**Jbone项目协作开发流程**

1. 在github上fork项目https://github.com/417511458/jbone
2. 在自己的项目上修改并提交
3. pull request
4. 管理员审核并合并

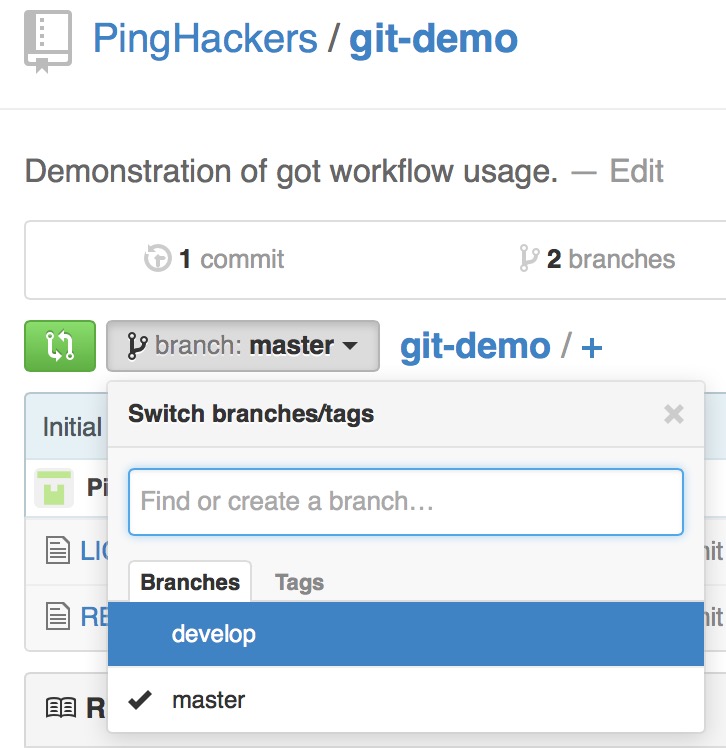
参考：https://www.cnblogs.com/wawahaha/p/5171187.html

**Git协作开发完整工作流**

啰嗦讲了这么多，概念永远是抽象的。对于新手来说，都喜欢一步一步的步骤傻瓜教程，接下来，我们就一步一步来操作上面所说的工作流程，大家感受一下：

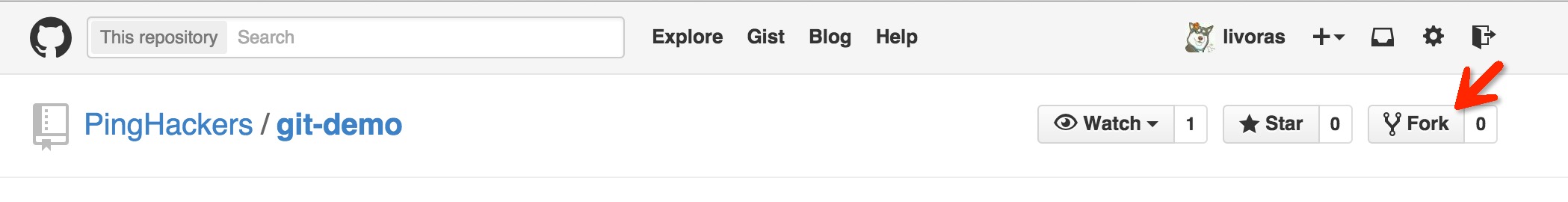
Step 1：源仓库的构建

这一步通常由项目发起人来操作，我们这里把管理员设为PingHackers，假设PingHackers已经为我们建立起了一个源仓库[PingHackers/git-demo](https://github.com/PingHackers/git-demo" \t "_blank)，并且已经初始化了两个永久性分支master和develop，如图：

