**最先进的分布式版本控制系统——git**

1. 为什么要用它？

它能自动记录每次文件的改动，还可以让其他人协作编辑，也不需要把文件传来传去。如果想查看某次改动，在里面查看说明就可以。

1. 集中式vs分布式

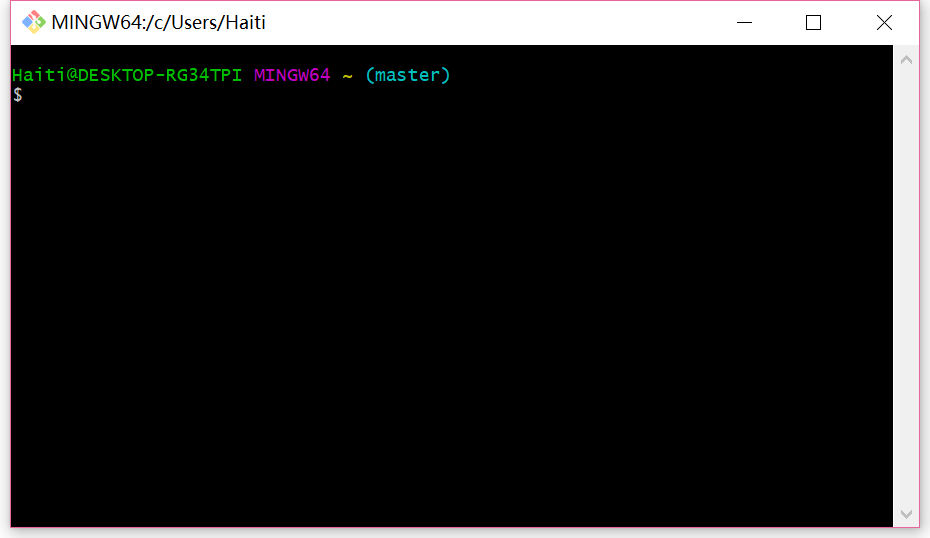
|  |  |
| --- | --- |
| 集中式 | 分布式 |
| 版本库是集中存放在中央服务器 | 每个人电脑里都有完整的版本库 |
| 安全性低 | 安全性要高 |
| 必须联网才能工作 | 不需联网也可以工作 |
| 直线管理 | 具有强大的分支管理功能 |

1. 安装git

Window系统：https://git-scm.com/downloads

安装完成后，在开始菜单里找到“Git”->“Git Bash”，蹦出一个类似命令行窗口的东西，就说明Git安装成功！





安装完成后，还需要最后一步设置，在命令行输入：

$ git config --global user.name "Your Name"

$ git config --global user.email "email@example.com"

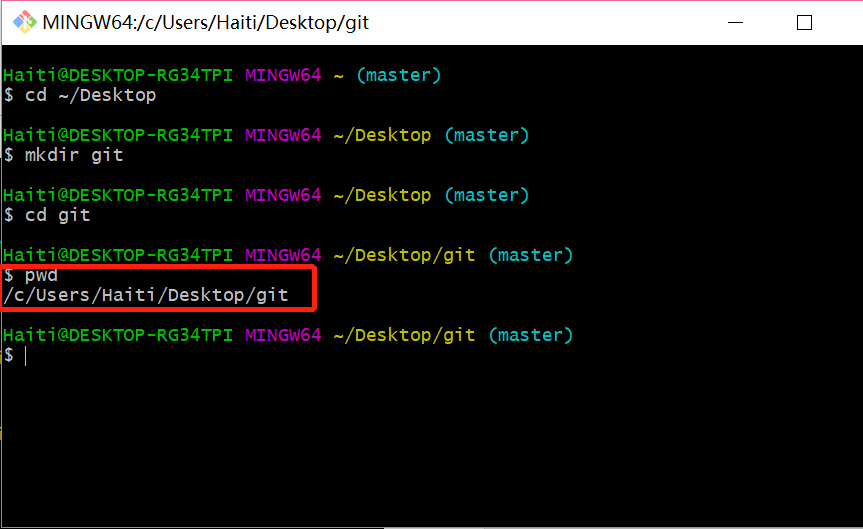
1. 什么是版本库？

理解成一个目录，这个目录里面的所有文件都可以被Git管理起来，每个文件的修改、删除，Git都能跟踪，以便任何时刻都可以追踪历史，或者在将来某个时刻可以“还原”。

1. **创建版本库**

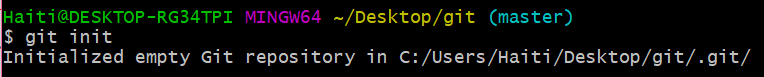
**第一步：在桌面新建了一个git文件夹（**如果你使用Windows系统，为了避免遇到各种莫名其妙的问题，请确保目录名（包括父目录）不包含中文。**）**

