Git：版本控制工具

Github：网站、远程代码管理仓库

**集中式 分布式**

集中式缺点：

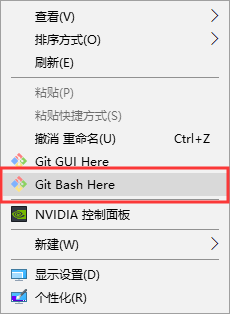
必须联网、比较慢、都是用一个中央服务器，有可能会造成数据丢失

分布式：

不用联网就能进行版本控制，速度快

**初始化版本控制状态：**

1. 找到要控制的文件目录
2. 鼠标右键，找到git bash here



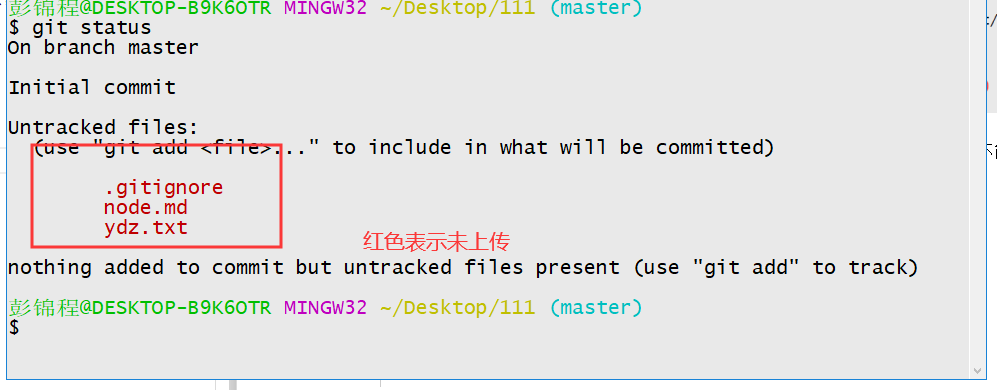
1. 打开控制台，输入git init



版本控制都是基于.git这个隐藏文件的，如果不小心把.git文件删除，就不能进行版本控制

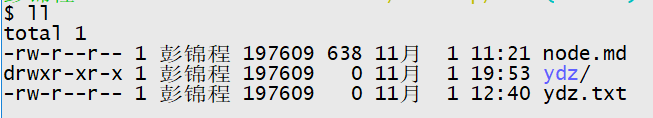
查看git状态

git status



通过Tab键补全文件

通过ll或者ls查看目录的文件





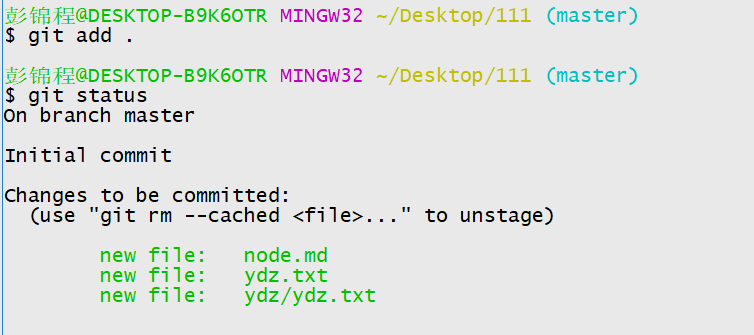
通过上下键去切换刚才输入的命令

工作区到暂存区：

git add 指定文件名

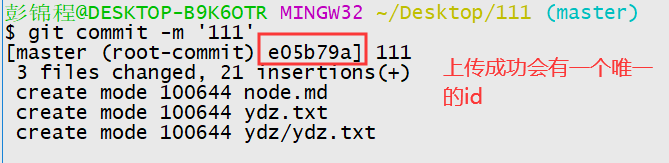


git add . 把当前目录所有文件都放到暂存区



暂存区到版本区

git commit -m ‘注释’



快速把工作区到版本区（已经被管理过的文件）

git commit -a -m ’注释 ’

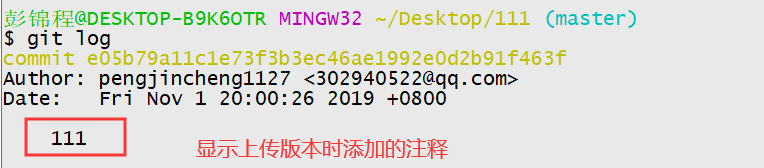


**注意**：

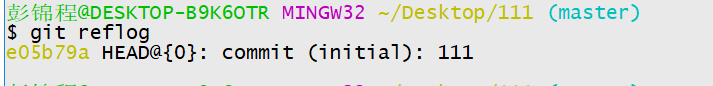
上面这个命令，前提条件是文件已经提交过一次才可以使用

查看版本：

git log



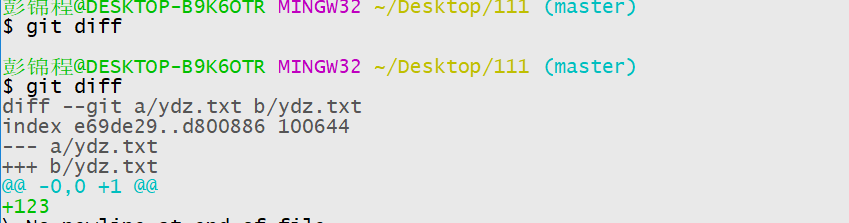
git reflog (操作的多的用这个)