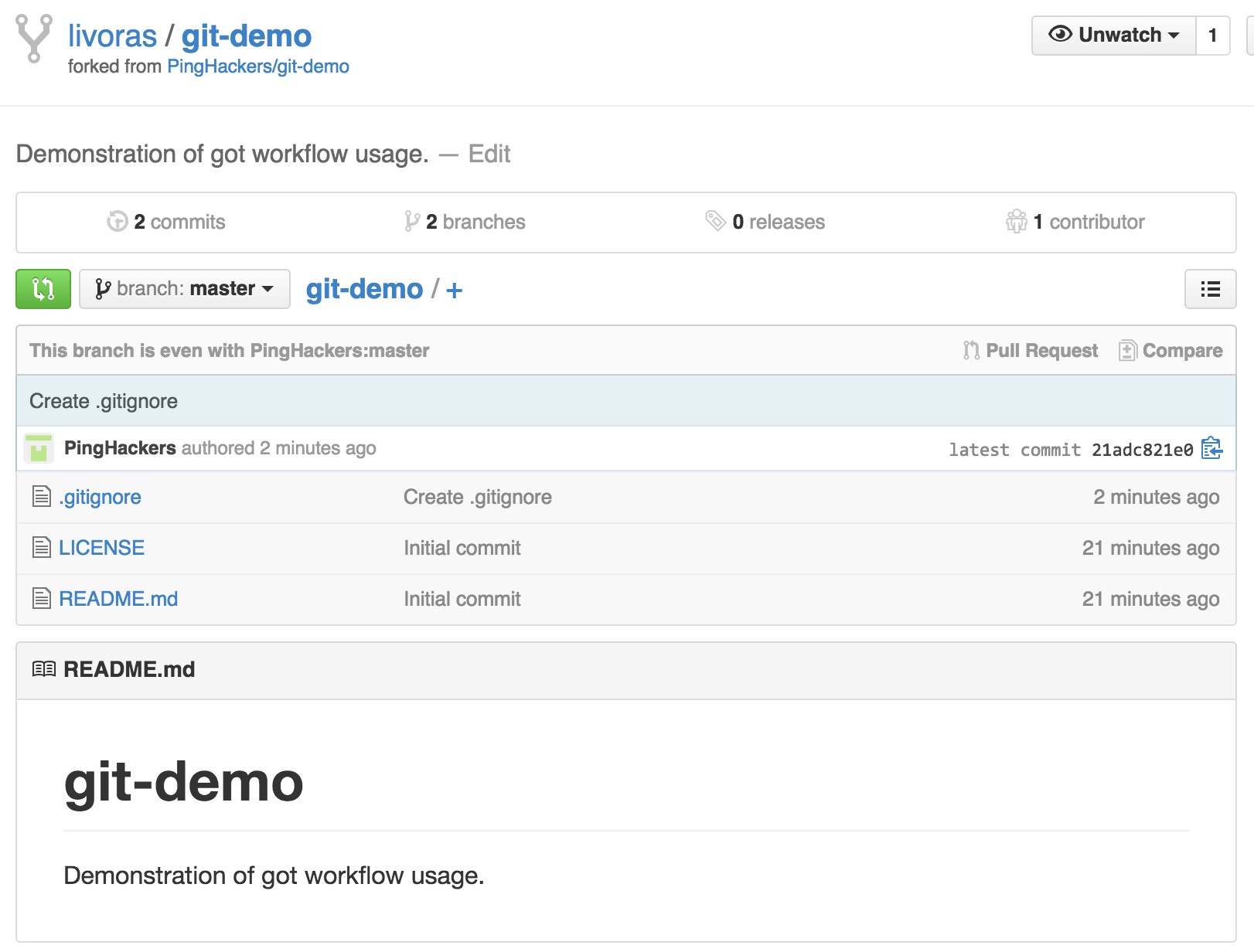


Step 2：开发者fork源仓库

源仓库建立以后，每个开发就可以去复制一份源仓库到自己的github账号中，然后作为自己开发所用的仓库。假设我是一个项目中的开发者，我就到[PingHackers/git-demo](https://github.com/PingHackers/git-demo" \t "_blank)项目主页上去fork：



