Github使用指南

一：Github注册与仓库创建

二：Github客户端使用

三：Github分支创建与合并

四：Github命令使用

五：开源项目的开发(Fork)

一：Github注册与仓库创建

（1）Github介绍 BitBucket

Git是一个分布式的版本控制系统，最初由Linus Torvalds编写，用作Linux内核代码的管理。在推出后，Git在其它项目中也取得了很大成功，尤其是在Ruby社区中。目前，包括Rubinius、Merb和Bitcoin在内的很多知名项目都使用了Git。Git同样可以被诸如Capistrano和Vlad the Deployer这样的部署工具所使用。Github可以脱管各种Git库。

（2）注册与登录

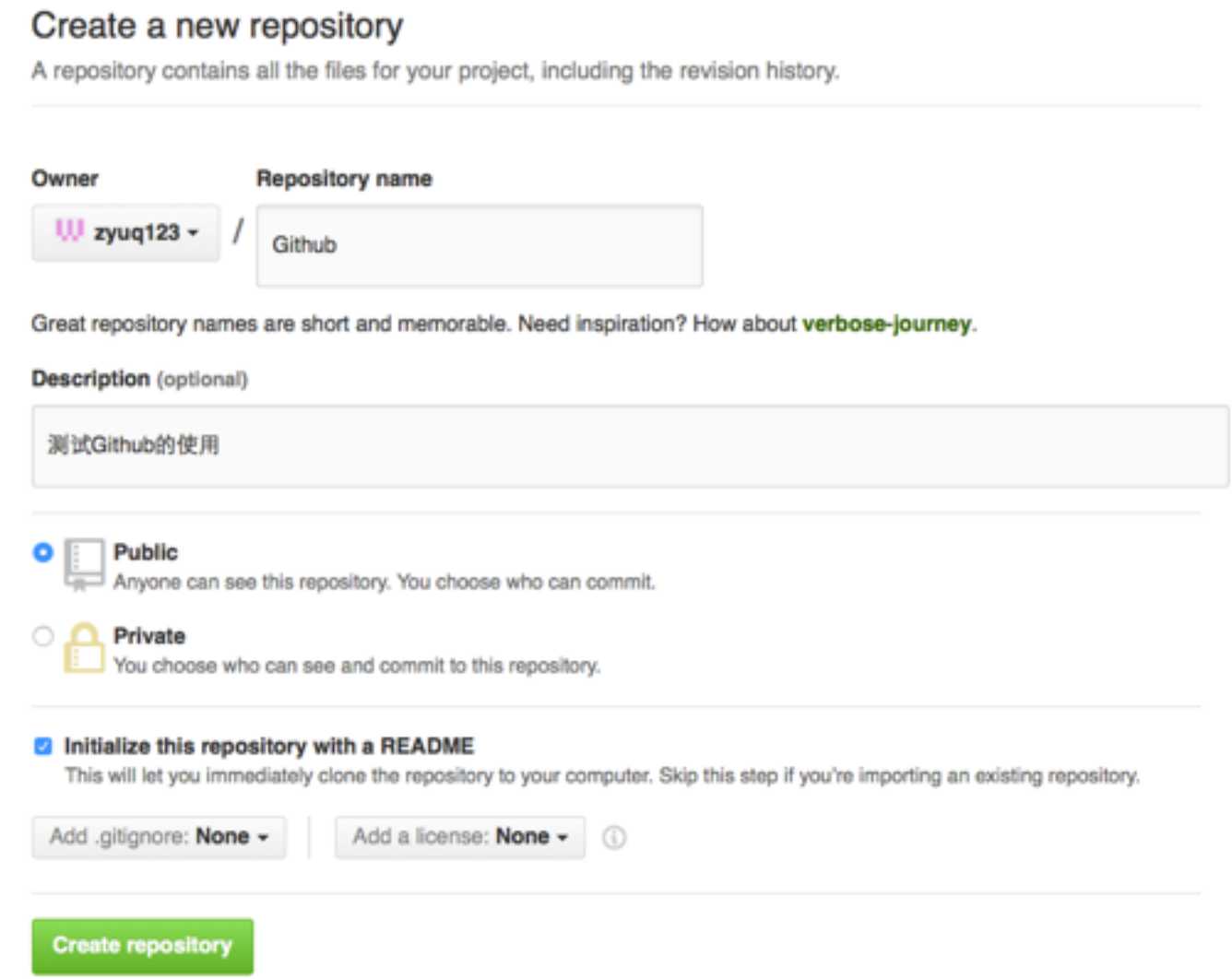
首先输入网址 https://github.com,输入用户名密码以及邮箱，选择免费的开发计划，同时进行邮箱的验证。这样，我们的Github账号就创建成功了，以后我们就可以使用该账号进行代码的管理。注册成功之后，我们要登录该账号。

