撤消操作

任何时候，你都有可能需要撤消刚才所做的某些操作。接下来，我们会介绍一些基本的撤消操作相关的命令。请注意，有些撤销操作是不可逆的，所以请务必谨慎小心，一旦失误，就有可能丢失部分工作成果。

修改最后一次提交

有时候我们提交完了才发现漏掉了几个文件没有加，或者提交信息写错了。想要撤消刚才的提交操作，可以使用 --amend 选项重新提交：

$ git commit --amend

此命令将使用当前的暂存区域快照提交。如果刚才提交完没有作任何改动，直接运行此命令的话，相当于有机会重新编辑提交说明，但将要提交的文件快照和之前的一样。

启动文本编辑器后，会看到上次提交时的说明，编辑它确认没问题后保存退出，就会使用新的提交说明覆盖刚才失误的提交。

如果刚才提交时忘了暂存某些修改，可以先补上暂存操作，然后再运行 --amend 提交：

$ git commit -m 'initial commit'

$ git add forgotten\_file

$ git commit --amend

上面的三条命令最终只是产生一个提交，第二个提交命令修正了第一个的提交内容。

取消已经暂存的文件

接下来的两个小节将演示如何取消暂存区域中的文件，以及如何取消工作目录中已修改的文件。不用担心，查看文件状态的时候就提示了该如何撤消，所以不需要死记硬背。来看下面的例子，有两个修改过的文件，我们想要分开提交，但不小心用 git add . 全加到了暂存区域。该如何撤消暂存其中的一个文件呢？其实，git status 的命令输出已经告诉了我们该怎么做：

$ git add .

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.txt

modified: benchmarks.rb

就在 “Changes to be committed” 下面，括号中有提示，可以使用 git reset HEAD <file>... 的方式取消暂存。好吧，我们来试试取消暂存 benchmarks.rb 文件：

$ git reset HEAD benchmarks.rb

Unstaged changes after reset:

M benchmarks.rb

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.txt

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: benchmarks.rb

这条命令看起来有些古怪，先别管，能用就行。现在 benchmarks.rb 文件又回到了之前已修改未暂存的状态。

取消对文件的修改

如果觉得刚才对 benchmarks.rb 的修改完全没有必要，该如何取消修改，回到之前的状态（也就是修改之前的版本）呢？git status 同样提示了具体的撤消方法，接着上面的例子，现在未暂存区域看起来像这样：

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: benchmarks.rb

在第二个括号中，我们看到了抛弃文件修改的命令（至少在 Git 1.6.1 以及更高版本中会这样提示，如果你还在用老版本，我们强烈建议你升级，以获取最佳的用户体验），让我们试试看：

$ git checkout -- benchmarks.rb

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.txt

可以看到，该文件已经恢复到修改前的版本。你可能已经意识到了，这条命令有些危险，所有对文件的修改都没有了，因为我们刚刚把之前版本的文件复制过来重写了此文件。所以在用这条命令前，请务必确定真的不再需要保留刚才的修改。如果只是想回退版本，同时保留刚才的修改以便将来继续工作，可以用下章介绍的 stashing 和分支来处理，应该会更好些。

记住，任何已经提交到 Git 的都可以被恢复。即便在已经删除的分支中的提交，或者用 --amend 重新改写的提交，都可以被恢复（关于数据恢复的内容见第九章）。**所以，你可能失去的数据，仅限于没有提交过的，对 Git 来说它们就像从未存在过一样。**