


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

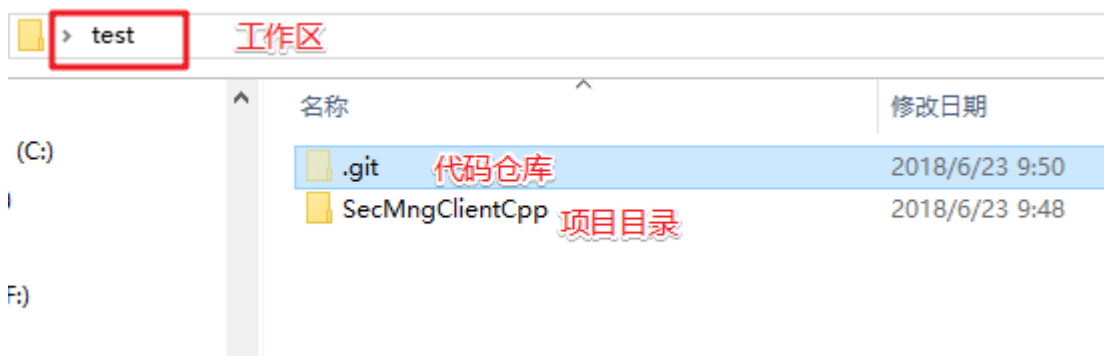
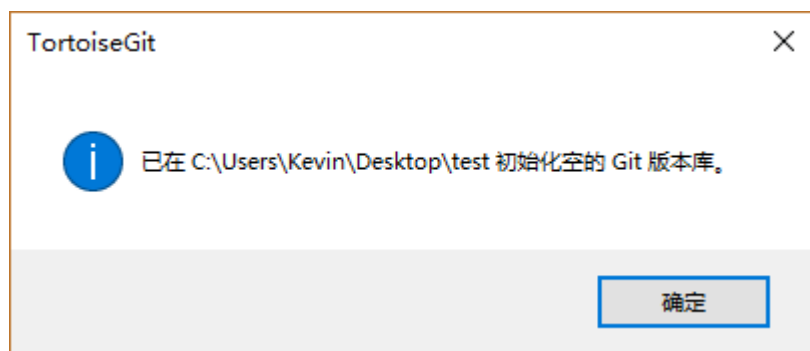
1. 创建本地版本库

1.1 创建本地版本库

创建一个目录如: test, 作为工作区, 在工作区中创建新版本库

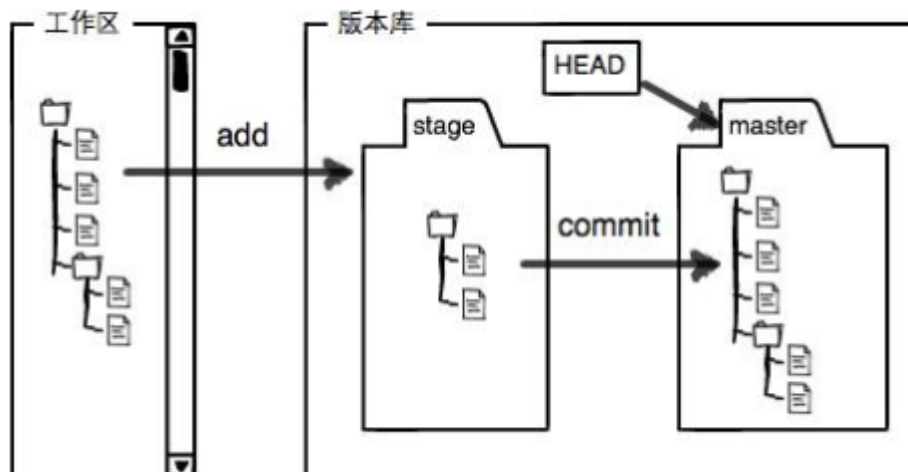
 Git 在这里创建版本库(Y)...





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: 存储项目文件的目录, 版本库需要创建到工作区中
- 版本库 - 创建出的隐藏目录 `.git`
 - stage - 暂存区
 - 当往工作区中添加了**新文件**之后, 需要将工作区文件添加到暂存区
 - 只需要做一次
 - master - 主分支
 - 默认只有这一个, 进行版本管理
 - HEAD - 操作master分支的指针

- 暂存区和分支的关系

- 当暂存区的文件内容发送变化, 需要将其提交的master分支
- 只有提交之后才会形成一个节点(一个版本)

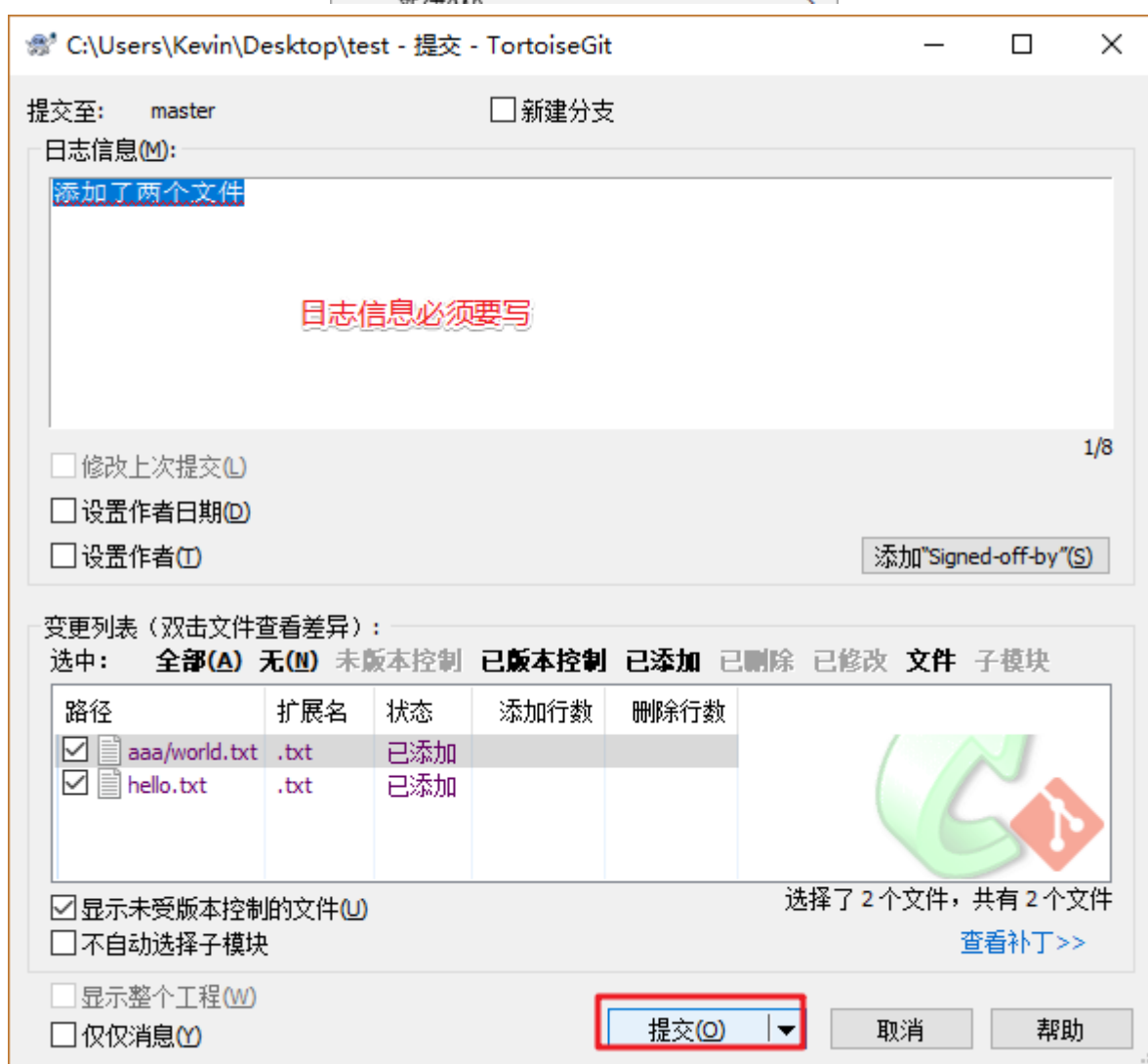
1.3 添加新文件到暂存区并提交

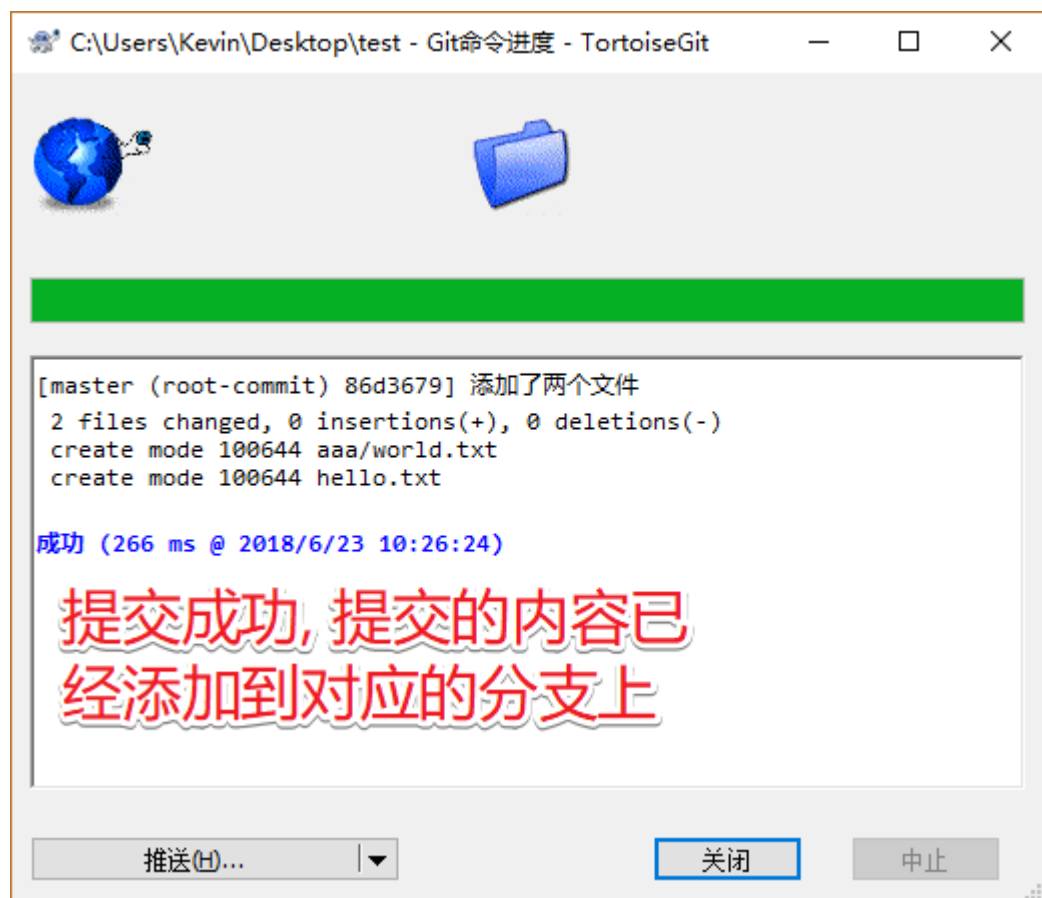
1. 在工作区创建一个新文件或者目录



后边直接next默认操作就可以了

2. 暂存区提交到分支





2. 修改文件

2.1 修改工作区文件

- 直接修改文件, 然后提交即可

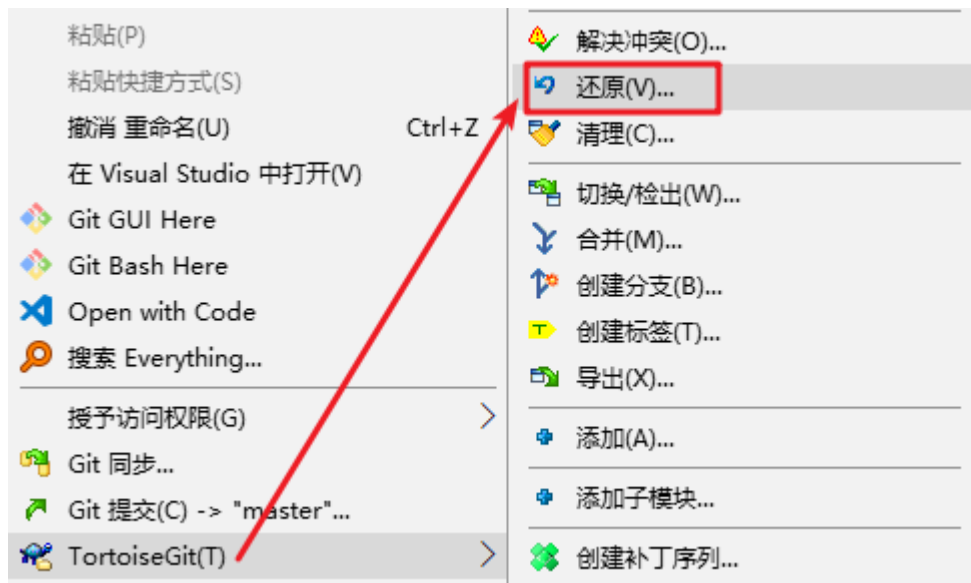
2.2 还原修改

1. 还原的本质:

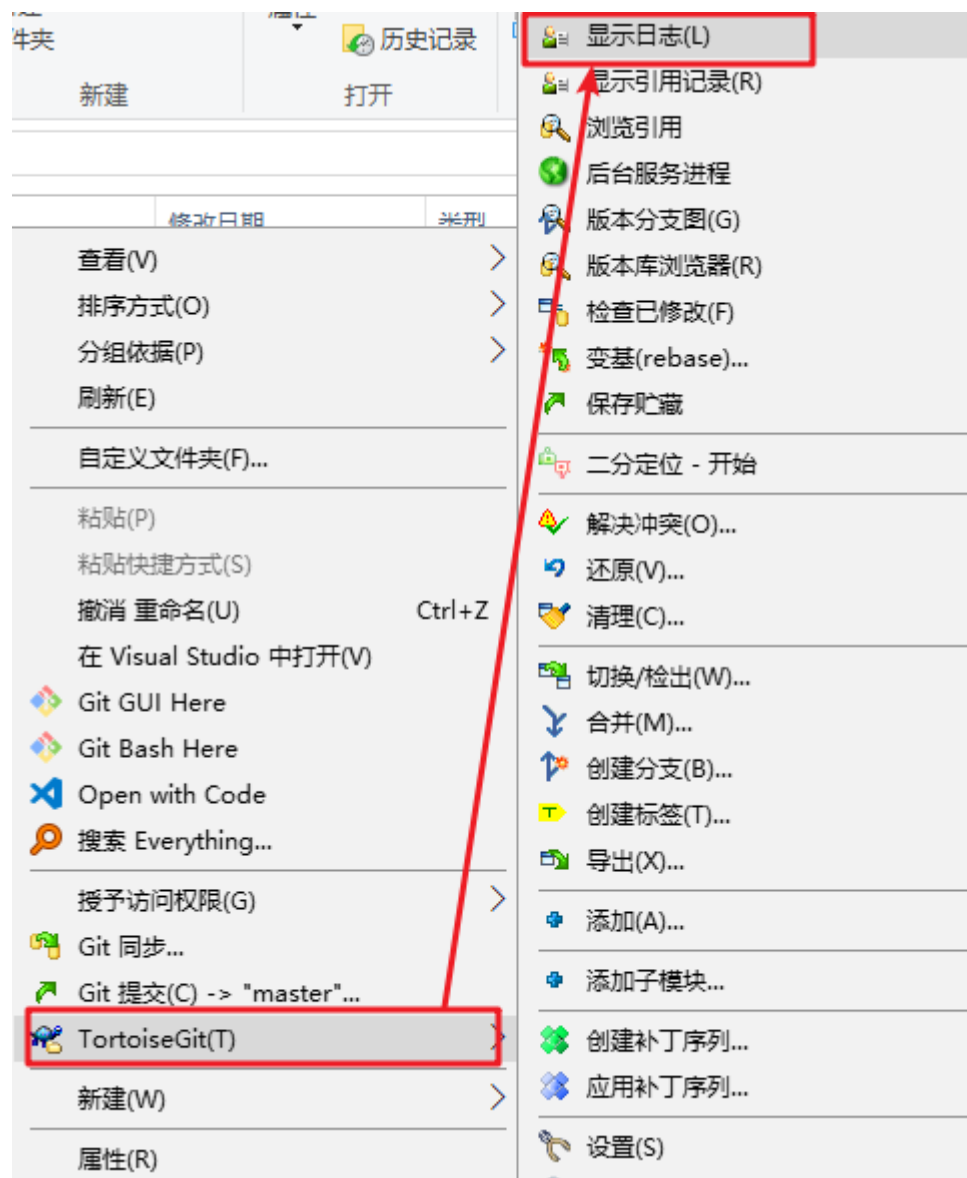
- 将工作区中修改的文件还原成最后一个提交的版本

2. 注意事项:

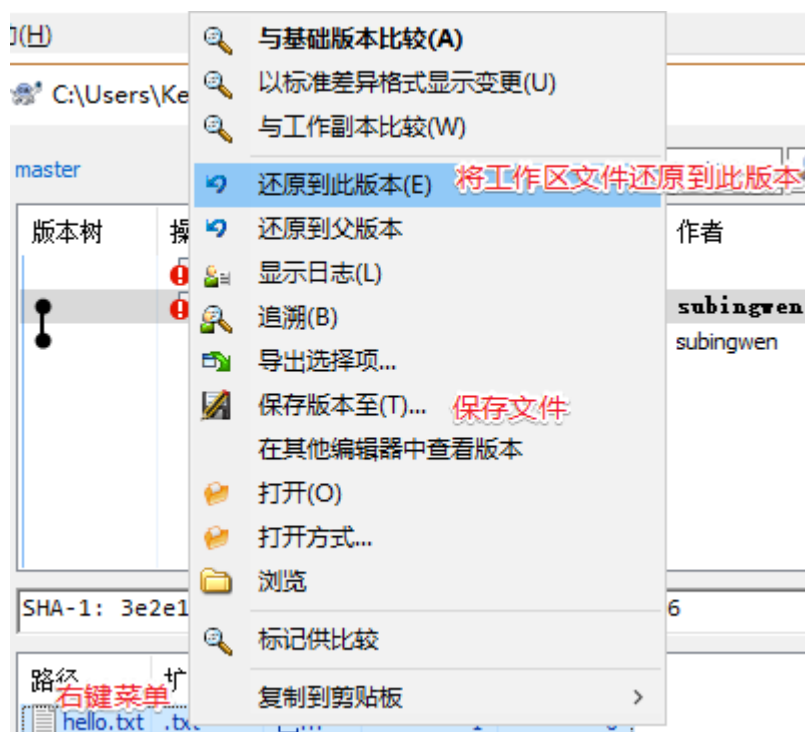
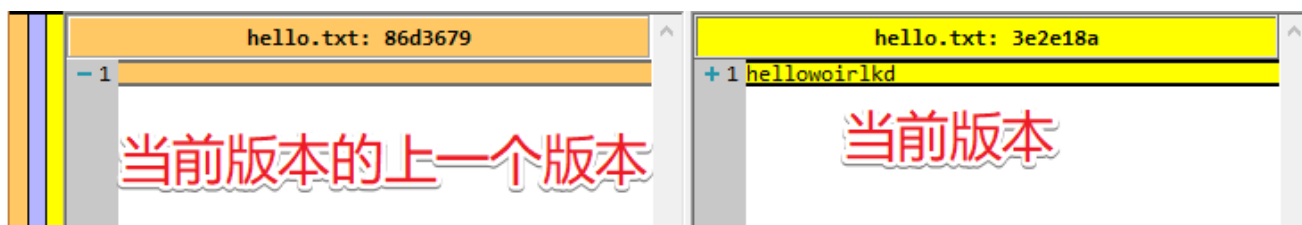
- 提交之前可以还原, 提交之后就无法通过该功能还原了
- 使用该还原功能之后, 文件内容就无法找回了



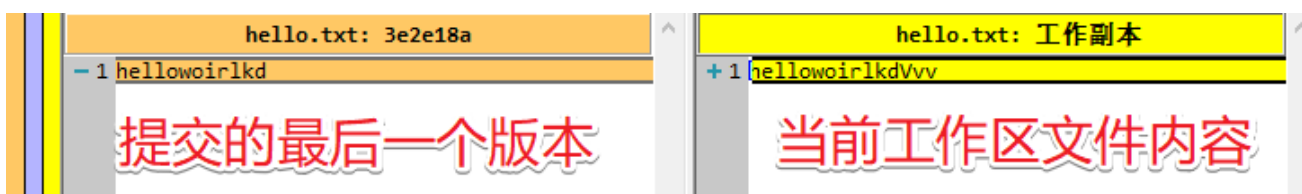
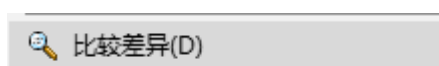
2.3 查看修改历史



版本的差异比较



2.4 差异比较



2.5 删除文件

1. 第一种删除方式:
 - o 直接使用del键删除
2. 使用删除功能

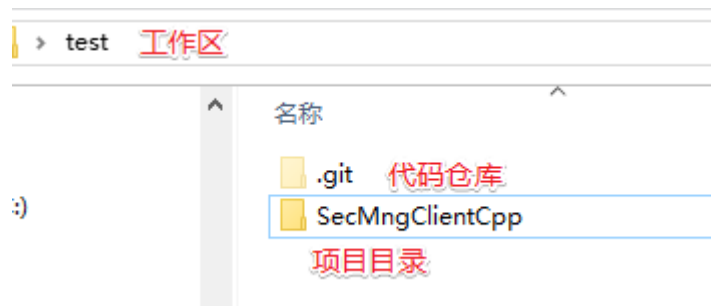


3. 注意事项:

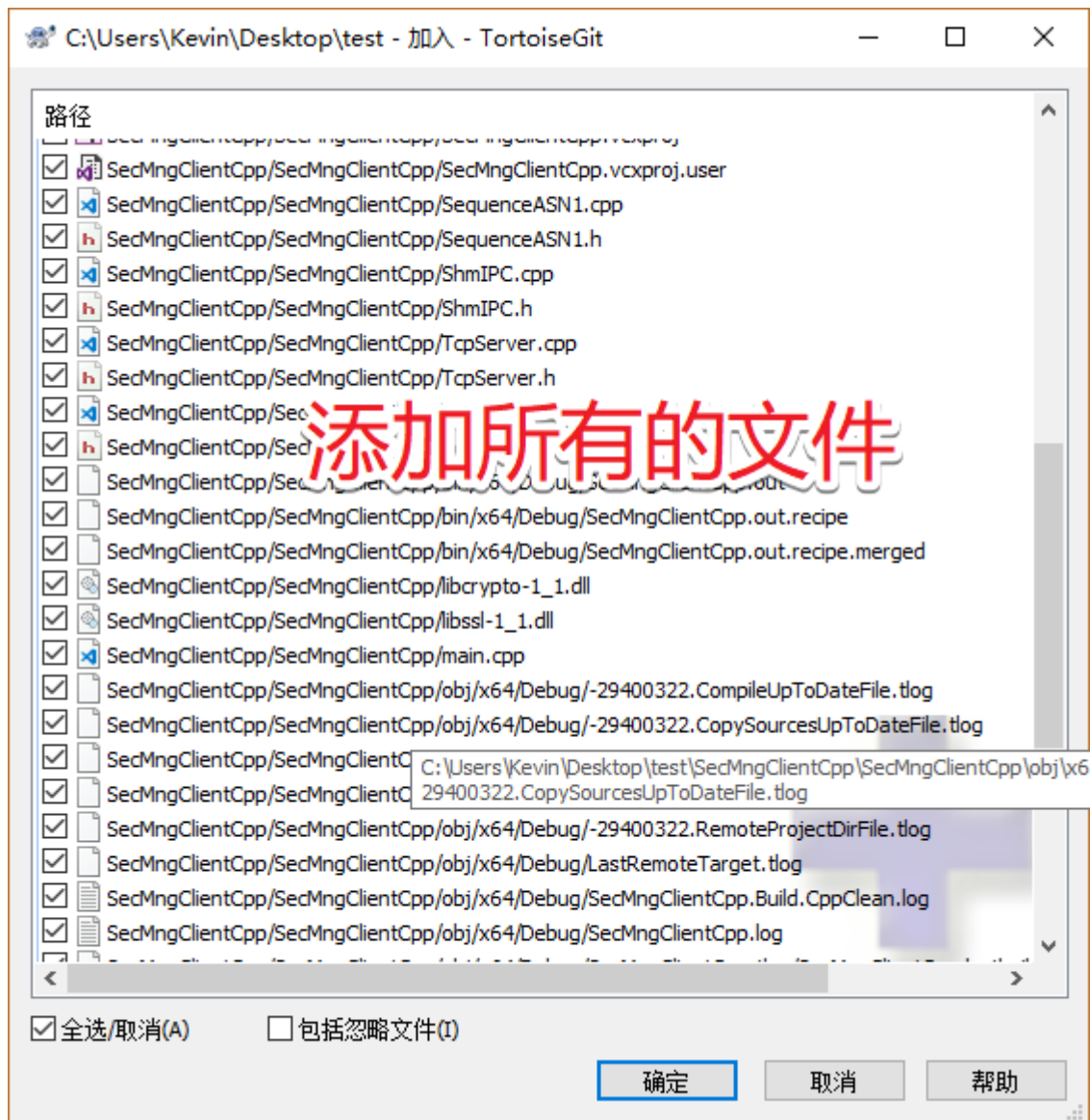
删除操作完成之后, 必须要提交

3. 案例: 添加一个本地项目到仓库

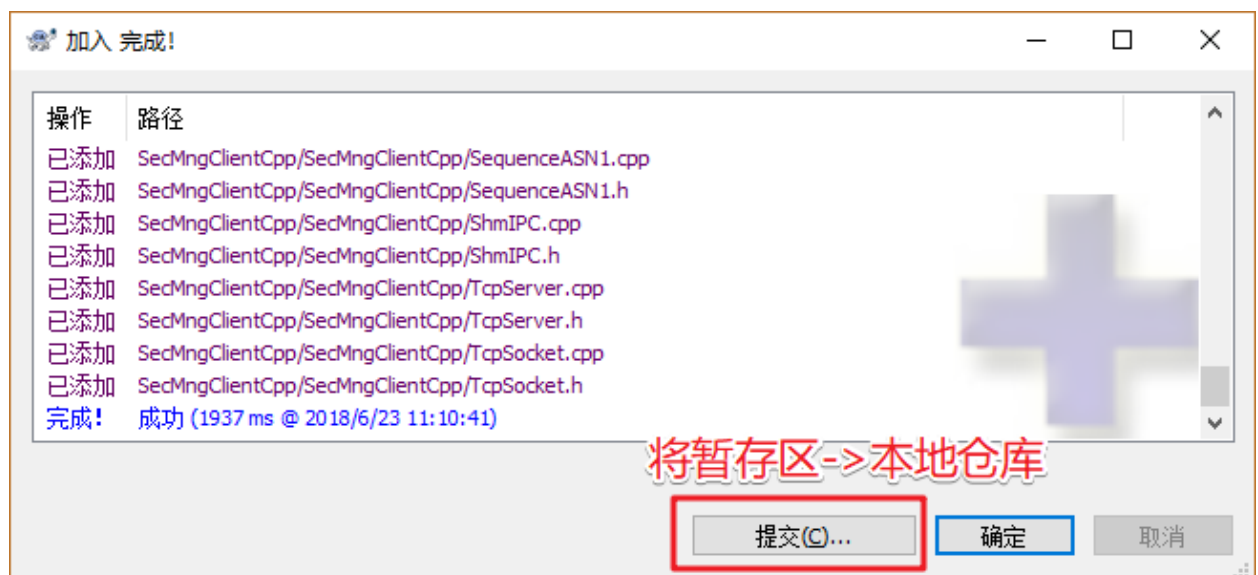
1. 创建一个工作区
2. 在工作区中添加一个代码仓库(版本库)
3. 将一个项目拷贝到工作区中



