**Git是什么？**

Git是目前世界上最先进的分布式版本控制系统（没有之一）。

Git有什么特点？简单来说就是：高端大气上档次！

**那什么是版本控制系统？**

如果你用Microsoft Word写过长篇大论，那你一定有这样的经历：

想删除一个段落，又怕将来想恢复找不回来怎么办？有办法，先把当前文件“另存为……”一个新的Word文件，再接着改，改到一定程度，再“另存为……”一个新文件，这样一直改下去，最后你的Word文档变成了这样：



过了一周，你想找回被删除的文字，但是已经记不清删除前保存在哪个文件里了，只好一个一个文件去找，真麻烦。

看着一堆乱七八糟的文件，想保留最新的一个，然后把其他的删掉，又怕哪天会用上，还不敢删，真郁闷。

更要命的是，有些部分需要你的财务同事帮助填写，于是你把文件Copy到U盘里给她（也可能通过Email发送一份给她），然后，你继续修改Word文件。一天后，同事再把Word文件传给你，此时，你必须想想，发给她之后到你收到她的文件期间，你作了哪些改动，得把你的改动和她的部分合并，真困难。

于是你想，如果有一个软件，不但能自动帮我记录每次文件的改动，还可以让同事协作编辑，这样就不用自己管理一堆类似的文件了，也不需要把文件传来传去。如果想查看某次改动，只需要在软件里瞄一眼就可以，岂不是很方便？

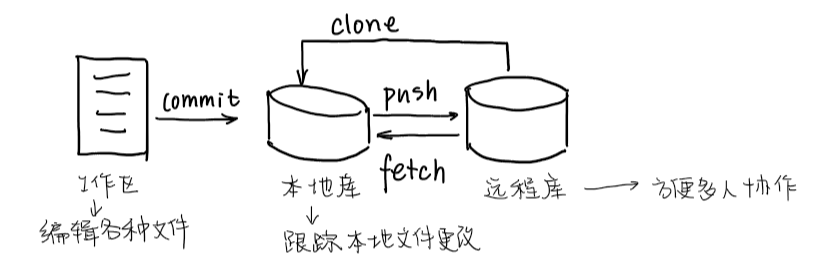
这个软件用起来就应该像这个样子，能记录每次文件的改动：

| **版本** | **文件名** | **用户** | **说明** | **日期** |
| --- | --- | --- | --- | --- |
| 1 | service.doc | 张三 | 删除了软件服务条款5 | 7/12 10:38 |
| 2 | service.doc | 张三 | 增加了License人数限制 | 7/12 18:09 |
| 3 | service.doc | 李四 | 财务部门调整了合同金额 | 7/13 9:51 |
| 4 | service.doc | 张三 | 延长了免费升级周期 | 7/14 15:17 |

