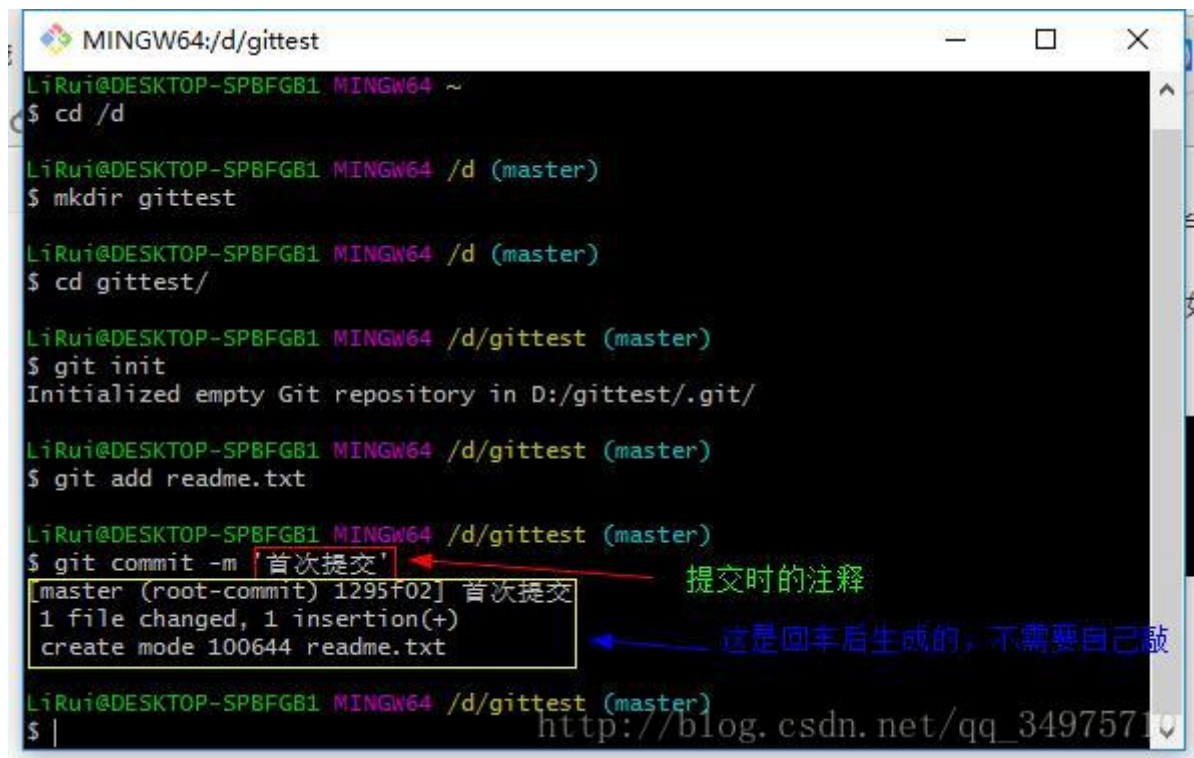


Sourcetree的分支创建与合并

一、Sourcetree简单介绍

通过Git可以进行对项目的版本管理，但是如果直接使用Git的软件会比较麻烦，因为是通过一条一条命令进行操作的。



```
MINGW64:/d/gittest
LiRui@DESKTOP-SPBFG81 MINGW64 ~
$ cd /d

LiRui@DESKTOP-SPBFG81 MINGW64 /d (master)
$ mkdir gittest

LiRui@DESKTOP-SPBFG81 MINGW64 /d (master)
$ cd gittest/

LiRui@DESKTOP-SPBFG81 MINGW64 /d/gittest (master)
$ git init
Initialized empty Git repository in D:/gittest/.git/

LiRui@DESKTOP-SPBFG81 MINGW64 /d/gittest (master)
$ git add readme.txt

LiRui@DESKTOP-SPBFG81 MINGW64 /d/gittest (master)
$ git commit -m 首次提交
[master (root-commit) 1295f02] 首次提交
1 file changed, 1 insertion(+)
create mode 100644 readme.txt

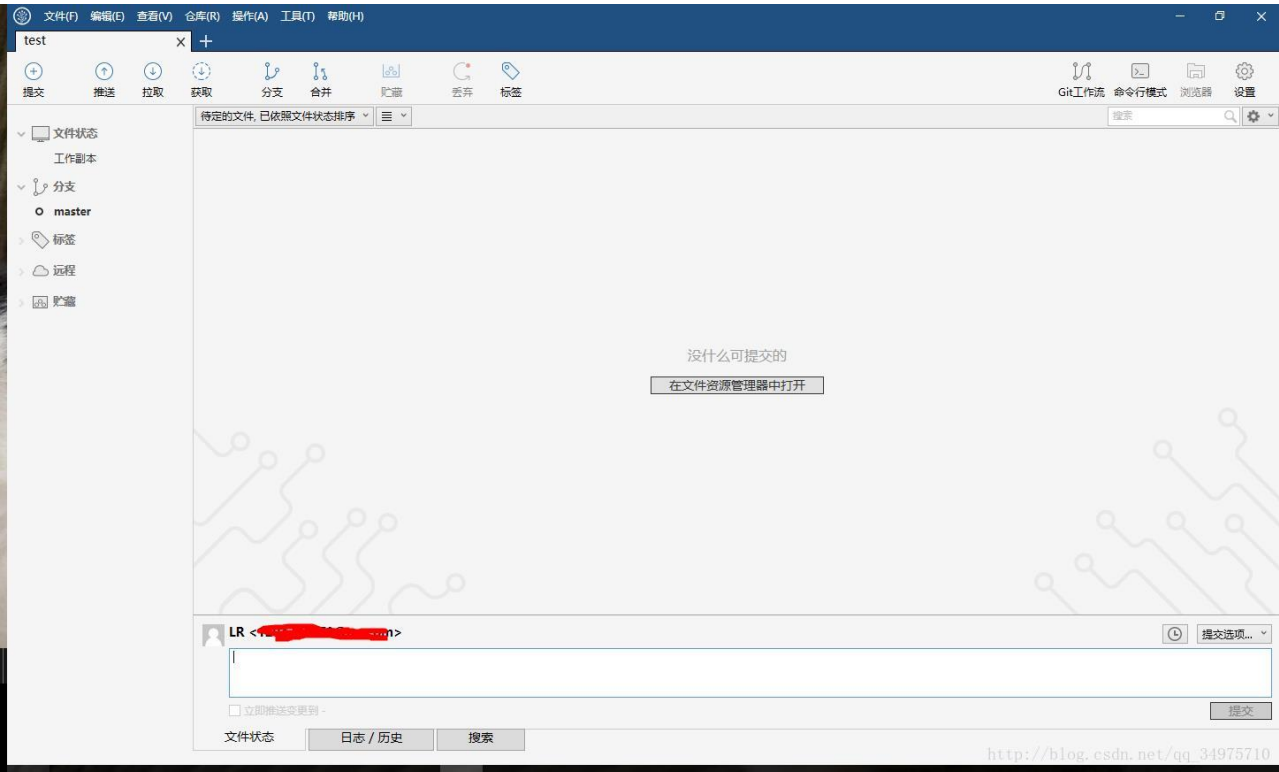
LiRui@DESKTOP-SPBFG81 MINGW64 /d/gittest (master)
$ |
```

提交时的注释

这是回车后生成的，不需要自己敲

http://blog.csdn.net/qq_34975710

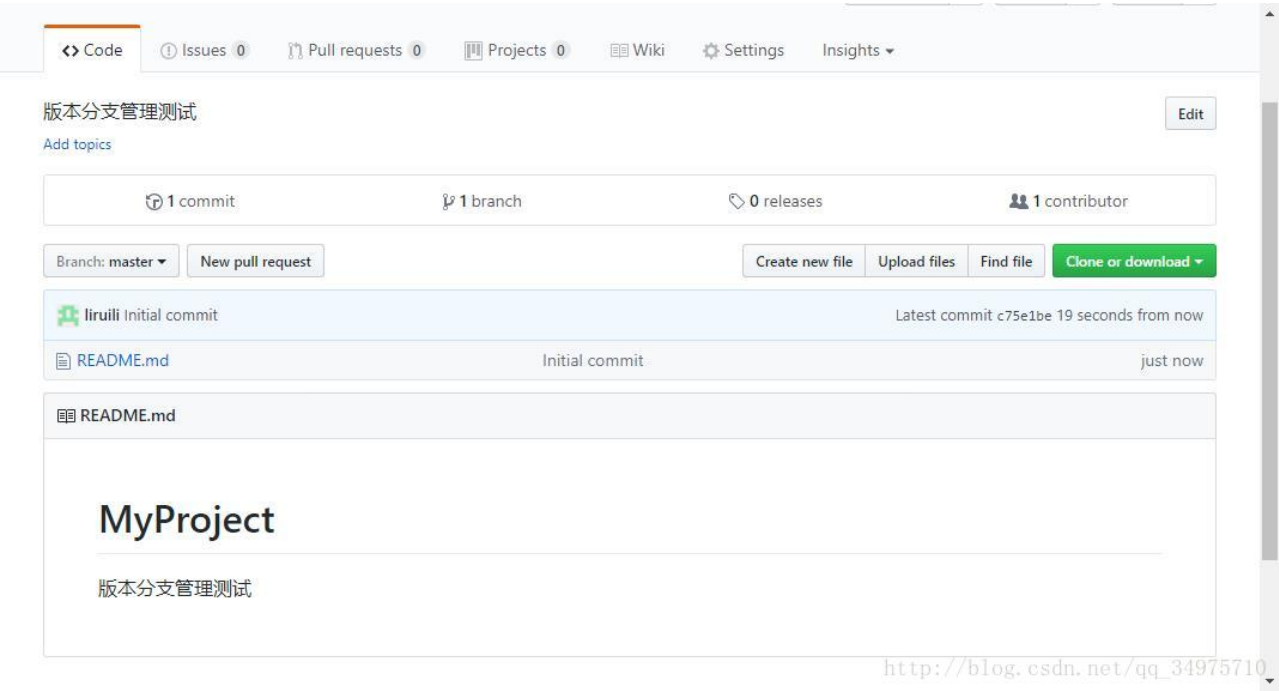
Sourcetree则可以与Git结合，提供图形界面，使用会方便很多。Git和Sourcetree的安装这里就不多说，网上大把的教程。



