


1. 创建本地版本库
 - 1.1 创建本地版本库
 - 1.2 工作区和暂存区
 - 1.3 添加新文件到暂存区并提交
2. 修改文件
 - 2.1 修改工作区文件
 - 2.2 还原修改
 - 2.3 查看修改历史
 - 2.4 差异比较
 - 2.5 删除文件
3. 案例: 添加一个本地项目到仓库
4. 添加远程仓库
 - 4.1 远程仓库的添加和创建
 - 4.2 本地仓库同步到远程仓库
 - 4.3 克隆远程仓库到本地
 - 4.4 ssh 配置
5. 管理分支
 - 5.1 创建分支
 - 5.2 切换分支
 - 5.3 合并分支
 - 5.4 解决冲突
6. 推送文件

1. 创建本地版本库

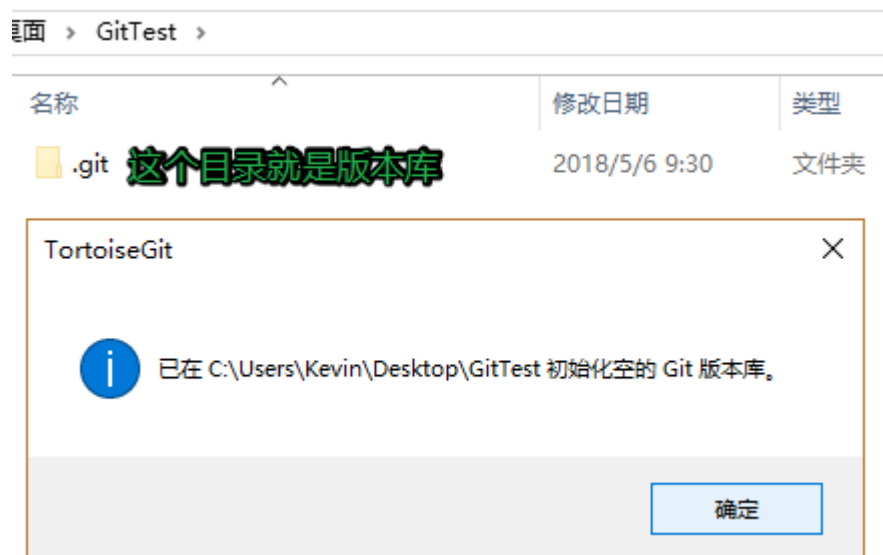
1.1 创建本地版本库

本地版本库就是一个叫 .git 的隐藏目录

1. 创建工作目录
2. 创建本地版本库
 - 需要在对应的工作目录中创建版本库

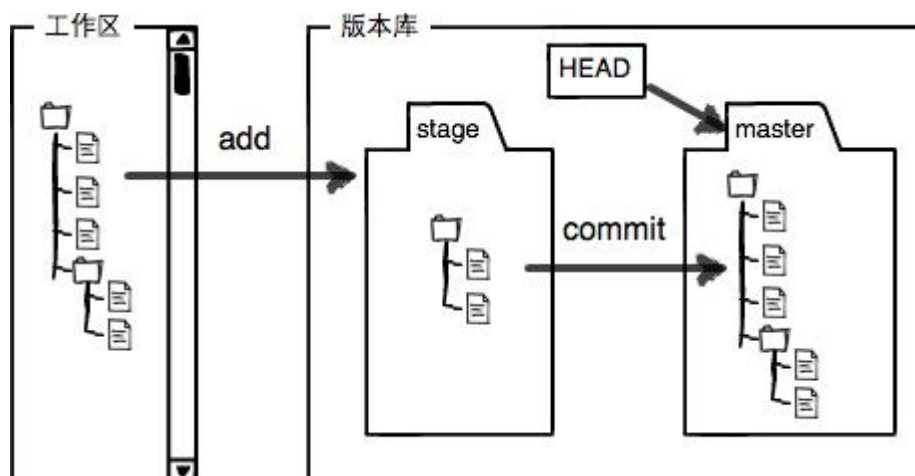
 Git 在这里创建版本库(Y)...





1.2 工作区和暂存区

在初始化git版本库之后会生成一个**隐藏的文件 .git**，可以将该文件理解为 git 的版本库 repository，而我们自己建立的项目文件夹即工作区 working directory，在 .git 文件夹里面还有很多文件，其中有一个 index 文件 就是暂存区也可以叫做 stage，git 还为我们自动生成了一个分支 master 以及指向该分支的指针 head，如下图

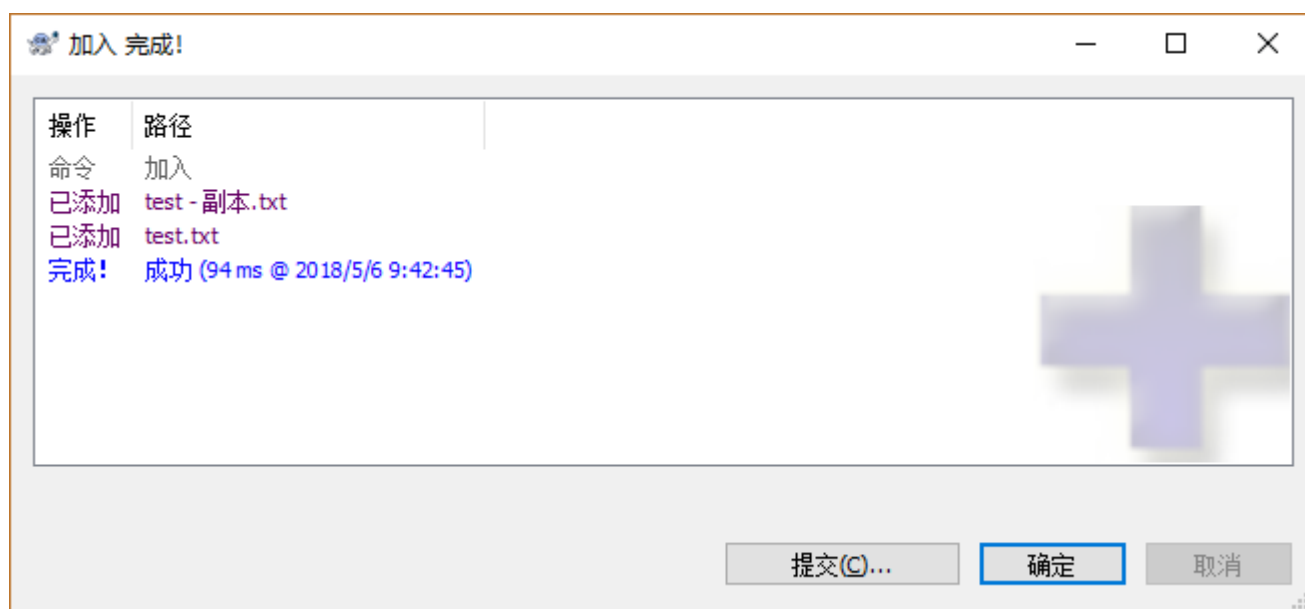
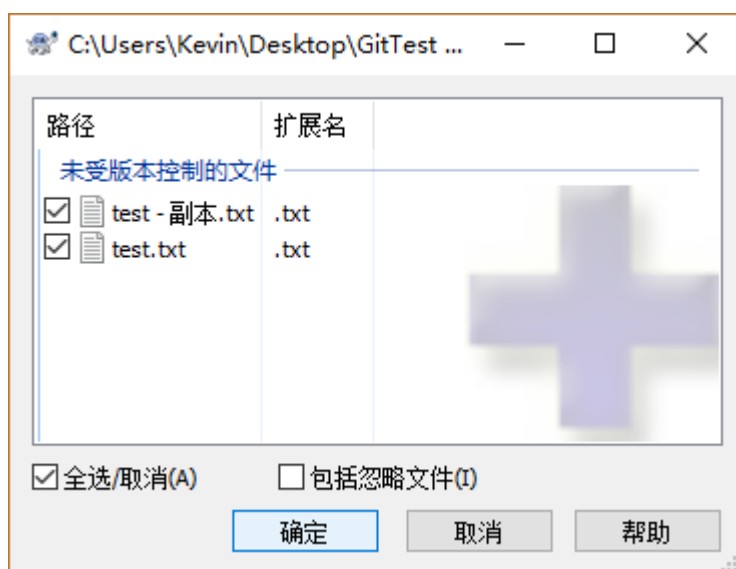


- 工作区: 创建的工作目录 GitTest
- 版本库: 是在工作目录被创建出来的
 - 里边有一个暂存区 stage
 - 如果在工作区创建了一个新文件, 需要将新文件添加到暂存区
 - 一个文件只需要做一次
 - 有一个默认叫master的主分支
 - 如果想完成文件的版本管理, 需要将暂存区数据, 提交到master主分支上
 - 提交之后, master分支上就会添加一对应的节点

1.3 添加新文件到暂存区并提交

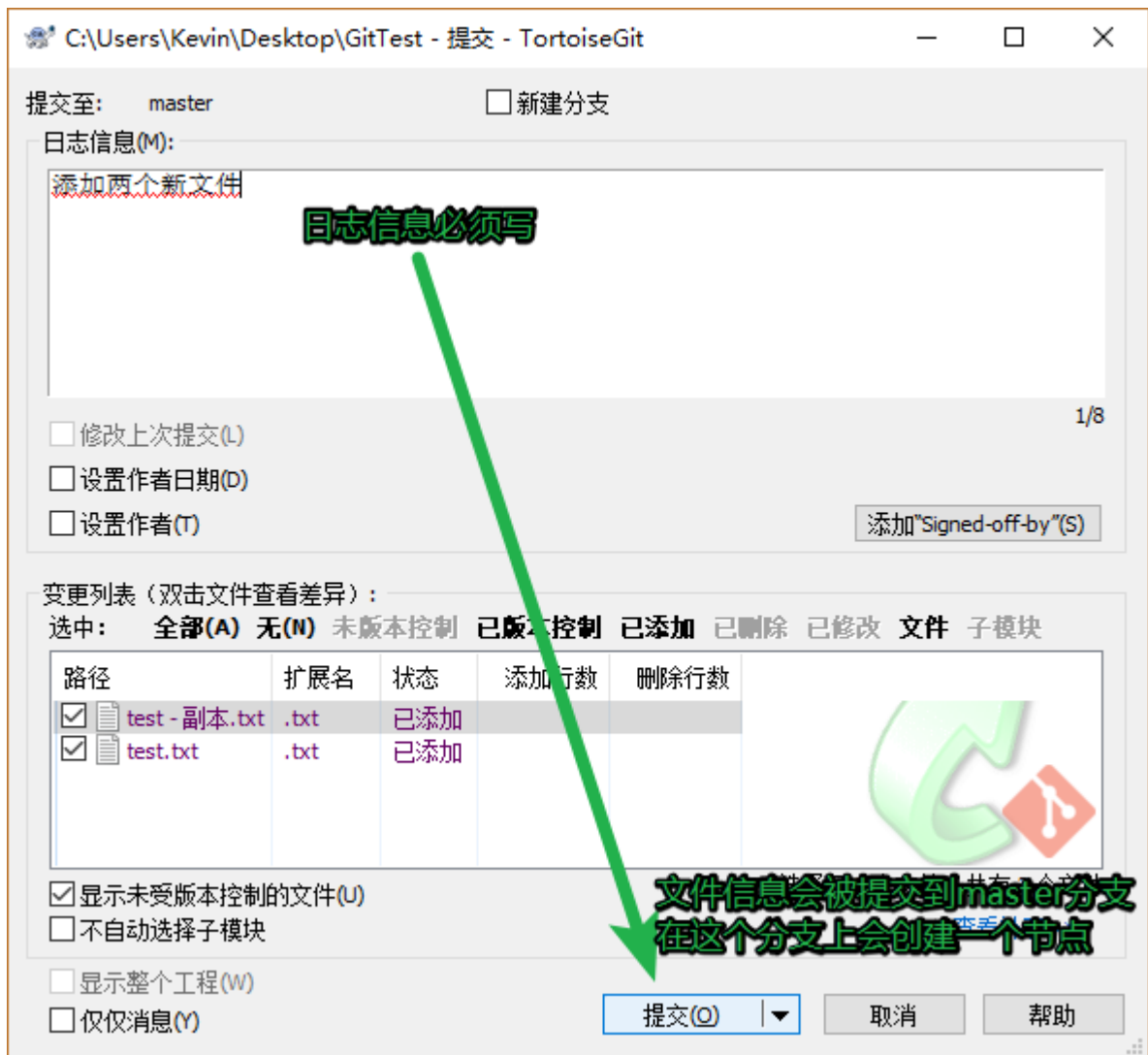
- 将新创建的文件添加到暂存区

- 工作目录中鼠标右键:



- 将暂存区的文件提交到默认分支





2. 修改文件

2.1 修改工作区文件

- 修改工作区文件
- 直接提交修改的内容到默认分支

Git 提交(C) -> "master"...

C:\Users\Kevin\Desktop\GitTest - 提交 - TortoiseGit

提交至: master

☐ 新建分支

日志信息(M):

添加了一个字符串

1/1

☐ 修改上次提交(L)


☐ 设置作者日期(D)


☐ 设置作者(T)

添加"Signed-off-by"(S)

变更列表 (双击文件查看差异):

选中: 全部(A) 无(N) 未版本控制 已版本控制 已添加 已删除 已修改 文件 子模块

路径	扩展名	状态	添加行数	删除行数
<input checked="" type="checkbox"/>  test.txt	.txt	已修改	3	1



☒ 显示未受版本控制的文件(U)

选择了 1 个文件, 共有 1 个文件

[查看补丁>>](#)

☐ 不自动选择子模块

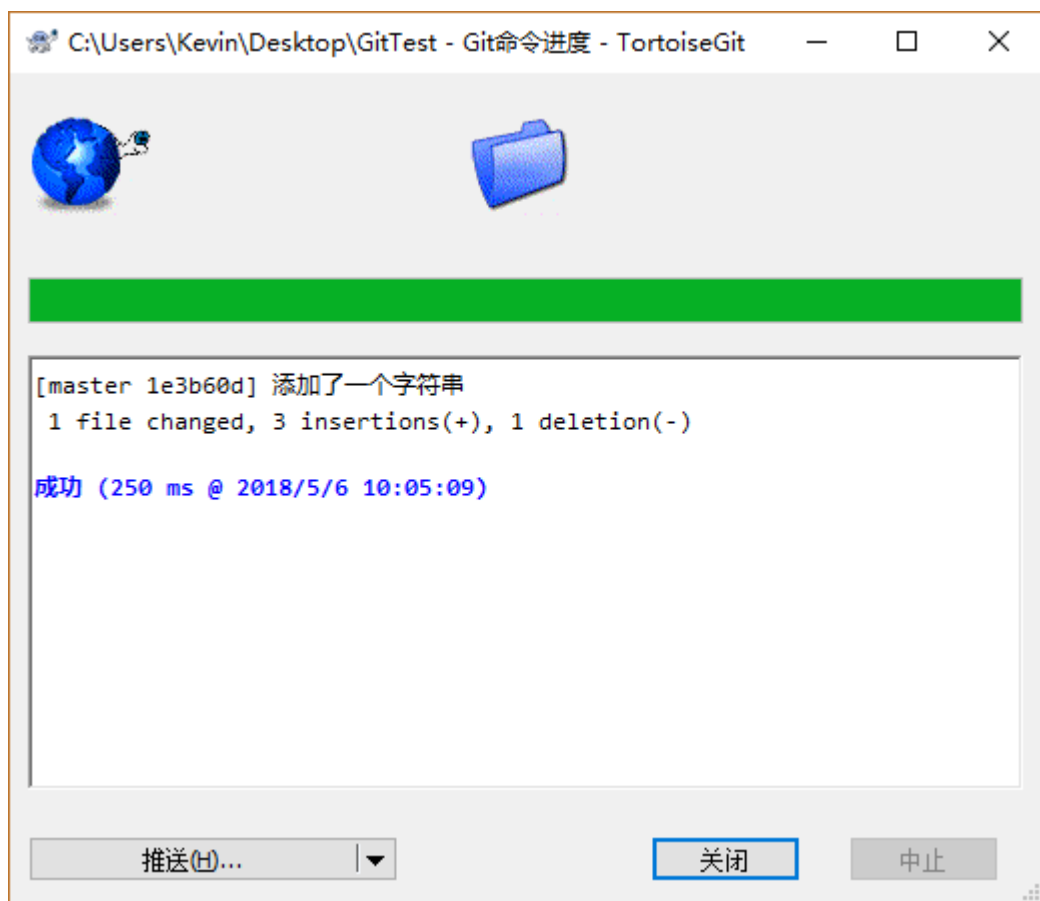
☐ 显示整个工程(W)

☐ 仅仅消息(M)

