## 创建项目

1）git config -global user.name "Your name"

git config -global user.email ["you@example.com"](mailto:\"you@example.com\")

1. 建立一个存放工程的文件夹

mkdir test

1. 进入此文件夹

Cd test

1. git init命令用于初始化当前所在目录的这个项目

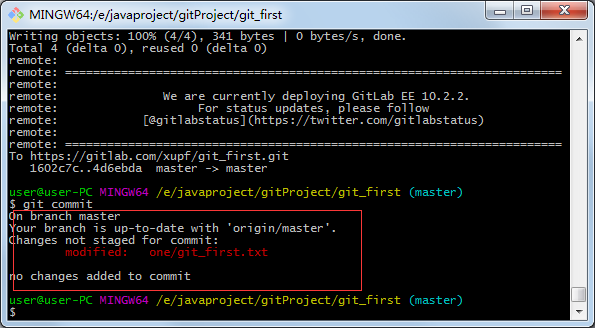
git init

这是会创建一个隐藏的.git文件

1. 在创建一个新的文件夹

mkdir test\_git

1. 在新创建的文件夹中创建一个文件
2. Git status查看项目日志
3. 之后git commit会提示下图所示：



说明没有将改变的文件放入缓存区中，这个时候可以git add ‘文件名’ 将其添加到缓存区中，之后再commit就没事了。

也可以直接用git commit -a 立即提交所有的改变

之后git push 就行了

## 创建分支

1. git branch ’experiment 分支的名字’ 创建一个分支

直接输入git branch，不加任何后续参数，就表示让git列出所有已存在的分支。

前面带“星号”的分支表示当前所在的分支。

1. 切换到分支experiment

git checkout experiment

1. 在git\_first.txt中增加一行，进行1--》8）步，在分干道上push时请用下列命令

git push --set-upstream origin experiment

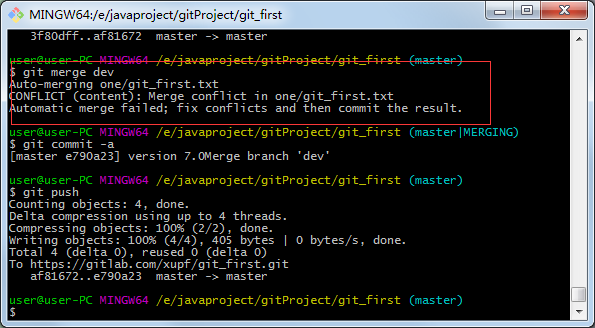
1. 切换到主干道master,在git\_first.txt中涨价一行，进行1--》8）步

