**SourceTree 的使用**

**一、Git的概念和原理**

1、Git概念

git是分布式版本控制系统，代码提交是在本地的（如此速度就快），当然生成补丁（patch）然后push到远程服务器上是需要联网的。

2、区别

CVS、SVN是集中式版本控制系统，代码提交都是提交到远程服务器上，是需要一直联网的（如此速度就慢）（这里的一直联网不是说你写代码的时候要联网，而是说你提交代码的时候必须联网；但是git不同，git提交代码是本地的不需要联网，生成patch后push patch才需要联网，相当于svn的远程的集中服务器对于git来说，这个集中的远程服务器就在本地）。这个地方比较难理解。

CVS、SVN这样的集中式版本控制系统，它的完整代码仓库（代码仓库不仅仅只包含了代码，还包含各个历史版本的信息等）在中心服务器上，一旦这个中心服务器挂了，也就是完整的代码仓库挂了，虽然你本地可能之前有从中心服务器上取得过代码，但是那些历史版本信息都没有了，而且你也无法再提交代码。

git不同，git没有中心服务器的概念，每一个git客户端（git节点）都含有一个完整的代码仓库（前提是你之前从远程git仓库fetch过代码），所以那些历史版本信息都在你本机上，假如哪一个git节点挂掉了，随意从其他git节点clone一个代码仓库过来就ok了， 那些原来的代码、版本信息之类的都还是完整的（当然如果你在这个挂掉的git节点上新增的代码是没有掉了的）。

综上，git的每一个节点（第一次从远程git仓库取得代码后，该git节点就是一个完整的代码仓库）相当于SVN的中心服务器，都包含完整的代码仓库。

3、Git原理

1）、如下图所示，版本的每一次提交（commit），git都将它们根据提交的时间点串联成一条线。刚开始是只有一条时间线，即master分支，HEAD指向的是当前分支的当前版本。



2）、当创建了新分支，比如dev分支（通过命令git branch dev完成），git新建一个指针dev，dev=master，dev指向master指向的版本，然后切换到dev分支（通过命令git checkout dev完成），把HEAD指针指向dev，如下图。



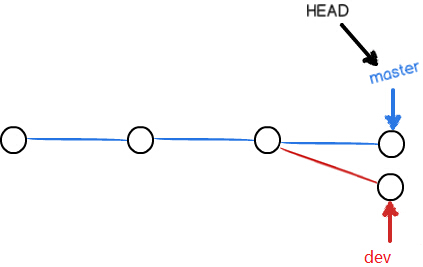
3）、在dev分支上编码开发时，都是在dev上进行指针移动，比如在dev分支上commit一次，dev指针往前移动一步，但是master指针没有变，如下：



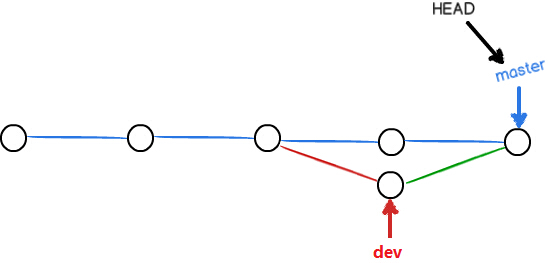
4)、当我们完成了dev分支上的工作，要进行分支合并，把dev分支的内容合并到master分支上（通过首先切换到master分支，git branch master，然后合并git merge dev命令完成）。其内部的原理，其实就是先把HEAD指针指向master，再把master指针指向现在的dev指针指向的内容。如下图。



5)、当合并分支的时候出现冲突（confict），比如在dev分支上commit了一个文件file1，同时在master分支上也提交了该文件file1，修改的地方不同（比如都修改了同一个语句），那么合并的时候就有可能出现冲突，如下图所示。



这时候执行git merge dev命令，git会默认执行合并，但是要手动解决下冲突，然后在master上git add并且git commit，现在git分支的结构如下图。