（3）创建代码仓库(Repository)

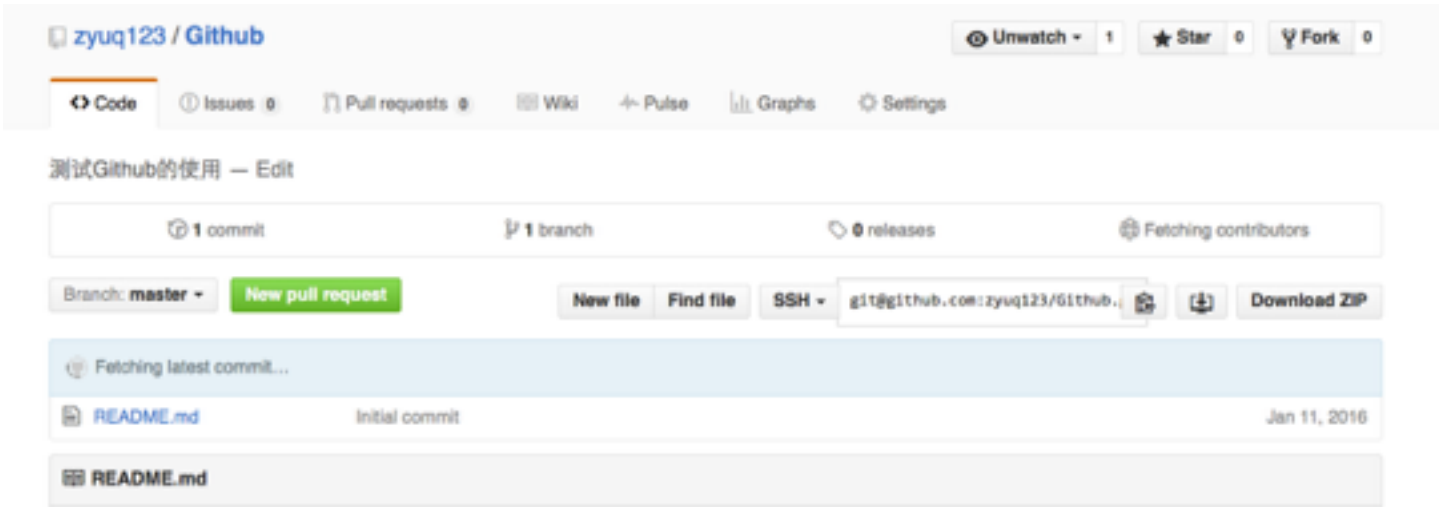
代码仓库是我们进行工作以及管理代码的地方。只有创建了代码仓库，我们才能进行远程代码的提交。我们按照如下步骤创建代码仓库。

1:点击右上角的Create New Repository进行代码仓库的创建。输入标题和详细的描述，选择Public的项目，也即别人可以搜索到我们的项目。选择Private的话，就需要支付一些金钱。

