

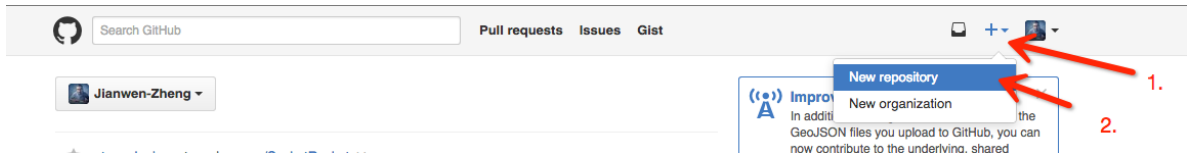
git 团队开发

JWDev.cn

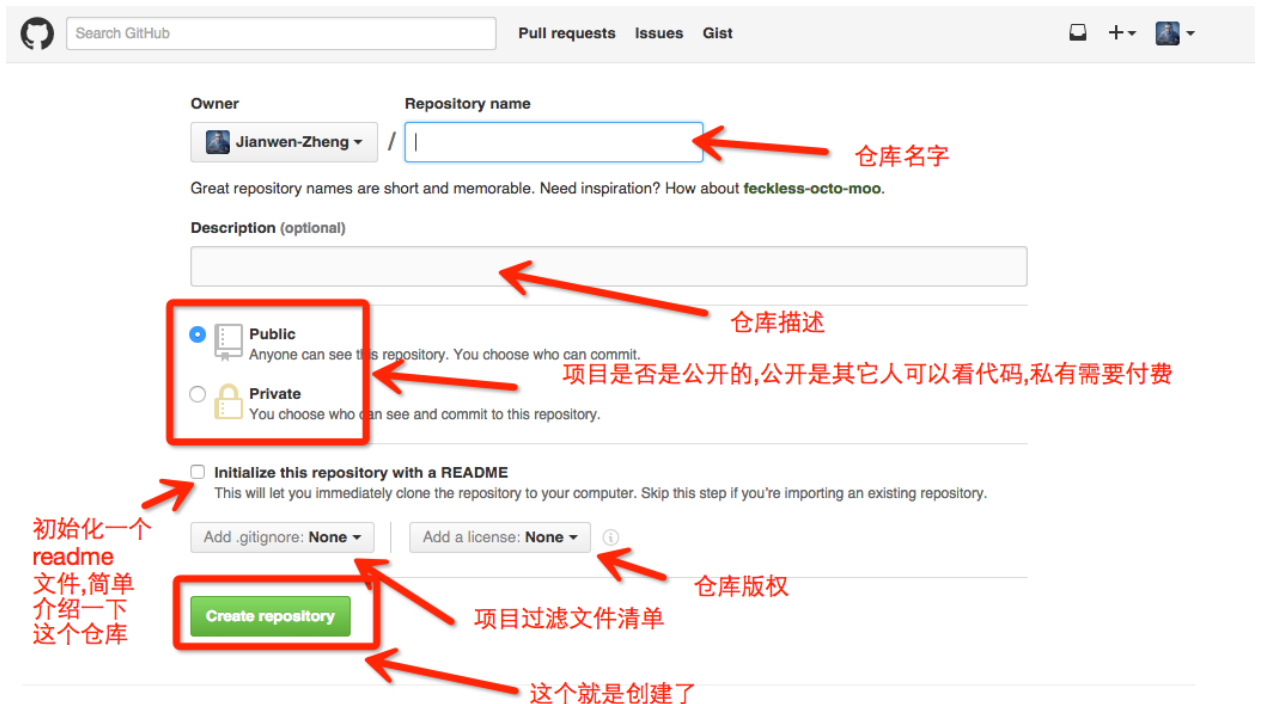
准备工作

- a). github 账号
- b). sourceTree

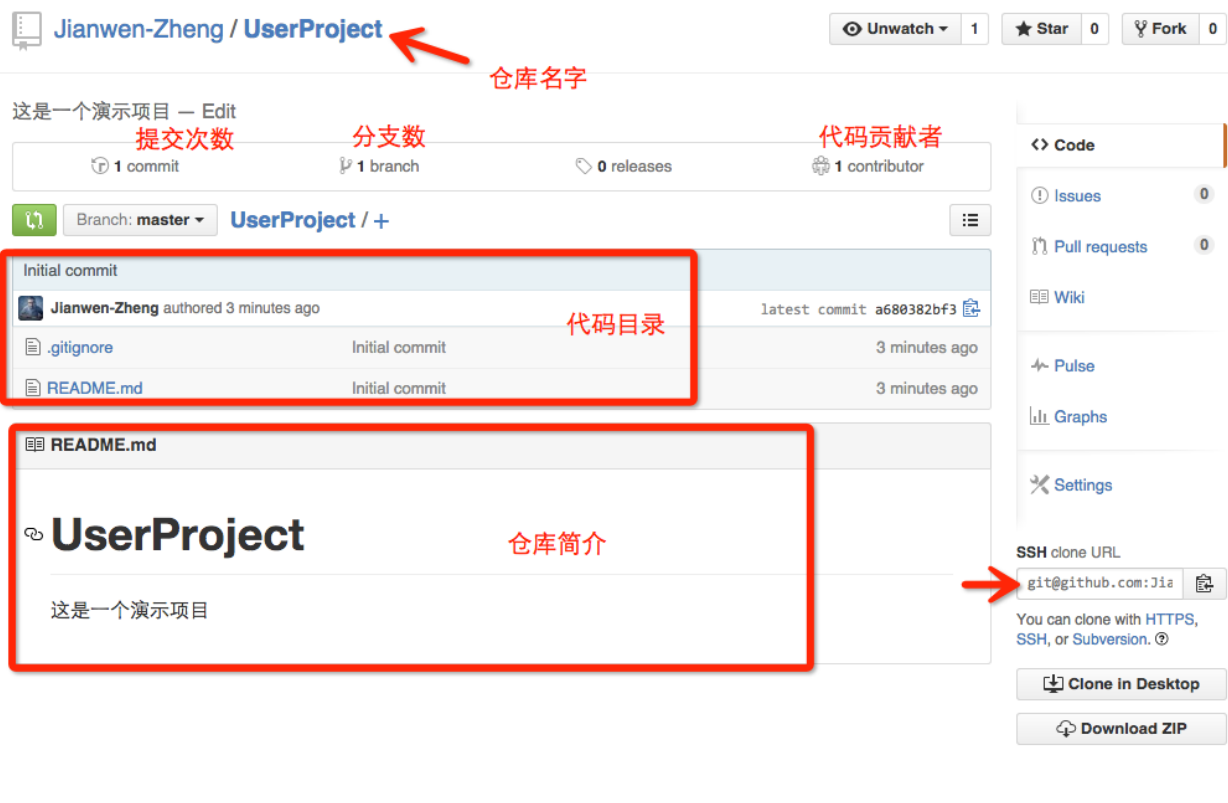
开始



1. 创建仓库 (登陆github网站,进入首页->点击右上角加号->选择'new repository')



2. 填写仓库信息



其中 项目过滤文件是用了配置那些文件可以不上传到github服务器的.(eg:我们用cocoapods时只需要上传podfile就行了,不需要上传pods文件夹中的文件了,我们就在这个过滤文件中添加一个 'pod' 这样上传项目代码是就不会上传第三方代码了)

3. 仓库界面

其中 **ssh clone url** 是 本仓库的克隆地址,以后可以通过这个地址把项目**clone**到本地

好了 我们在github 上的操作就到这儿了 -_- 是不是**简单**到不可思议

以后我们把我们的代码上传到github上我们刚创建的这个**远程仓库**是不是就行了!!!

但是!!!我们该如何来操作这个github的仓库呢?github上现在也没有我们的代码啊?github 上的仓库现在和我们的项目也没有半毛钱的关系啊?

不要急!!! 问题一个一个来解决

如何来操作这个远程仓库

既然要操作这个仓库,我们肯定要有这个仓库对不对?没有这个仓库,谈何出操作她呢?

想.....