可以使用如下命令查看分支合并情况。

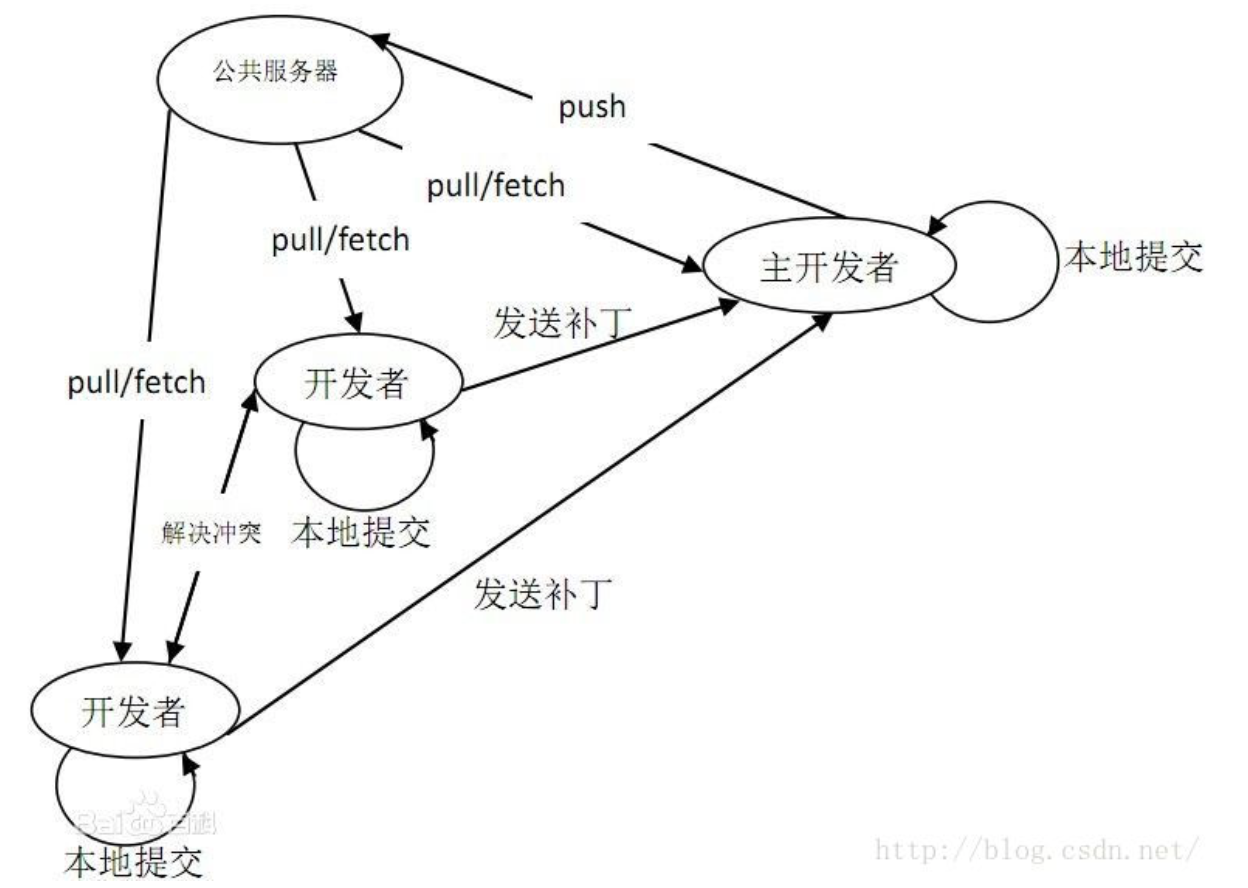
**[cpp]** view plain copy

1. git log --graph --pretty=oneline --abbrev-commit

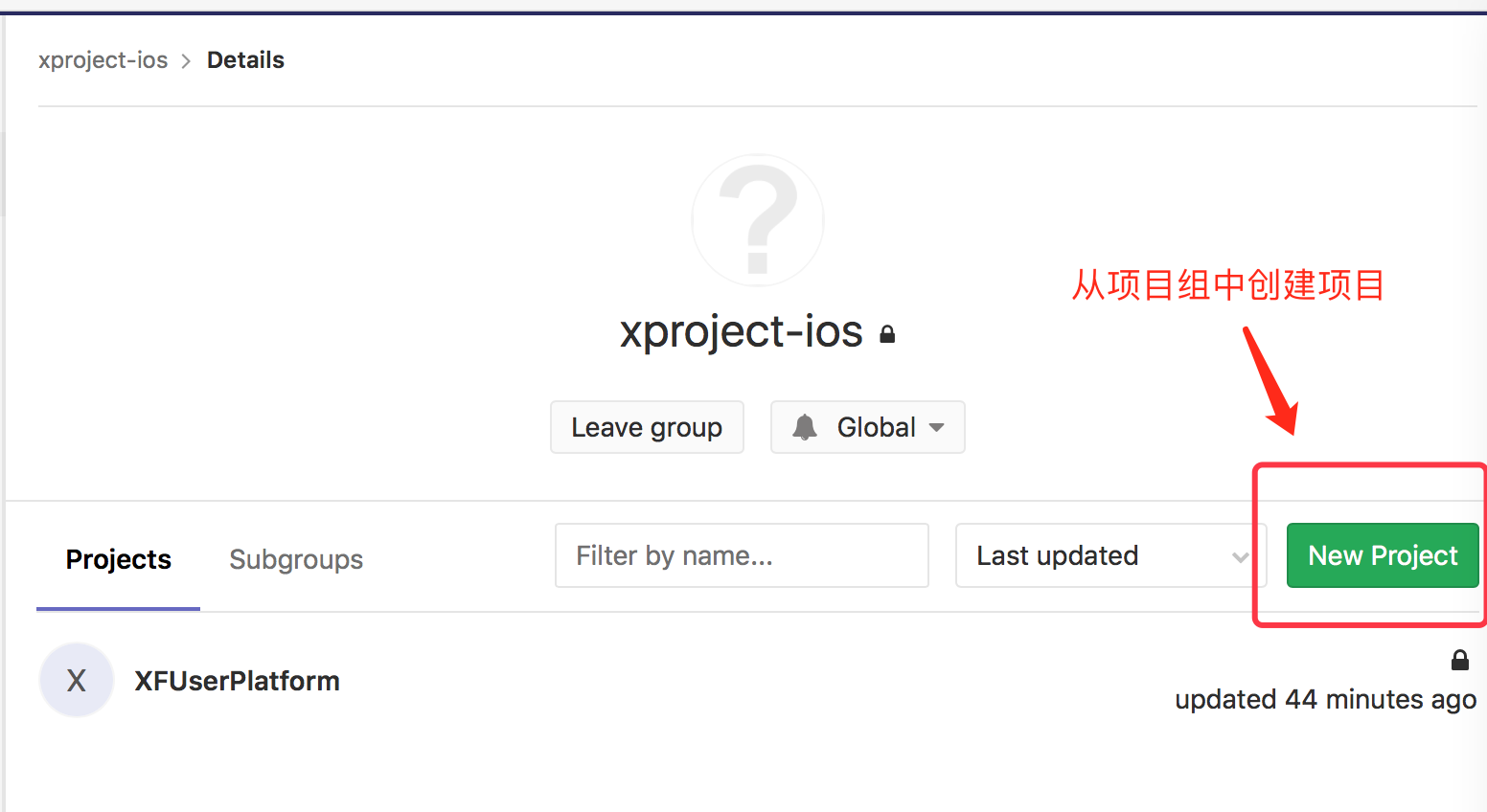
6)、合并完成后，就可以删除掉dev分支（通过git branch -d dev命令完成）。