如下图所示：



2:这样我们的代码仓库就创建成功了，我们点击代码仓库，可以看到内容如下图所示：



3:点击README.md，进行编辑，输入一些有关仓库的描述信息。
4:至此，我们服务器的配置就暂时完成了。下面我们开始讲解客户端的操作。

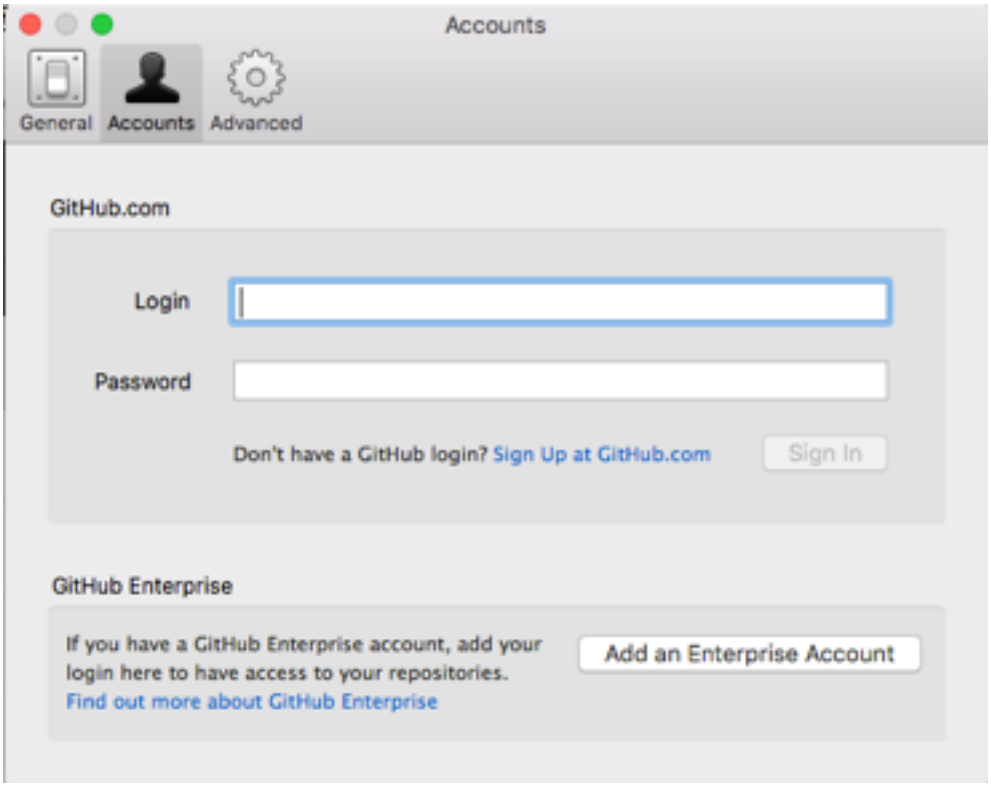
二：Github客户端使用

当我们使用git库的时候，一般情况下我们都要掌握命令来进行操作。但是，很多复杂的操作，通过命令进行操作的话，比较复杂。好在Github针对MAC开发了客户端，通过客户端我们可以更加方便的进行代码的管理。