* 1. **mkdir命令**用来创建目录
  2. **cd命令**用来切换工作目录至[dirname](http://man.linuxde.net/dirname)。 其中dirName表示法可为绝对路径或相对路径。若目录名称省略，则变换至使用者的home directory(也就是刚[login](http://man.linuxde.net/login)时所在的目录)。另外，~也表示为home directory的意思，.则是表示目前所在的目录，..则表示目前目录位置的上一层目录。



* 1. **pwd命令**以绝对路径的方式显示用户当前工作目录。命令将当前目录的全路径名称（从根目录）写入标准输出。全部目录使用/分隔。第一个/表示根目录，最后一个目录是当前目录。执行pwd命令可立刻得知您目前所在的工作目录的绝对路径名称。

**第二步：**通过git init命令把这个目录变成Git可以管理的仓库。



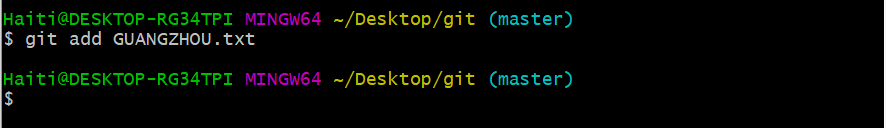


当前目录下多了一个.git的目录，这个目录是Git来跟踪管理版本库的。

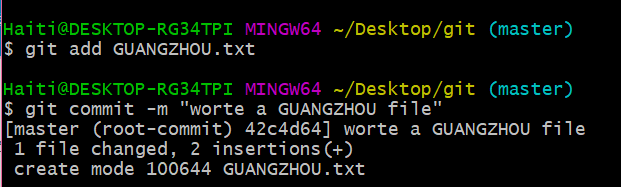
**第三步：把文件添加到版本库。**

编写一个GUANGZHOU.txt文件，要放在git目录下（子目录也行），因为这是一个Git仓库，放到其他地方Git再厉害也找不到这个文件。

用命令git add告诉Git，把文件添加到仓库：



用命令git commit告诉Git，把文件提交到仓库：



**git commit -m “**message**”：提交的描述信息，**可以输入任意内容**。**

git commit命令执行成功后会告诉你，1 file changed：1个文件被改动（我们新添加的readme.txt文件）；2 insertions：插入了两行内容（GUANGZHOU.txt有两行内容）。

为什么Git添加文件需要add，commit一共两步呢？因为commit可以一次提交很多文件，所以你可以多次add不同的文件，比如：

$ git add file1.txt

$ git add file2.txt file3.txt

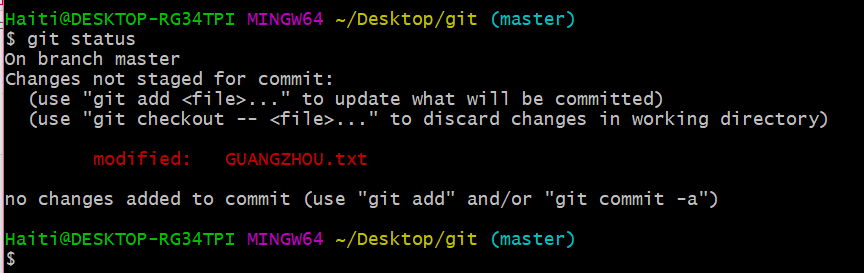
$ git commit -m "add 3 files."