这样，你就结束了手动管理多个“版本”的史前时代，进入到版本控制的20世纪。

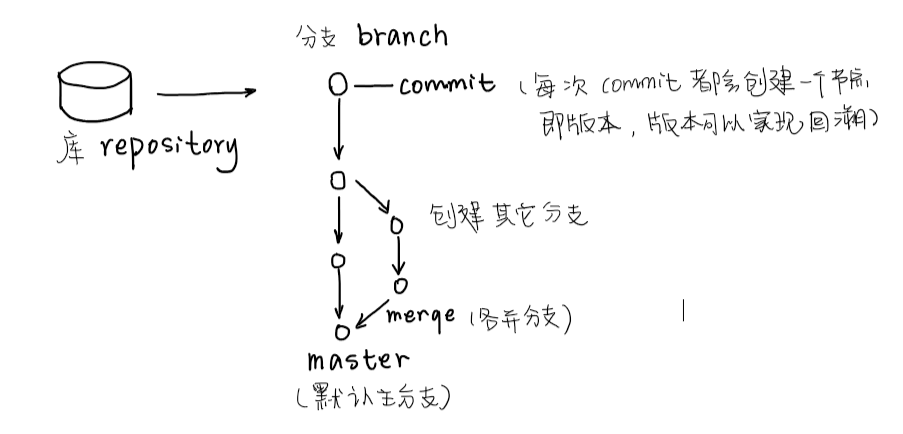
以上摘自廖雪峰的git教程

**Git如何工作？**（个人理解）



远程库和本地库结构完全相同

**Git的特色Branch**



分支就是科幻电影里面的平行宇宙，当你正在电脑前努力学习Git的时候，另一个你正在另一个平行宇宙里努力学习SVN。

如果两个平行宇宙互不干扰，那对现在的你也没啥影响。不过，在某个时间点，两个平行宇宙合并了，结果，你既学会了Git又学会了SVN！

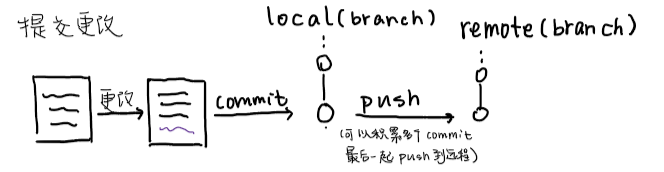


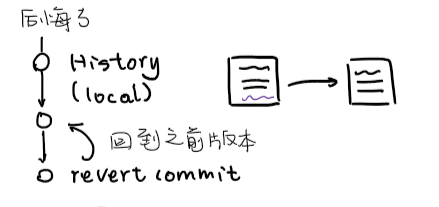
分支在实际中有什么用呢？假设你准备开发一个新功能，但是需要两周才能完成，第一周你写了50%的代码，如果立刻提交，由于代码还没写完，不完整的代码库会导致别人不能干活了。如果等代码全部写完再一次提交，又存在丢失每天进度的巨大风险。

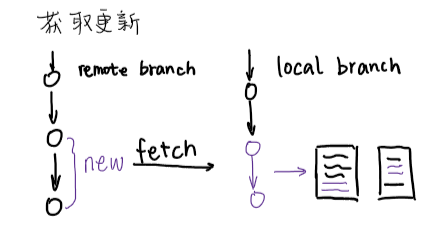
现在有了分支，就不用怕了。你创建了一个属于你自己的分支，别人看不到，还继续在原来的分支上正常工作，而你在自己的分支上干活，想提交就提交，直到开发完毕后，再一次性合并到原来的分支上，这样，既安全，又不影响别人工作。

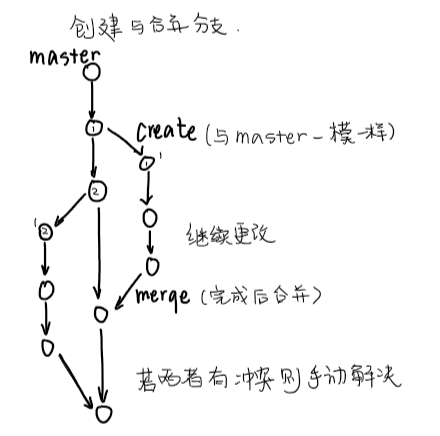
以上摘自廖雪峰的git教程

**Git的功能**









**Git的应用**

可以进行代码的跟踪管理，以及其他文件的版本控制。但只能对文本文件进行真正意义的追踪（如文件内部内容的变化，字符增减等都可以比较出来），二进制文件只能记录变化。

**本小组用git进行文档管理和代码控制**

**GitHub——提供远程仓库服务器的网站**

可以方便的进行版本控制而不用专门学习git

1. 注册github账号

https://github.com/ 这是GitHub的官方网站，在官网上可以注册属于自己的GitHub账号，网上是全英文的，对于英语不好的同学建议使用谷歌浏览器，谷歌浏览器可以翻译网页变为中文使用起来十分方便。