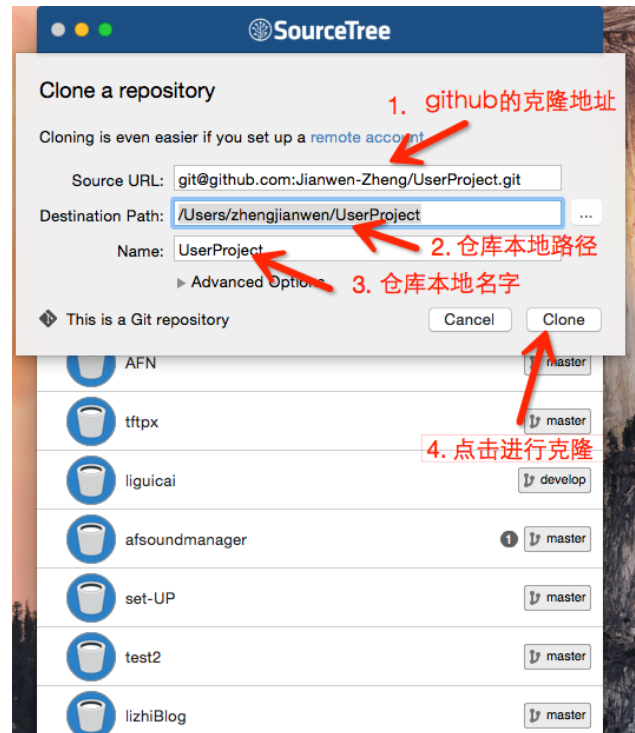
怎么能拿到这个仓库呢？ 下载？ 对啊.下载!!!但是在 git 中我们叫**克隆**

这时就用到了我们的git 工具了 ——> **SourceTree** (注意sourcetree只是一个git工具,他其实是一个由git命令组合而成的**可视化工具**)

在登陆后可能没有下图中的的那么的仓库,因为你没有克隆过,克隆过后就会有

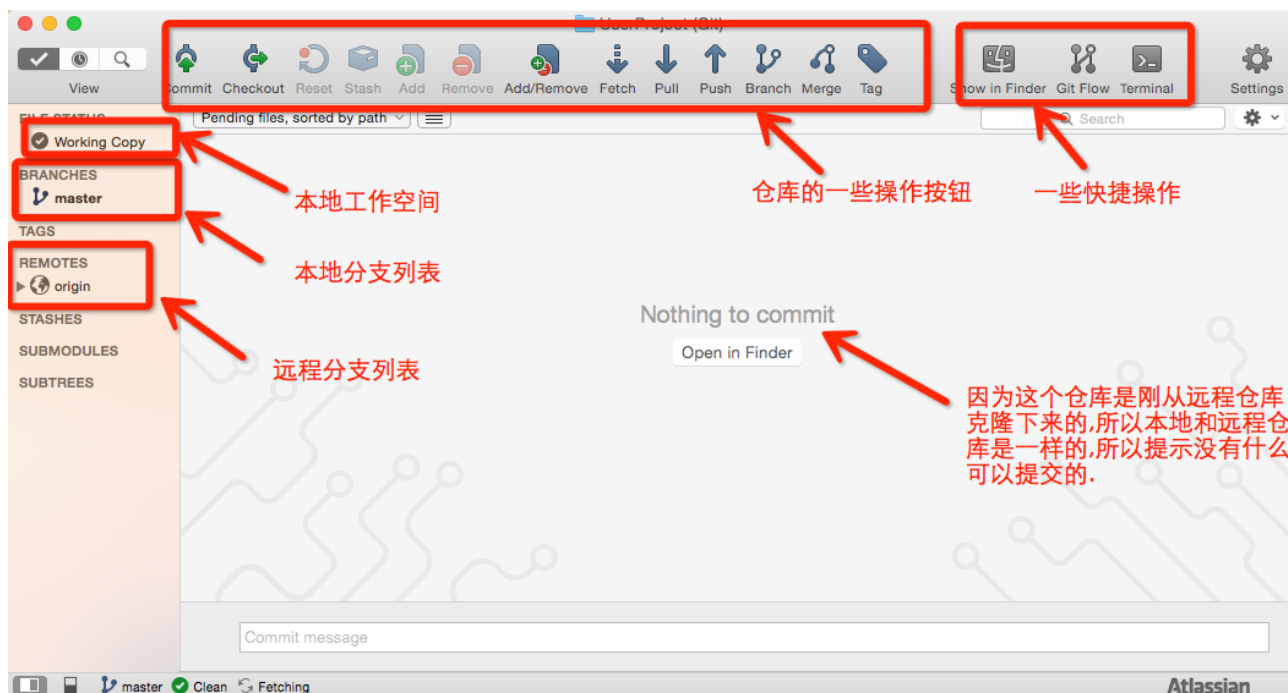
如何克隆呢？

在登陆你的github账号后可以点击 ‘+ New Repository’ 然后选择 ‘Clone from URL’



点击clone 后会出现如下界面

在点击 **clone** 后,仓库列表就会多一条仓库信息 ,**双击** 这条记录 就会进入这个仓库的**管理页面**



如果你也做到这儿了,说明你已经把远程的仓库克隆到本地了,但是这个仓库在哪儿呢?

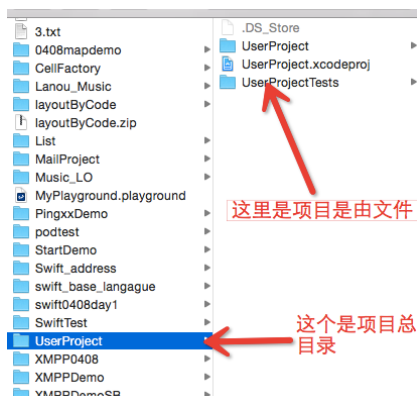
看上图右上角的 有个 '**show in finder**' 按钮么,点一下试试 !!!! 是不是 打开文件夹了,里面是不是有两个文件 **.gitignore** (注意这个是我们之前说的过滤文件,这是一个隐藏文件,如果你没有这个文件,自己百度一下怎么查看隐藏文件吧!) 和 **README.md**

ok! 这时,其实我们就可以写代码了,我们需要新建一个项目然后拖到这个文件夹中!

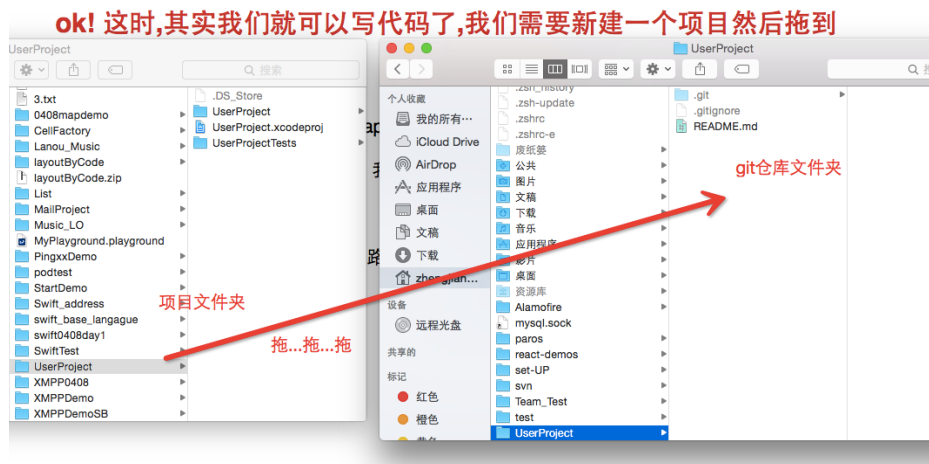
我们新建一个项目叫UserProject的ios application的项目(这个过程就不截图了啊...)