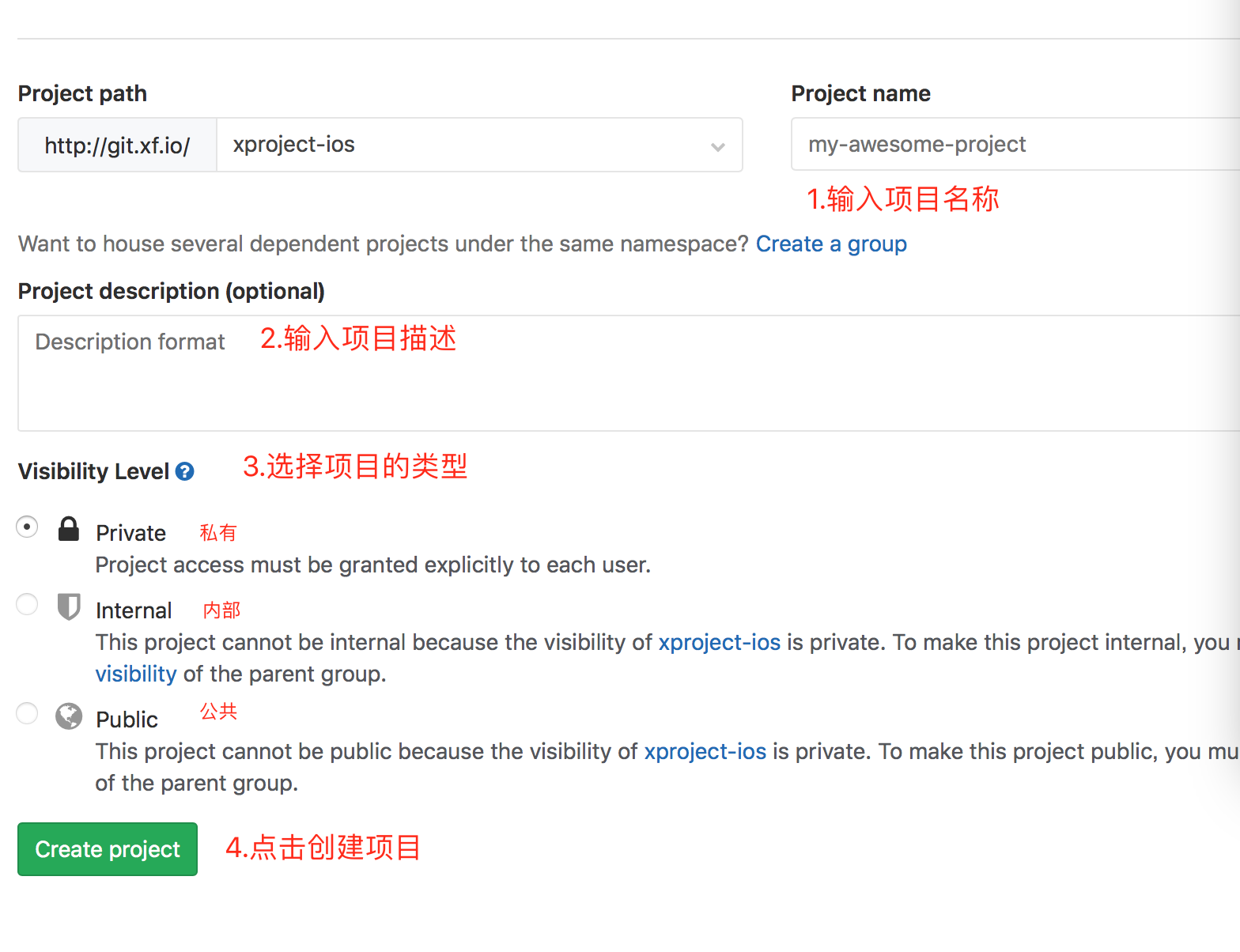
**二、流程图**

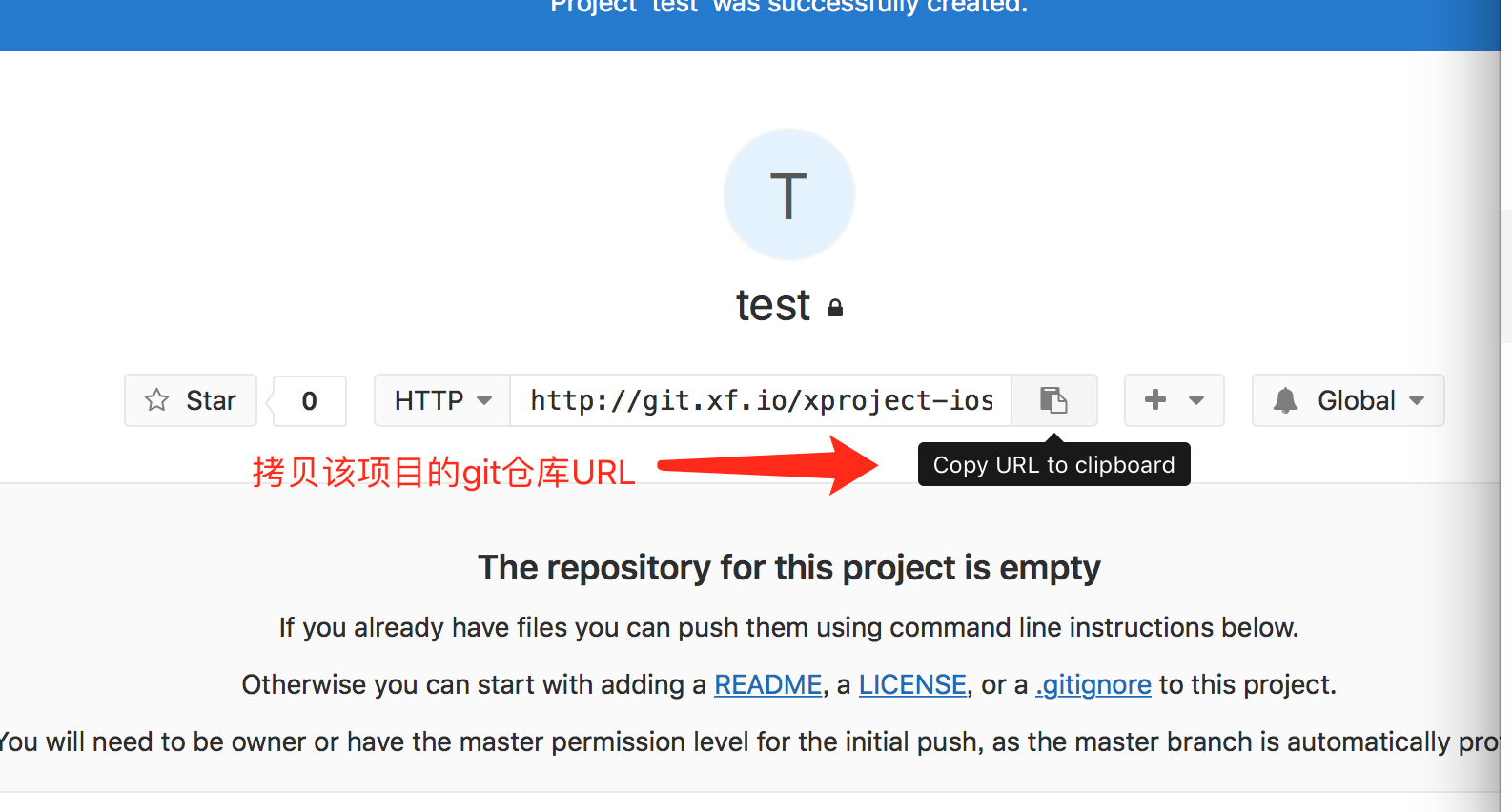
****

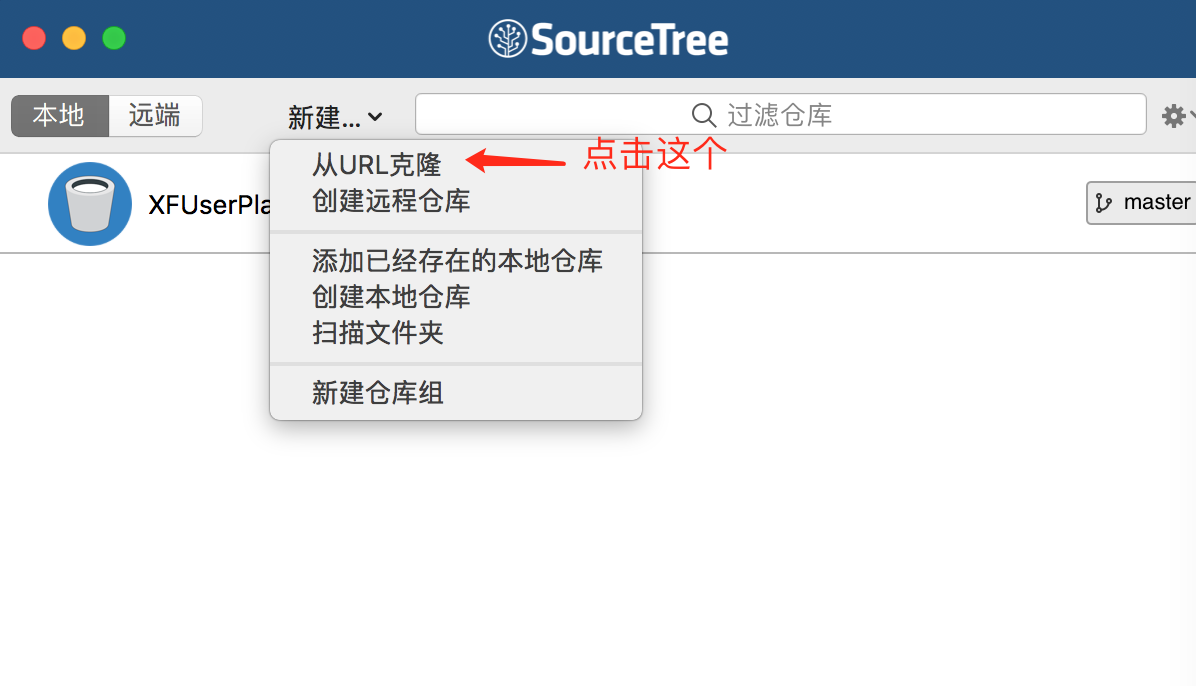
**三、具体流程**

1、

2、

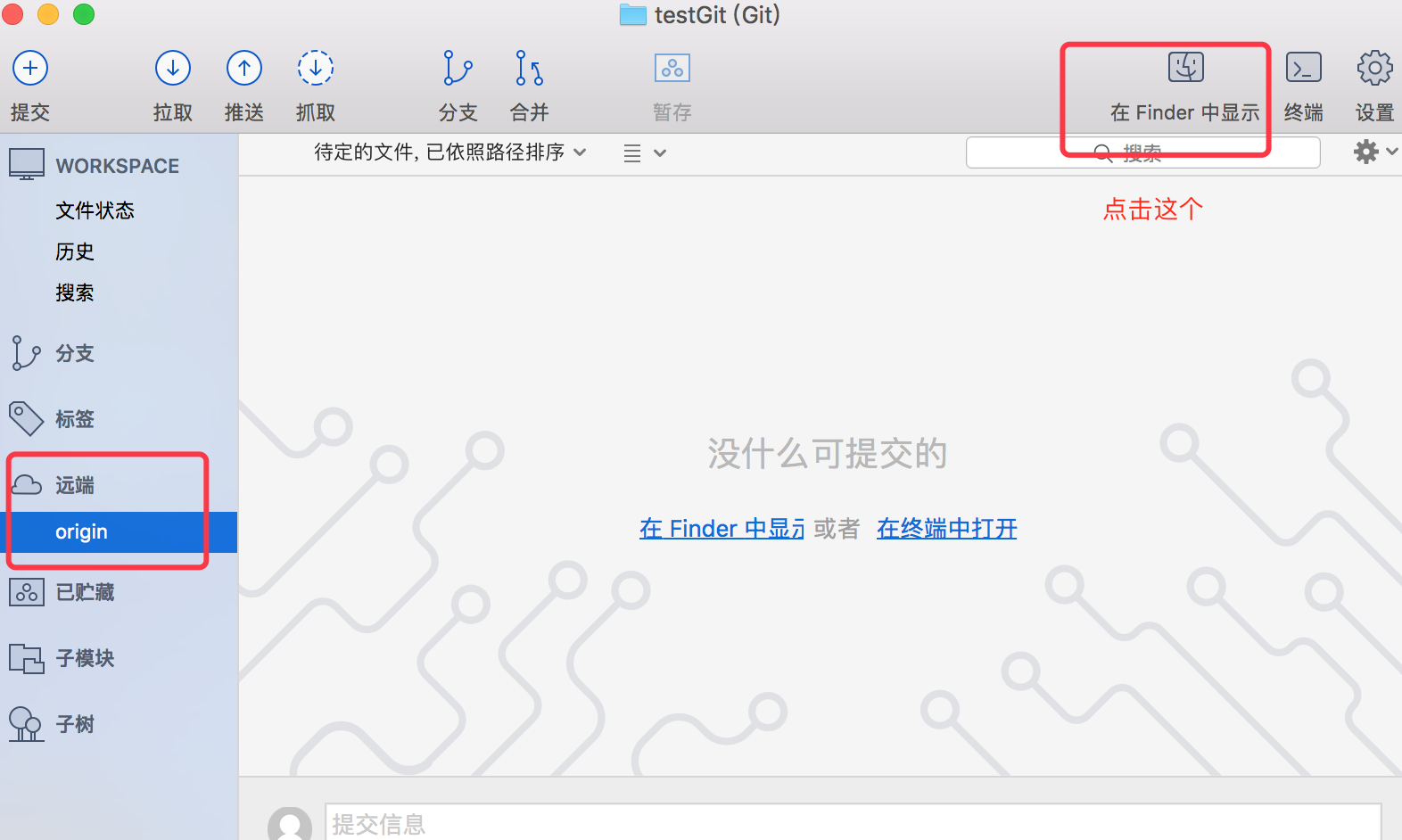


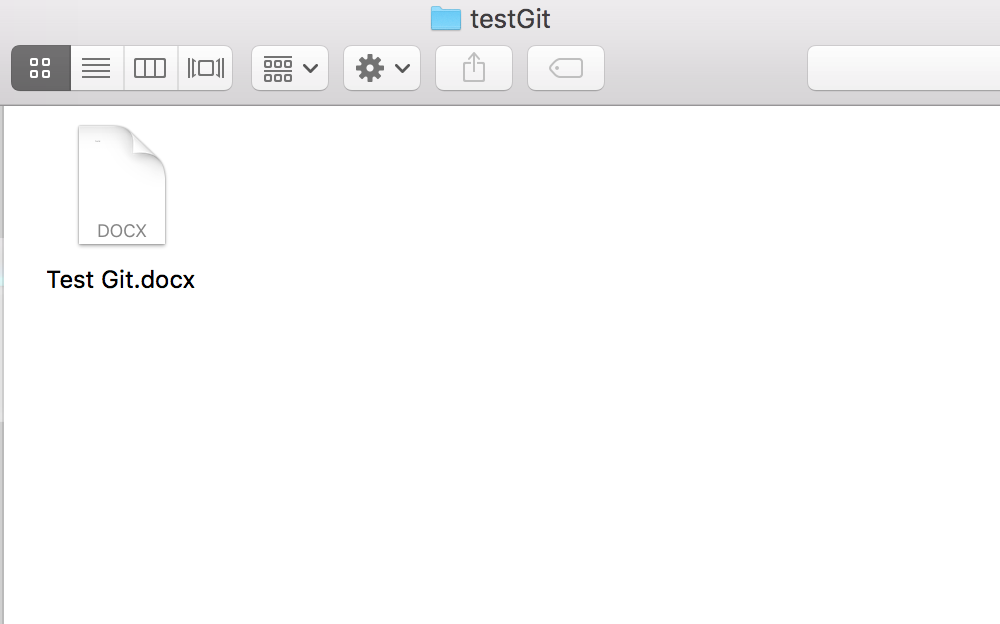
