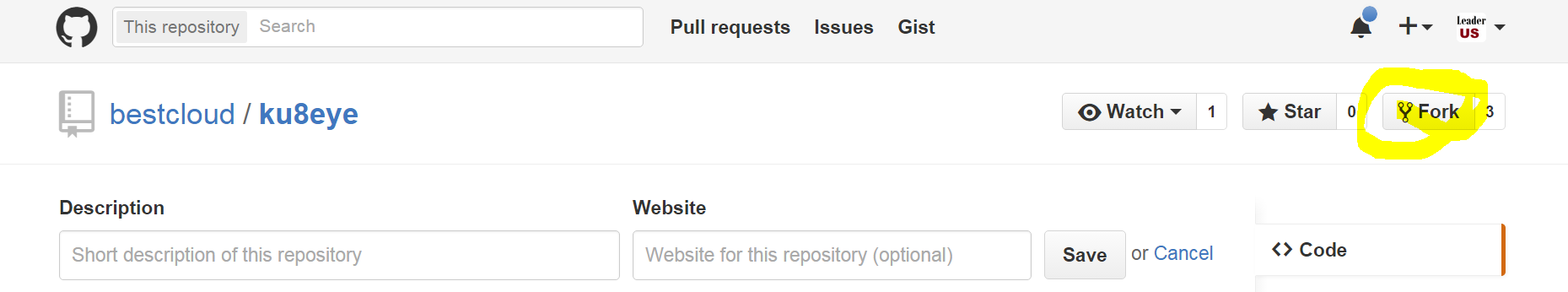
# 一：fork项目到自己的github账号

点击fork到自己的账号下



查看自己fork项目地址，用户名假如my，则自己的项目地址：

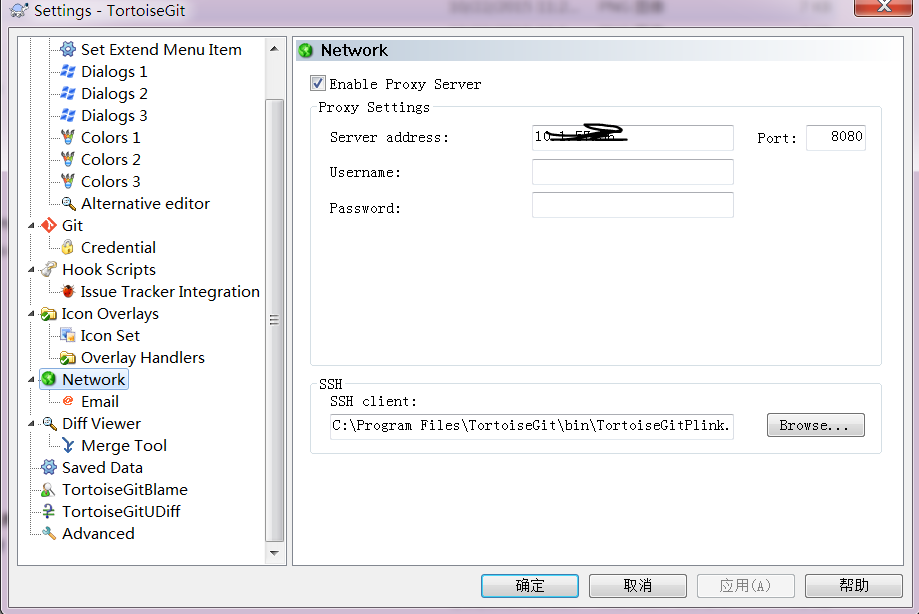
<https://github.com/my/ku8eye>

# 二：安装客户端工具

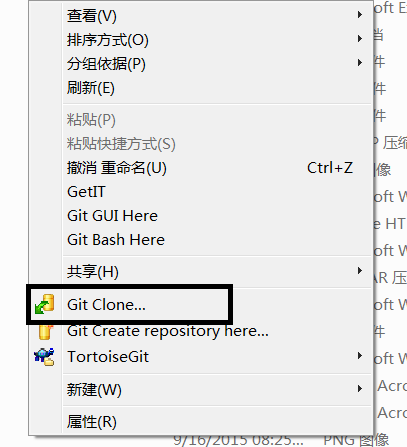
* Git-2.6.2-64-bit.exe
* TotoiseGit

# 三：检出项目

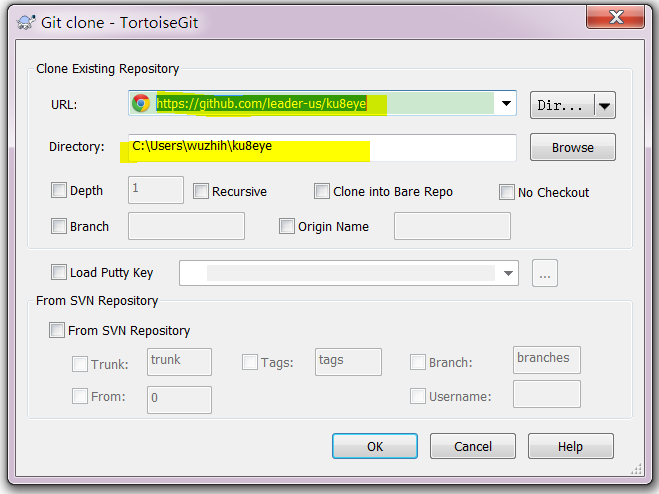
如果需要代理上网，则右键菜单TotoiseGit——》Settings——》Network设置代理



