[Git 问题, 一个 master, 多个新功能分支, 怎样有序地合并和提交?](https://segmentfault.com/q/1010000000181403)

* [github](https://segmentfault.com/t/github)

* [git](https://segmentfault.com/t/git)

[**题叶**](https://segmentfault.com/u/jiyinyiyong) 2013年03月22日提问

* 关注 **18** 关注
* 收藏 **28** 收藏，36.4k 浏览

问题对人有帮助，内容完整，我也想知道答案0问题没有实际价值，缺少关键内容，没有改进余地

以前大多个是一个的使用 Git, 到 Github 上提交的场景, 对多人开发合并项目经验不多,  
现在遇到的是在 Github 上存在主分支, 本地需要修改多个功能和 bug 等等,  
我是按以前实习回来的同学提示, 在多个分支开发不同的功能, 然后合并提交..  
合并和提交的顺序不是确定的, 因此不能简单直接用 merge 每次一个个叠加.  
有时我用 rebase, 但有发现 commit 顺序不是时间顺序, 到线上被 merge 以后也不是非常清晰  
于是我想问一下面对这样的场景, 用怎样的方式管理会更合适?

有在 Google, 但一些细节不清晰.. 比如 commit 显示顺序.. 还有再次被 merge 后的细节..

* [2013年03月22日提问](https://segmentfault.com/q/1010000000181403)
* [2 评论](javascript:void(0);)
* [邀请回答](javascript:;)
* [编辑](javascript:;)
* [Translate](javascript:;)

1

有一个著名的分支模型呢。也可以看看github flow [http://hooopo.writings.io/articles/fe...](http://hooopo.writings.io/articles/fe2b0791)

— [**hit9**](https://segmentfault.com/u/hit9) · 2013年03月22日

[展开评论](javascript:void(0);)

https://sponsor.segmentfault.com/lg.php?bannerid=0&campaignid=0&zoneid=7&loc=https%3A%2F%2Fsegmentfault.com%2Fq%2F1010000000181403&cb=6b2e087e2e

https://sponsor.segmentfault.com/lg.php?bannerid=0&campaignid=0&zoneid=8&loc=https%3A%2F%2Fsegmentfault.com%2Fq%2F1010000000181403&cb=b524eb1e15

[默认排序](https://segmentfault.com/q/1010000000181403#answers-title)[时间排序](https://segmentfault.com/q/1010000000181403?sort=created#answers-title)

3个回答

答案对人有帮助，有参考价值11答案没帮助，是错误的答案，答非所问

git支持很多种工作流程，我们采用的一般是这样，远程创建一个主分支，本地每人创建功能分支，日常工作流程如下：

去自己的工作分支  
$ git checkout work

工作  
....

提交工作分支的修改  
$ git commit -a

回到主分支  
$ git checkout master

获取远程最新的修改，此时不会产生冲突  
$ git pull

回到工作分支  
$ git checkout work

用rebase合并主干的修改，如果有冲突在此时解决  
$ git rebase master

回到主分支  
$ git checkout master

合并工作分支的修改，此时不会产生冲突。  
$ git merge work

提交到远程主干  
$ git push

这样做的好处是，远程主干上的历史永远是线性的。每个人在本地分支解决冲突，不会在主干上产生冲突。

* [2013年03月22日回答](https://segmentfault.com/q/1010000000181403/a-1020000000181433)
* [3 评论](javascript:void(0);)
* [Translate](javascript:;)
* [编辑](javascript:;)

[](https://segmentfault.com/u/coder)

[coder](https://segmentfault.com/u/coder)2k 声望

答案对人有帮助，有参考价值0答案没帮助，是错误的答案，答非所问

上面那位仁兄的回答很好，我这里再补充一个资料，给楼主作为参考。这个是阮兄写的，[git分支管理策略]([http://www.ruanyifeng.com/blog/2012/0...](http://www.ruanyifeng.com/blog/2012/07/git.html))。

* [2013年03月22日回答](https://segmentfault.com/q/1010000000181403/a-1020000000181554)
* [评论](javascript:void(0);)
* [Translate](javascript:;)
* [编辑](javascript:;)

[](https://segmentfault.com/u/juchen)

[巨臣](https://segmentfault.com/u/juchen)2 声望

答案对人有帮助，有参考价值0答案没帮助，是错误的答案，答非所问

可以在一条分支上一起开发，你有变更的时候，在提交前，使用

git stash

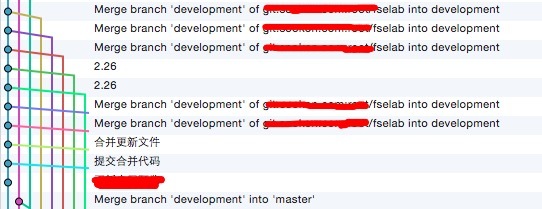
这样将本地的修改全部缓存在一个堆栈中了，然后把别人的修改同步过来

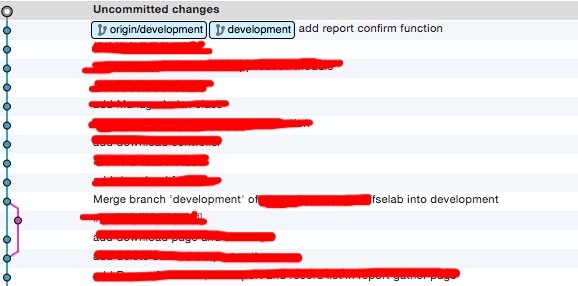
git pull --rebase

下一步是将自己的变更恢复到最新的节点上

git stash **pop**

然后再使用git commit提交，这样就会让一个分支的版本按顺序继续发展，而不是像织毛衣一样，你可以看一下我们使用这种方法前后的对比图

之前：  


之后  


* [2015年04月20日更新](https://segmentfault.com/q/1010000000181403/a-1020000002692071)
* [评论](javascript:void(0);)
* [Translate](javascript:;)
* [编辑](javascript:;)

[](https://segmentfault.com/u/liujin834)

[liujin834](https://segmentfault.com/u/liujin834)3 声望