https://xiedaimala.com/tasks/8c9f20da-c78c-4aed-821c-b671683bb483/text_tutorials/b8c17836-372b-4500-8f4b-e2b702485bd9

前言

其实脚本原本来自戏剧舞台，比如下面这个脚本：

> 公馆一室内
> 王妈：（小心翼翼地）小姐，您还是得注意身子，就吃点东西吧。
> 鸡小姐：（把碗砸在地上）不吃，我就是不吃。
> （王妈下）

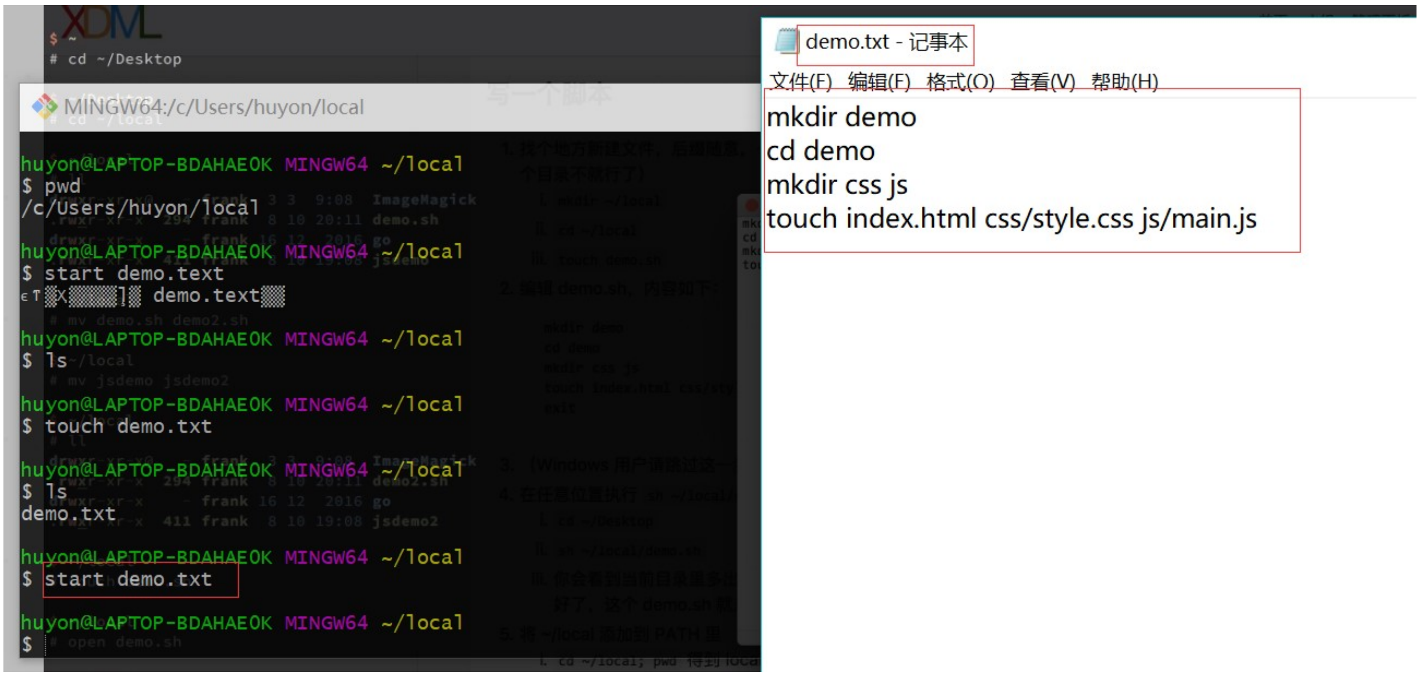
脚本主要由人物对话和舞台提示组成。演员和道具组只需要按照脚本说的做即可。

编程领域的脚本也是类似的，计算机只要照着脚本上说的做即可，比如下面这个脚本：

```
cd ~/Desktop
mkdir demo
cd demo
echo "hi" > index.html
cd ~/Desktop

```

所以说，脚本就是给计算机照着做的。这是我们对「脚本」的一个感性认识。

例子

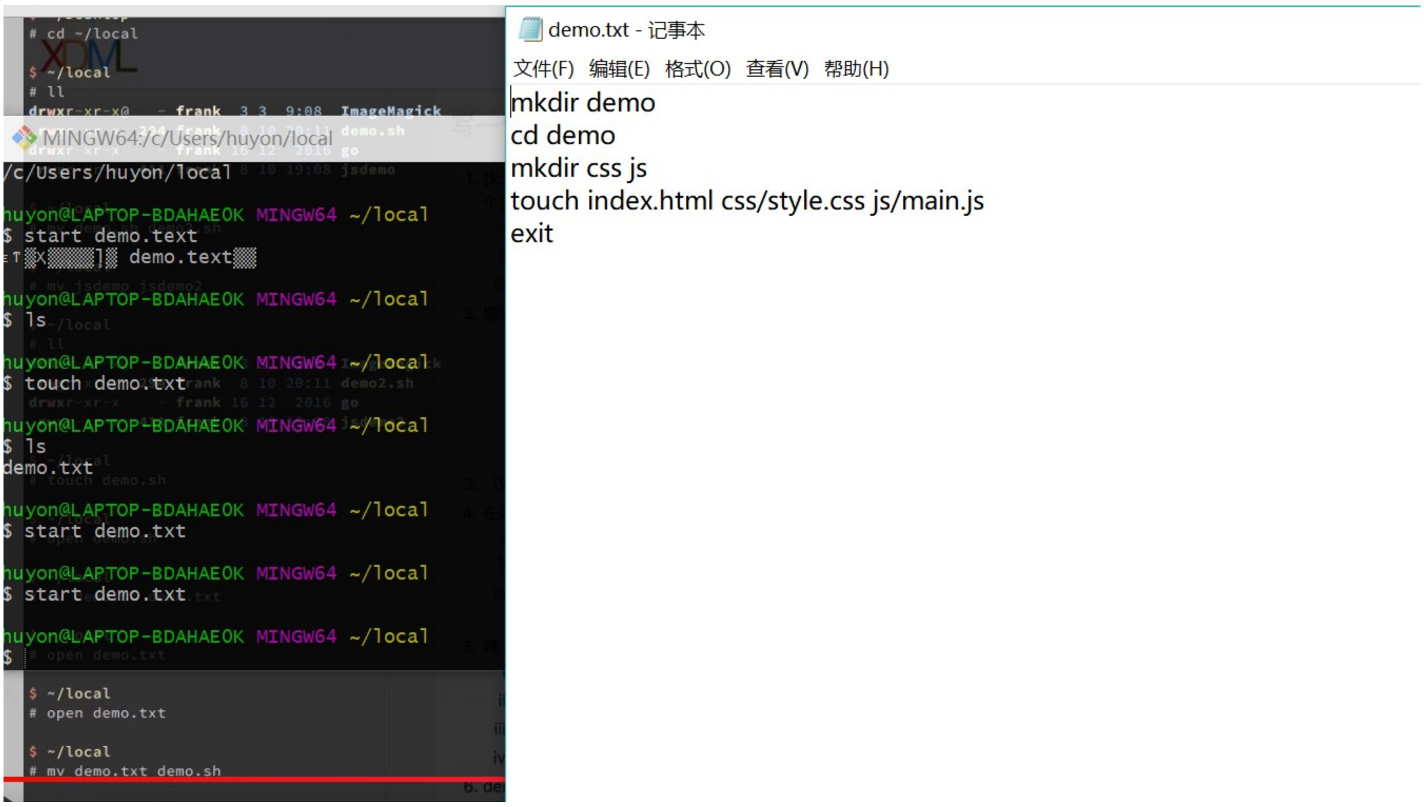

MINGW64:/c/Users/huyon/local

```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ pwd
/c/Users/huyon/local

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.text
ϵT▓X▓▓▓]▓ demo.text▓

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ touch demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```

demo.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
mkdir demo
cd demo
mkdir css js
touch index.html css/style.css js/main.js
```