下面我们就演示一下如何在客户端添加代码，并且将修改提交到远程代码仓库。

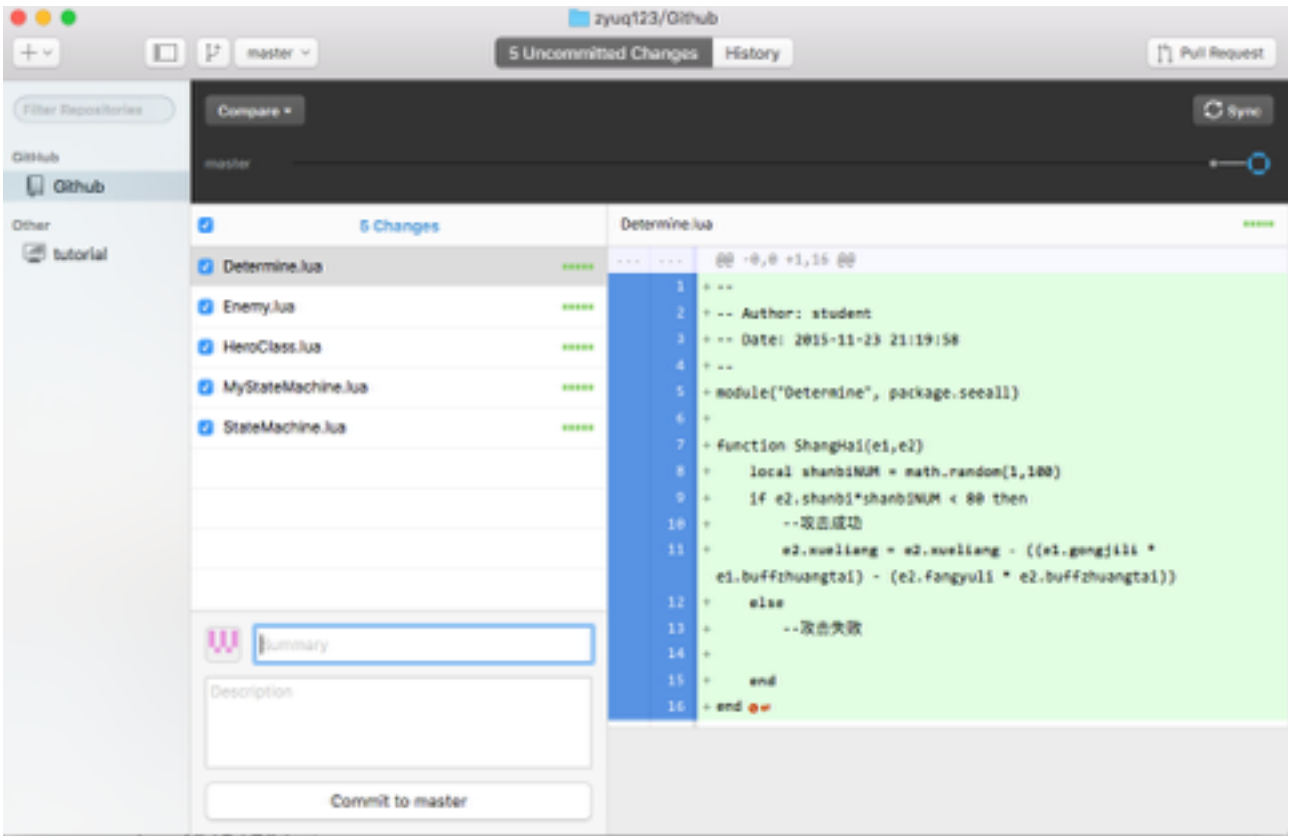
1:安装软件Github Desktop,安装完成之后打开。

2:第一次运行会出现如下所示的界面，要求我们输入用户名和密码。我们这里将注册Github时候的用户名和密码输入即可。同时在Advanced选项下，我们可以输入GitConfig来进行用户的标识。当我们提交代码的时候，此时你的配置将会显示在提交里面。



3:点击客户端左上角的“+”，选择第三个选项Clone，此时我们在网站上创建的项目就会显示在这里。我们点击我们刚才创建的Github，此时会弹出一个保存的选项，我们将其保存在桌面即可。

4:我们进入到克隆之后的文件夹，添加几个文件和代码。此时我们再次进入Github客户端，会发现有未提交的修改。如下图所示的界面。我们选中每一个文件，并且在下面的Commit和Summay输入一些必要的信息，点击最下面的Commit to master。此时我们的修改已经提交到了我们的本地代码仓库。