文件右键菜单：Git Clone

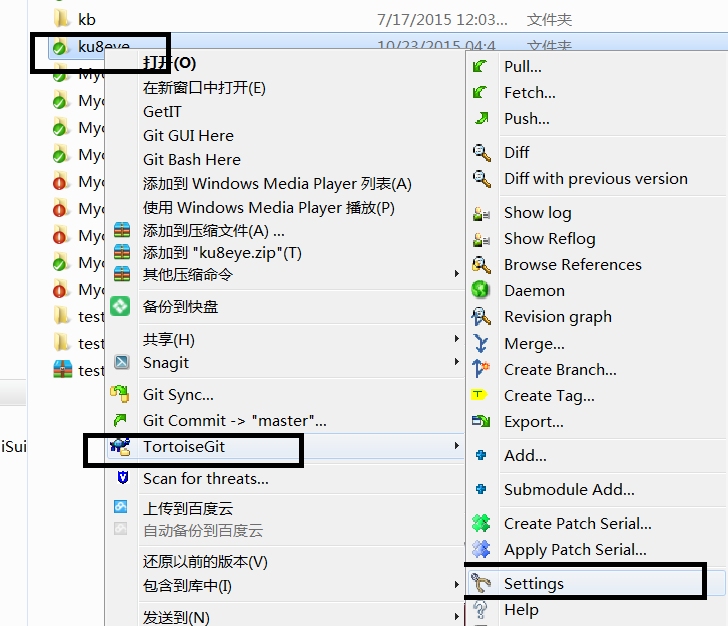


填入自己Fork的地址，注意不是官方的项目地址

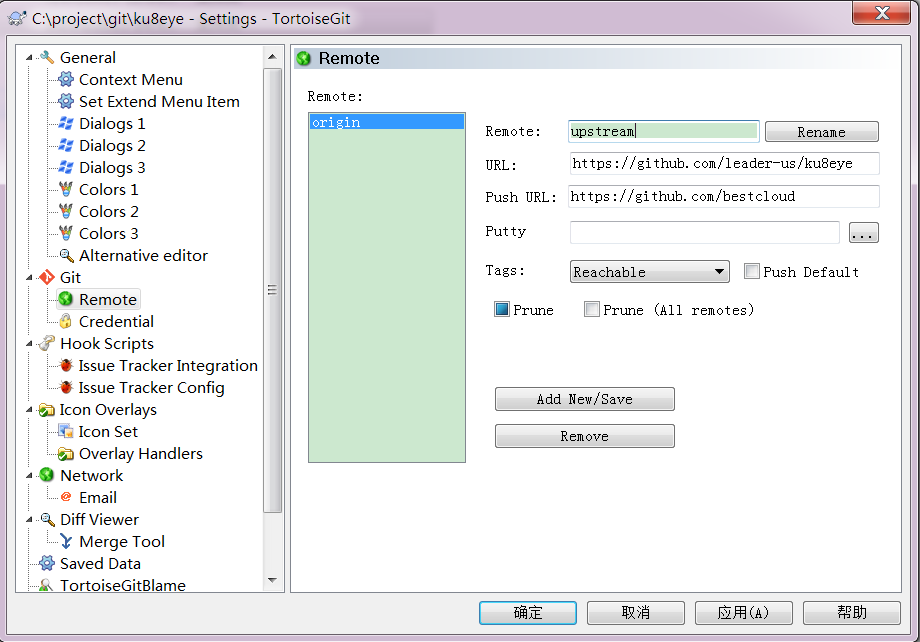


增加官方源地址:upstream

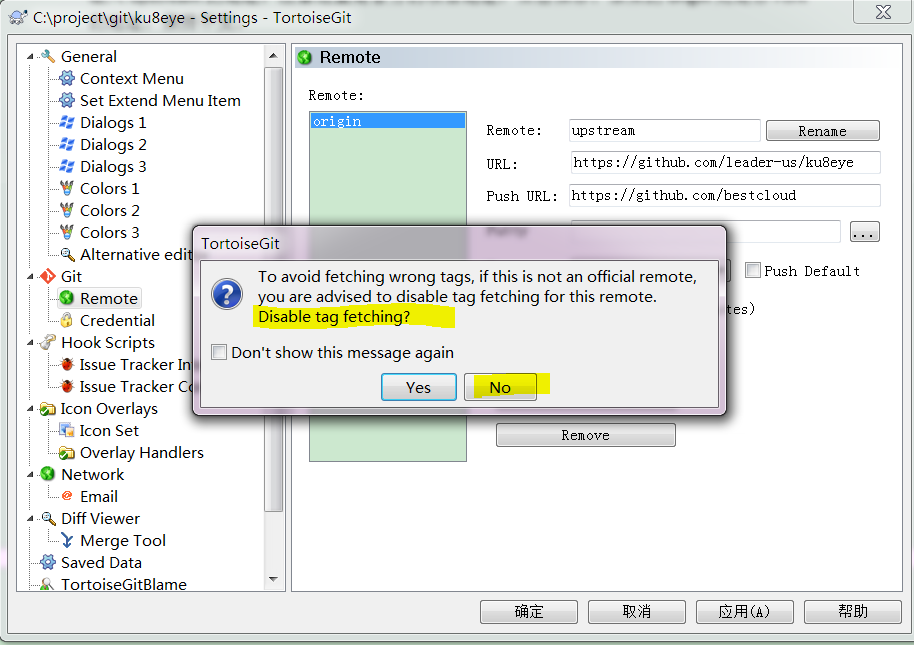
检出的项目的邮件菜单TotoiseGit——》Settings



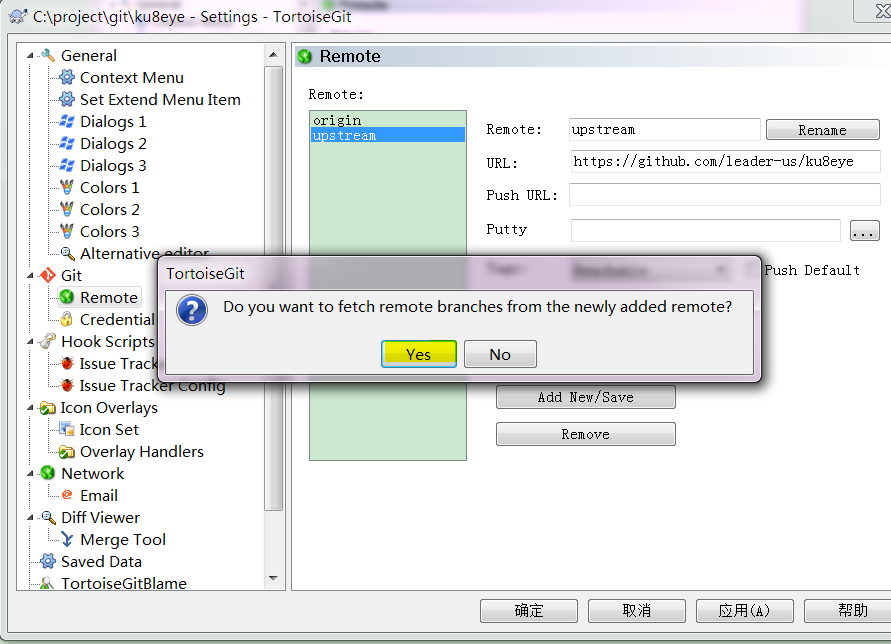
输入upstream的地址，注意这里是官方的项目地址，点击保存。原来的origin则是你Fork的地址，保持不变。