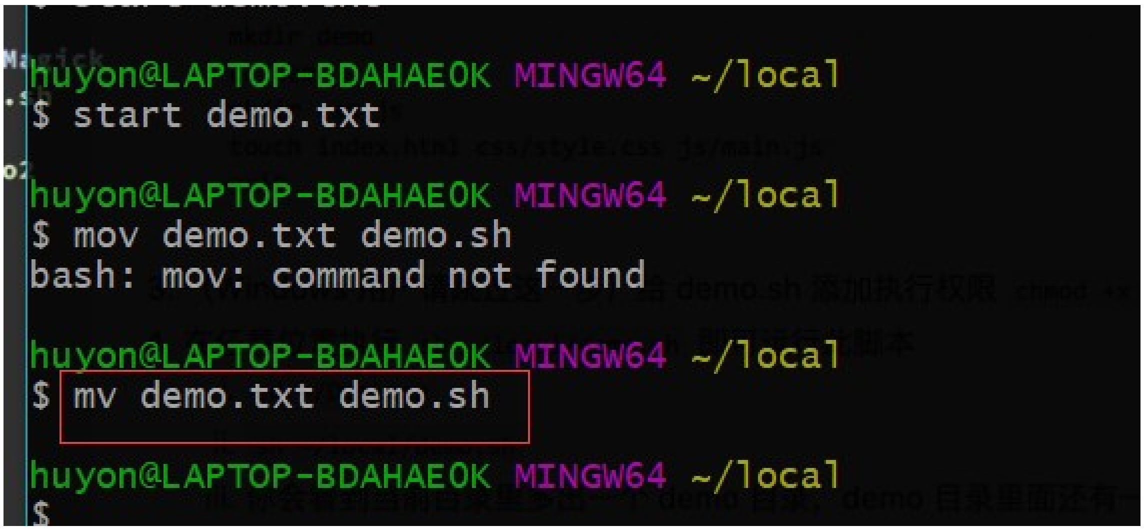

MINGW64:/c/Users/huyon/local

```
/c/Users/huyon/local

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.text
ϵT▓X▓▓▓]▓ demo.text▓

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ touch demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```

demo.txt - 记事本

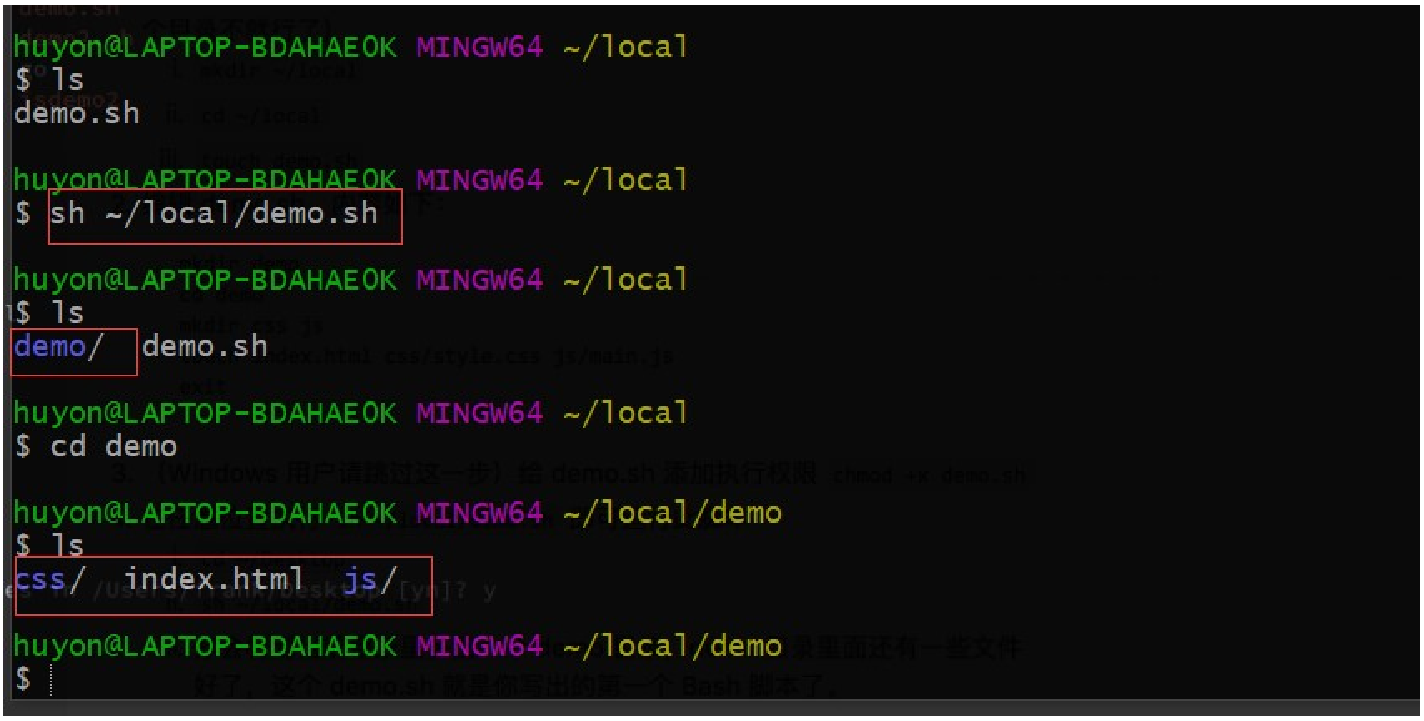
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
mkdir demo
cd demo
mkdir css js
touch index.html css/style.css js/main.js
exit
```



```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start demo.txt

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ mov demo.txt demo.sh
bash: mov: command not found

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ mv demo.txt demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```

非windows系统还要加上chmod +x filename 修改权限

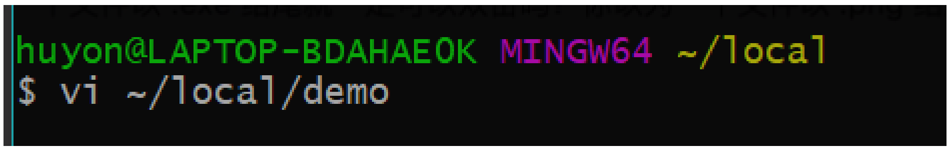


```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ sh ~/local/demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo/  demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ cd demo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/demo
$ ls
css/  index.html  js/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/demo
$
```

几个命令

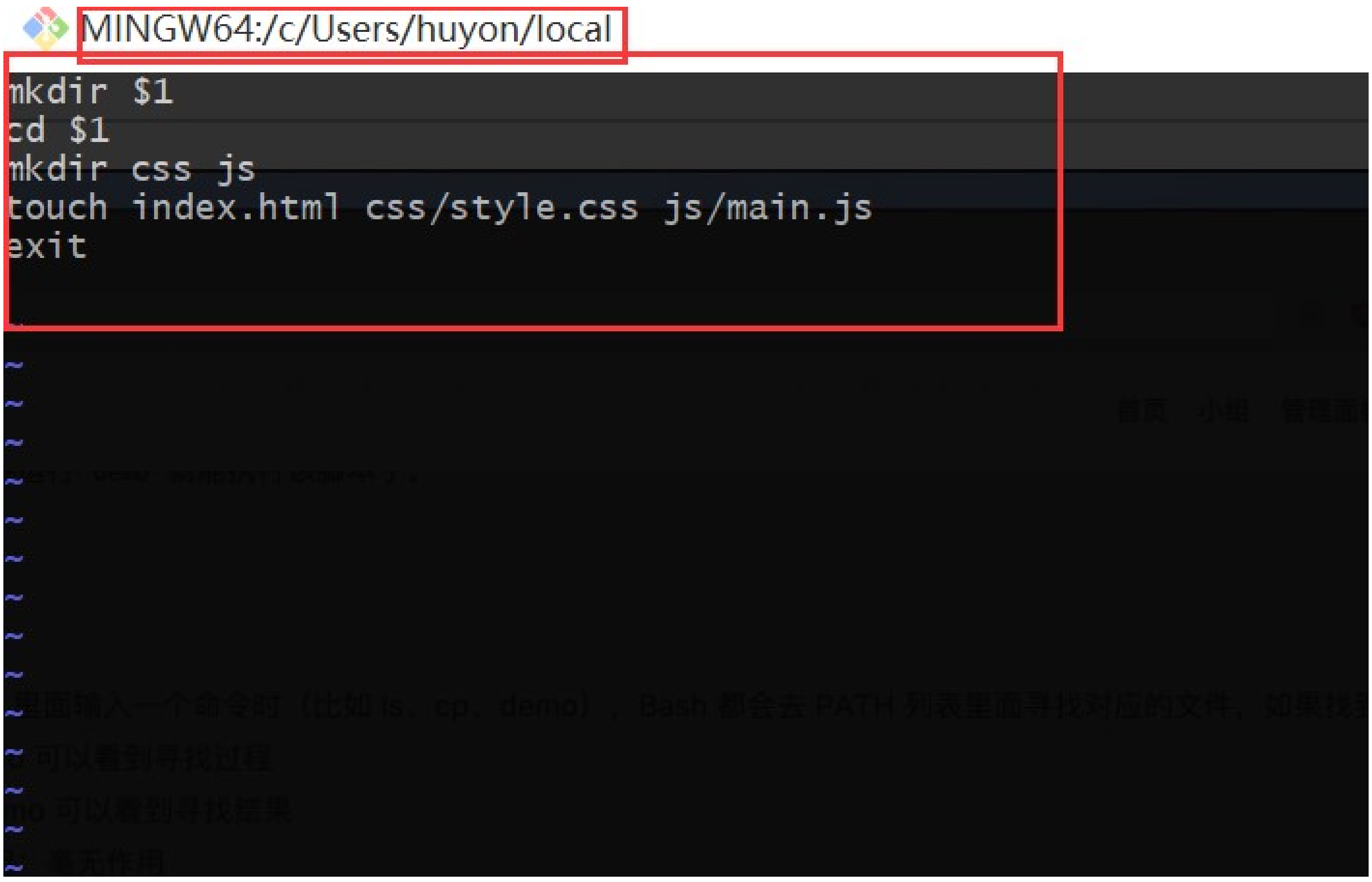
1. PATH 的作用
   你每次在 Bash 里面输入一个命令时（比如 ls、cp、demo），Bash 都会去 PATH 列表里面寻找对应的文件，如果找到了就执行。(所有命令都是脚本文件而已！linux里面可执行 文件皆命令,不可执行文件就是配置！)
2. 使用 type demo 可以看到寻找过程
3. 使用 which demo 可以看到寻找结果
4. 文件后缀的作用：毫无作用
   你以为一个文件以 .exe 结尾就一定可以双击吗？你以为一个文件以 .png 结尾就一定是图片吗？图样图森破！

创建任意目录的脚本



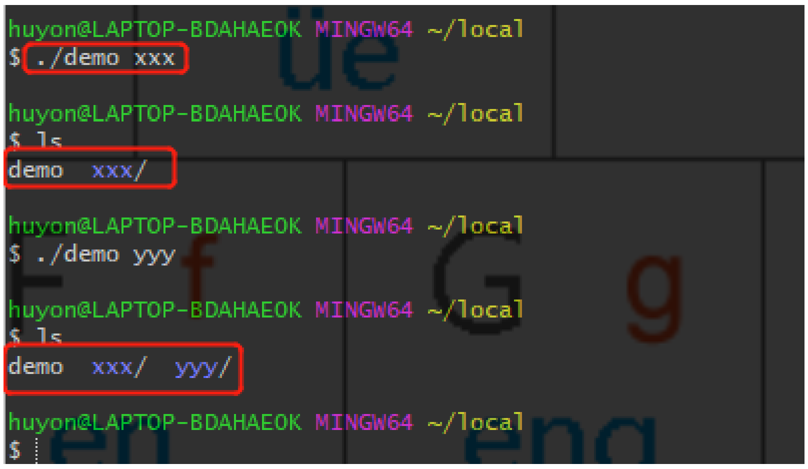
```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ vi ~/local/demo
```