## 分支合并

1. 合并分支

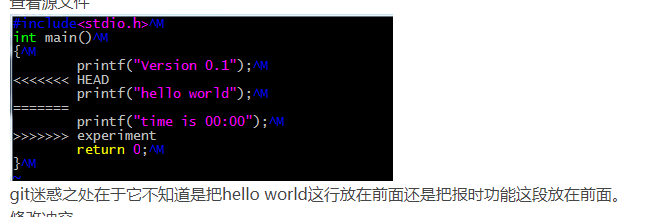
git merge experiment

如果出现下图，说明合并失败

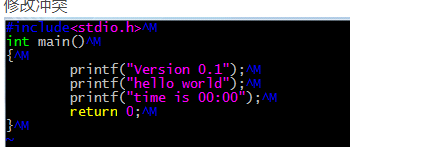


## 解决合并冲突

1. 查看源文件



1. 修复源文件



1. 重新提交文件

git commit -a

1. 合并分支后，对分支进行删除

git branch -d experiment

## 使用sourcetree合并分支

1. 首先将项目重新clone一遍到不同的文件夹下面
2. 为新clone的项目创建分支

git checkout -b dev master

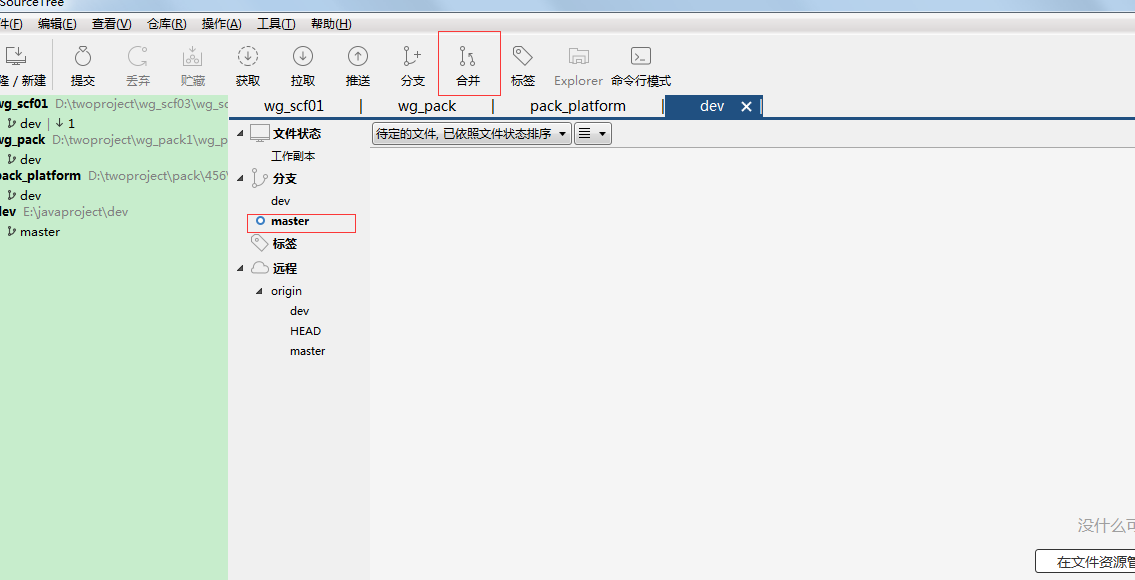
本命令会直接创建dev分支，并转到dev分支

1. 使用如下命令push dev分支到远程仓库中

git push --set-upstream origin dev

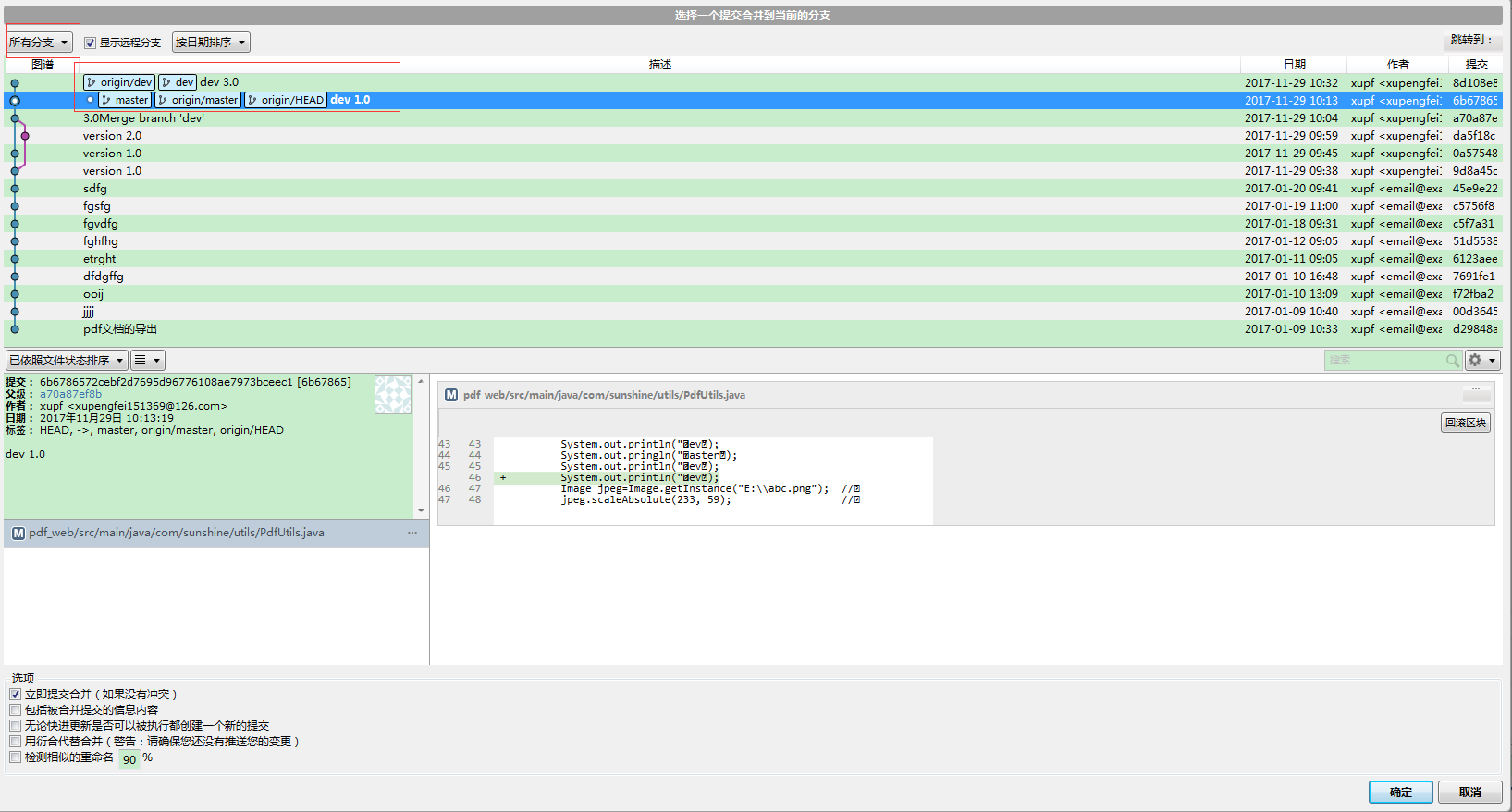
这是可以查看远程仓库中已经存在dev分支了，如果不执行这一命令在远程仓库中是找不到dev分支的