二、分支的创建与合并

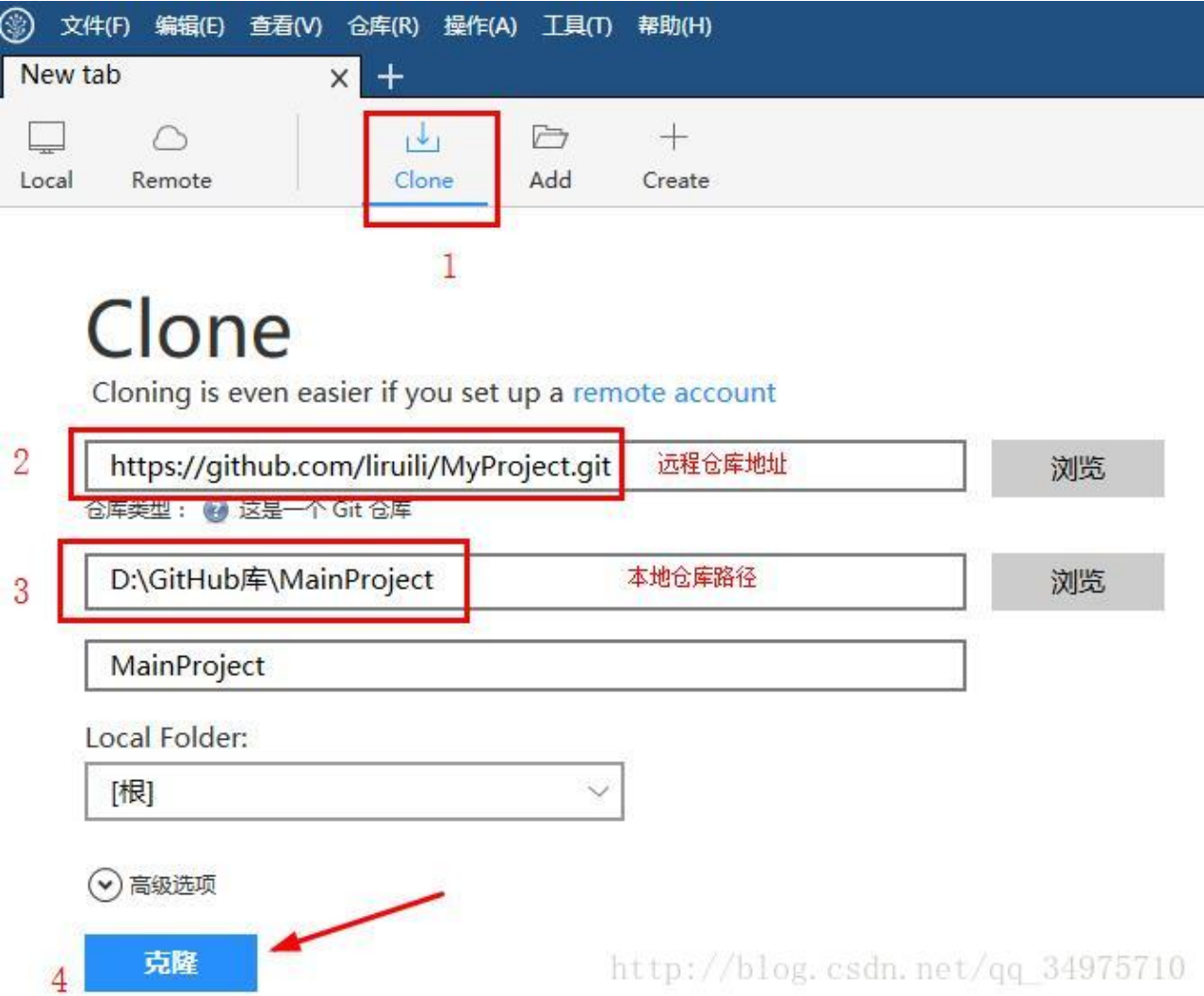
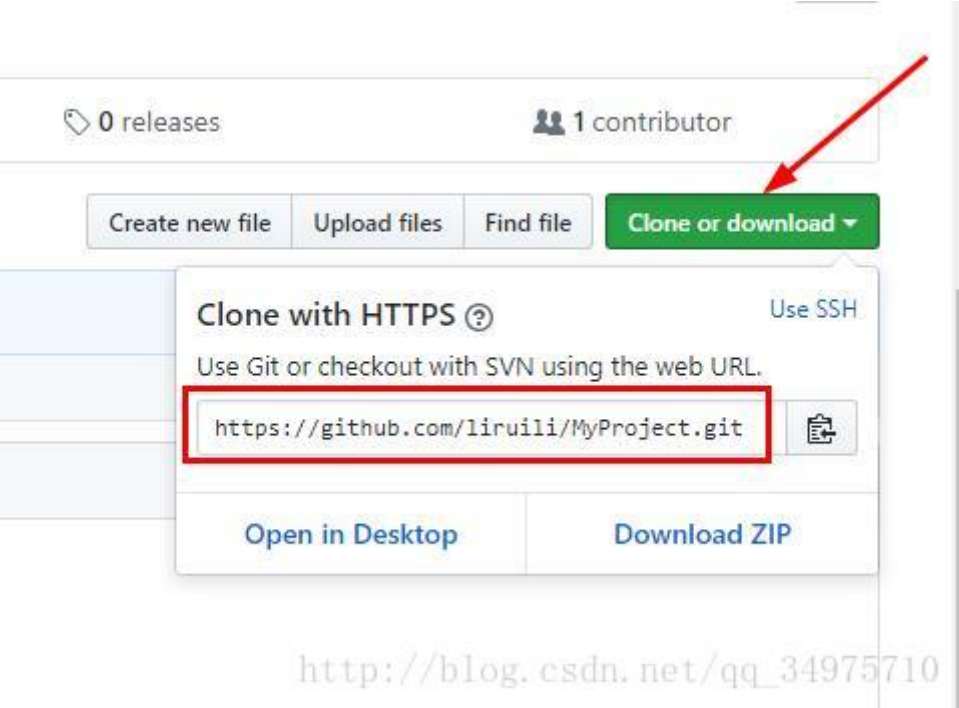
1、创建一个远程仓库

在GitHub官网上创建一个新的远程仓库：



2、用Sourcetree将这个远程仓库 clone到本地

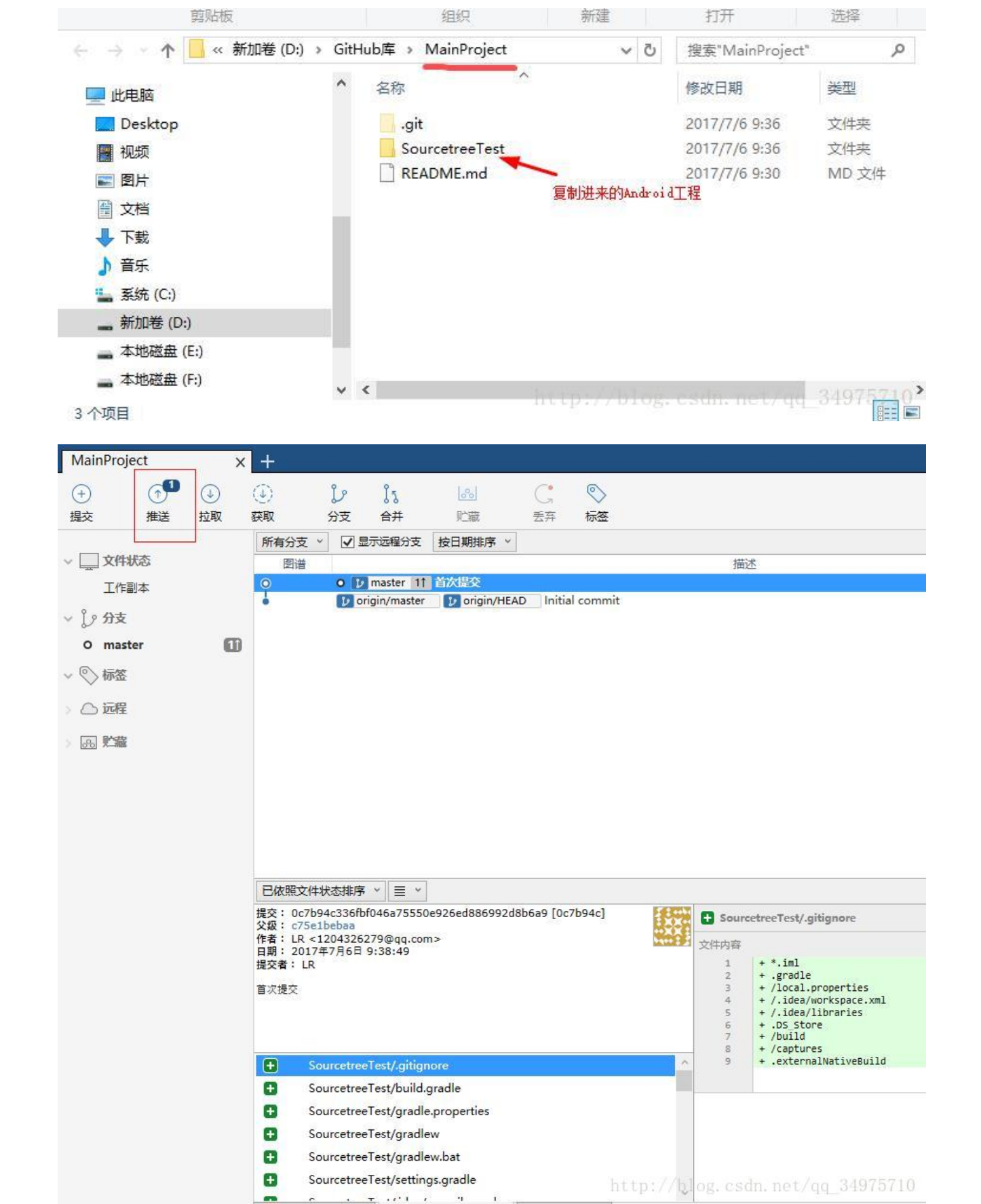
复制远成仓库的地址，然后利用改地址将远程仓库clone下来：



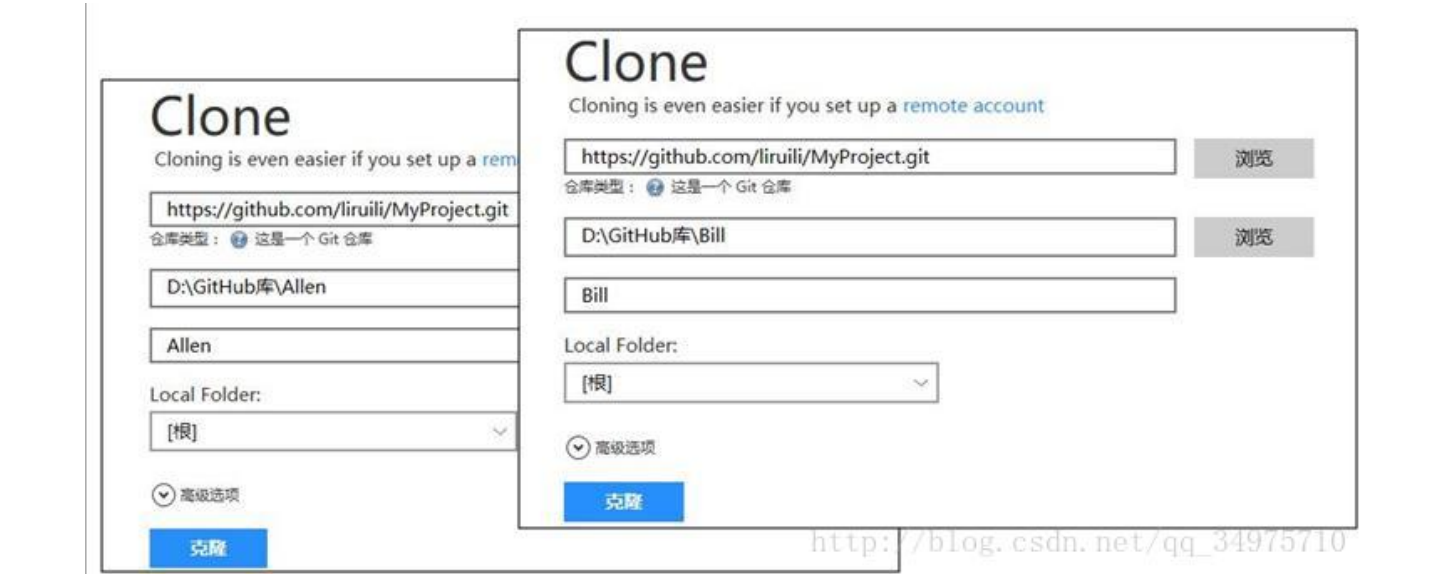
这样，在本地就创建好了一个本地仓库MainProject，可以到目标地址去查看一下。