```
mkdir $1
cd $1
mkdir css js
touch index.html css/style.css js/main.js
exit


```

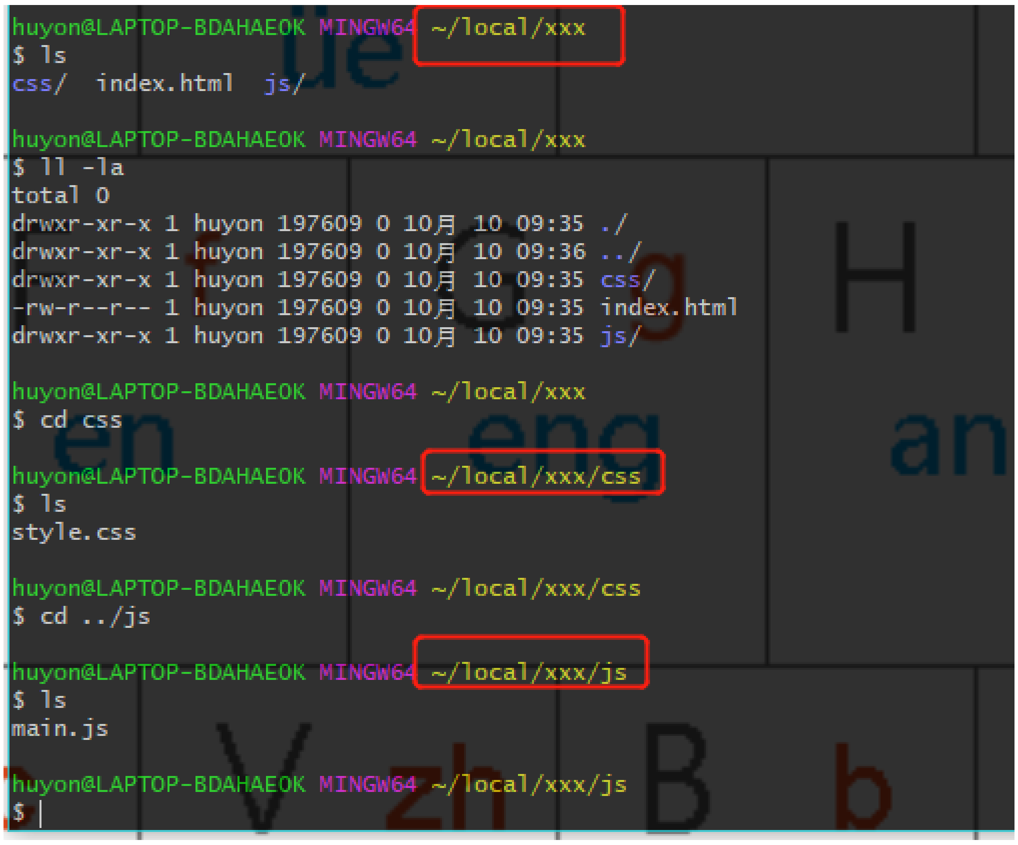


MINGW64:/c/Users/huyon/local

```
mkdir $1
cd $1
mkdir css js
touch index.html css/style.css js/main.js
exit
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

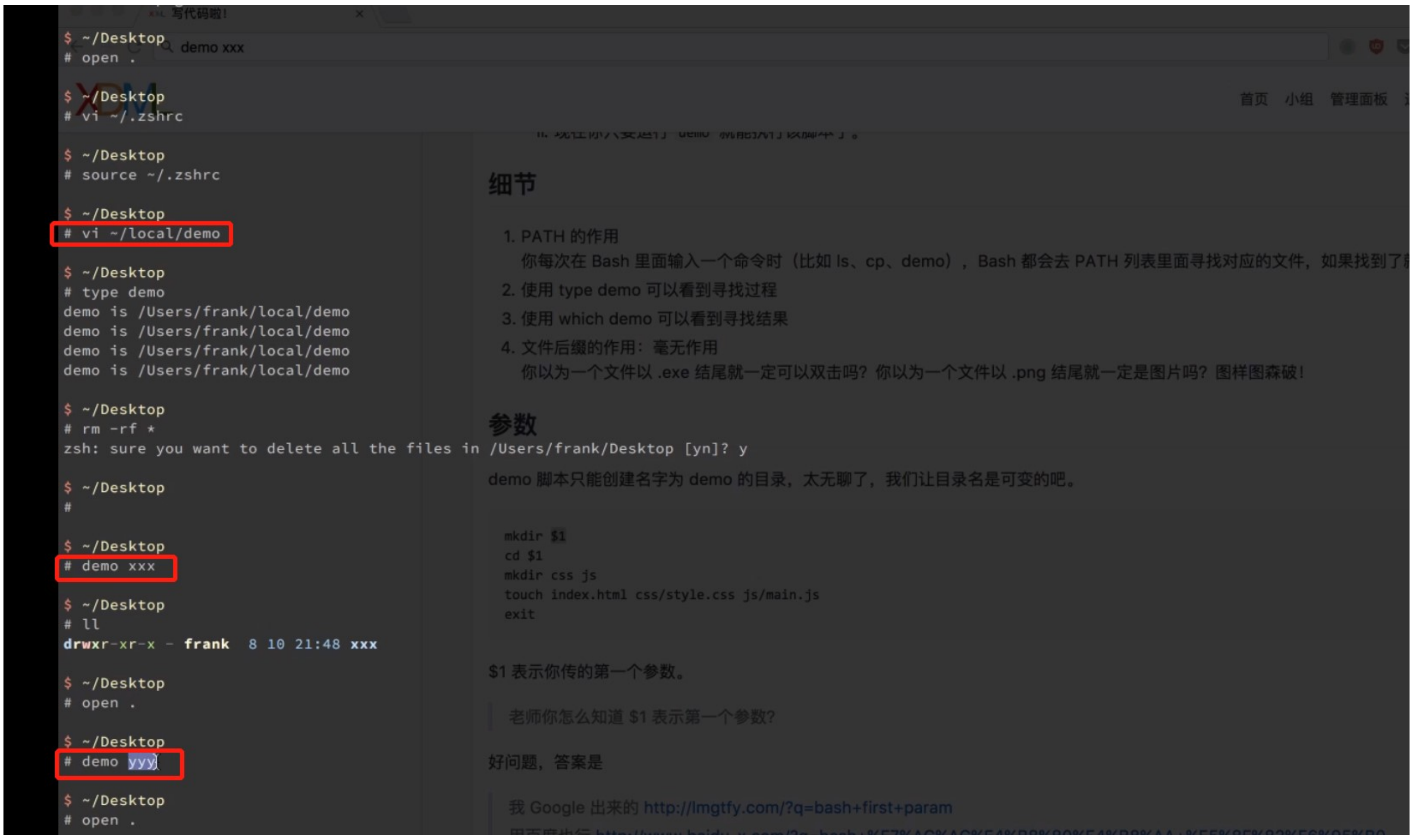


```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ./demo xxx

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo  xxx/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ./demo yyy

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo  xxx/  yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```



```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx
$ ls
css/  index.html  js/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx
$ ll -la
total 0
drwxr-xr-x 1 huyon 197609 0 10月 10 09:35 ./
drwxr-xr-x 1 huyon 197609 0 10月 10 09:36 ../
drwxr-xr-x 1 huyon 197609 0 10月 10 09:35 css/
-rw-r--r-- 1 huyon 197609 0 10月 10 09:35 index.html
drwxr-xr-x 1 huyon 197609 0 10月 10 09:35 js/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx
$ cd css

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx/css
$ ls
style.css

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx/css
$ cd ../js

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx/js
$ ls
main.js

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/xxx/js
$ |
```

注意MAC 和Linux(以及windows)有不一样的地方,执行文件需要加 `./`



```
$ ~/Desktop
# open .

$ ~/Desktop
# vi ~/.zshrc

$ ~/Desktop
# source ~/.zshrc

$ ~/Desktop
# vi ~/local/demo

$ ~/Desktop
# type demo
demo is /Users/frank/local/demo
demo is /Users/frank/local/demo
demo is /Users/frank/local/demo
demo is /Users/frank/local/demo

$ ~/Desktop
# rm -rf *
zsh: sure you want to delete all the files in /Users/frank/Desktop [yn]? y

$ ~/Desktop
#

$ ~/Desktop
# demo xxx

$ ~/Desktop
# ll
drwxr-xr-x - frank  8 10 21:48 xxx

$ ~/Desktop
# open .

$ ~/Desktop
# demo yyy

$ ~/Desktop
# open .
```

写代码啦!

demo xxx

首页 小组 管理面板

细节

1. PATH 的作用
   你每次在 Bash 里面输入一个命令时（比如 ls、cp、demo），Bash 都会去 PATH 列表里面寻找对应的文件，如果找到了
2. 使用 type demo 可以看到寻找过程
3. 使用 which demo 可以看到寻找结果
4. 文件后缀的作用：毫无作用
   你以为一个文件以 .exe 结尾就一定可以双击吗？你以为一个文件以 .png 结尾就一定是图片吗？图样图森破！

参数

demo 脚本只能创建名字为 demo 的目录，太无聊了，我们让目录名是可变的吧。

