**Git后悔药 - git commit --amend 命令**

这是我们Git中的第三种后悔药。

**1、git commit --amend 命令说明**

git commit --amend 命令：也叫追加提交，它可以在不增加一个新的commit的情况下，将新修改的代码追加到前一次的commit中。

使用该命令的好处，是避免许多无用的提交。

**说明：**

* 如果自上次提交以来，你还未做任何修改（例如，在上次提交后马上执行了此命令），那么快照会保持不变（不产生新的commit），而修改的只是提交信息。
* 或者你提交后发现忘记了暂存某些需要的修改，可以将该文件添加到暂存区，然后在使用git commit --amend 命令进行追加提交。

**常用参数**：

* --no-edit：使用选定的提交消息而无需启动编辑器。（追加提交，且不修改message信息。）
* --reset-author：加载提交用户的信息。
* 比如你从新配置本地用户的信息，然后使用该参数可以重新加载新用户的信息，并且更新在追加提交中。
* --author=：可修改Git 的 author 和 email 信息。
* 示例：git commit --amend --author='xxx <xxxx@xxx.xxx>'

**2、使用场景**

**（1）场景一**

**场景描述：**

开发代码已提交到本地版本库中，提交后发现这次提交的代码有问题，或者漏提交了一些文件，此时希望达到以下目的：

1. 修改有问题的代码。
2. 补足漏提交的文件。（一般是新增的文件没有git add ）
3. 把以上2点相关的代码，和前一次提交的代码合并成一个提交。
4. 给合并后的提交添加新的注释。

即：有时你提交过代码之后，发现一个地方改错了，你下次提交时不想保留上一次的记录，或者你上一次的commit message的描述有误，这时候你可以使用 git commit --amend 命令来解决。

**解决步骤：**

1. 修改问题代码。
2. 执行git add .命令，把漏提交的文件加入暂存区。
3. 执行git commit --amend -m "这里填写提交的注释"命令进行提交。

**示例：修改最新一次提交的message**

有的时候，我们刚刚进行一次提交，突然发现该提交的描述（message）不是很准确。所以需要对该提交的message进行修改（最近一次的commit）。

1）首先查看当前分支的历史提交情况

|  |
| --- |
| Bash # 1.查看版本库历史提交 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/amend\_test (master) $ git log --oneline bd4fd48 (HEAD -> master) 第3次提交，新增内容：git commit --amend v3 8b3a071 第2次提交，新增内容：git commit --amend v2 d3e2d8c 第1次提交，新增readme.txt文件  # 2.查看可回退历史提交信息。 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/amend\_test (master) $ git reflog bd4fd48 (HEAD -> master) HEAD@{0}: commit: 第3次提交，新增内容：git commit --amend v3 8b3a071 HEAD@{1}: commit: 第2次提交，新增内容：git commit --amend v2 d3e2d8c HEAD@{2}: commit (initial): 第1次提交，新增readme.txt文件 |

2）修改最新一次历史提交记录

最新一次历史提交记录，也就是第3次提交。

|  |
| --- |
| Bash L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/amend\_test (master) $ git commit --amend -m "我是第3次提交，重新修改的message" [master 69a9841] 我是第3次提交，重新修改的message  Date: Tue Apr 20 17:41:44 2021 +0800  1 file changed, 1 insertion(+) |

|  |
| --- |
| 提示：如果此时有需要新增或改动的文件，可以先把文件添加都暂存区，然后随着git commit --amend命令，一起追加提交。 |

3）再次查看当前分支的历史提交情况

|  |
| --- |
| Bash # 1.查看版本库历史提交 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/amend\_test (master) $ git log --oneline 69a9841 (HEAD -> master) 我是第3次提交，重新修改的message 8b3a071 第2次提交，新增内容：git commit --amend v2 d3e2d8c 第1次提交，新增readme.txt文件   # 2.查看可回退历史提交信息。 L@DESKTOP-T2AI2SU MINGW64 /j/git-repository/amend\_test (master) $ git reflog 69a9841 (HEAD -> master) HEAD@{0}: commit (amend): 我是第3次提交，重新修改的message bd4fd48 HEAD@{1}: commit: 第3次提交，新增内容：git commit --amend v3 8b3a071 HEAD@{2}: commit: 第2次提交，新增内容：git commit --amend v2 d3e2d8c HEAD@{3}: commit (initial): 第1次提交，新增readme.txt文件 |

我们可以看到：

1. 第三次提交的massage改变了。
2. 看到第三次提交的commit-id由bd4fd48变成了69a9841
3. 我们可以看到使用git commit --amend命令，表面上没有生成新的提交，只是进行了内容的追加。但实际上通过git reflog命令查看历史提交记录，我们还是看到了，其实是新生成了一个commit。

|  |
| --- |
| 说明：   1. 关于git commit --amend命令原理，往下看。 2. 关于如何修改更早提价的message，看后面的git rebase文章。 |

**（2）场景二**

**场景描述：**

假设新接到需求，需要基于master分支拉取一个feature分支，且这个feature分支只有你自己使用（这一点极其重要），由于开发周期较长，你不想每一次提交都产生一个新的commit，而是想每一次commit都修改前一次提交（一直追加提交）。

这样做的好处是，等到你的feature分支提测时，就只有一个干净的commit，没有乱七八糟的提交历史，你只要把这一个commit合并到master里就好了 。

**操作步骤：**

在feature分支上：

* 第1次提交代码时，使用命令git commit -a -m "第1次提交的注释"进行提交。
* 第2次以及只后的提交，使用命令git commit --amend -m "这里填写提交的注释"进行提交。 这样整个分支可以只有一个commit。

**示例：**

不推荐这样操作，多次提交可以使你选择撤销的余地比较多，简化提交次数是比较好的想法，但更推荐使用git rebase命令进行操作。

这里就不演示了。

**3、git commit --amend 命令原理**

git commit --amend这个命令给人的感觉，是用新提交的数据，追加到了前一个commit上。

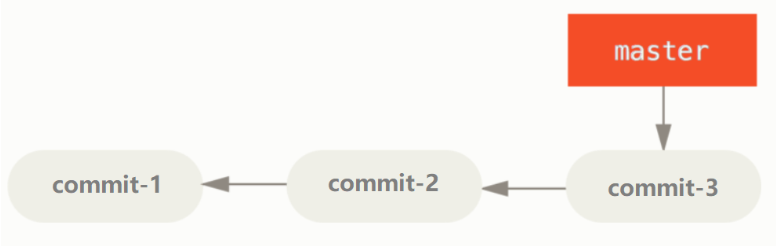
其实git commit --amend命令底层原理如下：

前提：你把遗漏提交的文件，或者修改的文件先提交到暂存区。

1. git reset --soft HEAD^：回退一个版本，且工作区和暂存区中的内容不会退。
2. git commit -c ORIG\_HEAD：提交操作。在根据此时的暂存区生成一个新的提交对象，代替原来提交的位置。
3. 说明：
4. -C <commit>，全称--reuse-message=<commit>：获取现有的提交对象，并在创建提交时重用现有提交对象的日志消息和作者信息（包括时间戳）。
5. -c <commit>，全称--reedit-message=<commit>：与-C类似，但是-c会调用编辑器，以便用户可以进一步编辑提交消息。

看如下图示：

假设master分支上的提交历史如下图:



运行git commit --amend命令，过程如下:



就相当于新建一个分支，把commit-3提交隐藏在历史提交记录的长河里。但使用reflog命令还是可以查看到的。

|  |
| --- |
| 参考：   * [https://blog.csdn.net/zxr97570912/article/details/108085873](https://links.jianshu.com/go?to=https%3A%2F%2Fblog.csdn.net%2Fzxr97570912%2Farticle%2Fdetails%2F108085873) * [https://www.cnblogs.com/chaoguo1234/p/5331586.html](https://links.jianshu.com/go?to=https%3A%2F%2Fwww.cnblogs.com%2Fchaoguo1234%2Fp%2F5331586.html) |

作者：繁华似锦Fighting 链接：https://www.jianshu.com/p/f015b97bda53 来源：简书 著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。