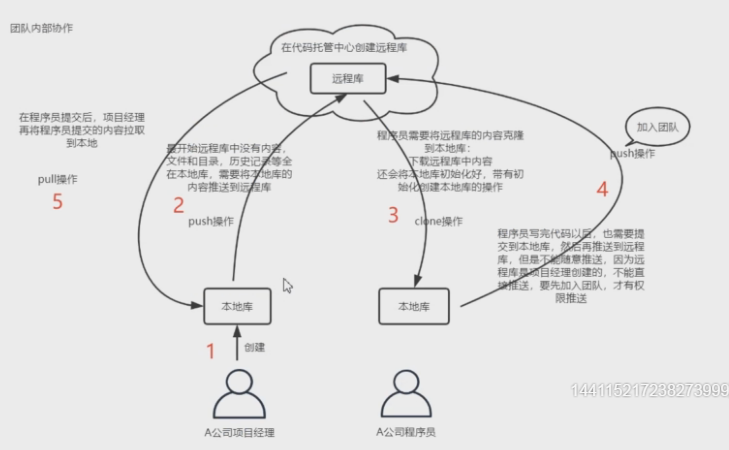
# Git



版本控制系统：

集中化版本控制系统：例如svn、cvs

分布式版本控制系统：客户端不止最新版本文件，而是把代码都镜像下来了。例如Git

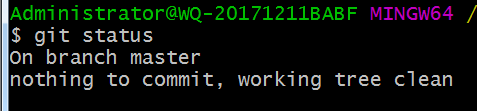
Svn存储的是当前版本和上一版本的差异。

Git储存的是当前版本和上一版本的索引。

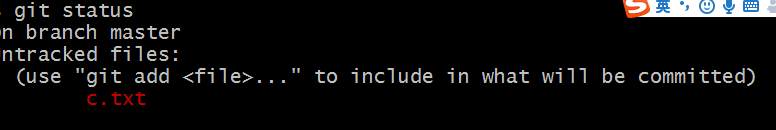
本地结构：工作区（写代码的地方）->Git.add->暂存区（临时存储）-> git.commit->本地库（历史版本信息）

## 使用方法：

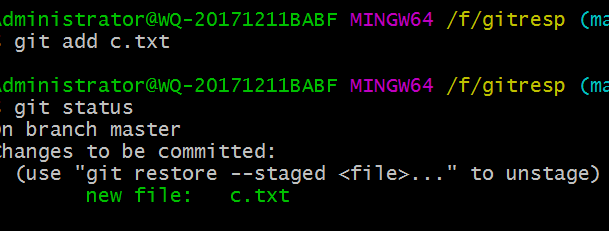
1. 在电脑里新建一个文件夹
2. 桌面右键选择git bash here
3. 然后右键修改字体和编码格式
4. 命令和linux一样：git –vrsion;clear;
5. 设置用户名和邮箱 ：git config --global user.name “niulipeng” ; git config --global user.email [784115501@qq.com](mailto:784115501@qq.com)
6. 本地仓库的初始化：cd命令进入本地文件夹 然后git init 。查看文件ll –la (隐藏目录.git文件此文件下的东西勿修改)
7. 本地新建文件（该文件需要放到本地仓库里）工作区git add test.txt –》暂存区git commit -m “注释” -》本地库。Ps:工作区就是本地磁盘
8. 查看当前状态 git status 如下：



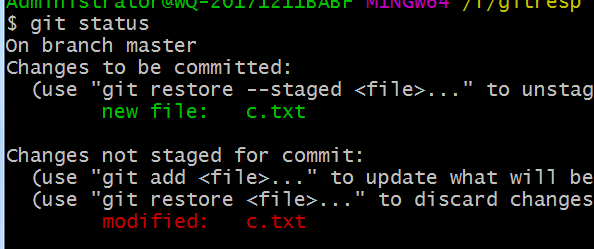
用命令mkdir 新建的文件会同步本地库 直接在文件夹新建的文件用此命令会报如下错误

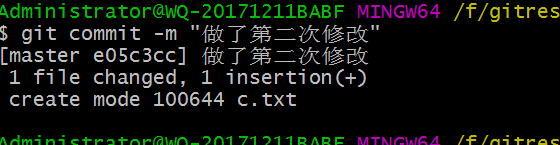


执行完 git add c.txt命令后如下：



修改c.txt文件之后 如下 需要重新添加到本地库





## 查看修改日志

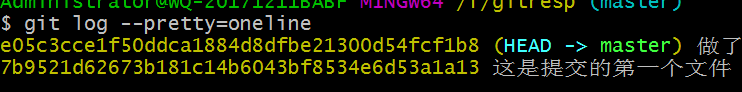
git log 命令

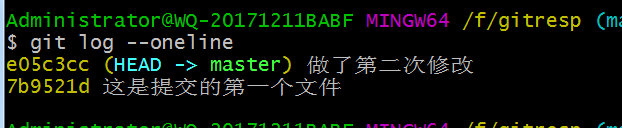


Commit后一连串的是当前提交版本的索引

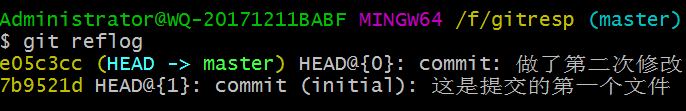
如果提交日志多会出现： 往下查找按 空格 往上查找按B 到尾页显示end 然后按 q退出