1. **文件内容修改及查看文件修改内容**

要随时掌握工作区的状态，使用git status命令。

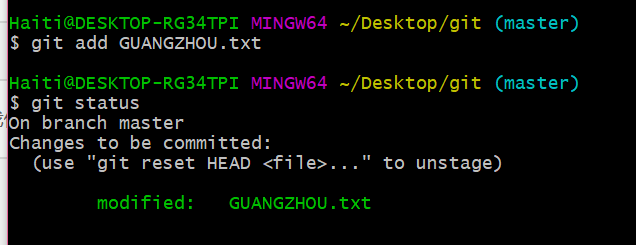
如果git status告诉你有文件被修改过，用git diff可以查看修改内容。

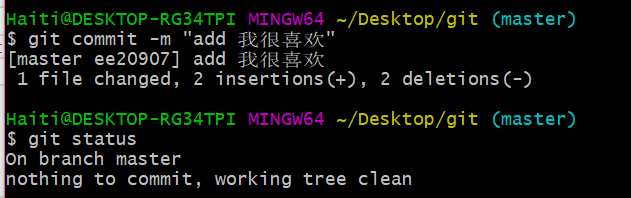
git status命令可以让我们时刻掌握仓库当前的状态，上面的命令输出告诉我们，GUANGZHOU.txt被修改过了，但还没有准备提交的修改。



git diff顾名思义就是查看difference，显示的格式正是Unix通用的diff格式。

**这一步知道修改了什么之后，可以放心提交给仓库，**提交修改和提交新文件是一样的两步，第一步是git add,第二步是git status查看当前仓库的状态，git status告诉我们，将要被提交的修改包括GUANGZHOU.txt，第三步是git commit -m “add 我很喜欢”





1. 版本回退

HEAD指向的版本就是当前版本，因此，Git允许我们在版本的历史之间穿梭，使用命令git reset --hard commit\_id。

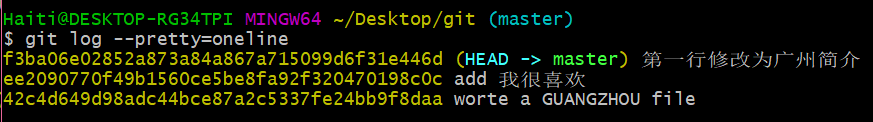
穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。

要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

git log命令查看历史记录，git log命令显示从最近到最远的提交日志。



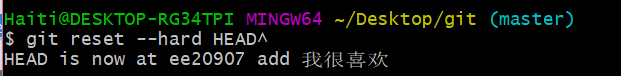
如果嫌输出信息太多，看得眼花缭乱的，可以试试加上--pretty=oneline参数：



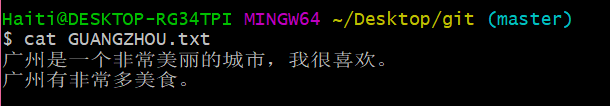
准备把GUANGZHOU.txt回退到上一个版本

首先，Git必须知道当前版本是哪个版本，在Git中，用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。

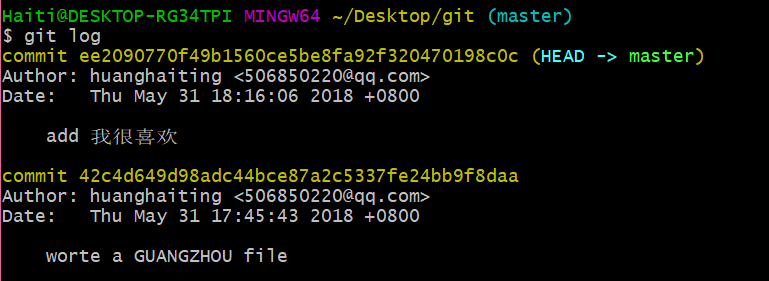
现在，我们要把当前版本回退到上一个版本，就可以使用git reset命令：



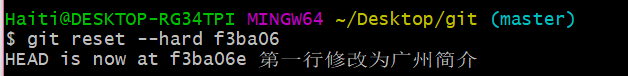
打开GUANGZHOU.txt文档看看，发现回到了“add 我很喜欢”