3、

4、

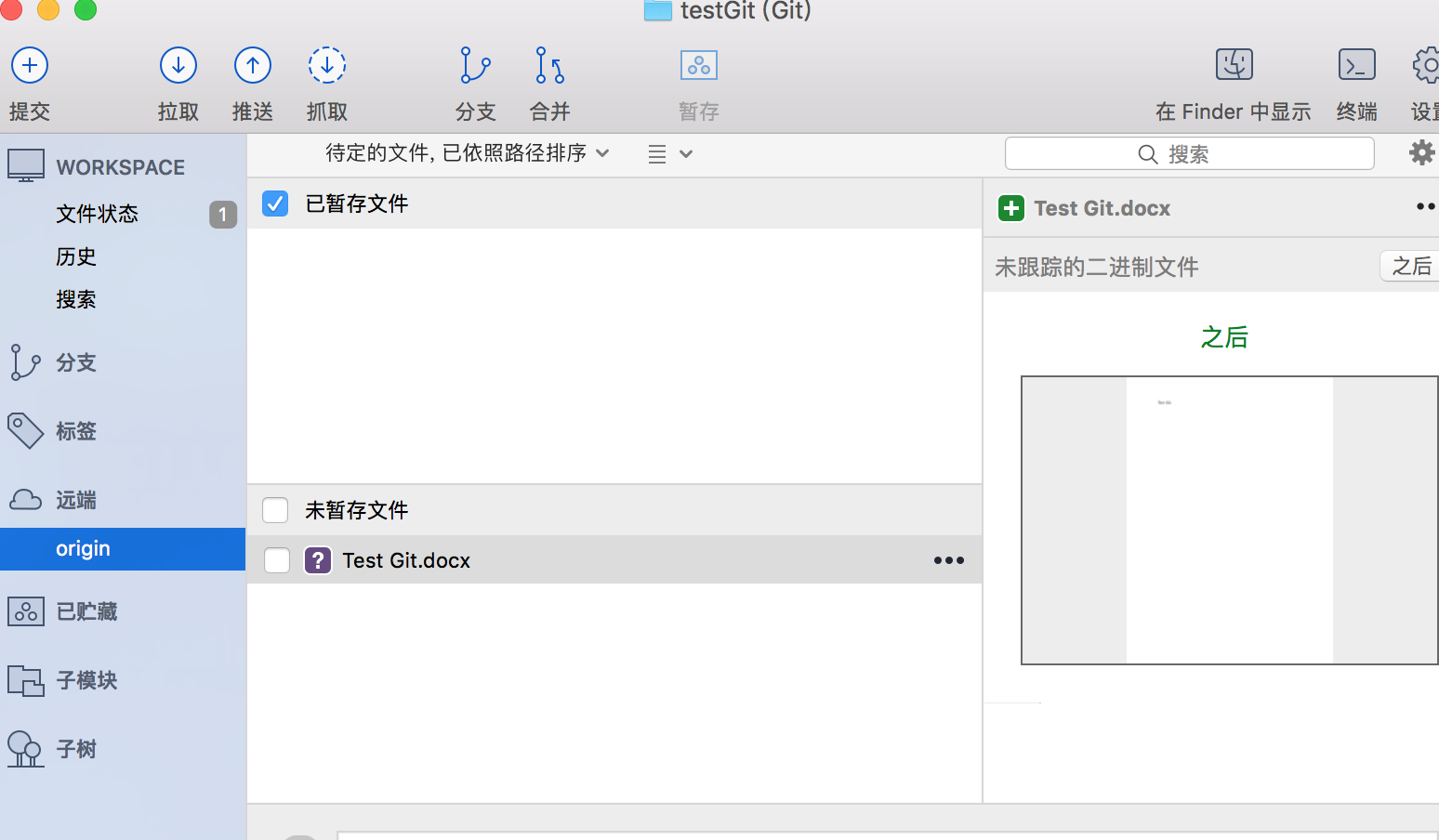
5、

6、

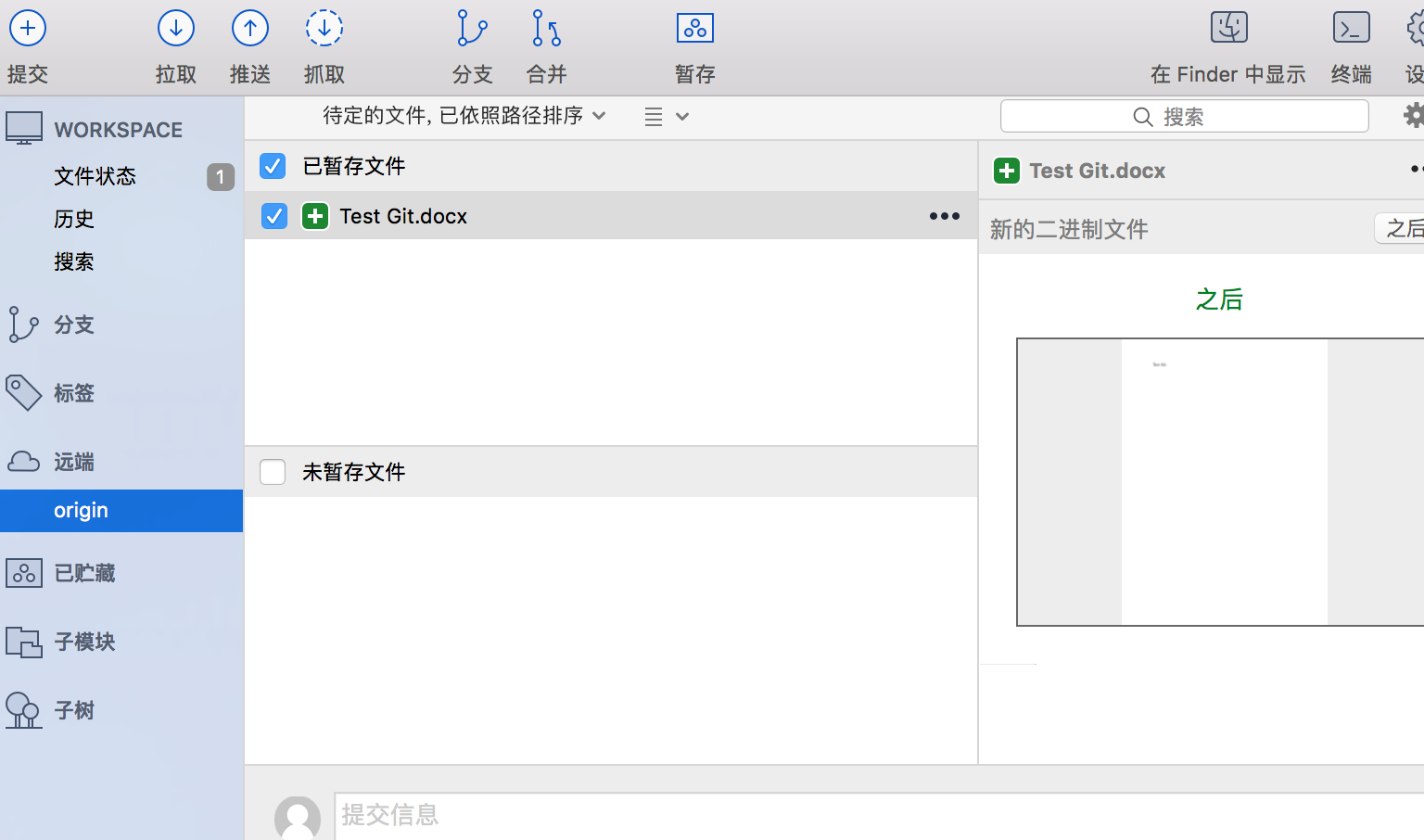
7、

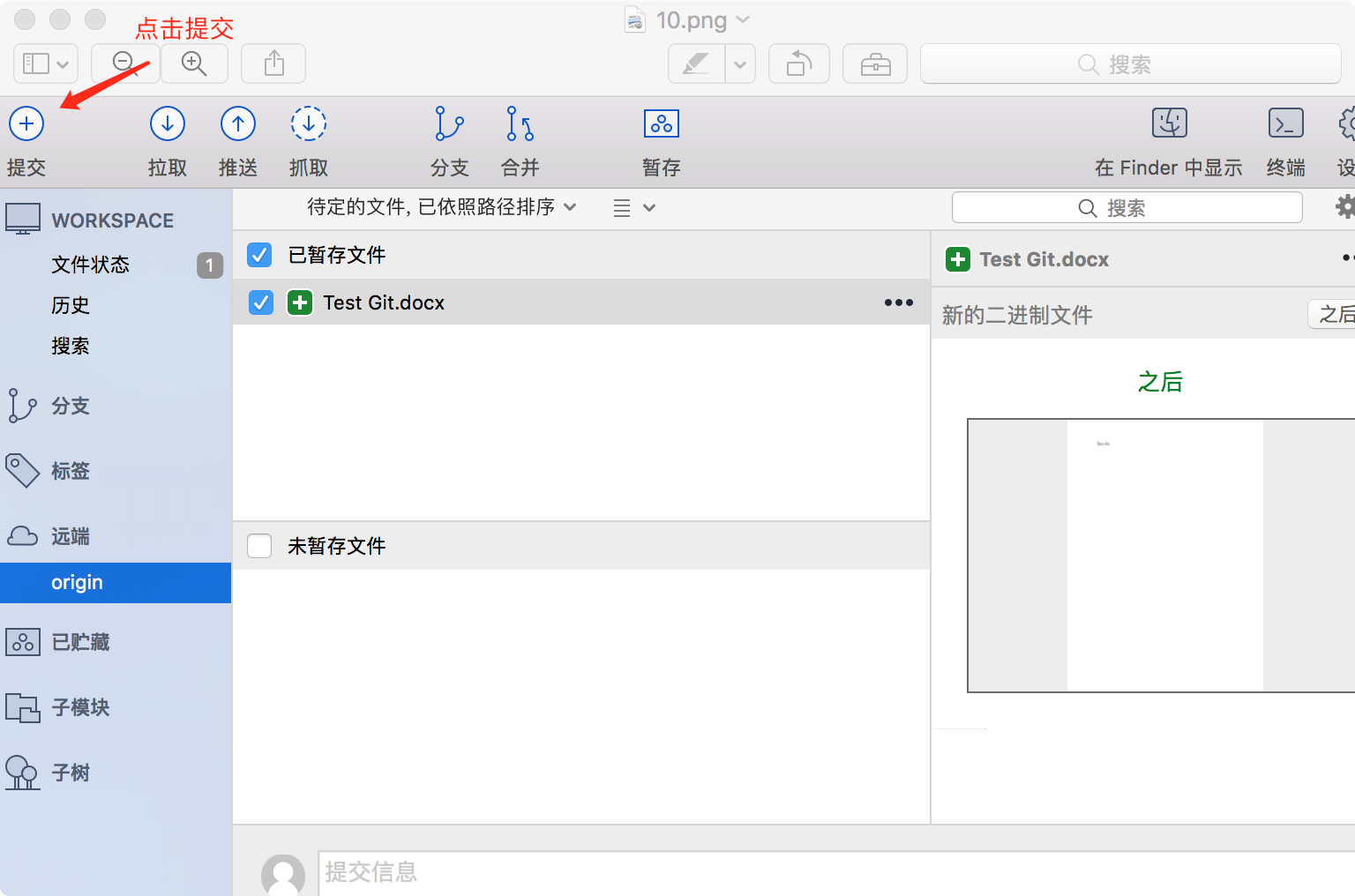
8、放你具体的项目文件进去

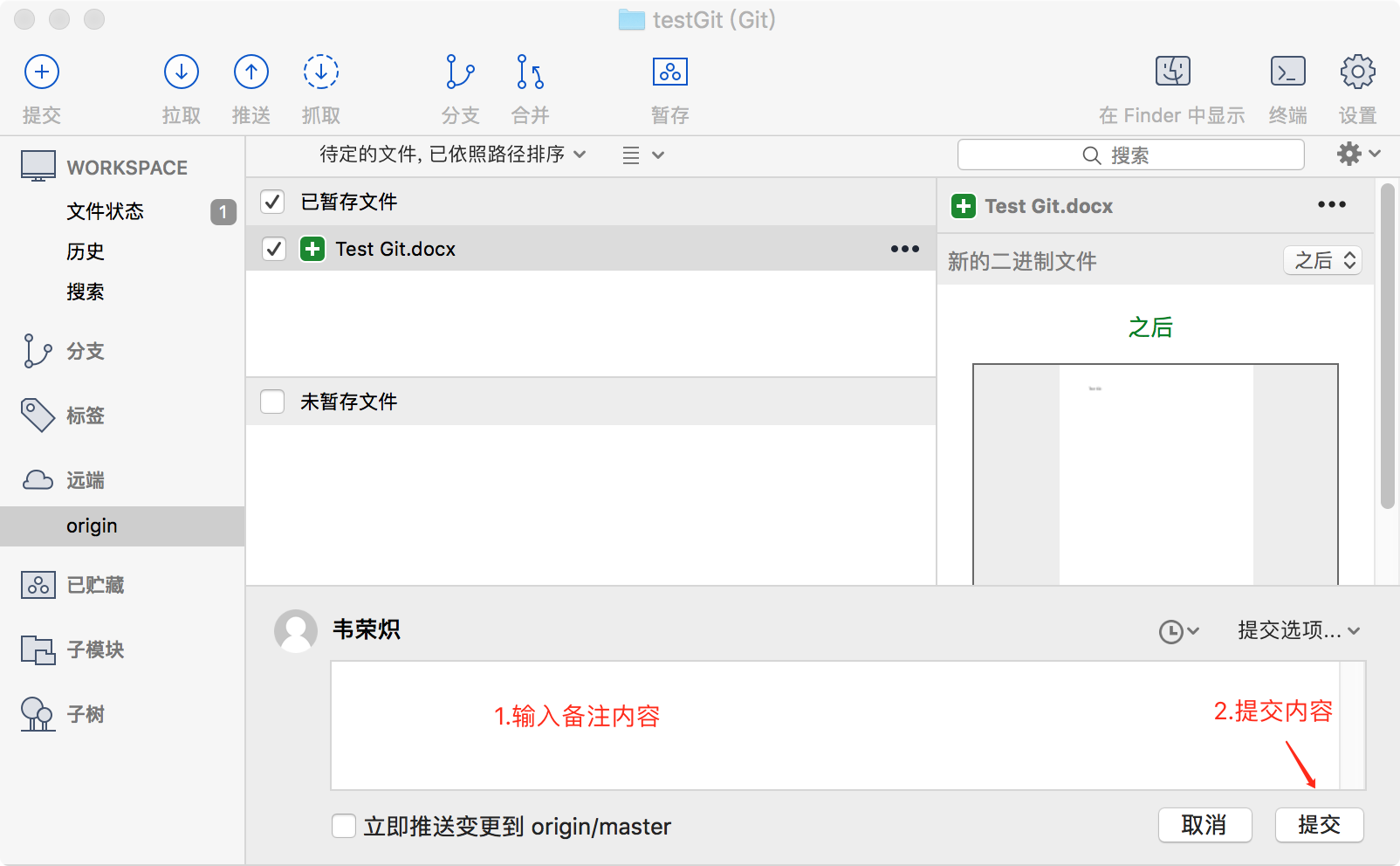