5:提交到远程仓库

刚才我们已经修改了本地的文件，并且提交到本地代码仓库。下面我们就将修改提交到远程服务器。

点击右上角的Sync，就可以将本地的修改同步到远程的master分支上。此时我们登录远程服务器，刷新界面，就可以看到我们刚才提交的文件。

同理，如果在服务器修改了一些数据，我们需要将其同步到本地。那么，我们也需要点击Sync即可。

三：Github分支创建与合并

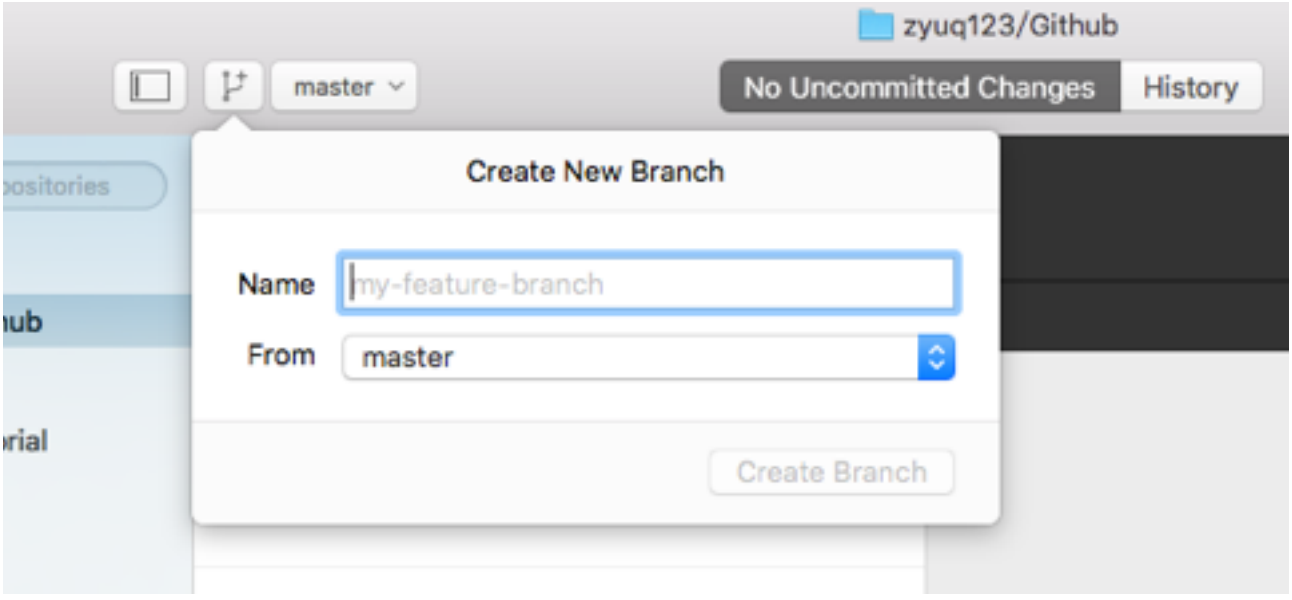
（1）分支介绍

建立一个开发之外的分支，当改变一个分支中的文件时，这些更改不会出现在开发主干和其他分支。

一般我们在项目开发出稳定版本之后，我们需要对其中的一些Bug进行修改。修复的时候，我们往往会新建一个分支，所有的修改全部在分支上进行。当修改完成之后，我们需要合并分支，将修改合并到主分支master之上。

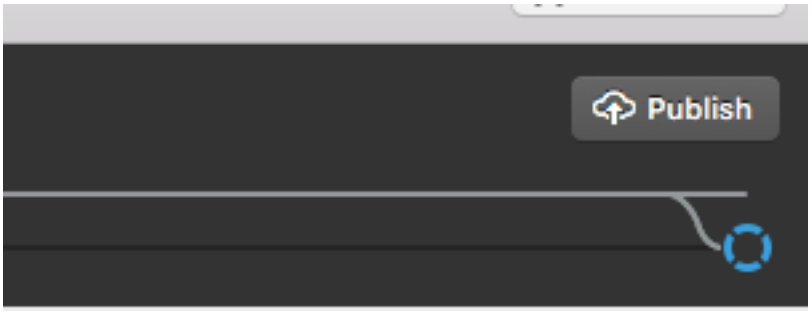
（2）分支的具体使用

1：点击左上角的创建分支选项，点击之后出现如下界面。



我们输入分支的名字，以及从哪一个分支扩展出来的。我们在这里输入Test，From master不修改。完成之后点击Create Branch即可。

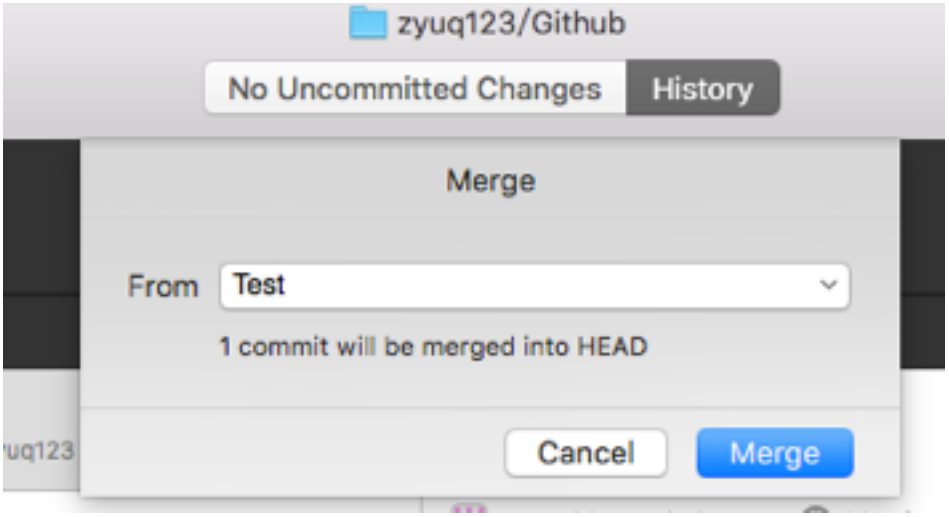
2：此时我们虽然创建的分支，但是我们服务器上不存在这个分支，所以我们需要发布分支。发布之后，服务器也存在这个分支了。我们点击下图中的Publish进行发布。此时查看服务器，可以看到我们发布的分支。



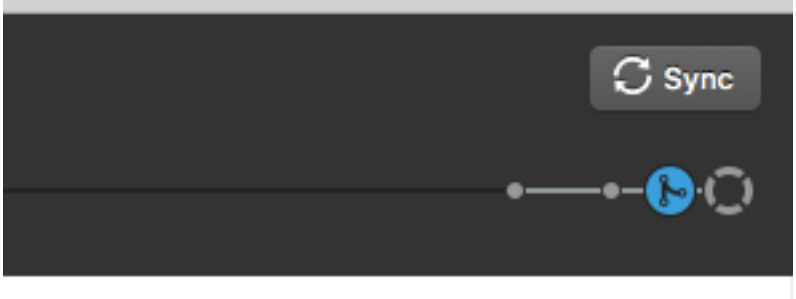
3:我们当前默认的分支就是Test,可以通过上方的下拉列表切换分支。我们此时修改Test分支下文件的内容,添加一行注释。此时回到Github的界面，我们会发现有未提交的修改。