其他日志展示方式



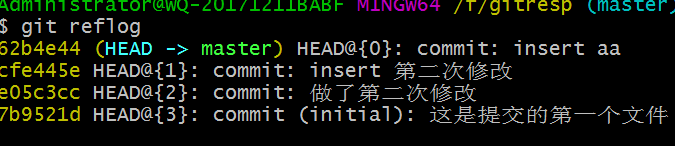


@里的数字表示当前库回退到该版本需要走的步数

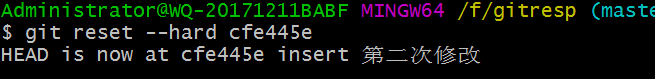


## Reset命令

git reset --hard [索引] 回退或者前进历史版本（工作区文件随着变更）如下



Hard 后面跟要跳转到版本的索引

·

此时当前文件的内容变更为指向的版本内容

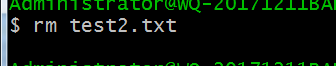
如果要回滚到当前版本需要从当前指向的索引修改到上一个索引

Git reset –mixed [索引] 只修改本地库和暂存区的文件，工作区的文件不变更

Git reset –soft [索引] 只修改本地库文件，暂存区和工作区的文件不变更

## 删除文件

**rm 文件名（只是删除工作区）**

****

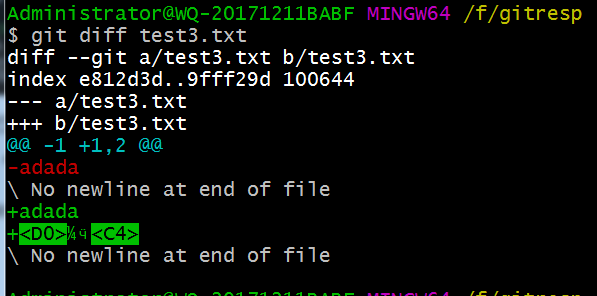
**然后以下操作删除本地库文件： git add test2.txt ； git commit –m “删除文件”**