4. 将项目中所有的文件添加到暂存区

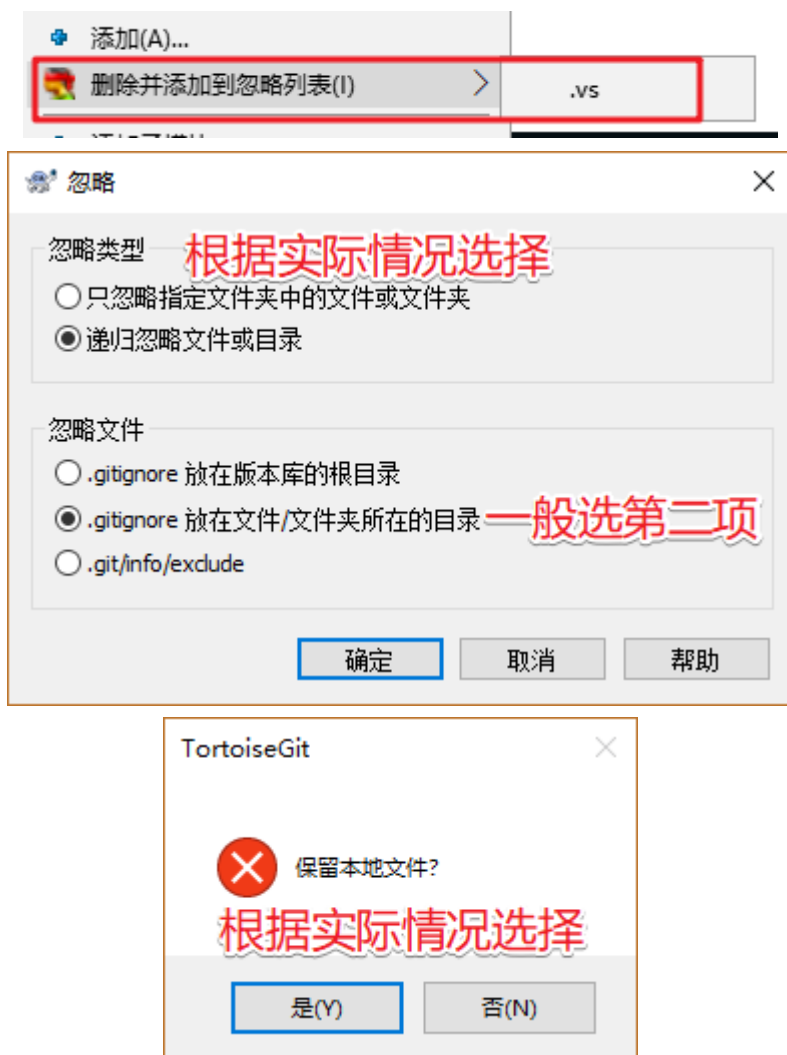


5. 提交



6. 设置文件忽略

在要忽略的文件上右键



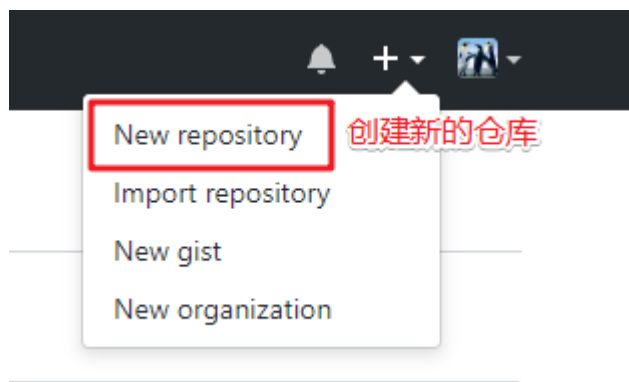
7. 提交修改

4. 添加远程仓库


在线代码托管平台

- Github: <https://github.com/>
- 码云: <https://gitee.com/>

4.1 远程仓库的添加和创建





Owner Repository name

 subwen / MyTest **仓库名** ✓

Great repository names are short and memorable. Need inspiration? How about [verbose-journey](#).

Description (optional)

可选项 - 描述

- ☒  **Public**
Anyone can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. **不建议** you're importing an existing repository.

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

Create repository

快速设置 - 如果你以前做过这种事情 **访问该仓库的两种方式**

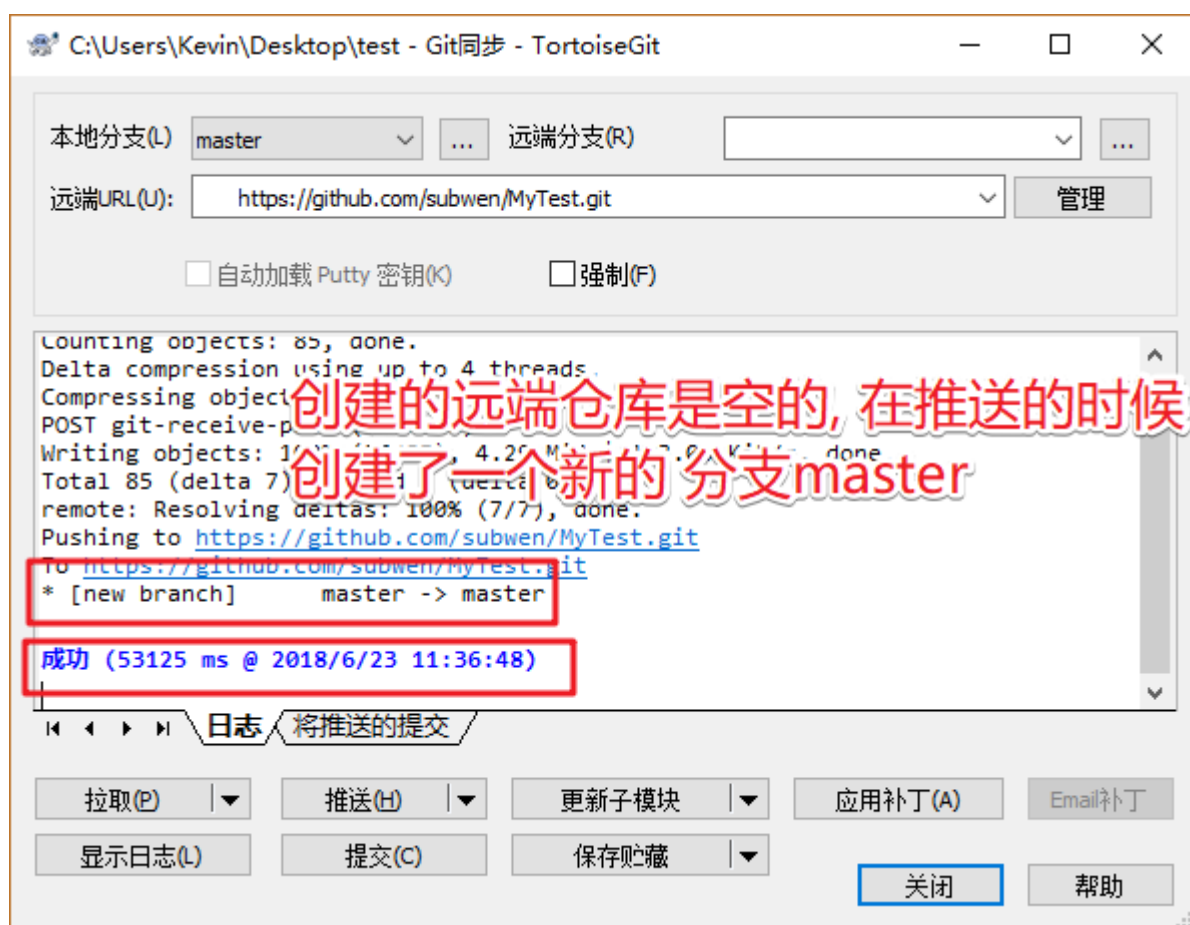
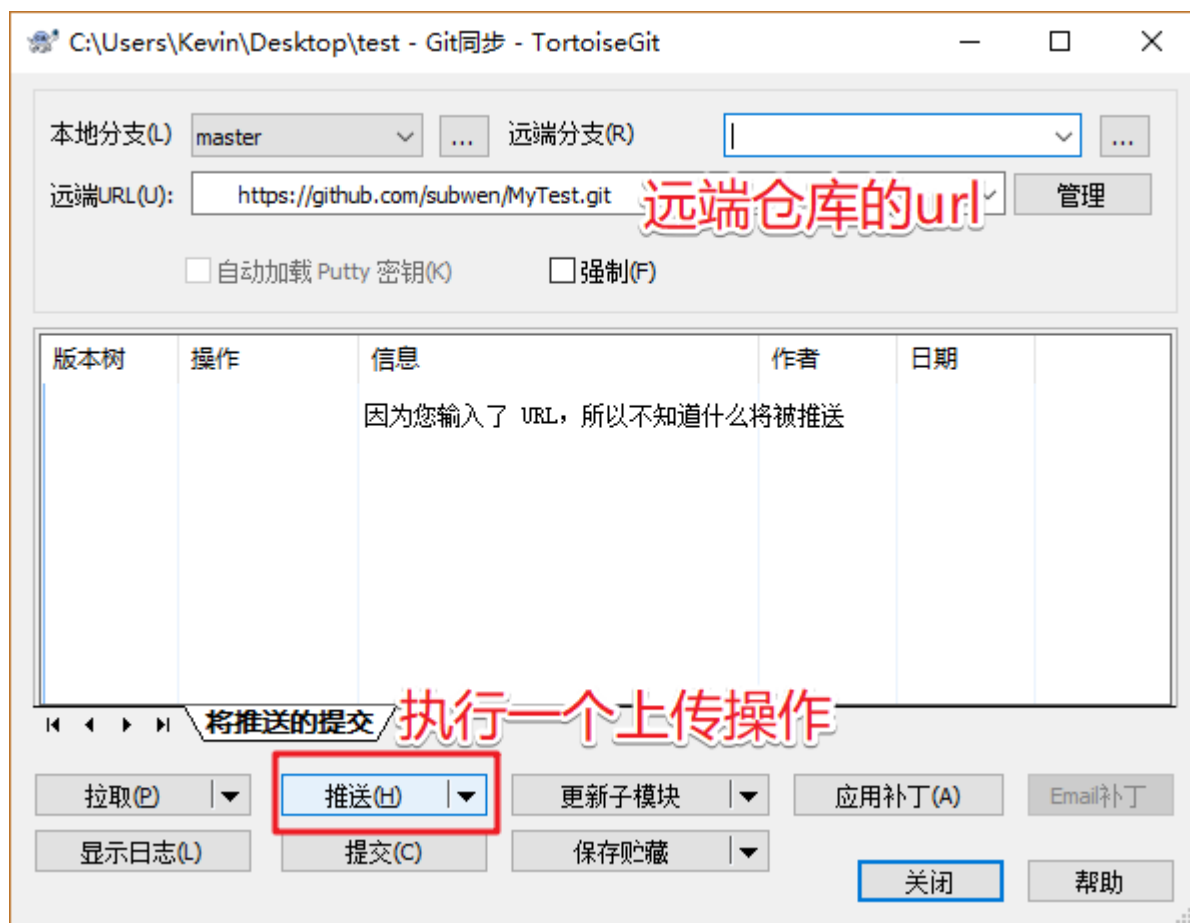
 在桌面上设置 要么 **HTTPS** **SSH** <https://github.com/subwen/MyTest.git>

我们建议每个存储库都包含一个 [README](#), [LICENSE](#)和[.gitignore](#)。

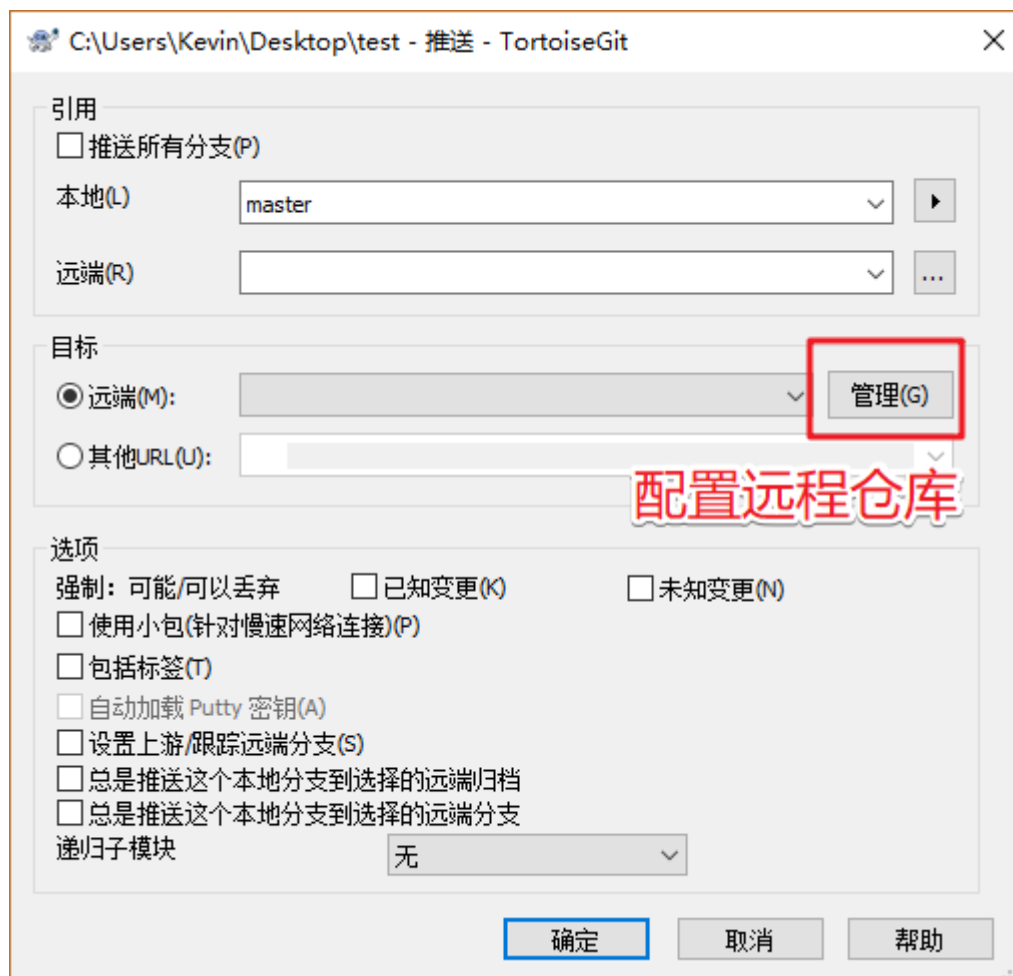
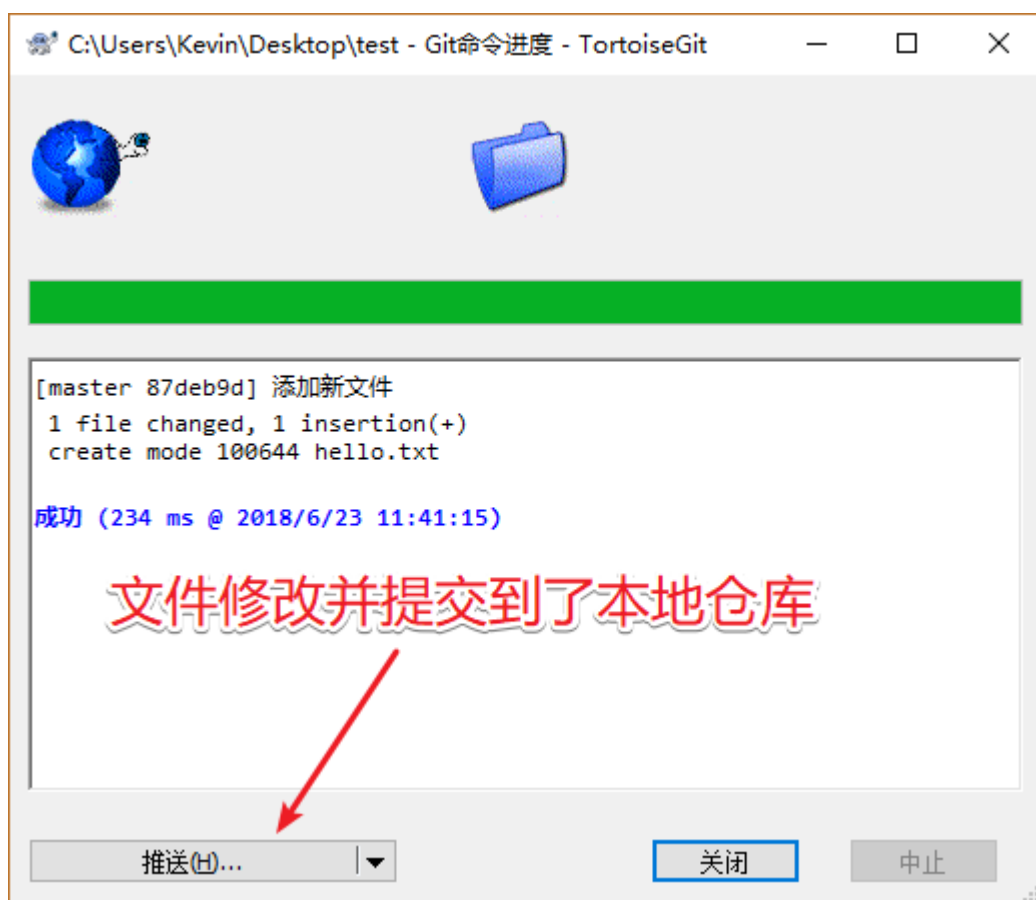
4.2 本地仓库同步到远程仓库