然后我们把这个新建的项目文件夹拖到 我们的仓库文件中

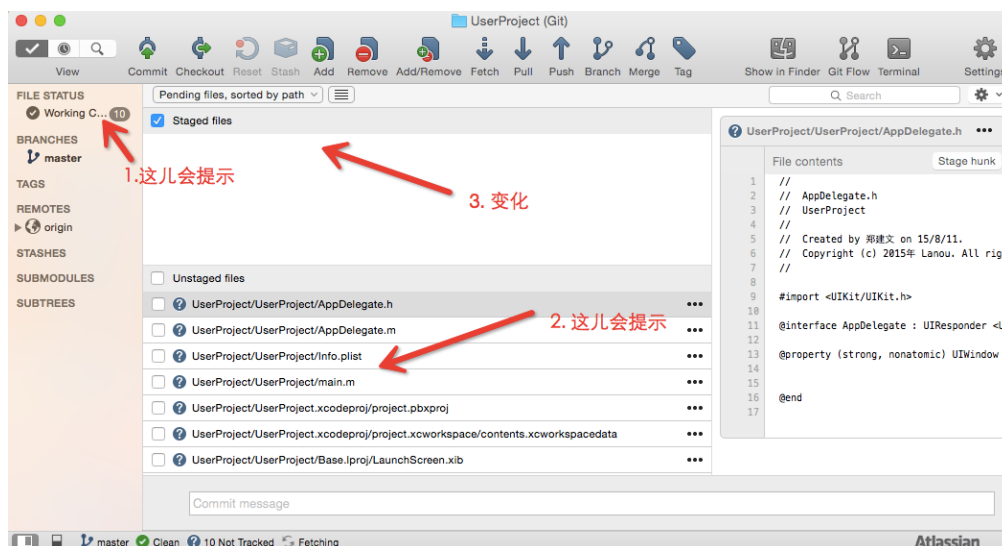
我们需要把项目文件夹拖到我们的仓库路径下



拖!



拖完之后!我们回到SourceTree 看一下 ,git仓库管理界面有什么不一样

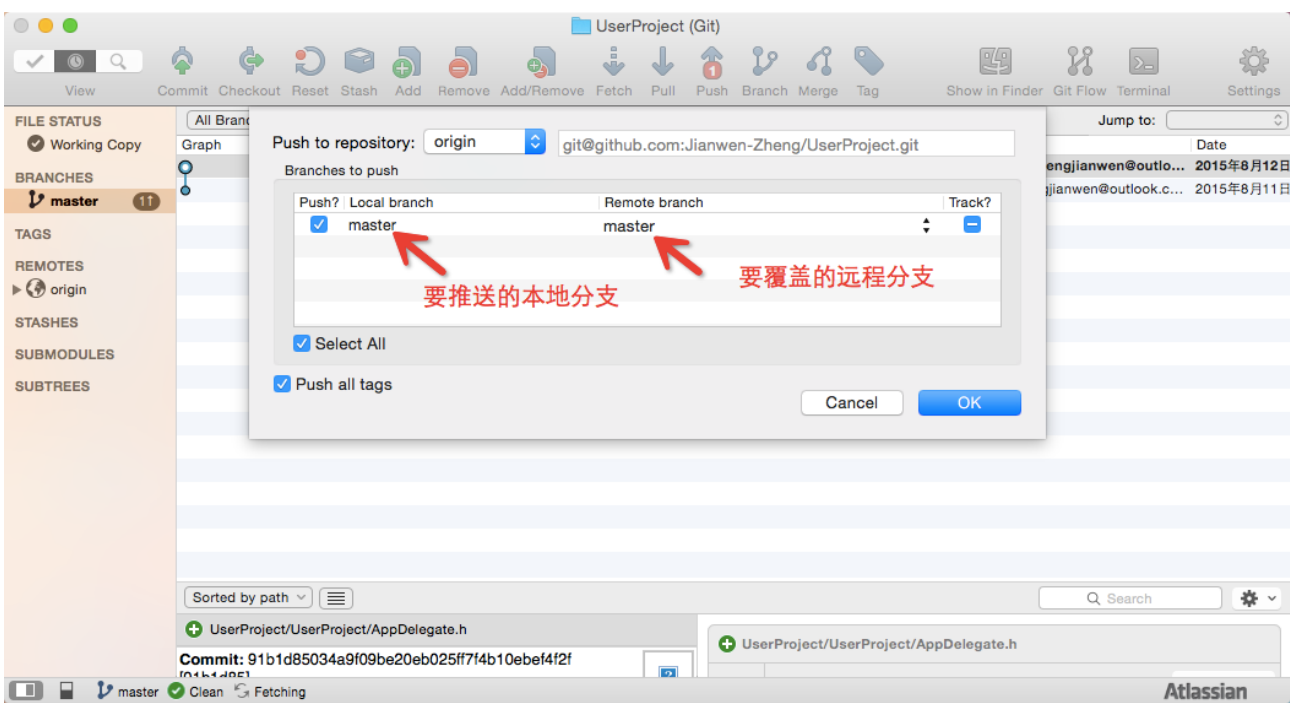
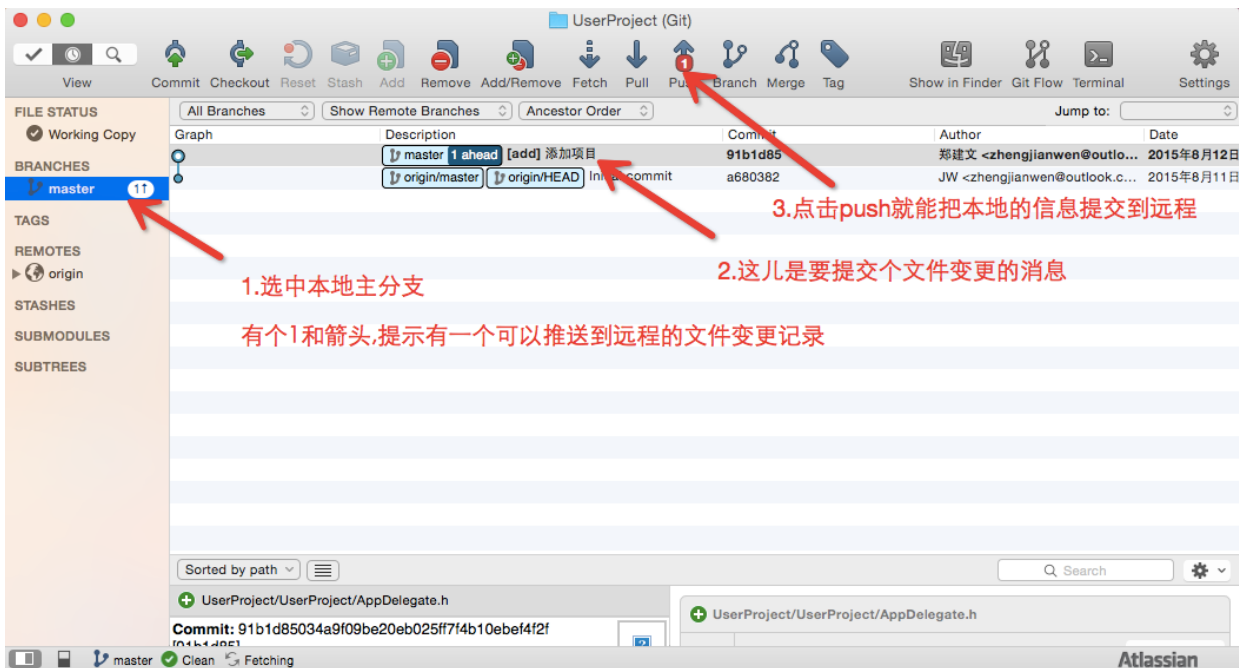
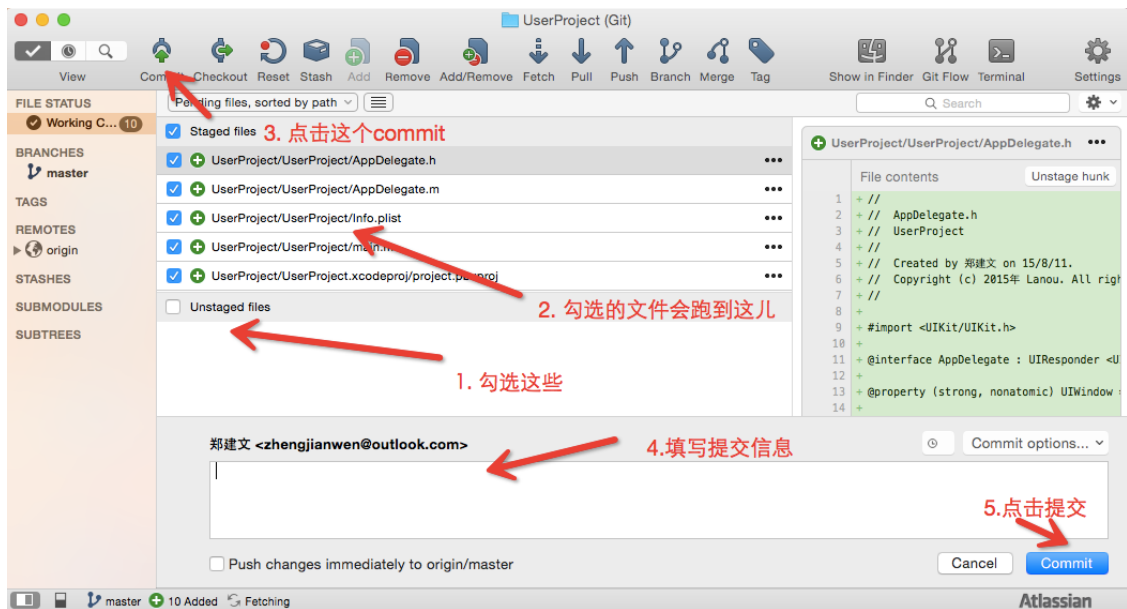


