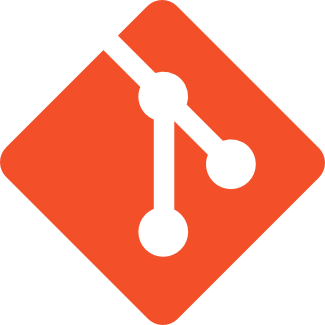
[[一介布衣](http://yijiebuyi.com/)](http://yijiebuyi.com/)

* [首页](http://yijiebuyi.com/)
* [前端](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)
* [Node.js](http://yijiebuyi.com/category/nodejs.html)
* [数据库](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)
* [系统](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)
* [大杂烩](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)
* [在线工具 **NEW**](http://yijiebuyi.com/tools.html)
* [招聘](http://yijiebuyi.com/category/%E6%8B%9B%E8%81%98.html)
* [关于](http://yijiebuyi.com/about.html)
* [首页](http://yijiebuyi.com/)
* [开发工具](http://yijiebuyi.com/category/%E5%BC%80%E5%8F%91%E5%B7%A5%E5%85%B7.html)
* git reset revert 回退回滚取消提交返回上一版本

git reset revert 回退回滚取消提交返回上一版本

[一介布衣](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)  2015-01-07  [开发工具](http://yijiebuyi.com/category/%E5%BC%80%E5%8F%91%E5%B7%A5%E5%85%B7.html)  284257

[分享到：QQ空间新浪微博腾讯微博人人网微信](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)



总有一天你会遇到下面的问题.

(1)改完代码匆忙提交,上线发现有问题,怎么办? 赶紧回滚.

(2)改完代码测试也没有问题,但是上线发现你的修改影响了之前运行正常的代码报错,必须回滚.

这些开发中很常见的问题,所以git的取消提交,回退甚至返回上一版本都是特别重要的.

大致分为下面2种情况:

1.没有push

这种情况发生在你的本地代码仓库,可能你add ,commit 以后发现代码有点问题,准备取消提交,用到下面命令

reset

git reset [--soft | --mixed | --hard

上面常见三种类型

--mixed

会保留源码,只是将git commit和index 信息回退到了某个版本.

git reset 默认是 --mixed 模式

git reset --mixed  等价于  git reset

--soft

保留源码,只回退到commit 信息到某个版本.不涉及index的回退,如果还需要提交,直接commit即可.

--hard

源码也会回退到某个版本,commit和index 都回回退到某个版本.(注意,这种方式是改变本地代码仓库源码)

当然有人在push代码以后,也使用 reset --hard <commit...> 回退代码到某个版本之前,但是这样会有一个问题,你线上的代码没有变,线上commit,index都没有变,当你把本地代码修改完提交的时候你会发现权是冲突.....

所以,这种情况你要使用下面的方式

2.已经push

对于已经把代码push到线上仓库,你回退本地代码其实也想同时回退线上代码,回滚到某个指定的版本,线上,线下代码保持一致.你要用到下面的命令

revert

git revert用于反转提交,执行evert命令时要求工作树必须是干净的.

git revert用一个新提交来消除一个历史提交所做的任何修改.

revert 之后你的本地代码会回滚到指定的历史版本,这时你再 git push 既可以把线上的代码更新.(这里不会像reset造成冲突的问题)

revert 使用,需要先找到你想回滚版本唯一的commit标识代码,可以用 git log 或者在adgit搭建的web环境历史提交记录里查看.

git revert c011eb3c20ba6fb38cc94fe5a8dda366a3990c61

通常,前几位即可

git revert c011eb3

git revert是用一次新的commit来回滚之前的commit，git reset是直接删除指定的commit

看似达到的效果是一样的,其实完全不同.

第一:

上面我们说的如果你已经push到线上代码库, reset 删除指定commit以后,你git push可能导致一大堆冲突.但是revert 并不会.

第二:

如果在日后现有分支和历史分支需要合并的时候,reset 恢复部分的代码依然会出现在历史分支里.但是revert 方向提交的commit 并不会出现在历史分支里.

第三:

reset 是在正常的commit历史中,删除了指定的commit,这时 HEAD 是向后移动了,而 revert 是在正常的commit历史中再commit一次,只不过是反向提交,他的 HEAD 是一直向前的.

你可能感兴趣

* [chrome 浏览器 console 加入 jquery 测试调试](http://yijiebuyi.com/blog/131c824ef0b370fc0ce11a00a0f0f11b.html)
* [nginx 命令行参数 启动 重启 重载 停止](http://yijiebuyi.com/blog/051422d9b1e6f477823fdcb79759b5b3.html)
* [git cherry-pick 把提交到A分支的部分commit 再提交到B分支上](http://yijiebuyi.com/blog/0e65f4a59a1cfa05c5b30ccb6c2f413d.html)
* [git 打标签并推送tag到托管服务器](http://yijiebuyi.com/blog/007269d04d5096d9397ce3daf9d84c48.html)
* [阿里云yum升级python2.7 CentOS默认自带pyton2.6](http://yijiebuyi.com/blog/108ae6186bb00cc708bc54f02adec277.html)

本文由 [**一介布衣**](http://yijiebuyi.com/) 创作，采用 [**知识共享署名 3.0**](http://creativecommons.org/licenses/by/3.0/cn) 中国大陆许可协议。   
可自由转载、引用，但需署名作者且注明文章出处。

本站部署于「[**UCloud云计算**](https://ucloud.cn/)」。  
**右侧微信扫描 「打赏」,感谢支持!**



* [reset](http://yijiebuyi.com/tag/reset.html)

* [revert](http://yijiebuyi.com/tag/revert.html)