```
mkdir $1
cd $1
mkdir css js
touch index.html css/style.css js/main.js
exit
```

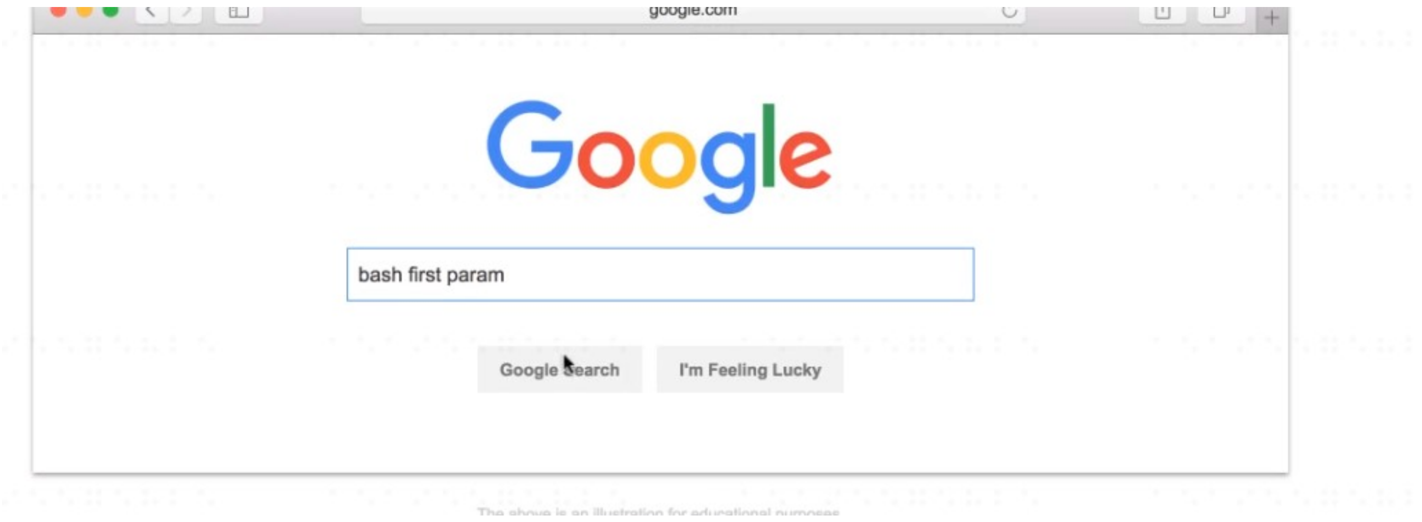
$1 表示你传的第一个参数。

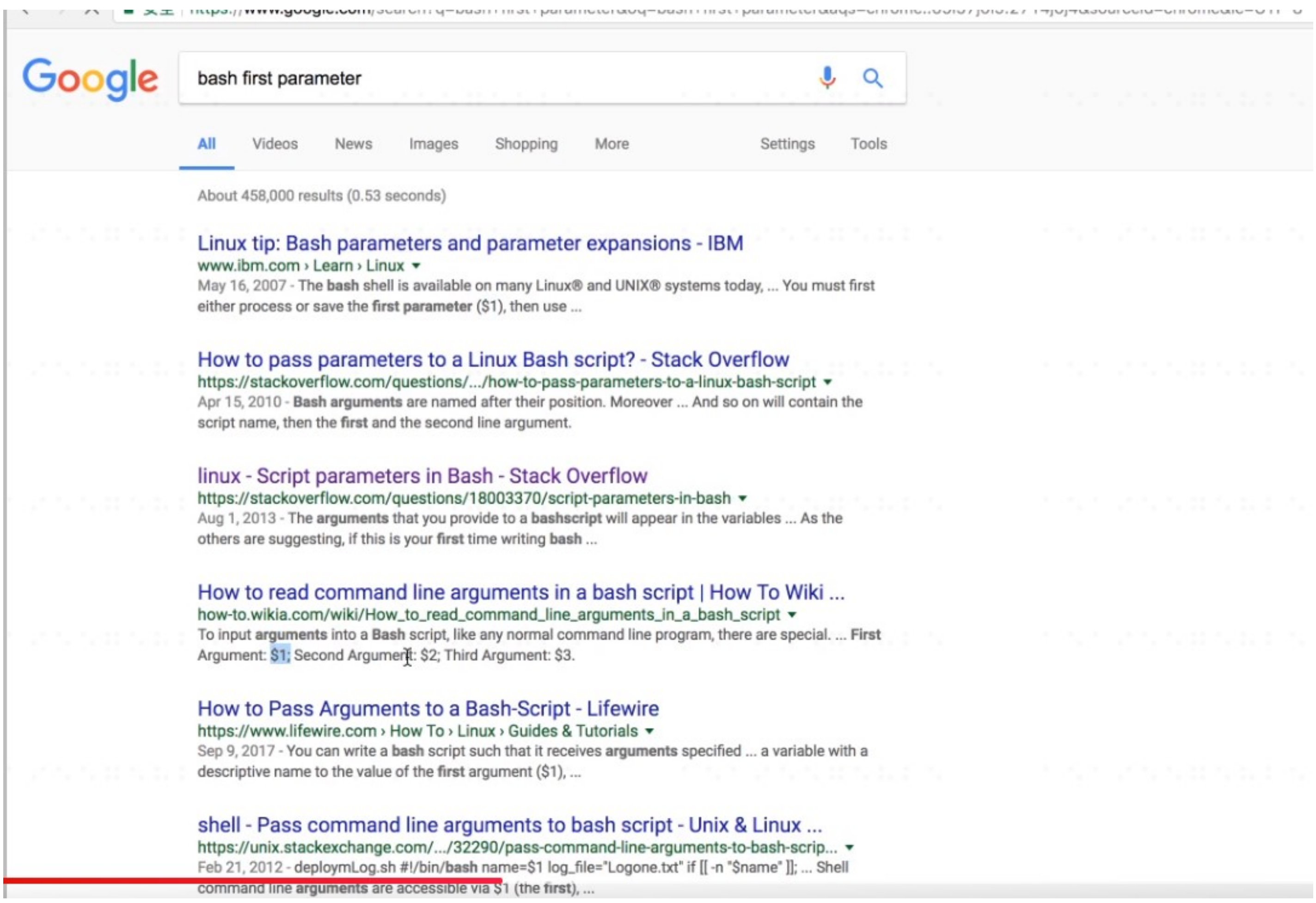
> 老师你怎么知道 $1 表示第一个参数?

好问题，答案是

> 我 Google 出来的 http://lmgtfy.com/?q=bash+first+param

原理讲解





About 458,000 results (0.53 seconds)

Linux tip: Bash parameters and parameter expansions - IBM
www.ibm.com › Learn › Linux
May 16, 2007 - The **bash** shell is available on many Linux® and UNIX® systems today, ... You must first either process or save the **first parameter** ($1), then use ...

How to pass parameters to a Linux Bash script? - Stack Overflow
https://stackoverflow.com/questions/.../how-to-pass-parameters-to-a-linux-bash-script
Apr 15, 2010 - **Bash arguments** are named after their position. Moreover ... And so on will contain the script name, then the **first** and the second line argument.

linux - Script parameters in Bash - Stack Overflow
https://stackoverflow.com/questions/18003370/script-parameters-in-bash
Aug 1, 2013 - The **arguments** that you provide to a **bashscript** will appear in the variables ... As the others are suggesting, if this is your **first** time writing **bash** ...

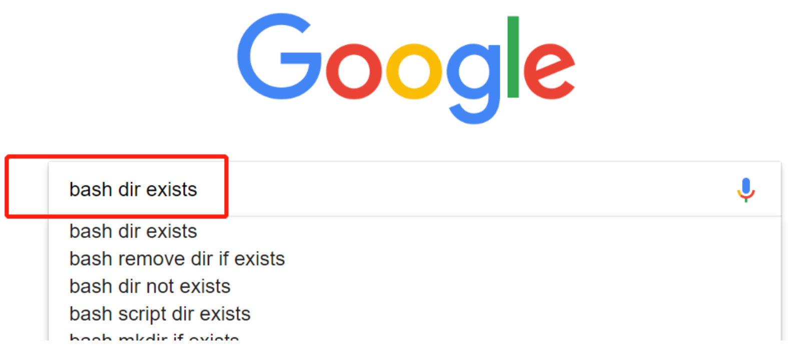
How to read command line arguments in a bash script | How To Wiki ...
how-to.wikia.com/wiki/How_to_read_command_line_arguments_in_a_bash_script
To input **arguments** into a **Bash** script, like any normal command line program, there are special. ... **First** Argument: $1; Second Argument: $2; Third Argument: $3.

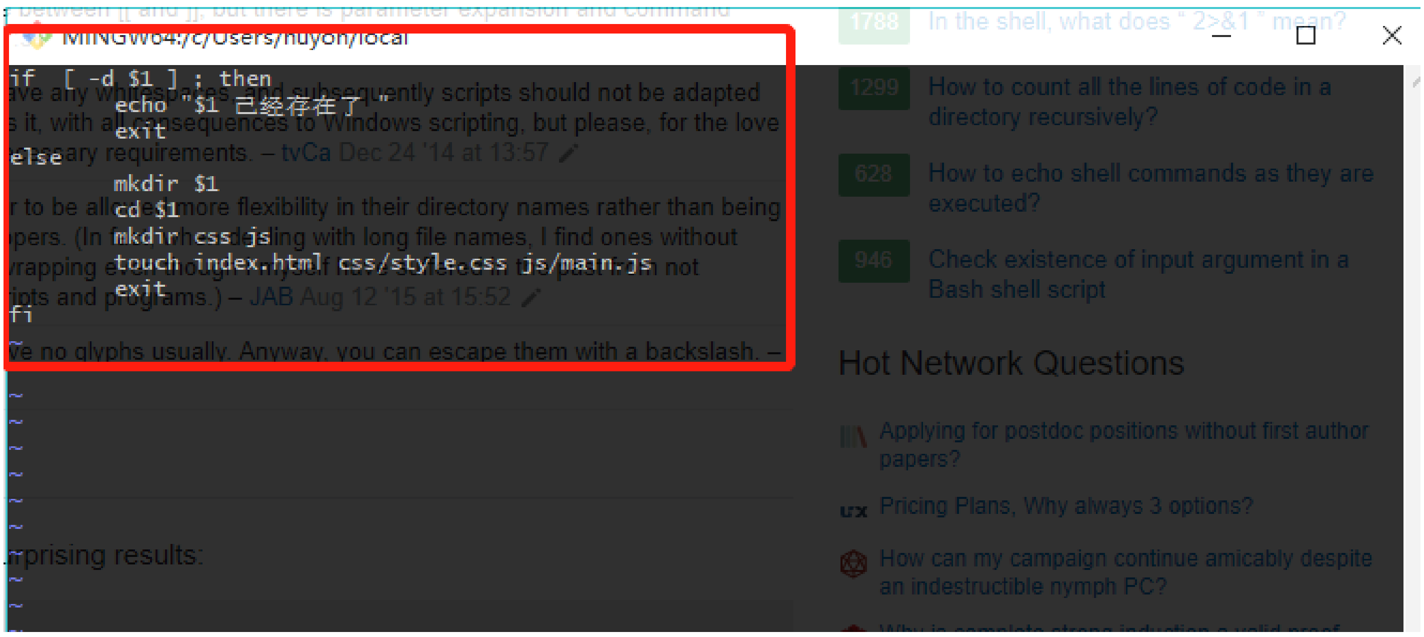
How to Pass Arguments to a Bash-Script - Lifewire
https://www.lifewire.com › How To › Linux › Guides & Tutorials
Sep 9, 2017 - You can write a **bash** script such that it receives **arguments** specified ... a variable with a descriptive name to the value of the **first** argument ($1), ...

shell - Pass command line arguments to bash script - Unix & Linux ...
https://unix.stackexchange.com/.../32290/pass-command-line-arguments-to-bash-scrip...
Feb 21, 2012 - deploymLog.sh #!/bin/**bash** name=$1 log_file="Logone.txt" if [[-n "$name"]]; ... Shell command line **arguments** are accessible via $1 (the **first**), ...

上面的 `$1` 为第一个参数

判断目录是否存在




MINGW64:/c/Users/huyon/local

```
if  [ -d $1 ] ; then
        echo "$1 已经存在了 "
        exit
else
        mkdir $1
        cd $1
        mkdir css js
        touch index.html css/style.css js/main.js
        exit
fi
~
~
~
~
~
~
~
~
~
~
```

1788 In the shell, what does " 2>&1 " mean?

1299 How to count all the lines of code in a directory recursively?

628 How to echo shell commands as they are executed?

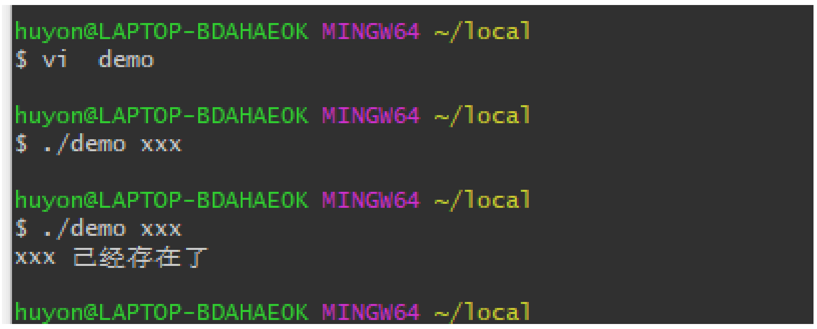
946 Check existence of input argument in a Bash shell script

Hot Network Questions

Applying for postdoc positions without first author papers?

Pricing Plans, Why always 3 options?

How can my campaign continue amicably despite an indestructible nymph PC?