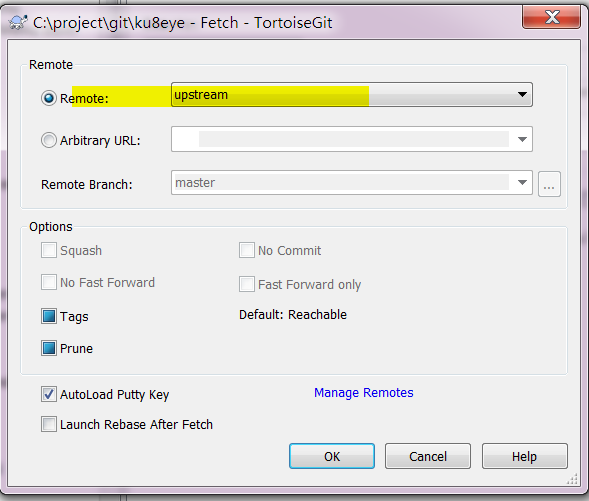


保存的时候，第一次会提升，防止出错，关闭默认从该地址pull/fetch的功能，我们要点击“No”，即默认从官方源pull/fetch，以同步到我们的本地库中。

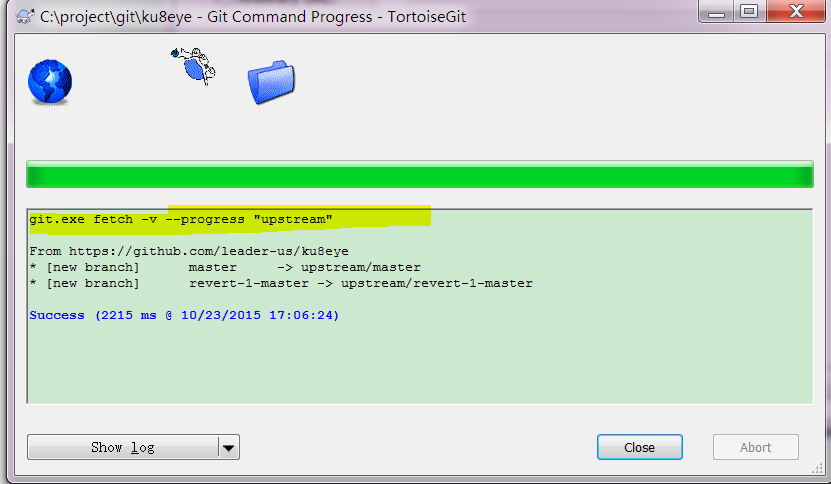


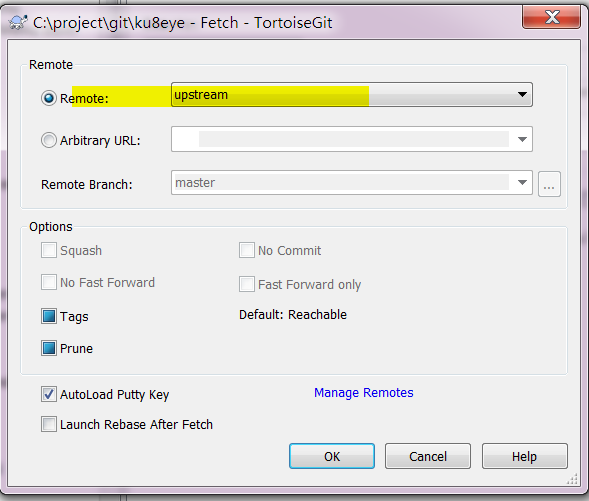
接下来会从upstream拉取一次源码：





效果如下：





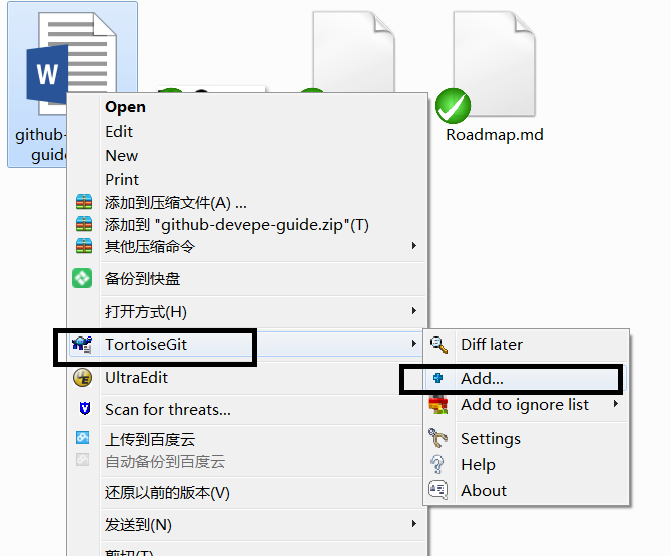
# 四：提交代码

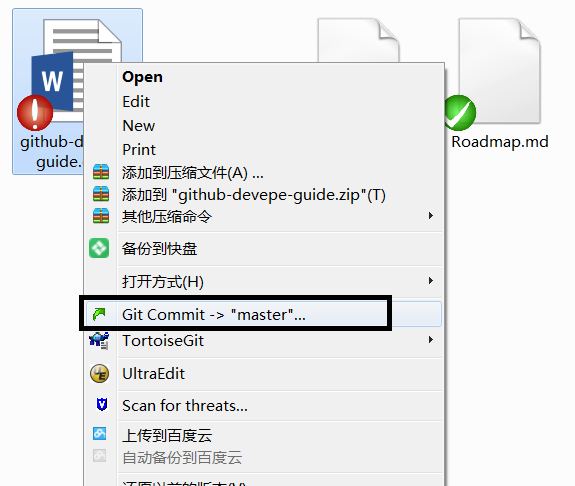
如果我们本地有代码改变，比如新增文件或代码，流程如下：

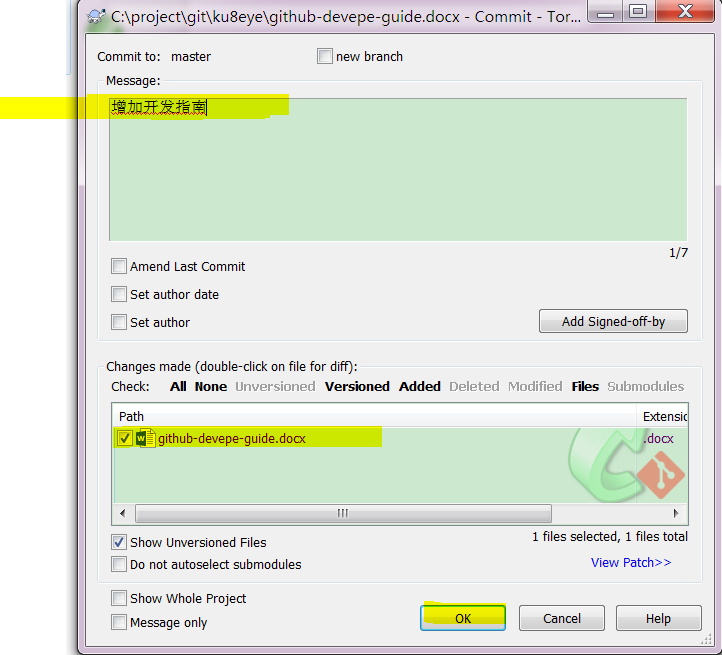
Commit（本地提交）->push（提交到自己Fork的地址）->发起pull request(可以工具里或者gihub自己的项目主页里)

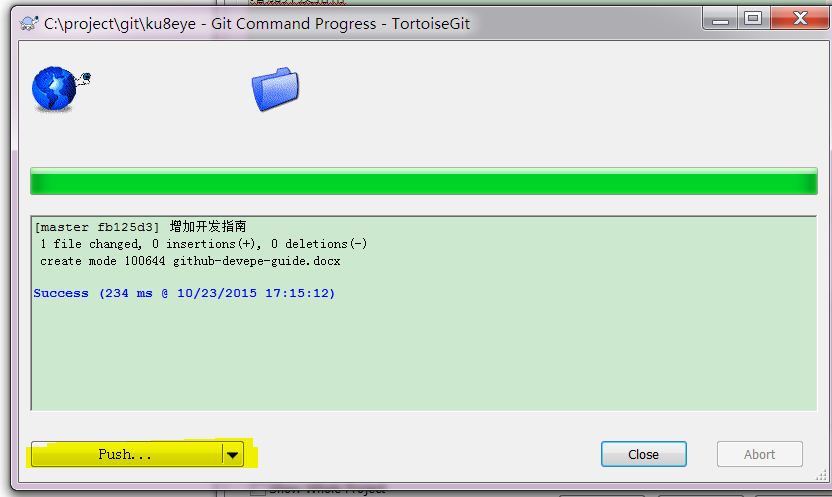
比如我们新增了一个文件，右键TotoiseGit –》Add，添加到版本控制文件列表中，