* [git回滚](http://yijiebuyi.com/tag/git%E5%9B%9E%E6%BB%9A.html)

* [取消提交](http://yijiebuyi.com/tag/%E5%8F%96%E6%B6%88%E6%8F%90%E4%BA%A4.html)

窗体顶端



窗体底端

[Node.js](http://yijiebuyi.com/so.html?k=node.js) + [LevelDB](http://yijiebuyi.com/so.html?k=leveldb) + [UCloud](https://www.ucloud.cn/) 驱动

[](http://www.ucloud.cn/site/active/gift.html?utm_source=yijiebuyi&utm_medium=content_pic_pc&utm_campaign=kuai&utm_content=gift&ytag=yijiebuyi)

专题

[](http://yijiebuyi.com/blog/f1b99a4af083dbb9d868e6c241e525f1.html)

最新发布

* [UCloud 云服务器硬盘扩容后 如何挂载到本机](http://yijiebuyi.com/blog/bd5f26420e5141e005fe5ced6b56c2a7.html)
* [[北京] 急聘 java高级架构师,高级开发工程师](http://yijiebuyi.com/blog/535940b52293e32d2260a2fd9a8769c2.html)
* [UCloud 云服务器 centOS6 64位 安装mongodb 数据库](http://yijiebuyi.com/blog/58cc4c3e2fb748dbcc57a30db9668776.html)
* [Cannot load from mysql.proc.The table is probably corrupted](http://yijiebuyi.com/blog/e8c3354b7fb946924ba77e56a92b9df5.html)
* [JetBrains 数据库管理工具 DataGrip for Mac 1.0.2](http://yijiebuyi.com/blog/c5a8327fd6277884a12dc66dd997120d.html)
* [lua Mac brew 安装使用 及简单语法介绍](http://yijiebuyi.com/blog/c0dbbc2f64fb2a5a1fe4e8781be592ff.html)
* [[投稿]长期招聘 node.js 实习生](http://yijiebuyi.com/blog/b8d8acaaa279ab7c159facc41fe069ad.html)
* [javascript 跨域的几种情况](http://yijiebuyi.com/blog/03fc0af64ac69cb90ff8b81749a6bee4.html)
* [sails modules 模型自带的方法介绍](http://yijiebuyi.com/blog/604c2bfc97fea1a0b1d70b0ebb8edab9.html)
* [[投稿] 北京 海淀区 招聘 高级 node.js 开发工程师](http://yijiebuyi.com/blog/21d0d89afeb88f64fac1894e298fc131.html)

入群讨论

群号: 178894305 点击加入: [攻城狮](http://shang.qq.com/wpa/qunwpa?idkey=2a1629dc2f48d8e27da9560aebd6ca237c916640dddcce11956a49c825518d05)

订阅 "一介布衣" 公众号



打开微信   
扫描左侧二维码   
或搜索公众号:   
yijiebuyicode

* [本站由 UCloud 提供云服务器支持](https://www.ucloud.cn/)

* [About](http://yijiebuyi.com/about.html)

* [站点地图](http://yijiebuyi.com/sitemap.xml)

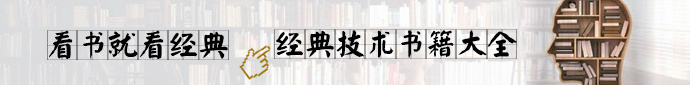
Copyright © 2015 [一介布衣](http://yijiebuyi.com/)

[http://eiv.baidu.com/hmt/icon/21.gif](http://tongji.baidu.com/hm-web/welcome/ico?s=89054854993e6ae0f2181a1a4078932f) [京ICP备14006909号-1](http://www.miitbeian.gov.cn/)

[回到顶部](http://yijiebuyi.com/blog/8f985d539566d0bf3b804df6be4e0c90.html)

* [首页](http://www.jobbole.com/)
* [资讯](http://top.jobbole.com/)
* [文章](http://blog.jobbole.com/)
* [资源](http://hao.jobbole.com/)
* [小组](http://group.jobbole.com/)
* [相亲](http://date.jobbole.com/)
* [登录](http://www.jobbole.com/login?redirect=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)
* [注册](http://www.jobbole.com/register)

[](http://blog.jobbole.com/)

[](http://blog.jobbole.com/106093/)

* [**首页**](http://blog.jobbole.com/)
* [**最新文章**](http://blog.jobbole.com/all-posts/)
* [**IT 职场**](http://blog.jobbole.com/category/career/)
* [**前端**](http://web.jobbole.com/)
* [**后端**](http://blog.jobbole.com/87700/)
* [**移动端**](http://blog.jobbole.com/87700/)
* [**数据库**](http://blog.jobbole.com/tag/database/)
* [**运维**](http://blog.jobbole.com/87700/)
* [**其他技术**](http://blog.jobbole.com/87700/)



[伯乐在线](http://www.jobbole.com/) > [首页](http://blog.jobbole.com/) > [所有文章](http://blog.jobbole.com/all-posts/) > [IT技术](http://blog.jobbole.com/category/it-tech/) > 如何在 Git 里撤销(几乎)任何操作

# 如何在 Git 里撤销(几乎)任何操作

2015/06/23 · [IT技术](http://blog.jobbole.com/category/it-tech/) · [2 评论](http://blog.jobbole.com/87700/#article-comment)· [Git](http://blog.jobbole.com/tag/git/), [Github](http://blog.jobbole.com/tag/github/)

分享到：**46**

* [**React.js入门与实战 开发适配PC及移动端新闻头条平台**](http://coding.imooc.com/class/83.html?from=jobboleblog)
* [**6小时用 jQuery 实现小应用**](http://coding.imooc.com/class/72.html?from=jobboleblog)
* [**Android专项测试-Python篇10年测试经验讲师**](http://coding.imooc.com/class/53.html?from=jobboleblog)
* [**Laravel和 AngularJS开发全栈知乎**](http://coding.imooc.com/class/65.html?from=jobboleblog)