```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ vi  demo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ./demo xxx

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ./demo xxx
xxx 已经存在了

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
```

```
if [ -d $1 ] ; then
    echo "$1 已经存在了 "
    exit
else
    mkdir $1
    cd $1
    mkdir css js
    touch index.html css/style.css js/main.js
    echo 'success'
    exit
fi
```

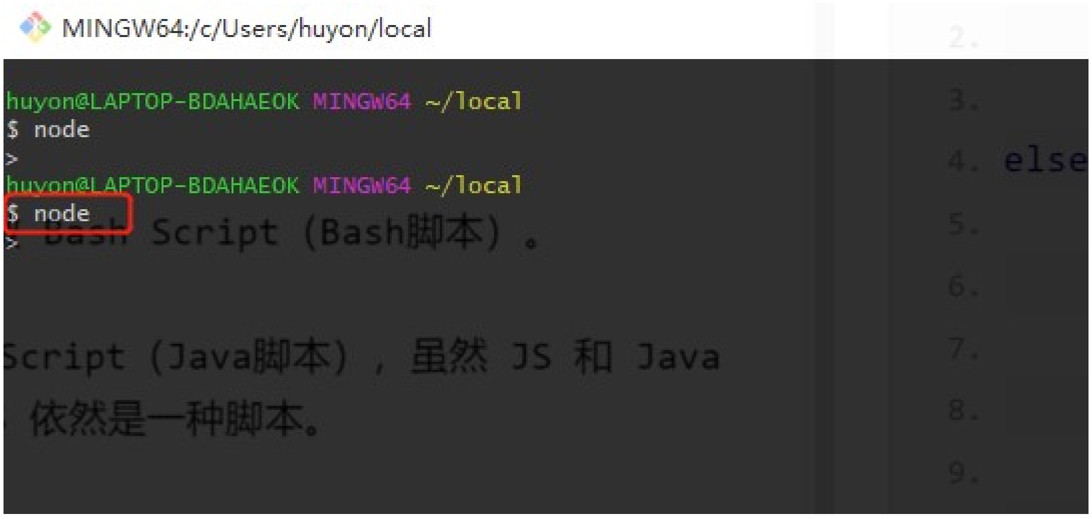
Node.js写脚本(ctrl +D 退出)

上面我们写的脚本叫做 Bash Script（Bash脚本）。

JS 的全称叫做 JavaScript（Java脚本），虽然 JS 和 Java 没什么关系，但是 JS 依然是一种脚本。

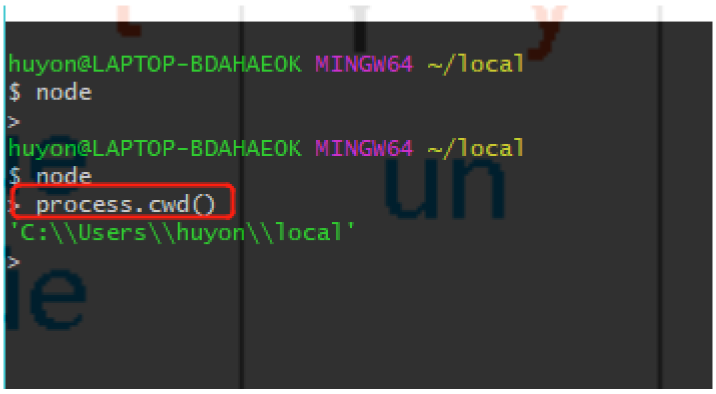
1. 我们在 Bash 命令行里输入 Bash 命令，也可以在 Node.js 命令行里输入 JS 命令（`Ctrl` + D 退出）
2. Bash 脚本能做的事情，JS 脚本也能做。(`sh demo.sh` 对应 `node demo.js`)

进入js命令行:**(ctrl +D 退出)**(其语法与bash完全不一样)


```
MINGW64:/c/Users/huyon/local

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
>
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
>
```

js里面的显示当前目录与切换目录

`process.cwd()` <--------> `pwd` (bash里面的)


```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
>
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
> process.cwd()
'C:\\Users\\huyon\\local'
>
```

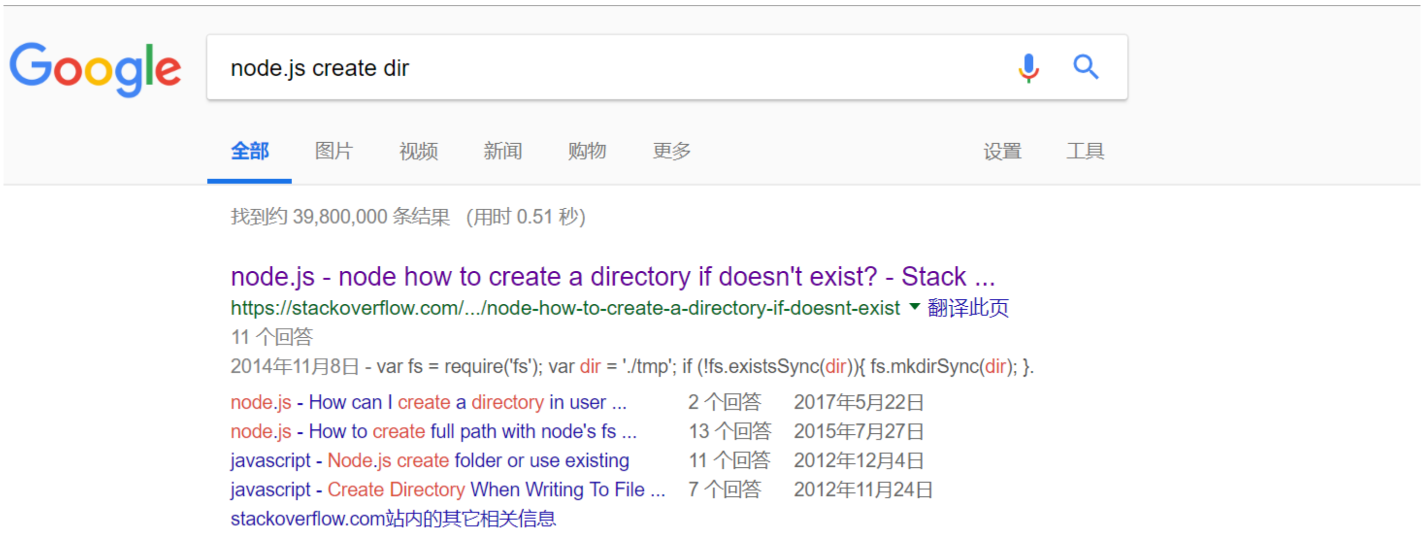
```
process.chdir("/Users/frank/Desktop")
```

chdir类似于change dir

js里面的print

`console.log(1)` <---------> `echo 1`

js里面的建立目录



node.js create dir

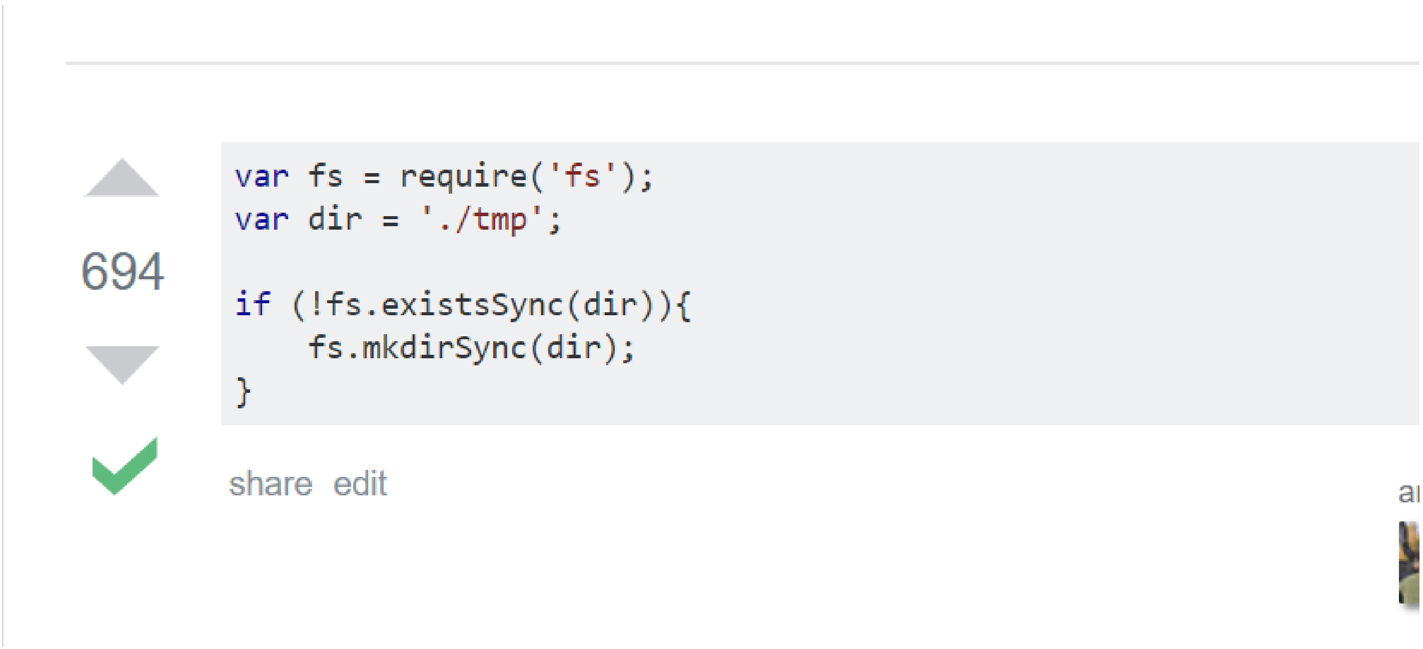
全部　图片　视频　新闻　购物　更多　设置　工具

找到约 39,800,000 条结果 （用时 0.51 秒）

node.js - node how to create a directory if doesn't exist? - Stack ...
https://stackoverflow.com/.../node-how-to-create-a-directory-if-doesnt-exist ▾ 翻译此页
11 个回答
2014年11月8日 - var fs = require('fs'); var dir = './tmp'; if (!fs.existsSync(dir)){ fs.mkdirSync(dir); }.

node.js - How can I create a directory in user ...	2 个回答	2017年5月22日
node.js - How to create full path with node's fs ...	13 个回答	2015年7月27日
javascript - Node.js create folder or use existing	11 个回答	2012年12月4日
javascript - Create Directory When Writing To File ...	7 个回答	2012年11月24日

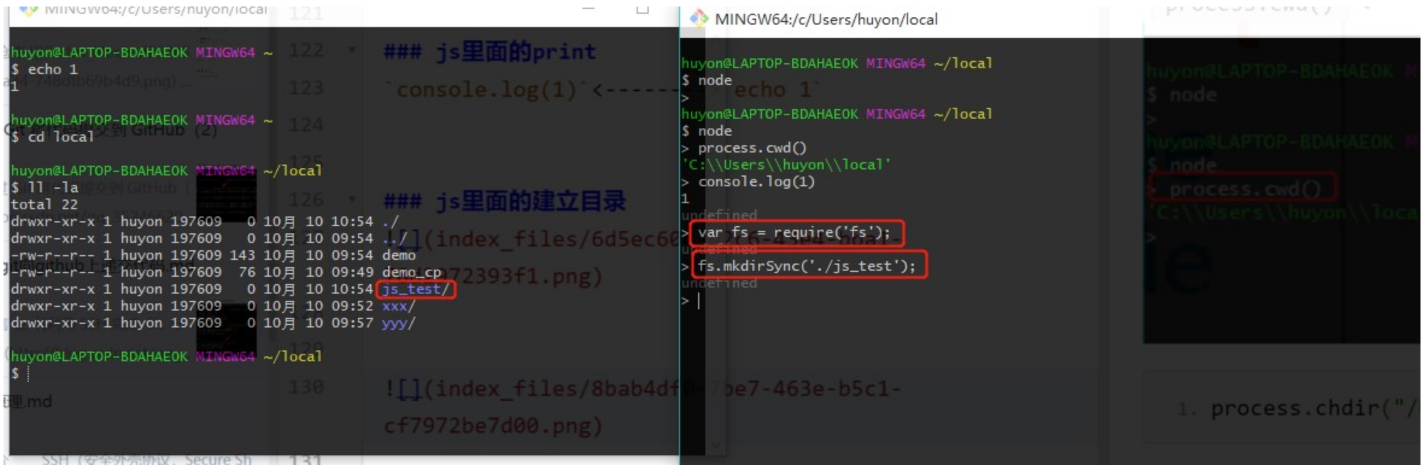
stackoverflow.com站内的其它相关信息



694