按照和刚才同样的步骤我们进行提交即可。Commit完成之后，我们需要点击Sync，将修改内容同步到远程服务器的Test分支。此时查看服务器的Test分支，可以看到修改的情况。

4:假设我们现在分支的所有功能都实现了，我们需要合并分支。合并分支的时候，我们需要首先切换分支到主分支下。然后选择菜单 “Branch”->”Merge into master”进行合并。

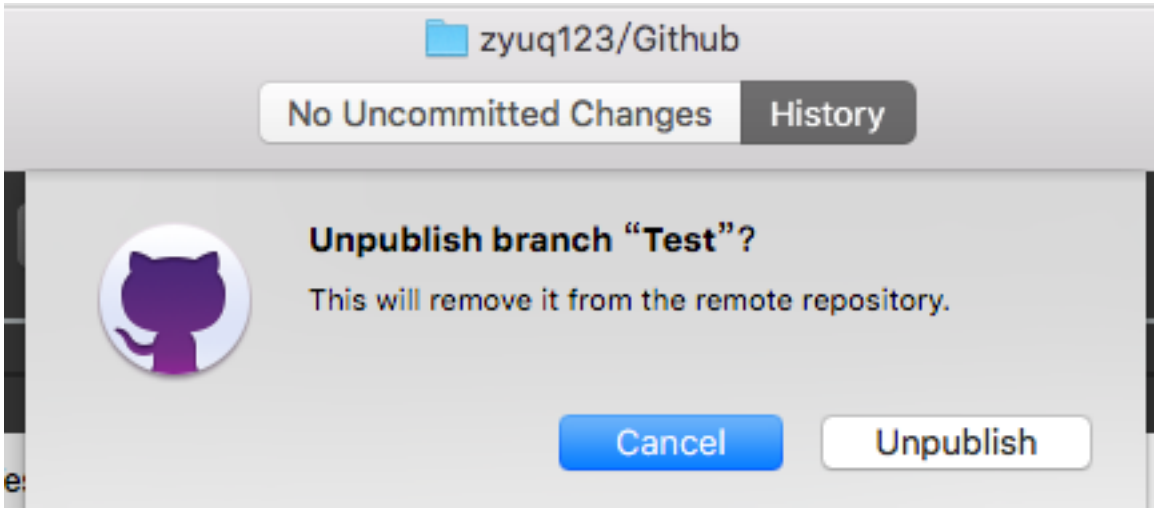


From选择你需要合并的分支，此时选择Test即可。点击Merge,就可以进行分支的合并。



此时主分支会出现一个符号，如上图所示，同时我们刚才的分支合并还没提交到服务器。我们点击上方的Sync进行同步即可。再次查看服务器，我们可以看到主分支已经合并了修改之后的内容。

5：删除分支，我们点击Branch->Delete Test就可以删除我们当前的分支。但是服务器上这个分支还是存在的，我们需要点击Unpublish，将此分支从服务器移除。如下图所示：



此时查看服务器，Test分支已经不存在了。然后我们将本地的分支直接删除(Branch->Delete)即可。

四：Github命令使用

我们上面讲解了基本的客户端使用。为了更好的提高我们对Git的熟悉程度。我们下面讲解一下使用命令对Git进行操作。

(1)生成SSH key

使用ssh key验证github的好处就是不用每次提交代码的时候都要输入用户名和密码，因为在一定程度上对效率有很大的影响。

在local打开terminal:

$cd ～/.ssh 检查是否已经存在ssh， 如果存在，先将已有的ssh备份，或者将新建的ssh生成到另外的目录下，如果不存在，通过默认的参数直接生成ssh。

生成过程如下:

$ssh-keygen -t rsa -C xxxxx@gmail.com（注册github时的email）

这个时候，在.ssh目录下有两个文件

id_rsa id_rsa.pub

其中id_rsa是私钥 id_rsa.pub是公钥。

然后，执行下面的命令，将生成的key添加

ssh-add id_rsa

然后将id_rsa.pub里面的内容复制下来，在github上的settings里面找到add keys，将其粘贴到key即可，title随便写就可以了。

这个时候可以在控制台上测试一下

ssh git@github.com

设置本地git个人信息:

```
$git config --global user.name "your real name"
$git config --global user.email "xxxxx@gmail.com"
```