通过简单的步骤之后你就会有一个属于自己的GitHub账号。再简单注册完成之后会需要验证你所输入的邮箱才能正常使用你的GitHub。

---------------------

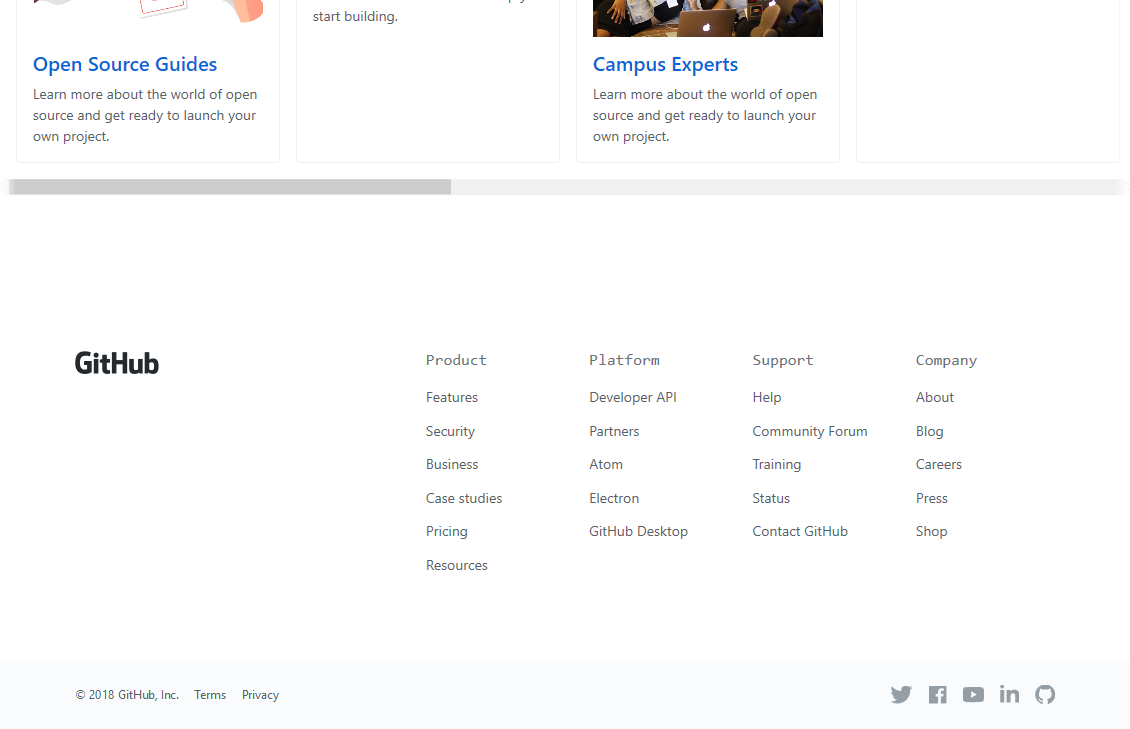
作者：Hanani\_Jia

来源：CSDN