```
var fs = require('fs');
var dir = './tmp';

if (!fs.existsSync(dir)){
    fs.mkdirSync(dir);
}
```

share edit



MINGW64:/c/Users/huyon/local

```
huyon@LAPTOP-BDAHAEOK MINGW64 ~
$ echo 1
1

huyon@LAPTOP-BDAHAEOK MINGW64 ~
$ cd local

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ll -la
total 22
drwxr-xr-x 1 huyon 197609   0 10月 10 10:54 ./
drwxr-xr-x 1 huyon 197609   0 10月 10 09:54 ../
-rw-r--r-- 1 huyon 197609 143 10月 10 09:54 demo
-rw-r--r-- 1 huyon 197609  76 10月 10 09:49 demo_cp
drwxr-xr-x 1 huyon 197609   0 10月 10 10:54 js_test/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:52 xxx/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:57 yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```

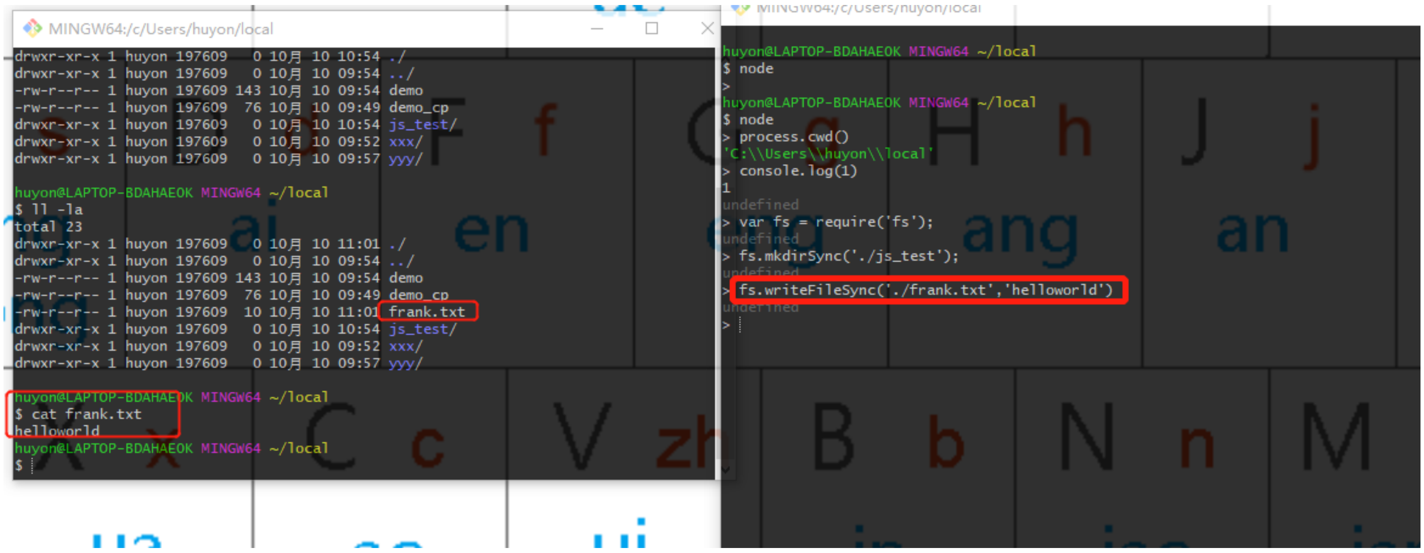
MINGW64:/c/Users/huyon/local

```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
>
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
> process.cwd()
'C:\\Users\\huyon\\local'
> console.log(1)
1
undefined
> var fs = require('fs');
undefined
> fs.mkdirSync('./js_test');
undefined
> |
```

bash命令行和Node.js命令行实质上是一样的,只不过语法格式不一样!