至此，git和github的设置就完成了，下面就是如何将本地代码push到github上，以及如何从github上pull代码。

（2）本地添加代码仓库

在本地创建代码库：

```
$mkdir test
touch README.md //新建一个记录提交操作的文档
git init //初始化本地仓库
git add README.md //添加
git commit -m "first commit"//提交到ben地仓库，并写一些注释
git remote add origin git@github.com:youname/Test.git //连接远程仓库并建了一个名叫：origin的别名
 git push -u origin master //将本地仓库的东西提交到地址是origin的地址，master分支下

git push -u origin :A //删除远程服务器A分支
git push -u origin master:A //将本地master推送到远程A分支
git reset —hard HEAD
git reset —hard 0a12345
```

这样就可以将本地的代码push到github的repository中了

基本上就这么一个过程了，如果你的本地址仓库还有很多其它东西要提交，先要在本地仓库commit， 方法也是先git add 然后 git commit，这些都是commit到本地仓库，然后再push到选程。

下一次操作时，名叫origin的远程地址上次已经建好了，想看看可以用git remote -v 查看地址。

（3）获取远程代码仓库

push的时候，先pull一下，本地同步下，再push。

从github中pull代码：

```
$git pull git@github.com:xxxxx/ruby_koans.git
```

将github上的代码pull到local repository中

（4）Git常用命令

Git常用命令

（1）git help查看命令的帮助文档

（2）git log查看git的日志信息

（3）git init可以创建一个新的代码库，或者是初始化一个已经存在的代码库，例如我想在本地创建一个myrepo代码库，可以先使用mkdir创建目录，然后再执行git init,在终端执行以下命令

```
mkdir myrepo
cd myrepo
```

git init

使用mkdir创建一个目录（普通目录，并不是代码库），git init会在myrepo下生成一个隐藏的.git目录。

（4）git add .

该命令用来更新索引，记录下哪些文件有修改或者添加了哪些文件。该命令没有更新代码库，只有在提交的时候才将这些变化更新到代码库中。在终端执行以下命令

git add .

可以将当前目录以及子目录下的所有新添加和修改的文件添加到索引中，如果只想木一个文件添加到索引中，可以使用如下命令

git add filename或者git add *.txt

(5)git rm

该命令用来删除索引或者代码库中的文件，然后通过提交命令将变化更新到索引中。在终端执行以下命令git rm filename或者git rm *.txt

(6)git commit

该命令用来更新缓存中的索引，但未被保存到代码库中的内容。在终端执行以下命令

git commit -m ‘tony commit”

其中—m设定注释信息

（7）git status

该命令可以显示当前git的状态，包括哪些文件修改，添加和删除了，但是没有提交的信息

（5）Git分支管理

Git的分支在版本控制中占据着非常重要的地位，无外乎验证一些想法，版本发布，bug修改等目的。

（1）创建分支

```
git branch testing
```

创建了一个testing的分支，在终端输入git branch可以查看当前的分支，以及当前所处的分支，带*的表示当前分支

（2）切换分支

编辑工程的时候，需要在不同分支之间切换，在终端执行以下命令，切换分支

```
git checkout testing
```

我们在testing分支修改了代码之后，执行以下命令

```
git add .
git commit - m  ‘testing branch’
```

(3)合并分支

如果我们把testing分支合并到master分支，需要先切换回master分支，接着在终端里执行如下命令切换回master分支

```
git checkout master
git merge testing
```

(4)冲突解决，我们可以再次提交

```
git commit -a -m
```

相当于git add .和git commit -m的执行效果。

（5）删除分支

在分支合并完成并且不在使用的情况下，可以使用git branch -d 删除分支，若要删除testing分支，

在终端输入

```
git branch -d testing
```

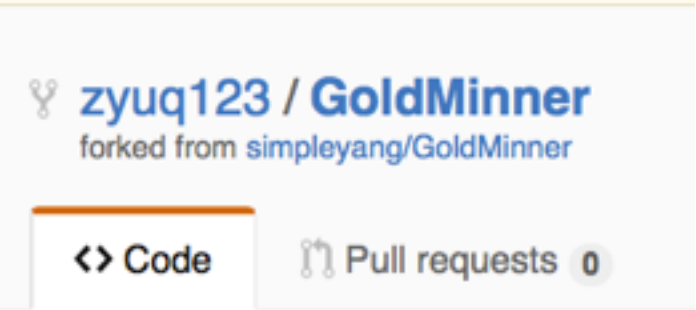
这就是Git常见的一些命令。希望大家能够多使用Git，进而达到熟练的程度。

五：开源项目的开发(Fork)

