https://blog.csdn.net/dajijidanaizi/article/details/52698971

**在Github和Git上fork之简单指南**

2016年09月29日 09:59:08 [dajijidanaizi](https://me.csdn.net/dajijidanaizi" \t "_blank) 阅读数：2197

from:https://linux.cn/article-4292-1-rss.html

以我的经验来看，刚接触Git和GitHub时，最困扰的一件事情就是尝试解决下面的问题：在Git和GitHub上，我能做什么？

Git教程往往不会解决这个问题，因为它集中篇幅来教你Git命令和概念，并且不认为你会使用GitHub。[GitHub帮助教程](https://help.github.com/" \t "_blank)一定程度上弥补了这一缺陷，但是它每篇文章的关注点都较为狭隘，而且没有提供关于"Git vs GitHub"问题的概念性概述。

如果你是习惯于先理解概念，再着手代码的学习者，而且你也是Git和GitHub的初学者，我建议你先理解清楚什么是fork。为什么呢 ？

1. Fork是在GitHub起步最普遍的方式。
2. Fork只需要很少的Git命令，但是起得作用却非常大。
3. Fork提供了对Git和GitHub最基础的了解，有益于你之后的工作。



本篇指南使用两张简单的图表，来教会你fork的两种主要工作流程。我并不打算涉及任何代码，但是在结论中，我会把你需要使用的代码的链接给你。

**fork并且更新一个仓库**

现在有这样一种情形：有一个叫做Joe的程序猿写了一个游戏程序，而你可能要去改进它。并且Joe将他的代码放在了GitHub仓库上。下面是你要做的事情：



Alt text

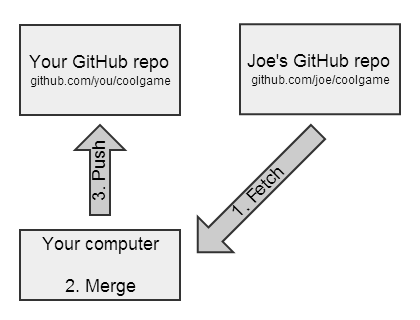
fork并且更新GitHub仓库的图表演示

1. Fork他的仓库：这是GitHub操作，这个操作会复制Joe的仓库（包括文件，提交历史，issues，和其余一些东西）。复制后的仓库在你自己的GitHub帐号下。目前，你本地计算机对这个仓库没有任何操作。
2. Clone你的仓库：这是Git操作。使用该操作让你发送"请给我发一份我仓库的复制文件"的命令给GitHub。现在这个仓库就会存储在你本地计算机上。
3. 更新某些文件：现在，你可以在任何程序或者环境下更新仓库里的文件。
4. 提交你的更改：这是Git操作。使用该操作让你发送"记录我的更改"的命令至GitHub。此操作只在你的本地计算机上完成。
5. 将你的更改push到你的GitHub仓库：这是Git操作。使用该操作让你发送"这是我的修改"的信息给GitHub。Push操作不会自动完成，所以直到你做了push操作，GitHub才知道你的提交。
6. 给Joe发送一个pull request：如果你认为Joe会接受你的修改，你就可以给他发送一个pull request。这是GitHub操作，使用此操作可以帮助你和Joe交流你的修改，并且询问Joe是否愿意接受你的"pull request"，当然，接不接受完全取决于他自己。

如果Joe接受了你的pull request，他将把那些修改拉到自己的仓库。胜利！

**同步一个fork**

Joe和其余贡献者已经对这个项目做了一些修改，而你将在他们的修改的基础上，还要再做一些修改。在你开始之前，你最好"同步你的fork"，以确保在最新的复制版本里工作。下面是你要做的：



同步GitHub fork的图表示意图

1. 从Joe的仓库中取出那些变化的文件：这是Git操作，使用该命令让你可以从Joe的仓库获取最新的文件。
2. 将这些修改合并到你自己的仓库：这是Git操作，使用该命令使得那些修改更新到你的本地计算机（那些修改暂时存放在一个"分支"中）。记住：步骤1和2经常结合为一个命令使用，合并后的Git命令叫做"pull"。
3. 将那些修改更新推送到你的GitHub仓库（可选）：记住，你本地计算机不会自动更新你的GitHub仓库。所以，唯一更新GitHub仓库的办法就是将那些修改推送上去。你可以在步骤2完成后立即执行push，也可以等到你做了自己的一些修改，并已经本地提交后再执行推送操作。

比较一下fork和同步工作流程的区别：当你最初fork一个仓库的时候，信息的流向是从Joe的仓库到你的仓库，然后再到你本地计算机。但是最初的过程之后，信息的流向是从Joe的仓库到你的本地计算机，之后再到你的仓库。

**结论**

我希望这是一篇关于GitHub和Git 的 [fork](https://help.github.com/articles/fork-a-repo)有用概述。现在，你已经理解了那些概念，你将会更容易地在实际中执行你的代码。GitHub关于fork和[同步](https://help.github.com/articles/syncing-a-fork" \t "_blank)的文章将会给你大部分你需要的代码。

如果你是Git的初学者，而且你很喜欢这种学习方式，那么我极力推荐书籍[Pro Git](http://git-scm.com/book" \t "_blank)的前两个章节，网上是可以免费查阅的。

如果你喜欢视频学习，我创建了一个[11部分的视频系列](http://www.dataschool.io/git-and-github-videos-for-beginners/)（总共36分钟），来向初学者介绍Git和GitHub。