Node.js文档见下:
https://nodejs.org/api/fs.html#fs_fs_mkdirsync_path_mode

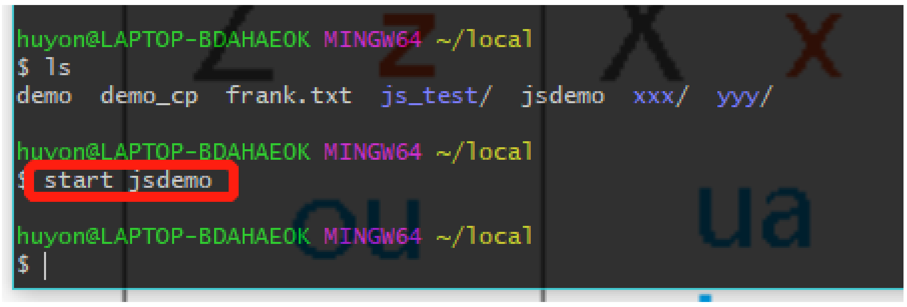
js里面读写



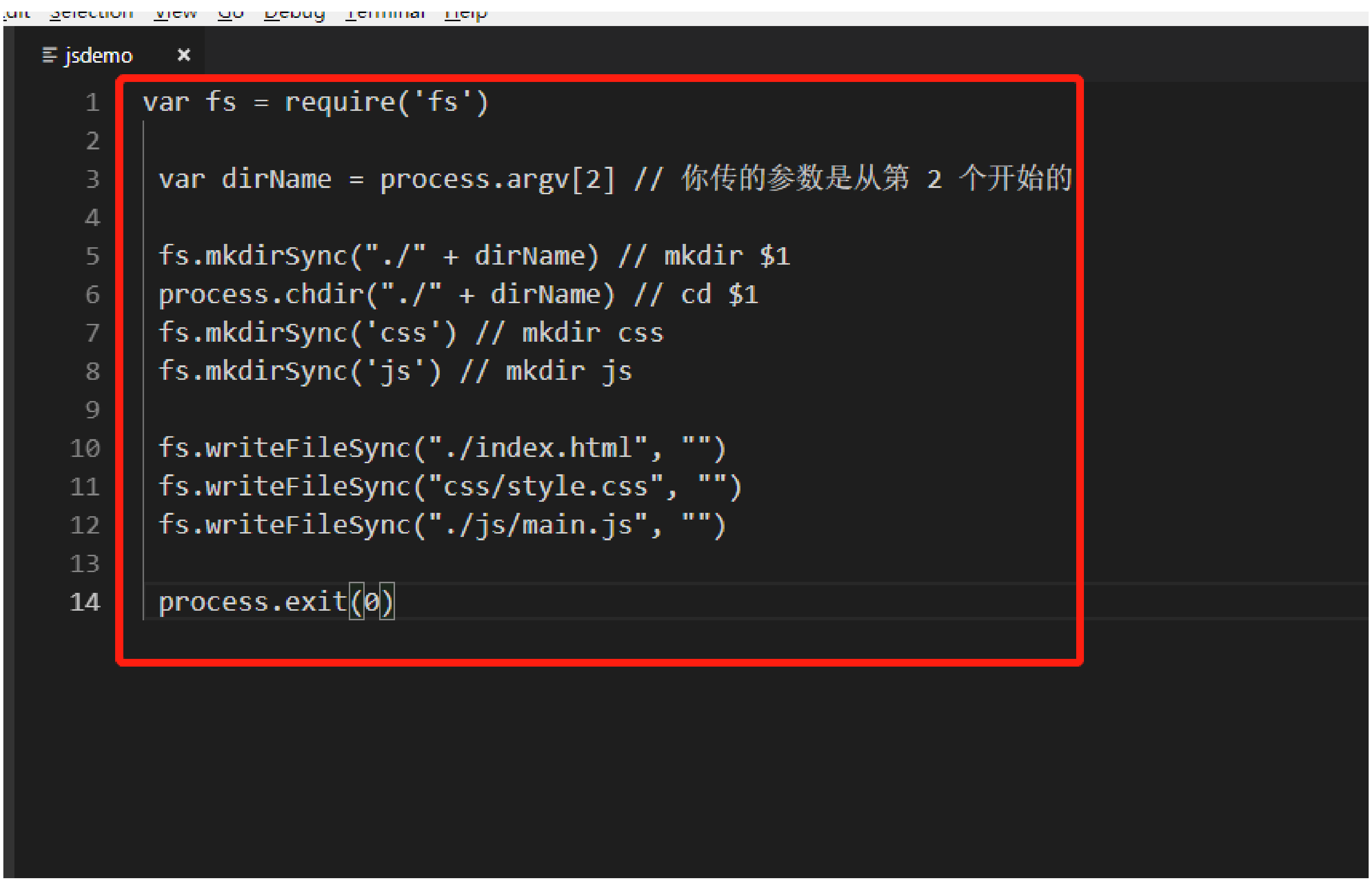
```
MINGW64:/c/Users/huyon/local
drwxr-xr-x 1 huyon 197609   0 10月 10 10:54 ./
drwxr-xr-x 1 huyon 197609   0 10月 10 09:54 ../
-rw-r--r-- 1 huyon 197609 143 10月 10 09:54 demo
-rw-r--r-- 1 huyon 197609  76 10月 10 09:49 demo_cp
drwxr-xr-x 1 huyon 197609   0 10月 10 10:54 js_test/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:52 xxx/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:57 yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ll -la
total 23
drwxr-xr-x 1 huyon 197609   0 10月 10 11:01 ./
drwxr-xr-x 1 huyon 197609   0 10月 10 09:54 ../
-rw-r--r-- 1 huyon 197609 143 10月 10 09:54 demo
-rw-r--r-- 1 huyon 197609  76 10月 10 09:49 demo_cp
-rw-r--r-- 1 huyon 197609  10 10月 10 11:01 frank.txt
drwxr-xr-x 1 huyon 197609   0 10月 10 10:54 js_test/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:52 xxx/
drwxr-xr-x 1 huyon 197609   0 10月 10 09:57 yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ cat frank.txt
helloworld
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$
```

```
MINGW64:/c/Users/huyon/local
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
>
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node
> process.cwd()
'C:\\Users\\huyon\\local'
> console.log(1)
1
undefined
> var fs = require('fs');
undefined
> fs.mkdirSync('./js_test');
undefined
> fs.writeFileSync('./frank.txt','helloworld')
undefined
>
```

Node.js脚本

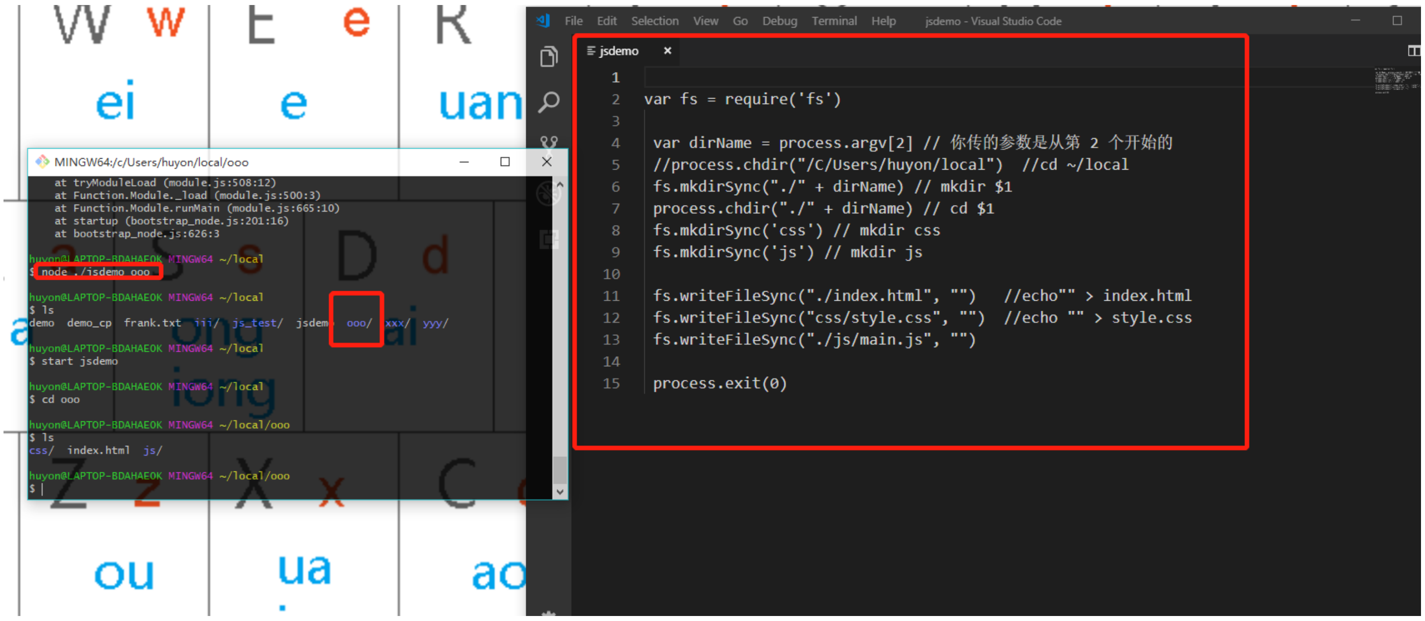


```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo  demo_cp  frank.txt  js_test/  jsdemo  xxx/  yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start jsdemo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ |
```



```
var fs = require('fs')

 var dirName = process.argv[2] // 你传的参数是从第 2 个开始的

 fs.mkdirSync("./" + dirName) // mkdir $1
 process.chdir("./" + dirName) // cd $1
 fs.mkdirSync('css') // mkdir css
 fs.mkdirSync('js') // mkdir js

 fs.writeFileSync("./index.html", "")
 fs.writeFileSync("css/style.css", "")
 fs.writeFileSync("./js/main.js", "")

 process.exit(0)
```

一般默认皆是用bash来执行的,故这里要添加一个node



```
MINGW64:/c/Users/huyon/local/ooo
    at tryModuleLoad (module.js:508:12)
    at Function.Module._load (module.js:500:3)
    at Function.Module.runMain (module.js:665:10)
    at startup (bootstrap_node.js:201:16)
    at bootstrap_node.js:626:3

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ node ./jsdemo ooo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ ls
demo  demo_cp  frank.txt  iii/  js_test/  jsdemo  ooo/  xxx/  yyy/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ start jsdemo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local
$ cd ooo

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/ooo
$ ls
css/  index.html  js/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/local/ooo
$ |
```

```js

var fs = require('fs')

 var dirName = process.argv[2] // 你传的参数是从第 2 个开始的
 //process.chdir("/C/Users/huyon/local")  //cd ~/local
 fs.mkdirSync("./" + dirName) // mkdir $1
 process.chdir("./" + dirName) // cd $1
 fs.mkdirSync('css') // mkdir css
 fs.mkdirSync('js') // mkdir js

 fs.writeFileSync("./index.html", "")   //echo"" > index.html
 fs.writeFileSync("css/style.css", "")  //echo "" > style.css
 fs.writeFileSync("./js/main.js", "")

 process.exit(0)
```

shebang

我们每次执行 ~/local/jsdemo.js 都要用 node 来执行，能不能做到不加 node 也能执行呢（也就是指定执行环境），可以，在 jsdemo.js 第一行加上这一句即可：

```
#!/usr/bin/env node

```

（以下操作在 Windows 上可能失败，失败了就算了）

1. 然后你就可以直接用 `~/local/jsdemo.js zzz` 了（省得输入 node 了）。
2. 如果你已经把 ~/local 加入了 PATH，那么甚至可以直接输入 `jsdemo.js zzz` 来执行。
3. 如果你再把 jsdemo.js 的后缀 .js 去掉，就可以直接 `jsdemo zzz` 了。