git log命令查看历史记录，出现两个版本。

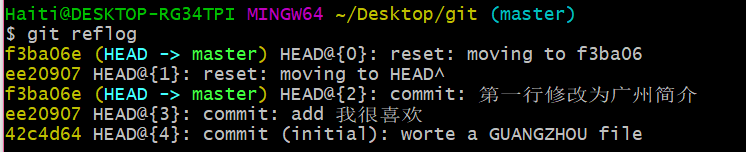


**如何恢复“第一行改为广州简介”版本呢？只需要输入版本号。**



现在，你回退到了某个版本，关掉了电脑，第二天早上就后悔了，想恢复到新版本怎么办？找不到新版本的commit id怎么办？

Git提供了一个命令git reflog用来记录你的每一次命令：



1. **工作区和暂存区**

**工作区**

**（**在电脑里能看到的目录，比如“GUANGZHOU.txt”**）**

**git add 操作后**

**暂存区**

**git commit -m”massge”**

往master分支上提交更改

**总结：**

把文件往Git版本库里添加的时候，是分两步执行的：

第一步是用git add把文件添加进去，实际上就是把文件修改添加到暂存区；

第二步是用git commit提交更改，实际上就是把暂存区的所有内容提交到当前分支。

因为我们创建Git版本库时，Git自动为我们创建了唯一一个master分支，所以，现在，git commit就是往master分支上提交更改。

你可以简单理解为，需要提交的文件修改通通放到暂存区，然后，一次性提交暂存区的所有修改。

1. 管理修改

第一次修改-> git add （第一次修改存入暂存区）-> 第二次修改 -> git commit（仅提交了第一次的修改）

所以：

第一次修改 -> git add -> 第二次修改 -> git add -> git commit

并说明一个问题：每次修改，如果不用git add到暂存区，那就不会加入到commit中。

10、撤销修改

**第一种：**git checkout -- file可以丢弃工作区的修改

把GUANGZHOU.txt文件在工作区的修改全部撤销，这里有两种情况：

一种是GUANGZHOU.txt自修改后还没有被放到暂存区，现在，撤销修改就回到和版本库一模一样的状态；

一种是GUANGZHOU.txt已经添加到暂存区后，又作了修改，现在，撤销修改就回到添加到暂存区后的状态。

总之，就是让这个文件回到最近一次git commit或git add时的状态。

git checkout -- file命令中的--很重要，没有--，就变成了“切换到另一个分支”的命令。

**第二种：**git add到暂存区了，在commit之前，你发现了这个问题，用命令git reset HEAD <file>可以把暂存区的修改撤销掉（unstage），重新放回工作区。git reset命令既可以回退版本，也可以把暂存区的修改回退到工作区。

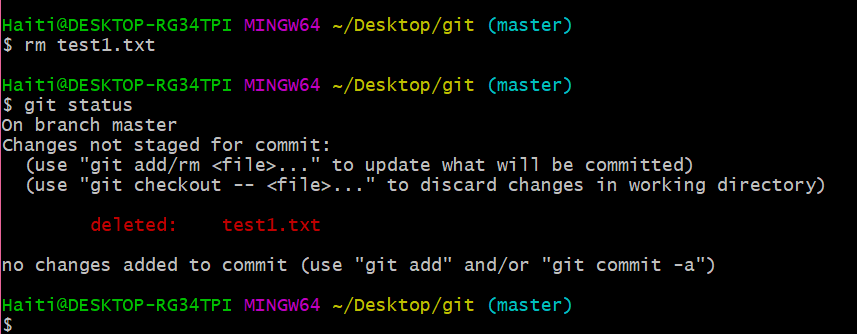
**总结：**

场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout -- file。

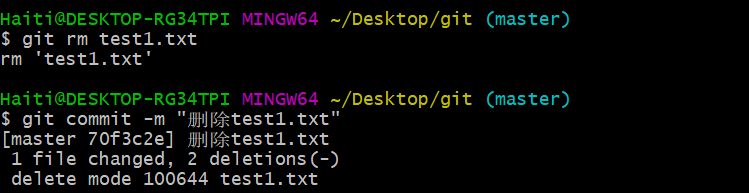
场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD <file>，就回到了场景1，第二步按场景1操作。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考[版本回退](https://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000/0013744142037508cf42e51debf49668810645e02887691000)，不过前提是没有推送到远程库。

1. 删除文件



Git知道你删除了文件，因此，工作区和版本库就不一致了，git status命令会立刻告诉你哪些文件被删除了，现在你有两个选择，一是确实要从版本库中删除该文件，那就用命令git rm删掉，并且git commit