1. 下载GitHub desktop

滑到官网最底部，platform里选择GitHub desktop

若没有则点击explore github

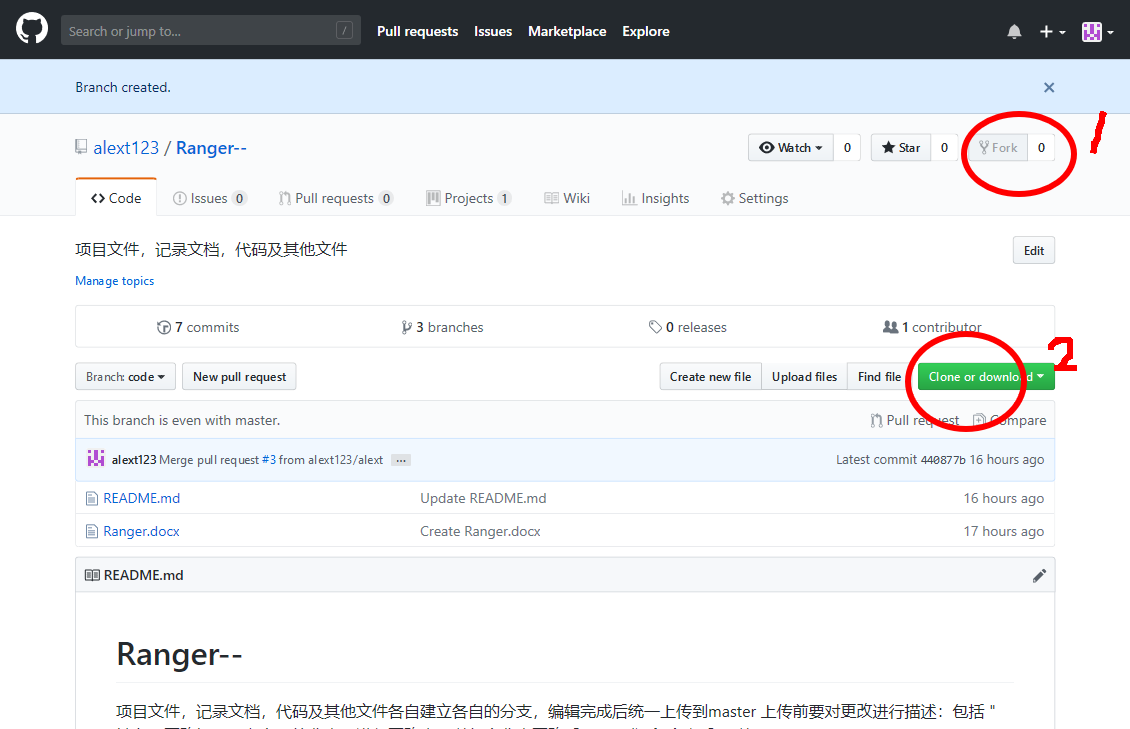


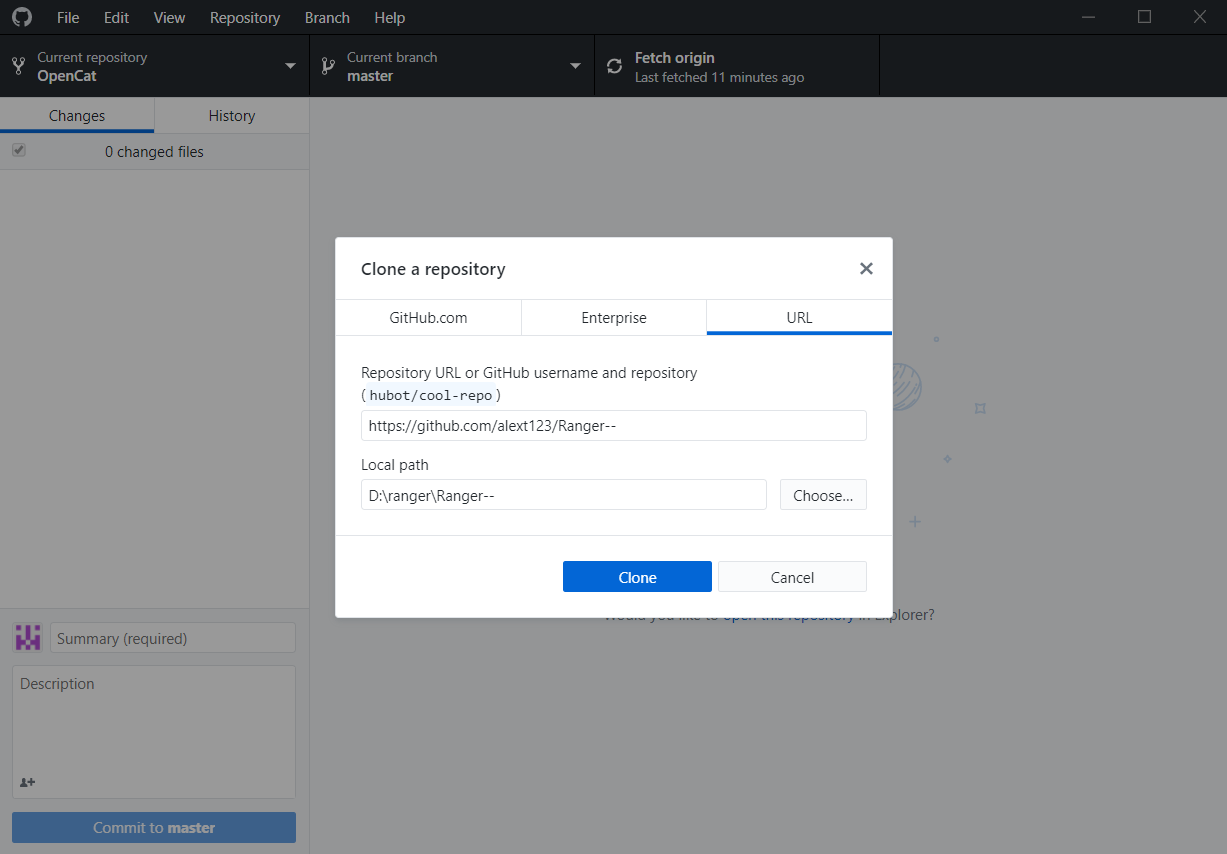