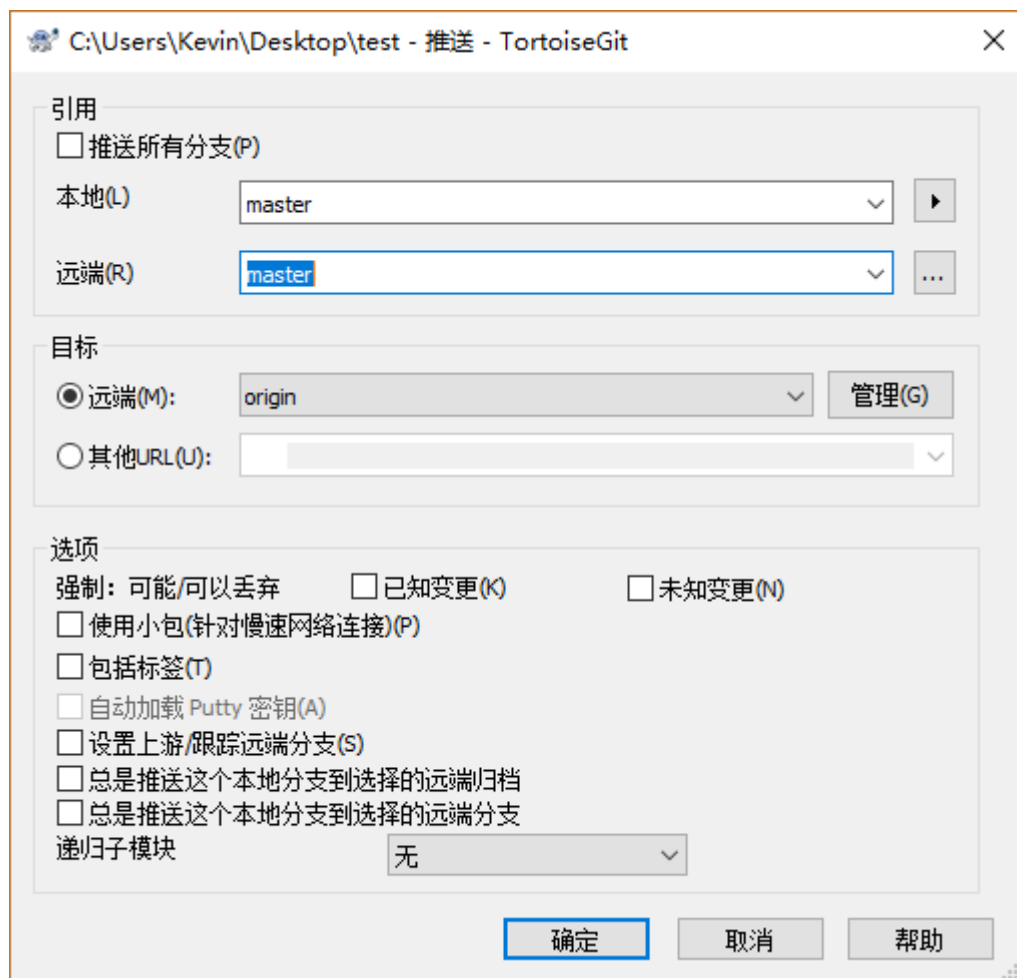
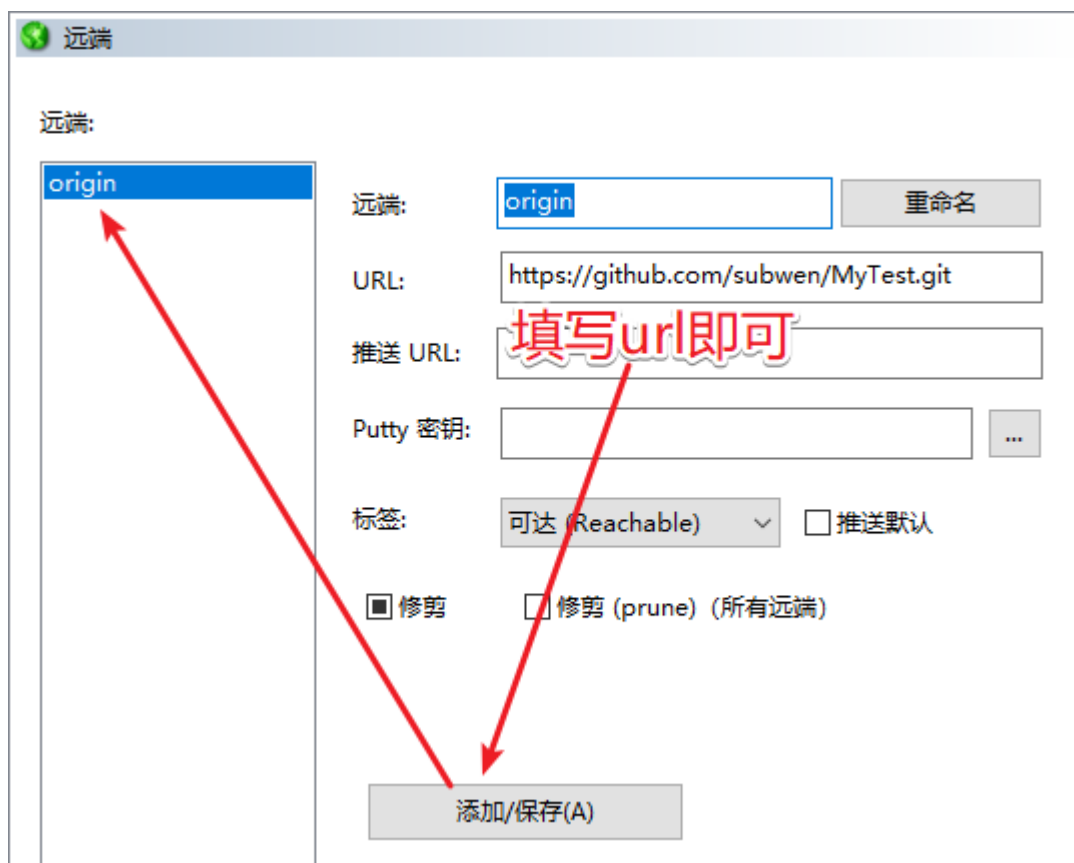
- 前提条件:
 - 创建的远程仓库是空的, 意味着还没有master分支

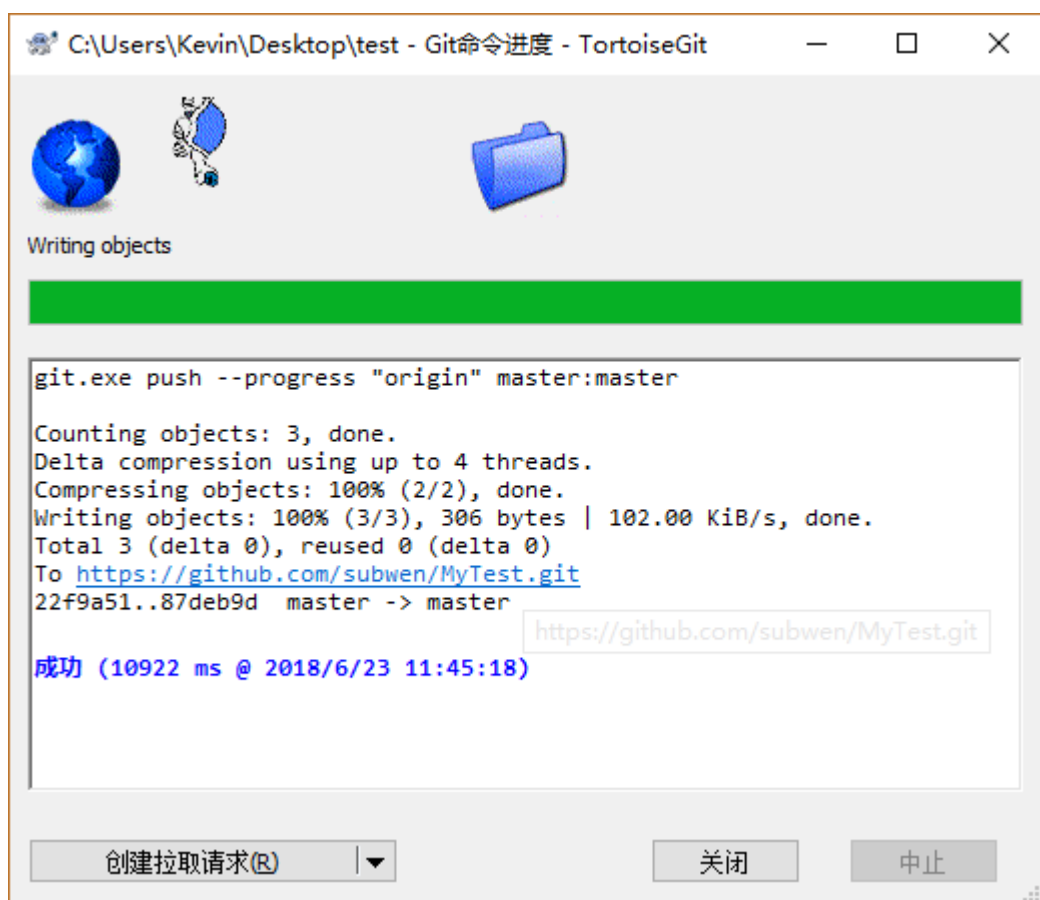
 Git 同步...