然后Commit到本地库里。



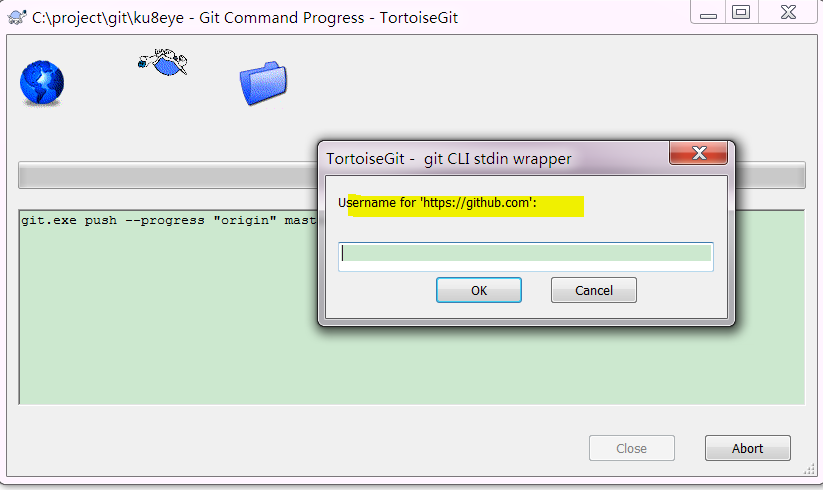


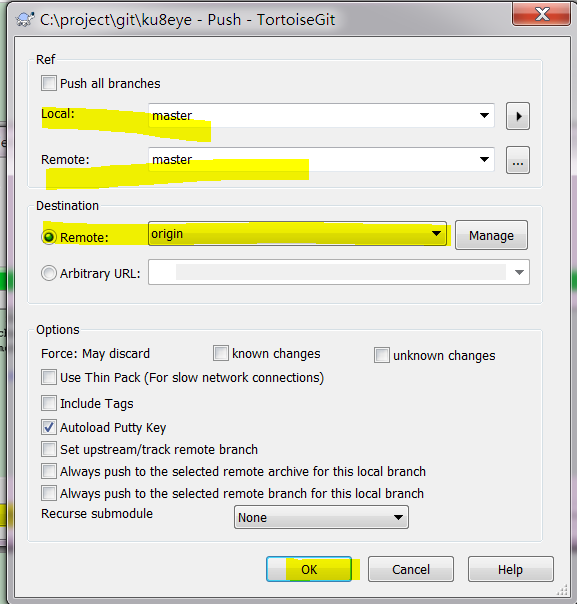




Commit完成以后可以继续Push，或者以后Push，只有Push到自己Fork地址以后，才是提交到github里了，否则只是本地保存了。

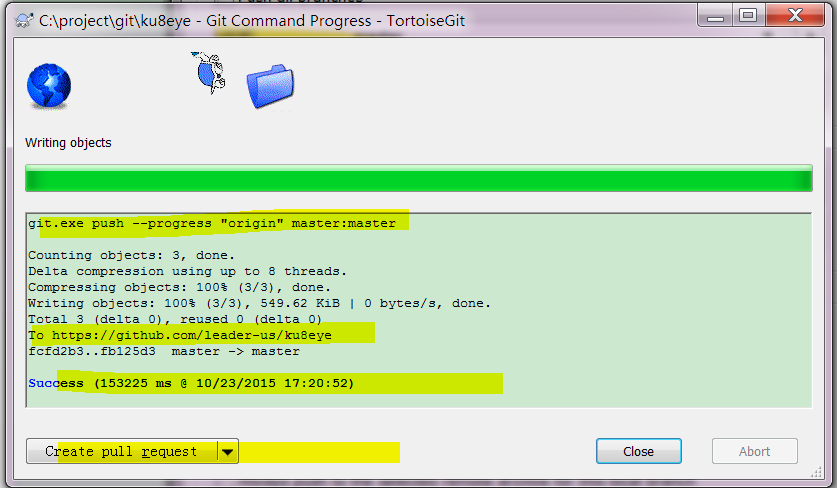
输入自己的用户名密码提交



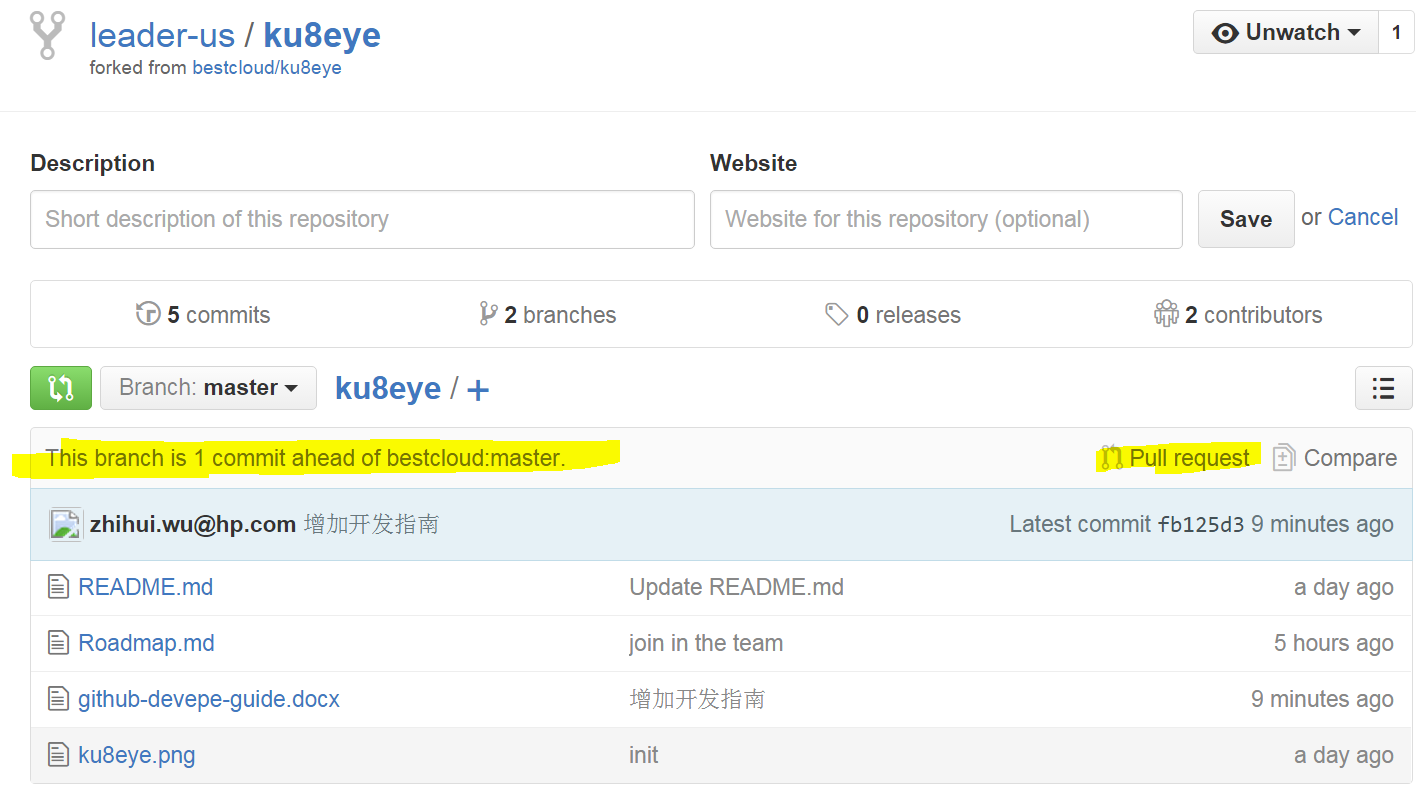