其中 2 是未暂存区 如果你添加或者修改了一个文件,这个文件暂时不会被git系统管理,这些文件就放在未暂存区.在这儿勾选一个文件后,就会将文件移到暂存区了.

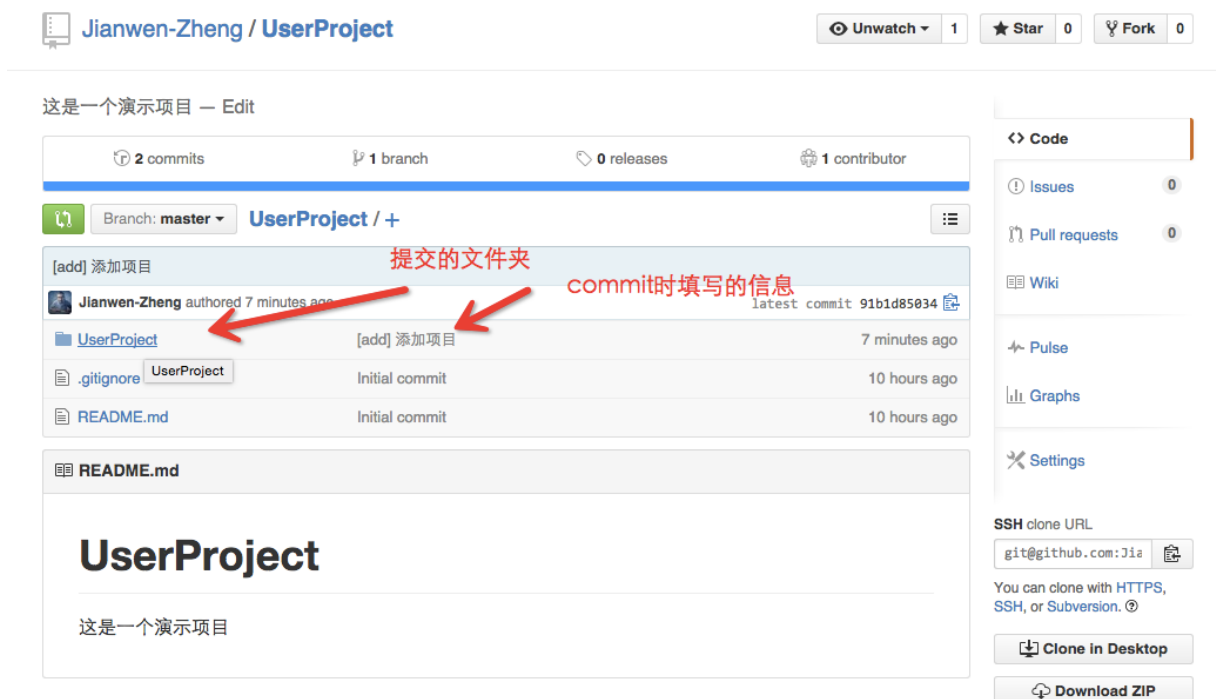
其中 3 是暂存区.如果一个文件被git系统管理,但是尚未被系统登记在案就会处在暂存区.未暂存区的文件想要被git记录就必须先进入暂存区

在这个界面我们要做的是把

1. 要用到git管理的文件,从未暂存区 选到 暂存区(勾选就会自动变化)
2. 点击commit(要填写提交信息),将暂存区的文件保存到git仓库中,这样文件的修改记录才会被保存到本地git仓库中
3. 将本地提交的文件变更信息提交到远程服务器中的仓库(push),点击



到这儿为止.我们已经将本地的仓库文件修改提交到我们在远程的仓库.我们可以去github上的项目仓库中看一眼是否多了一些文件



Jianwen-Zheng / UserProject

Unwatch 1 Star 0 Fork 0

这是一个演示项目 — Edit

2 commits 1 branch 0 releases 1 contributor

Branch: master UserProject / +

[add] 添加项目 提交的文件夹

Jianwen-Zheng authored 7 minutes ago latest commit 91b1d85034 commit时填写的信息

File	Commit	Time
UserProject	[add] 添加项目	7 minutes ago
.gitignore	Initial commit	10 hours ago
README.md	Initial commit	10 hours ago

README.md

UserProject

这是一个演示项目

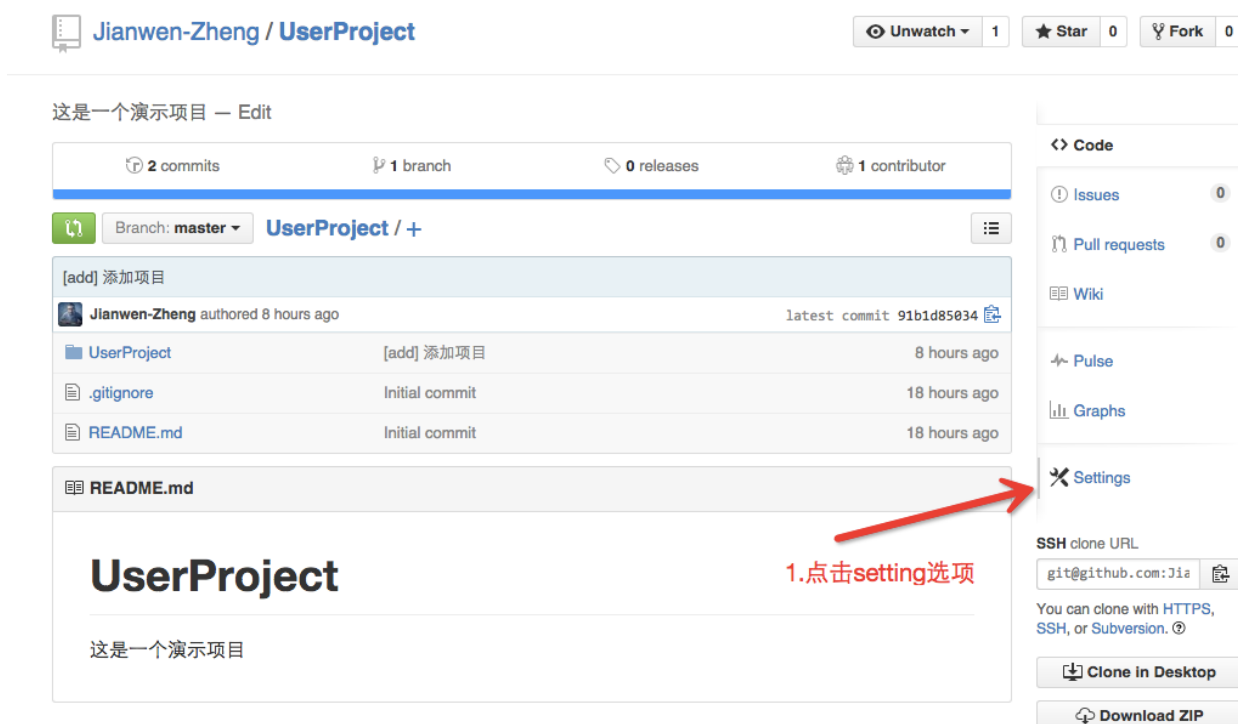
SSH clone URL: git@github.com:Jia

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop Download ZIP

ok! 简单提交完成了,接下来才是重头戏,如何用git进行团队开发?