1. 对dev分支进行再次开发
2. 对dev分支和maser分支进行合并，在sourcetree中这样操作：

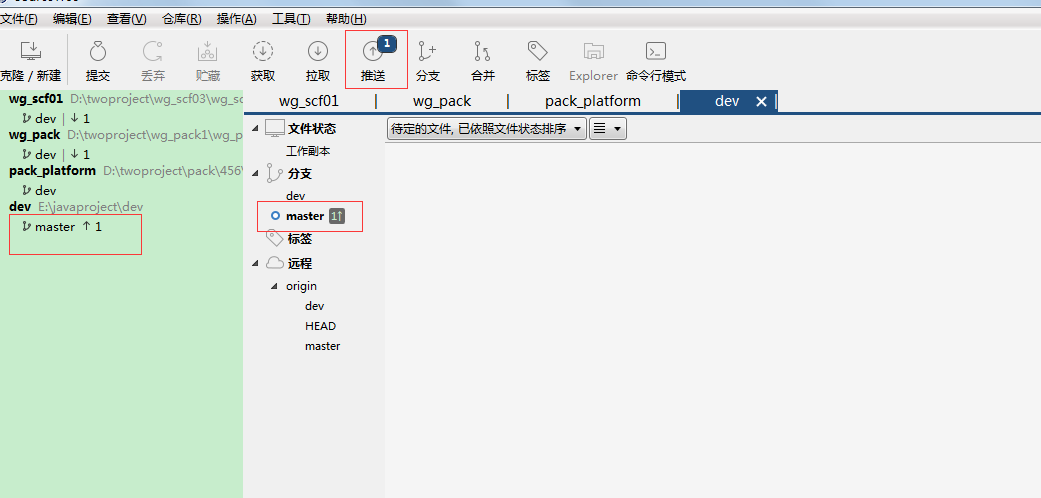


在合并之前，请切换到master主干道

1. 点击合并按钮弹出另一页面



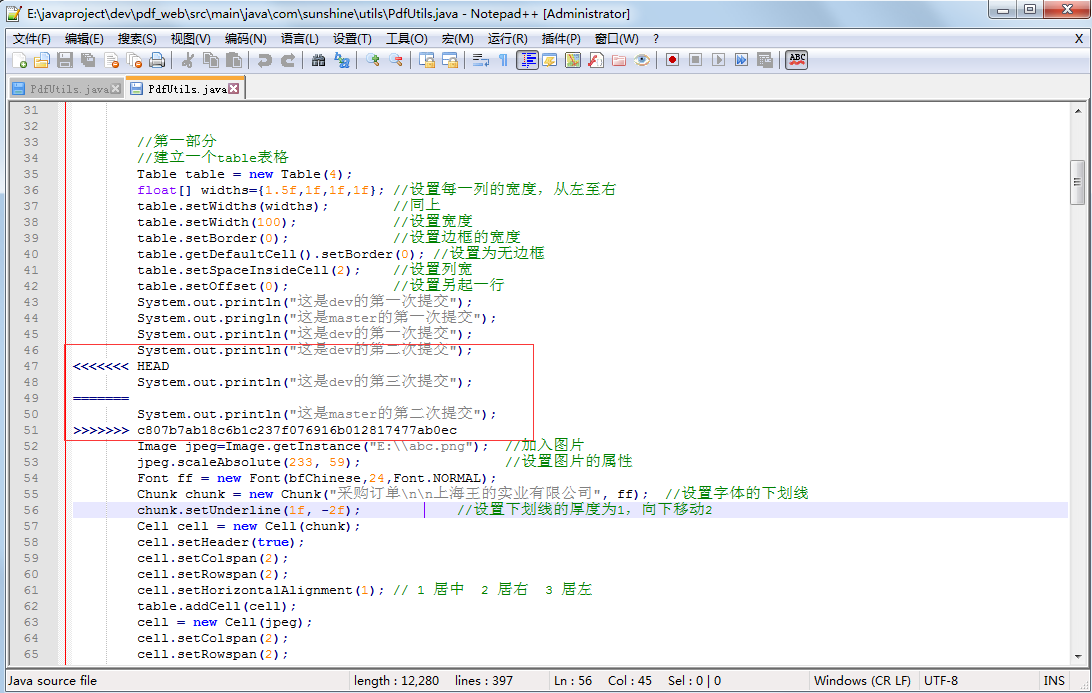
在点击确定后，过几秒会出现如下界面：



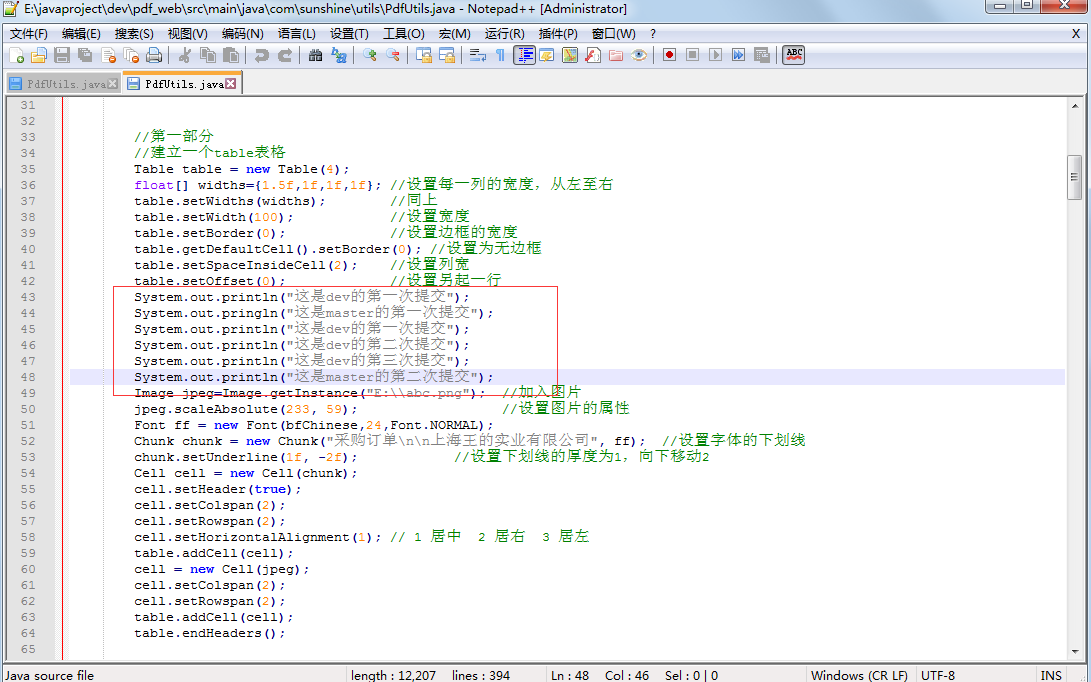
点击确定按钮之后，如果有冲突的话，会提示合并不成功



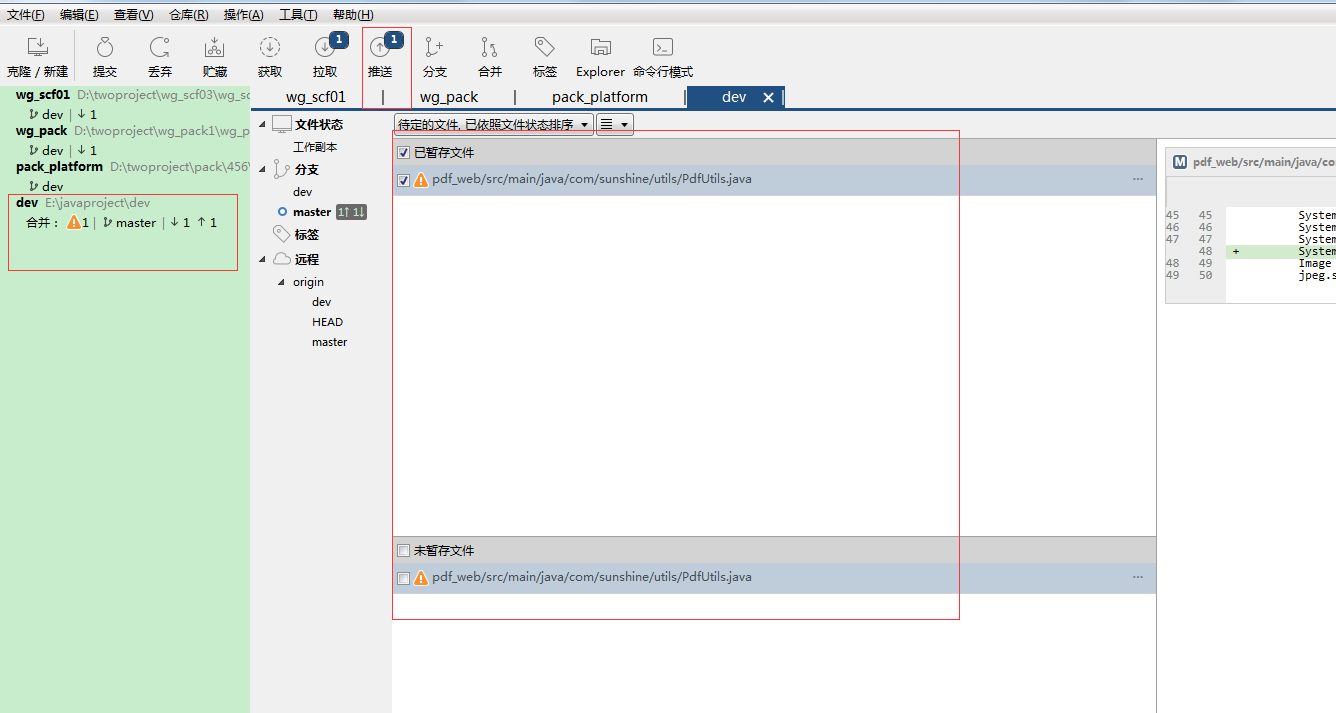
这个时候说明有冲突，我们要首先解决冲突，我们手动打开有冲突的文件：



解决完冲突后如下所示：



之后看一下sourcetree：