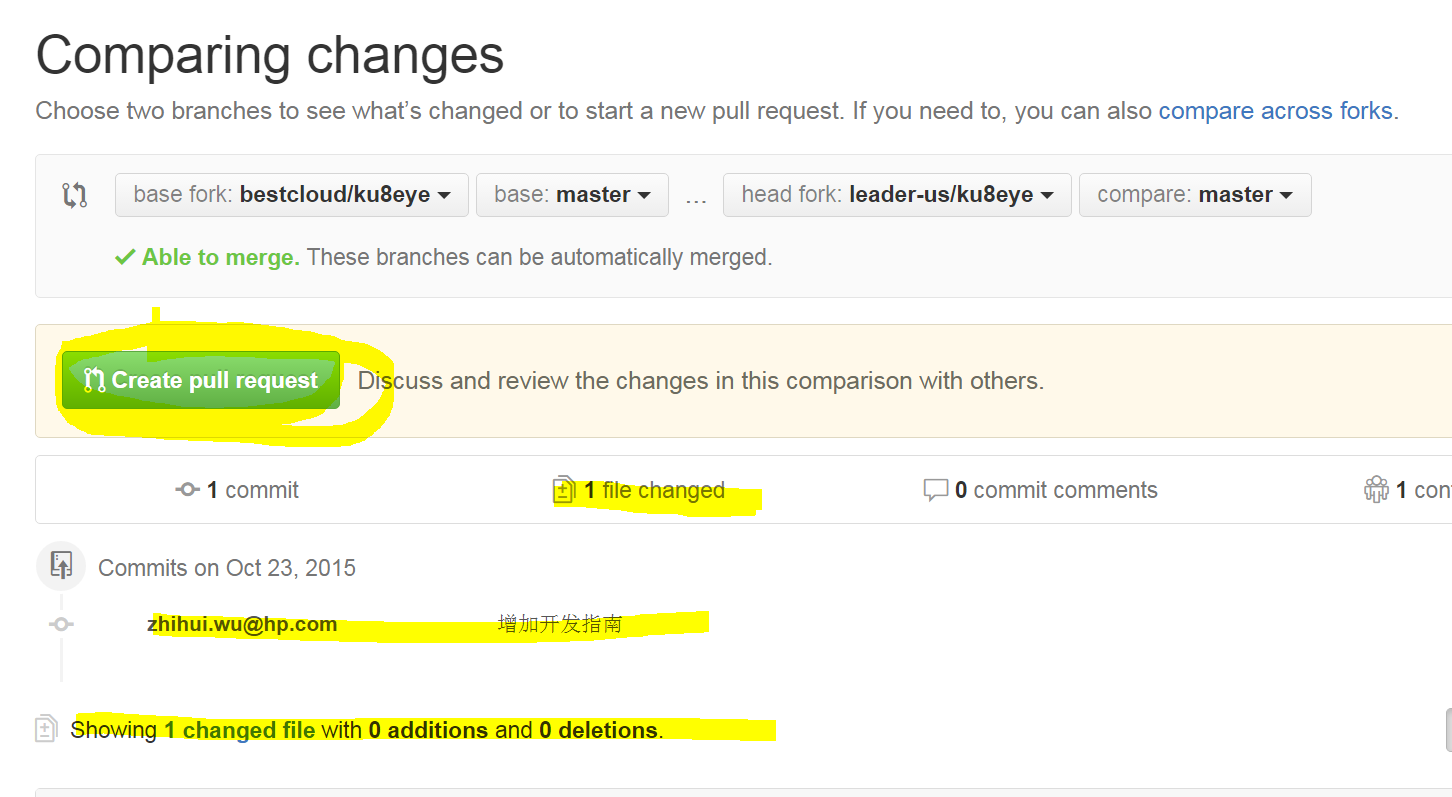
注意到，目标地址Remoet为Origin地址，即我们自己Fork的地址。

Push成功以后，显示如下内容，可以继续发起pull reqest请求，将变更提交到官方源地址(upstream)，请求合并你的变更：



一般是在github网站上发起pull request的，用自己的账号登录github，看到项目变动情况，发现版本是领衔官方源一个Commit，可以点击Pull request请求官方合并你的Commit