**Fork&clone**

找到我的库，名字：alext123/ranger

点击右上角的fork，将ranger复制到自己的账户里

然后点击clone，将远程库克隆到本地（一定要先fork再clone，否则不能成功提交）





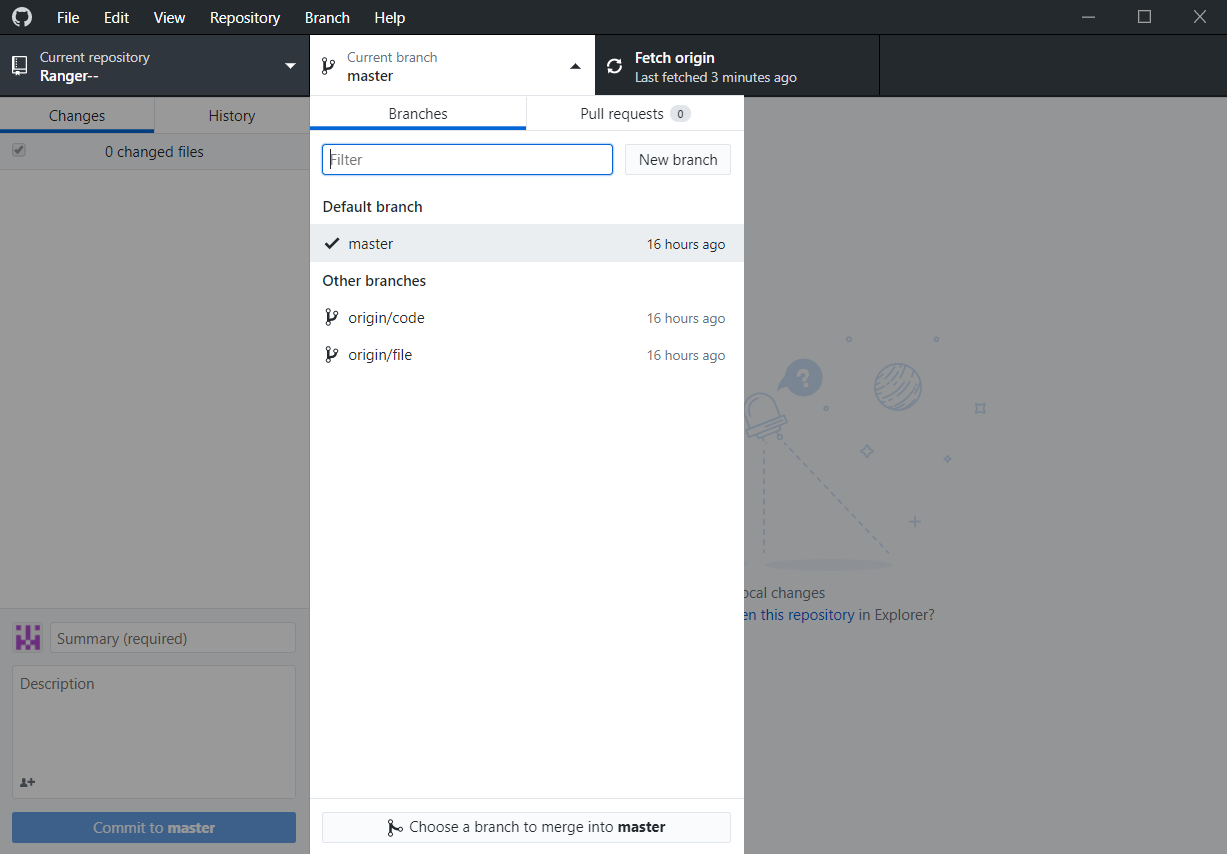
