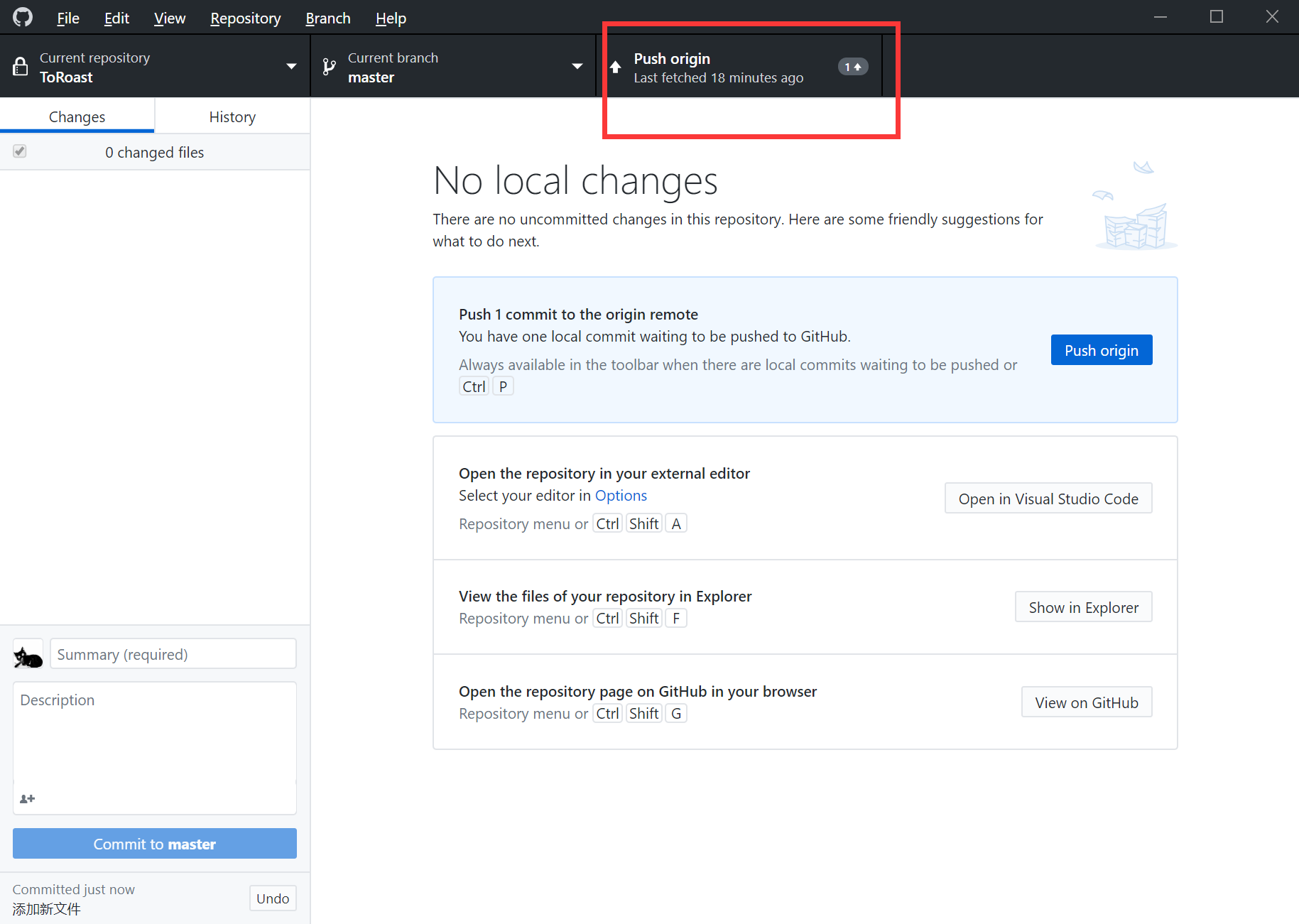
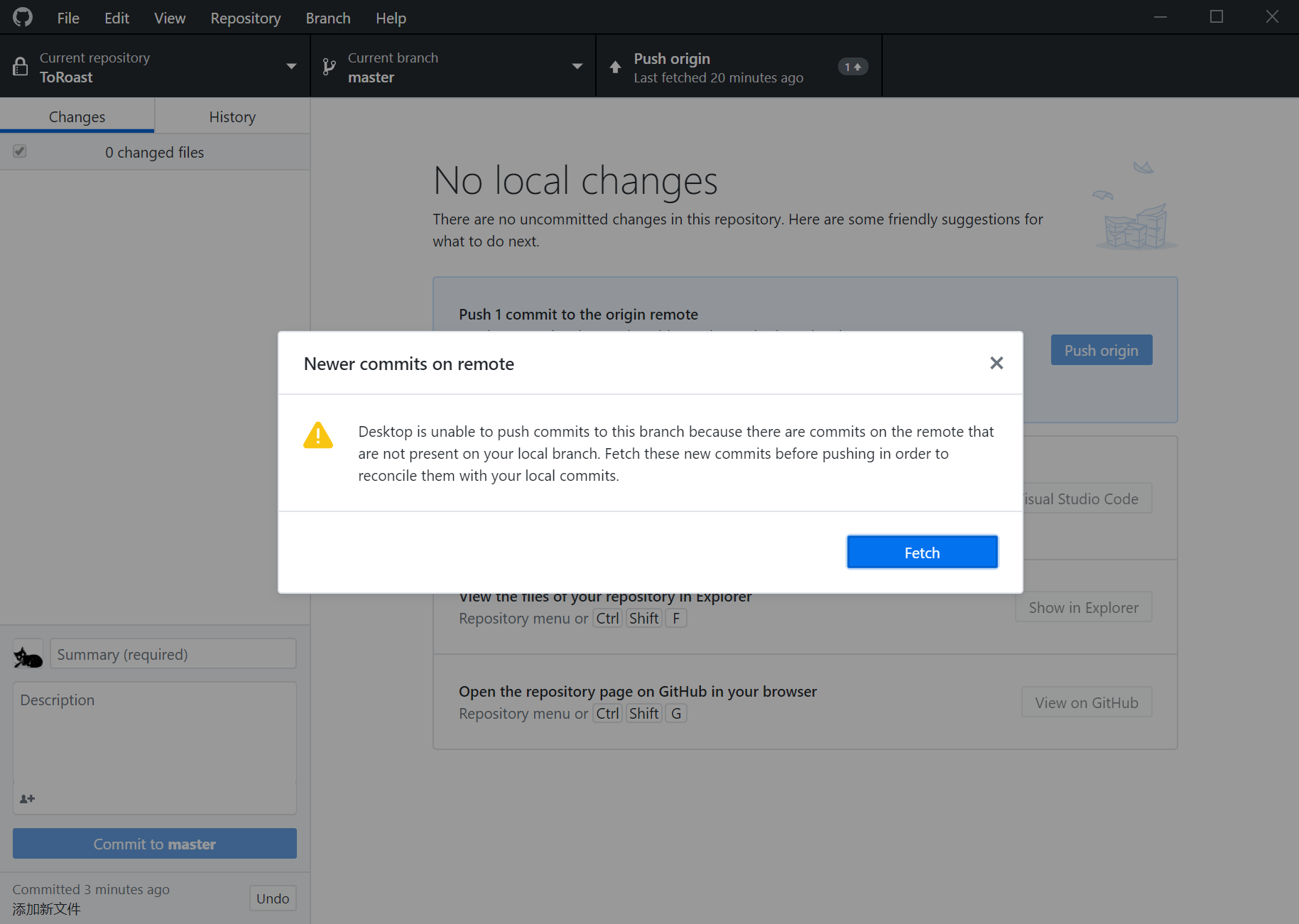
|  |  |  |
| --- | --- | --- |
| 任务编号 | 内容 | 要求 |
| 1 | 注册github，下载安装桌面版，并把用户名发给我 |  |
| 2 | 在群里通知已将各位添加为项目合作者后，从github桌面版里将合作项目ToRoast克隆到本地 |  |
| 3 | 从桌面版里的一个按钮打开项目的github网页版，在网页版里进行一次更新commit | 在网页的项目目录中打开README.md，按照里面的格式，在下面添加上自己的名字并commit  Commit标题应为：添加名字  描述应为：\*\*\*（你的名字）进行了一次网页版commit测试 |
| 4 | 在桌面版github中找到按钮打开本地的项目文件夹，在首目录下添加一文件，进行一次commit | 添加一个空的.txt文件，文件名为\*\*\*（你名字的英文大写首拼）.txt  Commit标题为：添加新文件  描述为：\*\*\*进行了一次桌面版commit测试 |
| 5 | 重新打开项目的网页版，看看4中的commit是否已更新进去，如果没有搞清楚为什么，该怎么解决。（这里体现出网页和桌面的commit不同） | 会做不会做，都记得看一眼下一页的参考答案 |
|  |  |  |