现在在github中的这个UserProject这个库,其他的人是没有修改和提交的权利的.我们需要向这个库中添加项目成员.这样添加过的项目成员就有了这个库的修改的权限了.



Jianwen-Zheng / UserProject

Unwatch 1 Star 0 Fork 0

这是一个演示项目 — Edit

2 commits 1 branch 0 releases 1 contributor

Branch: master UserProject / +

[add] 添加项目

Jianwen-Zheng authored 8 hours ago latest commit 91b1d85034

File	Commit	Time
UserProject	[add] 添加项目	8 hours ago
.gitignore	Initial commit	18 hours ago
README.md	Initial commit	18 hours ago

README.md

UserProject

这是一个演示项目

Settings

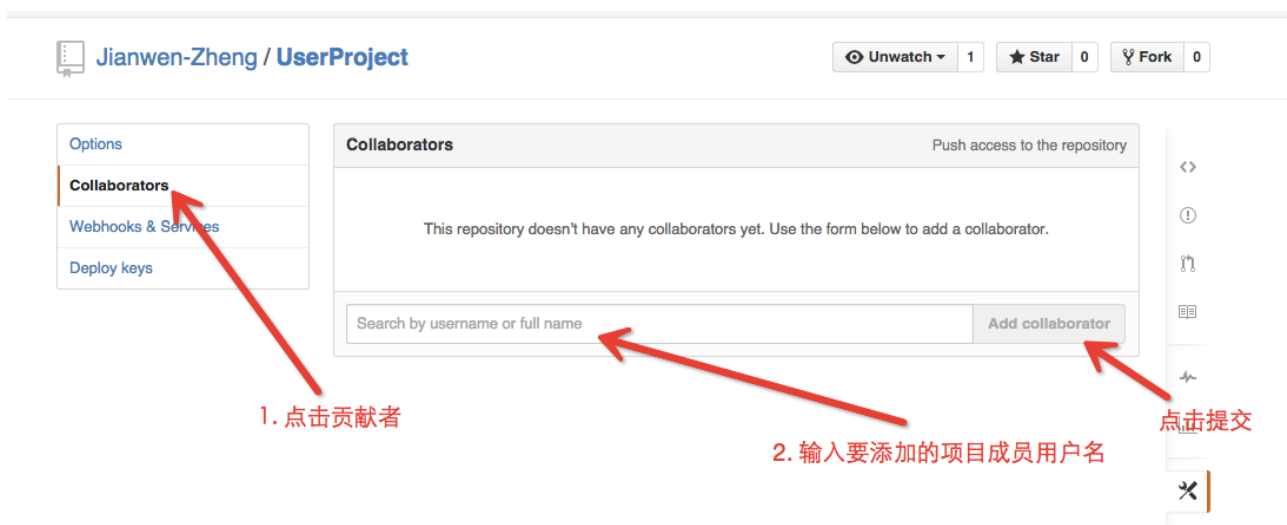
SSH clone URL: git@github.com:Jia

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop Download ZIP

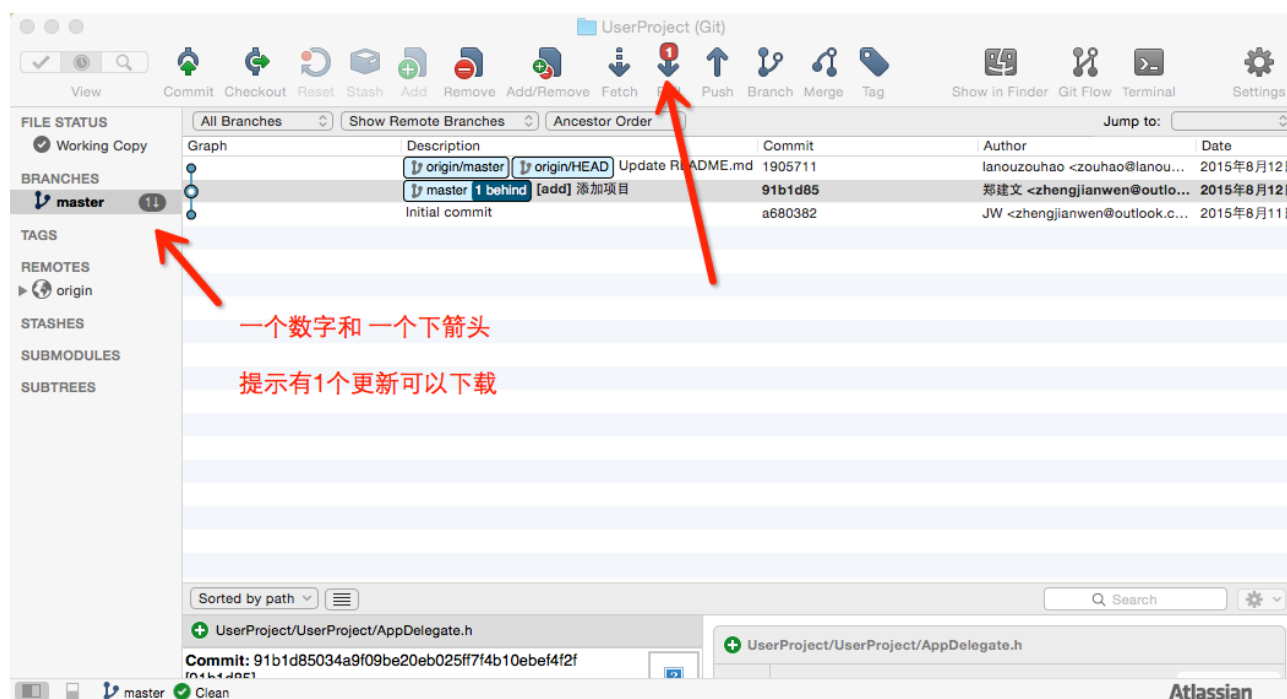
1.点击setting选项

然后进入这个页面

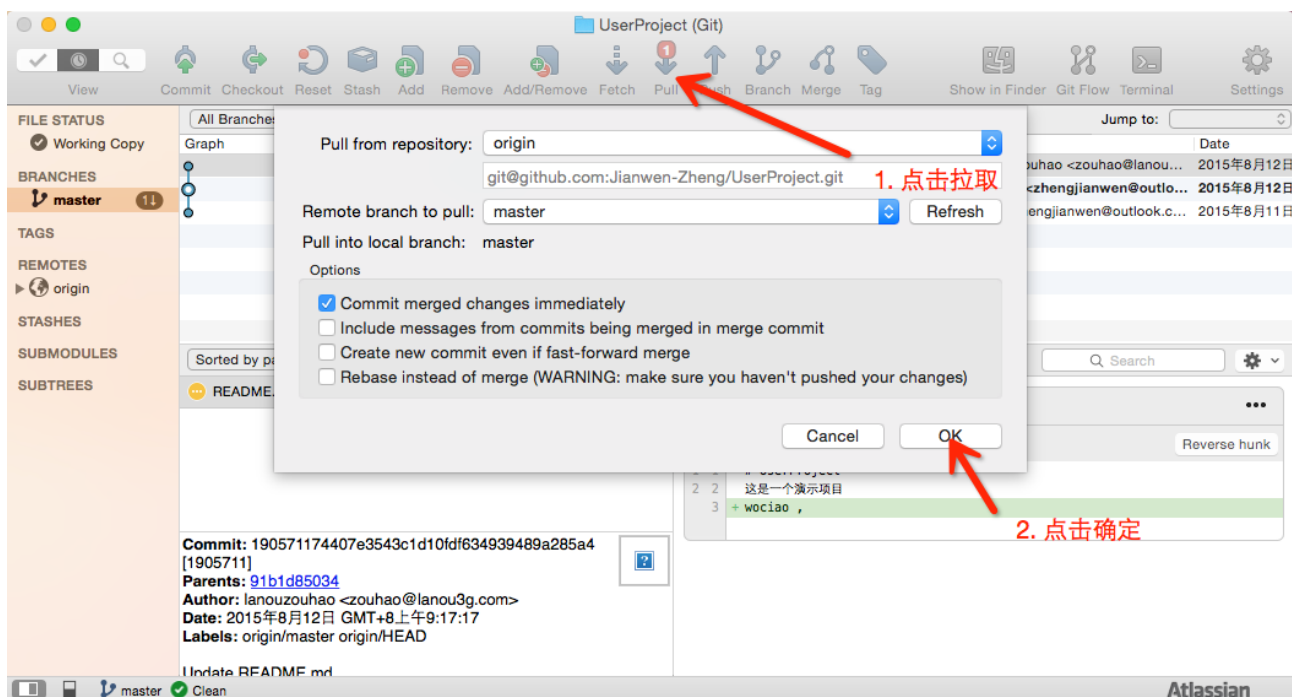


然后添加过的成员就有了这个仓库的权限了(如果你是被添加的那个用户,你可以修改一下readme了)

