Shell用途：



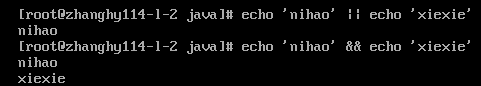
1. Ping一个网站通的话打印通，不通打印不通

Ping –c1 ‘www.baidu.com’ && echo “通” || echo ‘不通’

&&左边执行成功才继续执行右边

||表示左边执行成功右边不执行

C1表示执行一次



如果想ping 一个数据不想返回结果

Ping –c1 [www.baidu.com](http://www.baidu.com) &>/dev/null即可

1. **执行脚本**

编写的shell脚本默认是用bash这个命令行解释器，所以执行

./ \*.sh(但是要修改权限)

或者 bash \*.sh也行(无需修改权限)

1. **查找**

Find查找文件

Find / -iname nginx –type f

Iname不区分大小写

Whereis 查找程序位置

Whereis nginx

Which +命令

查找命令位置

Which bash

which cat

1. **编写脚本**

**#！shell命令解释器的路径，放在开头**

eg：#！/bin/bash

当然默认解释器就是bash

执行python脚本：  
python \*\*.py文件

1. **脚本执行方式（升级）**

c语言 编译型执行——转换成二进制码（其中根据CPU不同，编译的类型不同，如果CPU是X86那么会是复杂指令集，是power是精简指令集）

shell 解释型执行

python 既有编译也有解释，编译（转化为字节码，然后由PYTHON虚拟机来执行）

java 编译型执行——转换成字节码（由JAVA虚拟机来执行这个字节码） tomcat和hadoop都是java语言，所以运行要先安装JDK

perl 解释型执行

except 解释型执行（一般没安装）

程序是由数据和逻辑组成

执行脚本：

bash编辑器执行

bash a.sh

python 执行Py脚本

python a.py

**但是也可以直接执行脚本**

./a.sh 或者source a.sh

./a.sh有二点很重要

./表示当前目录，执行一个脚本是要用绝对路径的，但是./表示当前目录，在当前目录下是可以执行的

所以执行root下脚本可以使用/root/a.sh 也可以cd root 然后在这个目录下用相对路径 ./a.sh也可以

1. **输入重定向**

之前一直没弄清楚输入重定向，只知道输出重定向

输出重定向是将之前的命令执行内容打印到后面的文件里面

例: ls –al >> a.txt

这样cat a.txt文件内容就是ls –al的内容了

但是输入重定向就更牛逼了：是将输入的内容作为之前的命令来使用，总之就是输入作为命令,直到输入后面同样内容就停止

例： ls << EOF

-al

EOF

我输入了-al内容，他提示我还要继续输入，我想让他停止，就输入EOF，然后这个就相当于执行ls –al了

再举一例：cat << EOF

Nihao

Xiexie

EOF

Nihao

Xiexie

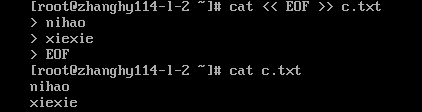
这个就相当于查看NIHAO xiexie这个内容，打印出来内容。

如果再将打印内容输出到文件呢

没错，输入输出一起用，我们输入的内容就写入到文件了

cat << EOF >> a.txt

这个命令就是将输入内容输入到a.txt文件中



1. **如果一个脚本文件执行多个解释器内容**

比如一个文件既有cat a.txt 又有print(‘hello world’)那么该怎么办？

这时候就需要用输入重定向了

a.sh文件

#!/bin/bash

cat a.txt

python << EOF

print(‘hello world’)

EOF

1. **当前shell与子shell**

当我们执行一个shell,比如bash \*.sh就会打开一个新的子shell

但是当我们直接执行，比如. \*.sh或者 source \*.sh就在当前执行

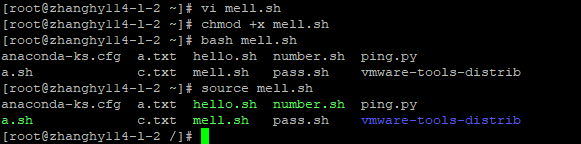
eg. vi mell.sh

ls

cd ..

然后修改权限

执行



可以发现，当我们用bash mell.sh执行，虽然执行结果，但是没有回到/目录

但是用source mell.sh就回到原始目录/

因为之前是新开了一个bash子shell中执行的，而第二个就是在当前shell执行的（类似于直接敲命令了）

但是还是有区别的，子shell有时候无法运行一些配置生效

比如：在 vi ~/.bashrc

加入 alias zhu=’rm -rf’

然后 bash ~/.bashrc

发现根本不生效，除非重启

但是执行 source ~/.bashrc就可以马上生效

1. **环境变量**

~/.bashrc

/etc/profile

/etc/.bash\_profile

~/.bash\_profile

这四个环境脚本文件

.文件，是因为这是隐藏文件，只有用 ls –al才看得见

~是用户目录，如果是root就是root目录，其他用户就是home/用户/

~/就是用户目录下

当我们任何一个用户进入家目录，ls -al都会有.bashrc 和.bash\_profile二个隐藏文件

其中.bash\_profile是当前用户下，只有在bash解释器下的设置一些环境变量和配置，

要让配置生效，必须是这个用户

而.bashrc也是在当前用户下，一些设置环境变量个人配置，这个用户设置了，其他用户只要以该用户身份运行一样可以

/etc/profile则是环境变量，任何用户都可以生效配置

/etc/.bash\_profile则是在bash解释器下面执行。任何用户配置都生效

10. 用户shell四个文件

当我们创建新用户，在每一个用户文件夹下面都有四个隐藏文件

.bashrc 开机就会运行，只对当前用户有效

.bash\_profile 用户环境配置文件，里面有执行.bashrc文件命令，所以先执行这个，就可以执行.bashrc

.bash\_logout 只要退出就会执行脚本的命令

.bash\_history 只要退出，就会把系统执行过的命令保存到这个文件，作为history，这样下次登录就能看见history

这四个shell文件非常重要

例1：有些黑客可以通过我们的history来查看我们的命令

我们想每次退出登录就删history文件内容怎么办？

在 .bash\_logout 添加这个命令就可以了

rm -f $HOME/.bash\_history  ，每次退出执行删除这个history文件命令

例2：

我们想配置一些命令，或者代理：  
vi ～/.bashrc

添加 ipconfig = ‘ip addr’

这样每次输入命令就可以查看IP地址了

例3：

我想把我一个安装包安装文件加入PATH路径，怎么办？

假如这个文件是/home/zhanghy/python/

因为用户目录～可以这样表示 $HOME

所以在～/.bash\_profile 添加

PATH=$HOME/path

我直接把系统搞崩溃了，bash 未找到命令

网上找到解决方法:

<https://blog.csdn.net/chenht8/article/details/67645649>