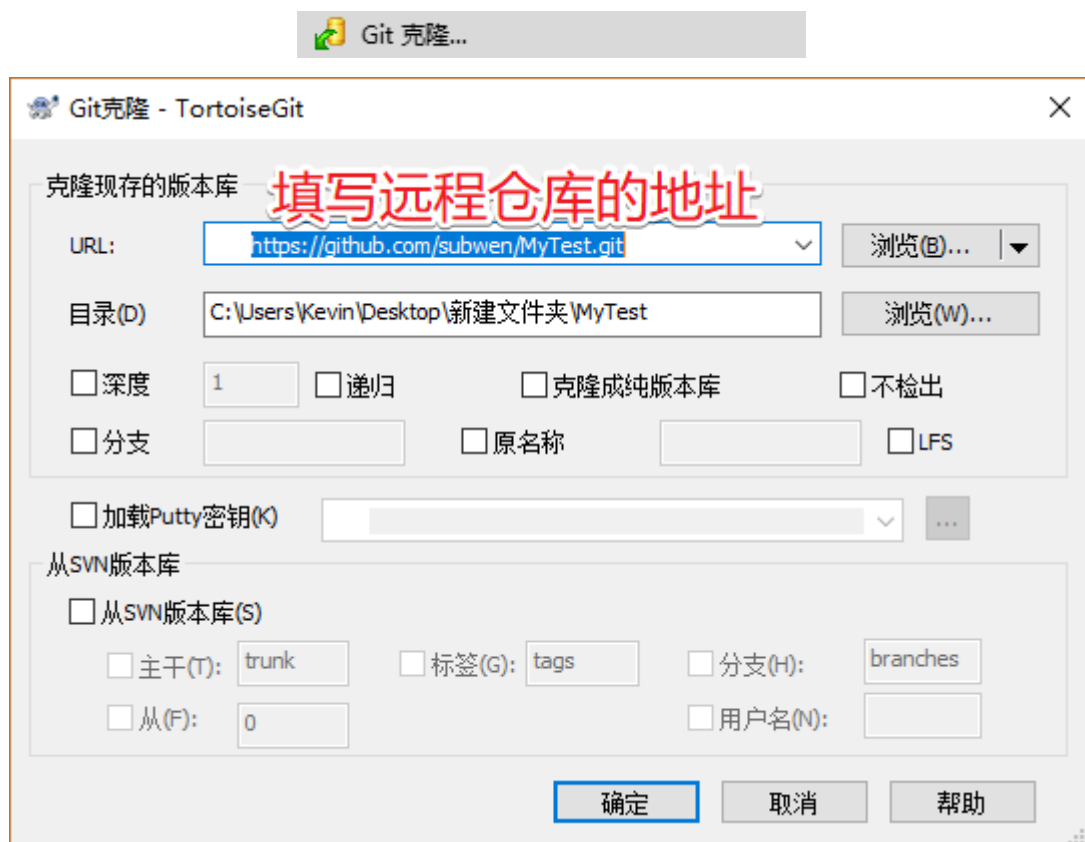
2. 修改了本地文件 -> 同步到远端仓库



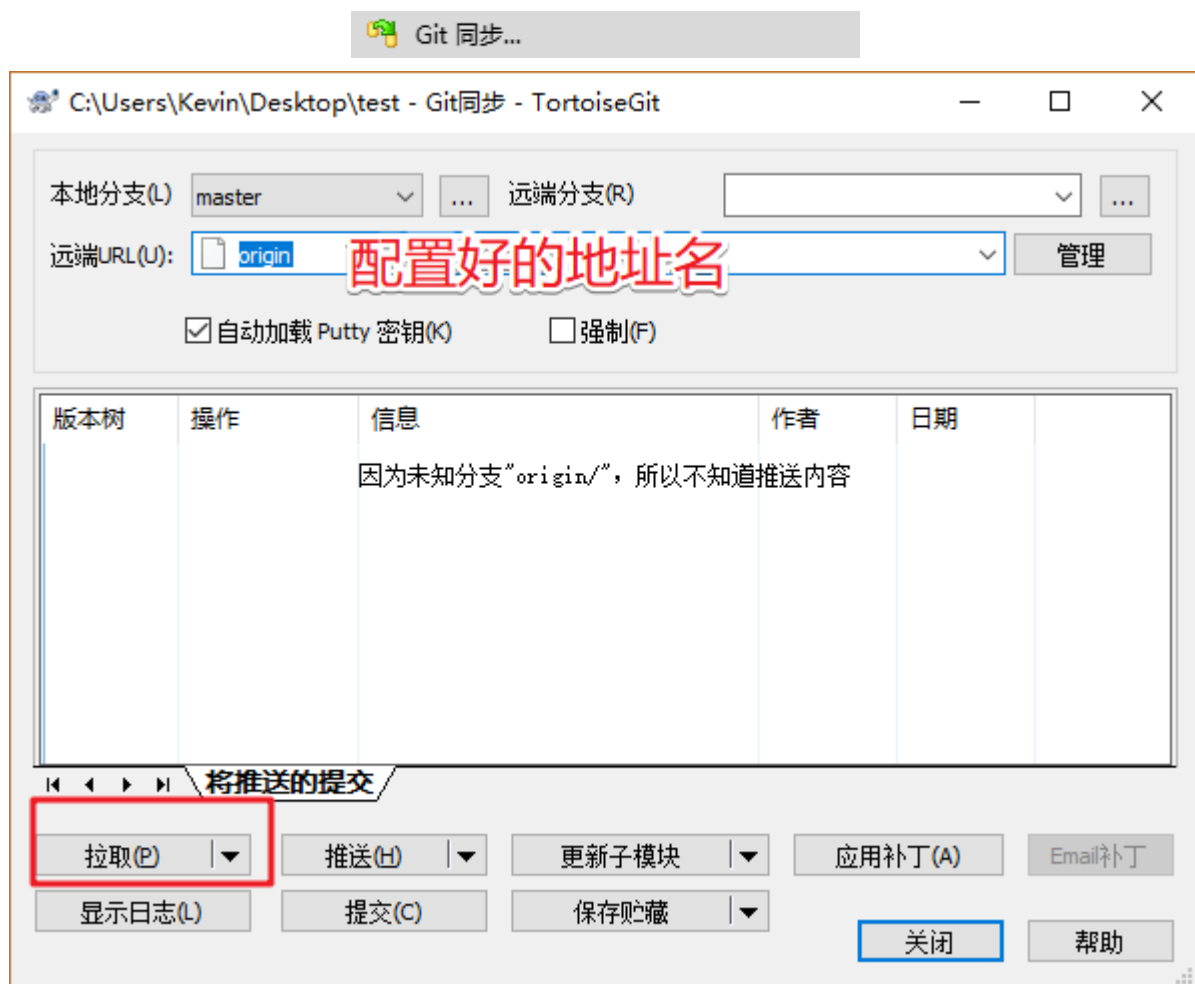




4.3 克隆远程仓库到本地



1. 从远程仓库将某个文件更新到本地



4.4 ssh 配置

1. 加密的方式是非对称加密, 需要生成一密钥对
 - o 客户端拿私钥
 - o github服务器拿公钥
2. 生产密钥的命令

```
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):
Created directory '/c/Users/Kevin/.ssh'.
Enter passphrase (empty for no passphrase): 直接回车*3
Enter same passphrase again:
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:Kea+0y15Ci5B8hytu1j0y9DMoBk6QkCRPh5IHYqxdqQ Kevin@X1-Carbon-2017
The key's randomart image is:
+---[RSA 2048]-----+
|.O+O.|
|o=+|.
|BE . .
|++o o . .
|o.+ = oo S
|o= ==o .
|* . =oo. o
|.. Boo..+ o
| . *++o.+
+----[SHA256]-----+
Kevin@X1-Carbon-2017 MINGW64 ~/Desktop
$

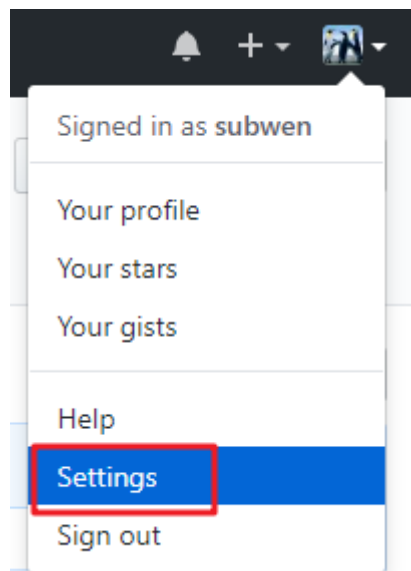
```

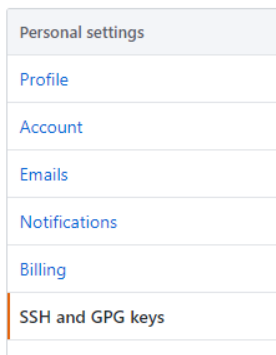
密钥对生成成功了

密钥对的位置:

```
1 | c:\user\用户名\ssh
```

将公钥添加到github上





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](#).

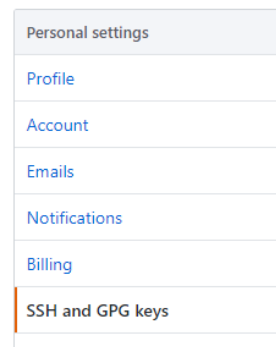
可以通过这个链接查看生成密钥对的命令

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 keys

New SSH key

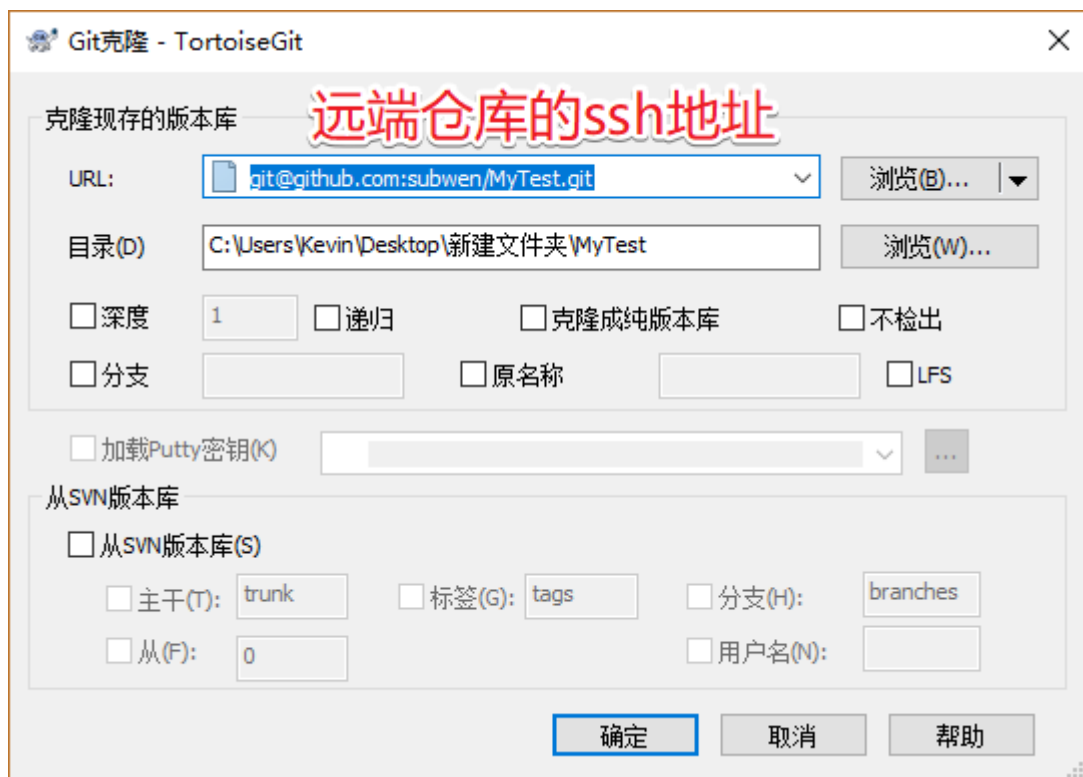
This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

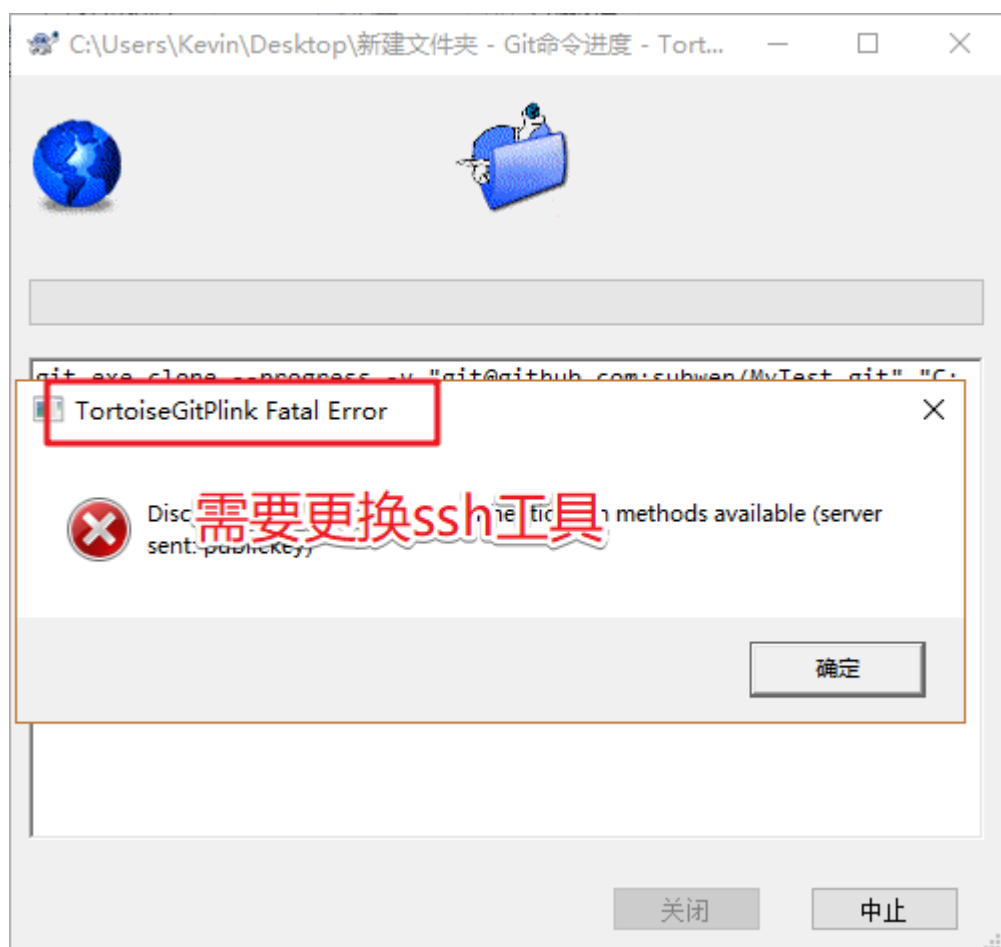

SSH

mykey
Fingerprint: 6a:88:6e:93:47:4a:13:d5:0a:e8:40:e9:f2:40:10:64
Added on 23 Jun 2018
Never used — Read/write