会在本地得到一模一样的库，我已经创建三个分支master，file，code

Master一般不做修改，file用来更改文档，code用来管控代码

**更改提交及上传**

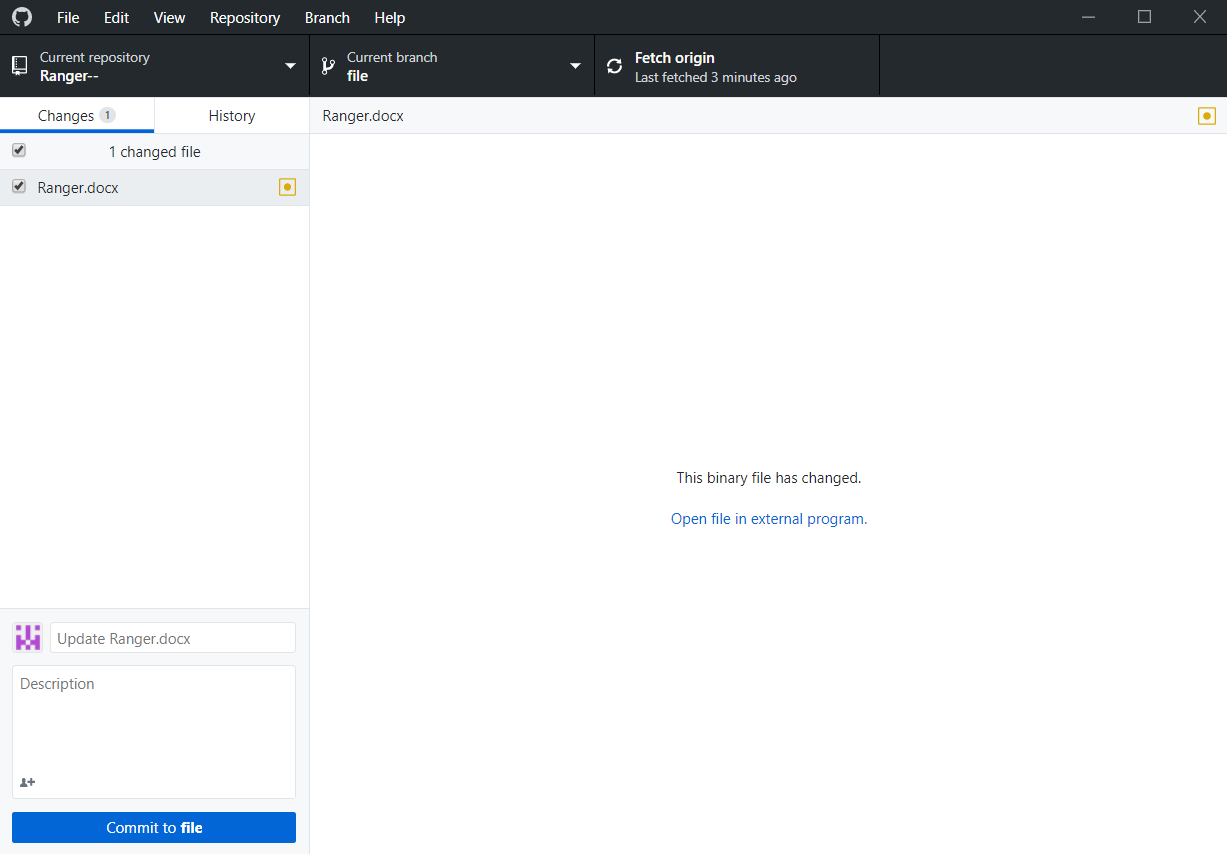
点击open this repository in explorer 可在文件夹中打开库。