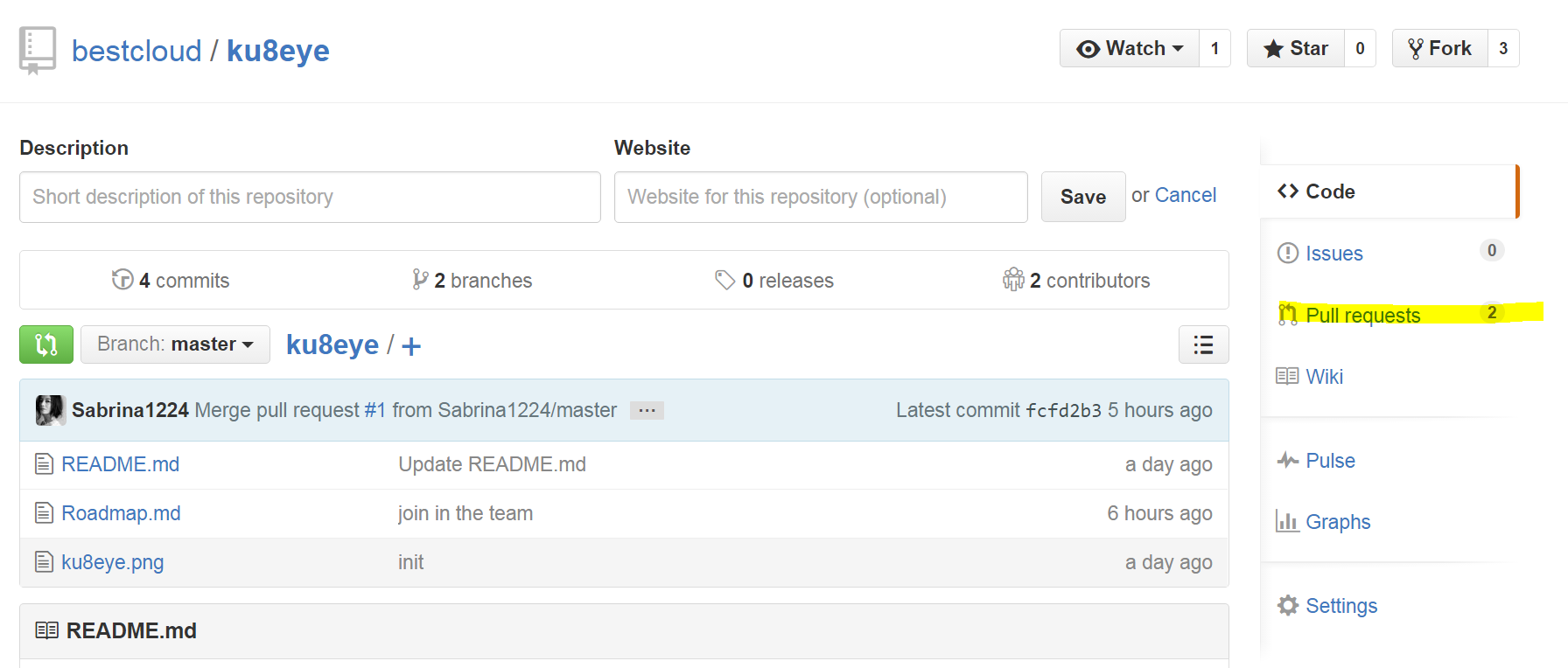




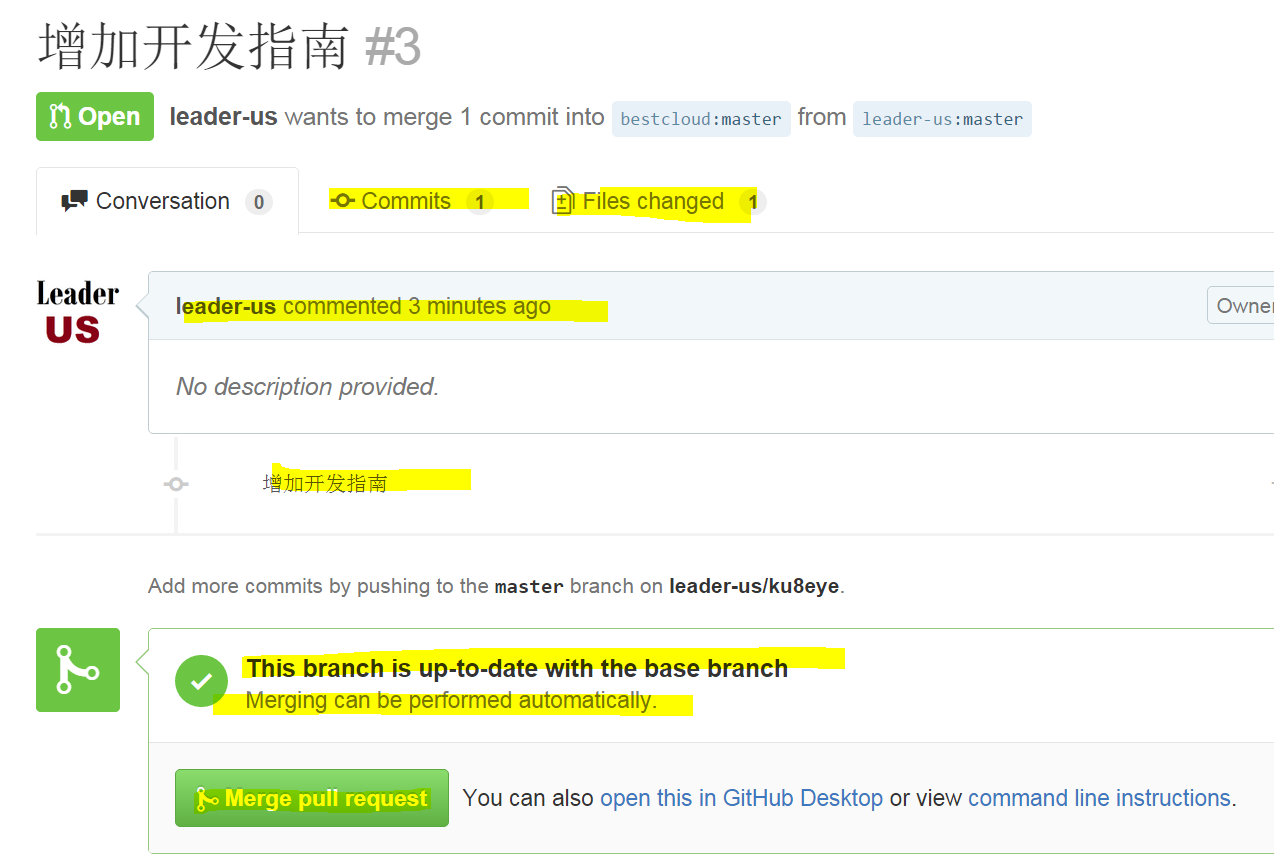
发起Pull Request的过程中，可以看到你提交的文件数量，具体文件，以及提交时候的注释内容，有助于 官方来确定你的提交是否能被接纳。