fork完以后，我就可以在我自己的仓库列表中看到一个和源仓库一模一样的复制品。这时就应该感叹，你以后要和它相依为命了：



Step 3：把自己开发者仓库clone到本地

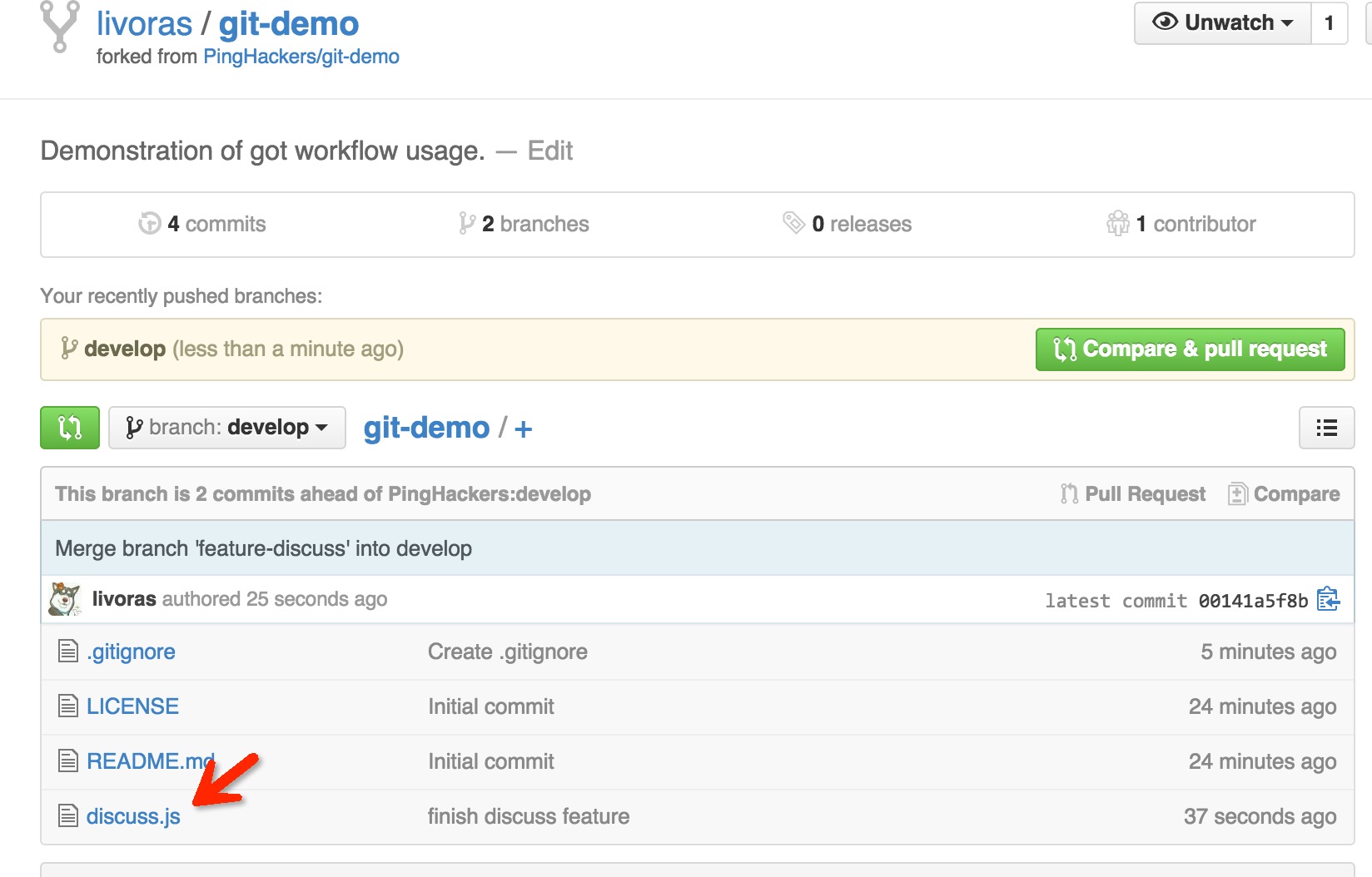
这一步应该不用教，git clone

Step 4：构建功能分支进行开发

进入仓库中，按照前面说所的构建功能分支的步骤，构建功能分支进行开发、合并，假设我现在要开发一个“讨论”功能：

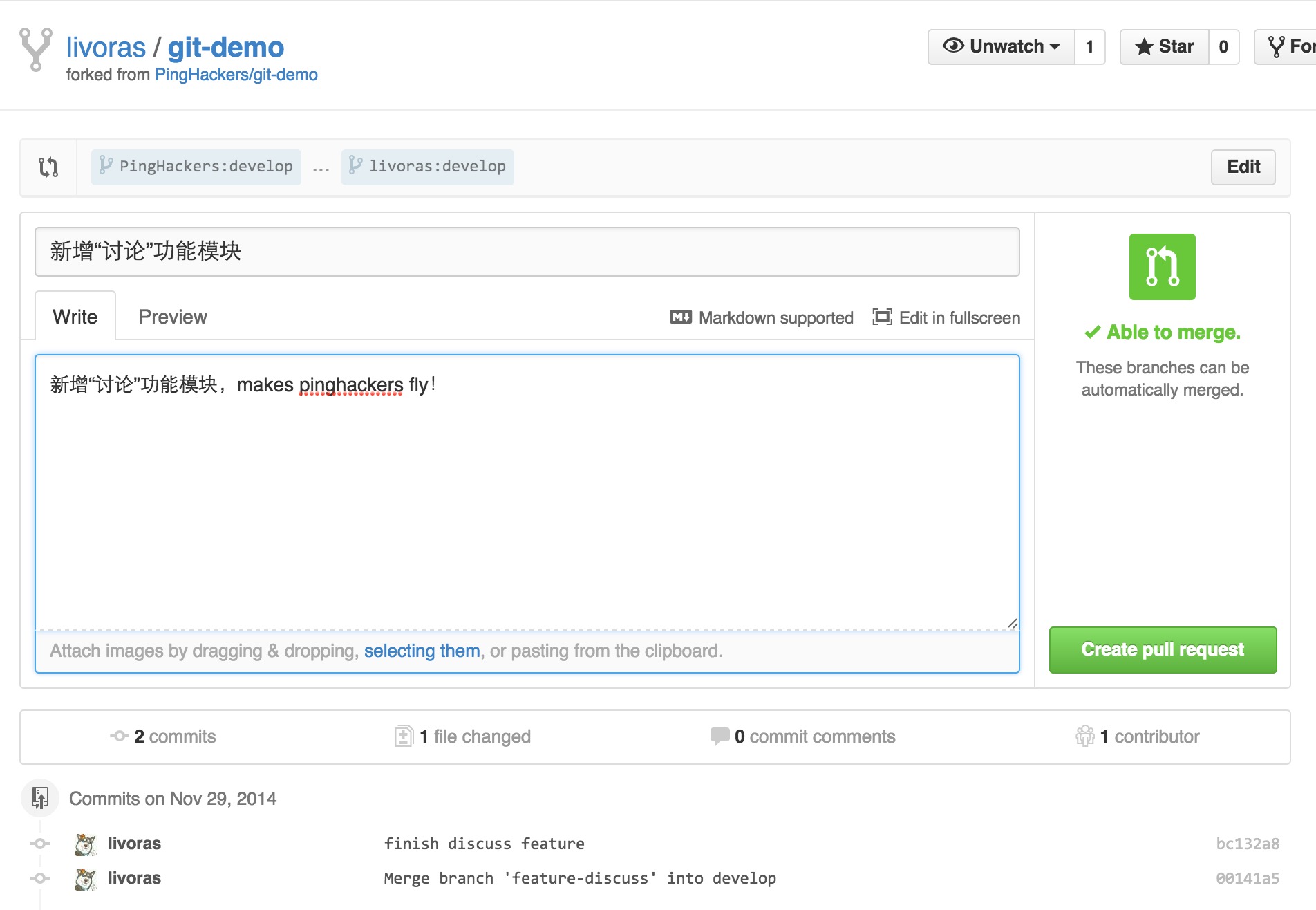
>>> git checkout develop # 切换到`develop`分支 >>> git checkout -b feature-discuss # 分出一个功能性分支 >> touch discuss.js # 假装discuss.js就是我们要开发的功能 >> git add . >> git commit -m 'finish discuss feature' # 提交更改 >>> git checkout develop # 回到develop分支 >>> git merge --no-ff feature-discuss # 把做好的功能合并到develop中 >>> git branch -d feature-discuss # 删除功能性分支 >>> git push origin develop # 把develop提交到自己的远程仓库中