编辑文件和平时没有不同，只是需要保存在库所在的文件夹里才能进行管控。



更改文件前将分支切换为file（分支一定要及时切换）

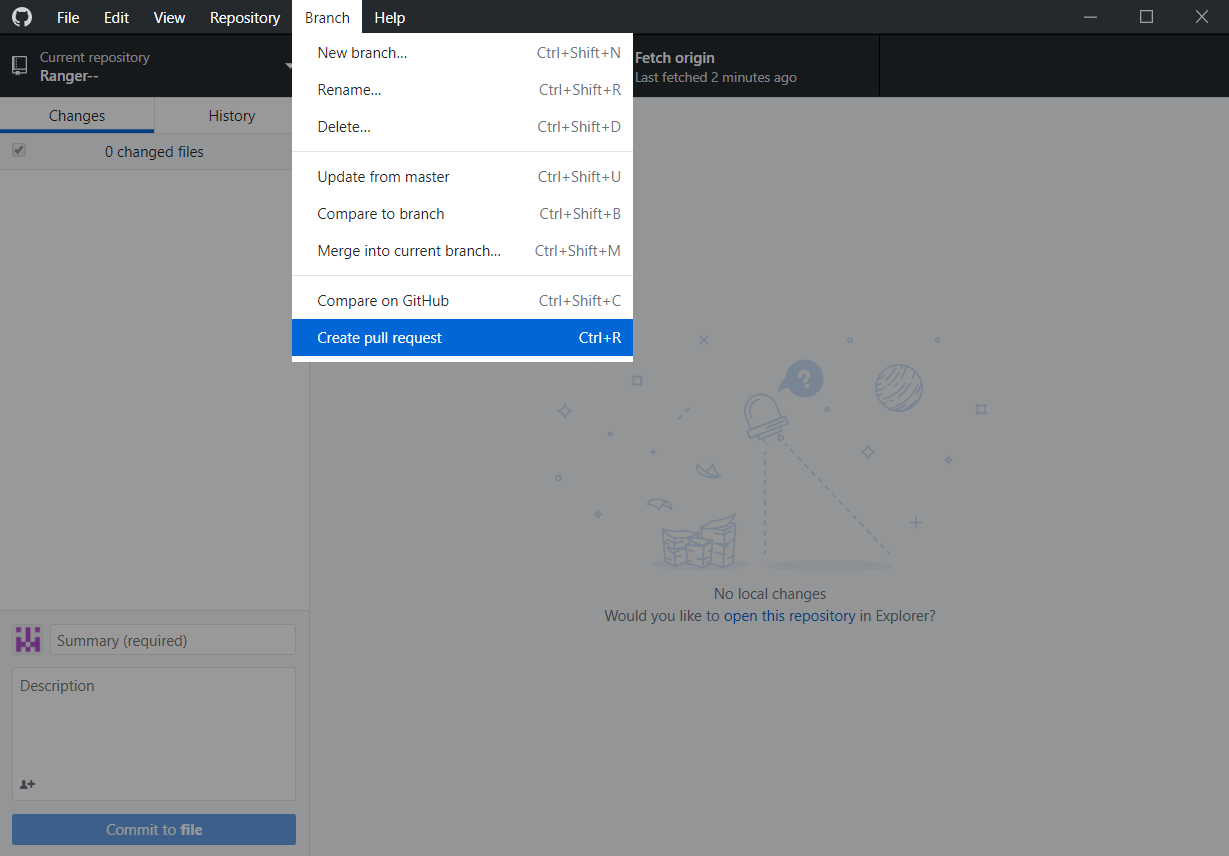
更改后会显示有change，此时点commit to file （提交到本地库file分支），然后右上角会变为push to origin，点击（提交到远程库file branch）

