一.修改最近一次提交

这是最常见的一种场景，往往刚刚提交后最容易发现问题。

方法一：用commit –amend

这种方法不仅可以修改commit message，也可以修改提交内容。这种方式在还没有推送到远端的情况下可以比较方便的保持原有的Change-Id，推荐使用（若已经推送到远端，Change-Id则会修改掉）。

```
#修改需要修改的地方（只是修改commit message就不用做）
git add .  #这一步如果只是修改commit message不用输入
git commit --amend
#输入修改后的commit message，保存
git push <remote> <branch> -f #若还没有推送到远端，不用输入
```

方法二：用reset后修改

这种方法与上面方法基本一致，也可以修改提交内容和commit message。这种方式在还没有推送到远端的情况下也可以比较方便的保持原有的Change-Id，（若已经推送到远端，Change-Id则会修改掉）。

```
git reset HEAD^
#修改需要修改的地方（只是修改commit message就不用做）
git add . #这一步如果只是修改commit message不用输入
git commit -m "new commit message" #或者git commit -c ORIG_HEAD
git push <remote> <branch> -f #若还没有推送到远端，不用输入
```

方法三：提交到了错误的分支上的处理

```
# 取消最新的提交，然后保留现场原状
git reset HEAD~ --soft
```

```
git stash
```

```
# 切换到正确的分支
git checkout name-of-the-correct-branch
git stash pop
git add .     # 或添加特定文件
git commit -m "你的提交说明"
```

```
# 现在你已经提交到正确的分支上了
```

遇到这种情况，很多人会说用cherry-pick(摘樱桃)，像下面这样。不过你自己看吧，哪个舒服用哪个。

```
git checkout name-of-the-correct-branch
# 把主分支上的最新提交摘过来，嘻嘻~~
git cherry-pick master
# 再删掉主分支上的最新提交
git checkout master
git reset HEAD~ --hard
```

二.修改很久之前的一次提交

> 1.查看修改

```
git rebase -i master~1 #最后一次
git rebase -i master~5 #最后五次
git rebase -i HEAD~3   #当前版本的倒数第三次状态
git rebase -i 32e0a87f #指定的SHA位置
```

2.显示结果如下，修改 pick 为 edit ，并 :wq 保存退出

```
pick 92b495b 2009-08-08: xxxxxxx

# Rebase 9ef2b1f..92b495b onto 9ef2b1f
#
# Commands:
#  pick = use commit
#  edit = use commit, but stop for amending //改上面的 pick 为 edit
#  squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
```

3.命令行显示：
Stopped at e35b8f3… reflog branch first commit
You can amend the commit now, with
git commit –amend
Once you are satisfied with your changes, run
git rebase –continue

```
#修改需要修改的地方（只是修改commit message就不用做）
git add . #这一步如果只是修改commit message不用输入
git commit --amend
#输入修改后的commit message，保存
```

4.使用 git rebase –continue 完成操作

```
git rebase --continue
```

5.推送到远端（若还没有推送到远端，不用处理）

```
git push <remote> <branch> -f
```