

Git 原理详解及实用指南 - 扔物线 - 掘金小册

高级 3：写错的不是最新的提交，而是倒数第二个？

`commit --amend` 可以修复最新 `commit` 的错误，但如果是倒数第二个 `commit` 写错了，怎么办？

rebase -i: 交互式 rebase

如果不是最新的 `commit` 写错，就不能用 `commit --amend` 来修复了，而是要用 `rebase`。不过需要给 `rebase` 也加一个参数：`-i`。

`rebase -i` 是 `rebase --interactive` 的缩写形式，意为「交互式 rebase」。

所谓「交互式 rebase」，就是在 `rebase` 的操作执行之前，你可以指定要 `rebase` 的 `commit` 链中的每一个 `commit` 是否需要进一步修改。

那么你就可以利用这个特点，进行一次「原地 rebase」。

例如你是在写错了 `commit` 之后，又提交了一次才发现之前写错了：

```
git log
```



```
commit 5b6778f3fefe32915e851a90c6f8829e57c67474 (HEAD -> new_feature)
Author: Kai Zhu <rengwuxian@gmail.com>
Date: Tue Nov 21 22:45:31 2017 +0800

    增加常见哭声集合

commit be62a93a69bc2024225d04a0890101e2bf04881f
Author: Kai Zhu <rengwuxian@gmail.com>
Date: Tue Nov 21 22:46:10 2017 +0800

    增加常见笑声集合

commit 1c2841cb8cd22fc0fddae2809e6e316c4593a11a (origin/master, origin/HEAD)
Merge: 57eeb24 46ed07f
Author: Kai Zhu <rengwuxian@gmail.com>
```

开启交互式 rebase 过程

现在再用 `commit --amend` 已经晚了，但可以用 `rebase -i`：

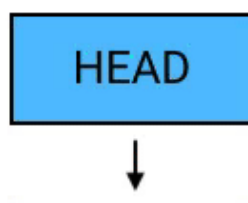
```
git rebase -i HEAD^^
```

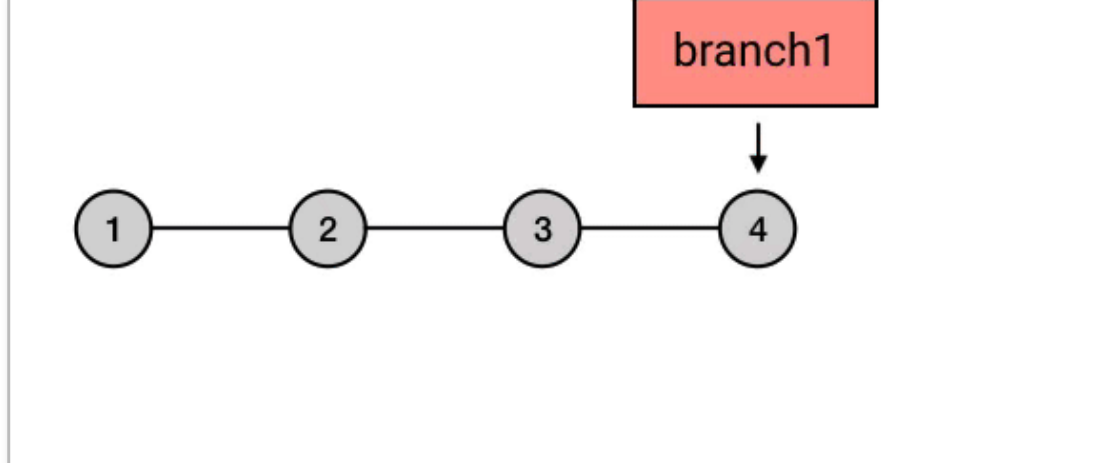
说明：在 Git 中，有两个「偏移符号」：`^` 和 `~`。

`^` 的用法：在 `commit` 的后面加一个或多个 `^` 号，可以把 `commit` 往回偏移，偏移的数量是 `^` 的数量。例如：`master^` 表示 `master` 指向的 `commit` 之前的那个 `commit`；`HEAD^^` 表示 `HEAD` 所指向的 `commit` 往前数两个 `commit`。

`~` 的用法：在 `commit` 的后面加上 `~` 号和一个数，可以把 `commit` 往回偏移，偏移的数量是 `~` 号后面的数。例如：`HEAD~5` 表示 `HEAD` 指向的 `commit` 往前数 5 个 `commit`。

上面这行代码表示，把当前 `commit`（`HEAD` 所指向的 `commit`）`rebase` 到 `HEAD` 之前 2 个的 `commit` 上：





如果没有 `-i` 参数的话，这种「原地 rebase」相当于空操作，会直接结束。而在加了 `-i` 后，就会跳到一个新的界面：

```
pick be62a93 增加常见笑声集合
pick 5b6778f 增加常见哭声集合

# Rebase 1c2841c..5b6778f onto 1c2841c (2 commands)
#
# Commands:
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
# Note that empty commits are commented out
```