Delete

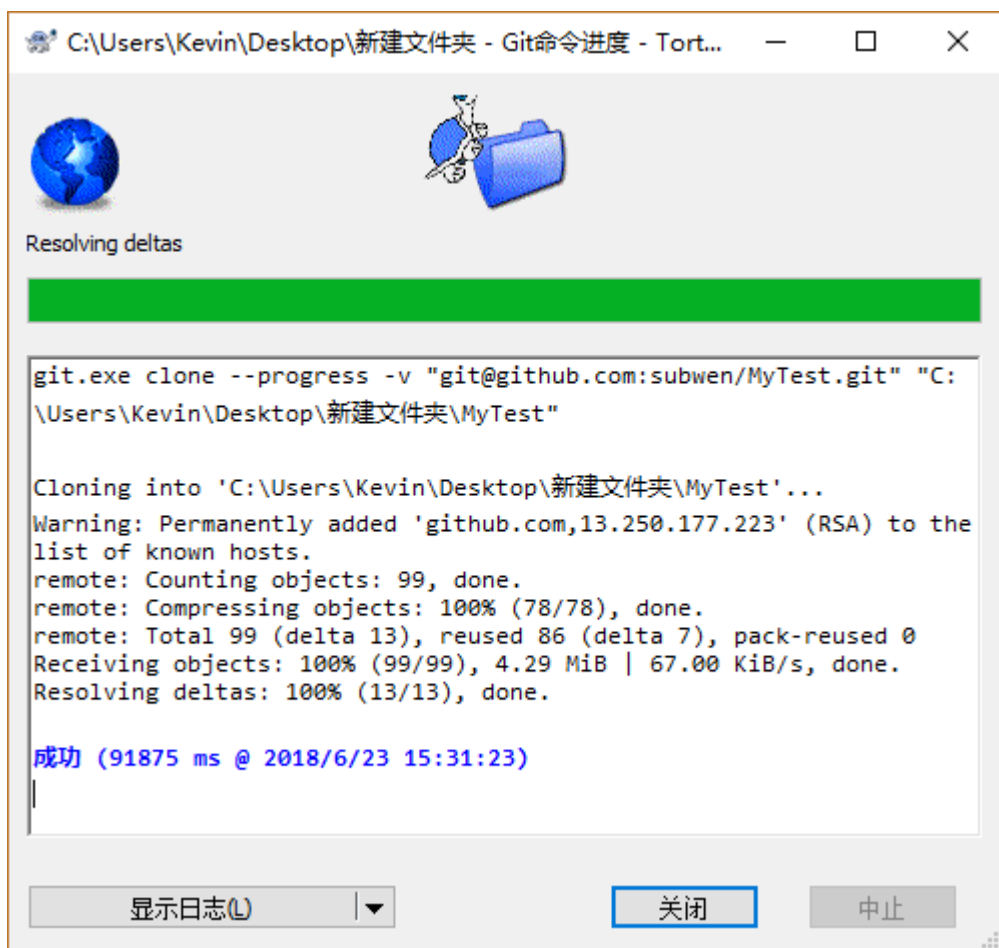
使用ssh的方式将远程仓库克隆到本地





更换ssh工具 - 打开小乌龟的设置窗口 -> 网络

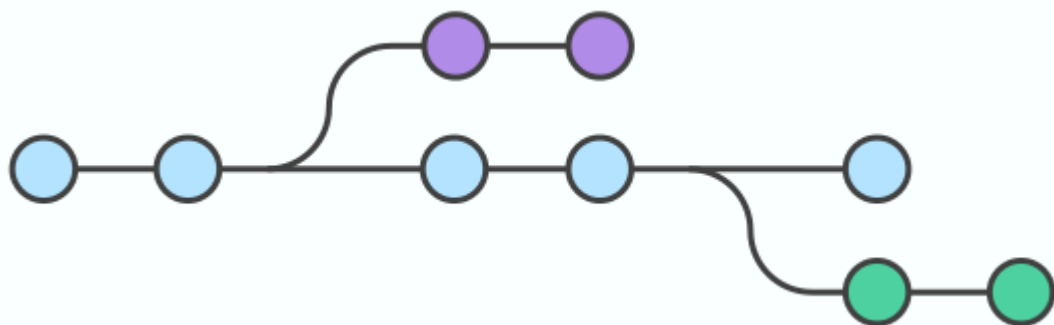




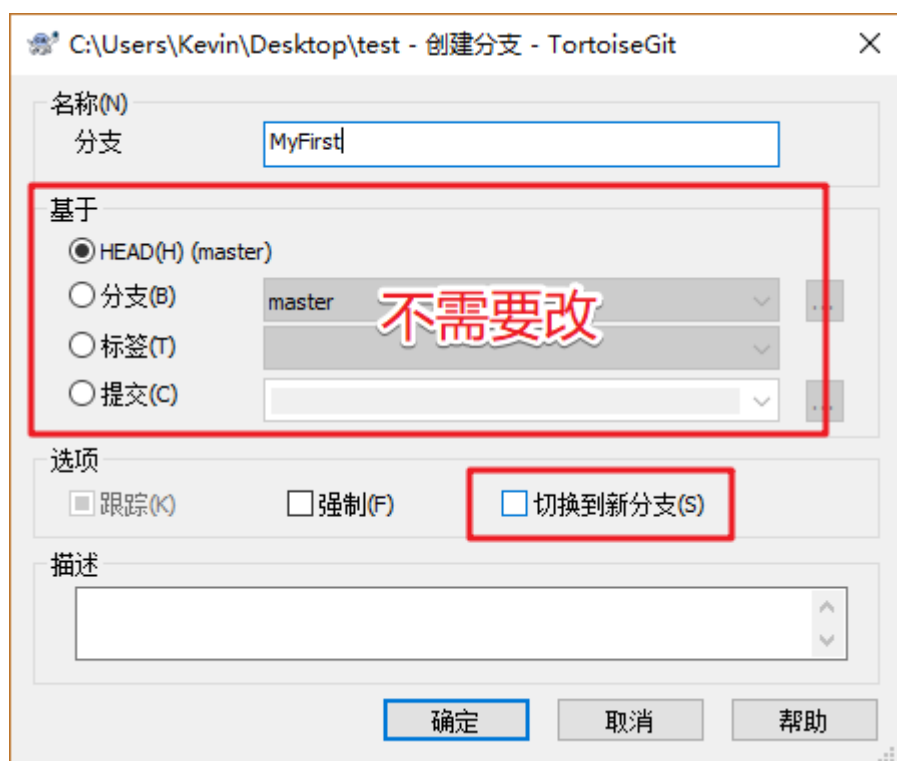
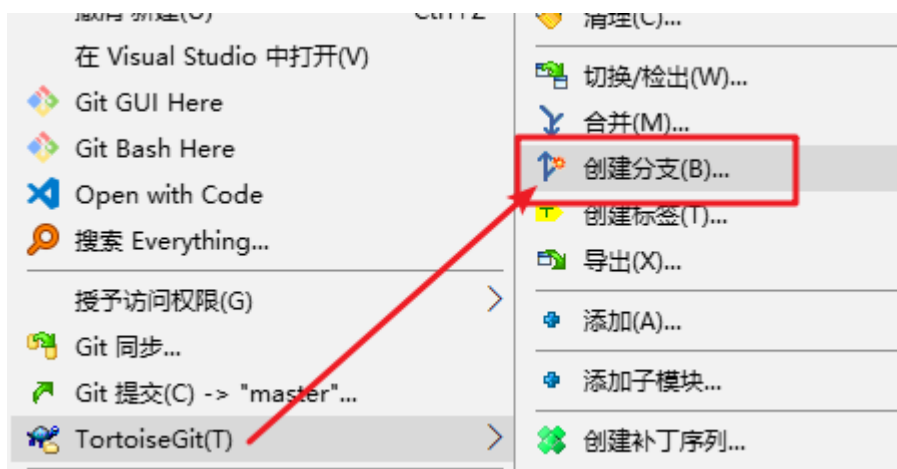
5. 管理分支

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

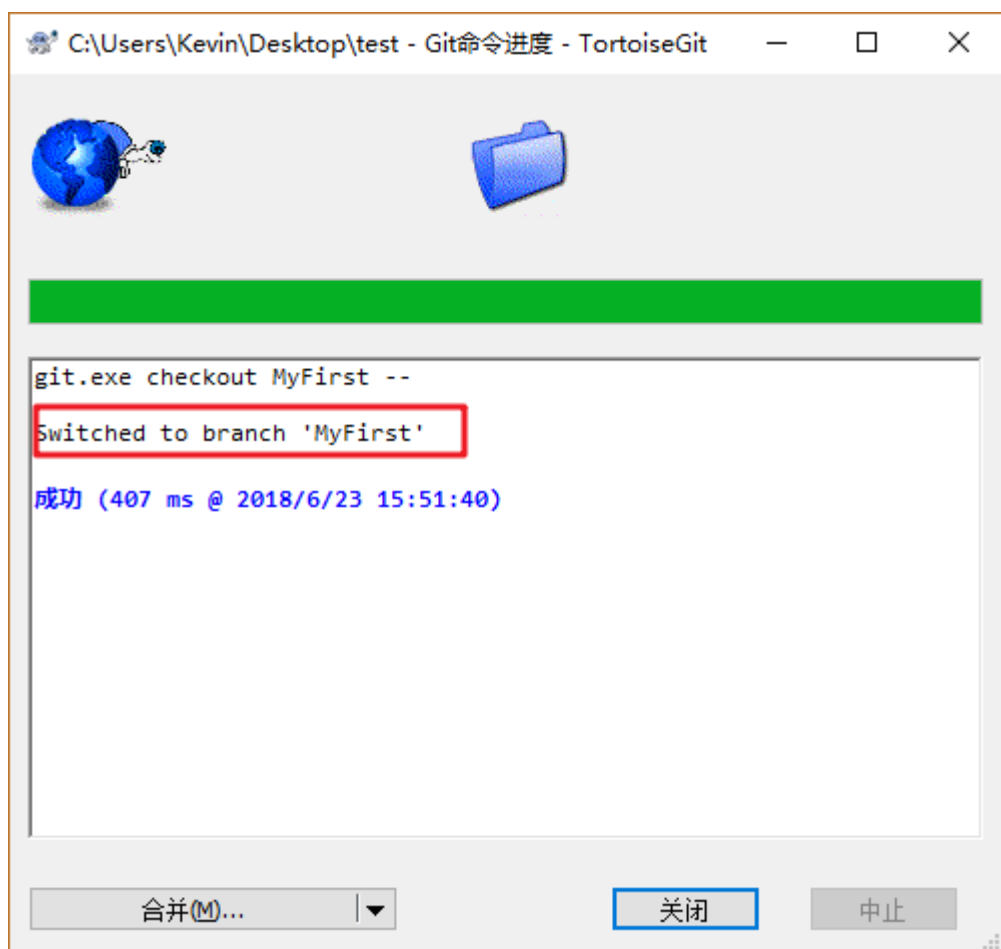
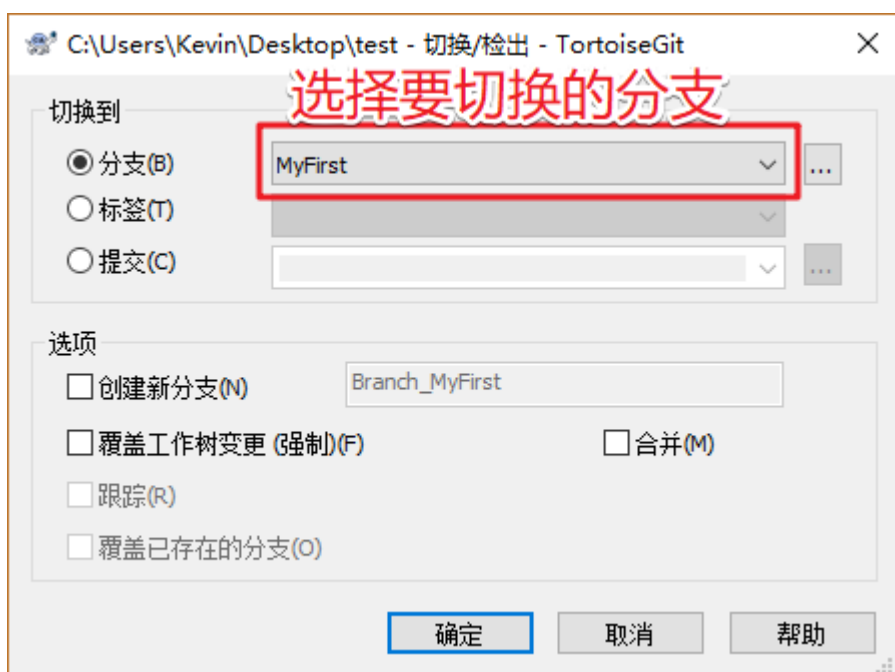


5.1 创建分支



5.2 切换分支

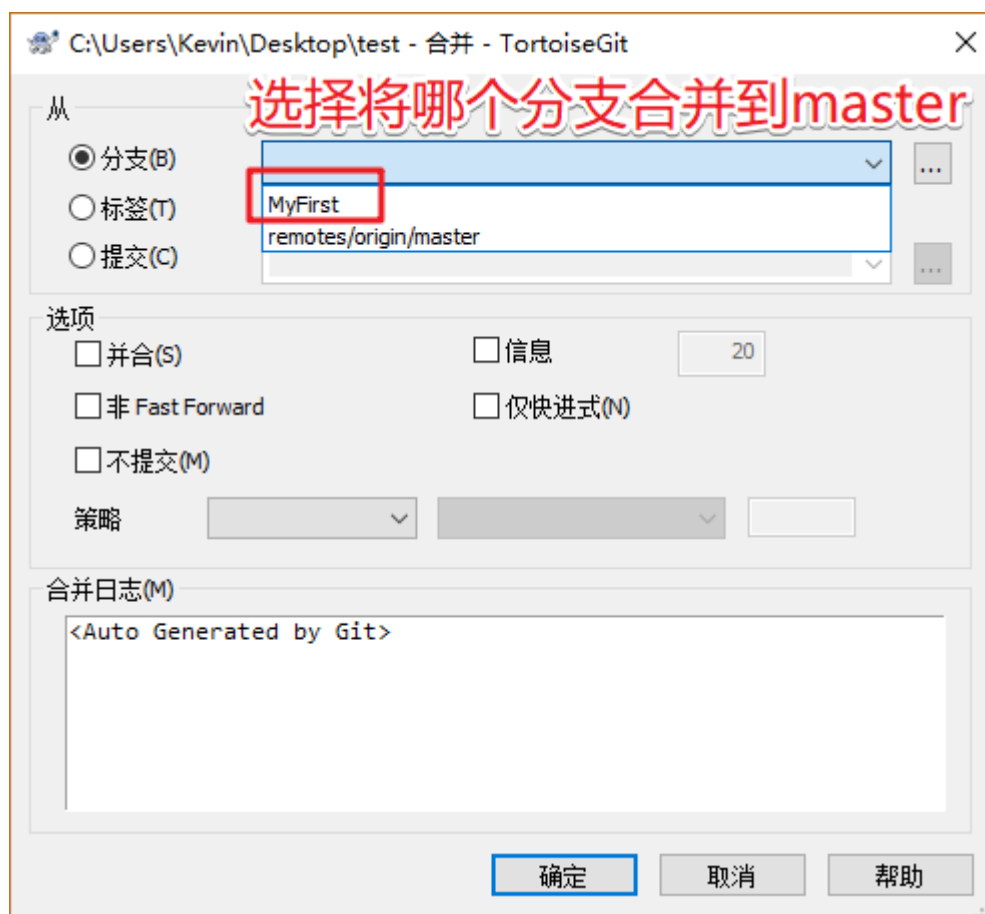
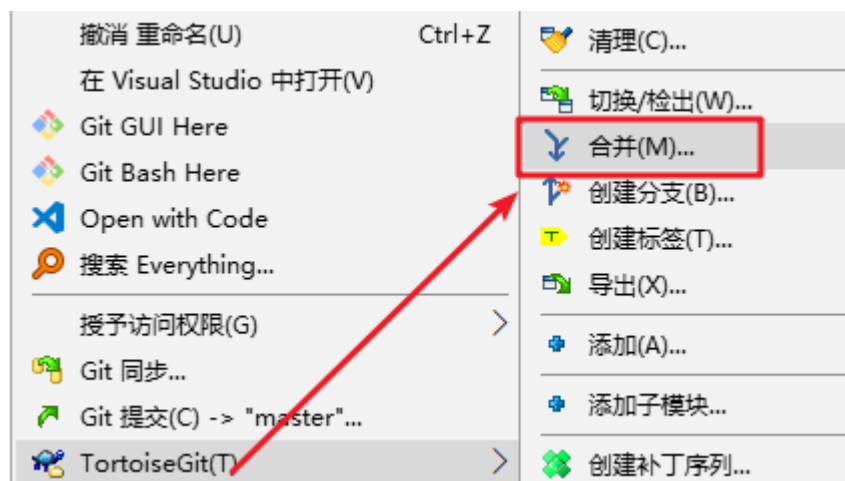




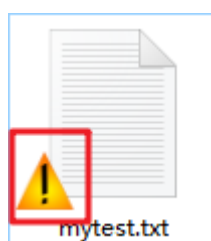
5.3 合并分支

将添加的myFirst分支合并到master分支

- 先切换到master分支
- 合并分支



5.4 解决冲突



<<<<<< HEAD 冲突的开始标记
我是master分支的文件
===== 两部分内容的分隔线
我是myfirst分支的一个文件
>>>>>> MyFirst 冲突的结束

解决冲突(O)...

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

提交至: master ☐ 新建分支

日志信息(M):

```
Merge branch 'MyFirst'
# Conflicts:
# mytest.txt
```

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"/> aaa.txt	.txt	已添加	0	0
<input checked="" type="checkbox"/> mytest.txt	.txt	已修改	3	1

☒ 显示未受版本控制的文件(U) 选择了 2 个文件, 共有 2 个文件
☐ 不自动选择子模块 [查看补丁>>](#)

☐ 显示整个工程(W) ☐ 仅仅消息(Y)

提交(O) 取消 帮助

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)

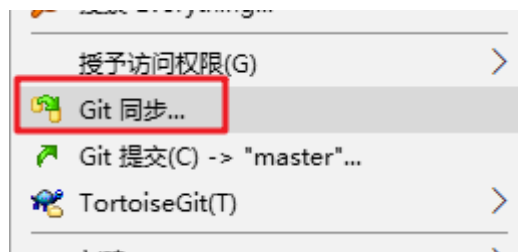
→ 放弃(A)

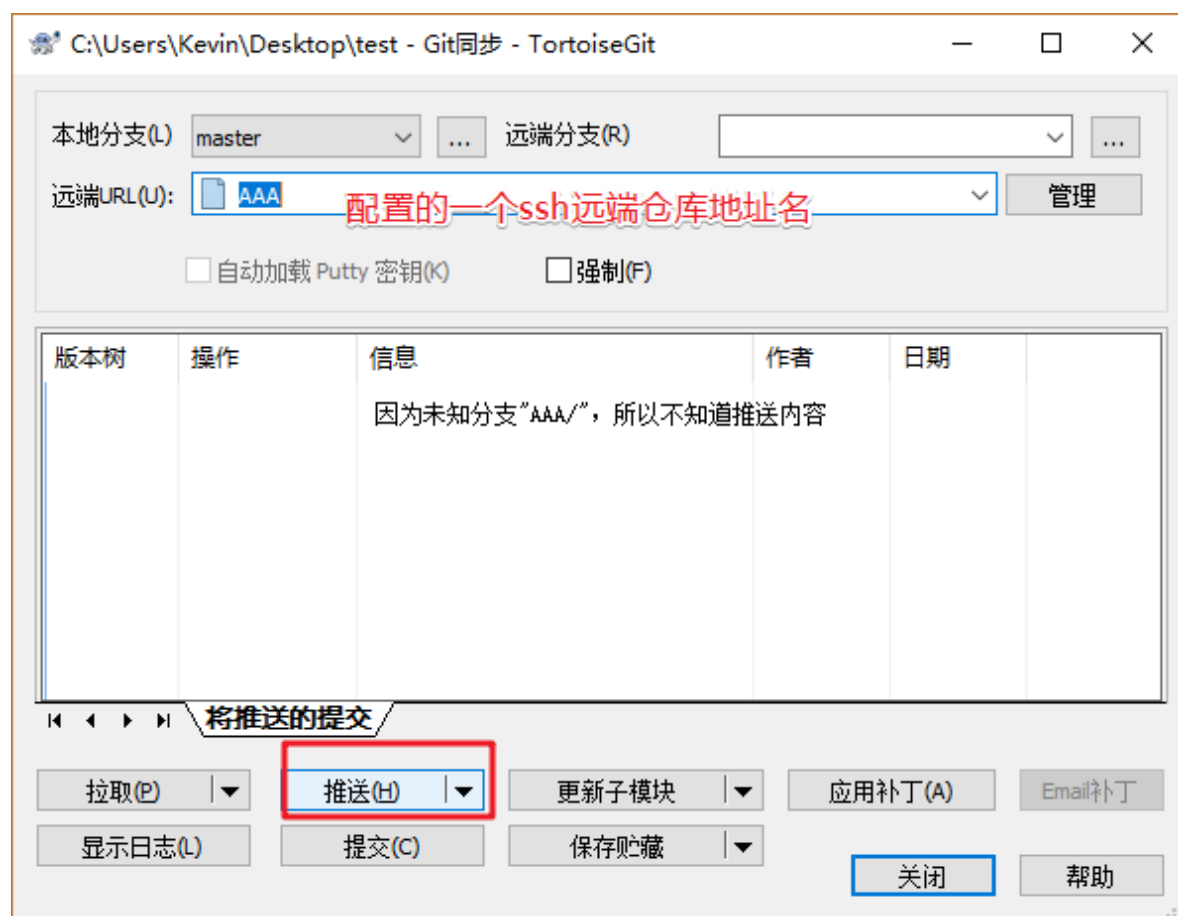
☐ 不再显示此消息(D)

6. 推送文件

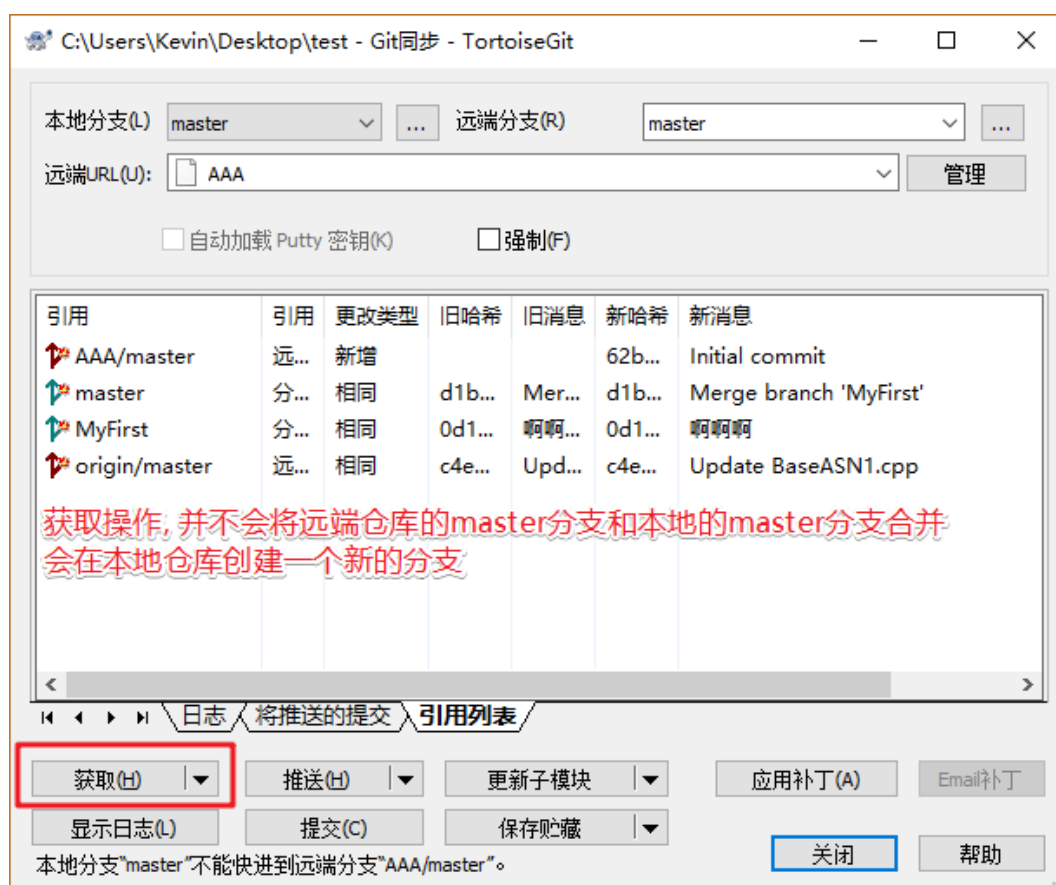
在本地有一个仓库, 管理了一个项目, 需要将本地仓库备份到远端仓库

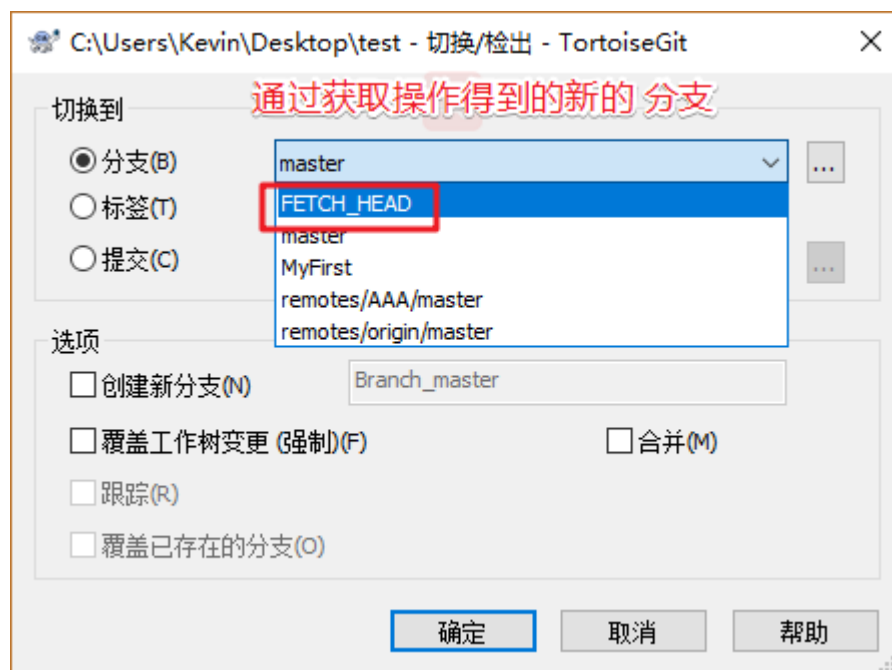
- 在github/码云创建一个仓库, 创建的仓库并不是空仓库
- 将本地仓库文件推送到远程仓库
- 在本地工作区