参考答案：本地commit后，还需要需要进行push操作才能将改动更新到云端，如下图：

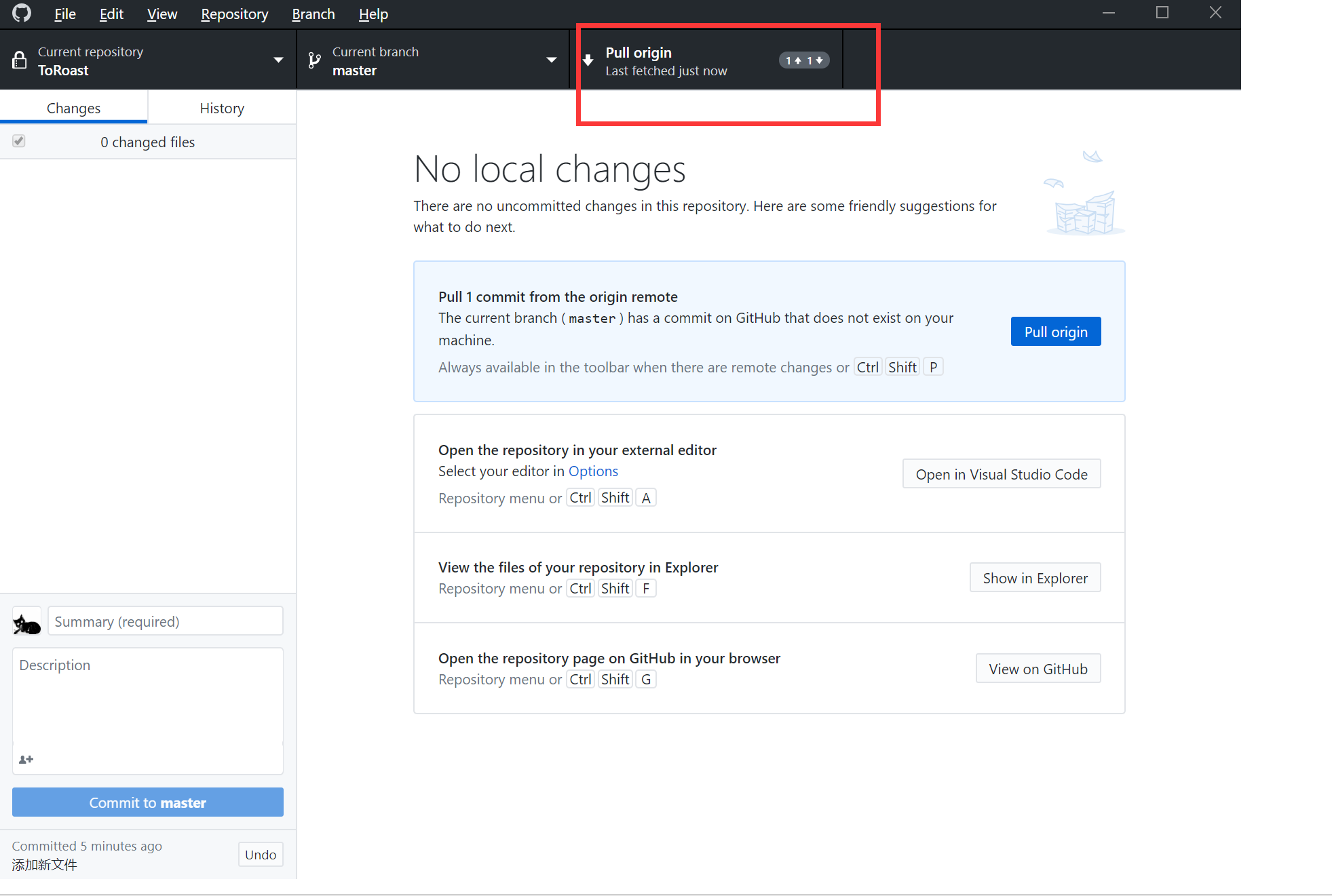


但是……



然后你会发现，push失败，因为在你push之前，云端项目已经与你本地改动前的项目不一致了（因为你刚刚在网页版进行了一次更新）,这时，需要先将最新版的项目fetch到本地除去你的新改动外其他项目内容都是最新的（PS如果碰巧当新版本与你的改动都是同一个文件时，这个时候会冲突，必须要舍去一个，所以要注意尽量避免这种情况），然后才能将你的新改动push上去

所以点击fetch后，再进行一次push



注意这里的图标变成了，代表一次下载和一次上传。

这里就体现了github相比于最无脑的传项目压缩包来合作的方便之处：①每次只上传和下载改动的部分，不用整个文件夹都上传下载，而且它能自动检测出改动了那些部分，不需要人为记忆交流。而且每次改动添加的commit标题就相当与改动的说明，进一步方便了其他人理解你的改动②传项目压缩包的方法，最可怕的是，不小心将某些没有更新的文件覆盖了其他人更新后的文件内容，而这里github能自动检测并提示，如果两个人不是恰好同一个文件进行改动冲突，它会智能的将两人的改动都加进去。如果冲突，它也不会直接覆盖一方，而是先提示，然后让我们确定一下到底采用谁的改动，不至于一声不吭直接覆盖。

③可以在history里非常简单明了地查看每一次更新改动，万一最近一次出了问题可以恢复到任意一次改动之前，而且这个记录也同样不需要重新下载重复的文件，只恢复改动的文件。

总而言之，使用github大大为我们合作开发时候的版本控制问题提供了便利，所以不用不行啊！哈哈！