git地址 <https://git-scm.com/>

git 发展史

linux系统 -> BitKeeper 2005开始收费

linus 开发了git

版本控制系统: 集群式版本控制(cvs, svn) 分布式版本控制(git)

版本控制可以回退版本

集中式: 通过一个服务器来维护版本 ,自身是不带版本控制,必须去集中式仓库查询版本,回退版本

分布式版本控制:每个计算机都有一套版本控制系统 ,回退到任何版本,只需要看本机的版本库就好

服务器里面的也有版本控制

分布式 : 多个服务器电脑,可以将同样的代码,推到多个服务器上面

一般使用github服务器托管

分为public和private public (免费的) private(收费的)

gitlab(公司内部使用)

git的优势

1.可以建立本地版本库, 可以重写本地提交说明

2.可以"后悔"

3.分支branch系统.根据不同的需求,开发不同的分支

4.增量与全量的关系

svn:增量 修改一个文件,其它文件不变,改什么,变什么,其余的不懂,要改什么,记录什么

git:全量 无论是修改了几个文件,每次更改都是全体变化,没有更改的那部分将上个版本的进行一下复制, 每一个版本里面都包含全部的文件,时刻保持数据的完整性

git的三种状态(个人理解是四种)

// 管理区(有些文件是自由文件,脱离git管理) 只有已经被git管理的文件才能被操控

已修改(modified) 更改完工作区里面的文件以后的状态

已暂存(staged) 将工作区的文件提交到暂存区之后的状态

已提交(commited) 将暂存区的文件提交到对象区之后的状态

工作区 工作区是有实际编写的文件

add之后进入暂存区

暂存区

commit将代码从暂存区提交到对象区

对象区 : 只能通过对象区将文件提交到服务器里面

push命令 将对象区代码提交到服务器里面

服务器:

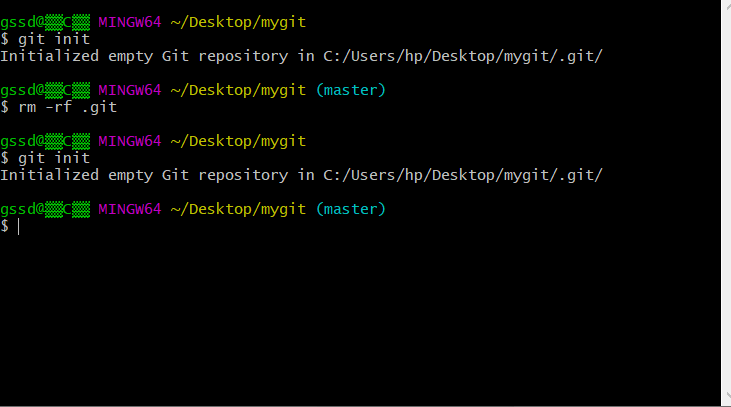
版本库

暂存区和对象区加起来可以看成版本库

git init 新建一个git管理目录,将某个版本纳入到管理目录,默认创建master目录

.git进行版本控制,

rm -rf .git 会将版本库全部删掉,然后版本控制消失，如下图



新建文件 touch命令

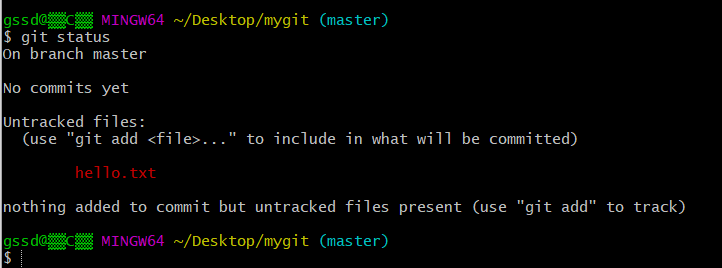
touch hello.txt

新建文件在工作区

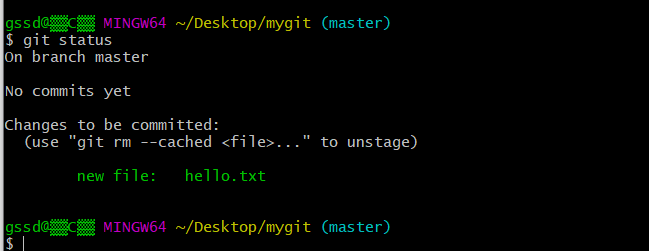
git status 查看当前的状态 （已修改，已暂存，已提交）

在master分支里面，还没有提交

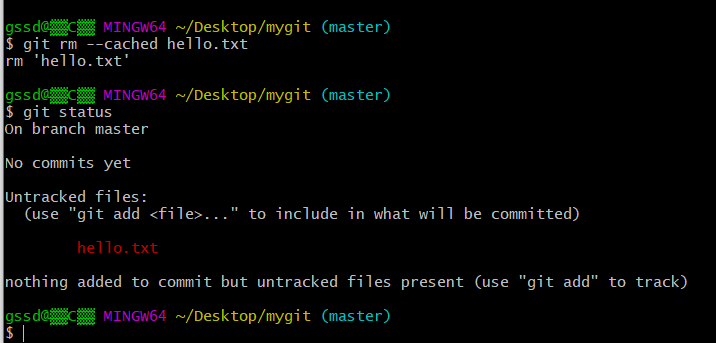
untracked files ，通过git add 命令将文件提交到暂存区



已经被提交了，但是现在处于stage状态，还没有进入对象区，可以通过git rm --chched <file>将文件回退到工作区



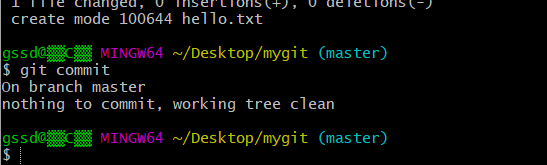
回退之后的状态



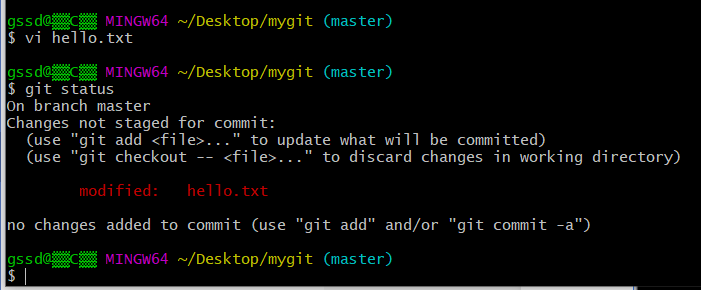
shift+z+z命令推出git commit的提交目录



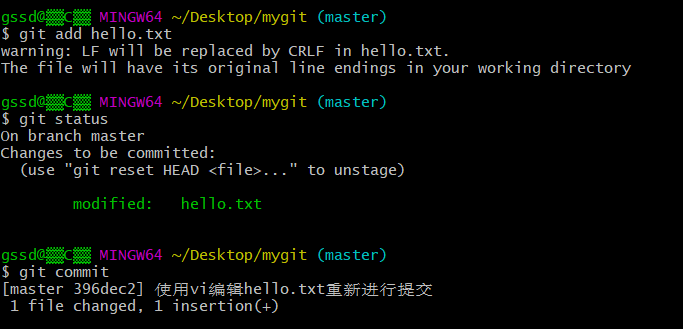
因为暂存区的所有文件在上一布全部都被提交成功，现在暂存区为空，没有文件，此时我们再使用git commit 就没有相应的作用了



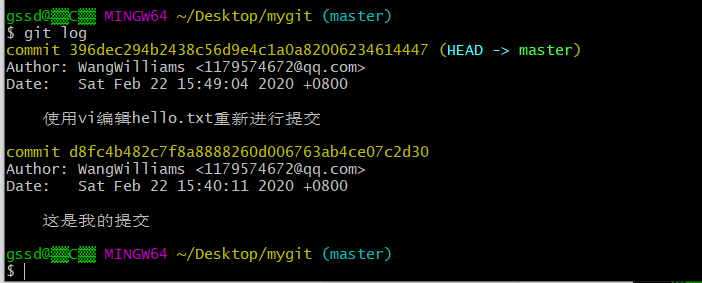
然后使用vi 命令修改了文件，现在重新查看文件的status，可以看到文件的是modified，显示已修改



在重新提交，发现 1 file changed ，1 insertion（+）变化了一个，插入了一个



然后使用git log 查看当前分支的日志，可以看到相应的提交信息



commit d8fc4b482c7f8a8888260d006763ab4ce07c2d30

分布式id生成器：后面的是通过sha1 \md5 计算出来的加密结果,这里用来实现随机数

通过唯一的随机数值，实现判断是谁提交的

通过

git log -2 查看最近的两次提交记录，-2中的数字2可以进行更改

git log --pretty=oneline 以一行的形式输出，如下图所示



git log --pretty=format:"%h - %an，%ar ： %s" 让日志按照某个格式来写