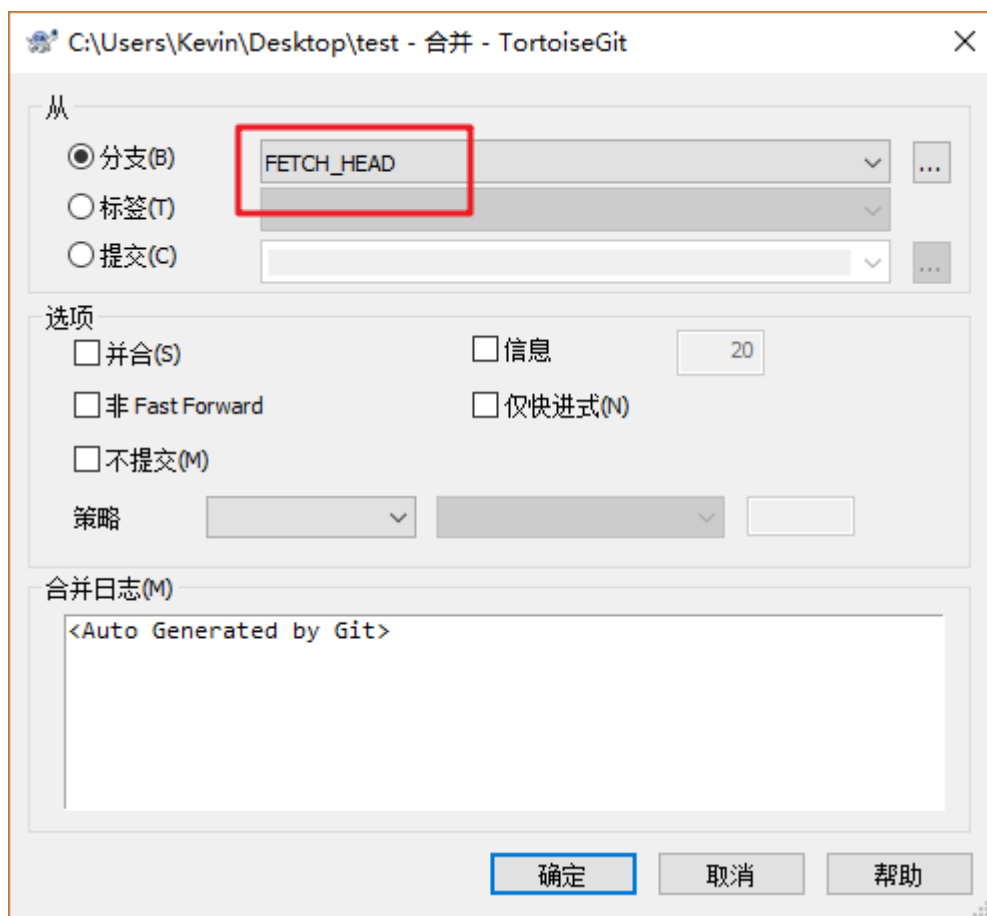


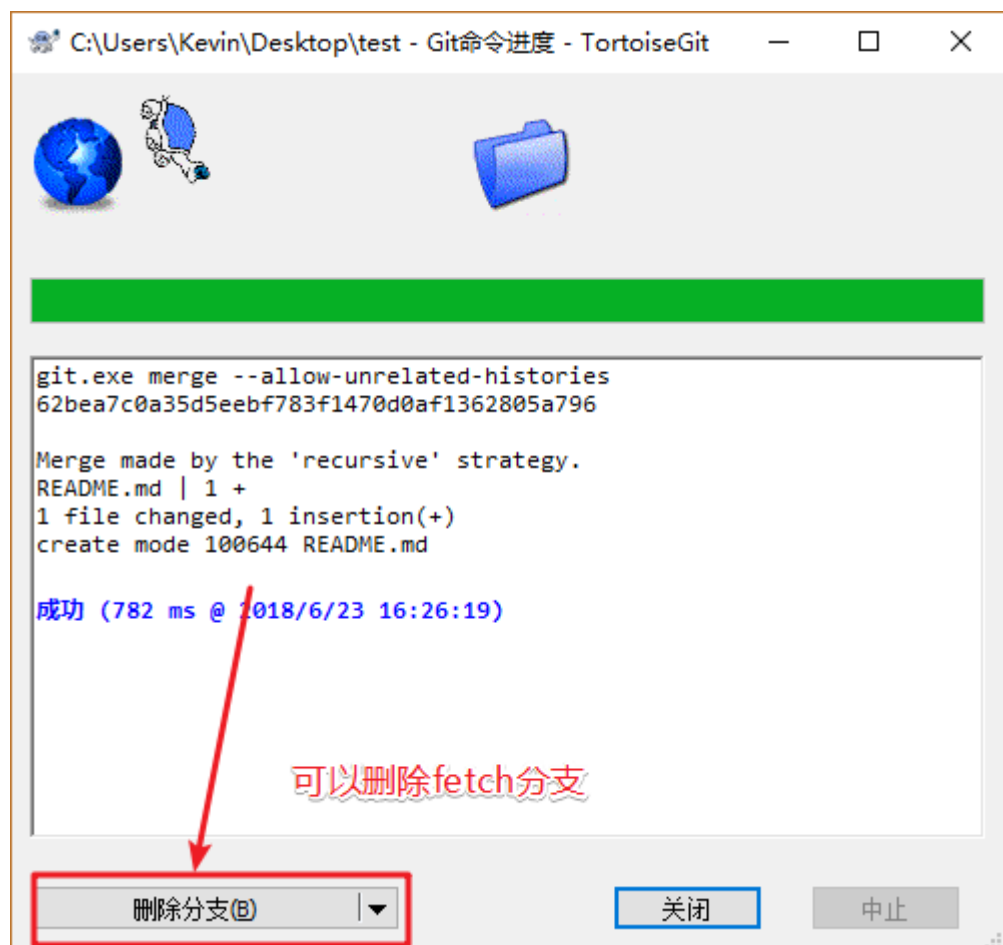
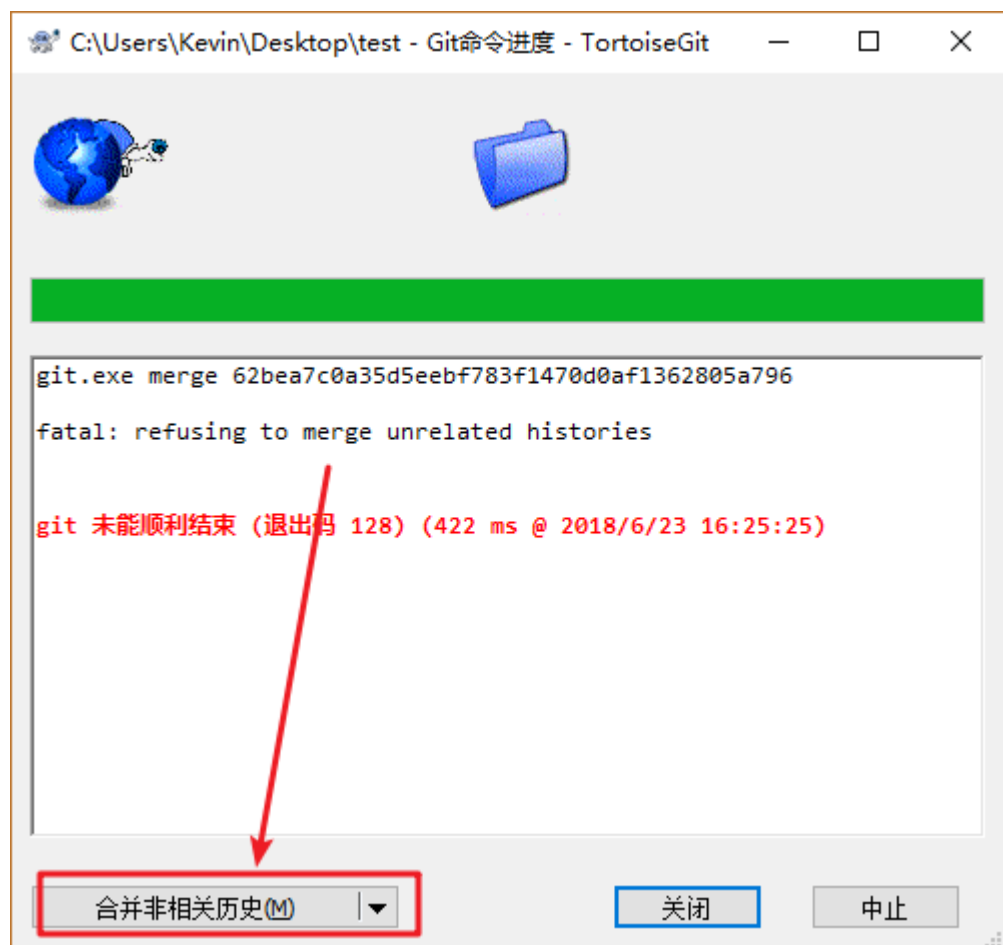
- 结果 - 推送失败
 - 首先需要先进行获取操作

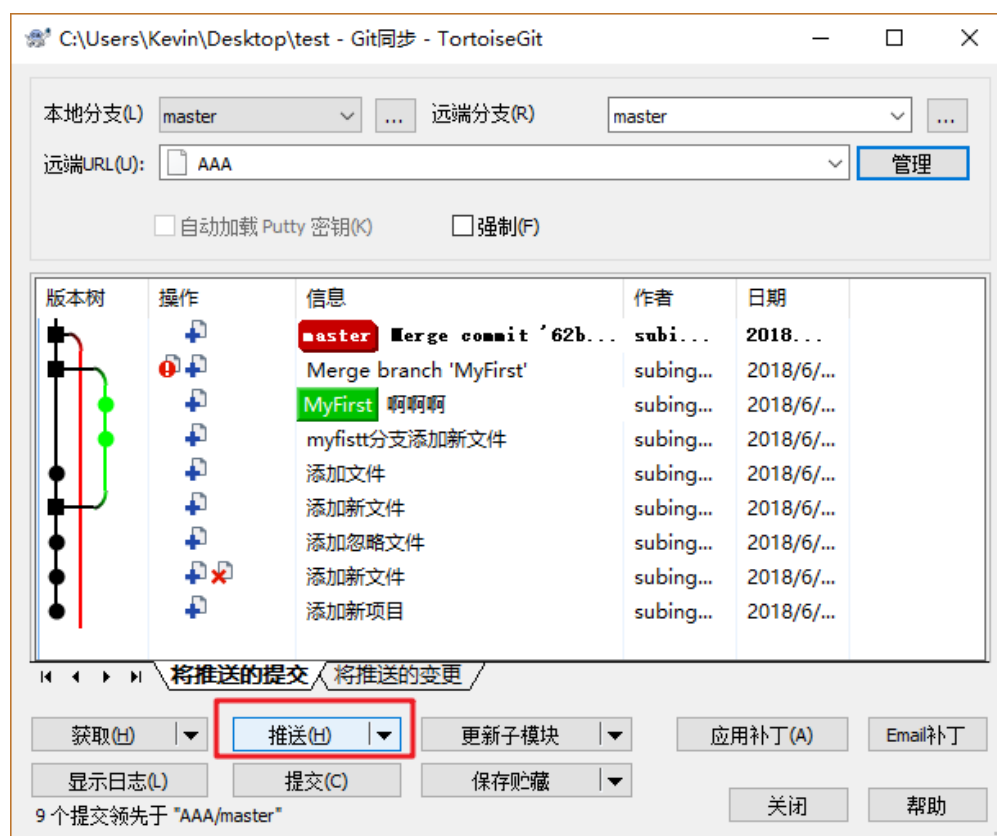


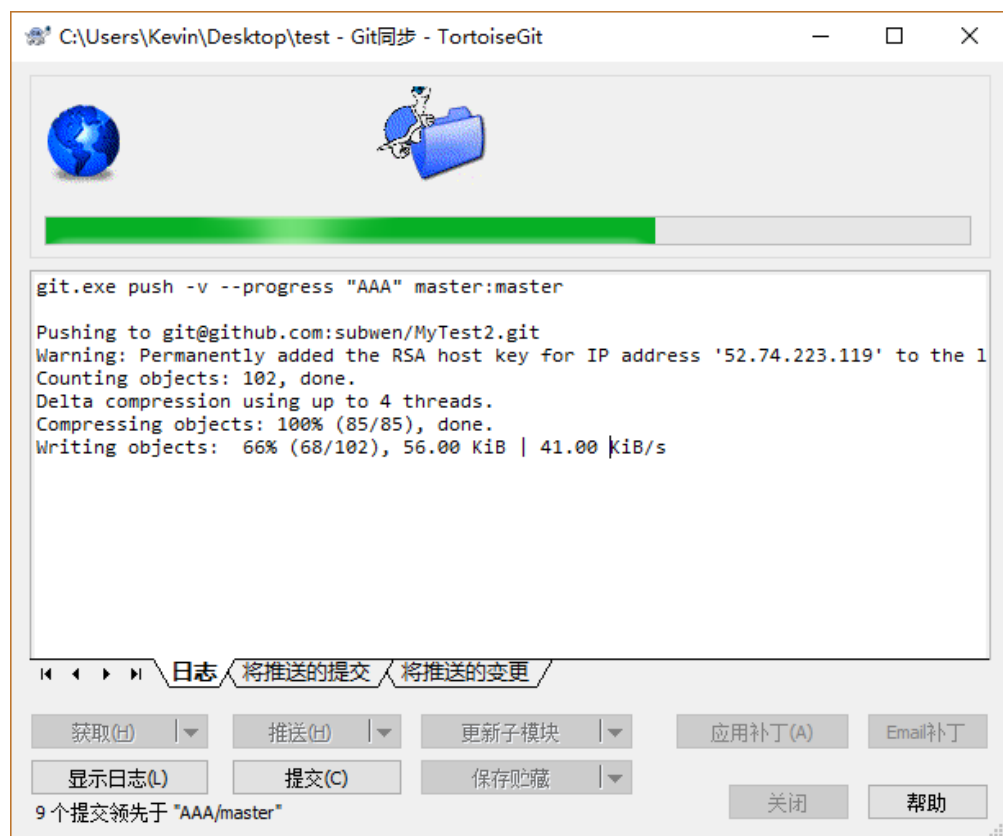


- 将得到的fetch分支合并到master分支

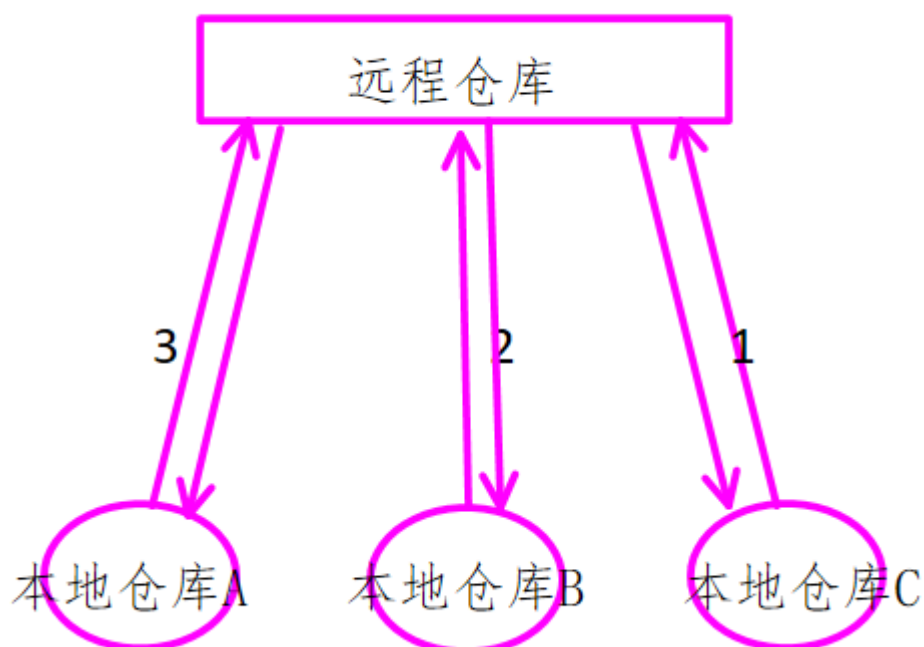








- 本地仓库和远程仓库的同步



1. 本地修改文件
2. 将本地仓库文件推送到远程仓库
 - 很大的概率执行失败
3. 所以不会直接推动, 先做拉取操作

- 将远程仓库的文件直接和本地文件合并
 - 有可能发生冲突
 - 解决冲突

4. 最后在推送