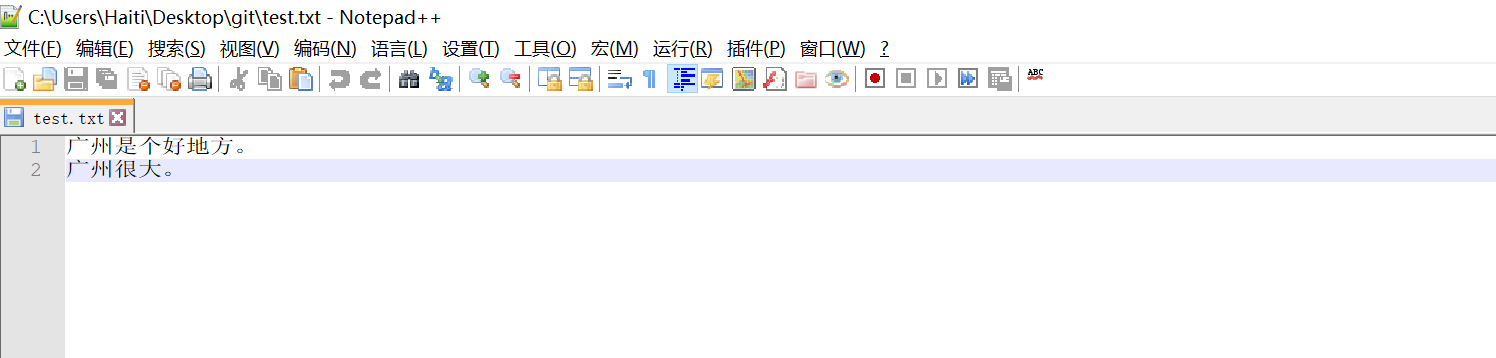
--------------------------------------文件就从版本库中被删除了----------------------------------

二是删错了，想后悔，使用git checkout。其实是用版本库里的版本替换工作区的版本，无论工作区是修改还是删除，都可以“一键还原”。

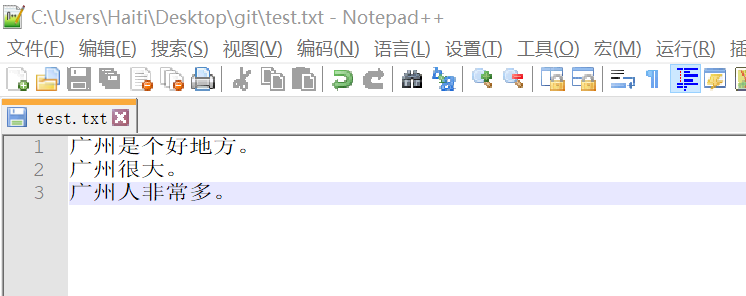
命令git rm用于删除一个文件。如果一个文件已经被提交到版本库，那么你永远不用担心误删，但是要小心，你只能恢复文件到最新版本，你会丢失**最近一次提交后你修改的内容**。