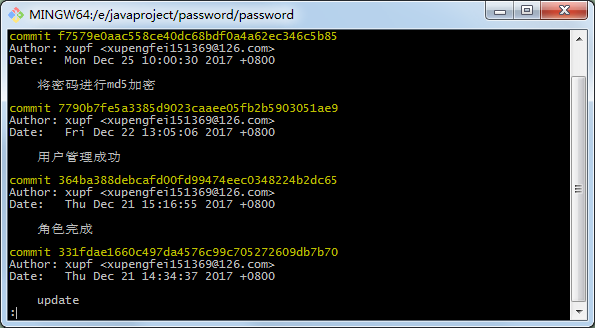
之后选中未暂存文件，进行提交和推送，执行成功

## 打一个tag标签，用作对之前版本的一个标记

首先说一下作用：Git 中的tag指向一次commit的id，通常用来给开发分支做一个标记，如标记一个版本号。

下面就说一下具体的用法：

1. git log 看一下提交的历史



找到最后一次的提交，之后进行添加标签

git tag -a ‘v0.1’ f7579e0aac558ce40dc68bdf0a4a62ec346c5b85 -m ‘v0.1’

1. 提交到远程仓库中

git push origin v0.1

****1.添加标签****： git tag -a version -m "note"

注解：git tag 是打标签的命令，-a 是添加标签，其后要跟新标签号，-m 及后面的字符串是对该标签的注释。

****2.提交标签到远程仓库****：git push origin -tags  
注解：就像git push origin master 把本地修改提交到远程仓库一样，-tags可以把本地的打的标签全部提交到远程仓库。  
****3.删除标签：****git tag -d version  
注解：-d 表示删除，后面跟要删除的tag名字  
****4.删除远程标签：****git push origin :refs/tags/version  
注解：就像git push origin :branch\_1 可以删除远程仓库的分支branch\_1一样， 冒号前为空表示删除远程仓库的tag。  
****5.查看标签：****git tag或者git tag -l