**如果想恢复** git reset --hard [索引]

## 对比文件

修改本地文件导致本地区和暂存区不一致：git diff 文件名

比对文件差异git 是按行管理数据

****

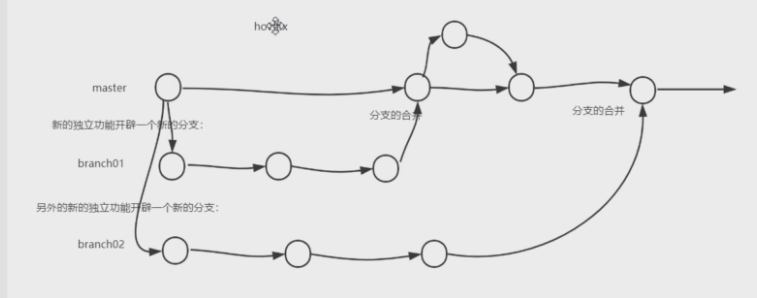
再添加

先删除一行

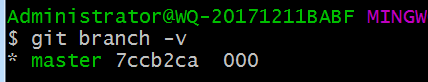
**Git diff 后不加文件名会展示工作区中和暂存区中所有文件的差异**

**Git diff HEAD指针/索引 文件名 比对暂存区和本地库的文件差异**

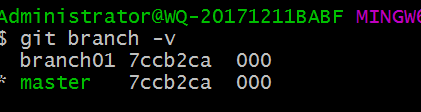
## 分支：

****

**查看分支 \*代表当期在哪个分支上**

****

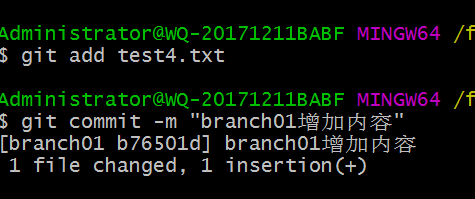
**创建分支**

****

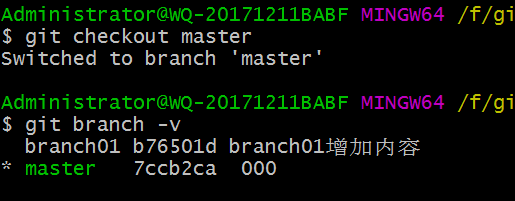
**切换分支**

****

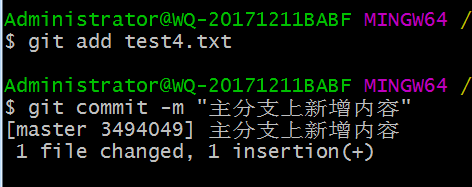
**在01分支下修改文件并提交**

****

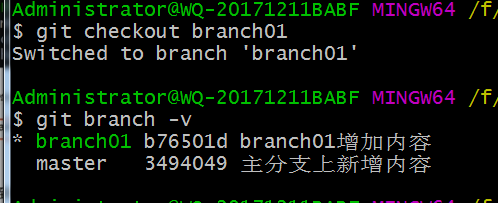
**然后切换到master分支**

****

**然后在主分支新增内容**

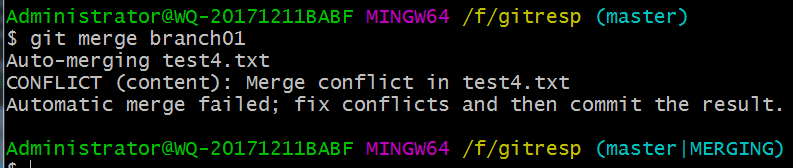
****

**然后再切换到01分支**

****

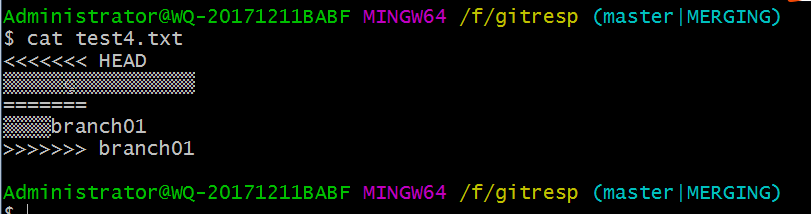
## 合并分支

**将01分支合并到主分支：先切换到主分支**

****

**出现冲突（修改同一个文件的同一个位置）merging表示正在合并中**

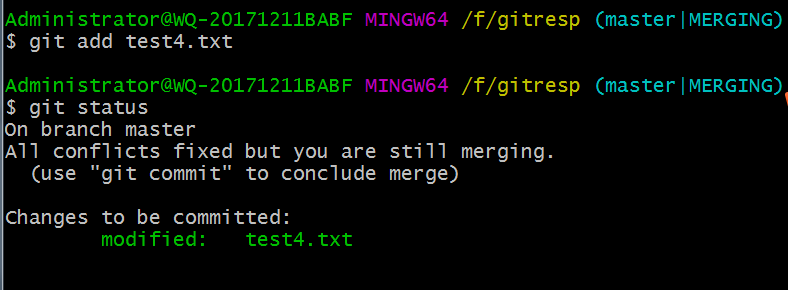
**查看文件（未修改txt的编码格式导致乱码）**

****

**合并修改的内容**

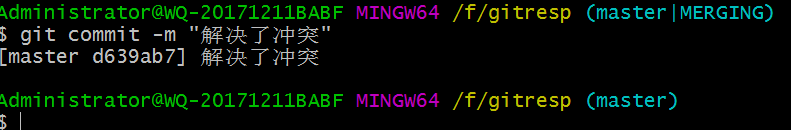
当前分支修改的内容

**去工作区修改最终的文件然后将文件添加到暂存区**

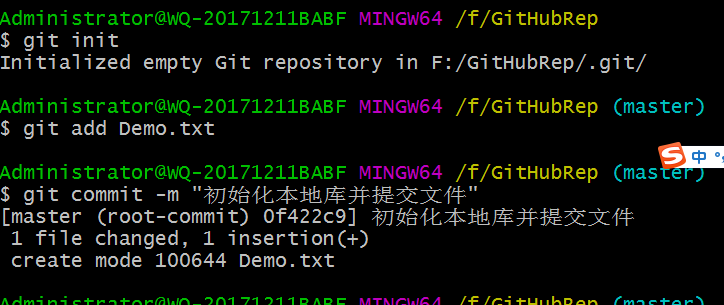
****

表示冲突已经解决但是仍处于合并状态

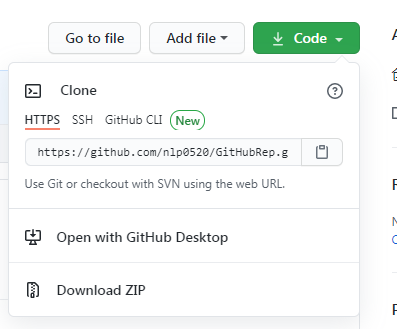
**然后提交到本地库**

****

## 创建GitHub远程库

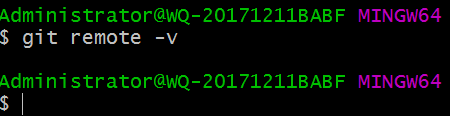