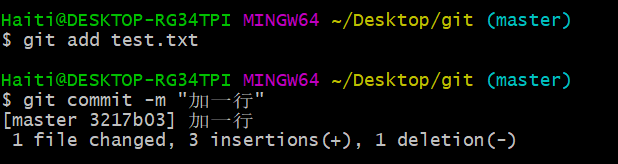
解释：

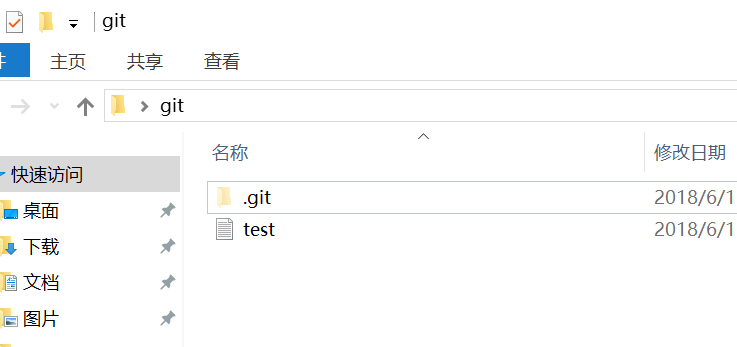
原本的文档内容



**编辑一次后，并commit进版本库**

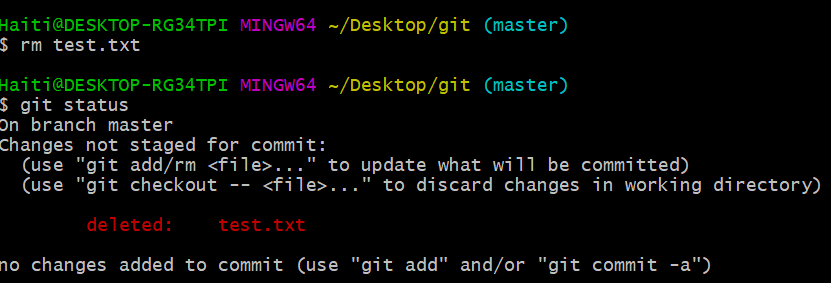




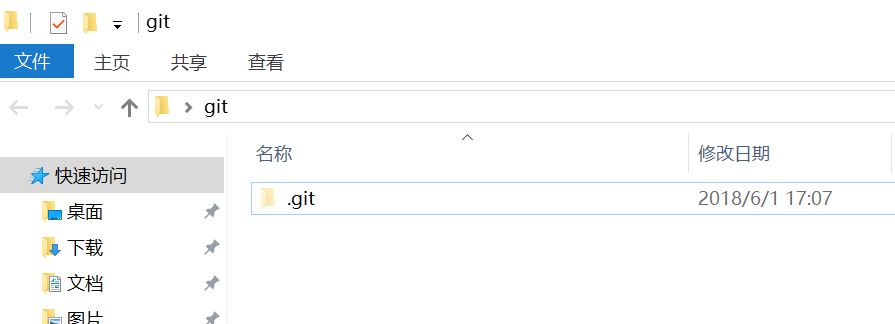


**此时的工作台文档还在！！**

**然后删除test.txt 文档**

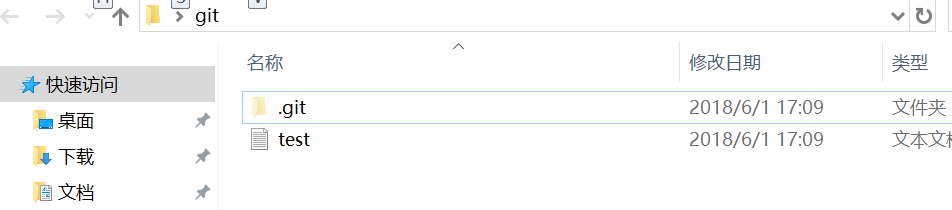


**此时工作台文档已被删除！！如果想要恢复，可根据提示输入git checkout -- <file>**

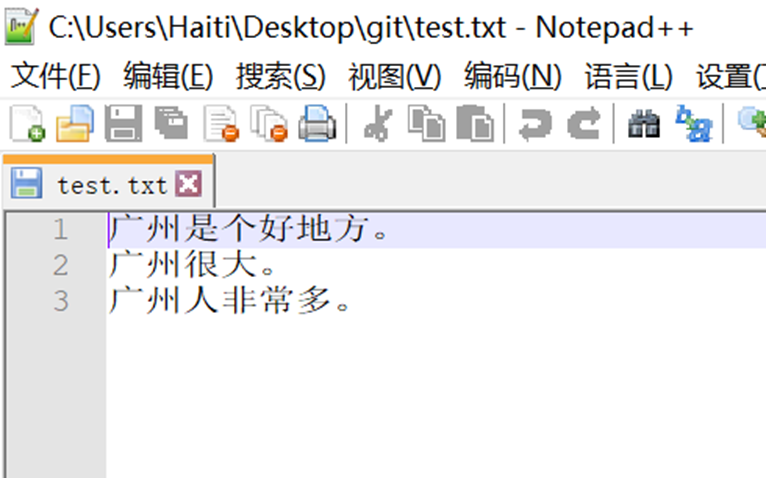




**工作台出现了test.txt 文档**



**打开文档恢复到上一次commit的内容。**

****