Git05-版本回退

回退缘由

比如我们使用git不断的新增、编辑和删除文件，每进行一次add操作和commit操作就会以当前时间点创建一个新的版本，每个版本都有一个版本ID号。

问题1：我想回退到以前的一个版本

问题2：我回退到了几天前的版本，但是后悔了，又怎么返回到最新的版本。

##版本回退

查看版本

```
➜ git log
```

执行上面的指令会进入一个窗口，信息如下：

```
commit 5274c016d2f2cdc2146eb62ecb05aa6a876aea1b (HEAD -> master)
Author: Jusen <1537017271@qq.com>
Date:   Tue Apr 3 20:36:37 2018 +0800

增加了新内容222

commit eac0506945bebf40cdafe45e974d266d5db7050a
Author: Jusen <1537017271@qq.com>
Date:   Tue Apr 3 20:33:57 2018 +0800

增加了新内容111

commit ebfdcf6e9c655292839eb7792e96c6d75d01565a
Author: Jusen <1537017271@qq.com>
Date:   Tue Apr 3 20:28:22 2018 +0800

新建了一个空白文件
(END)
```

它记录了我们操作的所有历史版本信息。我们可以按“q”退出该窗口。

如果我们的版本记录特别的多，我们可以使用简化的方式一行记录一个版本信息，操作如下：

```
➜  git log --pretty=oneline
```

窗口的信息如下：

```
5274c016d2f2cdc2146eb62ecb05aa6a876aea1b (HEAD -> master) 增加了新内容222
eac0506945bebf40cdafe45e974d266d5db7050a 增加了新内容111
ebfdcf6e9c655292839eb7792e96c6d75d01565a 新建了一个空白文件
(END)
```

我们注意到每一个版本记录信息都是一个长串，加上一段描述。其实这个长串就是版本ID，描述是我们使用`commit`提交版本的注释。

Git的commit id不是1，2，3……递增的数字，而是一个SHA1计算出来的一个非常大的数字，用十六进制表示。避免多人协同的时候出现使用相同的版本号冲突。

回退之前的版本

我们进行版本回退的时候都是以当前版本作为参考，我们以HEAD表示当前版本。上一个版本就是HEAD，上上一个版本就是HEAD，当然往上100个版本写100个比较容易数不过来，所以写成HEAD~100。

```
➜ git reset --hard HEAD^
HEAD is now at eac0506 增加了新内容111
```

控制台输出的`HEAD is now at eac0506 增加了新内容111`表示当前的HEAD版本ID为“eac0506”，这个是完整的版本ID前7位，后面是我们对这个版本的注释。

如果我们继续执行`git reset --hard HEAD^`，相当于在当前的版本再回退一个版本。

后悔药

问题2：我回退到了几天前的版本，但是后悔了，又怎么返回到最新的版本。

Git为我们提供了后悔药，其实只要我们知道需要回滚的版本ID即可，无论是在当前版本的前面还是后面，我们都可以滚动到相应的版本，只要我们知道版本ID即可。

那么如何获得比当前版本新的版本ID呢，使用`git log`只能获取当前版本及以前的版本信息，自然是不行的。我们可以使用`git reglog`来间接查找版本ID，该指令会记录我们的每一次操作。

```
➜  git reflog
eac0506 (HEAD -> master) HEAD@{0}: reset: moving to HEAD^
5274c01 HEAD@{1}: reset: moving to 5274c01
5274c01 HEAD@{2}: reset: moving to 5274c01
ebfdcf6 HEAD@{3}: reset: moving to HEAD^
eac0506 (HEAD -> master) HEAD@{4}: reset: moving to HEAD^
5274c01 HEAD@{5}: commit: 增加了新内容222
eac0506 (HEAD -> master) HEAD@{6}: commit: 增加了新内容111
ebfdcf6 HEAD@{7}: commit (initial): 新建了一个空白文件
(END)
```

根据全部的git操作记录，我们可以查找到我们需要的版本ID的前缀，没必要使用完整的ID，只要能够区分版本即可。

然后我们会退到任何我们想要的版本

```
➜  git reset --hard 5274c01
HEAD is now at 5274c01 增加了新内容222
```