****

**在github上创建一个库，查看地址（新版的在code下 ）**

****

**在git本地将地址，通过别名**

**查看别名 git remote –v**

****

**新增别名 git remote add origin** [**https://github.com/nlp0520/GitHubRep.git**](https://github.com/nlp0520/GitHubRep.git)

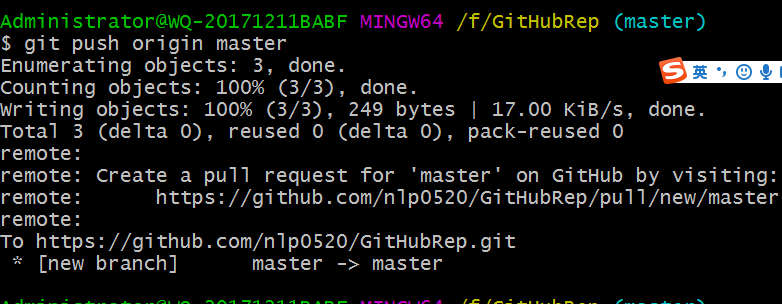
**Add 后的是起的别名**

****

## 推送本地文件

Git push origin master (origin->起的别名,master->分支名字)

输入完命令会出现登录github的账号和密码



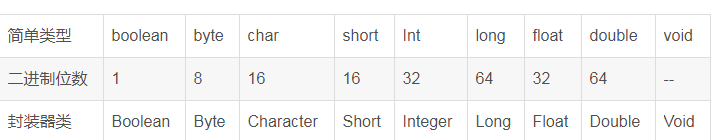
在远程库创建了新的分支

将本地库master分支提交到远程库master分支

# 多线程

# java基础

八大基本类型 一个字节8位



数字默认的类型是Int类型

\n换行 \t制表符 \r回车

运算符的优先级 && > ||

原码 1代表负 0代表正 、 反码 保留第一位其余位置取反 、补码 正数的补码还是本身负数的补码是反码+1

Switch 在JDK1.7之前不能判断String

斐波那切数列：第三位为前两位的和 。用递归算法 （有自身调用自身；有结束）getm()-1 +getm()-2

数组：长度确定 一旦创建不可改变 存储相同类型的集合包括基本数据类型和引用数据类型

声明一个数组就是在内存区开辟了一连串的存储空间。

Int[] s = new int[]{1,2} s在栈里指向new时在堆内存中创建的地址

栈stack存储的是变量：基本类型和引用类型 先进后出

堆 heap存储的是new出来的对象 先进后出

数组是引用类型，当创建数组完成以后是在 方法外定义一个变量，此时数组的值是有默认值的

数据结构：线性表 、非线性表、树、图、队列、栈

排序的时间复杂度：衡量一个数组结构是否合适的衡量标准

排序算法的稳定性：排序后值的跟排序之前的位置是否发生变化

二维数组被称为数组的数组 第一个数组指向第二个数组（可以是非固定数量）

OOP: Object Oriented Programming

对象：实实在在的一个东西例如喝水杯 ； 水杯一个名词是对象的抽象就是类

类是对象的一个抽象，对象是类的一个实例化。

对象 ：一组属性和方法构成。

Jvaa中方法的传参都是值传递。

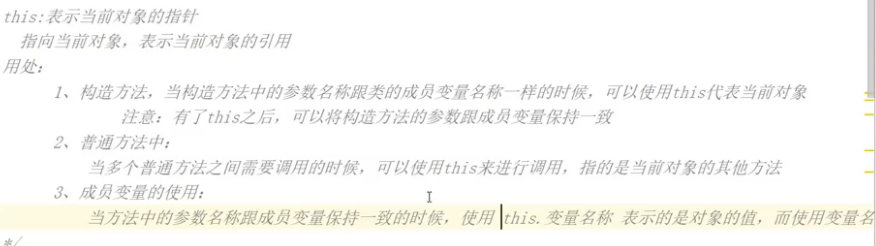
== 和 equals

字符串在比较的时候 == 比较的是地址 ,equals比较的是值

局部变量不包含默认值，使用前必须初始化。

创建对象new时，默认会调用构造方法来创建对象（在heap中开辟空间），可以完成成员变量的初始化操作。

This



Satic:

