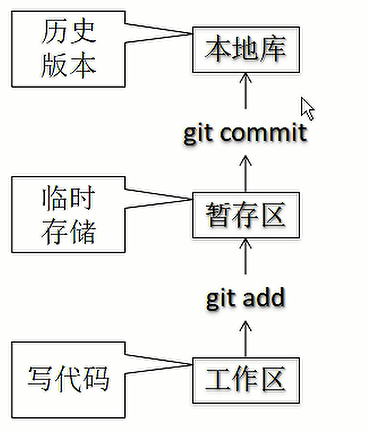
1. **git结构**



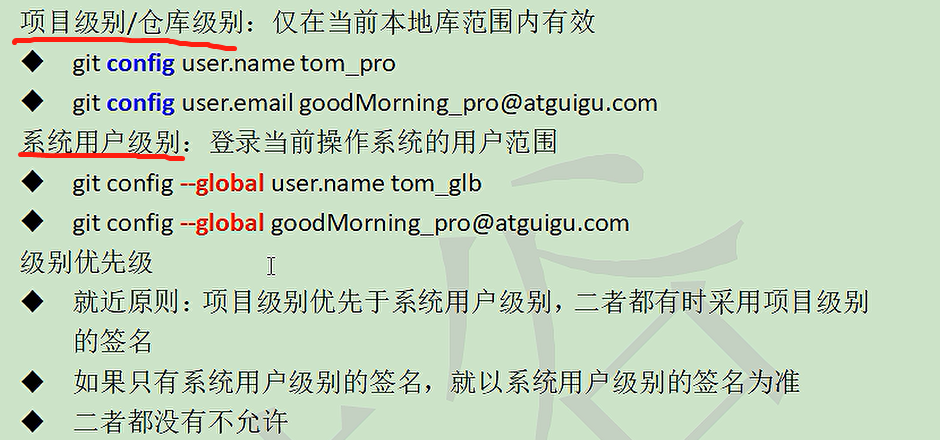
1. **git初始化**

git init 会创建.git文件夹

1. **设置签名**

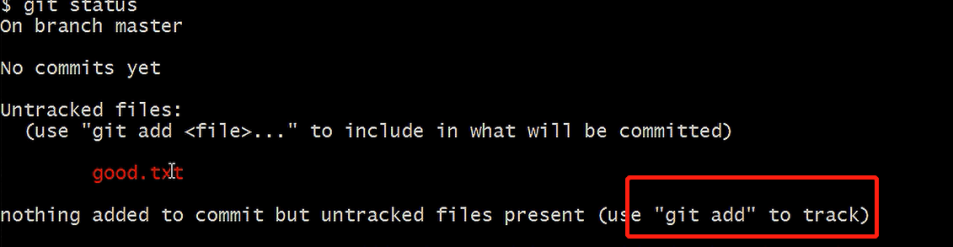
形式: 用户名：tom,EMAIL：aa@a.com作用：只是为了区分不同开发人员的身份，不会给你发邮件的。

级别：

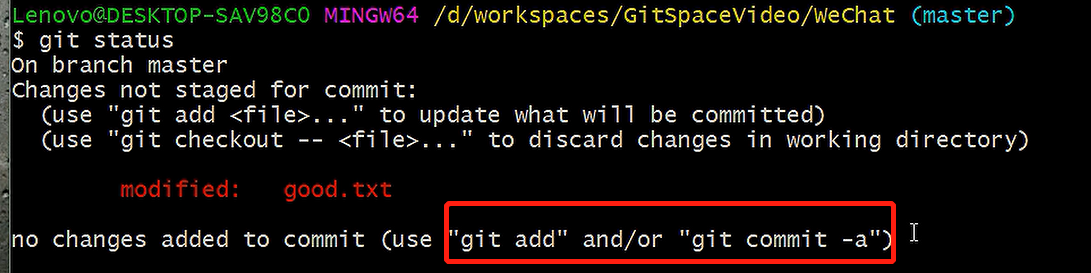


信息保存位置:项目级别保存在.git/config里，系统级别保存在~/.gitconfig里。一般来说只需要设置全局的就可以了。

1. **常规提交命令**
   1. **添加文件git add**

对要增加的文件进行追踪

* 1. **更新文件**



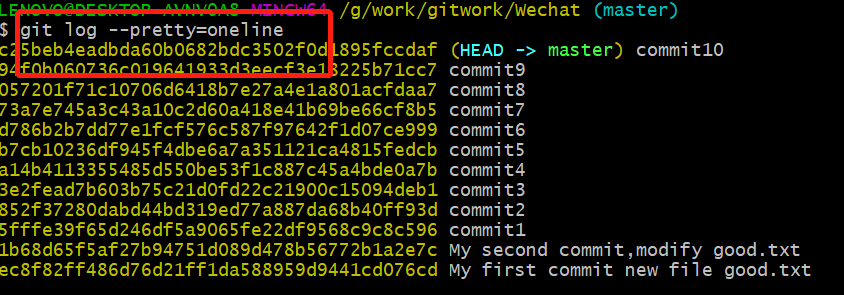
如果你已经追踪了，你可以git add添加到暂存区，也可以用git commit直接提交。但是如果直接提交的话，你就不能够撤销了。

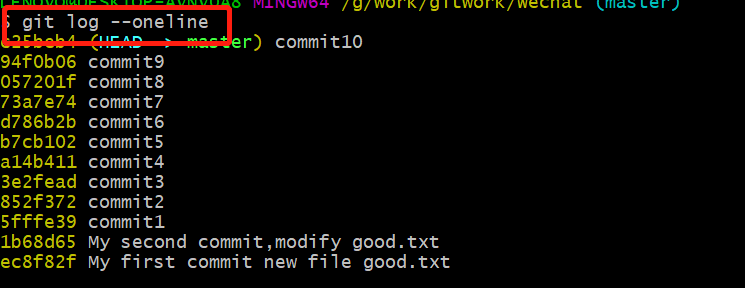
git commit -m ‘提交注释’ 文件名

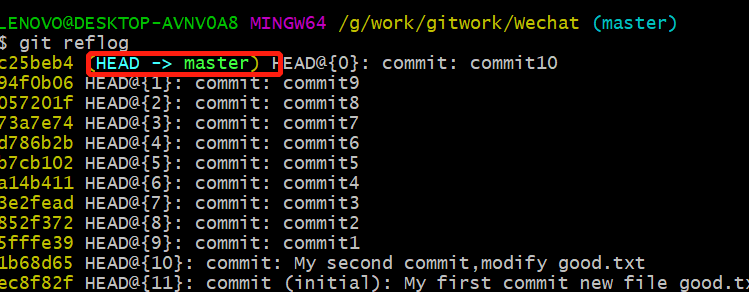
1. **版本穿梭,其实就是移动head的指针**
   1. **git log**



空格向下翻页，b向上翻页 q退出

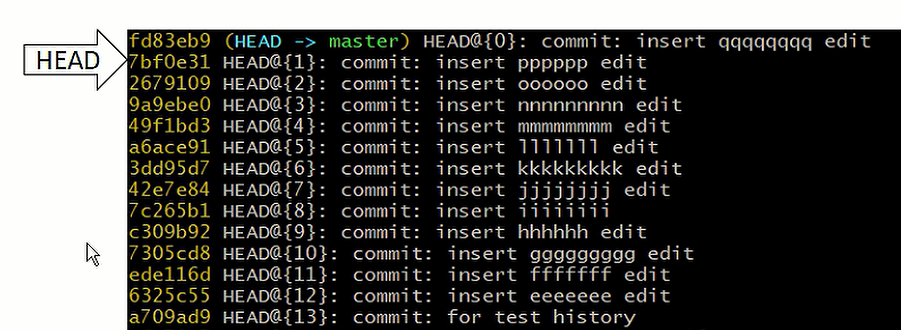






reflog显示的日志对于前进后退，移动指针是非常有参考价值的，head@{移动到当前版本需要的步数}。比如{2}就说明到这里你需要移动2步

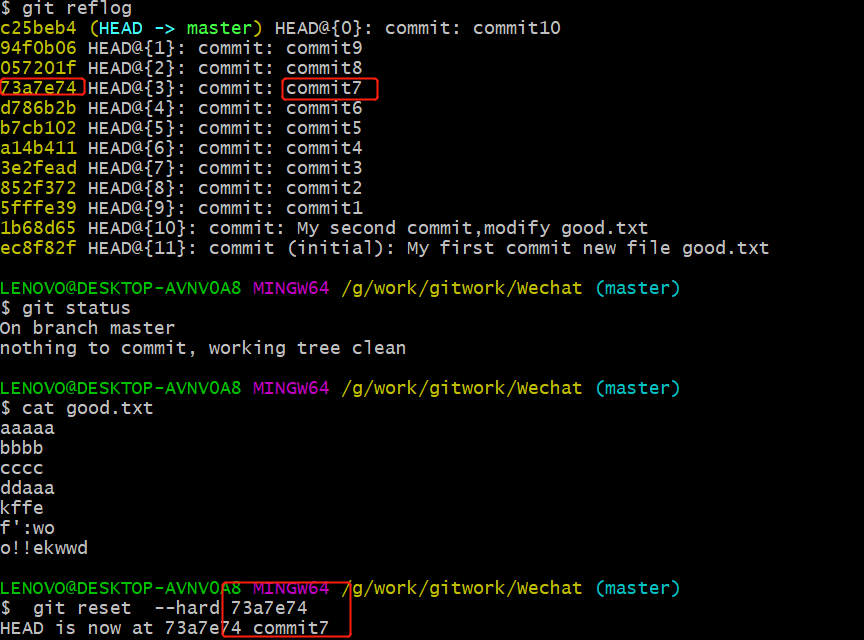
* 1. **前进后退历史版本操作的本质**

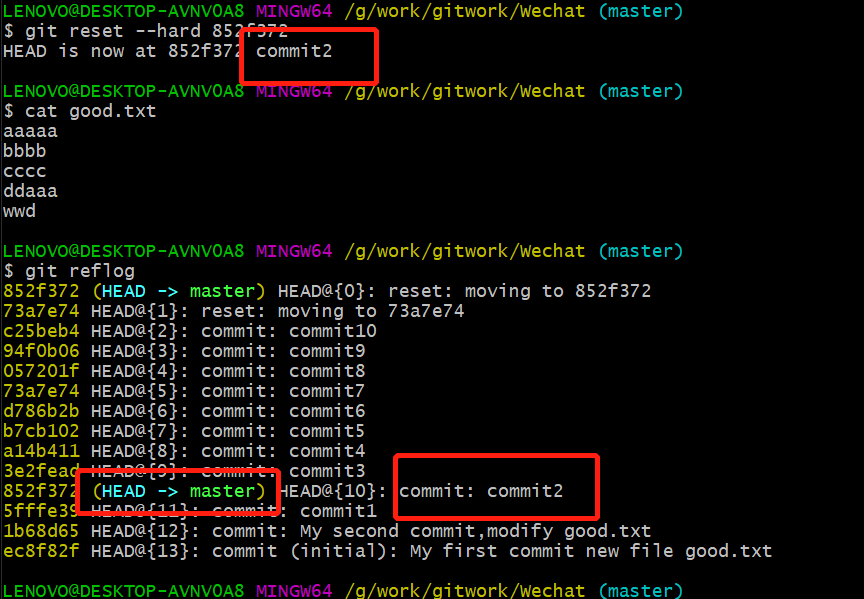


head就是这个指针的名字。这个指针为什么要给他起个名字？因为这样能方便我们去引用。

* 1. **版本前进后退**

有三种方式，一般只用基于索引值来操作。命令都是git reset --hard。这里的--hard说明本地库和暂存区都会被刷新（重置）。

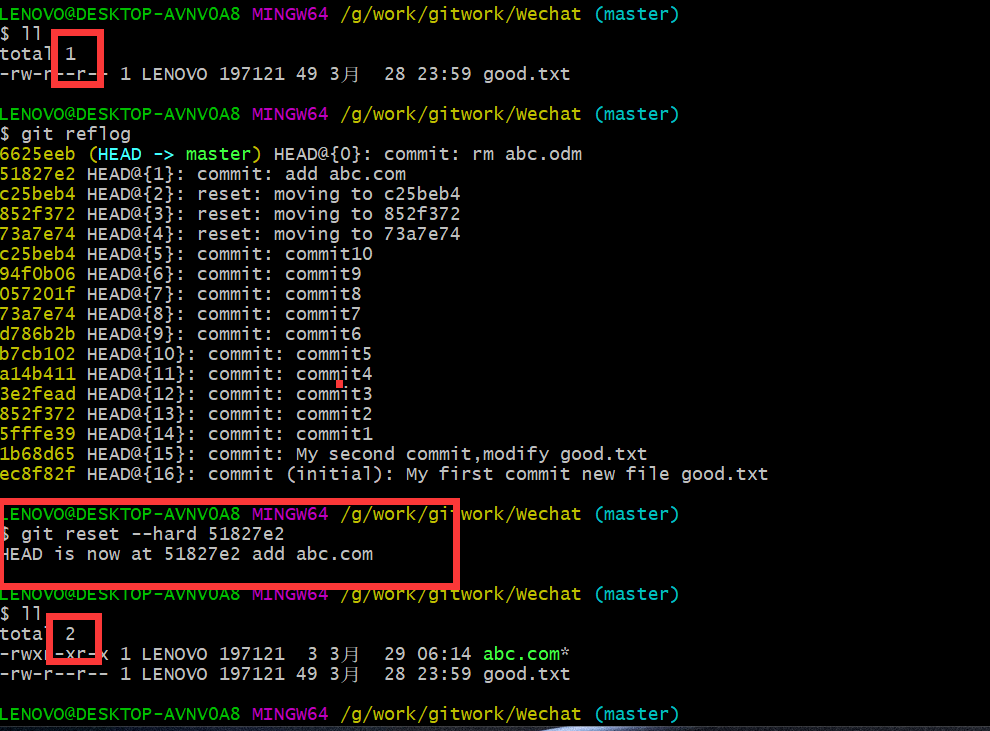




其他两种方式了解即可：使用^或~，只能后退，几个符号就是几步，如git reset --hard HEAD^^^ ==git reset --hard HEAD~3

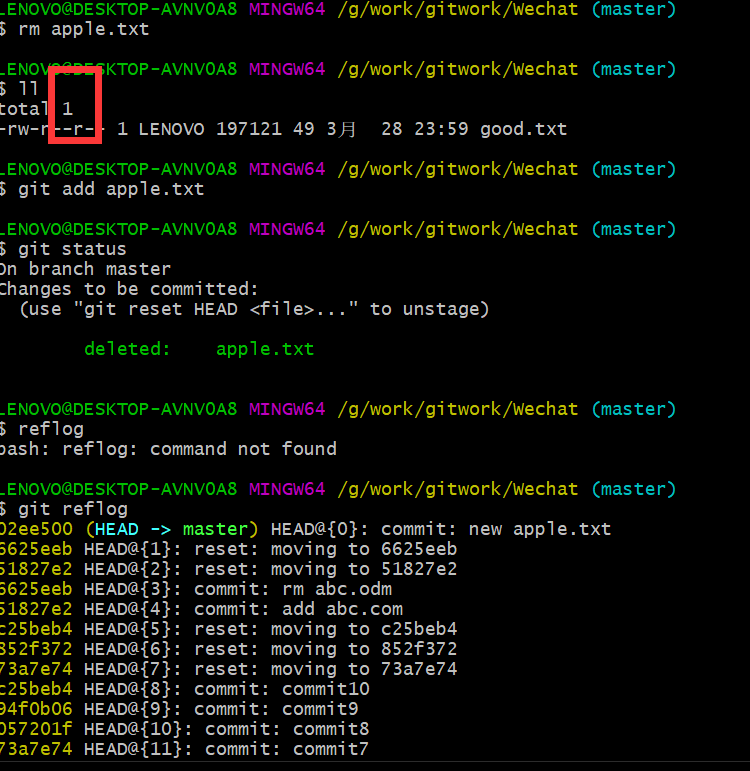
* 1. **删除文件后找回** 
     1. **永久删除文件后找回**

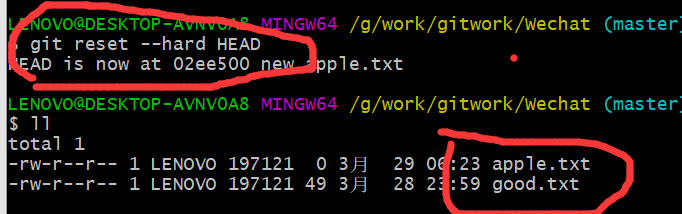
先要保证删除之前文件已经提交到本地库，然后删除文件，删除之后又提交到本地库了，那就直接把指针移动到还没删除之前那个版本就行了。



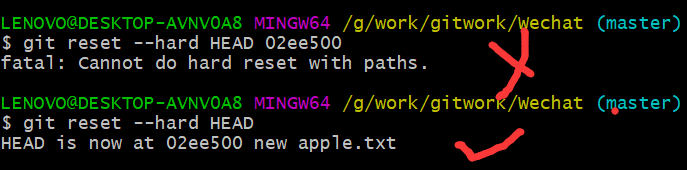
* + 1. **添加到暂存区的删除文件找回**

先要保证删除之前文件已经提交到本地库，然后删除文件，这时候把删除的文件添加到了暂存区，但还没有提交到本地库。



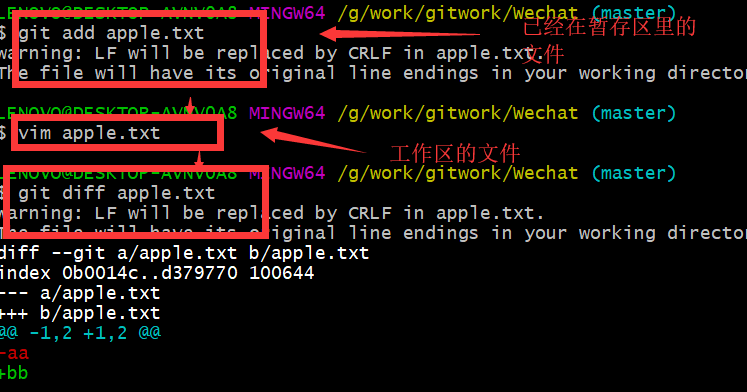


由于已经是最新的了，所以不能（注意是不能而不是不需要）指定版本号，只要打HEAD就可以了，否则会报下面错误



1. **比较文件**
   1. git diff [文件名]

将工作区的文件和暂存区的文件进行比较



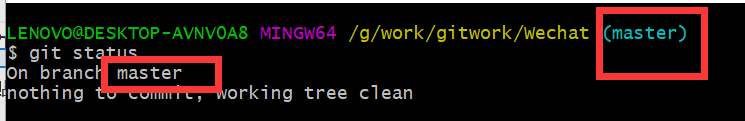
* 1. git diff [本地库的历史版本][文件名]

将工作区中的文件和本地库历史记录进行比较



* 1. 不带文件名就会比较多个文件

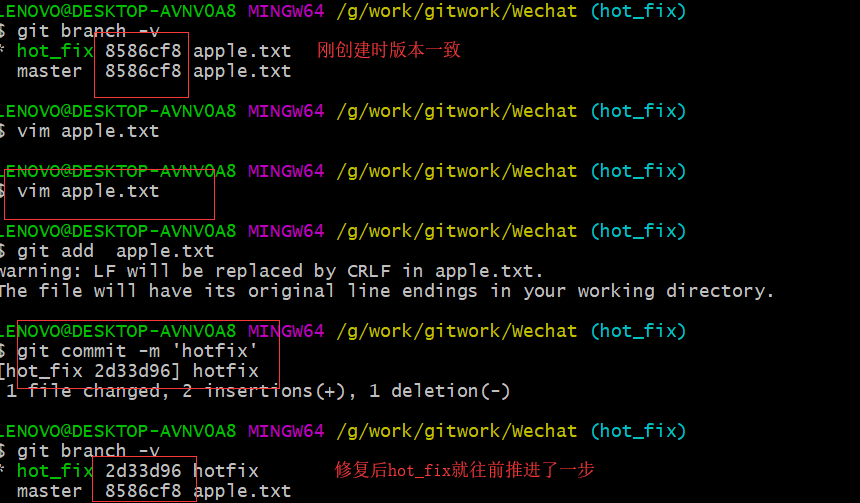
1. **分支**
   1. 初始话好以后就会有master分支，不需要手动创建



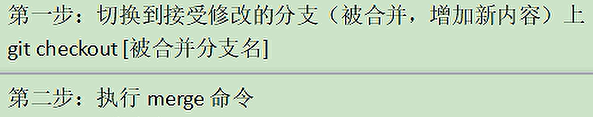
* 1. 创建,查看，切换分支



* 1. 合并分支（非常重要）
     1. 先在分支上修改并提交



* + 1. 开始合并





* + 1. 解决冲突