提交(O) | ▼

取消

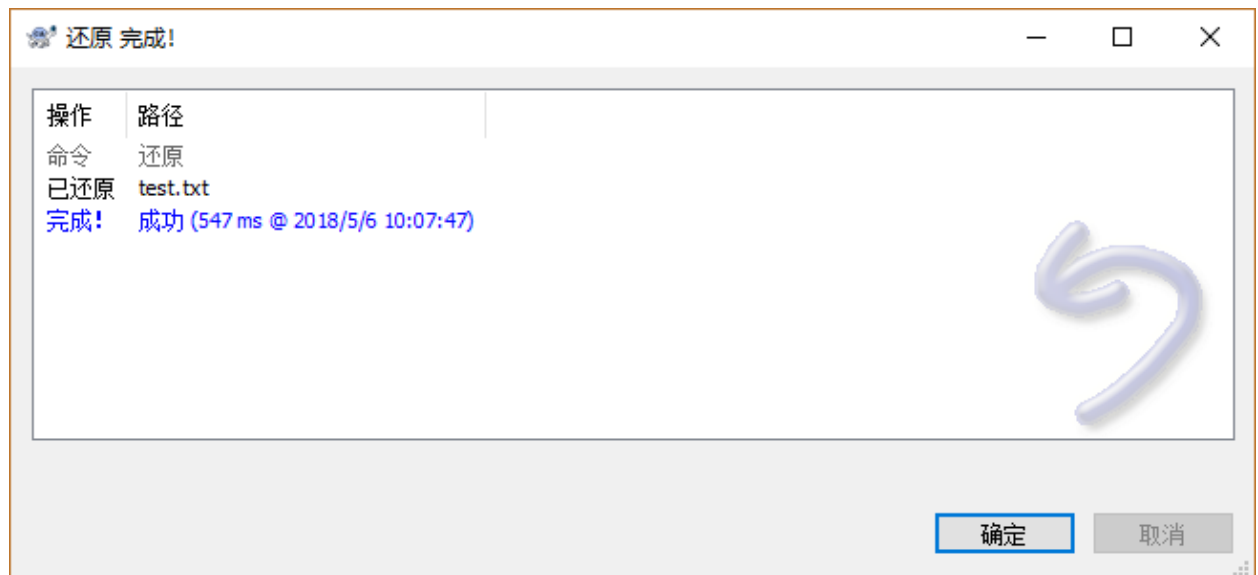
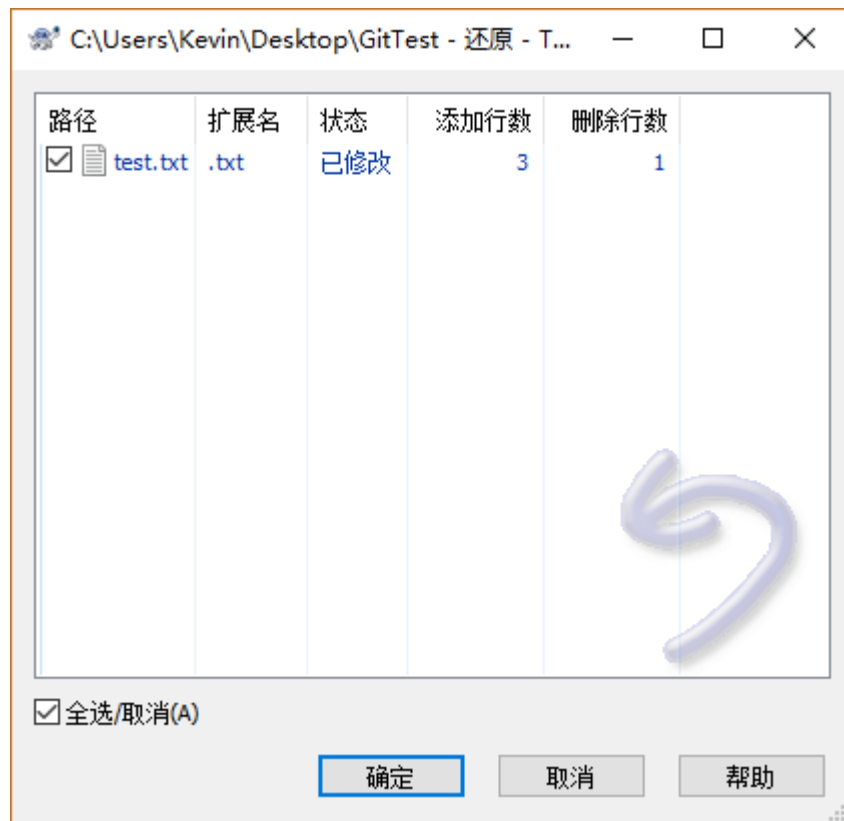
帮助



2.2 还原修改

1. 操作





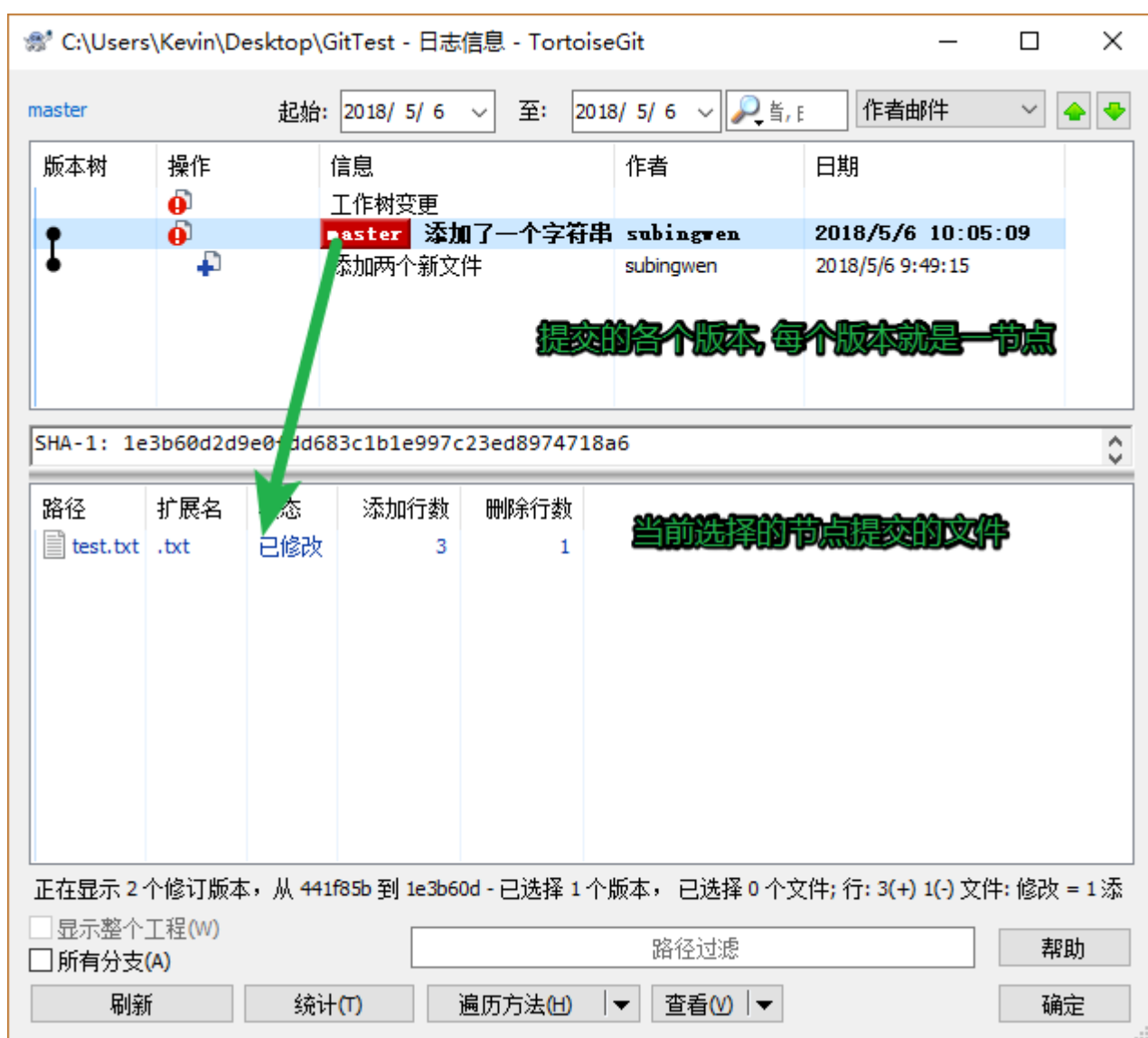
2. 注意事项

- 还原之后, 之前的修改就全部丢失了
- 什么时候才能还原?
 - 工作区文件做了修改, 还没有提交之前是可以还原的, 提交之后就无法使用还原功能了

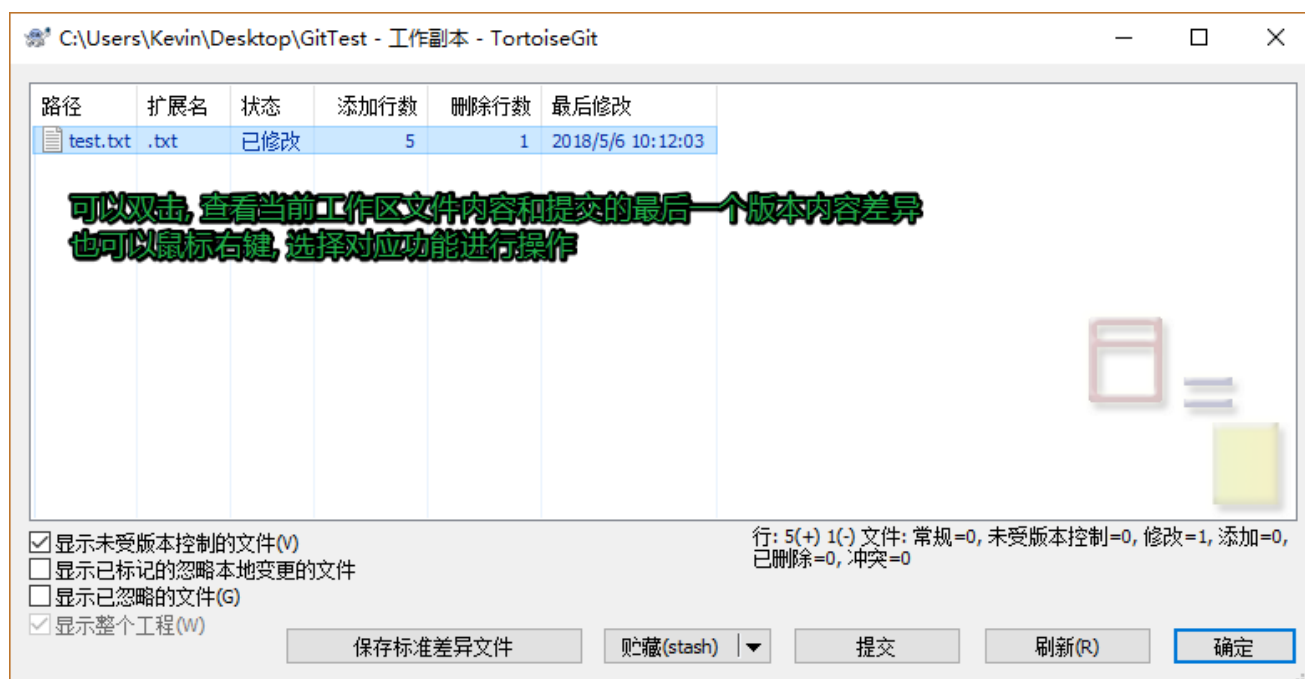
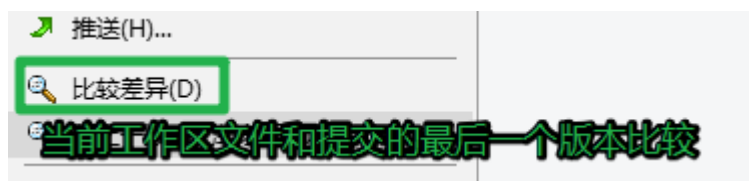
2.3 查看修改历史



查看当前版本和上一个版本差异



2.4 差异比较



2.5 删除文件

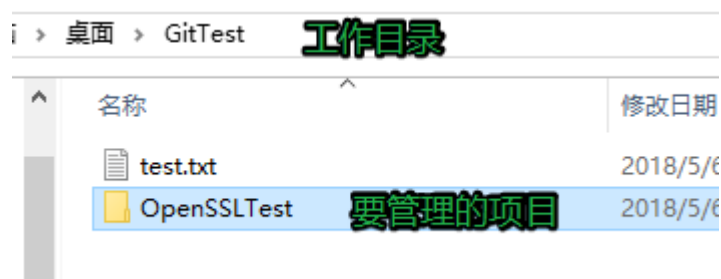
- 删除的版本库中管理的文件
- 删除的方式:
 - 直接通过键盘的del键删除
 - 通过git工具的右键菜单删除



- 最终将文件删除之后, 都需要做提交的操作

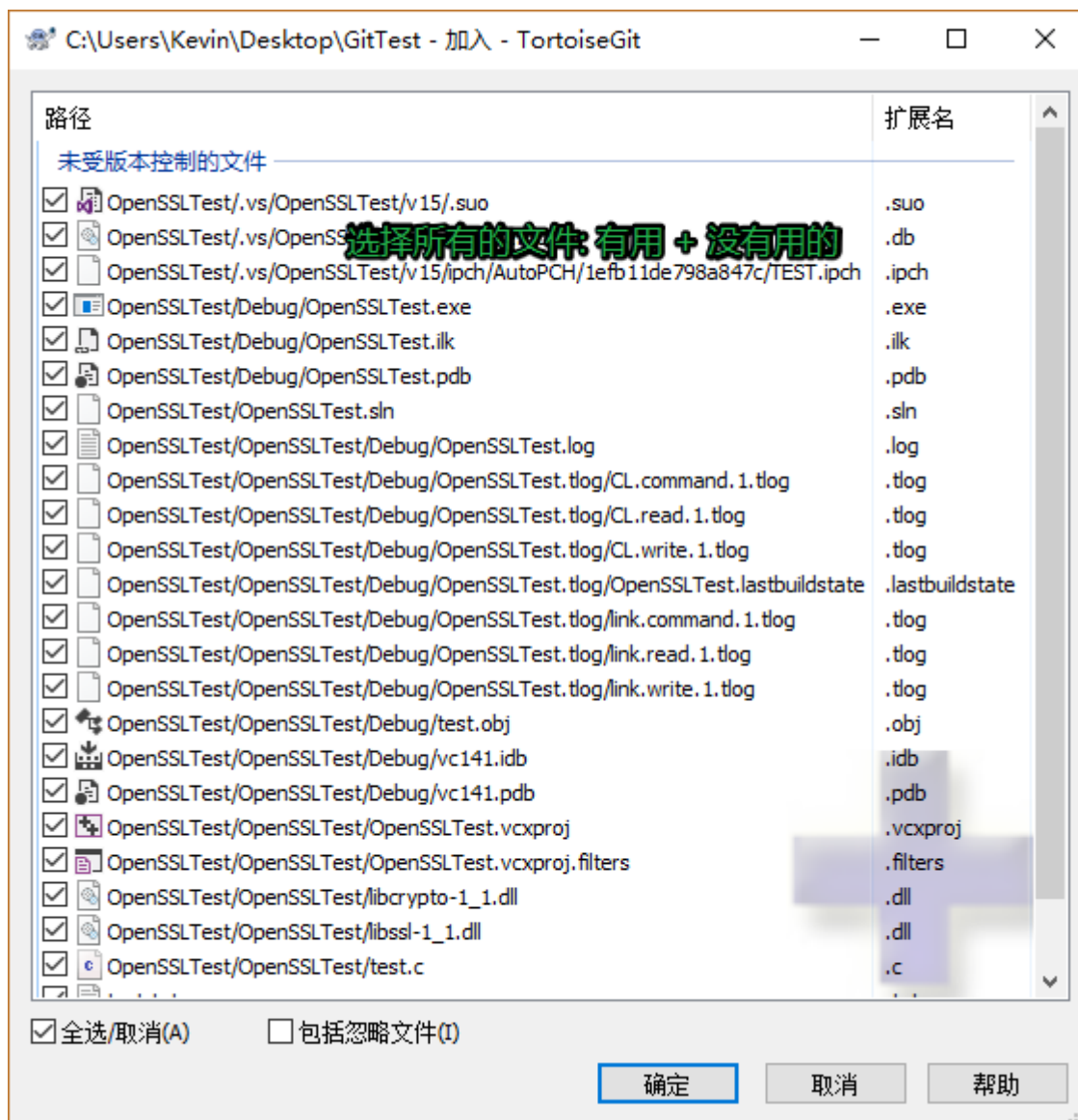
3. 案例: 添加一个本地项目到仓库

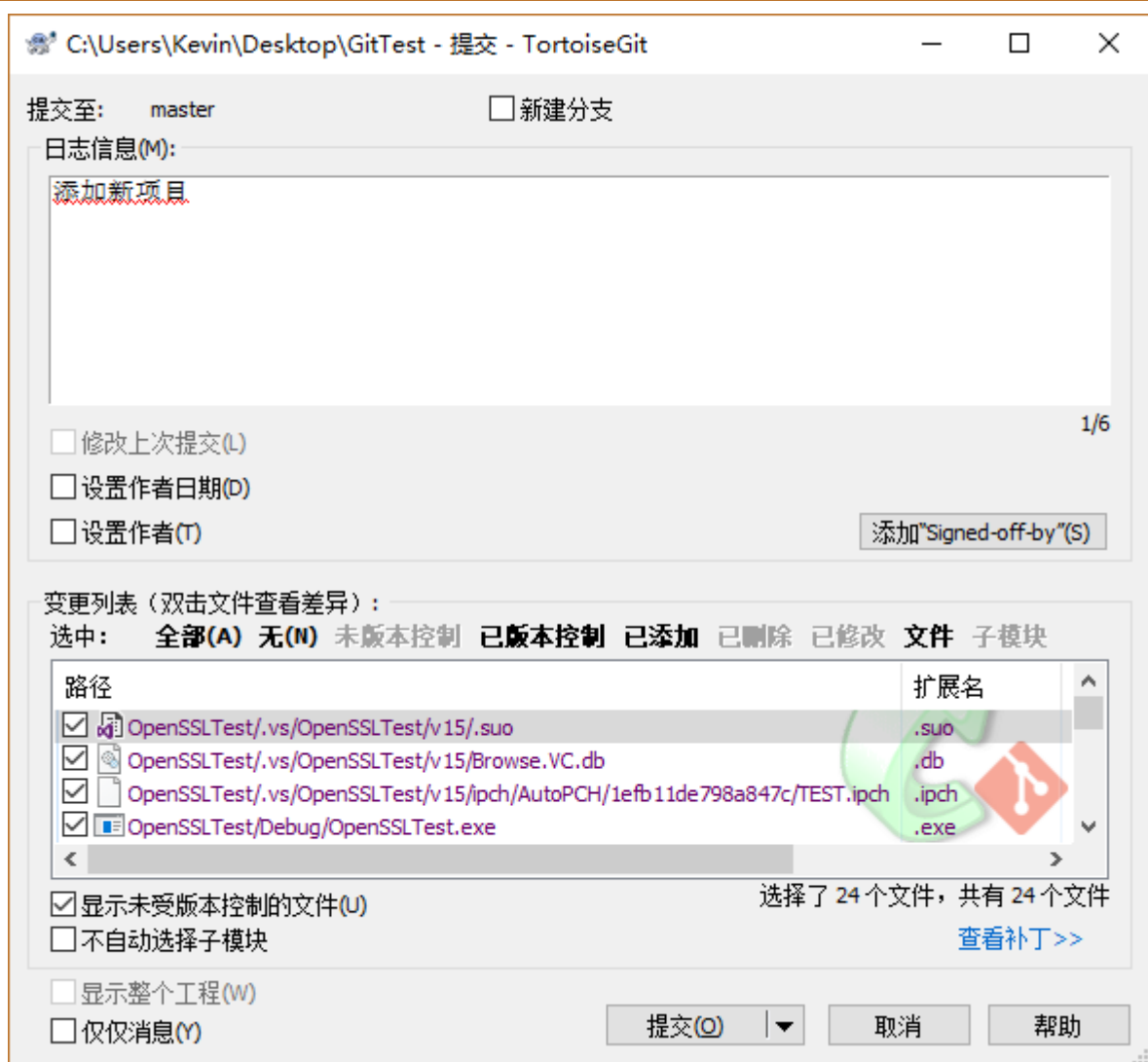
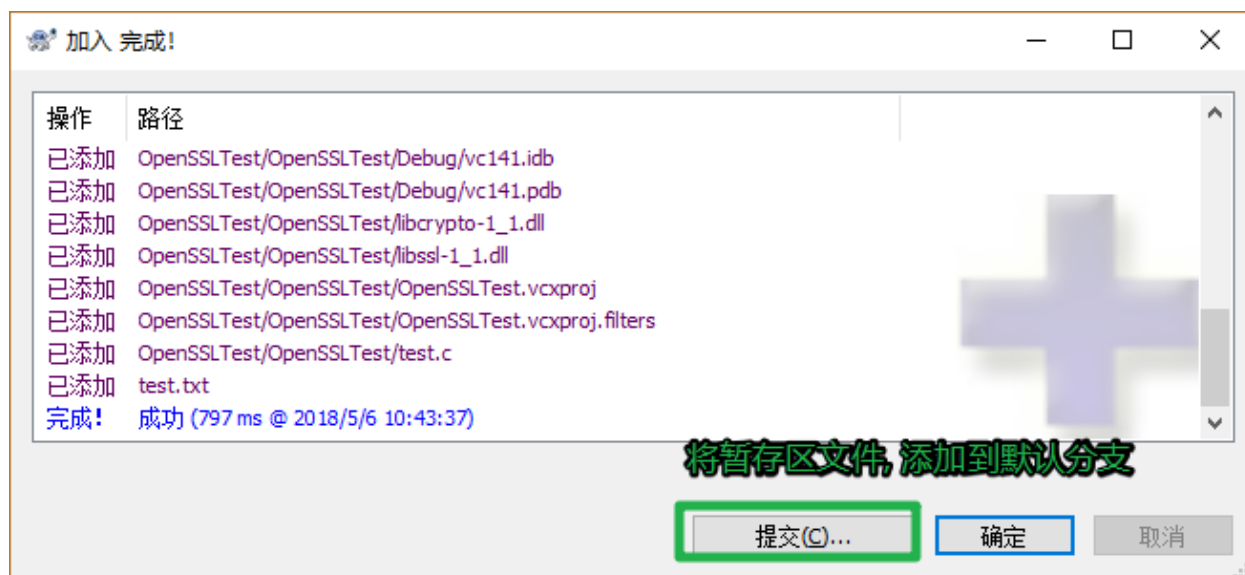
1. 确定一个工作目录

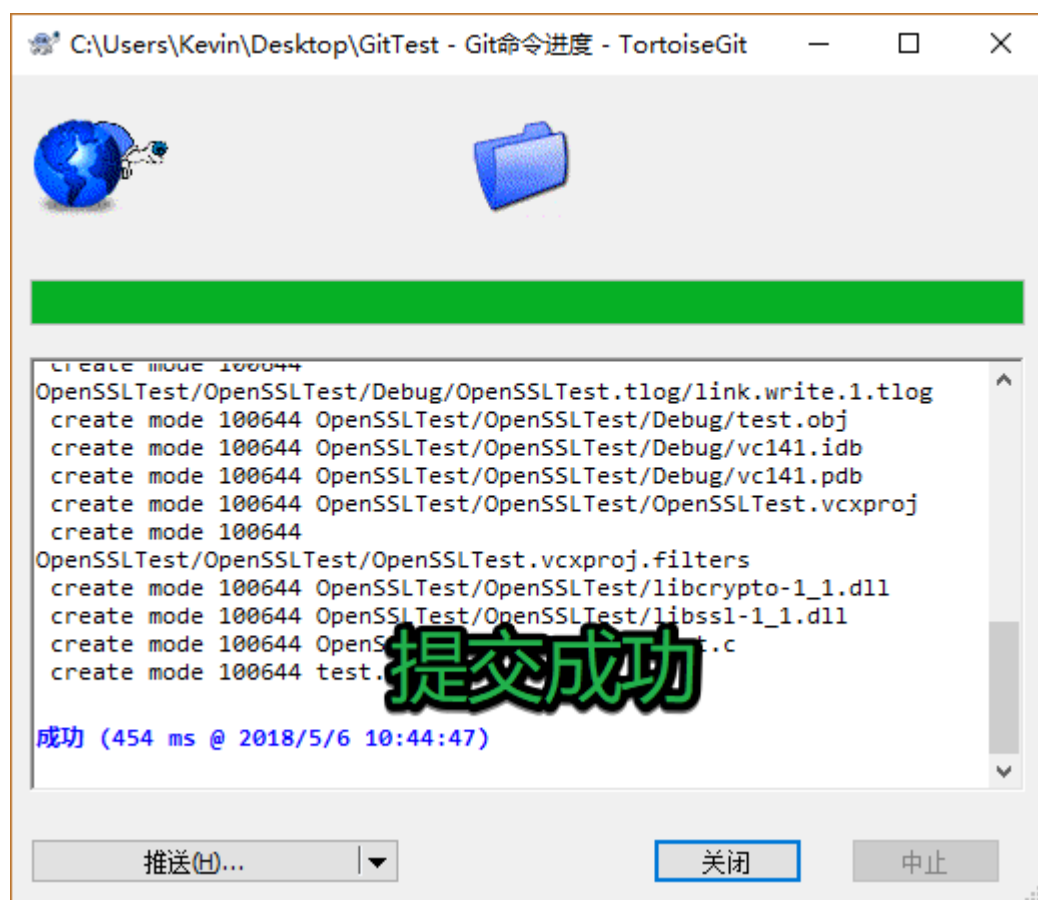


- 版本管理仓库应该在哪创建?
 - GitTest 中创建代码仓库

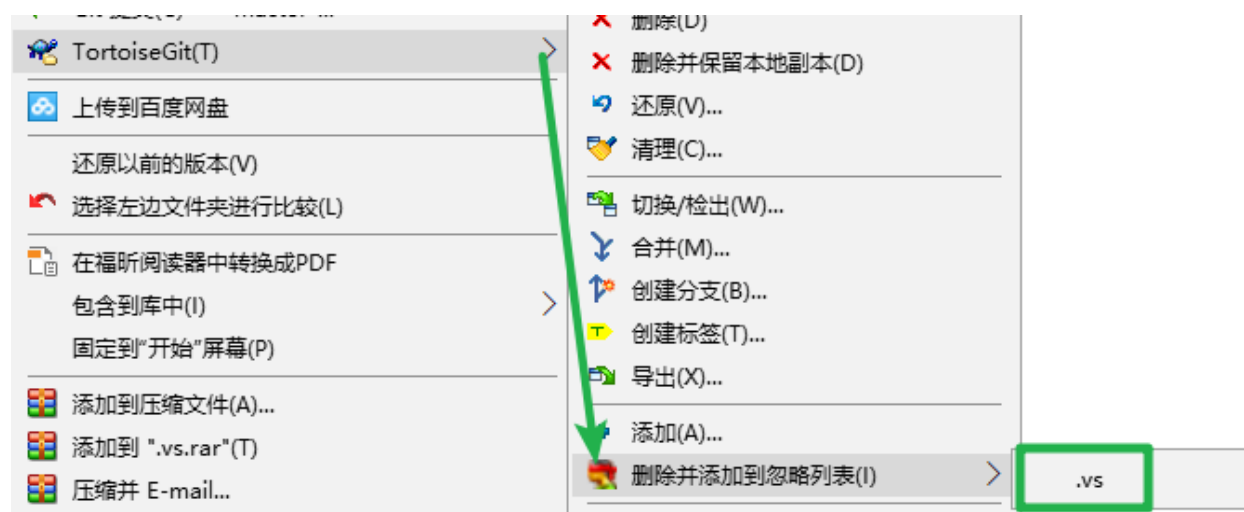
2. 添加新文件到暂存区

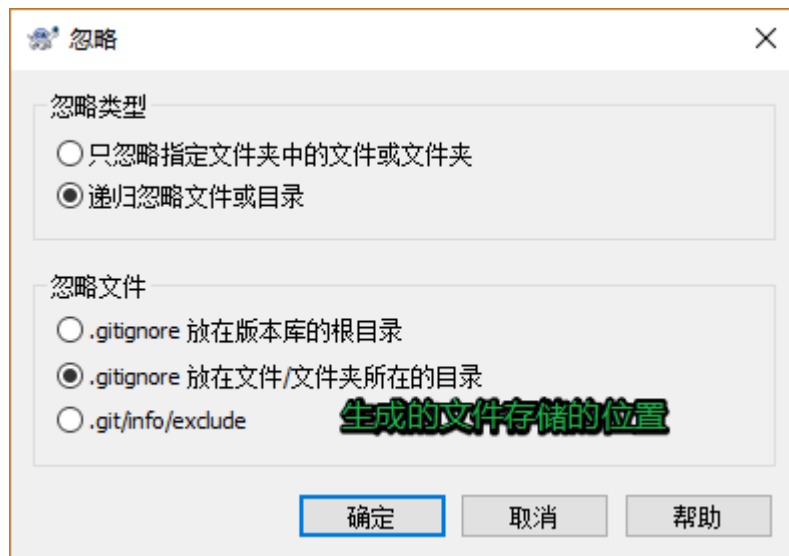




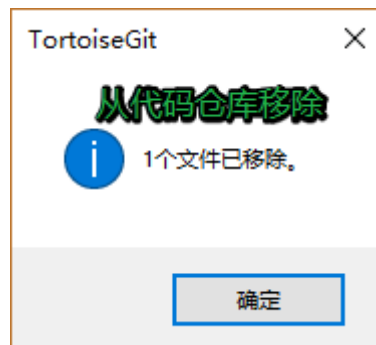
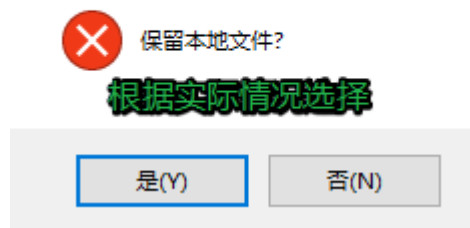


3. 添加忽略列表

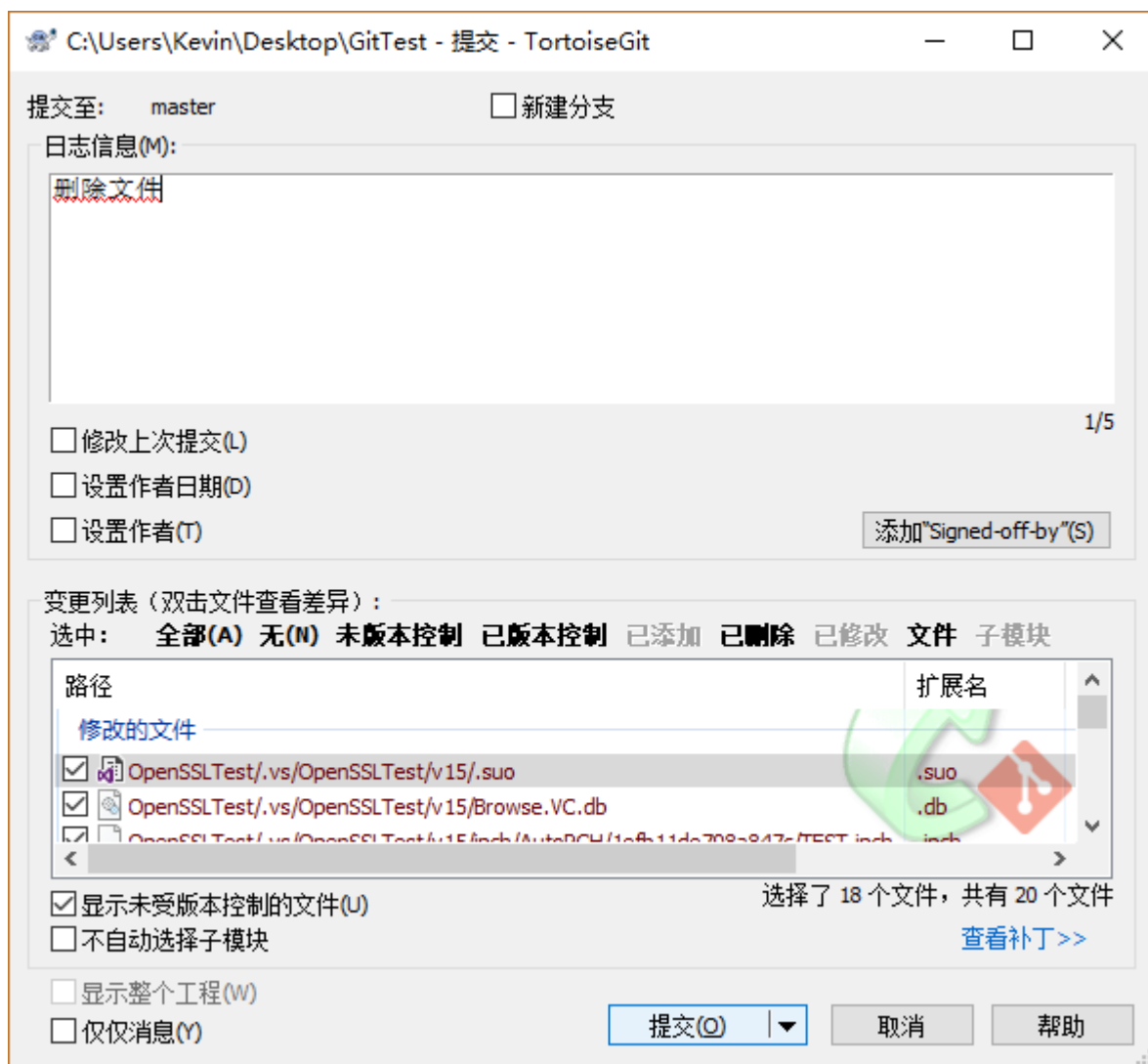




TortoiseGit



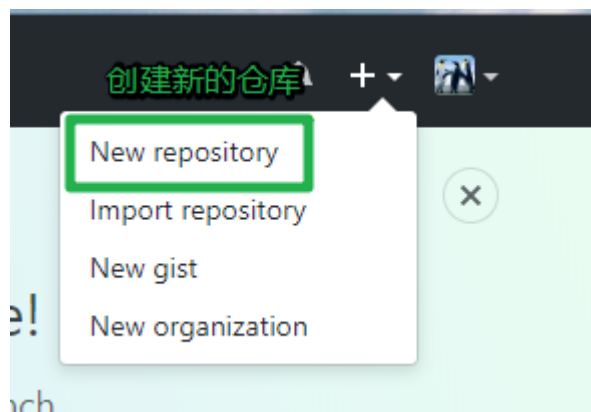
- 提交



4. 添加远程仓库

4.1 远程仓库的添加和创建

1. 创建仓库



Owner: subwen / Repository name: MyGitTest ✓

Great repository names are short and memorable. Need inspiration? How about **reimagined-dollop**.

Description (optional)

☒ Public **所有的人都可以访问**
Anyone can see this repository. You choose who can commit.

☐ Private **指定的人可以访问**
You choose who can see and commit to this repository.

☐ Initialize this repository with a README **先不要添加readme文件, 不添加创建的仓库就是一个空的仓库, 所谓的空仓库: 没有master默认分支**
This will let you immediately clone the repository to your computer, or you can clone an existing repository.

Add .gitignore: None | Add a license: None ⓘ

创建仓库
Create repository

○ 空仓库的信息

subwen / MyGitTest **这是一个空的仓库** Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or **HTTPS SSH** **https://github.com/subwen/MyGitTest.git** **访问该仓库的地址, 有两种方式**

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

4.2 本地仓库同步到远程仓库

- 前提条件: 远程仓库必须是空的

- 1. 拿到远程仓库的访问地址 https / ssh

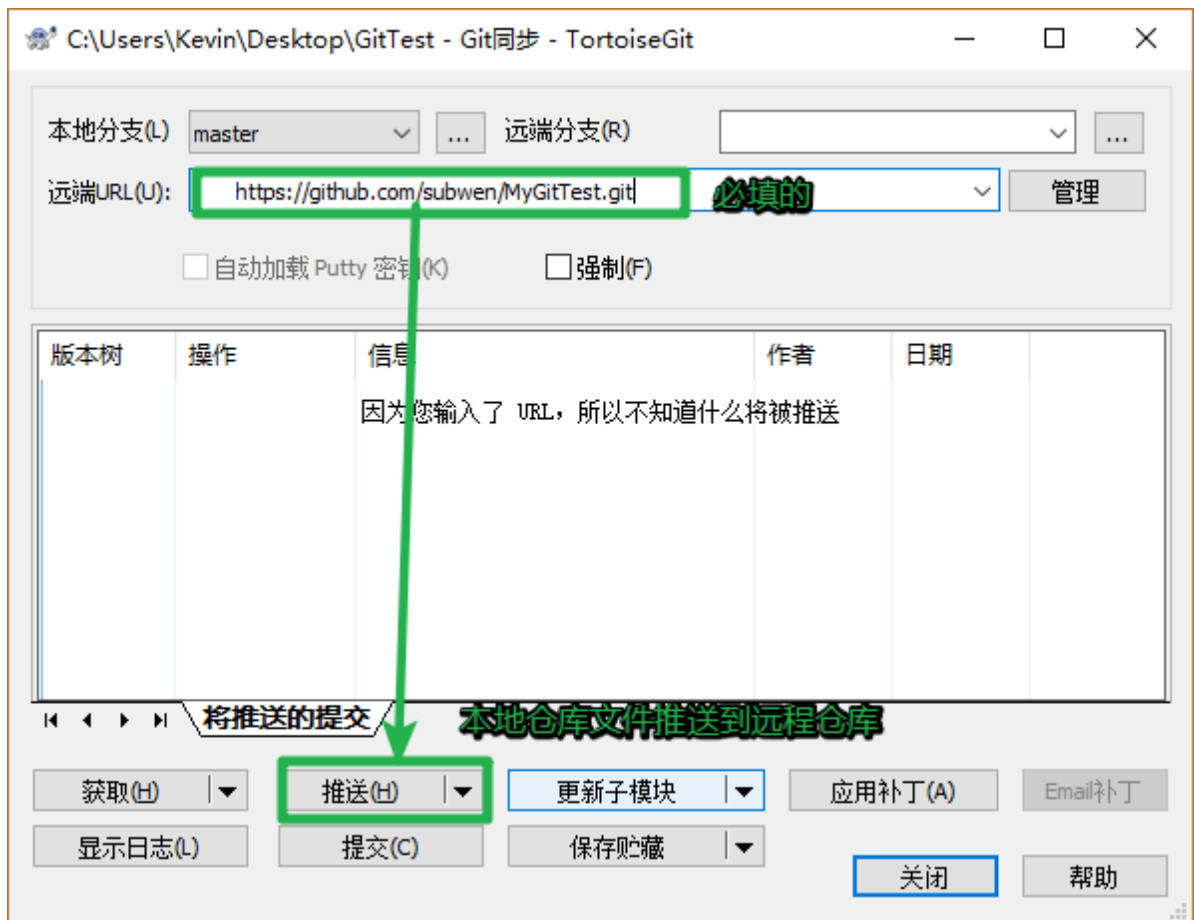
- https 可以直接使用
 - <https://github.com/subwen/MyGitTest.git>
 - ssh 需要配置
 - 需要在本地生成密钥对
 - 将公钥配置到github上, 自己保存私钥

- 使用https提交

- 1.



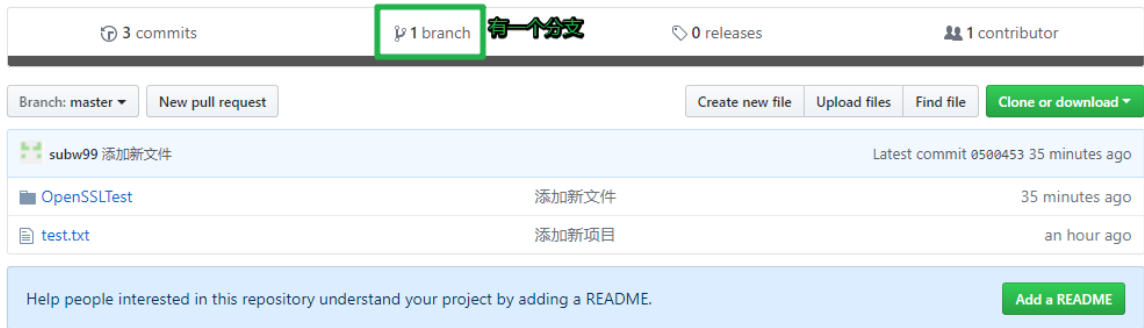
- 2.



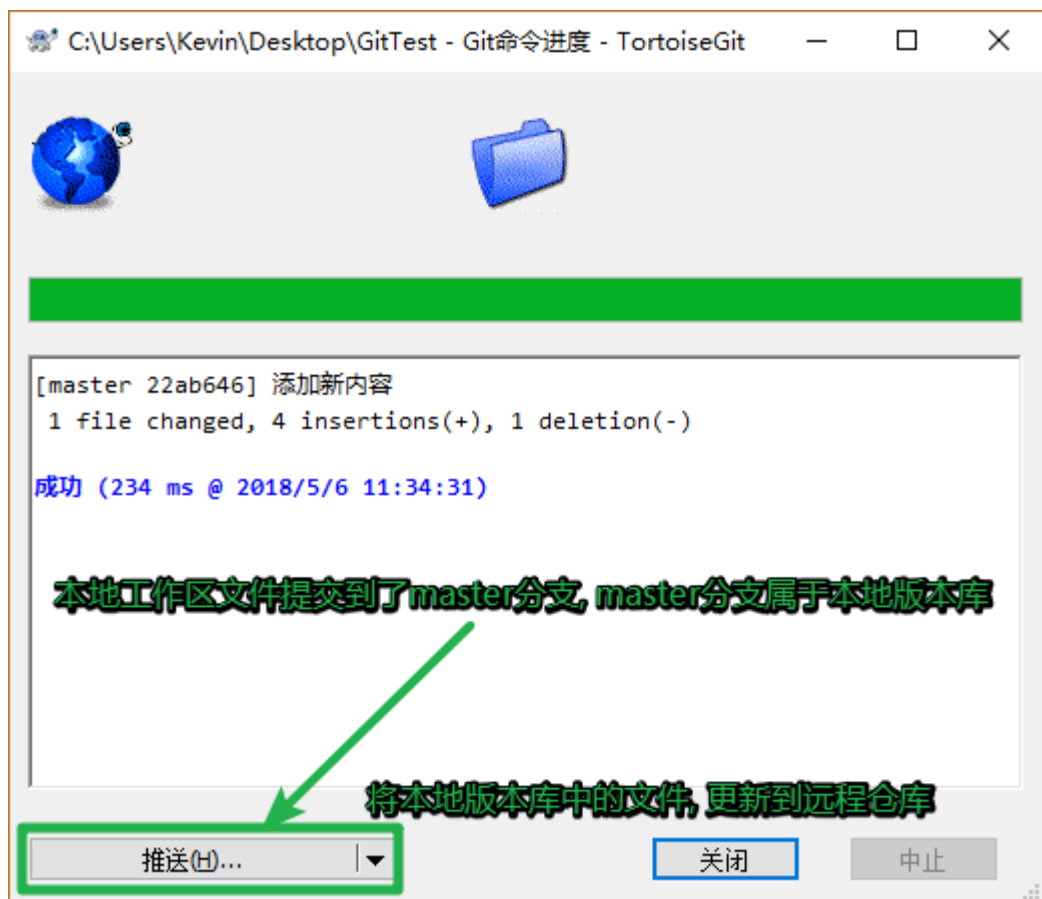
3.

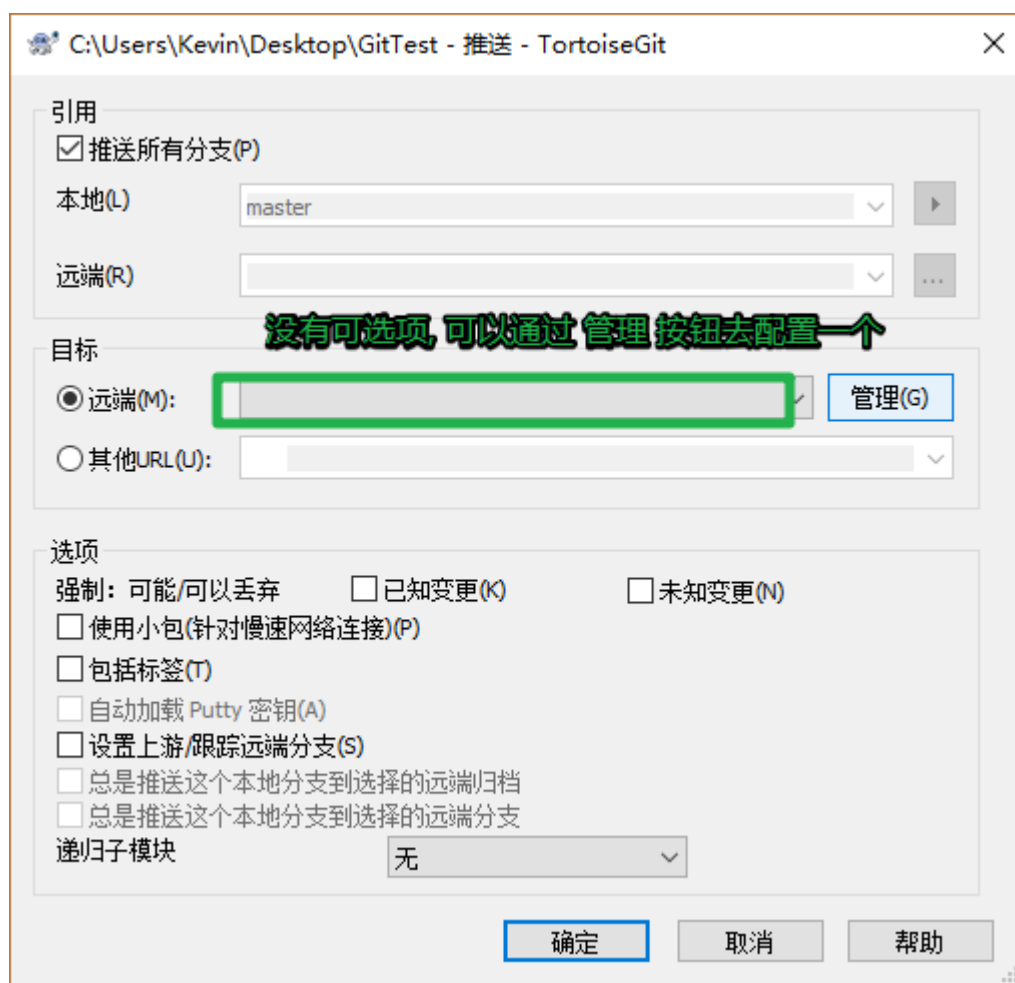


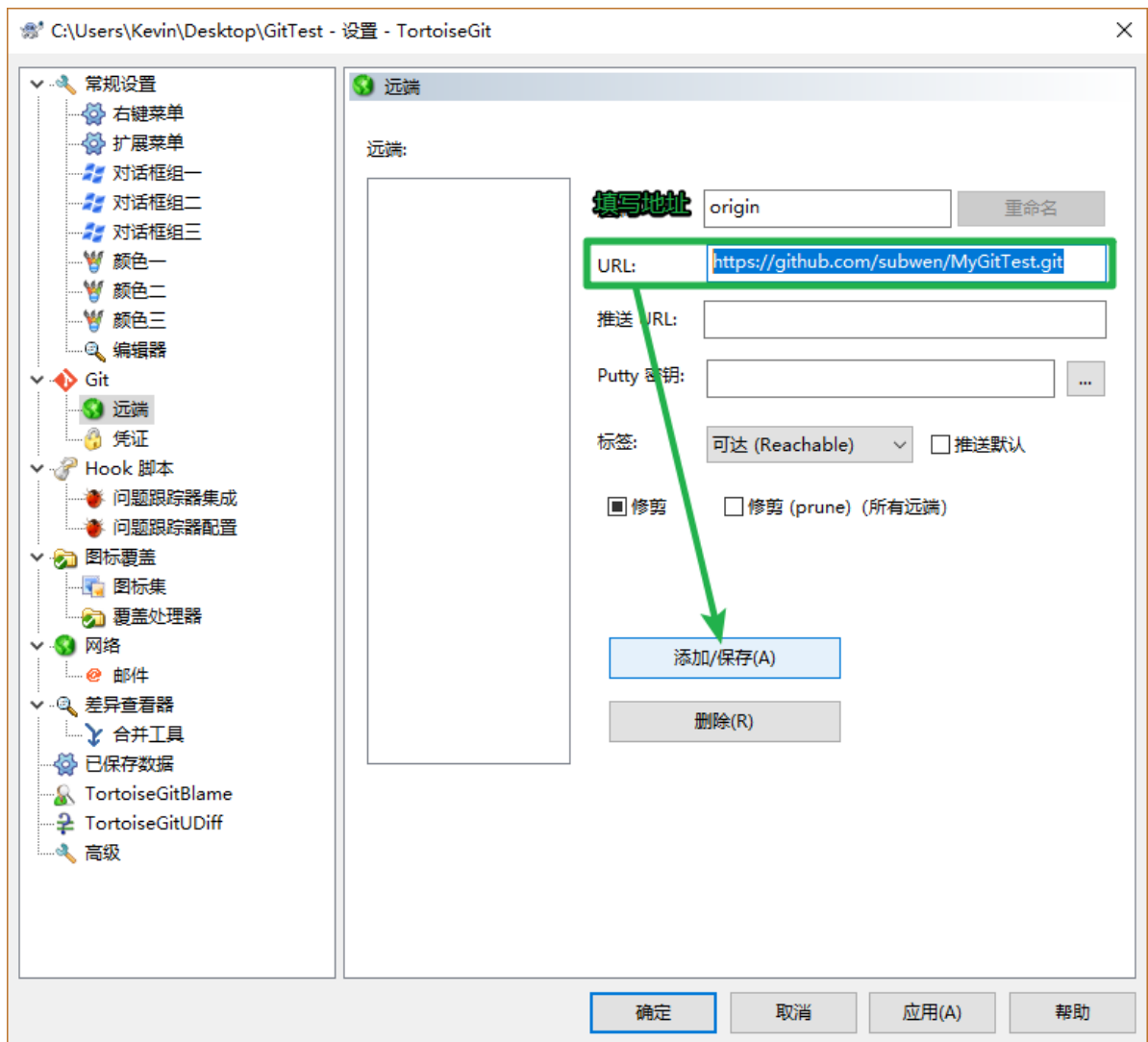
4. 远程仓库状态



5. 本地文件修改

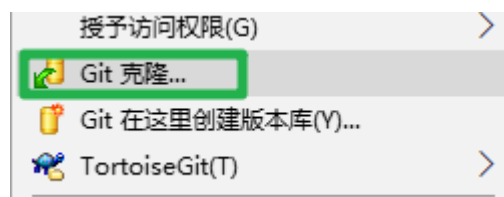


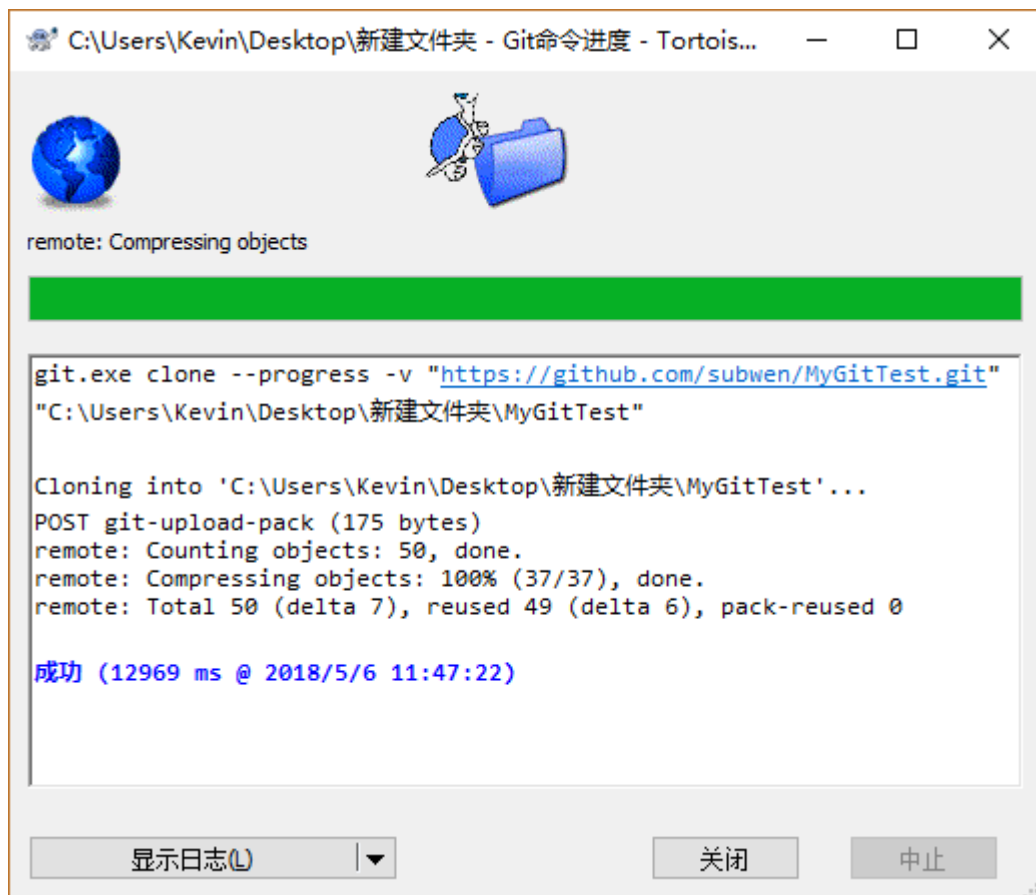
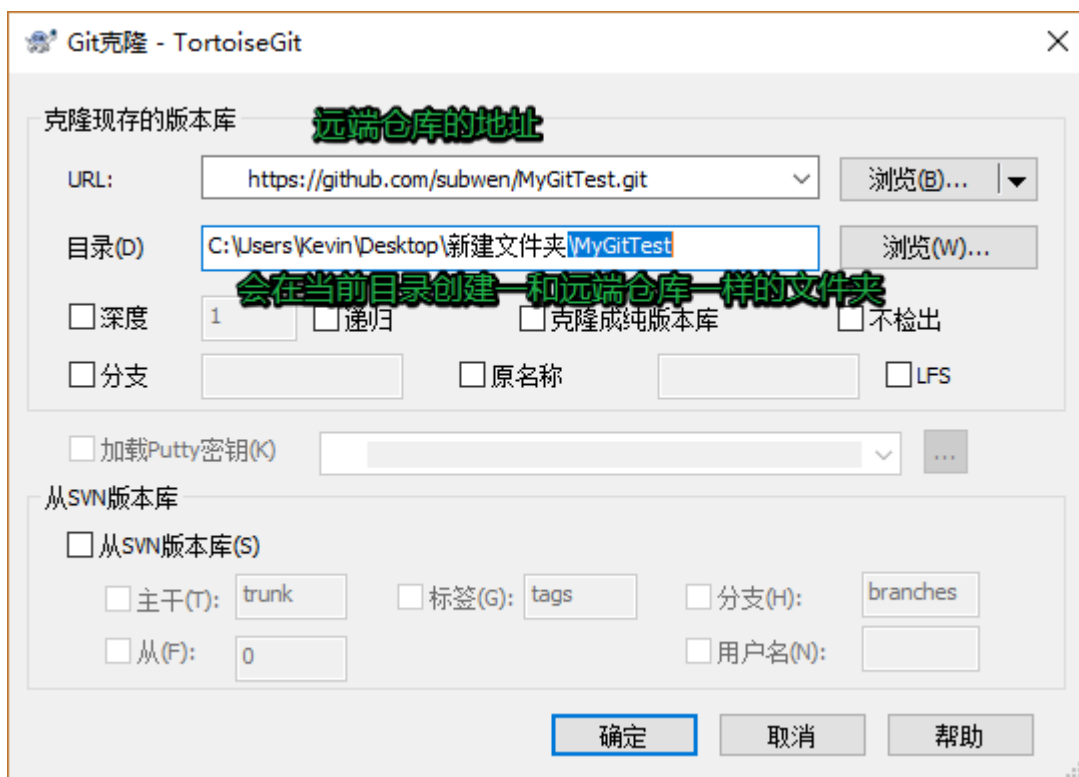




4.3 克隆远程仓库到本地

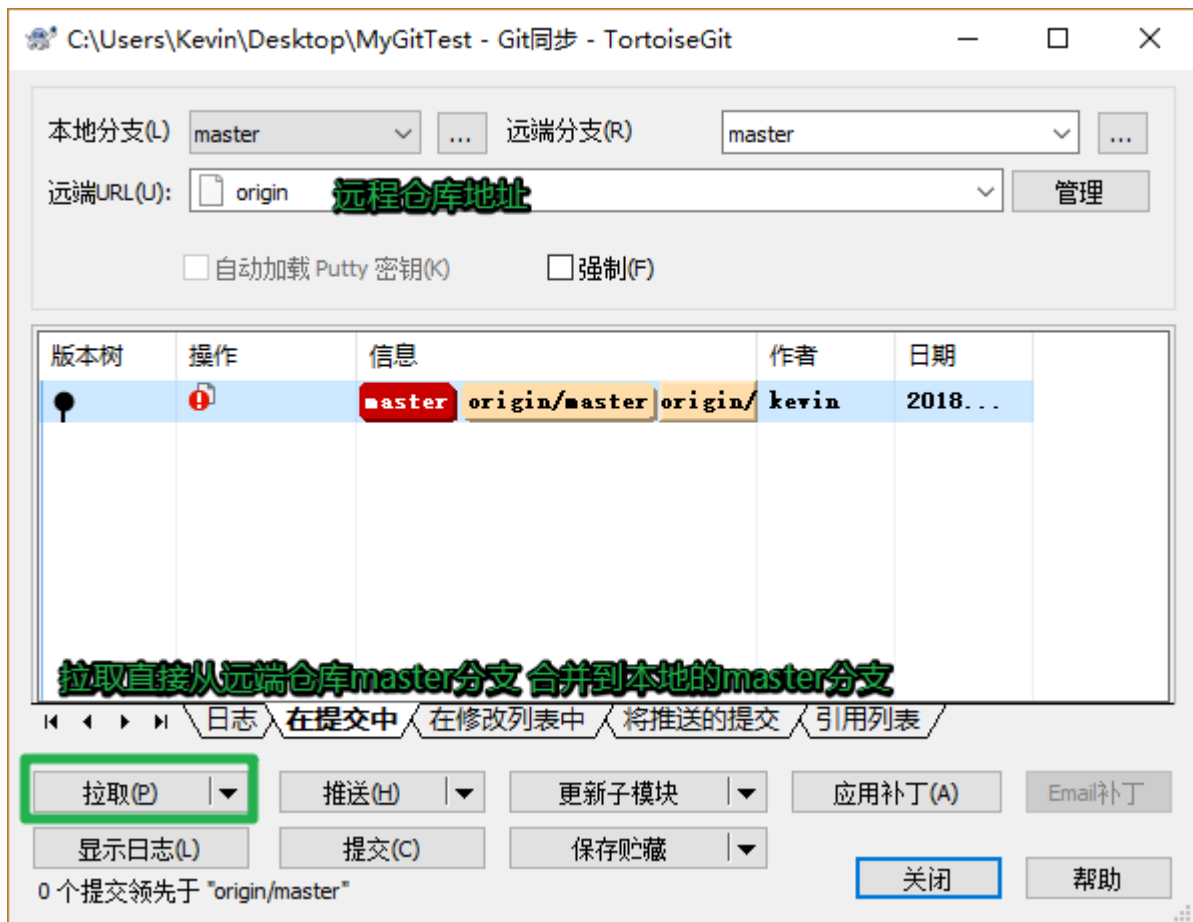
1. 克隆操作: 将远程仓库复制到本地





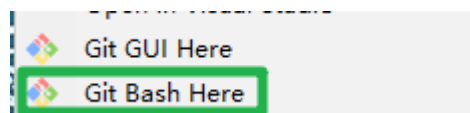
2. 从远程仓库向本地更新文件





4.4 ssh 配置

1. 生成一个密钥对称, 在git提供的命令行窗口中



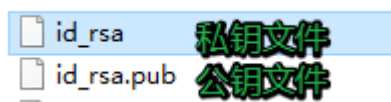
```
1 ssh-keygen -t rsa
```

```
Kevin@X1-Carbon-2017 MINGW64 ~/Desktop
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Kevin/.ssh/id_rsa): 直接回车
Enter passphrase (empty for no passphrase):
Enter same passphrase again: 直接回车*3
Your identification has been saved in /c/Users/Kevin/.ssh/id_rsa.
Your public key has been saved in /c/Users/Kevin/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:42ZDEELNAqkSx+zUU6Kpc8zgFkdhnIEJl2uX4rE8tnM Kevin@X1-Carbon-2017
The key's randomart image is:
+---[RSA 2048]-----+
|oo*oB+o|
|. +X=+oo.|
|.Boo +.|
|+=0 o .|
|+*+= S|
|.O* o .|
|. o =|
| o E o .|
| o|
+-----[SHA256]-----+
```

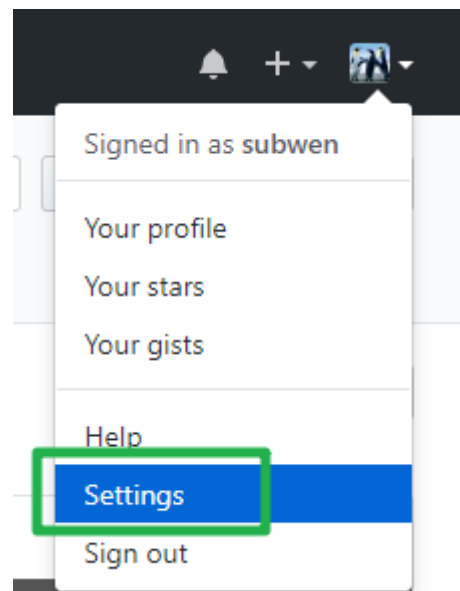
秘钥生成成功了

2. 秘钥生成的位置

1 C:\Users\电脑用户名\.ssh



3. 给github设置公钥



Personal settings

Profile

Account

Emails

Notifications

Billing

SSH and GPG keys

SSH keys

New SSH key

There are no SSH keys associated with your account.
Check out our guide to [generating SSH keys](#) or [troubleshoot common SSH Problems](#).

GPG keys

New GPG key

There are no GPG keys associated with your account.
Learn how to [generate a GPG key and add it to your account](#).

SSH密钥

新的SSH密钥

这是与您的帐户关联的SSH密钥列表。删除任何你不认识的钥匙。

SSH

mykey

指纹:

添加于

31:c2:63:c8:33:23:54:e4:fd:09:31:76:6d:af:e4:5c

6 May 2018

从未使用 - 读/写

删除

- 关于git工具的设置

授予访问权限(G)

Git 同步...

Git 提交(C) -> "master"...

TortoiseGit(T)

新建(W)

属性(R)

添加(A)...

添加子模块...

创建补丁序列...

应用补丁序列...

设置(S)

帮助(H)

SSH

SSH 客户端(S):

默认ssh客户端, 需要替换为git的ssh客户端

C:\Program Files\TortoiseGit\bin\TortoiseGitPlink.exe

浏览...

SSH

SSH 客户端(S):

找到git的安装目录, 并进入到 usr/bin/ssh.exe

C:\Program Files\Git\usr\bin\ssh.exe

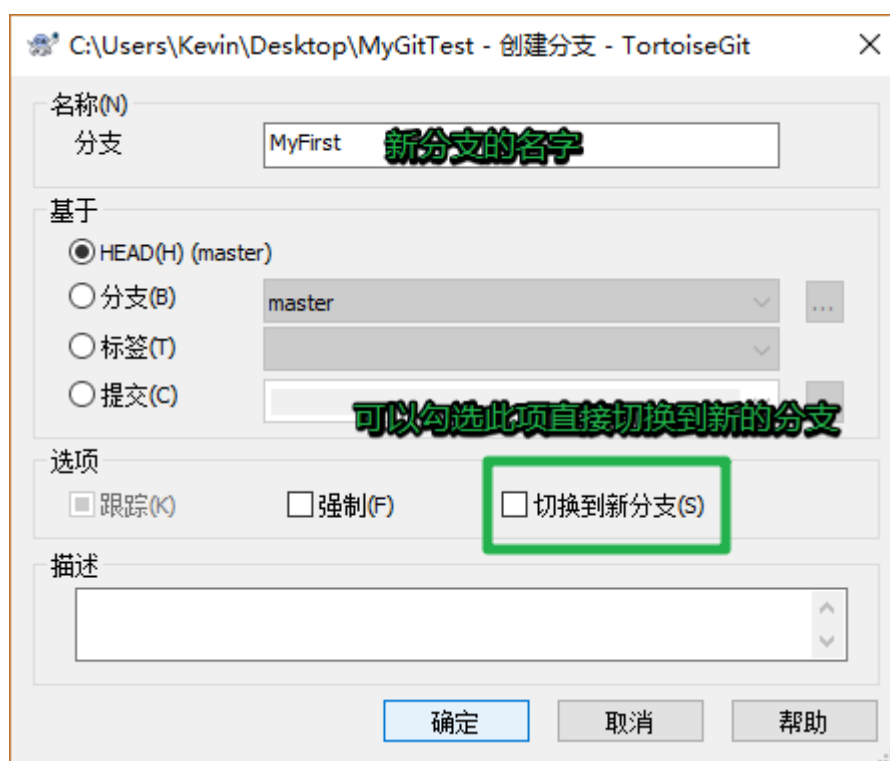
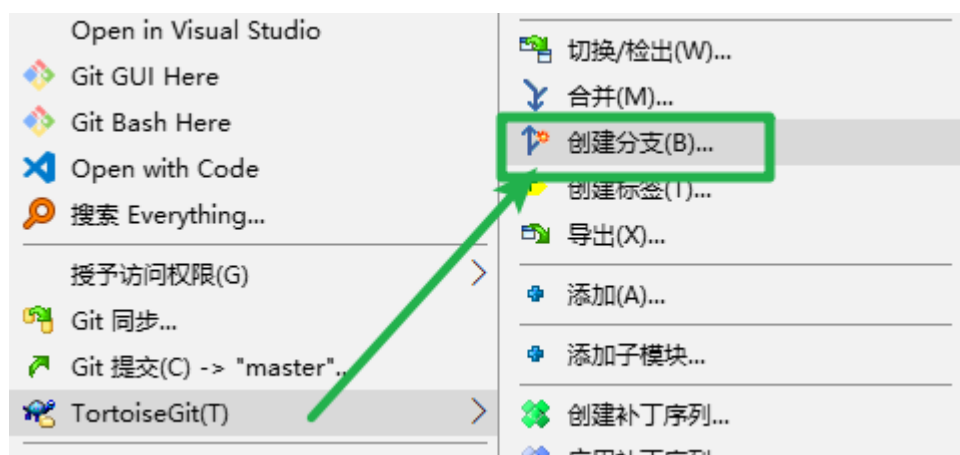
浏览...

5. 管理分支

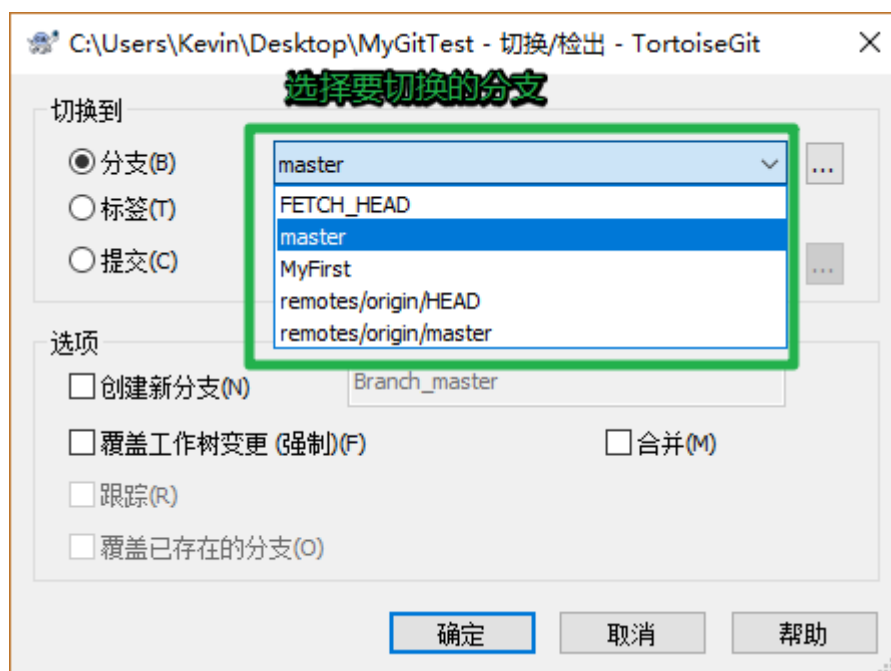
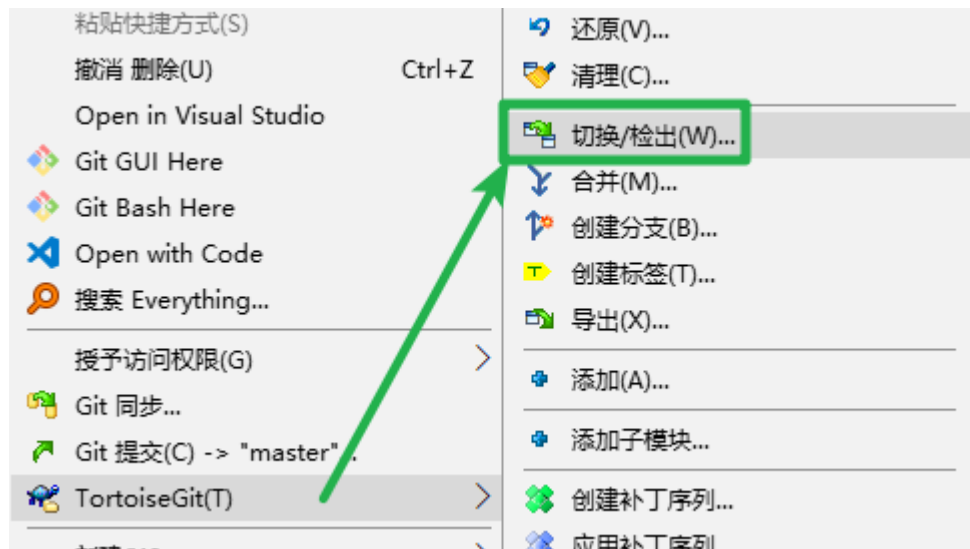
在git中默认只有一个分支 master

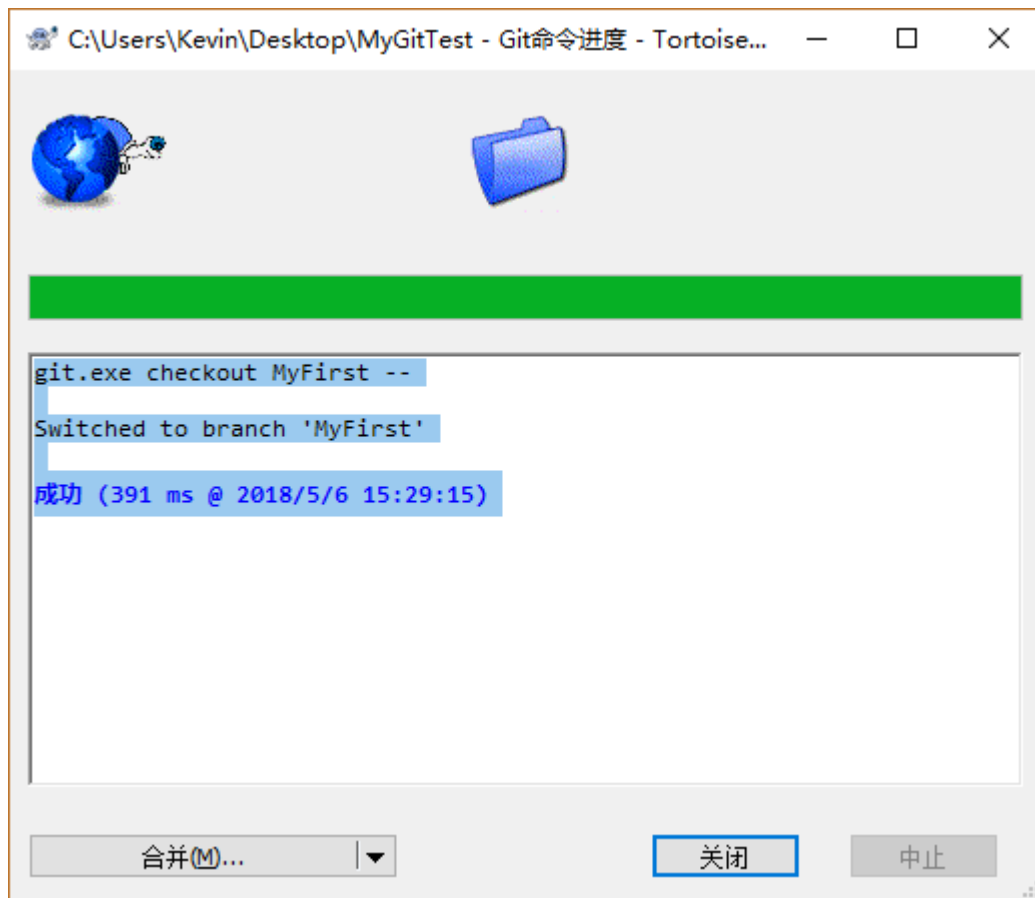
如果创建了分支, 各个分支都是独立的, 互不影响的

5.1 创建分支

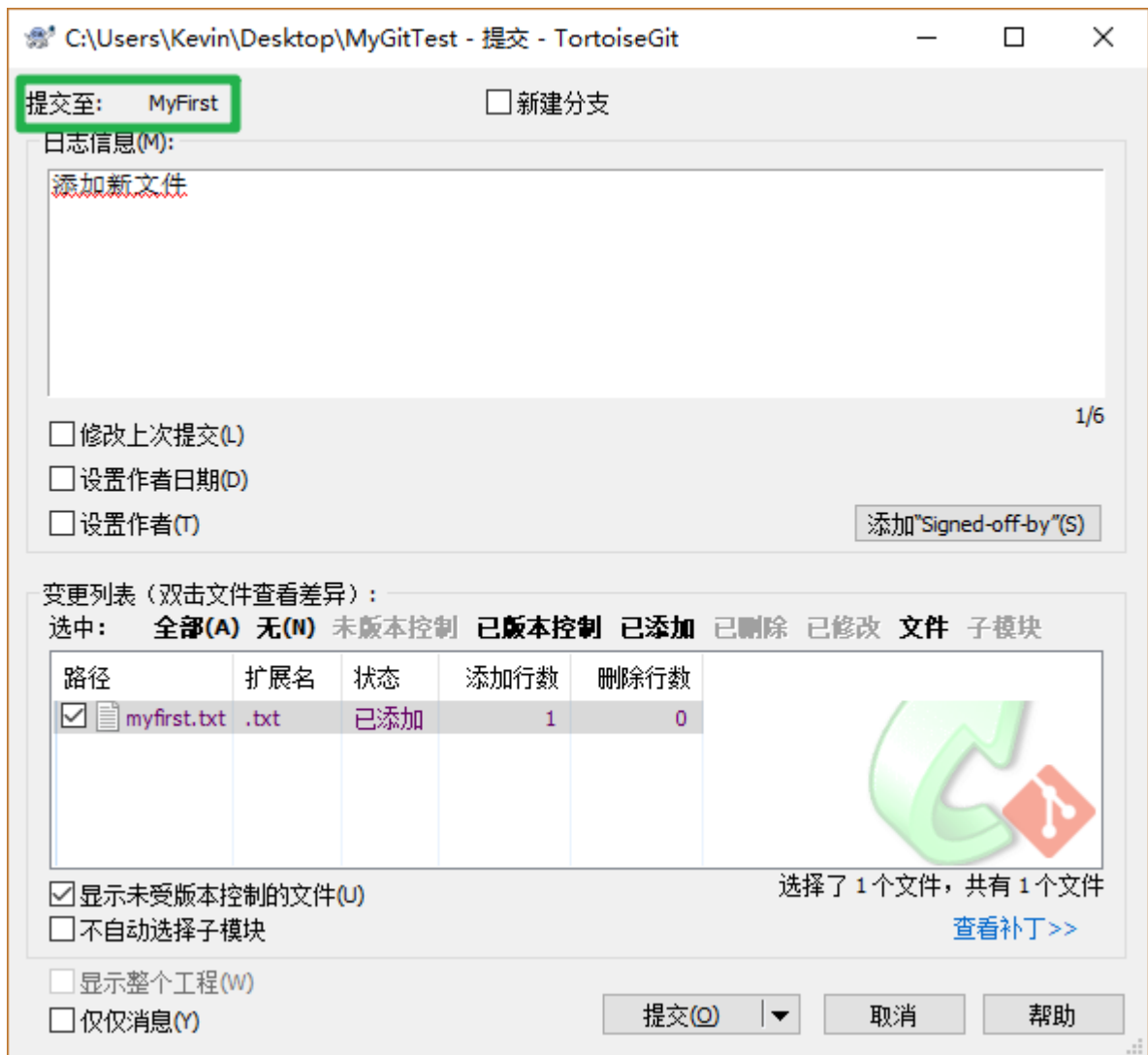


5.2 切换分支





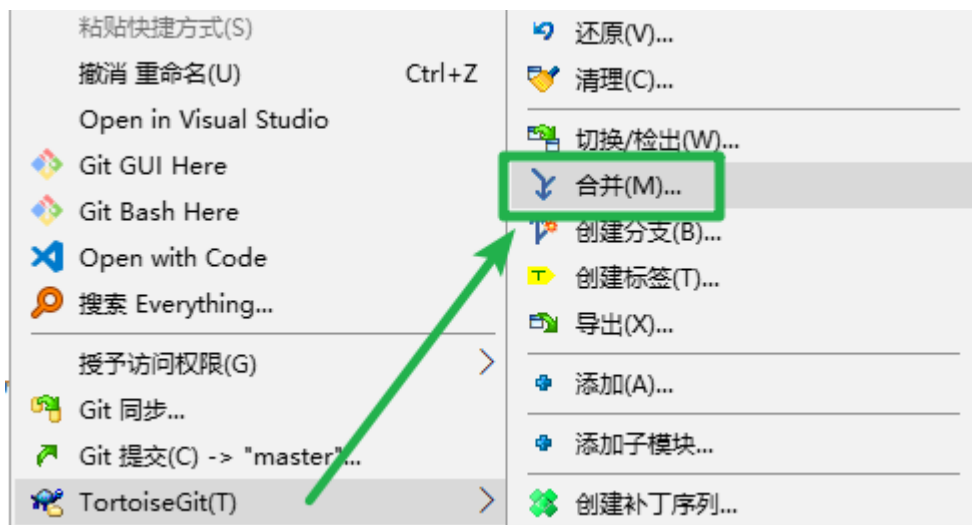
- 给创建的myfirst分支添加新文件, 并提交

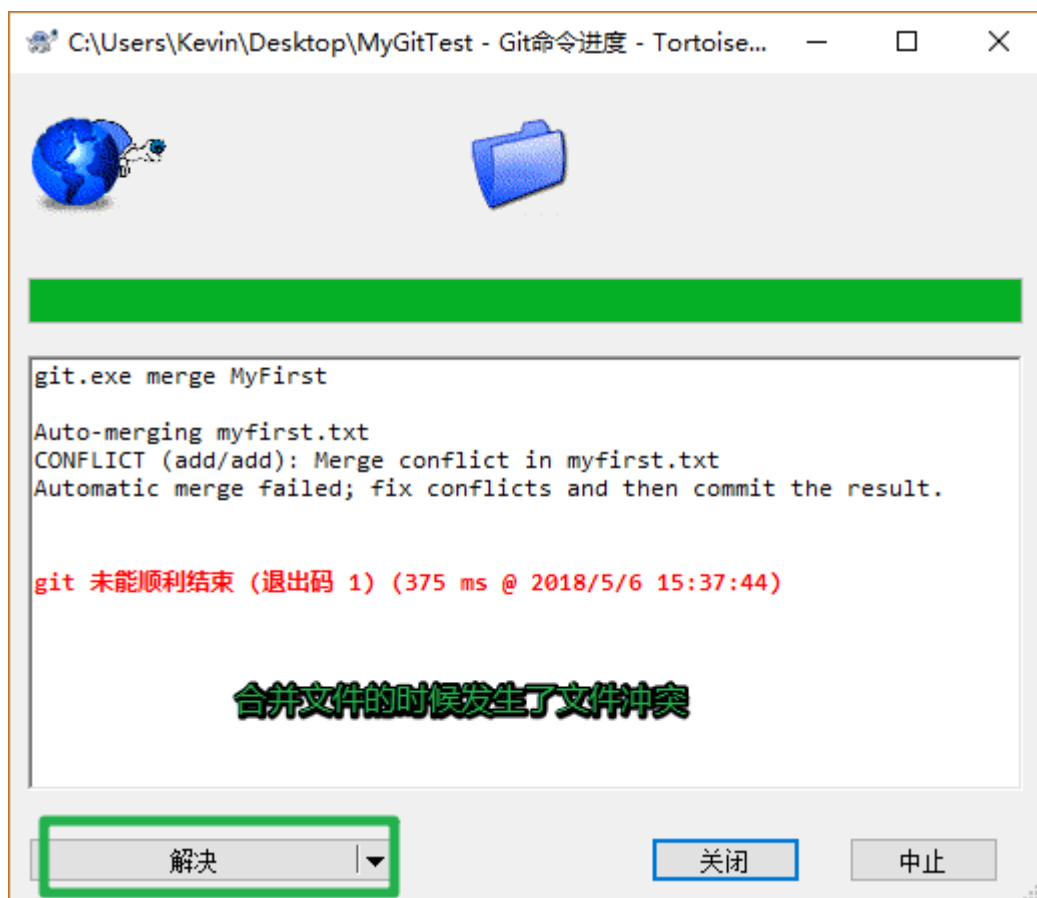
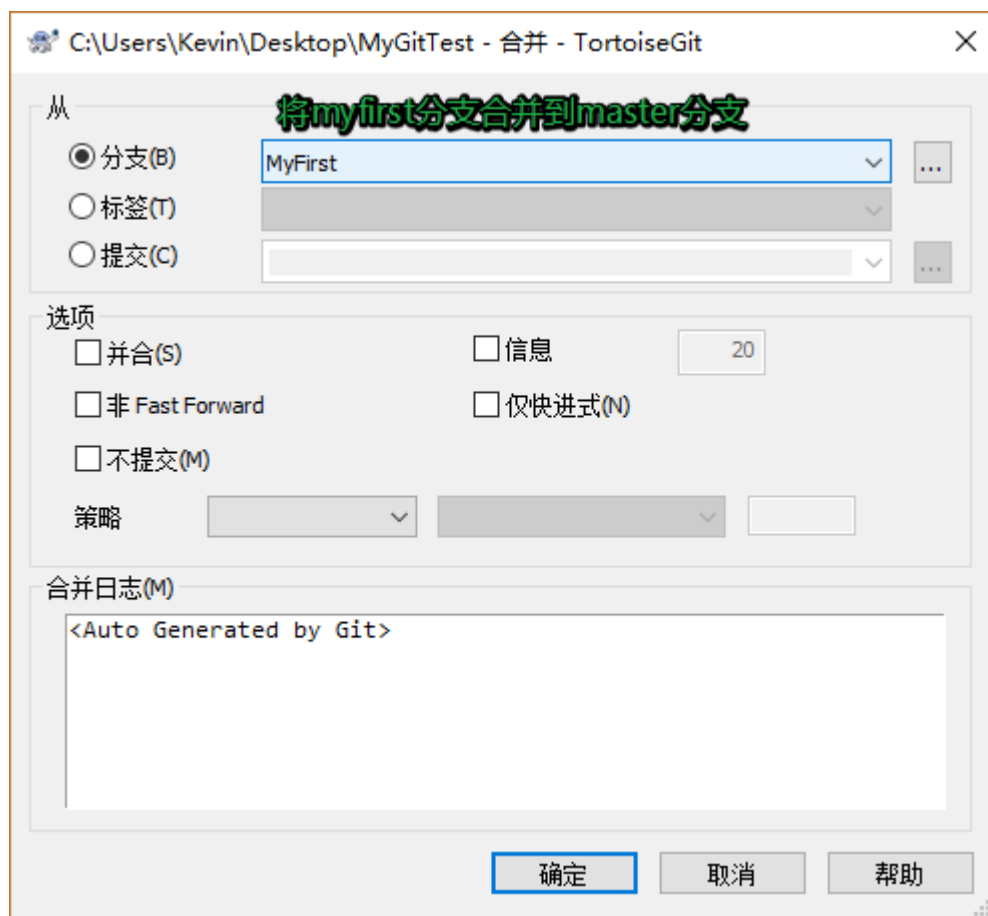


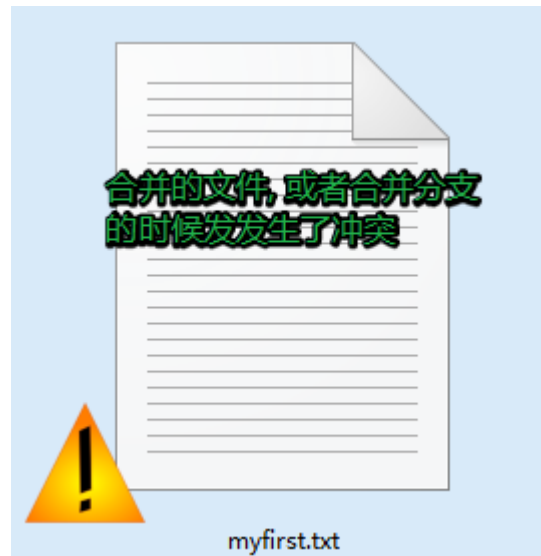
5.3 合并分支

将创建的myfirst分支合并到master分支上

1. 现在使用的分支必须是master分支
2. 选择合并分支





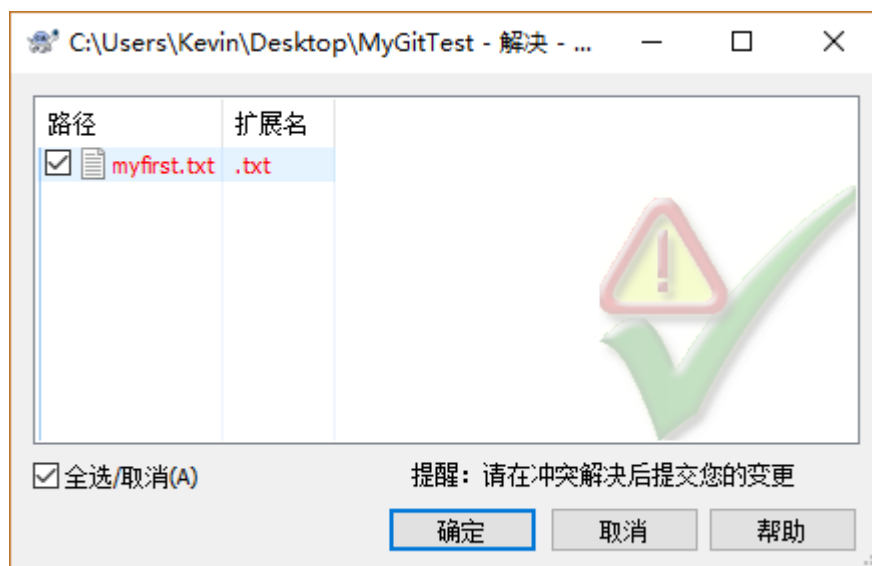


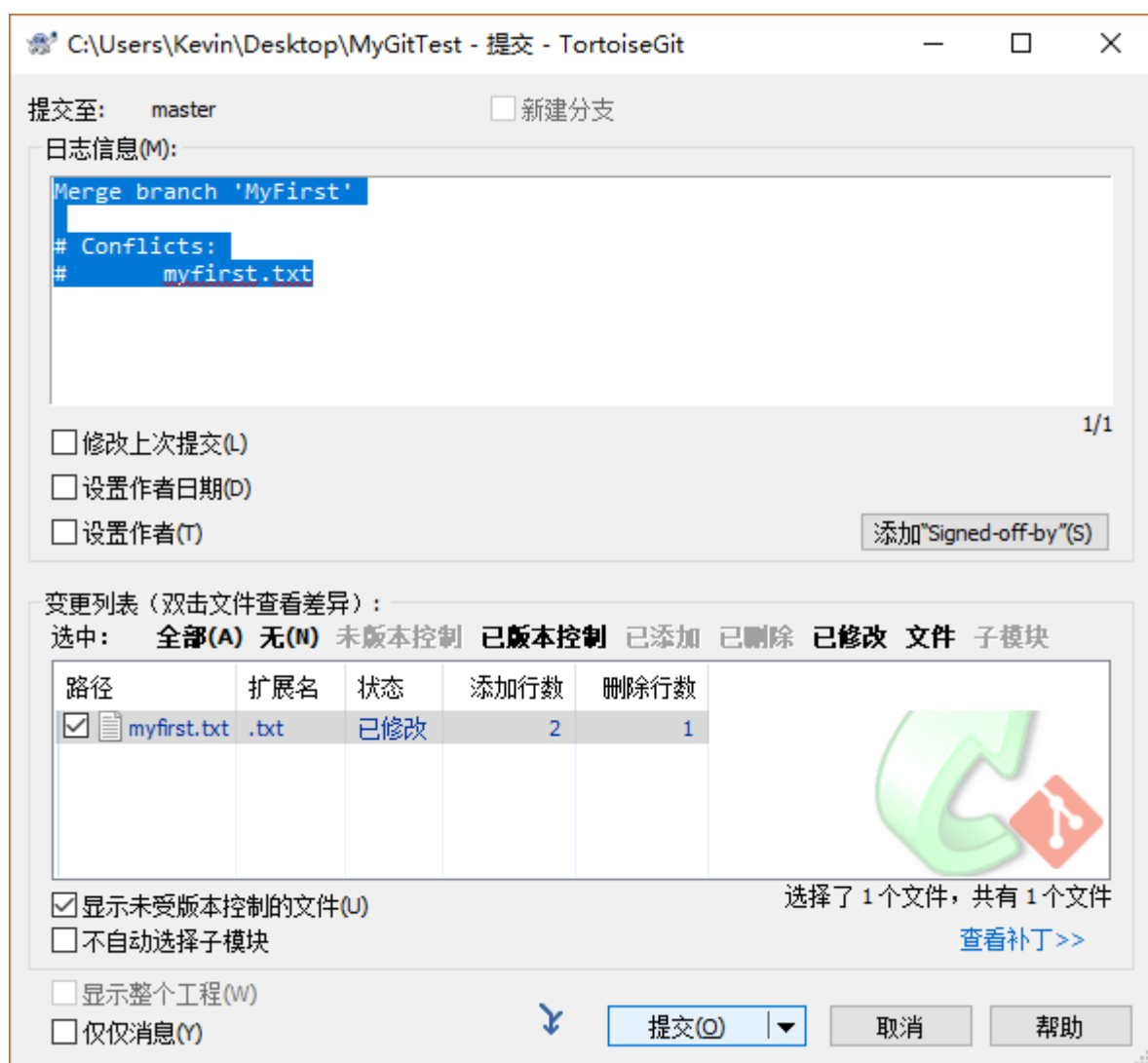
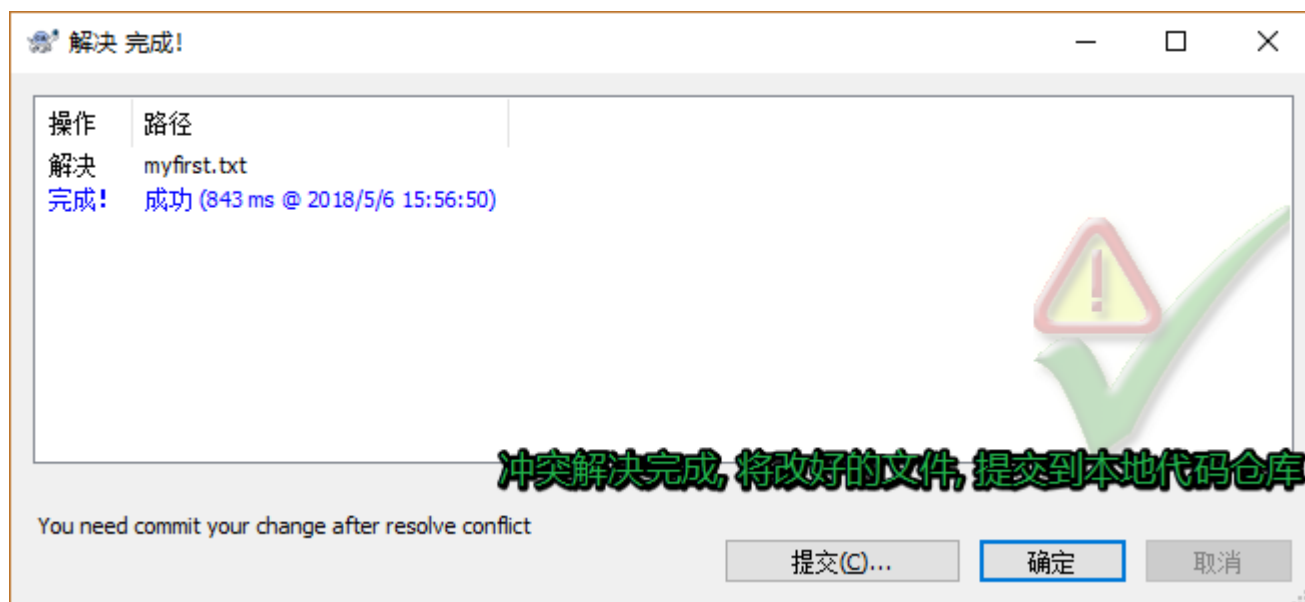
5.4 解决冲突

解决冲突的过程是手动去完成的

```
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
<<<<<<< HEAD 冲突的开始
我是master分支添加的文件
===== 分界线
我是myfirst分支的一个文件
>>>>>>> MyFirst 冲突的结束
```

在解决好的冲突文件上, 鼠标右键, 找解决冲突





TortoiseGit



看上去您的提交信息中有一个冲突提示（一行如 "# Conflicts:"）。此提示由 Git 为 cli 用户自动添加，不需要保留它。

您是否要忽略本警告并保留这些行，或者中止提交以编辑提交信息？

您可以在 TortoiseGit 设置中启用“剥离提交信息中以 # 开头的行”来自动移除这些行。

→ 忽略(I)

→ 放弃(A)

确定冲突改好了, 忽略本次提示

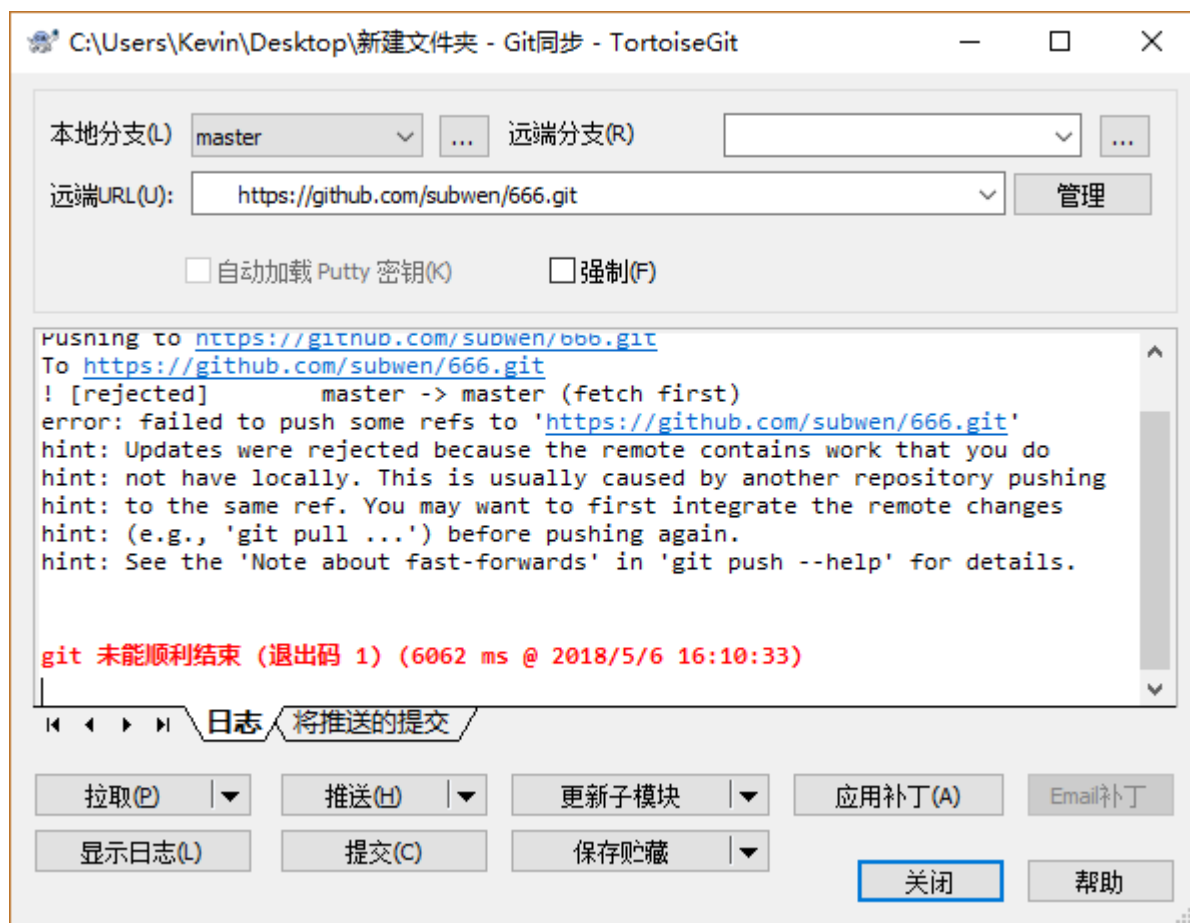
☐ 不再显示此消息(D)

6. 推送文件

将本地仓库的文件, 推动到远端仓库中

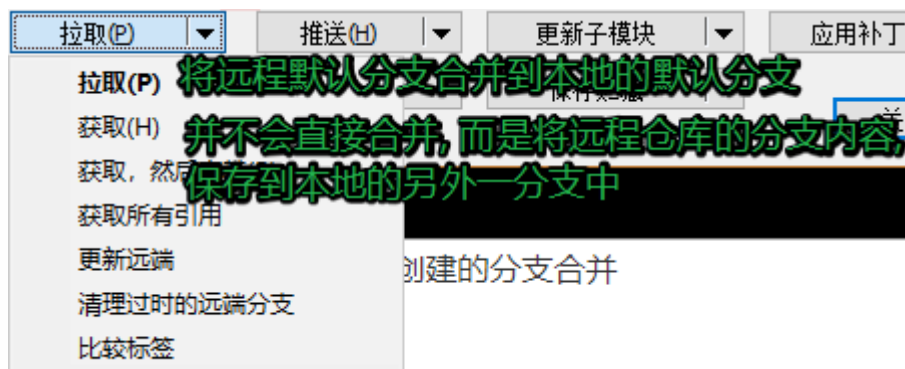
- 远端仓库并不是空仓库
- 远端仓库已经存在master分支

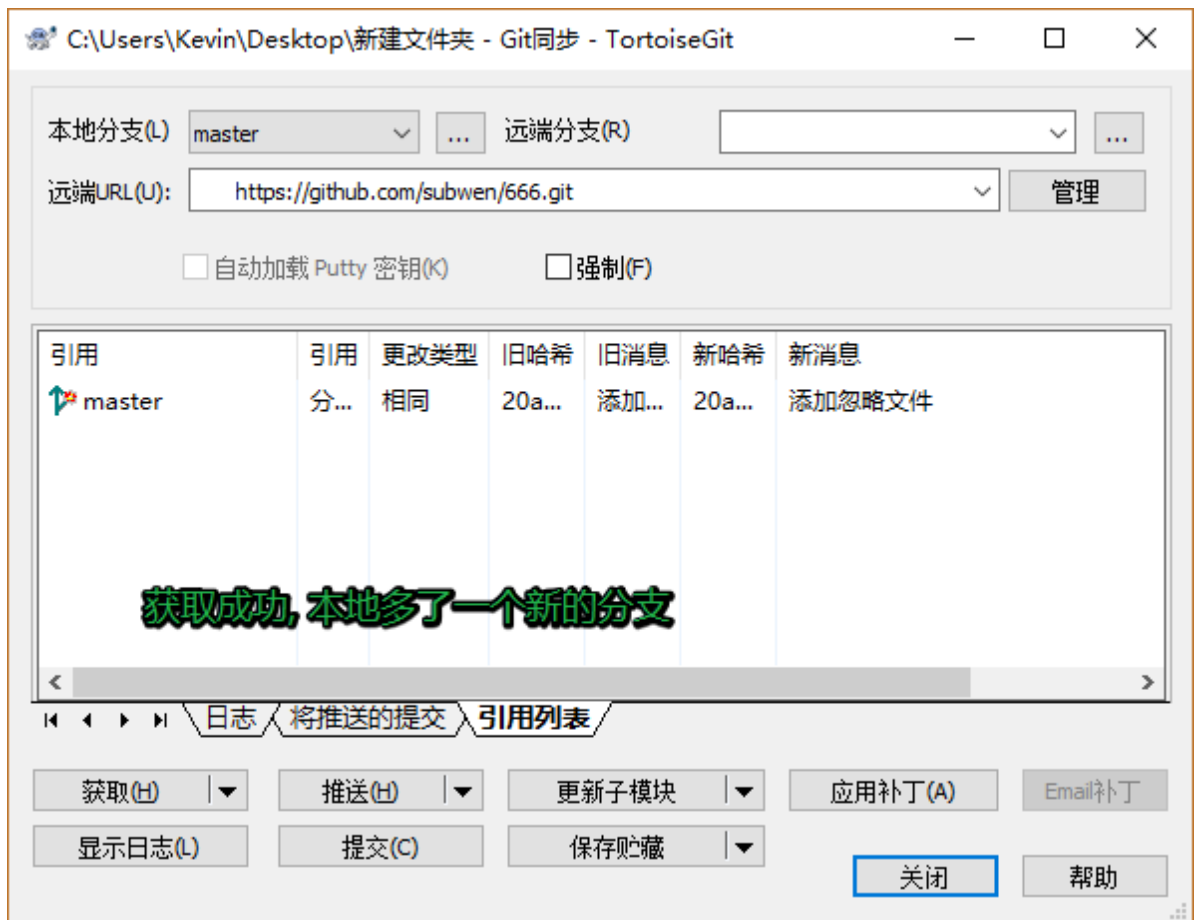
直接推送会出现错误, 如图:



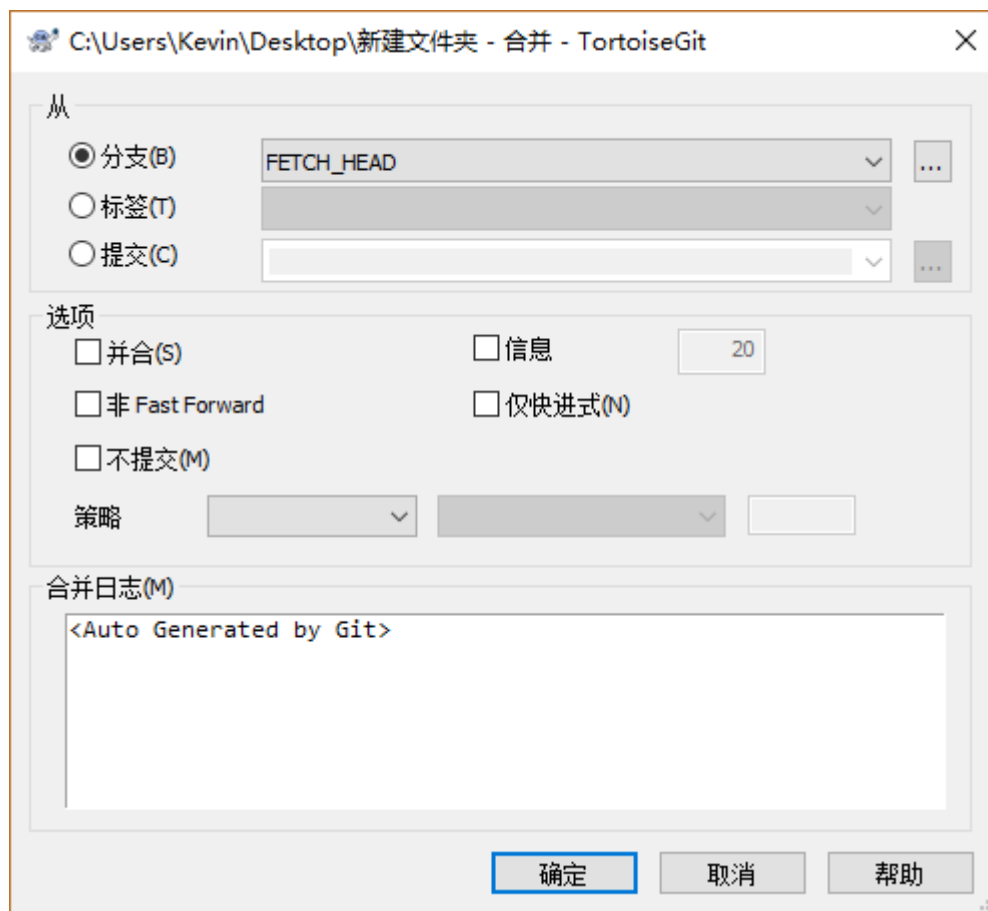
如何解决这个问题

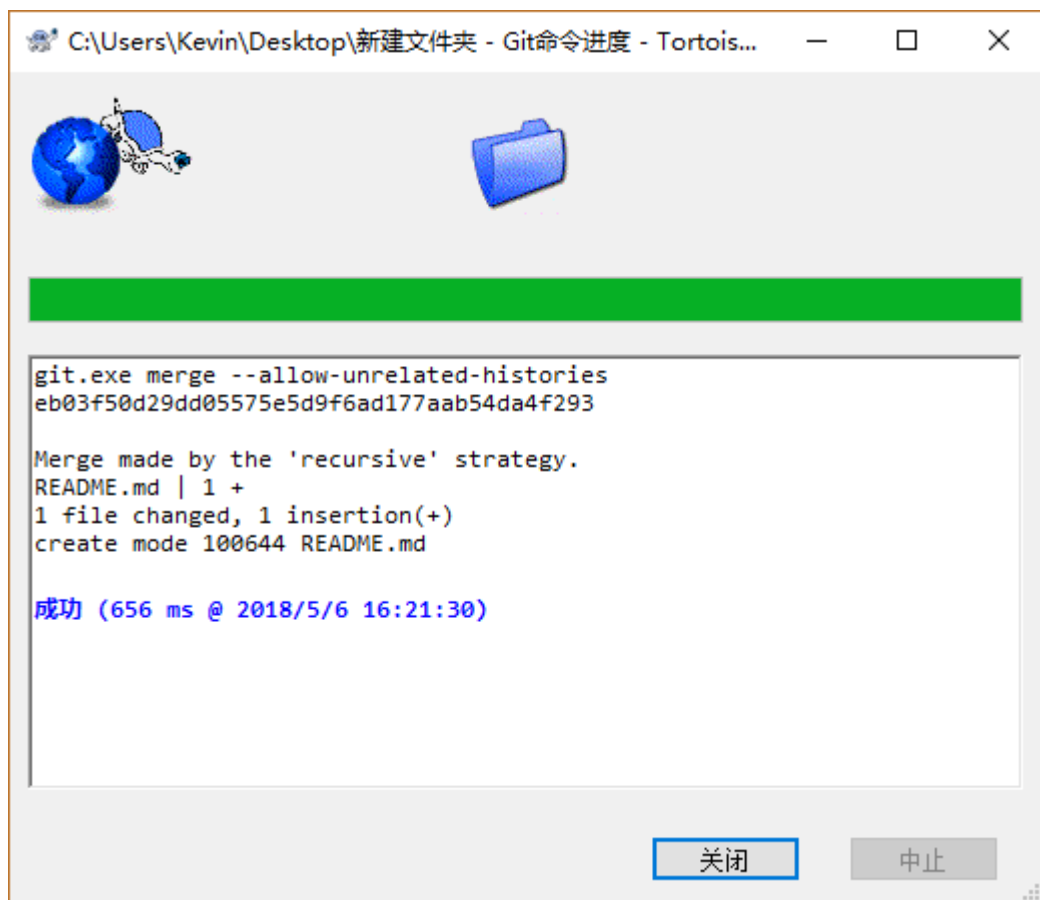
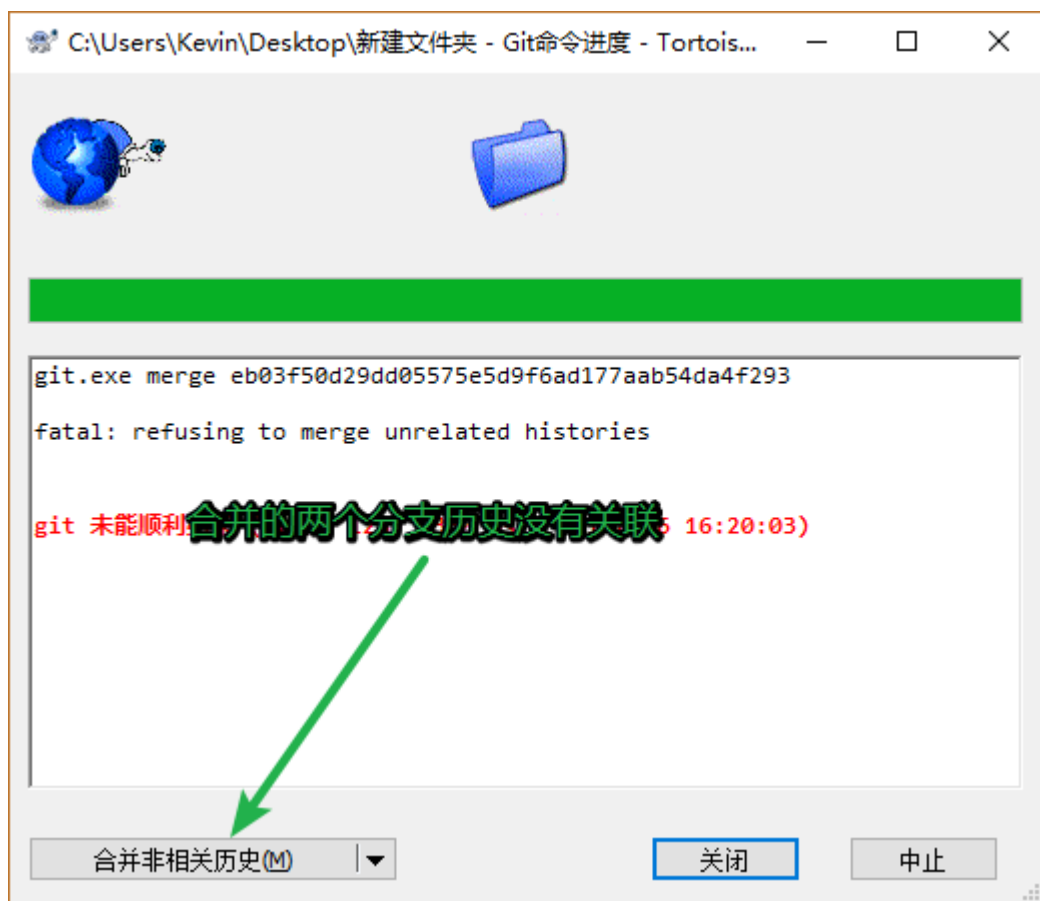
1. 将远程仓库分支中的内容获取到本地
 - o 额外创建了一个分支



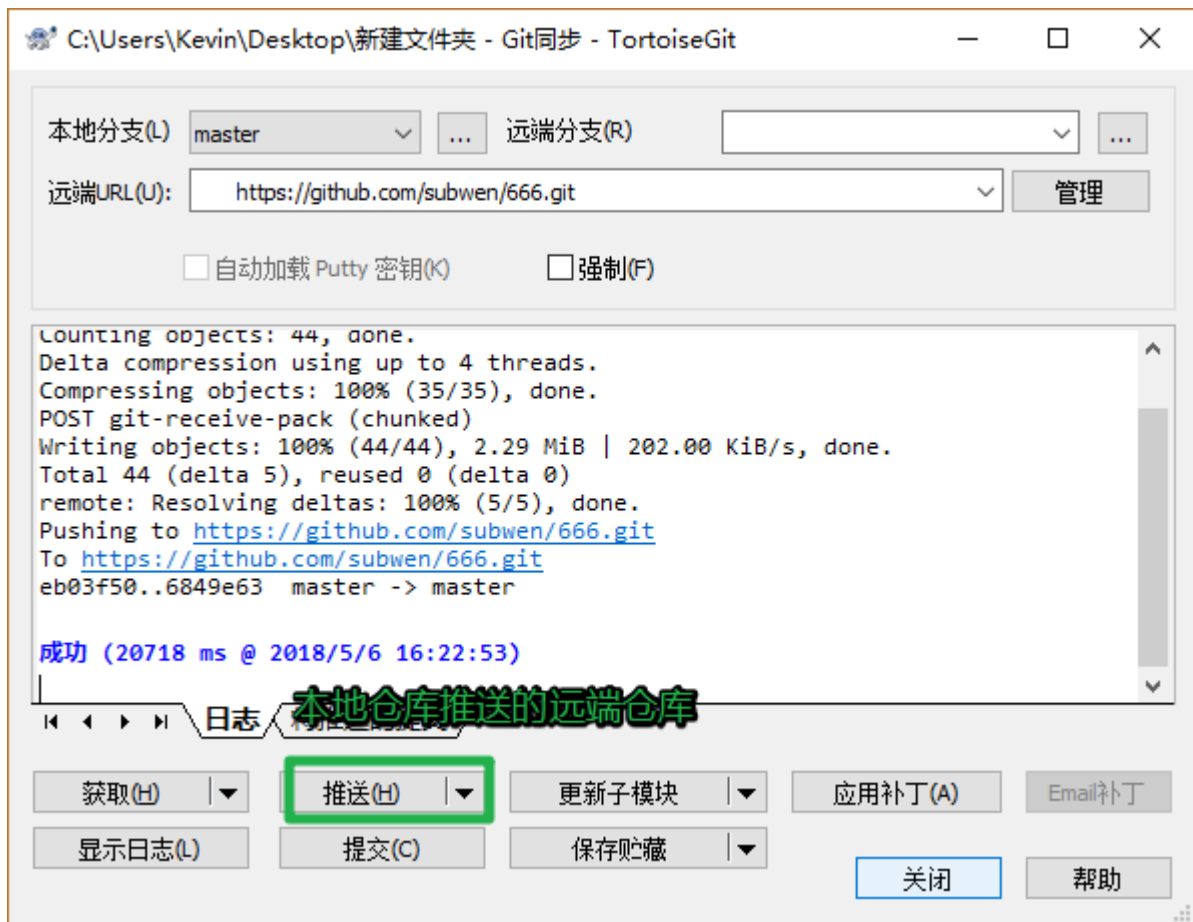


2. 让本地master分支和新创建的分支合并





3. 合并成功之后, 再推送



注意事项:

在向远端仓库推送文件之前, 必须先做拉取操作

- 冲突, 解决冲突
 - 提交 -> 推送
- 没有冲突
 - 在这个基础上修改, 或者直接推送