本文由 [伯乐在线](http://blog.jobbole.com/) - [老码农](http://www.jobbole.com/members/laomanong) 翻译。未经许可，禁止转载！  
英文出处：[GitHub 官博文章](https://github.com/blog/2019-how-to-undo-almost-anything-with-git)。欢迎加入[翻译组](http://group.jobbole.com/category/feedback/trans-team/)。

任何版本控制系统的一个最有的用特性就是“撤销 (undo)”你的错误操作的能力。在 Git 里，“撤销” 蕴含了不少略有差别的功能。

当你进行一次新的提交的时候，Git 会保存你代码库在那个特定时间点的快照；之后，你可以利用 Git 返回到你的项目的一个早期版本。

在本篇博文里，我会讲解某些你需要“撤销”已做出的修改的常见场景，以及利用 Git 进行这些操作的最佳方法。

## 撤销一个“已公开”的改变

**场景:** 你已经执行了 git push, 把你的修改发送到了 GitHub，现在你意识到这些 commit 的其中一个是有问题的，你需要撤销那一个 commit.

**方法:** git revert <SHA>

**原理:** git revert 会产生一个新的 commit，它和指定 SHA 对应的 commit 是相反的（或者说是反转的）。如果原先的 commit 是“物质”，新的 commit 就是“反物质” — 任何从原先的 commit 里删除的内容会在新的 commit 里被加回去，任何在原先的 commit 里加入的内容会在新的 commit  里被删除。

这是 Git 最安全、最基本的撤销场景，因为它并不会改变历史 — 所以你现在可以  git push 新的“反转” commit 来抵消你错误提交的 commit。

## 修正最后一个 commit 消息

**场景:** 你在最后一条 commit 消息里有个笔误，已经执行了 git commit -m "Fxies bug #42"，但在 git push 之前你意识到消息应该是 “Fixes bug #42″。

**方法:** git commit --amend 或 git commit --amend -m "Fixes bug #42"

**原理:** git commit --amend 会用一个新的 commit 更新并替换最近的 commit ，这个新的 commit 会把任何修改内容和上一个 commit 的内容结合起来。如果当前没有提出任何修改，这个操作就只会把上次的 commit 消息重写一遍。

## 撤销“本地的”修改

**场景:** 一只猫从键盘上走过，无意中保存了修改，然后破坏了编辑器。不过，你还没有 commit 这些修改。你想要恢复被修改文件里的所有内容 — 就像上次 commit 的时候一模一样。

**方法:** git checkout -- <bad filename>

**原理:** git checkout 会把工作目录里的文件修改到 Git 之前记录的某个状态。你可以提供一个你想返回的分支名或特定 SHA ，或者在缺省情况下，Git 会认为你希望 checkout 的是 HEAD，当前 checkout 分支的最后一次 commit。

记住：你用这种方法“撤销”的任何修改**真的**会完全消失。因为它们从来没有被提交过，所以之后 Git 也无法帮助我们恢复它们。你要确保自己了解你在这个操作里扔掉的东西是什么！（也许可以先利用 git diff 确认一下）

## 重置“本地的”修改

**场景:** 你在本地提交了一些东西（还没有 push），但是所有这些东西都很糟糕，你希望撤销前面的三次提交 — 就像它们从来没有发生过一样。

**方法:** git reset <last good SHA> 或 git reset --hard <last good SHA>

**原理:** git reset 会把你的代码库历史返回到指定的 SHA 状态。 这样就像是这些提交从来没有发生过。缺省情况下， git reset 会保留工作目录。这样，提交是没有了，但是修改内容还在磁盘上。这是一种安全的选择，但通常我们会希望一步就“撤销”提交以及修改内容 — 这就是 --hard 选项的功能。

## 在撤销“本地修改”之后再恢复

**场景:** 你提交了几个 commit，然后用 git reset --hard 撤销了这些修改（见上一段），接着你又意识到：你希望还原这些修改！

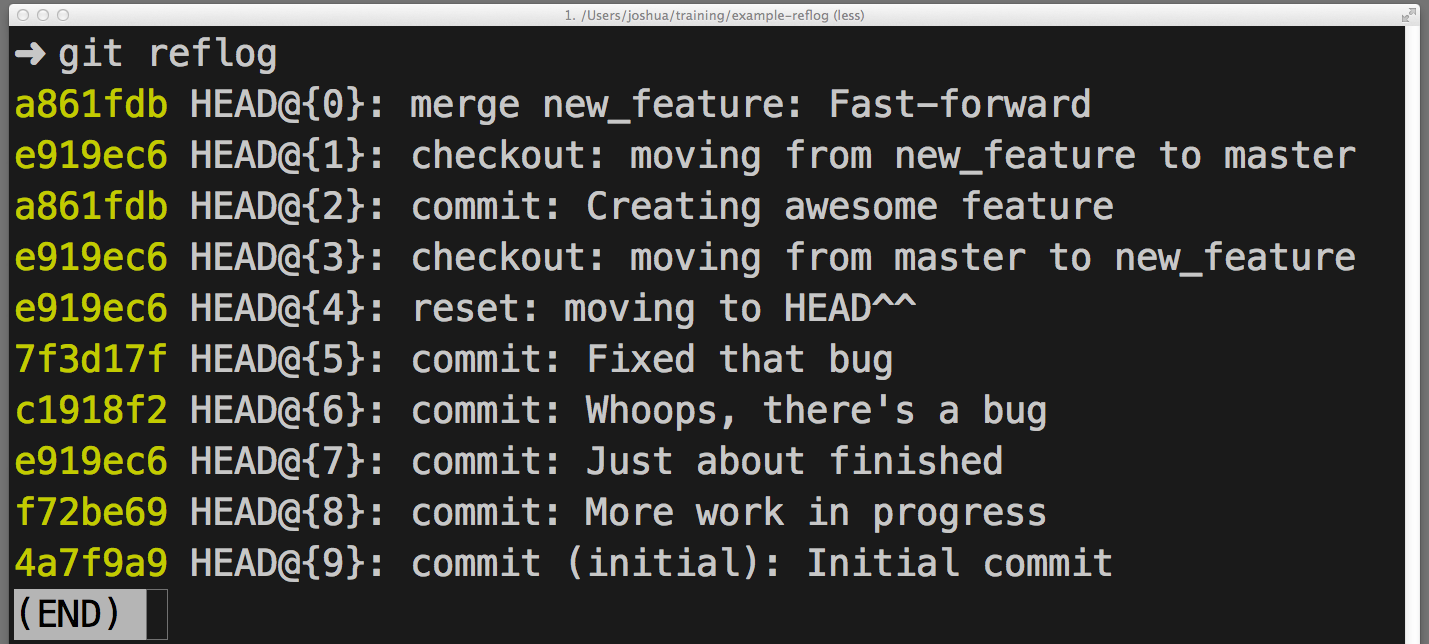
**方法:** git reflog 和 git reset 或 git checkout