查看每个区域的不同点：

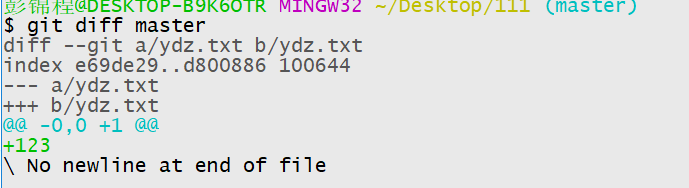
工作区和暂存区的区别

git diff



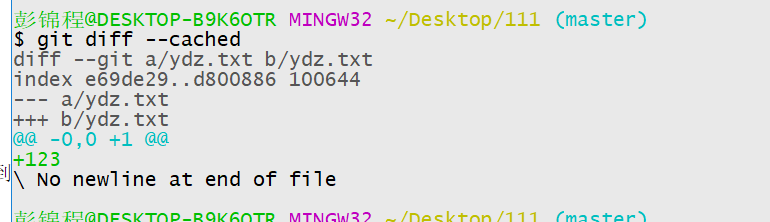
工作区和版本区的区别

git diff master



暂存区跟版本区的区别：

git diff --cached



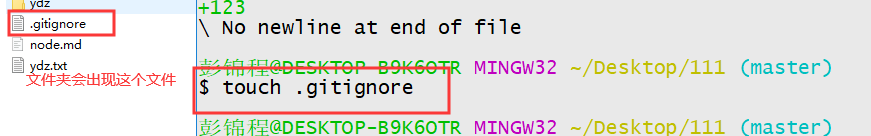
**设置作者信息**

设置用户名 git config --global user.name ”你的名字（英文的）”

设置邮箱 git config --global user.email ”你的邮箱（必须为可以收到邮件的）”

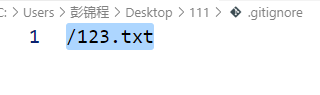
**创建过滤指定文件**

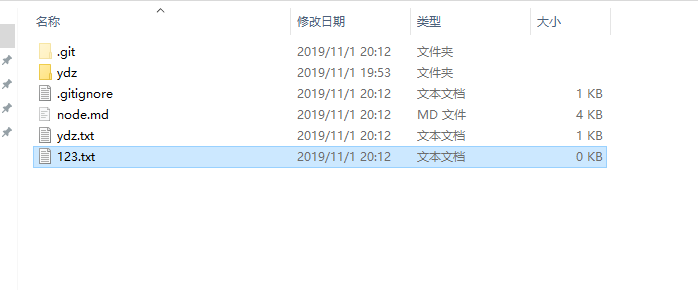
创建一个.gitignore的文件

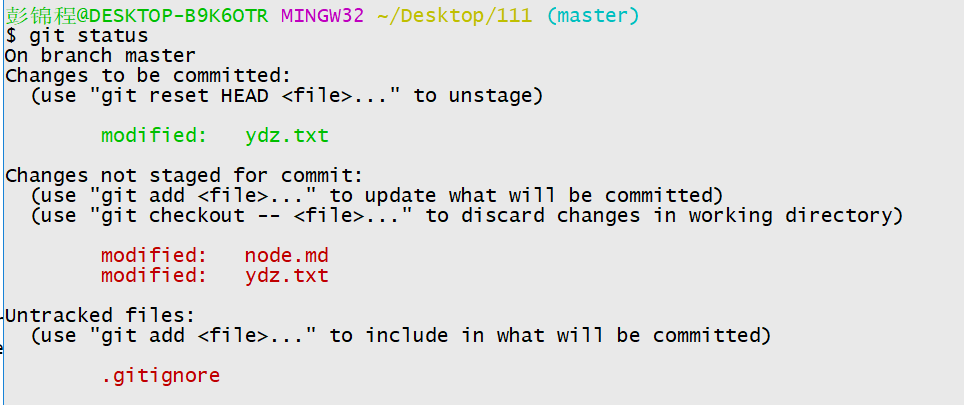


配置过滤项

过滤文件







过滤文件夹



如果说配置不起作用，可以先把之前的操作清除一下，再操作

清除命令

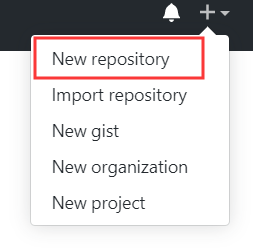
git rm -r --cached

撤销回滚：

git reset --hard 版本ID

**Github**

在github上创建一个项目



1. 绑定密钥

ssh-keygen -t rsa -C “邮箱”