当你的同伙修改过项目文件,并push后,观察你的项目管理界面,是不是出现了下面的界面



出现这个界面,说明远程仓库有更新,可以将远程仓库的更新拉取下来保存到本地.



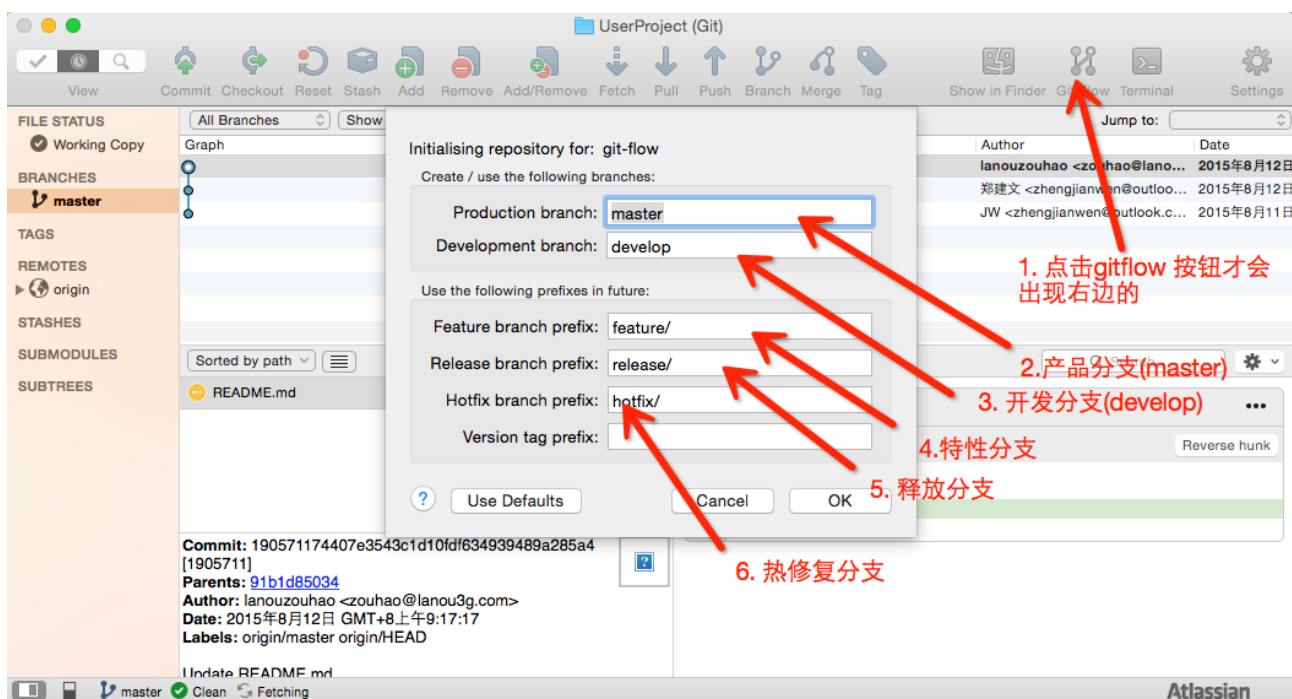
我们点击pull后会出现什么界面,点击ok 就可以把远程更新保存到本地了

分支

以上的操作都是在一个master分支上操作的,也就是我们一直在操作master分支,这样管理起来非常的不方便的,代码也比较容易混乱

为了不污染我们的主分支,我们一般都拷贝一个master的副本(创建分支),然后修改副本的代码,这样我们一直操作的就不是master分支了. 而是master分支的副本了,也就是我们说的子分支,当我们在子分支上完成一个功能后将子分支上代码给覆盖到主分支上(分支合并).

接下来我们看看使用SourceTree 怎么去创建分支



解释

git flow 是一种比较方便的开发过程,

1. **develop** : 创建一个develop分支,在develop分支上进行开发

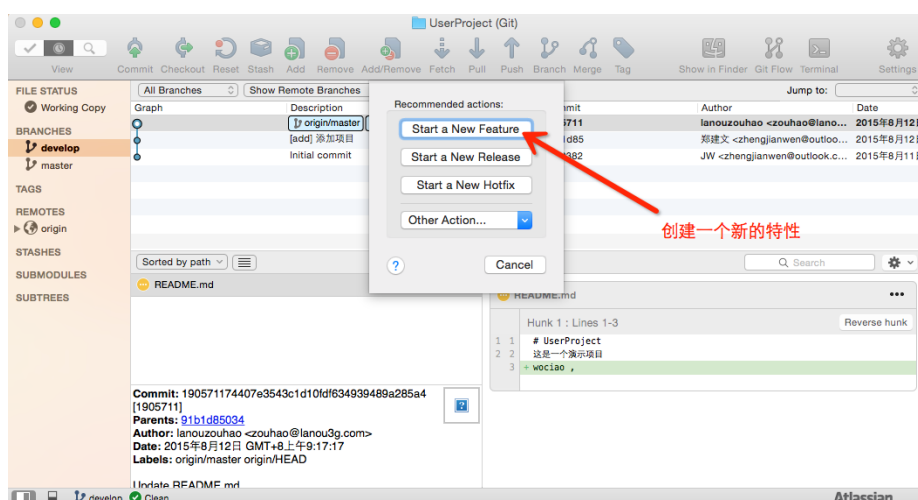
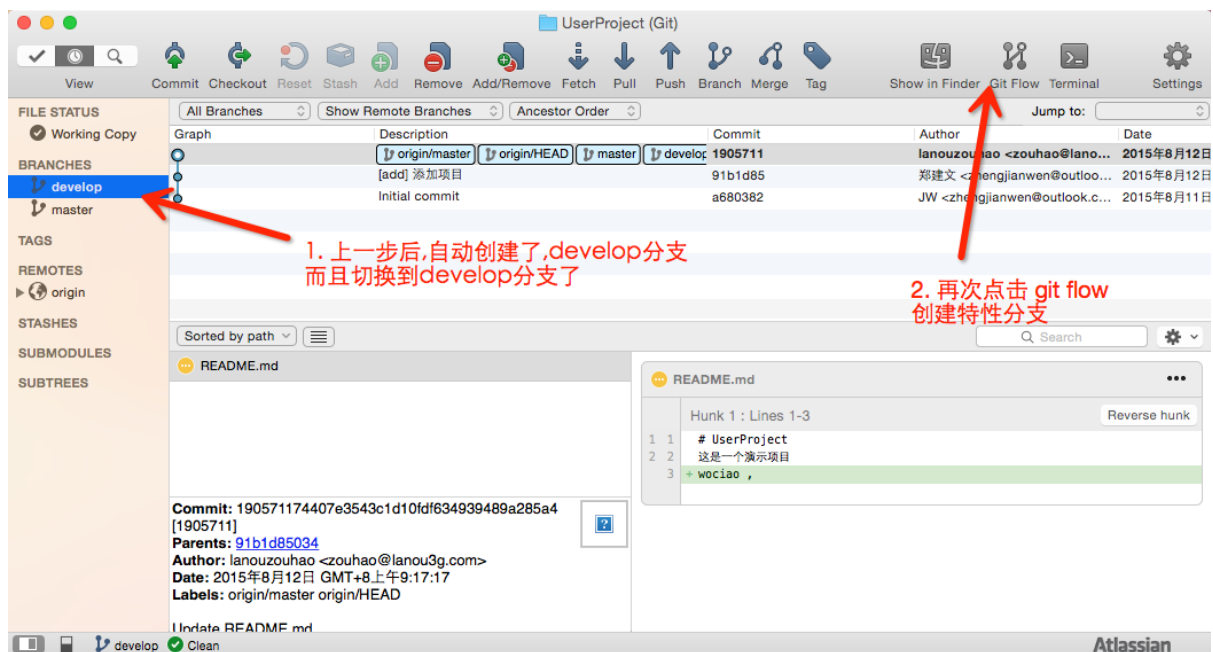
2. **feature** : 然后在develop上附属创建一些特性分支,比如我们可以为登陆功能创建一个login分支,为注册功能创建一个register分支,为用户管理创建一个user分支,这些功能分支都属于feature分支