9、添加了之后，在SourceTree的界面中会出现如下的状况



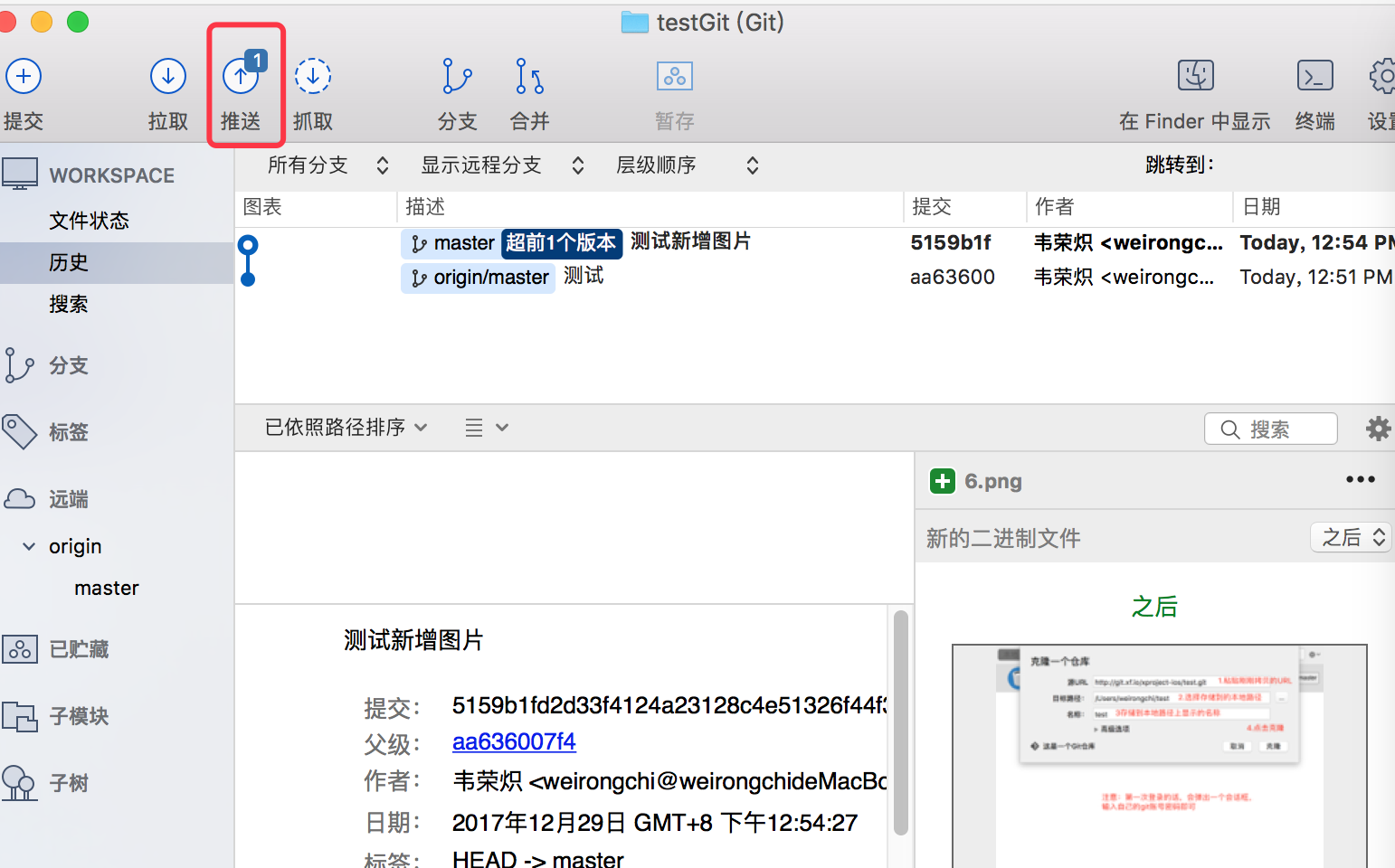
10、选中文件到暂存

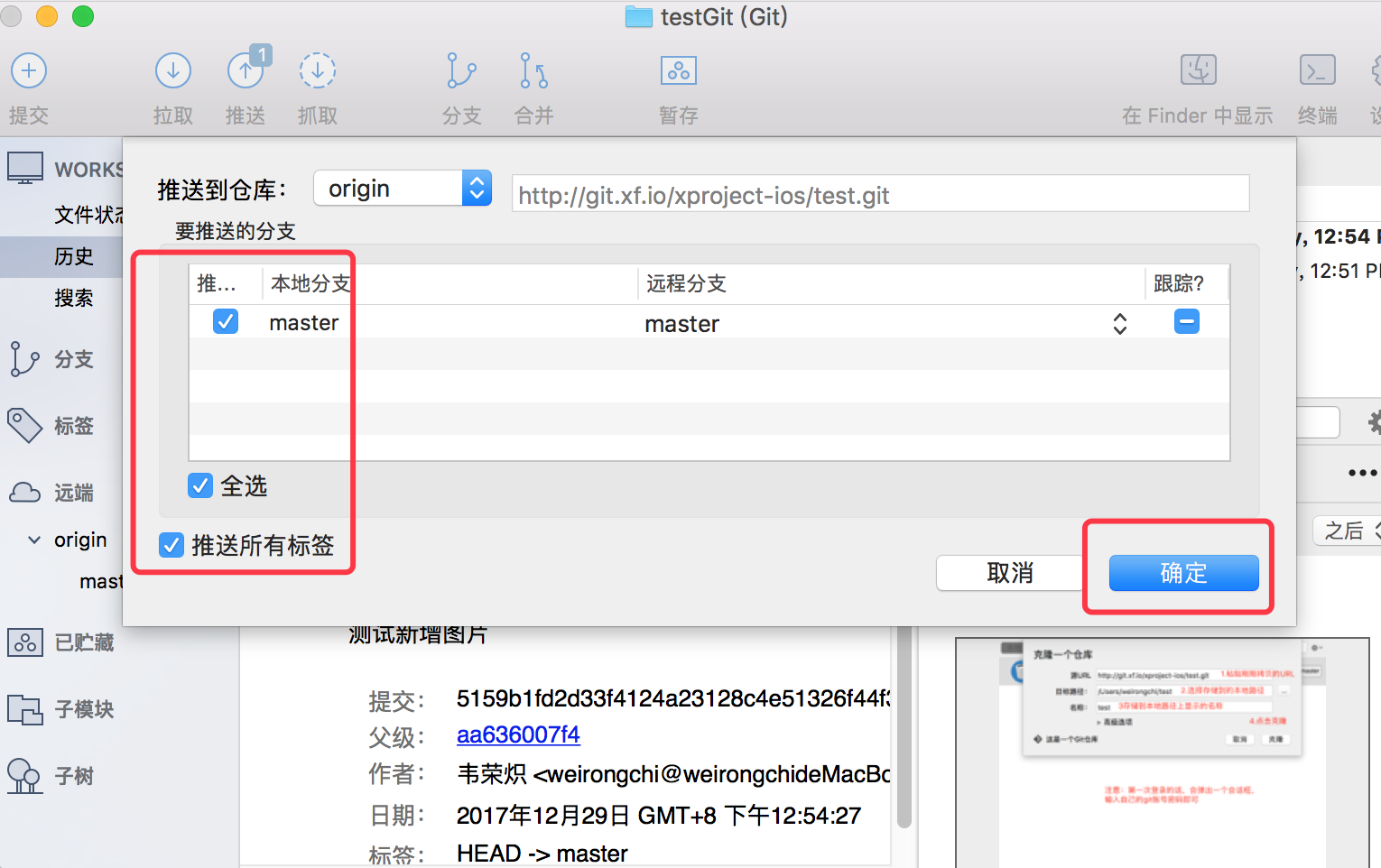


11、点击提交

12、输入备注然后提交

13、然后点击推送按钮



14、选中对应的文件，然后确定，就会将文件发送到远程仓库

相关的关键词说明

1.提交:指本地仓库的文件提交到本地仓库保存