检测是否绑定

ssh -T [git@github.com](mailto:git@github.com)

1. 确定版本库时最新的（没有东西可提交了）
2. 查看远程仓库

git remote -v (使用git init的时候时查不出东西的)

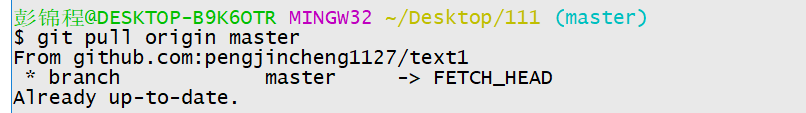


没有添加会什么都不显示

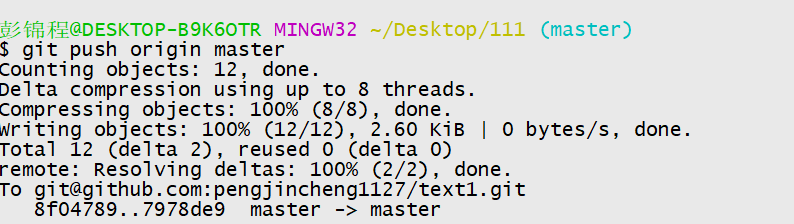
1. 添加远程仓库



1. git pull origin master(保证能够成功上传，进行远程和本地相合并)

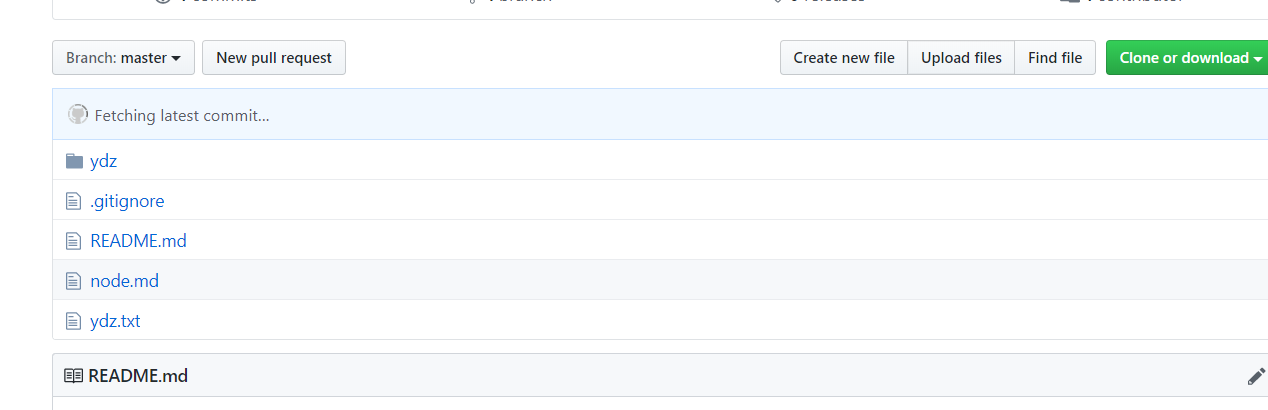


1. git push origin master 上传



**第二种方式：**

1. 现在远程仓库中创建一个项目
2. git clone 项目的地址
3. git add . git commit -m ‘’
4. git push origin master



**Node**

NPM跟着node安装一起安装下来的模块，是目前世界上最大的资源管理平台

Yran 是最快的资源管理平台

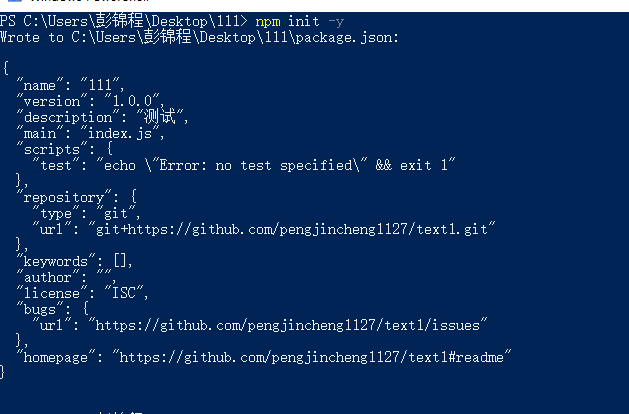
创建项目：

在文件夹按shift+右键选中



npm init -y 会生成一个package.json文件，里面放的都是所有的项目配置依赖





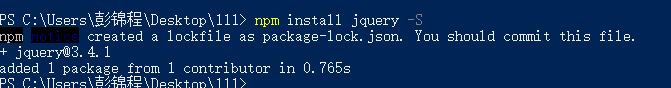
下载资源:

npm install 资源名

-g 全局安装

-S 项目依赖

**例如：**



下载下来的时候会自动生成一个node\_modules的文件夹，文件夹中方的就是你所需要的资源

npm uninstall 删除安装程序