3、准备测试用工程

这里创建一个android工程，不需要写什么代码。将创建好的整个android工程放到本地仓库的文件夹中，然后推送到远程仓库，这样测试用的工程基本准备好了：

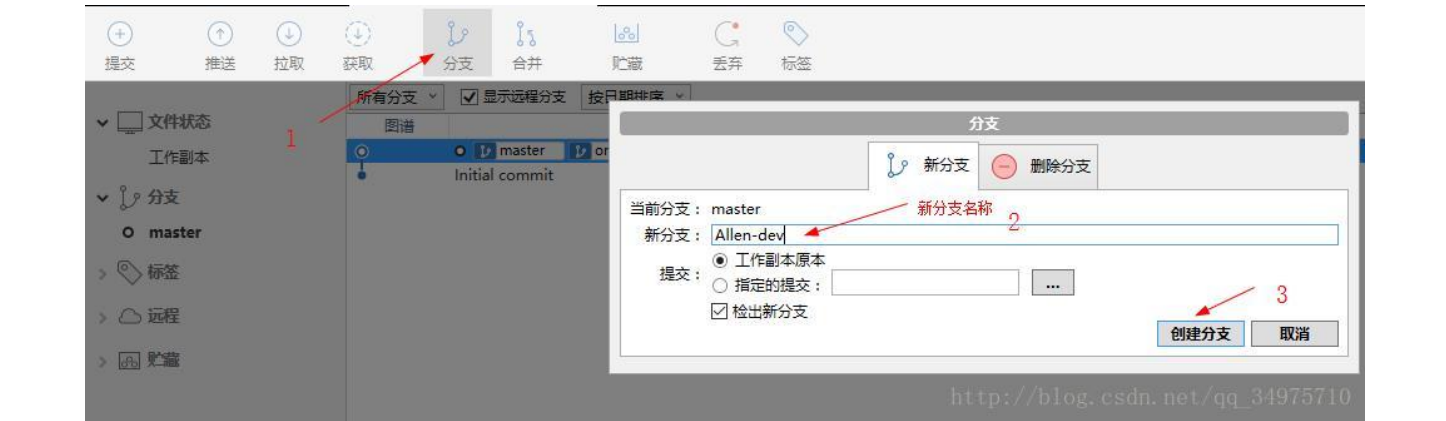


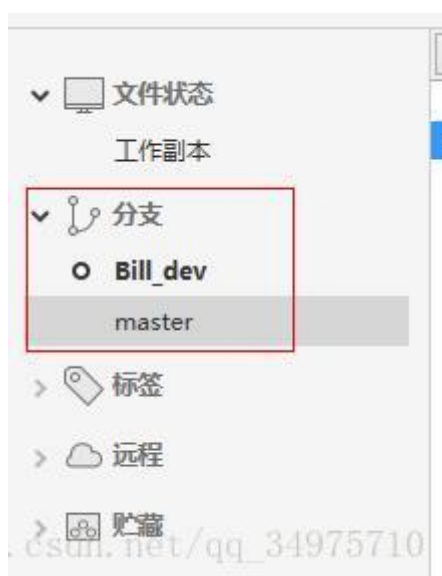
此时远程仓库中已经有一个项目了，下面模拟一个场景：假设有两个程序猿Allen、Bill同时在开发这个项目，项目经理要求Allen增加一个听歌的功能，要求Bill增加一个游戏的功能，那此时这两人就必须将远程仓库中的项目clone到他两各自的本地（这里就用一台电脑模拟，clone两次创建两个本地仓库）：



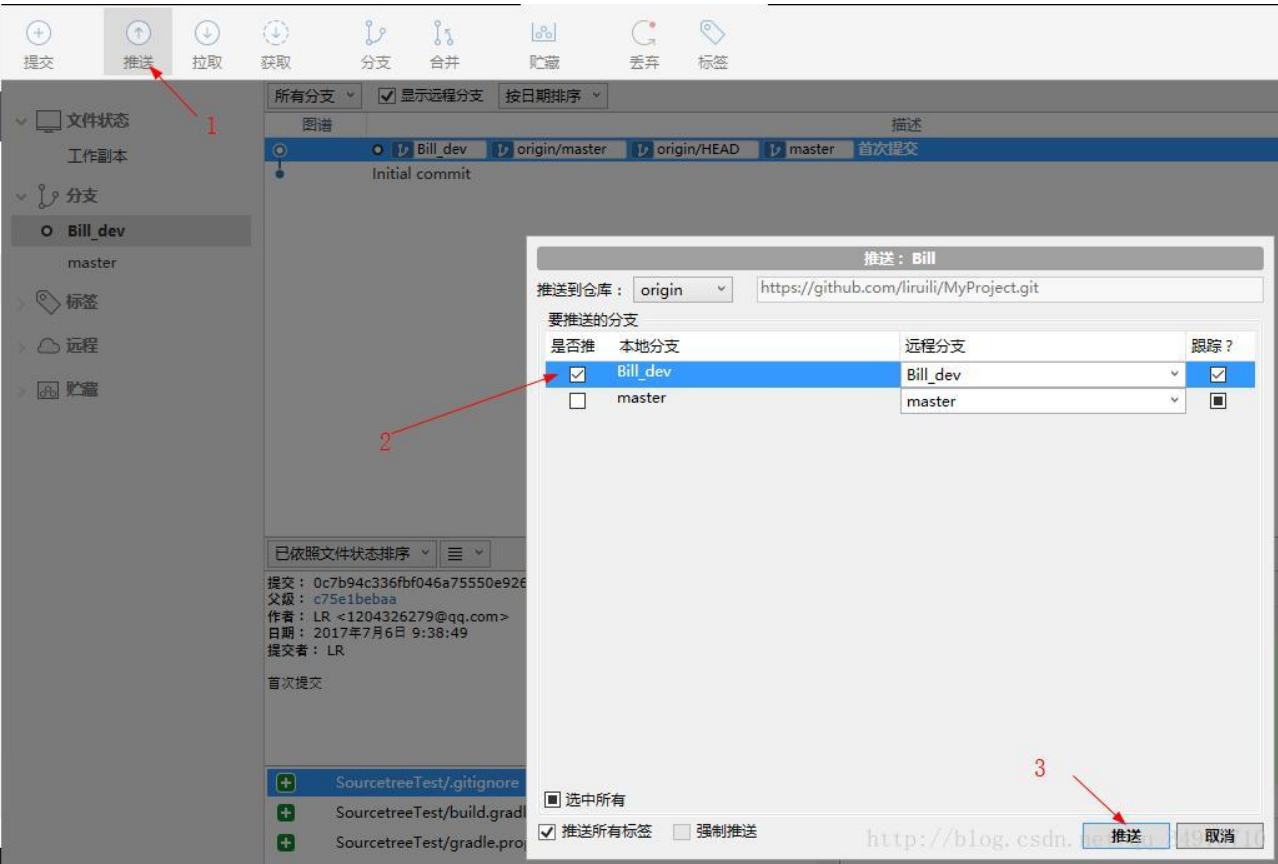
clone下来后会有一个默认的分支master，可以理解成主分支，那去进行项目开发的话不会直接使用master，会去创建一个新分支进行开发，避免直接使用master改来改去最后一团糟那就该崩溃了。每个人在各自的新分支中开发完成后将新分支合并到主分支中就可以了。

接下来Allen和Bill各自创建一个新分支：Allen_dev和Bill_dev

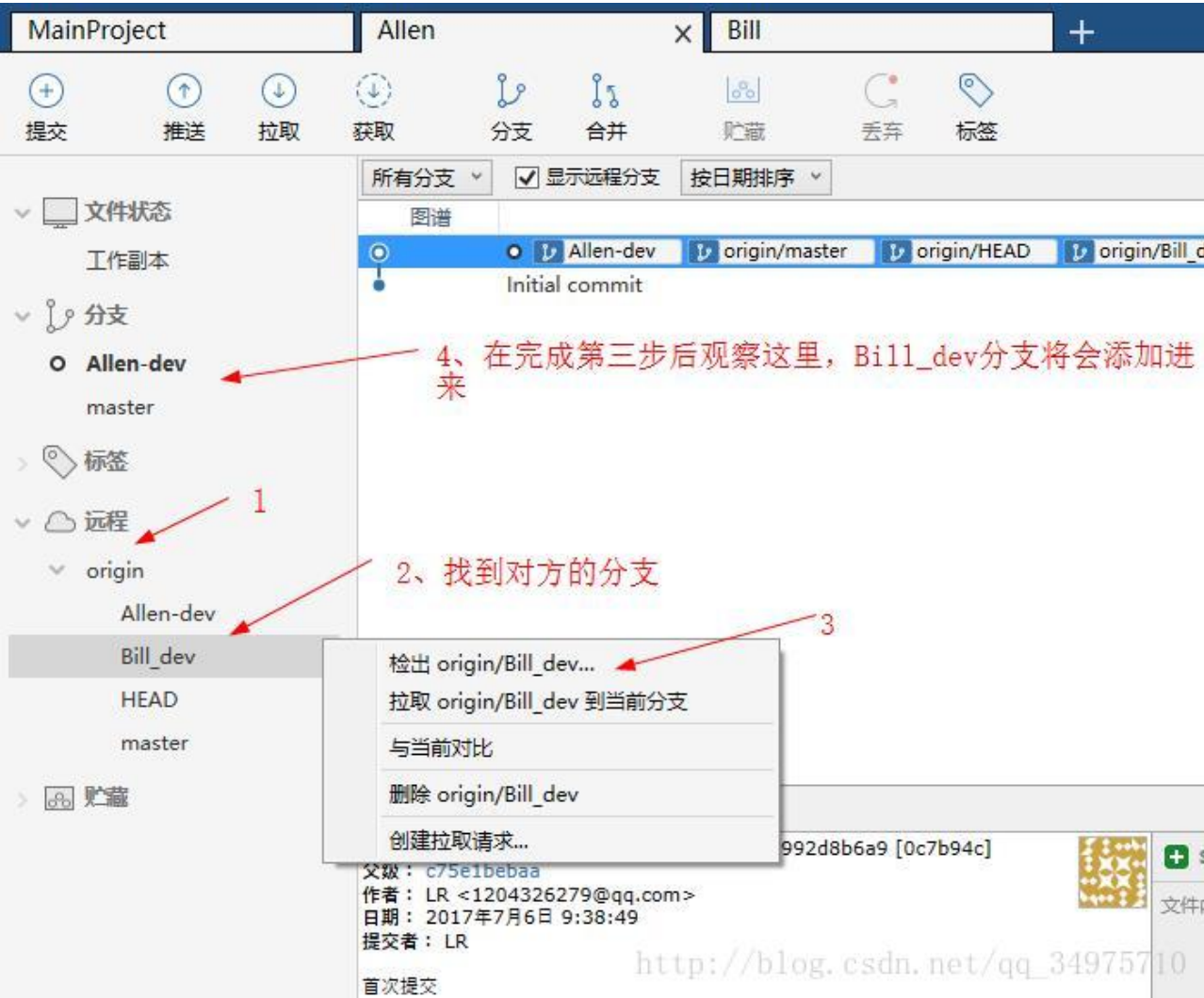




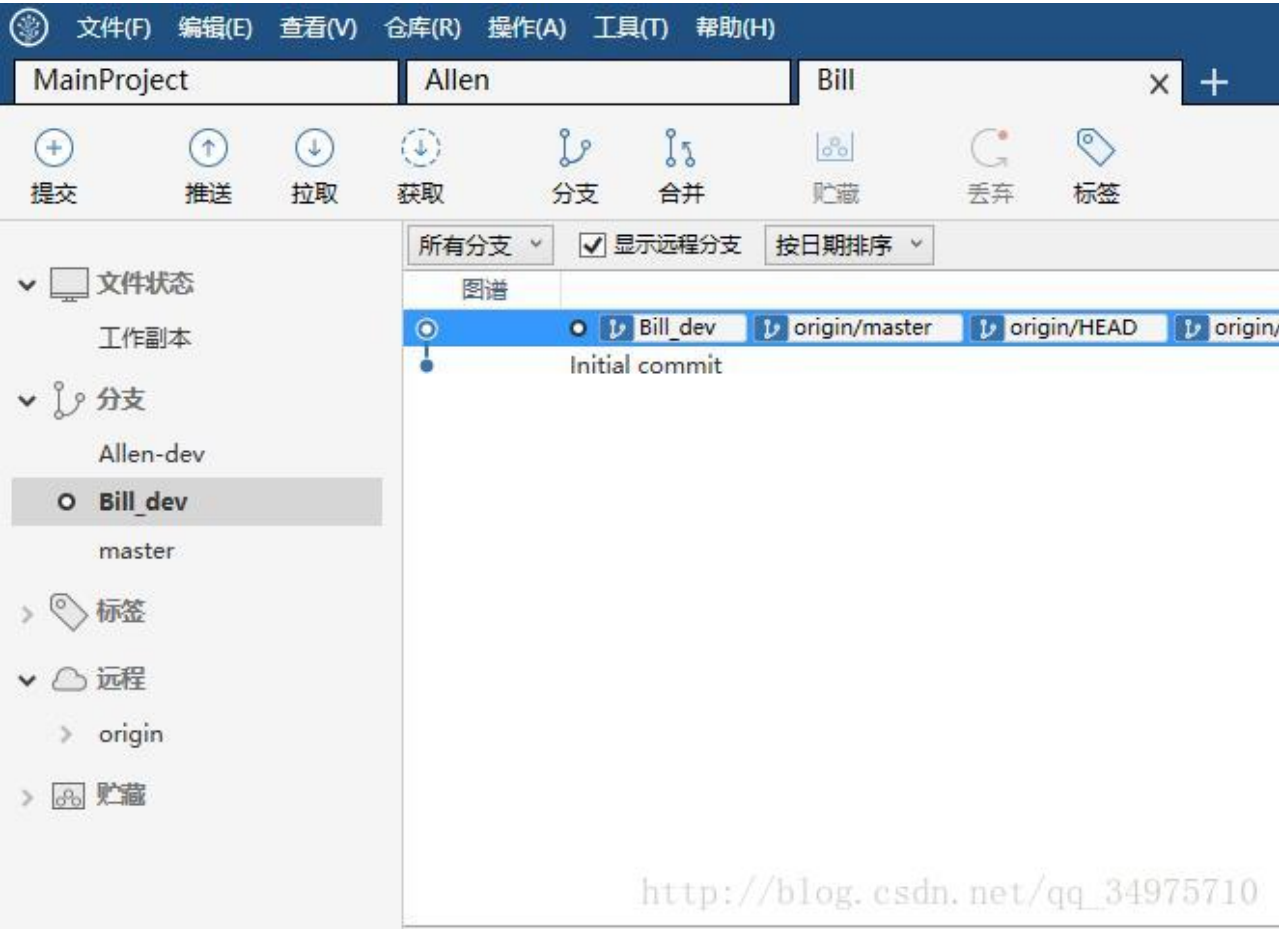
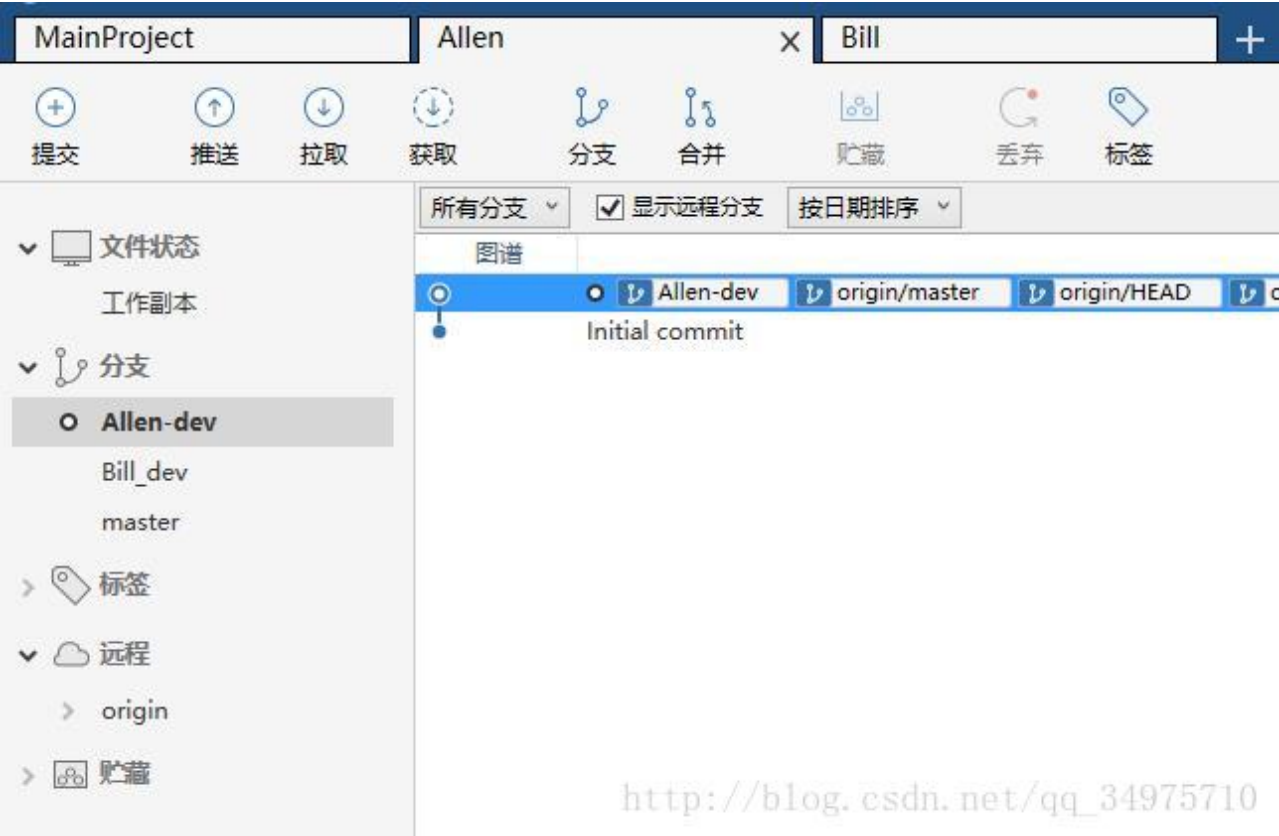
那如果Allen和Bill在开发过程中想看看对方的代码，那就必须将各自创建的新分支推送到远程仓库，然后将对方的分支拉取下来，每次想看的话先获取，然后再拉取最新的代码到本地仓库即可。



首次拉取别人的分支，在上面的获取之后，按如下操作，这里是Allen获取Bill的分支 Bill_dev，Bill获取Allen的步骤一样：



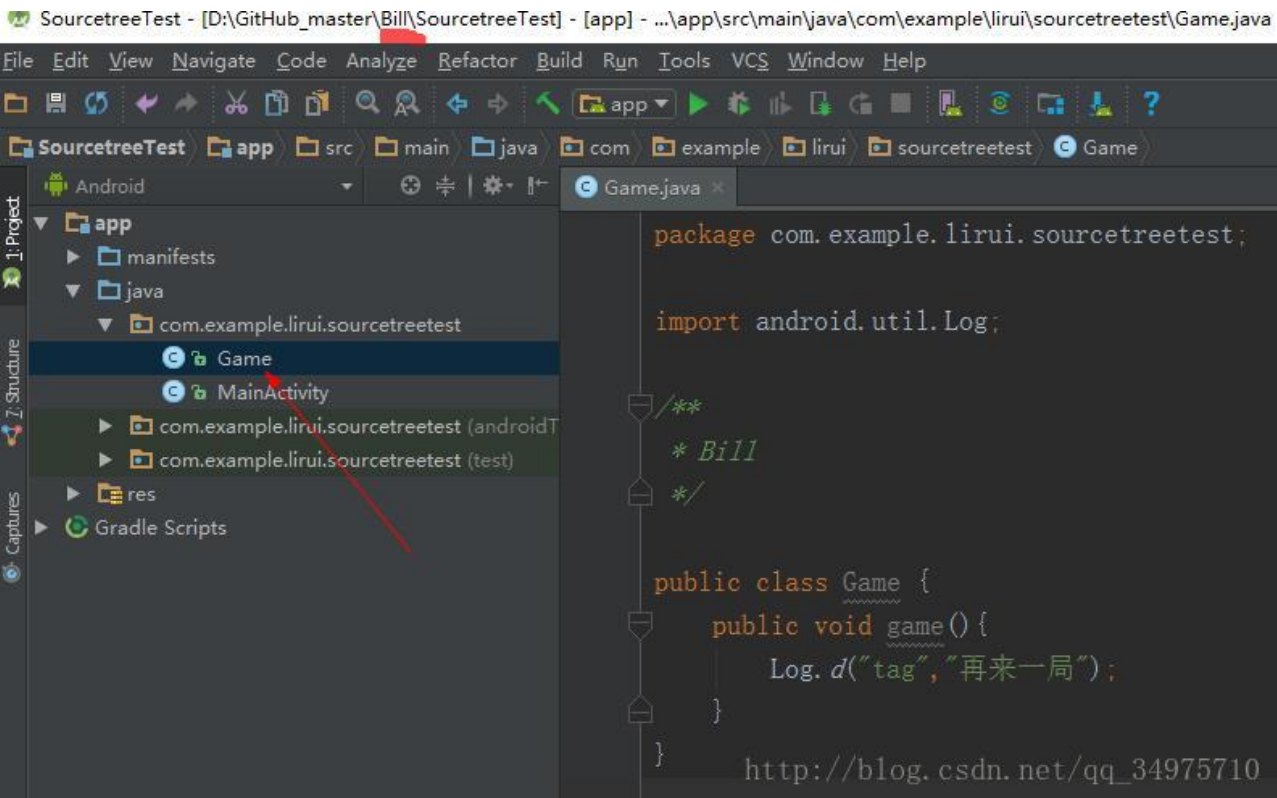
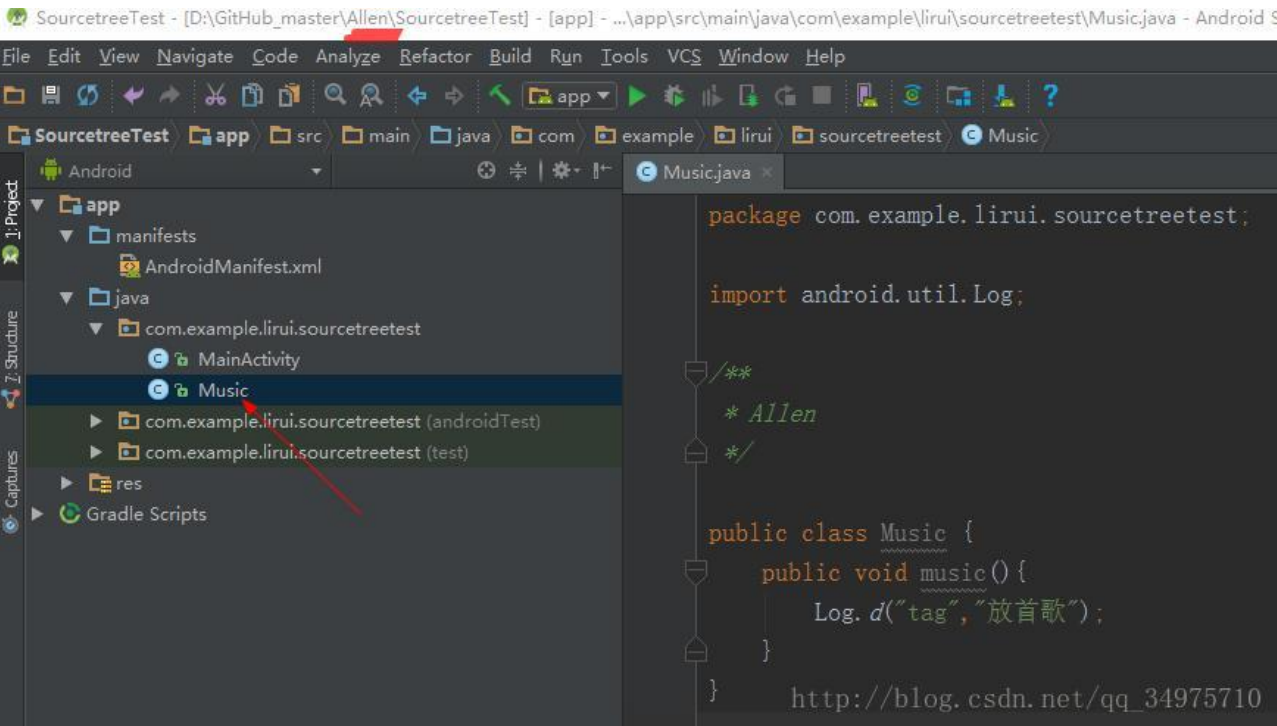
注意：黑色加粗表示当前所处的分支，可以任意双击切换！



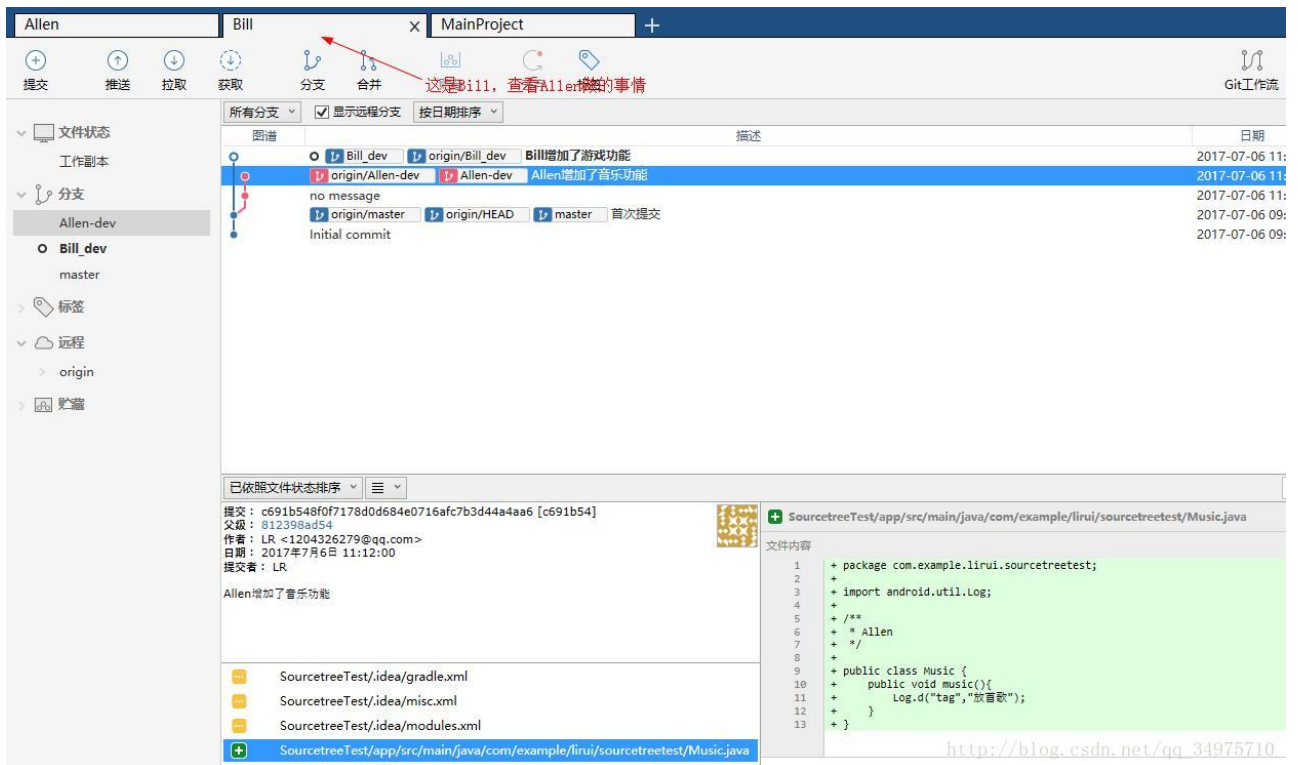
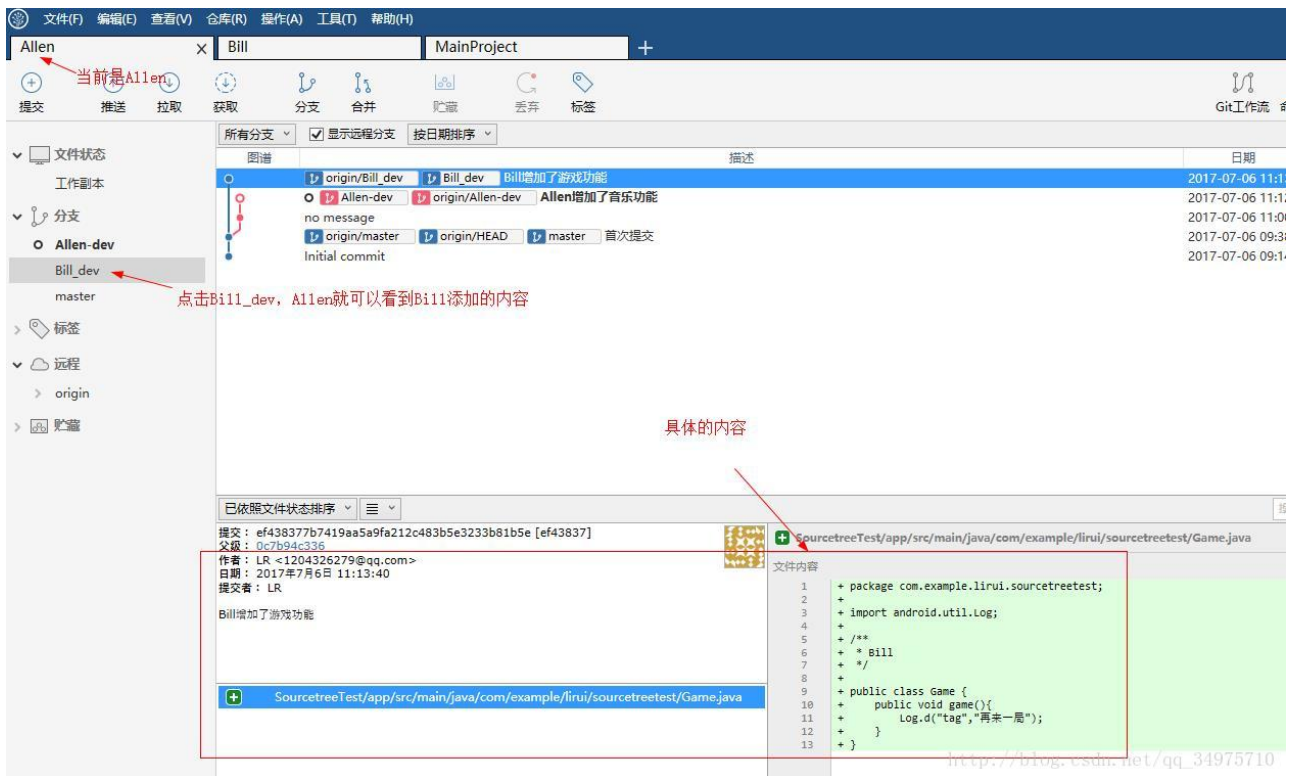
到此，各自的分支已经创建好了，下面Allen和Bill就可以在各自的新分支上进行项目经理安排的任务。

3、分支合并

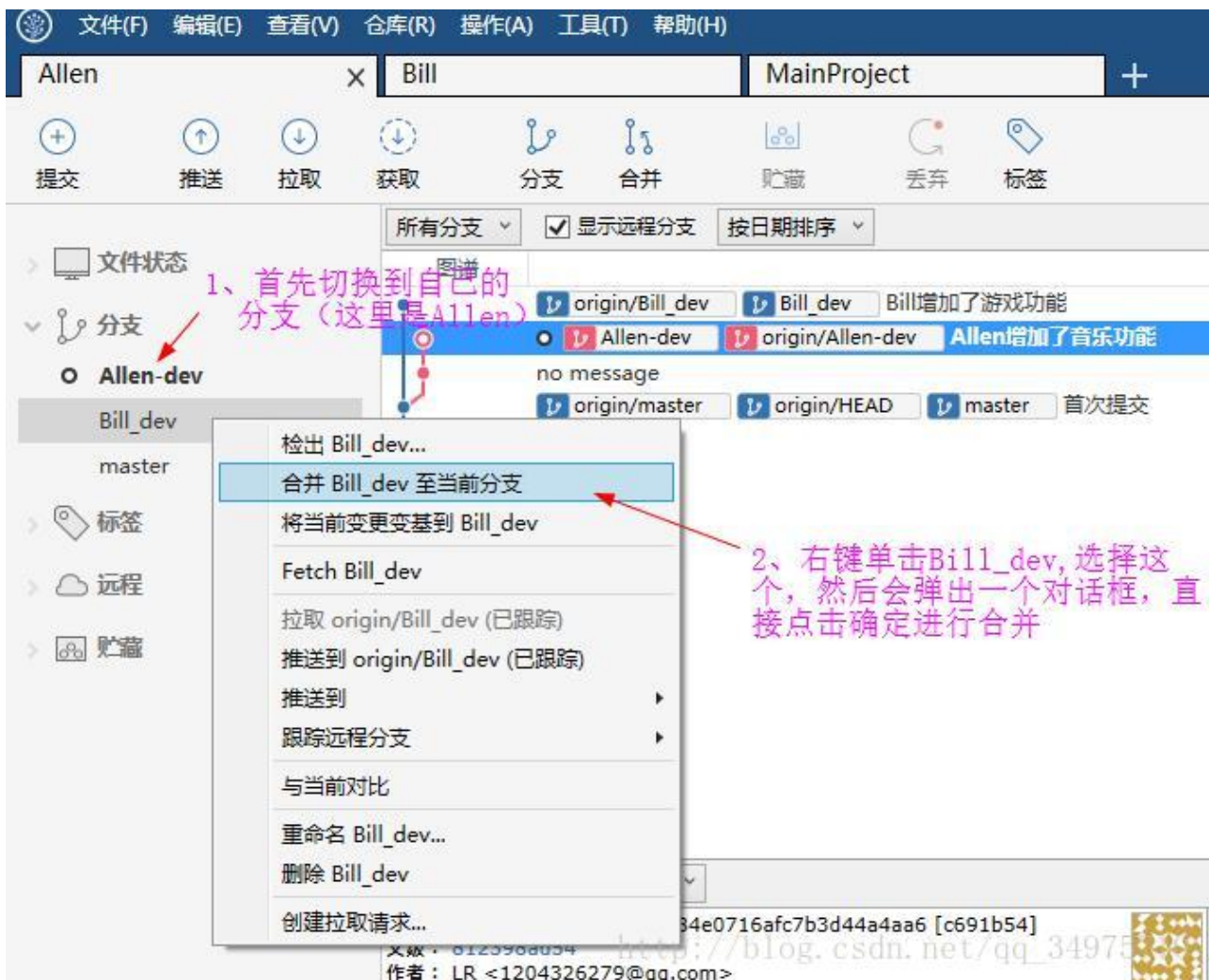
Allen和Bill打开各自本地仓库中的项目代码进行开发，顺利完成项目经理交代的任务，然后各自的分支推送到远程仓库：



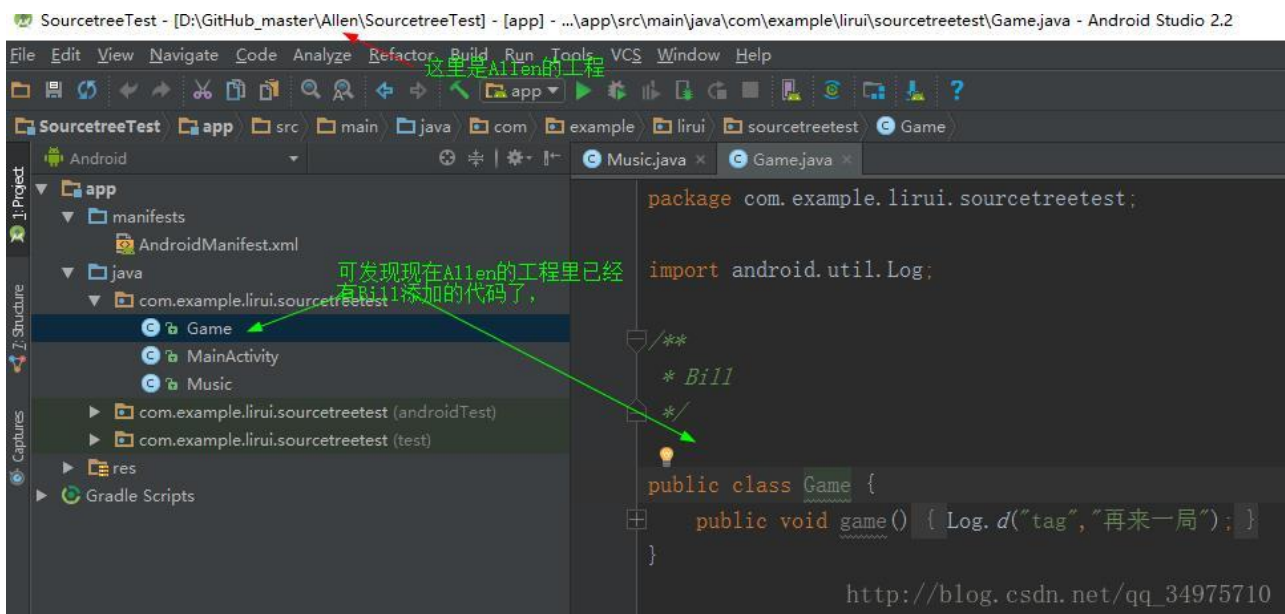
此时双方可以获取最新代码，拉取最新分支代码，拉取完成后就可以看到对方所增加的内容：

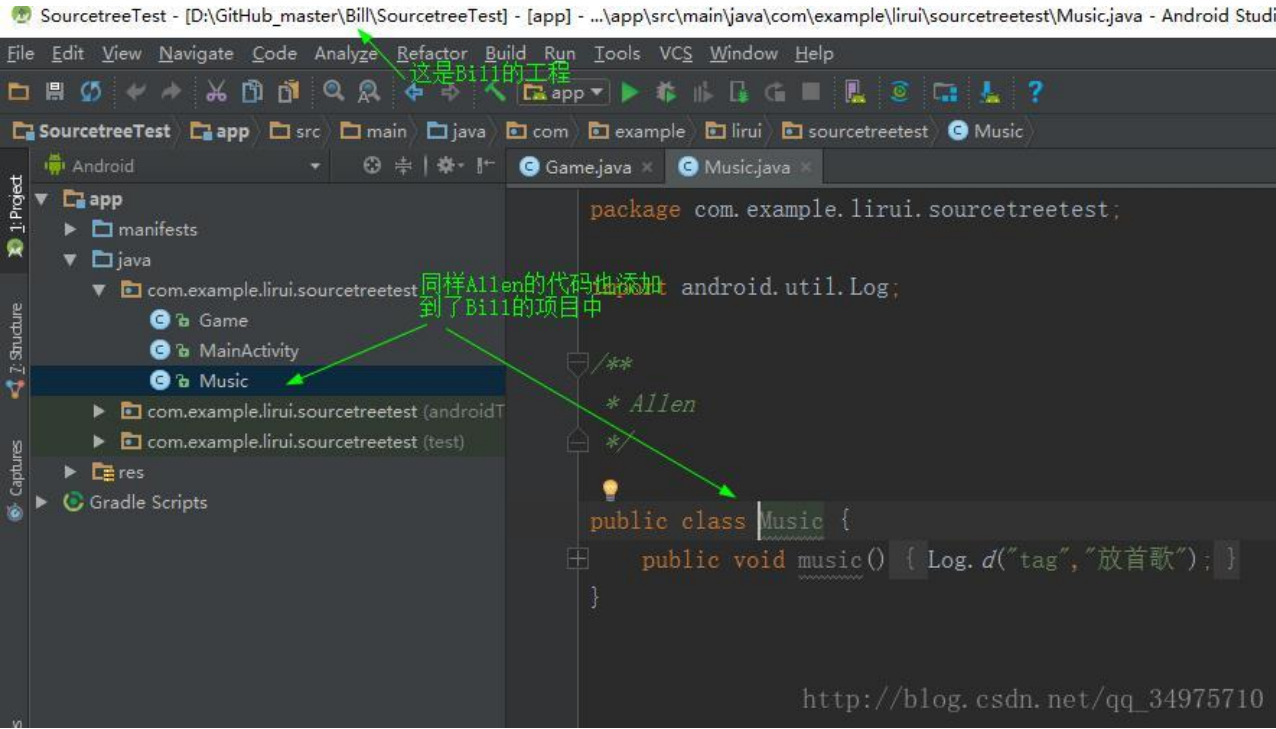


此时，各自都想将对方的代码整合到自己的项目中，这就需要分支的合并。例如Allen合并Bill的分支：



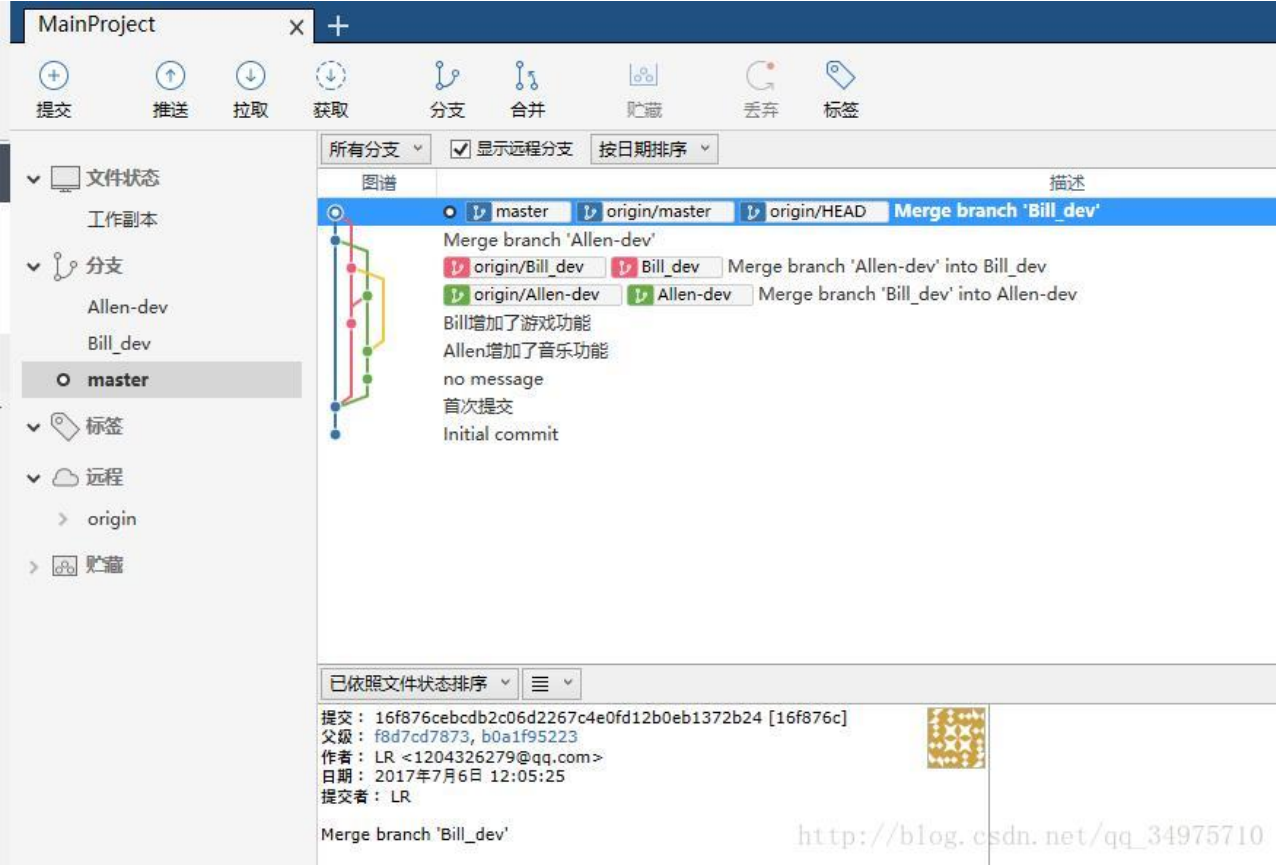
合并完成后, 可以看到之前打开的Allen的Android工程, 会发现Bill添加的功能已经在Allen的项目中显示出来了。





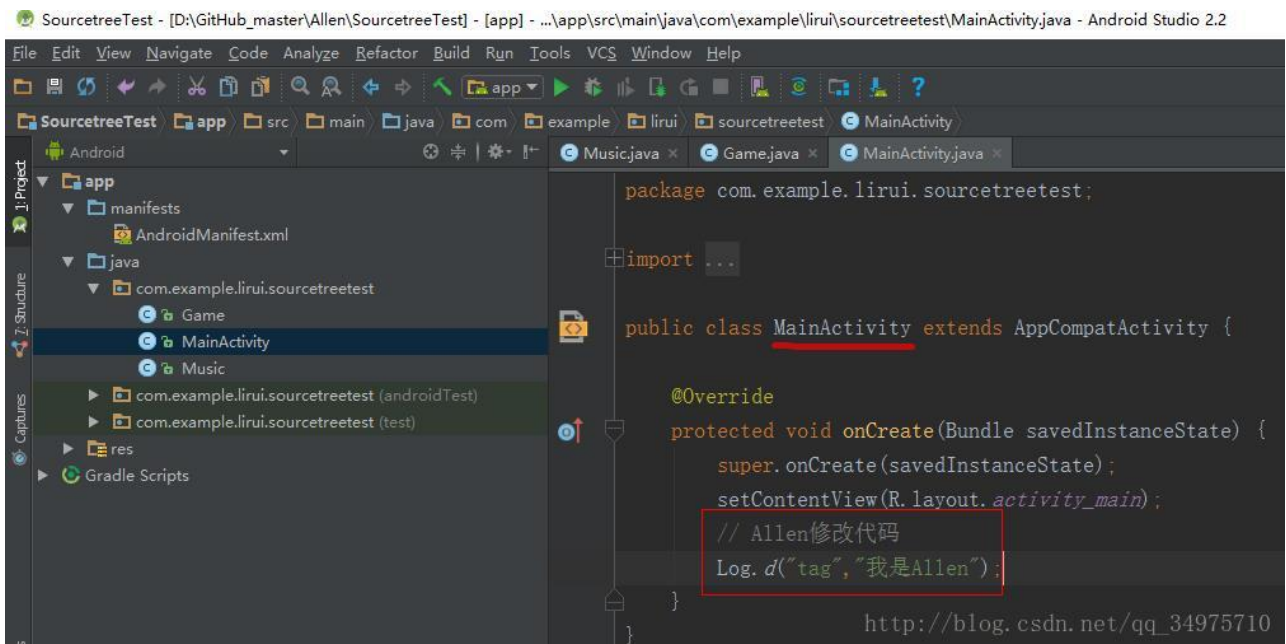
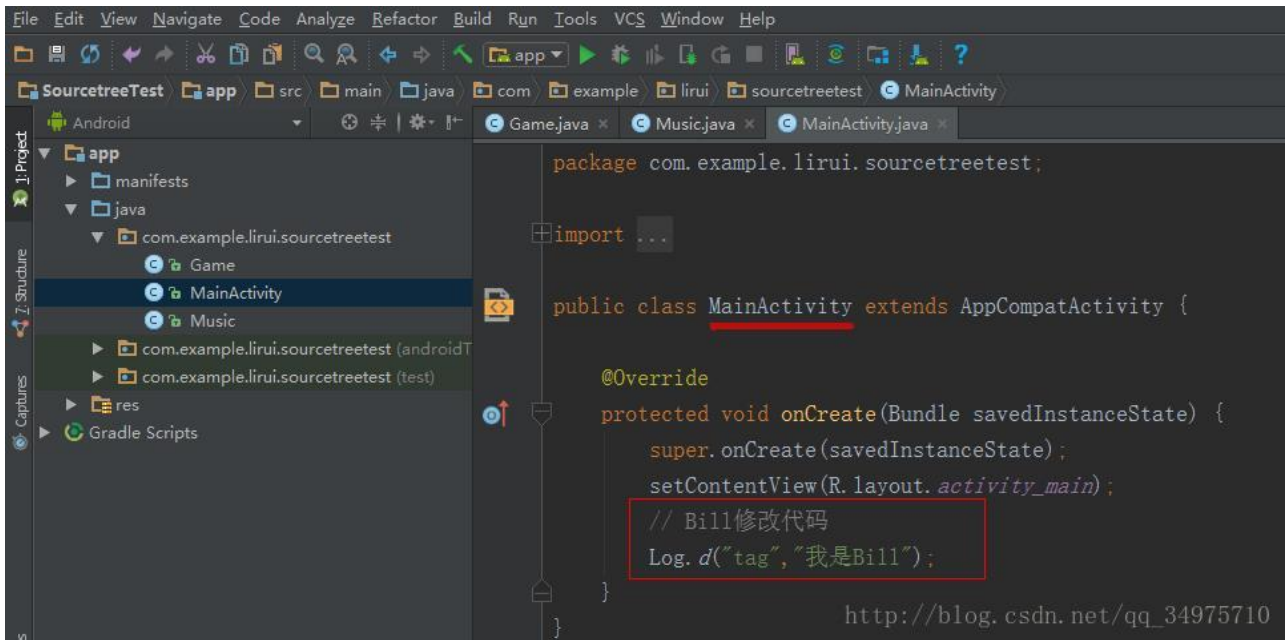
最后不要忘了将各自合并后的分支推送。

现在Allen和Bill的工作已经完成了，各自的分支也推送到了远程仓库，此时项目经理就可以clone远程仓库的项目到本地，拉取Allen和Bill的提交的最新分支，将它们合并到主分支master中。这里就拿最开始创建的MianProject看作是项目经理，合并完成并推送。



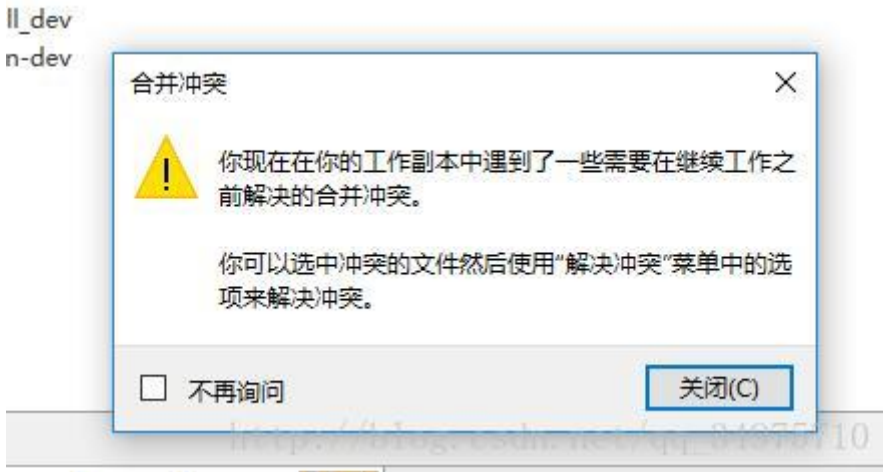
3、解决冲突

到目前为止Allen和Bill各自进展顺利，但是假若两人**同时**在**同一个文件**进行了操作，那最后合并时**就会出现冲突**。比如在MainActivity：

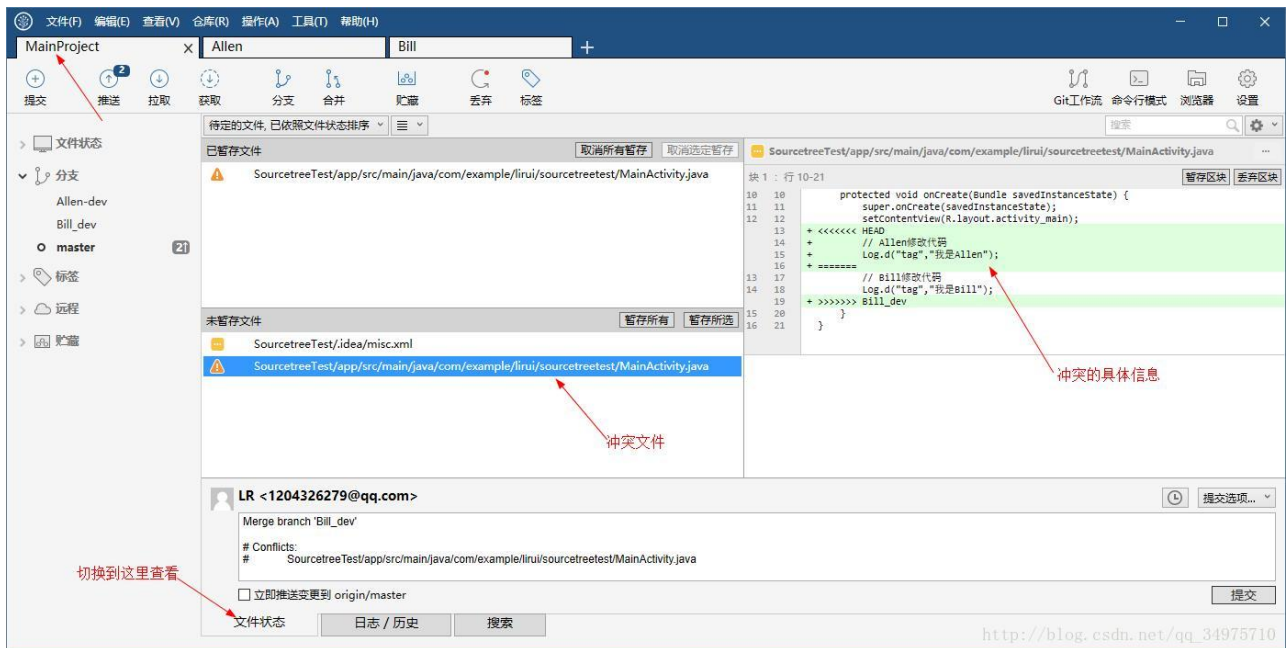


此时当Allen和Bill推送自己的分支到远程仓库后，项目经理MainProject拉取代码进行合并：

假设首先合并Allen_dev到master没有问题，但是当合并Bill_dev到master时就会提示合并出现了冲突，需要解决。



点击关闭，然后切换到文件状态，找到出现冲突的文件，会有相应的冲突信息：

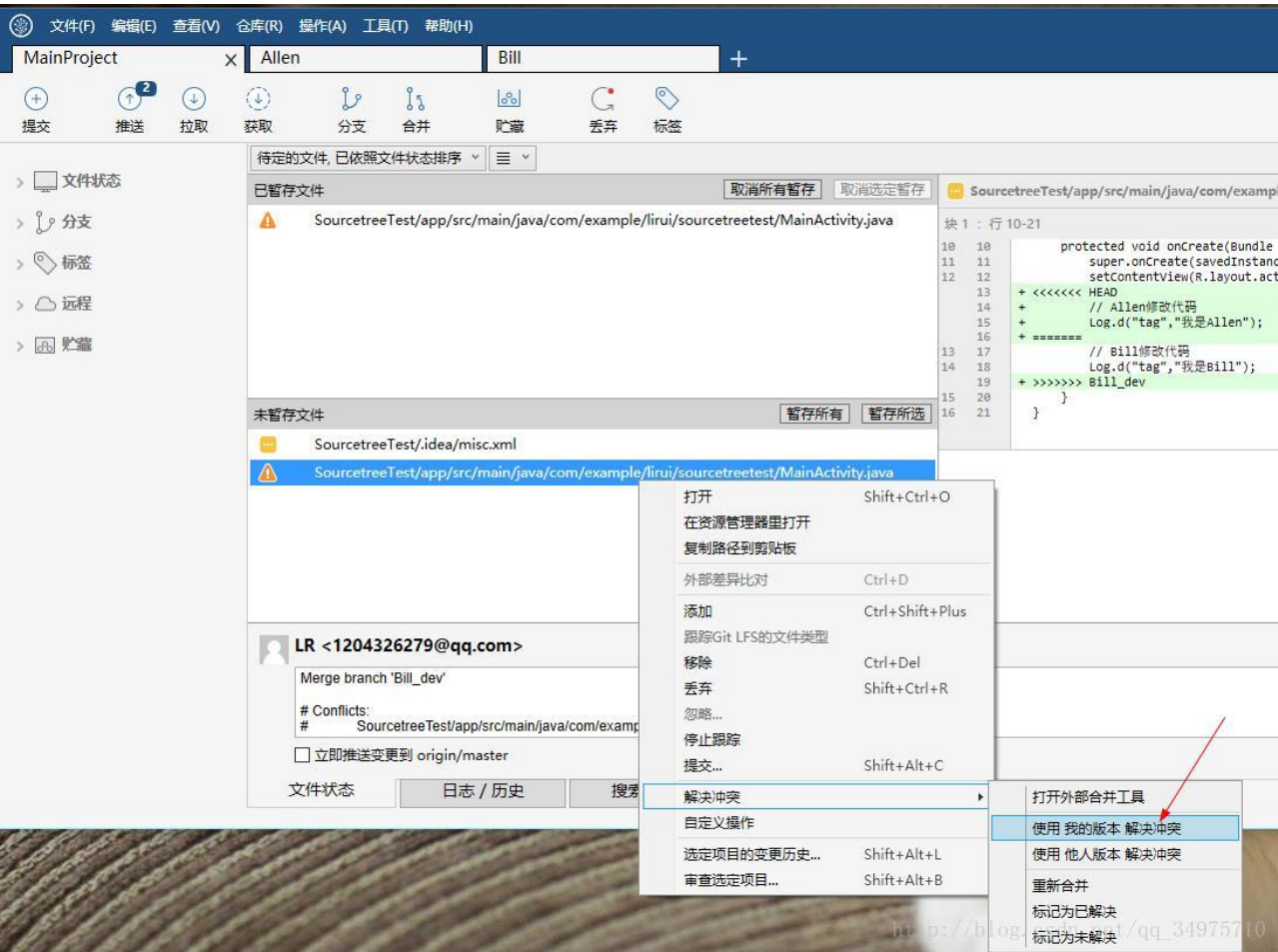


有冲突就要解决，右键单击冲突文件，选择解决冲突，这里有两个选项：

- 1、使用 我的版本 解决冲突
- 2、使用 他人版本 解决冲突

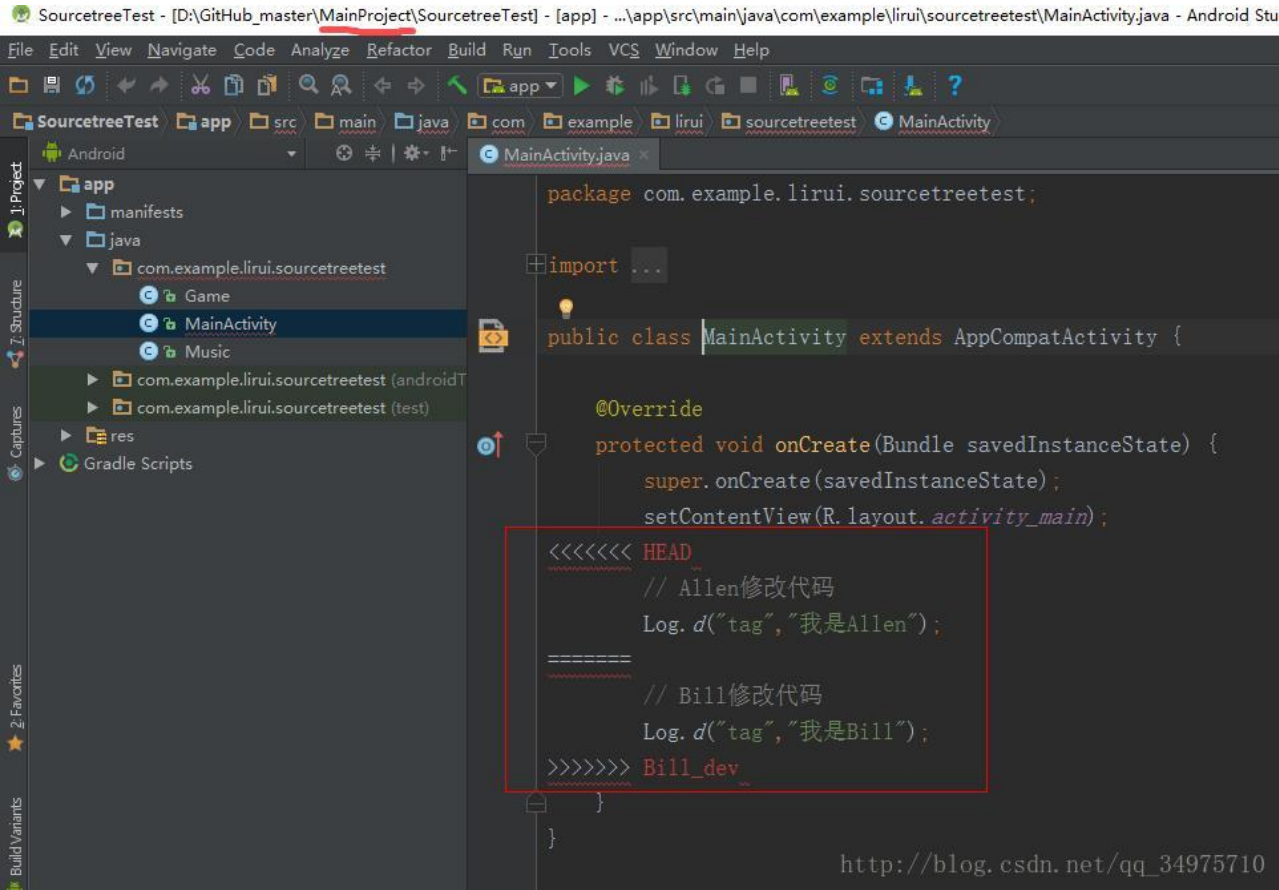
这里项目经理MainProject首先是将Allen的分支合并到主分支master，那么“我的版本”就是对应的Allen的，“他人版本”对应的就是Bill的。如果首先合并Bill的分支，那么对应关系就要对调一下。总的来说，“我的版本”对应的是首先合并到主分支master的。

采用一个人的版本，那么在冲突文件中就只会保留该人修改的代码，例如我这里就选择“使用 我的版本 解决冲突”，那么在MainActivity中就只会保留Allen添加的代码。

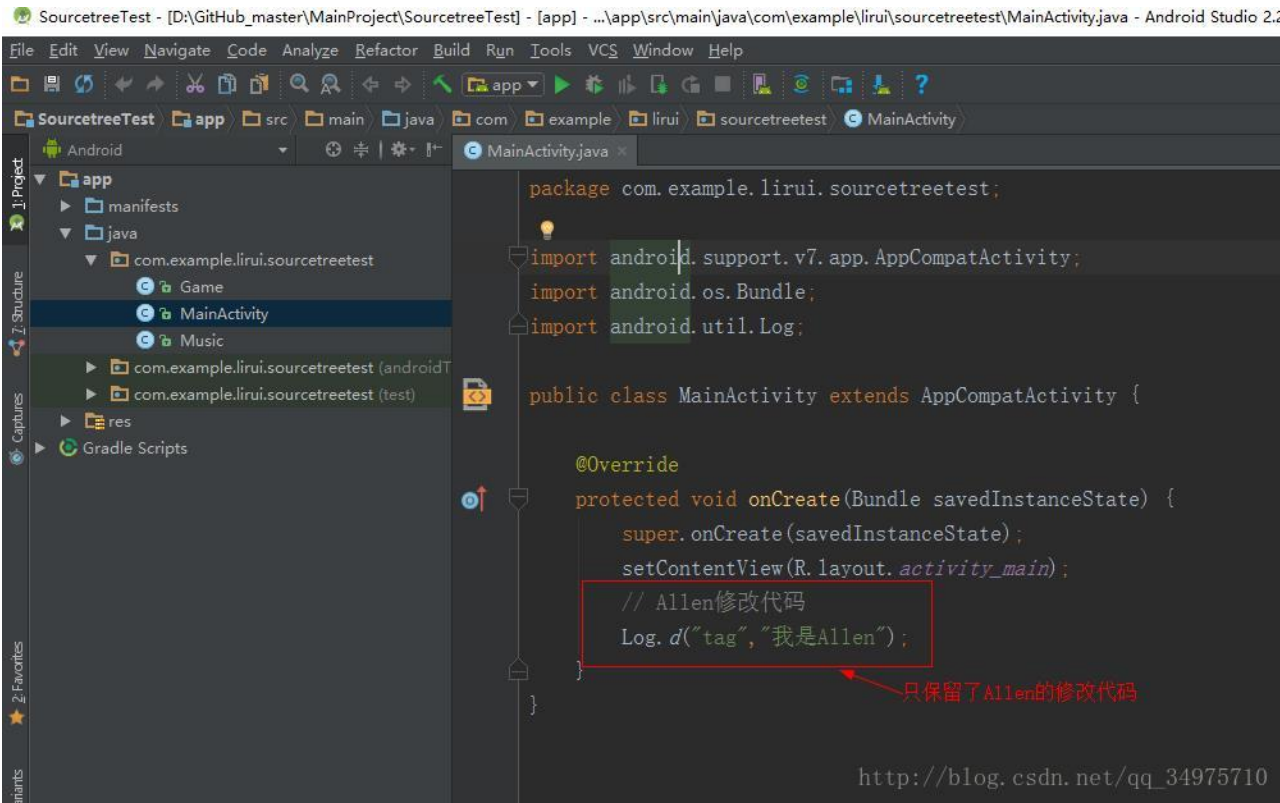


弹出对话框点击确定。

解决前：



解决后：



好了，到这为止基本的分支创建与合并的简单应用，还有合并冲突解决就介绍完了。

生活不只是敲代码，如果你或你身边的人喜欢摄影或者生活的点点滴滴，可以关注下我亲爱的公众号~



イマーシブリーダーに関するフィードバックをお寄せください。この Web ページのコンテンツは正しく表示されましたか?