编辑界面：选择 commit 和对应的操作

这个编辑界面的最顶部，列出了将要「被 rebase」的所有 `commit` s，也就是倒数第二个 `commit` 「增加常见笑声集合」和最新的 `commit` 「增加常

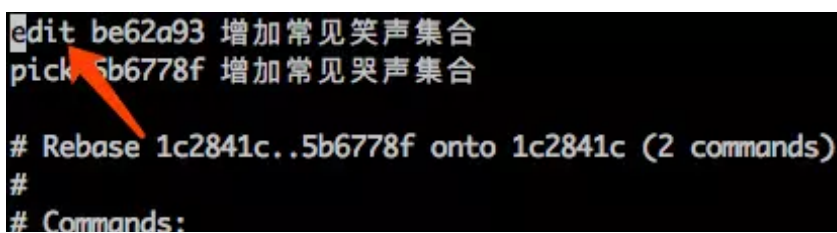
见哭声集合」。需要注意，这个排列是正序的，旧的 `commit` 会排在上面，新的排在下面。

这两行指示了两个信息：

- 需要处理哪些 `commit` s;
- 怎么处理它们。

你需要修改这两行的内容来指定你需要的操作。每个 `commit` 默认的操作都是 `pick`（从图中也可以看出），表示「直接应用这个 `commit`」。所以如果你现在直接退出编辑界面，那么结果仍然是空操作。

但你的目标是修改倒数第二个 `commit`，也就是上面的那个「增加常见笑声集合」，所以你需要把它的操作指令从 `pick` 改成 `edit`。`edit` 的意思是「应用这个 `commit`，然后停下来等待继续修正」。其他的操作指令，在这个界面里都已经列举了出来（下面的 "Commands..." 部分文字），你可以自己研究一下。



```
edit be62a93 增加常见笑声集合
pick 5b6778f 增加常见哭声集合

# Rebase 1c2841c..5b6778f onto 1c2841c (2 commands)
#
# Commands:
```

```
# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's log message
# x, exec = run command (the rest of the line) using shell
# d, drop = remove commit
"
```

把 `pick` 修改成 `edit` 后，就可以退出编辑界面了：

```
→ git-practice git:(new_feature) git rebase -i HEAD^^
Stopped at be62a93... 增加常见笑声集合
You can amend the commit now, with

    git commit --amend

Once you are satisfied with your changes, run

    git rebase --continue
```

上图的提示信息说明，`rebase` 过程已经停在了第二个 `commit` 的位置，那么现在你就可以去修改你想修改的内容了。

修改写错的 commit

修改完成之后，和上节里的方法一样，用 `commit --amend` 来把修正应用到当前最新的 `commit`：

```
git add 笑声
git commit --amend
```

```
→ git-practice git:(be62a93) git commit --amend  
[detached HEAD 632d0b5] 增加常见笑声集合  
Date: Tue Nov 21 22:46:10 2017 +0800  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 feature4
```

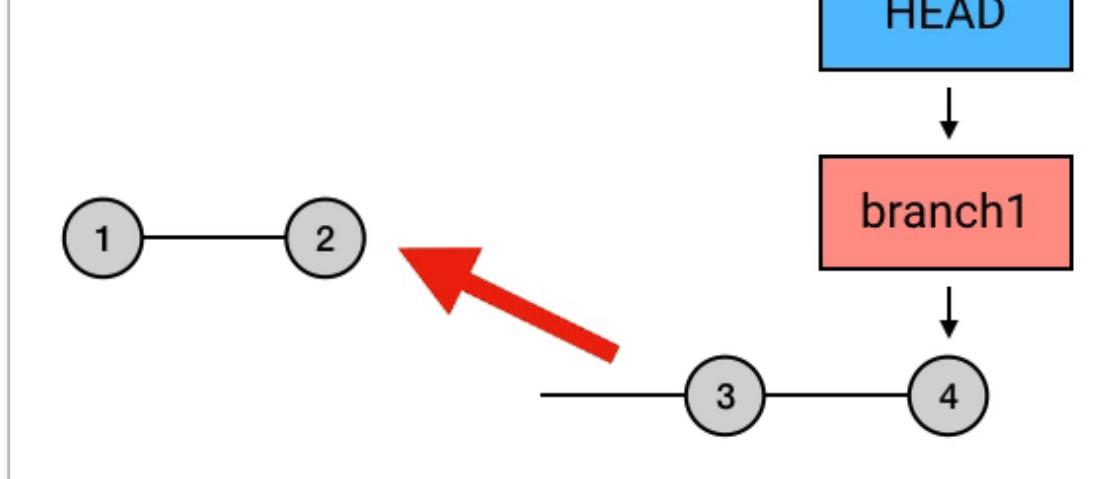
继续 rebase 过程

在修复完成之后，就可以用 `rebase --continue` 来继续 `rebase` 过程，把后面的 `commit` 直接应用上去。

```
git rebase --continue
```

```
→ git-practice git:(632d0b5) git rebase --continue  
Successfully rebased and updated refs/heads/new_feature.  
→ git-practice git:(new_feature) █
```