**原理:** git reflog 对于恢复项目历史是一个超棒的资源。你可以恢复**几乎**任何东西 — 任何你 commit 过的东西 — 只要通过 reflog。

你可能已经熟悉了 git log 命令，它会显示 commit 的列表。 git reflog 也是类似的，不过它显示的是一个 HEAD 发生改变的时间列表.

一些注意事项：

* 它涉及的只是 HEAD 的改变。在你切换分支、用 git commit 进行提交、以及用 git reset 撤销 commit 时，HEAD 会改变，但当你用  git checkout -- <bad filename> 撤销时（正如我们在前面讲到的情况），HEAD 并不会改变 — 如前所述，这些修改从来没有被提交过，因此 reflog 也无法帮助我们恢复它们。
* git reflog 不会永远保持。Git 会定期清理那些 “用不到的” 对象。不要指望几个月前的提交还一直躺在那里。
* 你的 reflog 就是你的，只是你的。你不能用 git reflog 来恢复另一个开发者没有 push 过的 commit。

[](https://cloud.githubusercontent.com/assets/2077/6953866/f6b9f054-d891-11e4-8c53-838eff9f40ae.png)

那么…你怎么利用 reflog 来“恢复”之前“撤销”的 commit 呢？它取决于你想做到的到底是什么：

* 如果你希望准确地恢复项目的历史到某个时间点，用 git reset --hard <SHA>
* 如果你希望重建工作目录里的一个或多个文件，让它们恢复到某个时间点的状态，用 git checkout <SHA> -- <filename>
* 如果你希望把这些 commit 里的某一个重新提交到你的代码库里，用 git cherry-pick <SHA>

## 利用分支的另一种做法

**场景:** 你进行了一些提交，然后意识到你开始 check out 的是 master 分支。你希望这些提交进到另一个特性（feature）分支里。

**方法:** git branch feature, git reset --hard origin/master, and git checkout feature

**原理:** 你可能习惯了用 git checkout -b <name> 创建新的分支 — 这是创建新分支并马上 check out 的流行捷径 — 但是你不希望马上切换分支。这里， git branch feature 创建一个叫做 feature 的新分支并指向你最近的 commit，但还是让你 check out 在 master 分支上。

下一步，在提交任何新的 commit 之前，用 git reset --hard 把 master 分支倒回 origin/master 。不过别担心，那些 commit 还在 feature 分支里。

最后，用 git checkout 切换到新的 feature 分支，并且让你最近所有的工作成果都完好无损。

## 及时分支，省去繁琐

**场景:** 你在 master 分支的基础上创建了 feature 分支，但 master 分支已经滞后于 origin/master 很多。现在 master 分支已经和 origin/master 同步，你希望在 feature 上的提交是从现在开始，而不是也从滞后很多的地方开始。

**方法:** git checkout feature 和 git rebase master

**原理:** 要达到这个效果，你本来可以通过 git reset (不加 --hard, 这样可以在磁盘上保留修改) 和 git checkout -b <new branch name> 然后再重新提交修改，不过这样做的话，你就会失去提交历史。我们有更好的办法。

git rebase master 会做如下的事情：

* 首先它会找到你当前 check out 的分支和 master 分支的共同祖先。
* 然后它 reset 当前  check out 的分支到那个共同祖先，在一个临时保存区存放所有之前的提交。
* 然后它把当前 check out 的分支提到 master 的末尾部分，并从临时保存区重新把存放的 commit 提交到 master 分支的最后一个 commit 之后。

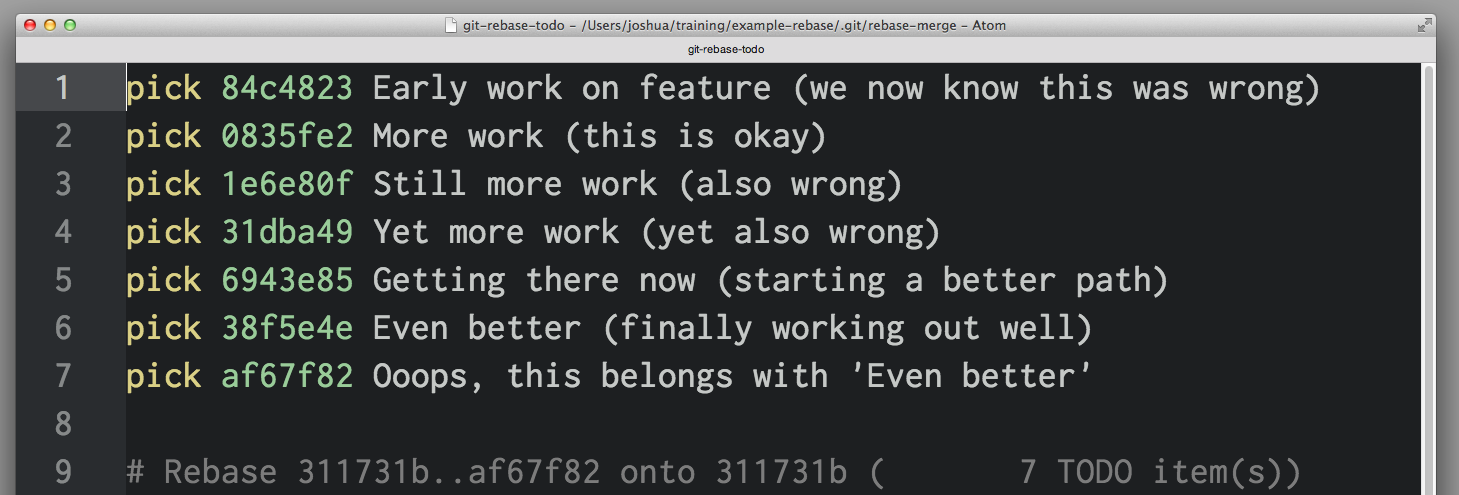
## 大量的撤销/恢复

**场景:** 你向某个方向开始实现一个特性，但是半路你意识到另一个方案更好。你已经进行了十几次提交，但你现在只需要其中的一部分。你希望其他不需要的提交统统消失。

**方法:** git rebase -i <earlier SHA>

**原理:** -i 参数让 rebase 进入“交互模式”。它开始类似于前面讨论的 rebase，但在重新进行任何提交之前，它会暂停下来并允许你详细地修改每个提交。

rebase -i 会打开你的缺省文本编辑器，里面列出候选的提交。如下所示：

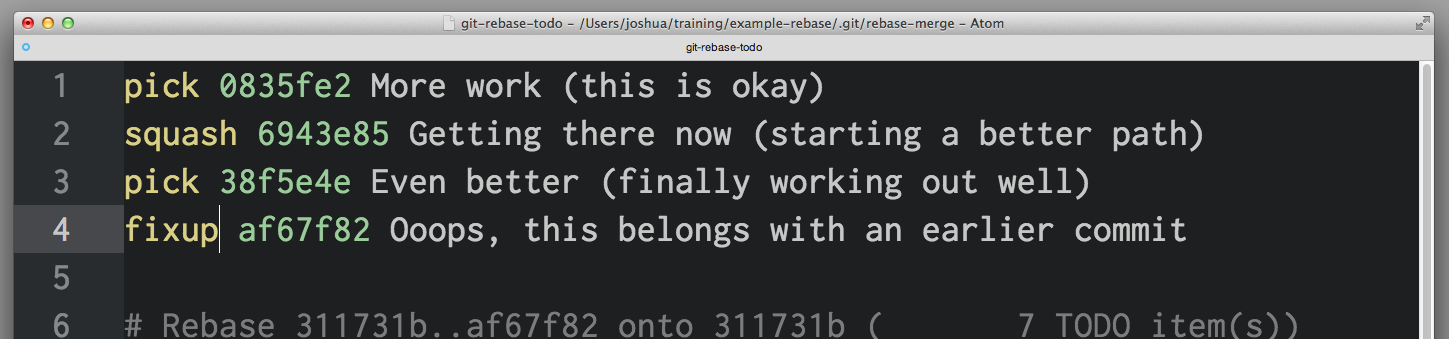
[](https://cloud.githubusercontent.com/assets/2077/6953863/f6b1ab88-d891-11e4-97c1-e0630ac74e74.png)

前面两列是键：第一个是选定的命令，对应第二列里的 SHA 确定的 commit。缺省情况下， rebase -i  假定每个 commit 都要通过  pick 命令被运用。

要丢弃一个 commit，只要在编辑器里删除那一行就行了。如果你不再需要项目里的那几个错误的提交，你可以删除上例中的1、3、4行。

如果你需要保留 commit 的内容，而是对 commit 消息进行编辑，你可以使用 reword 命令。 把第一列里的 pick 替换为 reword (或者直接用 r)。有人会觉得在这里直接重写 commit 消息就行了，但是这样不管用 —rebase -i 会忽略 SHA 列前面的任何东西。它后面的文本只是用来帮助我们记住 0835fe2 是干啥的。当你完成 rebase -i 的操作之后，你会被提示输入需要编写的任何 commit 消息。

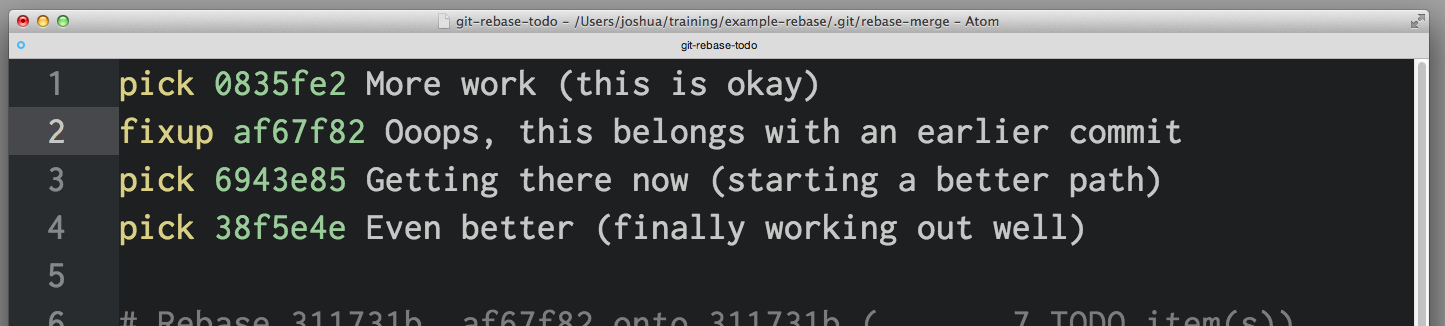
如果你需要把两个 commit 合并到一起，你可以使用 squash 或 fixup 命令，如下所示：

[](https://cloud.githubusercontent.com/assets/2077/6953865/f6b605ca-d891-11e4-98cf-d567ca9f4edc.png)

squash 和 fixup 会“向上”合并 — 带有这两个命令的 commit 会被合并到它的前一个 commit 里。在这个例子里， 0835fe2 和 6943e85 会被合并成一个 commit， 38f5e4e 和 af67f82 会被合并成另一个。

如果你选择了 squash， Git 会提示我们给新合并的 commit 一个新的 commit 消息； fixup 则会把合并清单里第一个 commit 的消息直接给新合并的 commit 。 这里，你知道 af67f82 是一个“完了完了….” 的 commit，所以你会留着 38f5e4e as的 commit 消息，但你会给合并了 0835fe2 和 6943e85 的新 commit 编写一个新的消息。

在你保存并退出编辑器的时候，Git 会按从顶部到底部的顺序运用你的 commit。你可以通过在保存前修改 commit 顺序来改变运用的顺序。如果你愿意，你也可以通过如下安排把 af67f82 和 0835fe2 合并到一起：

[](https://cloud.githubusercontent.com/assets/2077/6953864/f6b4a9d2-d891-11e4-9ac9-10039c031d06.png)

## 修复更早期的 commit

**场景:** 你在一个更早期的 commit 里忘记了加入一个文件，如果更早的 commit 能包含这个忘记的文件就太棒了。你还没有 push，但这个 commit 不是最近的，所以你没法用 commit --amend.

**方法:** git commit --squash <SHA of the earlier commit> 和 git rebase --autosquash -i <even earlier SHA>

**原理:** git commit --squash 会创建一个新的 commit ，它带有一个 commit 消息，类似于 squash! Earlier commit。 (你也可以手工创建一个带有类似 commit 消息的 commit，但是 commit --squash 可以帮你省下输入的工作。)

如果你不想被提示为新合并的 commit 输入一条新的 commit 消息，你也可以利用 git commit --fixup 。在这个情况下，你很可能会用commit --fixup ，因为你只是希望在 rebase 的时候使用早期 commit 的 commit 消息。

rebase --autosquash -i  会激活一个交互式的 rebase 编辑器，但是编辑器打开的时候，在 commit 清单里任何 squash! 和 fixup! 的 commit 都已经配对到目标 commit 上了，如下所示：

[](https://cloud.githubusercontent.com/assets/2077/6953862/f6a7a1d8-d891-11e4-8784-c32262ff54da.png)

在使用 --squash 和 --fixup 的时候，你可能不记得想要修正的 commit 的 SHA 了— 只记得它是前面第 1 个或第 5 个 commit。你会发现 Git 的 ^ 和 ~ 操作符特别好用。HEAD^ 是 HEAD 的前一个 commit。 HEAD~4 是 HEAD 往前第 4 个 – 或者一起算，倒数第 5 个 commit。

## 停止追踪一个文件

**场景:** 你偶然把 application.log 加到代码库里了，现在每次你运行应用，Git 都会报告在 application.log 里有未提交的修改。你把 \*.login 放到了 .gitignore 文件里，可文件还是在代码库里 — 你怎么才能告诉 Git “撤销” 对这个文件的追踪呢？

**方法:** git rm --cached application.log

**原理:** 虽然 .gitignore 会阻止 Git 追踪文件的修改，甚至不关注文件是否存在，但这只是针对那些以前从来没有追踪过的文件。一旦有个文件被加入并提交了，Git 就会持续关注该文件的改变。类似地，如果你利用 git add -f 来强制或覆盖了 .gitignore， Git 还会持续追踪改变的情况。之后你就不必用-f  来添加这个文件了。

如果你希望从 Git 的追踪对象中删除那个本应忽略的文件， git rm --cached 会从追踪对象中删除它，但让文件在磁盘上保持原封不动。因为现在它已经被忽略了，你在  git status 里就不会再看见这个文件，也不会再偶然提交该文件的修改了。

这就是如何在 Git 里撤销任何操作的方法。要了解更多关于本文中用到的 Git 命令，请查看下面的有关文档：

* [checkout](http://git-scm.com/docs/git-checkout)
* [commit](http://git-scm.com/docs/git-commit)
* [rebase](http://git-scm.com/docs/git-rebase)
* [reflog](http://git-scm.com/docs/git-reflog)
* [reset](http://git-scm.com/docs/git-reset)
* [revert](http://git-scm.com/docs/git-revert)
* [rm](http://git-scm.com/docs/git-rm)

**打赏支持我翻译更多好文章，谢谢！**

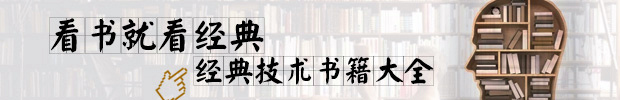
[打赏译者](http://blog.jobbole.com/87700/#rewardbox)

 2 赞  14 收藏 [2 评论](http://blog.jobbole.com/87700/#article-comment)

### 关于作者：[老码农](http://www.jobbole.com/members/laomanong)

[](http://www.jobbole.com/members/laomanong)

搞得定代码，罩得住娃；治得好跟腱，踢得了球。Hi，我是老码农，蜀黍有练过，小盆友们不要随便模仿喔。（新浪微博：@老码农的自留地）[个人主页](http://www.jobbole.com/members/laomanong) · [我的文章](http://blog.jobbole.com/author/laomanong/) · [122](http://www.jobbole.com/members/laomanong/reputation/) ·

[](http://blog.jobbole.com/106093/)

### 相关文章

* [研发团队 GIT 开发流程新人学习指南](http://blog.jobbole.com/110052/?utm_source=blog.jobbole.com&utm_medium=relatedPosts)
* [Git 最佳实践：分支管理](http://blog.jobbole.com/109466/?utm_source=blog.jobbole.com&utm_medium=relatedPosts) · [3](http://blog.jobbole.com/109466/#comments)
* [Git 最佳实践：commit msg](http://blog.jobbole.com/109197/?utm_source=blog.jobbole.com&utm_medium=relatedPosts)
* [SVN、GIT日常看我就够了](http://blog.jobbole.com/108834/?utm_source=blog.jobbole.com&utm_medium=relatedPosts)
* [Git 王者超神之路](http://blog.jobbole.com/108828/?utm_source=blog.jobbole.com&utm_medium=relatedPosts) · [4](http://blog.jobbole.com/108828/#comments)

### 可能感兴趣的话题

* [目前关于P2P穿透内网的技术资料各位大神有没有推荐](http://group.jobbole.com/29177/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic) · [1](http://group.jobbole.com/29177/#comments)
* [求助，滚动条滑至底部自动加载数据问题](http://group.jobbole.com/29175/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic) · [2](http://group.jobbole.com/29175/#comments)
* [网易2017内推笔试编：地牢逃脱](http://group.jobbole.com/29121/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic)
* [工作六年感觉遇到瓶颈](http://group.jobbole.com/29163/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic) · [4](http://group.jobbole.com/29163/#comments)
* [在技术领域，大家瓶颈期是咋度过的？](http://group.jobbole.com/29146/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic)
* [求助，c#软件开发专科学生，我想专接本计算机科学与技术，不知道值不值？有...](http://group.jobbole.com/29139/?utm_source=blog.jobbole.com&utm_medium=article-related-group-topic) · [7](http://group.jobbole.com/29139/#comments)

[登录后评论](http://www.jobbole.com/login?redirect=http://blog.jobbole.com/87700/) [新用户注册](http://www.jobbole.com/register)   
  
直接登录 [http://blog.jobbole.com/wp-content/plugins/wp-connect/images/login/btn_weixin.png](http://blog.jobbole.com/wp-content/plugins/wp-connect/login.php?go=weixin&redirect_url=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)[http://blog.jobbole.com/wp-content/plugins/wp-connect/images/login/btn_qzone.png](http://blog.jobbole.com/wp-content/plugins/wp-connect/login.php?go=qzone&redirect_url=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)[http://blog.jobbole.com/wp-content/plugins/wp-connect/images/login/btn_sina.png](http://blog.jobbole.com/wp-content/plugins/wp-connect/login.php?go=sina&redirect_url=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)[http://blog.jobbole.com/wp-content/plugins/wp-connect/images/login/btn_douban.png](http://blog.jobbole.com/wp-content/plugins/wp-connect/login.php?go=douban&redirect_url=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)[http://blog.jobbole.com/wp-content/plugins/wp-connect/images/login/btn_github.png](http://blog.jobbole.com/wp-content/plugins/wp-connect/login.php?go=github&redirect_url=http%3A%2F%2Fblog.jobbole.com%2F87700%2F)

##### 最新评论

* [](http://www.jobbole.com/members/51chenyiquan)

[面拾人生](http://www.jobbole.com/members/51chenyiquan) （ [1](http://www.jobbole.com/members/51chenyiquan/reputation/) ·   ） [2015/06/24](http://blog.jobbole.com/87700/#comment-152571)

好好学git了

 赞 [回复](http://blog.jobbole.com/87700/?replytocom=152571#respond)

* [](http://www.jobbole.com/members/%E6%88%91%E4%BB%AC%E5%9C%A8%E8%B7%AF%E4%B8%8A)

[我们在路上](http://www.jobbole.com/members/%E6%88%91%E4%BB%AC%E5%9C%A8%E8%B7%AF%E4%B8%8A)  [2016/12/21](http://blog.jobbole.com/87700/#comment-159575)

写的很不错，学习了

 赞 [回复](http://blog.jobbole.com/87700/?replytocom=159575#respond)

[](http://group.jobbole.com/22945/)

* [**本周热门文章**](http://blog.jobbole.com/87700/#tab1)
* [**本月热门文章**](http://blog.jobbole.com/87700/#tab2)
* [**热门标签**](http://blog.jobbole.com/87700/#tab3)

**0**[为什么我再也不想和 Google HR...](http://blog.jobbole.com/110340/)

**1**[编程界 12 位祖师爷级别的 “大牛”](http://blog.jobbole.com/110321/)

**2**[看过这两张图，就明白 Buffer 和 C...](http://blog.jobbole.com/109987/)

**3**[也许，这样理解HTTPS更容易](http://blog.jobbole.com/110354/)

**4**[硬盘写到一半时断电，文件系统里会发...](http://blog.jobbole.com/110310/)

**5**[HTTPS 为什么更安全，先看这些](http://blog.jobbole.com/110373/)

**6**[会编程的 AI + 会修 Bug 的...](http://blog.jobbole.com/110326/)

**7**[生活中该最应该了解哪些重要的事？](http://blog.jobbole.com/110197/)

**8**[想写无Bug的安全代码？看防御性编程...](http://blog.jobbole.com/110123/)

**9**[WASD 何时成了游戏操作的主流？](http://blog.jobbole.com/110329/)

[](http://blog.jobbole.com/107308/)

### [业界热点资讯更多 »](http://top.jobbole.com/category/news/)

[](http://top.jobbole.com/36350/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)

[**美五角大楼准备开源代码库**](http://top.jobbole.com/36350/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)  
20 小时前 ·  6

[](http://top.jobbole.com/36354/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)

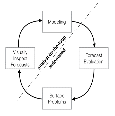
[**在美国入境被考算法，非洲程序员的亲身经历**](http://top.jobbole.com/36354/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)  
12 小时前 ·  5

[](http://top.jobbole.com/36351/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)

[**Google E2EMail 加密电子邮件代码宣告开源**](http://top.jobbole.com/36351/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)  
20 小时前 ·  4

[](http://top.jobbole.com/36343/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)

[**SHA1 碰撞攻击的第一位受害者：WebKit 版本控制系统**](http://top.jobbole.com/36343/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)  
1 天前 ·  8

[](http://top.jobbole.com/36347/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)

[**Facebook 开源大规模预测工具 Prophet：支持 Python...**](http://top.jobbole.com/36347/?utm_source=blog.jobbole.com&utm_medium=sidebar-top-news)  
1 天前 ·  4

[](http://blog.jobbole.com/84342/)

### [精选工具资源更多资源 »](http://hao.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources-more)

[](http://hao.jobbole.com/static_code_analysis_tool_list_opensource_lang/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)

[**静态代码分析工具清单：开源篇（各语言）**](http://hao.jobbole.com/static_code_analysis_tool_list_opensource_lang/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)  
[静态代码分析](http://hao.jobbole.com/category/cpp-lang/cpp-static-code-analysis/)

[](http://hao.jobbole.com/bugsnag/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)

[**Bugsnag：跨平台的错误监测**](http://hao.jobbole.com/bugsnag/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)  
[开发库](http://hao.jobbole.com/category/android/android-dev-tools/)

[](http://hao.jobbole.com/static_code_analysis_tool_list_opensource/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)

[**静态代码分析工具清单：开源篇（多语言）**](http://hao.jobbole.com/static_code_analysis_tool_list_opensource/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)  
[静态代码分析](http://hao.jobbole.com/category/cpp-lang/cpp-static-code-analysis/)

[](http://hao.jobbole.com/tryton/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)

[**Tryton：一个通用商务框架**](http://hao.jobbole.com/tryton/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)  
[杂项](http://hao.jobbole.com/category/python-tools/python-misc/)

[](http://hao.jobbole.com/nltk/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)

[**NLTK：一个先进的用来处理自然语言数据的Python程序。**](http://hao.jobbole.com/nltk/?utm_source=blog.jobbole.com&utm_medium=sidebar-resources)  
[自然语言处理](http://hao.jobbole.com/category/python-tools/python-natural-lang/) · [1](http://hao.jobbole.com/nltk/#comments)

[](http://blog.jobbole.com/99322/?utm_source=blog.jobbole.com&utm_medium=sidebarBanner&utm_content=2016.4.5)

### 关于伯乐在线博客

在这个信息爆炸的时代，人们已然被大量、快速并且简短的信息所包围。然而，我们相信：过多“快餐”式的阅读只会令人“虚胖”，缺乏实质的内涵。伯乐在线内容团队正试图以我们微薄的力量，把优秀的原创文章和译文分享给读者，为“快餐”添加一些“营养”元素。

**快速链接**  
[网站使用指南 »](http://blog.jobbole.com/87622/)  
[问题反馈与求助 »](http://group.jobbole.com/category/feedback/thanks)  
[加入我们 »](http://group.jobbole.com/22422/)  
[网站积分规则 »](http://group.jobbole.com/158/)  
[网站声望规则 »](http://group.jobbole.com/175/)

### 关注我们

新浪微博：[@伯乐在线官方微博](http://weibo.com/jobbole)  
RSS：[订阅地址](http://blog.jobbole.com/feed/)  
推荐微信号   


**合作联系**  
Email：[bd@Jobbole.com](mailto:bd@jobbole.com)  
QQ： 2302462408 （加好友请注明来意）

### 更多频道

[小组](http://group.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 好的话题、有启发的回复、值得信赖的圈子  
[头条](http://top.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 分享和发现有价值的内容与观点  
[相亲](http://date.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 为IT单身男女服务的征婚传播平台  
[资源](http://hao.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 优秀的工具资源导航  
[翻译](http://fanyi.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 翻译传播优秀的外文文章  
[文章](http://blog.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 国内外的精选文章  
[设计](http://design.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – UI,网页，交互和用户体验  
[iOS](http://ios.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 专注iOS技术分享  
[安卓](http://android.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 专注Android技术分享  
[前端](http://web.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – JavaScript, HTML5, CSS  
[Java](http://www.importnew.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 专注Java技术分享  
[Python](http://python.jobbole.com/?utm_source=blog.jobbole.com&utm_medium=bottomText) – 专注Python技术分享

© 2017 伯乐在线 [文章](http://blog.jobbole.com/)   [小组](http://group.jobbole.com/)   [相亲](http://date.jobbole.com/)   [加入我们](http://group.jobbole.com/22422/)   [反馈](http://group.jobbole.com/category/feedback/thanks/)

跳到底部

返回顶部