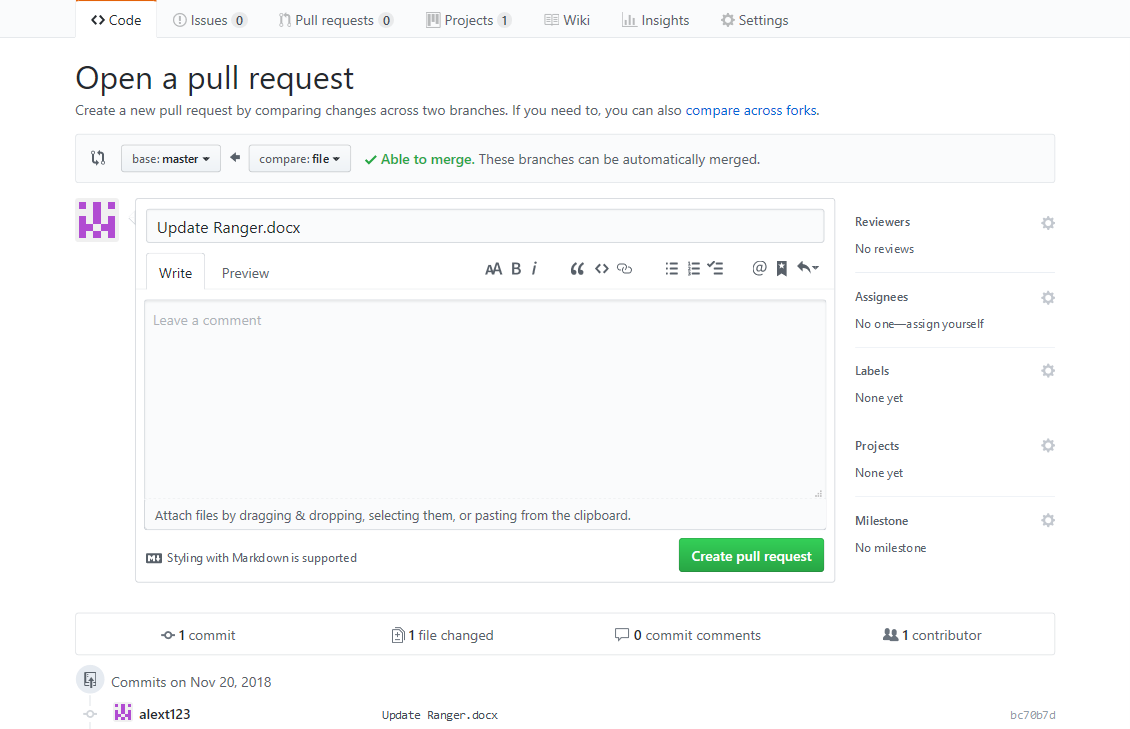


多人协作

以上操作均在自己的库中进行，小组合作要将更改提交到我的库中（alext123/ranger），此时需要pull request

选择将自己库中的file提交到（alext123/ranger file），表明更改内容，如Update Ranger.docx，然后create pull request，之后我会看到如果没有问题就会同意merge





其他：

File中已有文件：ranger记录文件，项目文件，采购文件，readme

参考上述方法可以将opencat的库fork到自己账户学习

大家每周都要pull一次，提交自己的进度，就不用在群里发文件了，平时更改在自己库里进行就行

自己编辑的时候也可以创建分支，自己研究一下很容易的，参考分支管理策略