[译] 在Git中如何撤销上一次的commit(s)?


可木Changer (/u/d6e83ee69161) + 关注
2017.02.12 14:16* 字数 1145 阅读 5134 评论 0 喜欢 2
(/u/d6e83ee69161)

http://stackoverflow.com/questions/927358/how-to-undo-last-commits-in-git (https://link.jianshu.com?t=http://stackoverflow.com/questions/927358/how-to-undo-last-commits-in-git)

问题:

我在Git中提交了错误的文件。我该如何撤销那一次commit呢?

高票答案1:

```
$ git commit -m "Something terribly misguided"              (1)
$ git reset HEAD~                                           (2)
<< edit files as necessary >>                               (3)
$ git add ...                                               (4)
$ git commit -c ORIG_HEAD                                   (5)
```

1. 这是你想撤销的那步操作
2. 这一步不会让你的工作区（你磁盘上文件的状态）发生改变，但是会撤销上次的commit，并且将你第一步commit的那些改动从暂存区移除（所以当你再次使用 `git status` 查看状态的时候，它们会出现在'Changes not staged for commit'这句话的下面。提交他们之前，你需要再把它们add一次）。如果你*只想*为之前的commit增加更多的改动，或者改变之前的提交信息，你应该使用 `get reset --soft HEAD~`，和 `git rest HEAD~` 有些类似，但是会将你的改动保留在暂存区内。
3. 纠正工作区的文件。
4. 使用 `git add` 将你想提交的文件添加进去
5. 提交这些变动，并使用旧的commit信息。 `reset` 命令会将老的head复制到 `.git/ORIG_HEAD` 中保存； `commit` 和 `-c ORIG_HEAD` 一起使用将会打开一个编辑器，里面包含着旧的提交所产生的日志信息，你可以对它们进行编辑。如果你不需要编辑的话，可以使用 `-C` 选项。

提示：假如你只是提交信息弄错了，你不需要重置到一个早期的提交点上。更简单的方式是使用 `git reset` （将那些改变移出暂存区），然后再使用 `git commit --amend`，这将会打开你的默认的提交信息编辑器，然后将默认设定为最近一次的提交信息。

但是，如果此时你在索引中添加了任何新的改动，使用 `commit --amend` 将会带着它们一起添加到上一次的提交中去。

高票答案2:

如果你不清楚它是如何工作的，撤销一个commit是有点吓人。但如果你理解了，实际上一点都不难。

假如你现在是这种情况，C是你的HEAD，(F)是你当前文件的状态。

```
   (F)
A-B-C
    ↑
  master
```

你想要除掉C，并且再也不想见到它。那么你这么做：

```
git reset --hard HEAD~1
```

执行完后的结果就是：

```
 (F)
A-B
  ↑
master
```

现在B是HEAD了。因为你使用了 `--hard` ，所以你的文件被重置到B的状态了。

额，但假设C并没有那么糟糕，只是略微有点问题。你想要**撤销提交，但保留你的修改**，然后做些小调整，再提交一遍。那么我们再来看看如何做，当前情况如下：

```
   (F)
A-B-C
    ↑
  master
```

你可以这么做，不用 `--hard`：

```
git reset HEAD~1
```

结果变成了：

```
   (F)
A-B-C
  ↑
master
```

在任何情况下，HEAD都是指向最近一次提交的一个指针。当你使用 `git reset HEAD~1` 命令时，你告诉Git将HEAD指针往前回退一次，到上一个commit上。但是（除非你使用 `--hard` 参数）你仍然没有改变文件。所以现在 `git status` 会告诉你之前提交到C的那些变化依然存在。你什么都没有丢失。

假如你想要最轻微的改动，你甚至可以**撤销你的提交，但保留你的文件和索引**：

```
git reset --soft HEAD~1
```

这不仅仅保持你的文件不动，也没有改变索引。当你使用 `git status` 的时候，你将会看到索引里没什么变化，和之前一样。实际上，在这次命令之后，你可以再次使用 `git commit` 命令，然后效果就和你之前的那次提交一样了。

再说一件事儿：**假设你在例1中销毁了一个commit**，但最后发现仍然需要它？很倒霉，是不是？

没那么严重，仍然有办法将它找回来。输入 `git reflog` ，然后你会看到一个提交信息列表，这里保存着所有之前的提交SHA（译者注：一种Hash算法）值。找到你销毁的commit，然后这么干：

```
git checkout -b someNewBranchName shaYouDestroyed
```

好了，现在你成功复活了那个提交。commits不会真正在Git中被销毁（大概90天），所以你通常能够回退并且拯救你之前除掉的那一个commit。

小礼物走一走，来简书关注我

赞赏支持

翻译 (/nb/9566903)

可木Changer (/u/d6e83ee69161) ♂
写了 41022 字，被 72 人关注，获得了 73 个喜欢
(/u/d6e83ee69161)

+关注

喜欢 | 2

更多分享

(http://cwb.assets.jianshu.io/notes/images/9110397

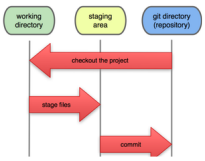
被以下专题收入，发现更多相似内容

gi Git (/c/70df798acc13?utm_source=desktop&utm_medium=notes-included-collection)

(/p/fe76f2890a14?utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)

Git全面教程 (/p/fe76f2890a14?utm_campaign=maleskine&utm_content…

Git全面教程 简介 Git分布式版本管理系统。 Linus在1991年创建了开源的Linux，但是一直没有一个合适的版本管理工具，在2002年以前，世界各地的…



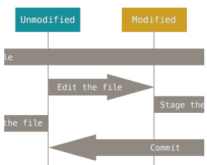
关玮琳linSir (/u/b51475b6a136?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

《Git权威指南》读书笔记 (Part 1) (/p/253e15e324cd?utm_campaign=ma…

本系列为《Git权威指南》的读书笔记，分为两个部分：Part 1 涵盖了书中第 1~3 篇共 20 章的内容，Part 2 为剩余部分。 git format-patch 命令 将从 v1 开始的历次提交逐一导出为补丁文件： $ git format-patch…

yestyle (/u/865b8d0a680e?utm_campaign=maleskine&utm_content=user&utm_medium=seo_notes&utm_source=recommendation)

(/p/cd1430161149?



utm_campaign=maleskine&utm_content=note&utm_medium=seo_notes&utm_source=recommendation)