

Oh Shit, Git!?!



Zine made for
programmers —
recipes for gitting
out of a mess.
ADS VIA CARBON

用好 Git 很难: 很容易就犯错了, 然后想自己弥补犯下的错, 简直太难了。查阅 Git 文档简直就像是个“鸡生蛋 蛋生鸡”的问题, *你得知道你要的是啥*, 但如果我知道的话, 我还他妈查个毛文档啊!

所以接下来我会分享一些我遇到过的抓狂的经历, 然后用 *白话* 来说说我是如何解决的。

哎呦我去, 我刚才好像犯了个大错, 能不能给我台时光机啊!?!

```
git reflog
# 你将看到你在 git 上提交的所有改动记录被列
# 了出来, 而且囊括了所有的分支, 和已被删除的
# commit 哦!
# 每一条记录都有一个类似 HEAD@{index} 的索
# 引编号
# 找到在犯错前的那个提交记录的索引号, 然后执
# 行:
git reset HEAD@{index}
# 哈哈, 这就是你要的时光机!
```

你可以用这个方法来回那些你不小心删除的东西、恢复一些你对 repo 改动、恢复一次错误的 merge 操作、或者仅仅想退回到你的项目还能正常工作的那一时刻。我经常使用 reflog, 在此我要向那些提案添加这个功能的人们表示感谢, 太谢谢他们了!

哎呦我去, 我刚提交 commit 就发现还有一个小改动需要添加!

```
# 继续改动你的文件
git add . # 或者你可以添加指定的文件
git commit --amend --no-edit
# 你这次的改动会被添加进最近一次的 commit 中
# 警告: 千万别对公共的 commit 做这种操作
```

这经常发生在我提交了 commit 以后立马发现, 妈蛋, 我忘了在某个等号后面加空格了。当然, 你也可以提交一个新的 commit 然后利用 rebase -i 命令来合并它们, 但我觉得我的这种方式比你快 100 万倍。

警告: 你千万不要在已推送的公共分支上做这个 amend 的操作! 只能在你本地 commit 上做这种修改, 否则你会把事情搞砸的!

哎呦我去, 我要修改我刚刚 commit 提交的信息!

```
git commit --amend
# 按照提示修改信息就行啦
```

使用繁琐的提交信息格式

哎呦我去, 我不小心把本应在新分支上提交的东西提交到了 master!

```
# 基于当前 master 新建一个分支
git branch some-new-branch-name
# 在 master 上删除最近的那次 commit
git reset HEAD~ --hard
git checkout some-new-branch-name
# 只有在这个新分支上才有你最近的那次 commit 哦
```

注意: 如果你已将这个 commit 推送到了公共分支, 那这波操作就不起作用了。如果你在此之前做了些其他的操作, 那你可能需要使用 HEAD@{number-of-commits-back} 来替代 HEAD~。另外, 感谢很多人提出了这个我自己都不知道的超棒的解决方法, 谢谢大家!

哎呦我去, 我把这个 commit 提交错分支了!

```
# 撤回这次提交, 但保留改动的内容
git reset HEAD~ --soft
git stash
# 现在切到正确的那个分支去
git checkout name-of-the-correct-branch
git stash pop
git add . # 或者你可以添加指定的文件
git commit -m "your message here";
# 现在你的改动就在正确的分支上啦
```

很多人建议使用 cherry-pick 来解决这个问题, 其实两者都可以, 你只要选择自己喜欢的方式就行了。

```
git checkout name-of-the-correct-branch
# 抓取 master 分支上最新的那个 commit
git cherry-pick master
# 然后删掉 master 上的那个 commit
git checkout master
git reset HEAD~ --hard
```

哎呦我去, 我想用 diff 命令看下改动内容, 但啥都没看到?!

如果对文件做了改动, 但是通过 diff 命令却看不到, 那很可能是你执行过 add 命令将文件改动添加到了 暂存区 了。你需要添加下面这个参数。

```
git diff --staged
```

这些文件在这里 ㄟ(ˋ)ㄏ (是的, 我知道这是一个 feature 而不是 bug, 但它第一次发生在作为初学者的你身上时, 真的很让人困惑!)

哎呦我去, 我想撤回一个很早以前的 commit!

```
# 先找到你想撤销的那个 commit
git log
# 如果在第一屏没找到你需要的那个 commit, 可以用上下
# 箭头来滚动显示的内容, 找到了以后记下 commit 的
# hash 值
git revert [刚才记下的那个 hash 值]
# git 会自动修改文件来抵消那次 commit 的改动, 并创
# 建一个新的 commit, 你可以根据提示修改这个新 commit
# 的信息, 或者直接保存就完事了
```

这样你就不需要用回溯老版本然后再复制粘贴的方式了, 那样做太费事了! 如果你提交的某个 commit 导致了 bug, 你直接用 revert 命令来撤回那次提交就行啦。

你甚至可以恢复单个文件而不是一整个 commit! 但那是另一套 git 命令咯...

哎呦我去, 我想撤回某一个文件的改动!

```
# 找到文件改动前的那个 commit
git log
# 如果在第一屏没找到你需要的那个 commit, 可以用上下
# 箭头来滚动显示的内容, 找到了以后记下 commit 的
# hash 值
git checkout [刚才记下的那个 hash 值] -- path/to/file
# 改动前的文件会保存到你的暂存区
git commit -m "这样就不需要通过复制粘贴来撤回改动啦"
```

我花了好长好长, 真他妈长的时间才搞明白要这么做。说真的, 用 checkout -- 来撤回一个文件的改动, 这算什么鬼方式啊?! 向 Linus Torvalds 摆出抗议姿势:

去屎吧, 这些乱七八糟烦人的文件, 我放弃啦。(那些 untracked 的文件)

```
cd ..
sudo rm -r fucking-git-repo-dir
git clone https://some.github.url/fucking-git-repo-dir.git
cd fucking-git-repo-dir
```


感谢 Eric V. 提供了这个事例, 如果对 sudo 的使用有什么的质疑的话, 可以去向他提出。

不过说真的, 如果你的分支真的这么糟糕的话, 你应该使用 "git-approved" 的方法来重置你的 repo, 可以试试这么做, 但要注意这些操作都是破坏性的, 不可逆的!

```
# 获取远端库最新的状态
git fetch origin
git checkout master
git reset --hard origin/master
# 删除 untracked 的文件和目录
git clean -d --force
# 对每一个有问题的分支重复上述 checkout/reset/clean 操作
```

*免责声明: 本网站并不是一个详尽完整的参考文档。当然, 我知道还有很多其他更优雅的方法能达到相同的效果, 但我是通过不断的尝试、不停的吐槽最终解决了这些问题。接着我就有了这个奇妙的想法, 通过这种方式, 使用一些比较诙谐的脏话来分享我的经历和发现。希望你也觉得这很有意思, 但如果你不能接受的话请移步别处。

非常感谢每一位为网站添加新语言翻译的人, 你们真的太棒了! [Björn Söderqvist \(sv\)](#) · [Moritz Stückler \(de\)](#) · [Daniil Golubev \(ru\)](#) · [Łukasz Wójcik \(pl\)](#) · [fedemcmac \(it\)](#) · [Michel \(fr\)](#) · [Andriy Sultanov \(ua\)](#) · [Meiko Hori \(ja\)](#) · [Alex Tzimas \(gr\)](#) · [Martijn ten Heuvel \(nl\)](#) · [Elad Leevy \(he\)](#) · [Franco Fantini \(es\)](#) · [Catalina Focsa \(ro\)](#) · [Davi Alexandre \(pt_BR\)](#) · [Nemanja Vasić \(sr\)](#) · [Tao Jiayuan \(zh\)](#) · [Eduard Tomek \(cs\)](#) · [Ricky Gultom \(id\)](#) · [Khaja Md Sher F Alam \(bn\)](#) · [Rahul Dahal \(ne\)](#) · [Taha Paksu \(tr\)](#) · [Kitt Tientanopajai \(th\)](#) · [Gyeongjae Choi \(ko\)](#) . With additional help from [Iain Murray](#) · [Frank Taillandier](#) · [David Fyffe](#) · [Lucas Larson](#) · [Artem Vorotnikov](#)

如果你也想为网站添加一个新语言, 可以在这里提交 PR  [GitHub](#)