注意，你每次执行前最好删掉 zzz 目录，以免发生冲突。

总结

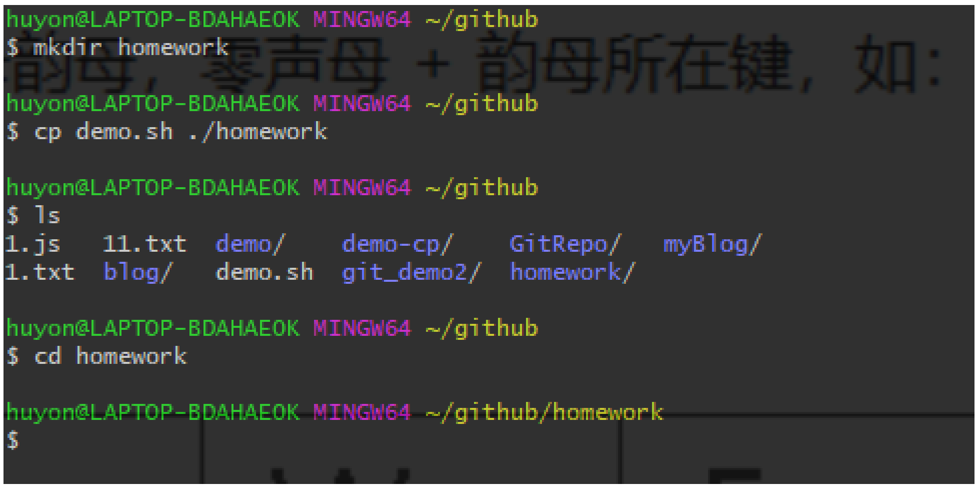
1. 脚本就是给机器一行一行执行的文本
2. Bash 脚本有 Bash 脚本的语法，Node.js 脚本有 JS 语法
3. 不管是那种脚本，能实现的功能都差不多，只是语法不同
4. Bash 脚本的语法挺奇葩的，比如 1# 等符号
5. 不用特别去学 Bash 脚本的用法，遇到不会的就 Google
6. 不用特别去学 Node.js 脚本的用法，遇到不会的就 Google
7. 新人写代码最大的问题就是「抄错了」
   a. 多写了一个空格
   b. 少写了一个空格
   c. 单词拼错了

d. 没有加分号

e. 多加了分号

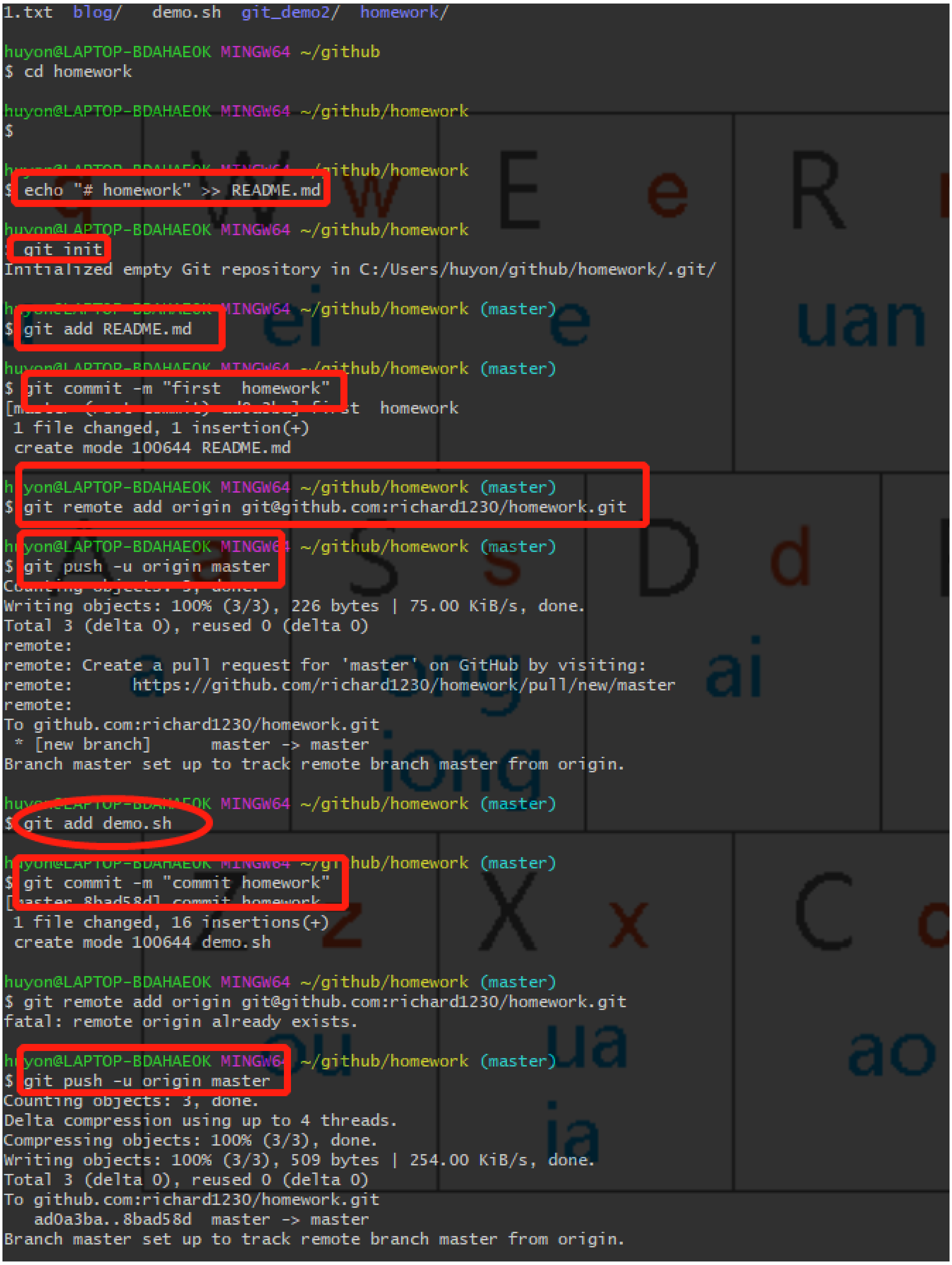
作业题(重点有上传模板)

事先把 `demo.sh` 这个脚本写好,而后copy进homework这个目录下面;

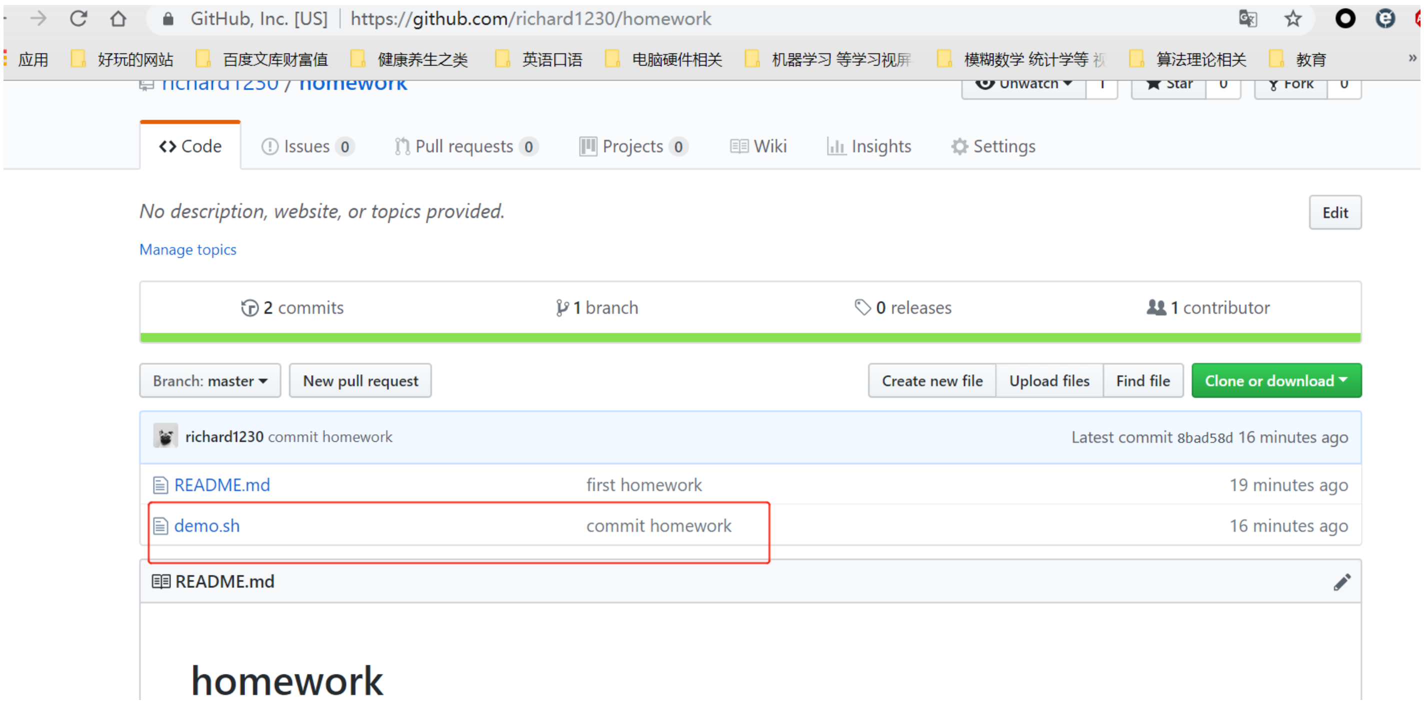


```
huyon@LAPTOP-BDAHAEOK MINGW64 ~/github
$ mkdir homework

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github
$ cp demo.sh ./homework

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github
$ ls
1.js    11.txt  demo/    demo-cp/    GitRepo/   myBlog/
1.txt   blog/   demo.sh  git_demo2/  homework/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github
$ cd homework

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework
$
```

```
echo "# homework" >> README.md
git init
git add README.md
git commit -m "first  homework"
git remote add origin git@github.com:richard1230/homework.git
git push -u origin master

git add demo.sh
git commit -m "commit homework"
git push -u origin master
```



```
1.txt   blog/   demo.sh   git_demo2/   homework/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github
$ cd homework

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework
$

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework
$ echo "# homework" >> README.md

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework
$ git init
Initialized empty Git repository in C:/Users/huyon/github/homework/.git/

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git add README.md

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git commit -m "first  homework"
[master (root-commit) ad0a3ba] first  homework
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git remote add origin git@github.com:richard1230/homework.git

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git push -u origin master
Counting objects: 3, done.
Writing objects: 100% (3/3), 226 bytes | 75.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/richard1230/homework/pull/new/master
remote:
To github.com:richard1230/homework.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git add demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git commit -m "commit homework"
[master 8bad58d] commit homework
 1 file changed, 16 insertions(+)
 create mode 100644 demo.sh

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git remote add origin git@github.com:richard1230/homework.git
fatal: remote origin already exists.

huyon@LAPTOP-BDAHAEOK MINGW64 ~/github/homework (master)
$ git push -u origin master
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 509 bytes | 254.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:richard1230/homework.git
   ad0a3ba..8bad58d  master -> master
Branch master set up to track remote branch master from origin.
```



GitHub, Inc. [US] | https://github.com/richard1230/homework

应用 好玩的网站 百度文库财富值 健康养生之类 英语口语 电脑硬件相关 机器学习 等学习视屏 模糊数学 统计学等 算法理论相关 教育

richard1230 / homework

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. Edit

Manage topics

2 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

richard1230 commit homework		Latest commit 8bad58d 16 minutes ago
README.md	first homework	19 minutes ago
demo.sh	commit homework	16 minutes ago

README.md

homework