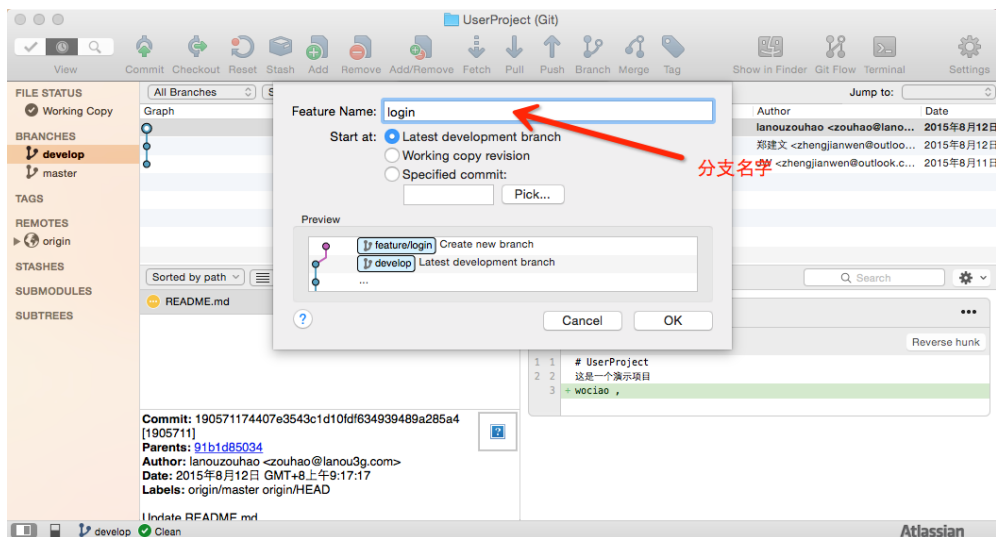
3. **release** : 我们每次产品发布时都会将代码更新到master上进行打包上传,然后更新打包代码到release分支上,这样就可以在以后只查看释放分支来查看每次上线时产品的代码了.

4. **hotfix** : 在代码更新到master分支后,突然发现代码中有bug,这是我们直接创建一个热修复分支,改完这个bug后立即更新到master分支了.

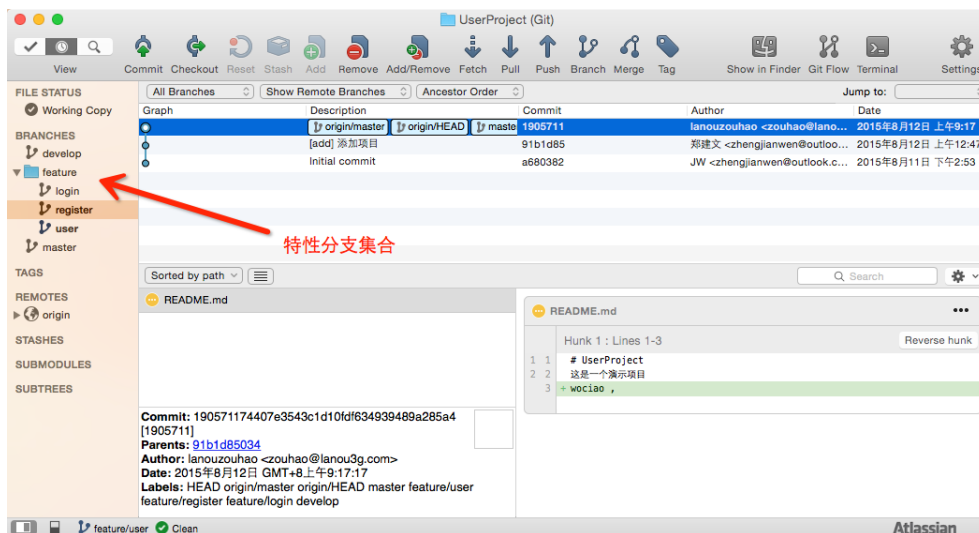
我们点击ok 后出现下面的界面

这时只有开发者分支,但是没有特性分支,再次点击git flow 创建特性分支





这是可以多创建几个分支了 (login,register,user)

