

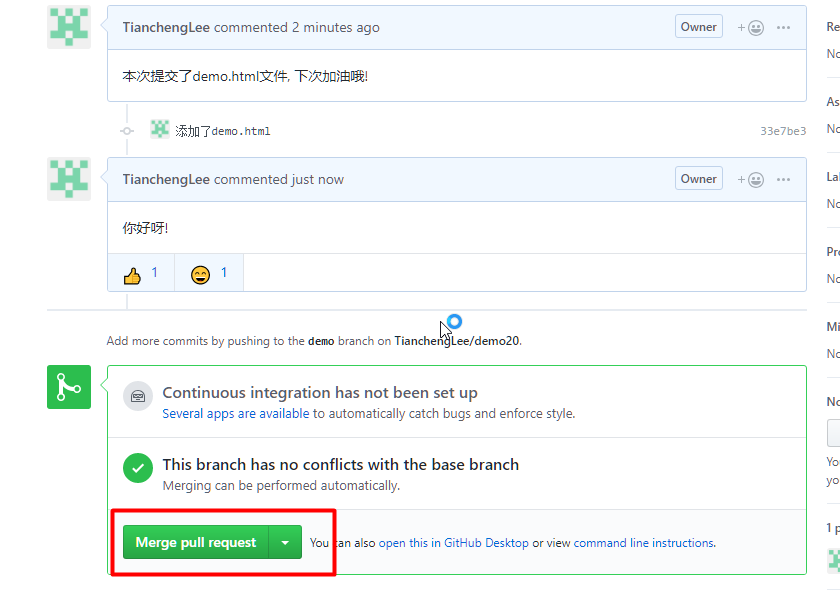
1. 提交完毕后在github中进行new pull request将demo分支合并至master分支
2. 在本地切换到master分支, 使用git pull 更新最新的代码即可

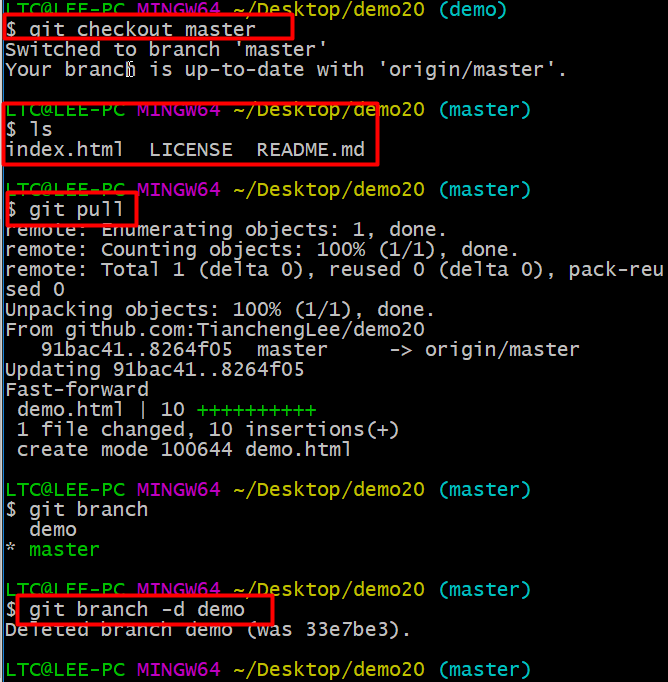












场景2: 两人同时对一个文件进行了修改, 出现了冲突, 在github中解决冲突后合并

两边同时修改index.html文件

1. 在本地新建分支并提交修改的代码
2. 分别提交自己的分支到仓库
3. 在不同的分支new pull request合并代码到master
4. 后new pull request的分支会有冲突, 可以在线解决冲突
5. 解决完后标记为已解决
6. 合并分支