# 五：官方Merge请求

所有的Pull request会在官方项目信息里看到



如果没有冲突，则官方会选择合并，



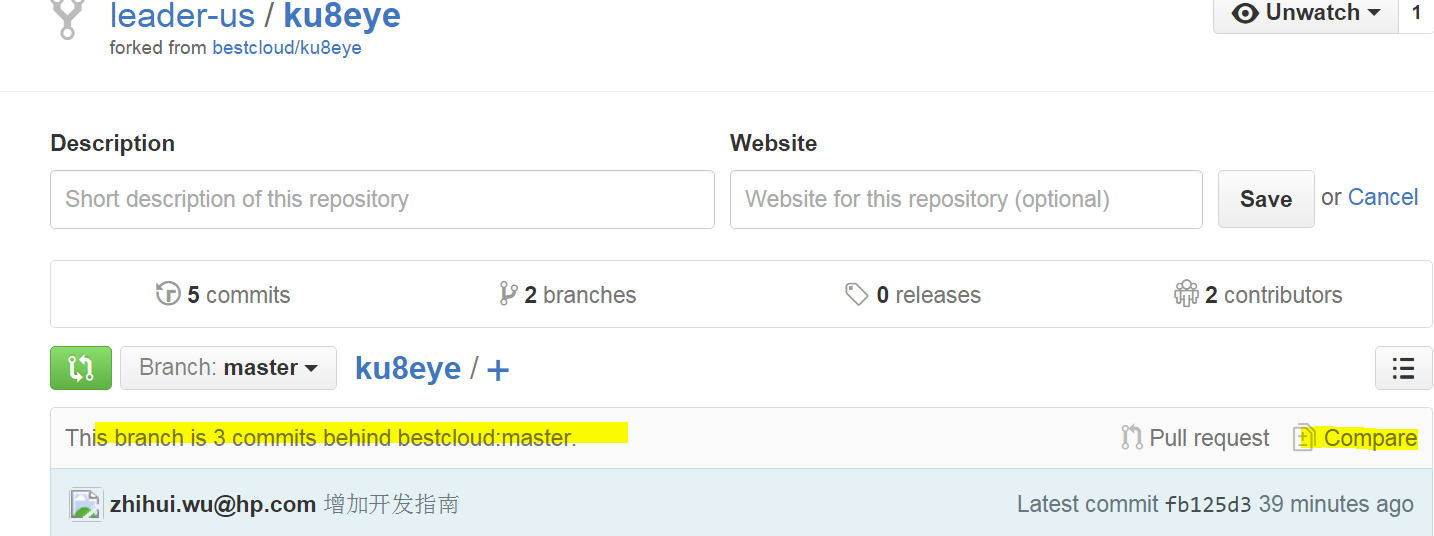
也可能会拒绝，拒绝的话，最好提供详细的拒绝理由，供请求者参考和改进，重新发起Pull。