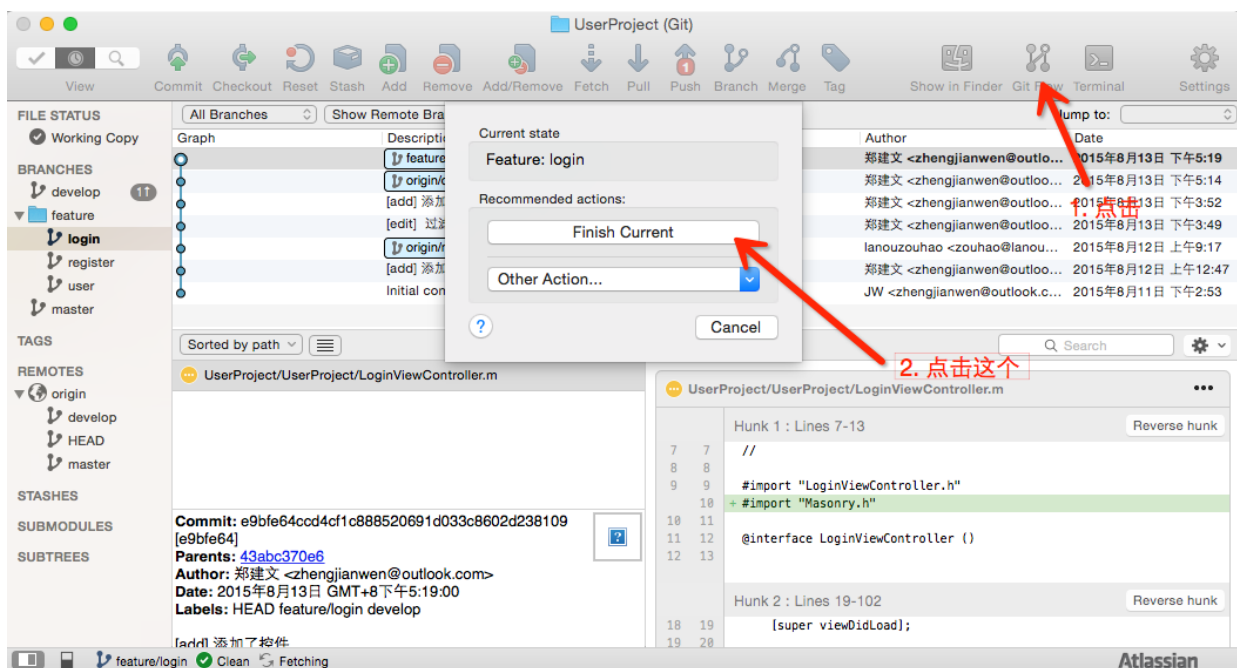


这时,如果我们想切换到那个分支只需要双击那个分支名就可以了.我们先切换到login 分支来做登陆界面

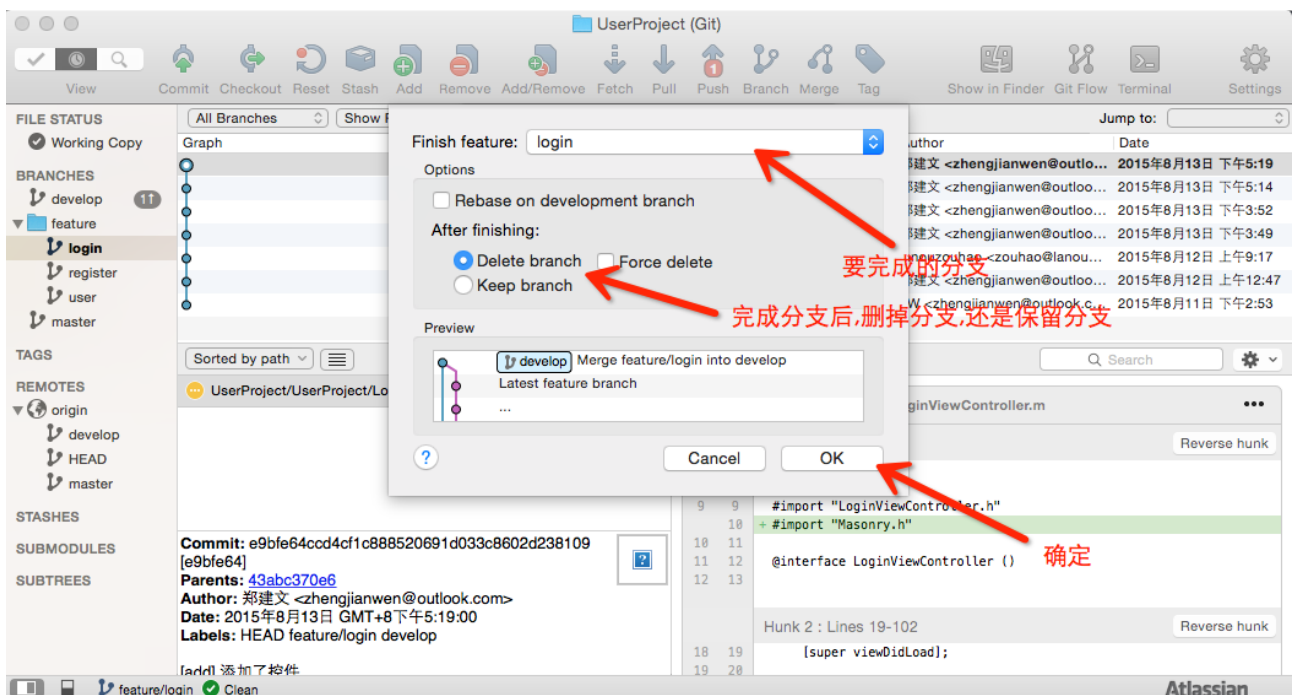


ok,等我们写完登陆功能后 ,需要将我们做的这个登陆功能代码合并到开发者分支中.

login分支是一个功能分支,我们完成这个功能了,需要把这个功能合并到develop分支上.合并过程见下图.



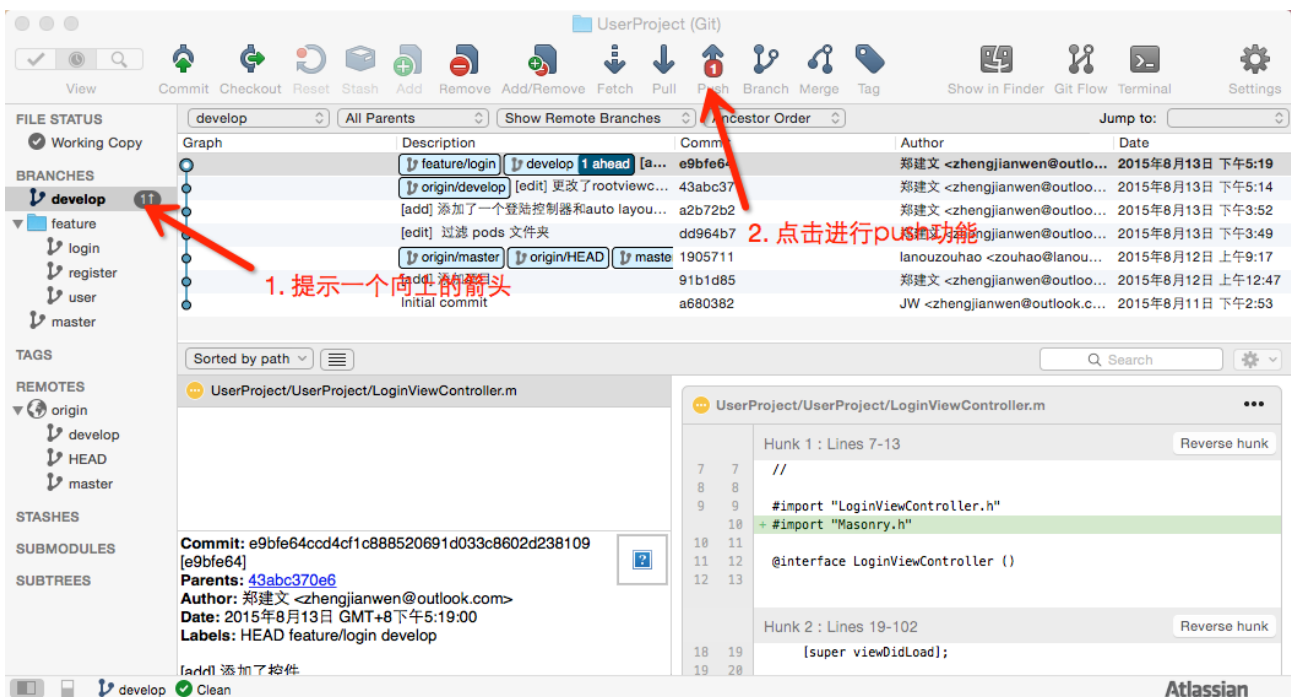
点击finish后,见下图



注意这个删除还是保留分支,如果删掉分支,以后就查不到这个分支.保留分支就行了,这样下次还可以查到这个分支.

在完成这个分支后develop分支上就又有login分支的代码了,不同的分支交给不同的人去完成,比如:张三负责login分支,李四负责register分支,王五负责user分支,这些人切换到各种的分支,然后开发,各自的功能开发完了,将各自的代码提交到develop分支,这样等我们项目开发的一定阶段了,比如:一版上线,演示版本,我们就将develop分支合并到master分支,然后由特定的人去master分支上进行打包上传就行了。

我们现在所有的操作都是在本地仓库中分支进行操作的,第一次使用这个远程仓库的话,远程仓库中是只有一个master分支的,其他的人使用你的远程仓库是看不到develop分支的,所以我们需要将develop分支提交到github,这时就是到push这个核心的功能了。



当我们将更新提交到本地后,系统会在develop分支上提示有几个可以提交到远程的分支,这时我们点击一下有数字下标的 push 按钮就可以将远程的仓库个本地的同步了。

下次我们再在这个分支上更改代码,只需要点击push 代码就可以提交到远程分支了

当我们同步后,你的其他的同事会在sourcecetree中的远程分支中看到新的分支了.右击本地没有的分支,然后进行checkout 分支,这时本地就有新的分支。

ok ! git 的基本操作就到这儿了.

其中有几个注意的地方!

4个分区!!!

1. 每次我们修改文件后,git系统就会把这个文件放到未暂存区(workspace),这个区的文件是会被git记录的.
2. 把未暂存区的文件保存到(add)暂存区,这个过程就像是一群要看病的人,进入医院后,只有排队领号后,才可以被医院收治.保存到暂存区才能算是进入了git系统,会被管理了.
3. 进入暂存区后,还需要一部提交(commit)到本地的系统中,提交就想把病人信息归档到档案室,以后可以通过提交信息查到这个病人的信息.
4. (远程仓库)前面3步都是在本地操作的,最后一部就是将本地仓库和远程仓库同步,同步后其他同事才能查看你的代码.

分支图