然后，这次交互式 `rebase` 的过程就完美结束了，你的那个倒数第二个写错的 `commit` 就也被修正了：



实质上，交互式 `rebase` 并不是必须应用在「原地 rebase」上来修改写错的 `commit`，这只不过是它最常见的用法。你同样也可以把它用在分叉的 `commit` 上，不过这个你就可以自己去研究一下了。

小结

这节介绍了交互式 `rebase`，它可以在 `rebase` 开始之前指定一些额外操作。交互式 `rebase` 最常用的场景是修改写错的 `commit`，但也可以用作其他用途。它的大致用法：

- 使用方式是 `git rebase -i 目标commit`；
- 在编辑界面中指定需要操作的 `commit` s 以及操作类型；
- 操作完成之后用 `git rebase --continue` 来继续 `rebase` 过程。

留言

评论将在后台进行审核，审核通过后对所有人可见



sailingfaraway java web

修改 commit 之前是不是要先拷贝一份备份一下，不然一顿骚操作之后改错了怎么办，还是有其他更好的备份方法？

▲0

评论 1 月前



ViYaYaYa

~ 和 ^ 符号的解释不太正确，具体看图

<http://static.paulboxley.com/images/2011/git-graph.svg>

▲1

评论 1 月前



尚寂清君

怎么把多个 commit 合并成一个 commit 呢?

▲0

收起评论 2 月前

•

dhucst

在编辑界面中使用
squash 命令

2 月前

评论审核通过后显示

评论

•

订阅公众号

这么说，两个「偏移符号」的作用是一模一样的了??

▲0

评论 3 月前

•

订阅公众号

写下 rebase 指令之后，隐隐约约感觉到自己变基了。。。

▲0

评论 3 月前



碧海银剑

变基用法讲解的很清晰！

▲0

评论 3 月前



沐小枫、

You must edit all merge conflicts and then mark them as resolved using git add
修改完执行 git rebase --continue 出现上面的报错，要怎么处理？

▲0

收起评论 4 月前



zsp109096

学生 @ 南昌大学

处理掉冲突

3 月前

评论审核通过后显示

评论

●
xgw_shen php 后端程序员 @ mohism

难道只有我美咋滴懂。。。。。。。。

▲0 评论 5 月前

●
HydeSong 前端开发工程师 @ 蘑菇公寓

问题：怎么我执行 git rebase -i HEAD^^ 后，有三条 pick？

pick 47dec9b branch feature1

pick 0a8610d hahaha

pick 2af2c7f ahead of master 1 commit

▲0 评论 5 月前

●
zp9011203932

同一个文件的提交 修改不了

▲0 评论 9 月前

●
chenxiaokang9731

Android 开发 @ longbill

牛牛牛



Amor Android 开发工程师

为什么我写的在 git rebase --continue 的时候会提示我要 merge... 我的订正文件发现里面有冲突了.



博弈史密斯

Android 开发工程师

倒数第二次 commit 和 倒数第一次 commit, 不能是作用同一个文件。同一个文件 因两次提交的文件的 对应行内容不同, 导致冲突。两次 commit 针对不同的文件 即可。

9 月前

评论审核通过后显示

评论

●

冰块儿

这个操作真的太爽了！测试了一下自由度很高！

修改个文案什么的再也不用改完再推了！分支变得很整洁！很棒！

▲0 评论 9 月前

●

请叫我王磊同学 前端工程师 @ 帆软软件

能麻烦告诉一下 edit squash 和 fixup 之前的区别，这部分没看懂

▲0 收起评论 9 月前

⬆

● 请叫我王磊同学
前端工程师 @ 帆
软软件
之间的区别
9 月前

● CCsguai
全站打砸 @ (๑•̀ㅂ
•)و✧
squash 会合并

commit 同时保留
commit 信息 fixup
类似 squash 但不会
保留 commit 信息
edit 凯哥上面已经说
了不是

9 月前

评论审核通过后显示

评论

•

boboyuwu

这个跟 rebase 分支有什么区别？

▲0

收起评论 9 月前

•

Landroid

Android 应用开发
工程师

rebase 分支是为了
去掉某些分支，这个
骚操作是为了修正某
条分支上某个
commit

6 月前

评论审核通过后显示

评论

●

码青 安卓软件开发 @ 无

如果已经 push 后发现第二个有错误就不能这样操作了吧?

▲0

收起评论 9 月前

●

HowieChang
Android 开发 @
海康威视

不影响, push 只是将本地操作上传, 你这样操作以后再 push 就行

9 月前

●

HowieChang
Android 开发 @

海康威视

不过好像会提示文件过期, 不给 push, 要

pull 下来解决一下冲突

9 月前

- 扔物线
Android 布道师
(假装) @
HenCoder
push 之后才发现错误
可以看第 16 节
9 月前

评论审核通过后显示

评论

•

noone9

感谢

▲0

评论 9 月前

全文完