这时候，你上自己github的项目主页中develop分支中看看，已经有discuss.js这个文件了：

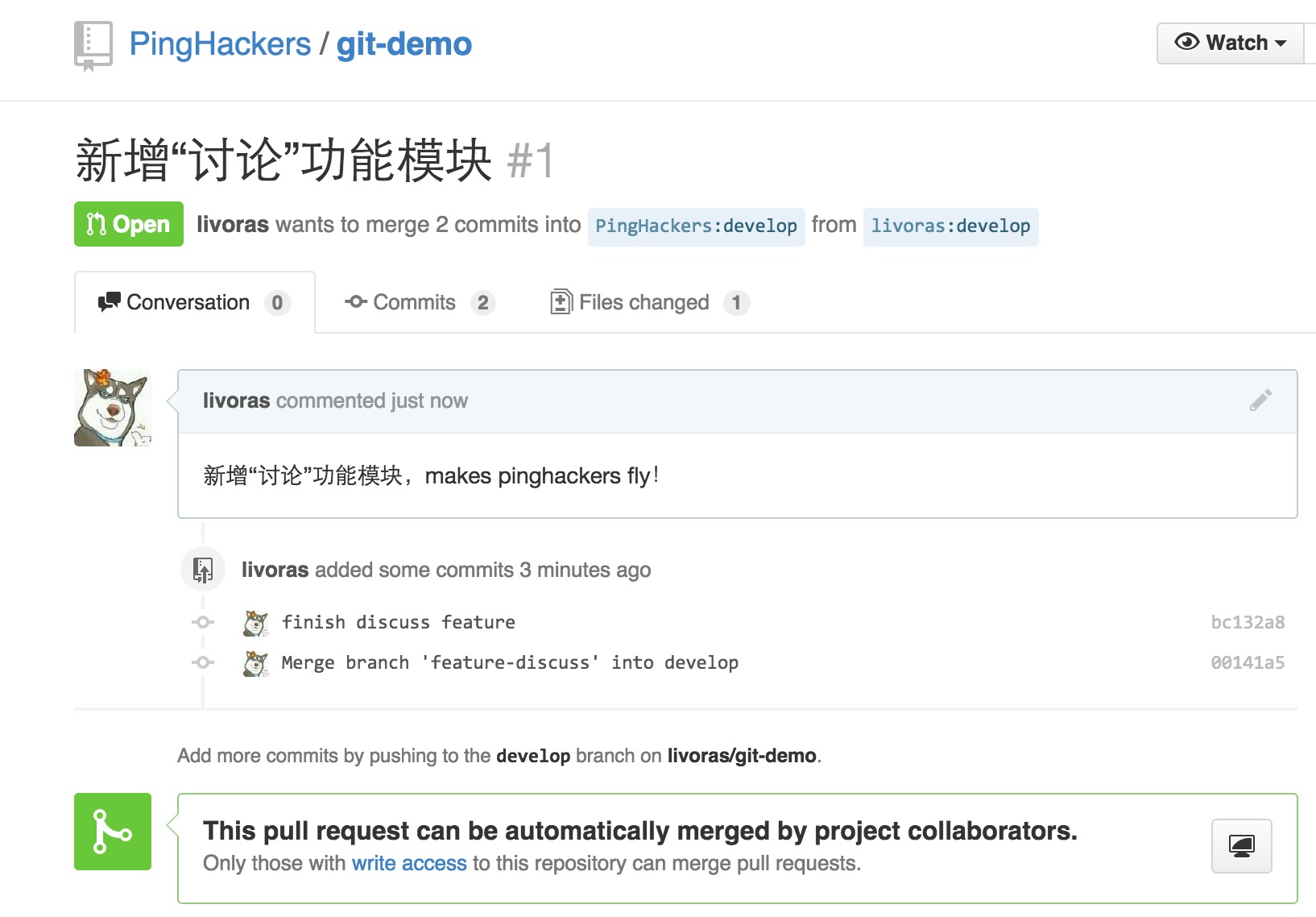


Step 5：向管理员提交pull request

假设我完成了“讨论”功能（当然，你还可能对自己的develop进行了多次合并，完成了多个功能），经过测试以后，觉得没问题，就可以请求管理员把**自己仓库的develop分支**合并到**源仓库的develop**分支中，这就是传说中的pull request。