2.拉取:从关联账号的托管平台(比如github)的众多远程仓库中下载一个仓库到本地,前提是你得账号里创建的有仓库

3.推送:将本地仓库源文件提交到本地并更新到托管平台的远程仓库中,即本地和原唱仓库都保存此时仓库中的所有内容(如果不配置ignore忽略文件的话)

4.抓取:从远程仓库抓取本仓库的更新内容(比如这个项目是团队开发,别人也可能代码到这个仓库,这时候你可能就需要更新一下)并更新到本地仓库

5.分支:分支本质上其实就是一个指向某次提交的可变指针HEAD,指向当前默认分支master或者上一分支.我们之所以需要创建分支分两种情况:

1）一种是当团队开发时,为了各自干自己的活不相互影响,这时候每个哥们儿都可以从主分支上建立一个属于自己的分支出来,并在自己的分支开发,当开发完毕时,再合并到主分支即可

2）另一种情况是个人开发,你需要集成某个功能但你又不确定自己是否可以集成成功且不影响源码,这时可以建立新分支,然后再开发,如果集成失败了,又把分支源码搞的乱七八糟,你还可以直接回滚到主分支,重新再来

6.合并:这个当然是在有分支的前提下进行了,只有创建了分支,在分支上开发完毕后,就可以把当前分支合并到主分支或上一分支了

7.暂存:好像是吧,这个英语怎么写我JB也不会.你可以理解为暂时保存,即当你修改项目时,SourceTree会检测出你哪些文件更改了,并显示到未暂存文件列表中,为什么不直接显示在已暂存文件中呢?这就好比你编辑一个word文档,在你写的时候是属于未暂存的word文件,当你不点保存就关闭word时,它会提醒你要不要保存,你点保存才会成为已暂存文件

对应的命令行图