（1）Fork代码仓库

有的时候，我们需要使用别人开发的第三方类库。通过github进行学习是最快的成长手段。我们可以浏览别人的优秀代码。但只看不动手还是成长得很慢，因此为别人贡献代码才是明智之举。因此贡献代码，参与开源项目是有益无害的！

如果要从别人那里派生代码库，可以修改该代码库，然后提供回开发者。我们可以把它理解为代码库级别的分支。



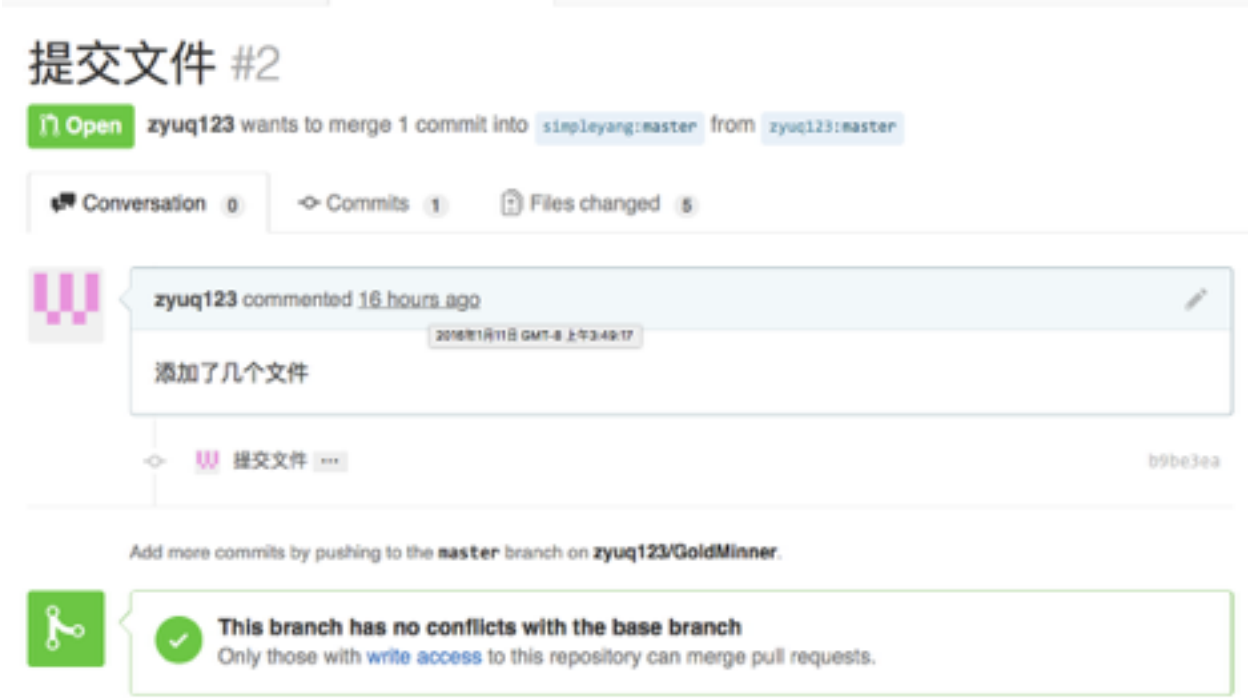
我们首先在Github中搜索代码仓库，在命令栏输入仓库名称回车，搜索到代码之后。点击进入代码之中，右上方出现Fork按钮，将拷贝一个备份到我们的项目里面。此时我们的代码仓库出现刚才我们Fork的项目。如下图所示：

下面我们将代码克隆到本地，进行修改，看如何进行代码的合并。

（2）修改代码

添加几个文件，并且提交到你的代码仓库。

点击客户端右上角的PullRequest，可以将当前的修改提交给项目的源仓库。此时我们进入网站，可以看到我们项目里面的Pull Request，状态变为Open，表示我们提交了一个合并请求。如下图所示：



此时，开源仓库的发布者登录他的账号，进入GoldMiner，就可以看到当前我们发送的合并请求。检查无误之后，进行代码的合并即可。我们再次进入我们的PullRequest,发现此时的状态变为Close，表示合并完成。