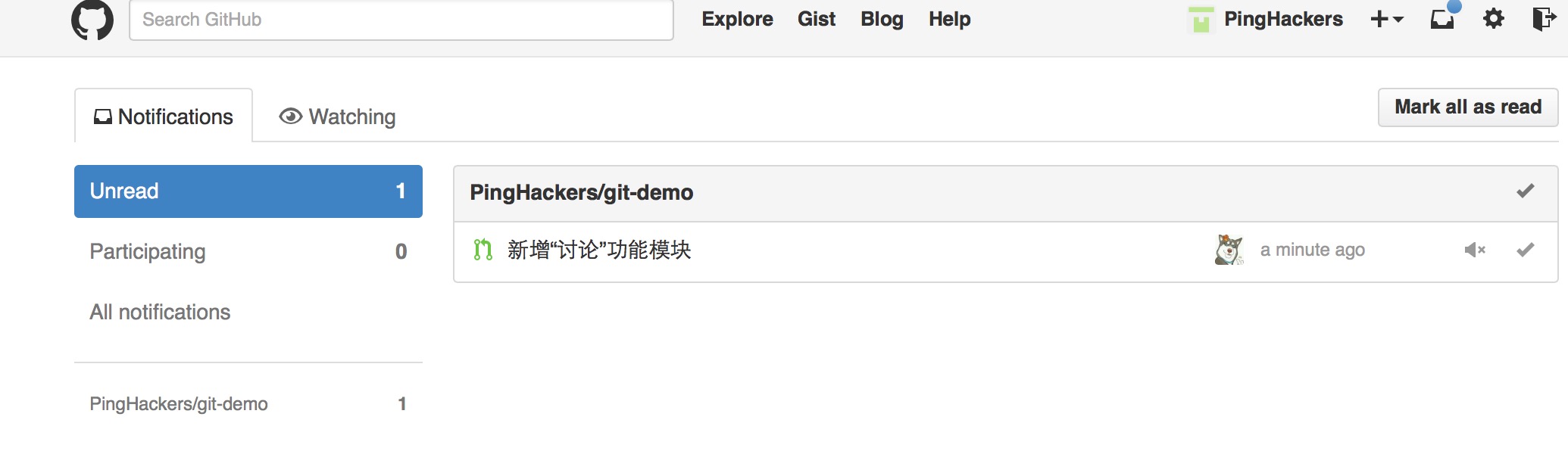


点击上图的绿色按钮，开发者就可以就可以静静地等待管理员对你的提交的评审了。



Step 6 管理员测试、合并

接下来就是管理员的操作了，作为管理员的PingHackers登陆github，便看到了我对源仓库发起的pull request。



这时候PingHackers需要做的事情就是：

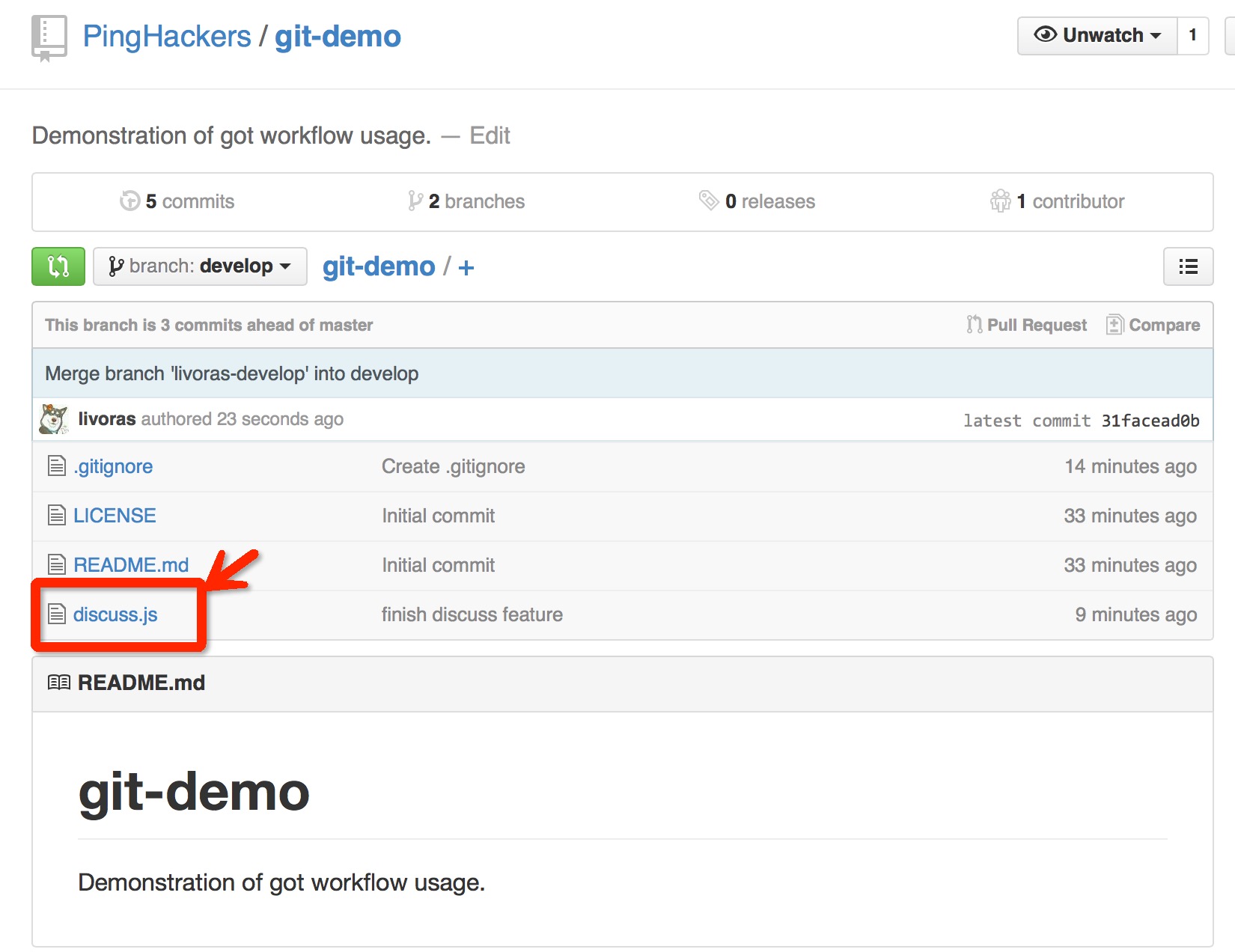
1. **对我的代码进行review**。github提供非常强大的代码review功能：  
   
2. **在他的本地测试新建一个测试分支**，测试我的代码：

>> git checkout develop # 进入他本地的develop分支 >> git checkout -b livoras-develop # 从develop分支中分出一个叫livoras-develop的测试分支测试我的代码 >> git pull https://github.com/livoras/git-demo.git develop # 把我的代码pull到测试分支中，进行测试

1. **判断是否同意合并到源仓库的develop中**，如果经过测试没问题，可以把我的代码合并到源仓库的develop中：

>> git checkout develop >> git merge --no-ff livoras-develop >> git push origin develop

注意，PingHakers一直在操作的仓库是源仓库。所以我们经过上面一系列操作以后，就可以在源仓库主页中看到：



经过辗转曲折的路程，我们的discuss.js终于从我的开发仓库的功能分支到达了源仓库的develop分支中。以上，就是一个git & github协同工作流的基本步骤。