如果无法自动合并，则表明冲突了，需要发起者同步一次官网最新代码并且Merge以后再提交。

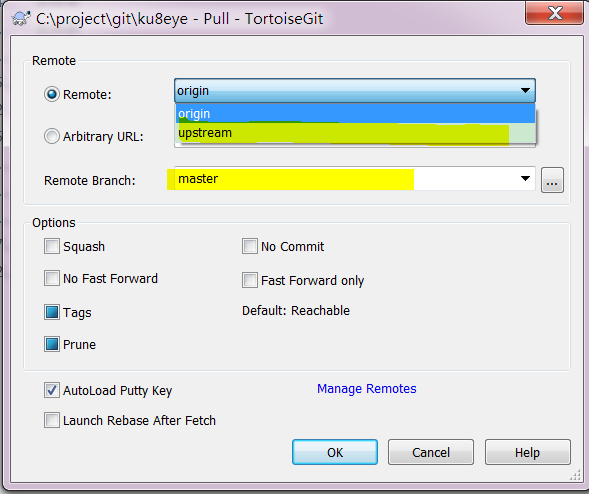
# 六：从官网源同步代码

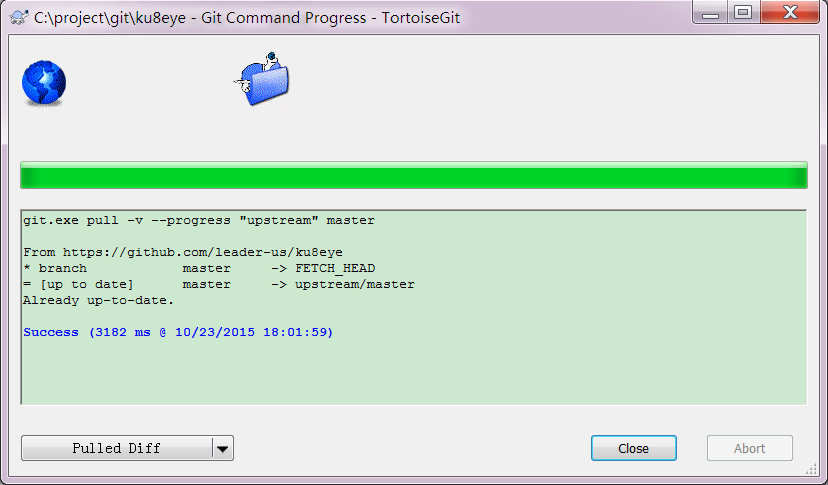
可以用pull 或者fetch两种方式从官网源（upstream）同步代码，fetch是执行了pull +merge的操作，即pull下来与本地的仓库进行merge操作，建议百度更详细的说明。

在自己fork的项目下，可以看到当前版本与官网源的差距，如下图所示：

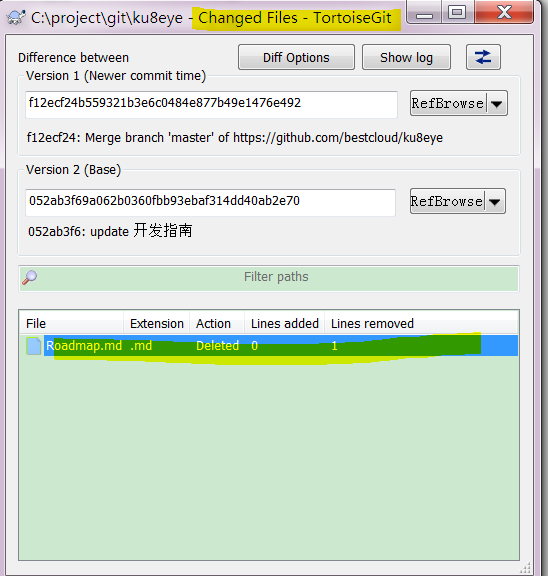


此次，是可以从官方源fetch/pull最新代码的，如下图所示，Remote选择upstream（官方源）：





可以点击Pulled Diff来查看具体的变化信息：



# diff时忽略空格

有些修改只是增减了空格，在URL中添加?w=1就可以忽略。

# git did not exit cleanly (exit code 128)

“git did not exit cleanly (exit code 128)”错误

1、鼠标右键 -> TortoiseGit -> Settings -> Network

2、SSH client 原值： C:\Program Files\TortoiseGit\bin\TortoisePlink.exe

修改为 C:\Program Files (x86)\Git\bin\ssh.exe

# **Github上fork之后如何保持同步（Windows环境）**

git remote -v

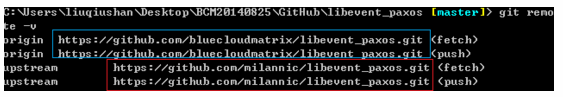
git fetch **upstream**

**git checkout master**

**git merge upstream/master**

我们会去fork别人的一个项目，这就在自己的Github上生成了一个与原作者项目互不影响的副本，自己可以将自己Github上的这个项目再clone到本地进行修改，修改后再push，只有自己Github上的项目会发生改变，而原作者项目并不会受影响，避免了原作者项目被污染。但是如果原作者在不断更新他的项目，如何也让自己Github上的也跟着同步？这里需要借助在windows下安装github出现的一个工具Git Shell，以前一直不知道它的用处，这次体会一下。

1）进入本地项目目录，输入 git remote -v:



篮框内的url是我Github上的项目，红框内的url是原作者项目。如果没有upstream，即没有原作者项目的url，你需要自己添加：$ git remote add upstream <原作者项目的URL>

2）将原作者项目更新的内容同步到我的本地项目（不是我Github网上的项目）：

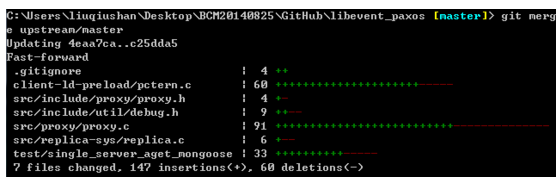
a) Fetch the branches and their respective commits from the upstream repository. Commits to **master**will be stored in a local branch, **upstream/master**.



b) Check out your fork's local **master**branch.



c) 接下来就是合并这两个分支，将原作者项目的修改同步到自己这里（注意还是指本地项目，不是自己Github空间里的项目）。Merge the changes from***upstream/master***into your local ***master***branch. This brings your fork's ***master***branch into sync with the upstream repository, without losing your local changes.



至此我的本地项目已经于原作者项目同步了。

3）也让自己Github空间里的项目得到同步：

通过另一个工具Github.exe，将本地项目的修改（即与原作者项目同步的内容）push到自己的Github上