

## 模式匹配: case

---

### 一、case 语法结构

```
case 变量 in
模式 1)
命令序列 1
;;
模式 2)
命令序列 2
;;
模式 3)
命令序列 3
;;
*)
无匹配后命令序列
esac
```

---

### 案例 1: 简单的模式匹配

确定要继续删除吗 yes/no: " y

### 案例 2: 系统管理工具箱

```
Command action
h 显示命令帮助
f 显示磁盘分区
d 显示磁盘挂载
m 查看内存使用
u 查看系统负载
q 退出程序
```

```
Command (h for help): m
total used free shared buffers cached
Mem: 7628 840 6788 0 29 378
Swap: 2047 0 2047
```

### 案例 3: 简单的 JumpServer

```
跳板主机
1) mysql1
2) mysql2
3) bj-web1
h) help
q) exit
```

请选择要连接的主机[1-3]: 1

Last login: Sun Sep 6 04:18:01 2015 from 192.168.122.1

[yang@yang1 ~]\$ ip a show eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast

state UP qlen 1000

link/ether 52:54:00:ea:e7:d1 brd ff:ff:ff:ff:ff:ff

inet 192.168.122.186/24 brd 192.168.122.255 scope global eth0

inet6 fe80::5054:ff:feea:e7d1/64 scope link

valid\_lft forever preferred\_lft forever

=====

Shell 循环: for

循环次数是固定的

=====

### 一、for 语法结构

Shell:

for 变量名 [ in 取值列表 ]

do

循环体

done

C 语言:

for ((初值;条件;步长))

do

循环体

done

=====

### 案例 1: ping 测试主机

[root@tianyun scripts]# cat ip.txt

172.16.8.100

172.16.8.4

172.16.100.254

### 案例 1: 通过用户列表文件创建用户

[root@tianyun scripts]# cat user.txt

zhuzhu1

yang1

zhang2

## Shell 循环: while until

循环次数不一定是固定的

可以固定

可以不固定

=====

### 一、while 语句结构

while 条件测试

do

循环体

done

==当条件测试成立（条件测试为真），执行循环体

### 二、until 语法结构

until 条件测试

do

循环体

done

==当条件测试成立（条件测试为假），执行循环体

=====

## Shell 并发控制

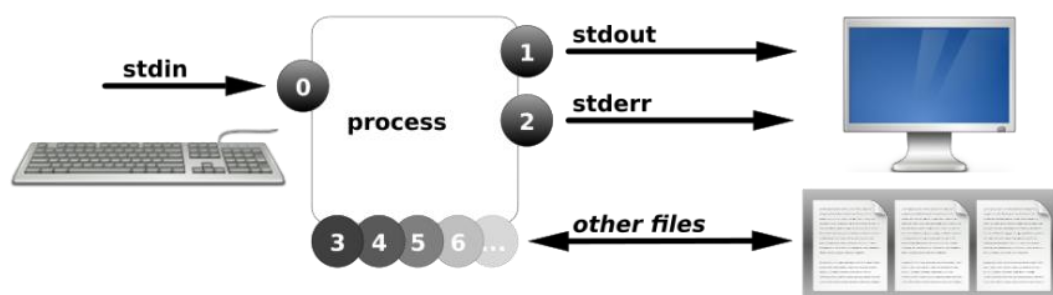
File Descriptors (FD, 文件描述符) 或 文件句柄:

进程使用文件描述符来管理打开的文件

```
[root@tianyun ~]# ls /proc/$$/fd
```

```
0 1 2 3 4
```

0, 1, and 2, known as standard input, standard output, and standard error



```
[root@tianyun ~]# ll /proc/$$/fd
```

```
total 0
lr-x----- 1 root root 64 Sep 6 13:32 0 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 1 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 2 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 15:38 255 -> /dev/pts/0
```

```
[root@tianyun ~]# touch /file1
```

```
[root@tianyun ~]# exec 6<> /file1 //打开文件
```

```
[root@tianyun ~]# ll /proc/$$/fd
```

```
total 0
lr-x----- 1 root root 64 Sep 6 13:32 0 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 1 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 2 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 15:38 255 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 6 -> /file1
```

```
[root@tianyun ~]# echo "tianyun" > /proc/$$/fd/6
```

```
[root@tianyun ~]# cat /proc/$$/fd/6
```

```
tianyun
```

```
[root@tianyun ~]# cat /file1
```

```
tianyun
```

```
[root@tianyun ~]# rm -rf /file1
```

```
[root@tianyun ~]# ll /proc/$$/fd
```

```
total 0
lr-x----- 1 root root 64 Sep 6 13:32 0 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 1 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 2 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 15:38 255 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 6 -> /file1 (deleted)
```

```
[root@tianyun ~]# cat /proc/$$/fd/6
```

```
yangsheng
```

```
yangsheng
```

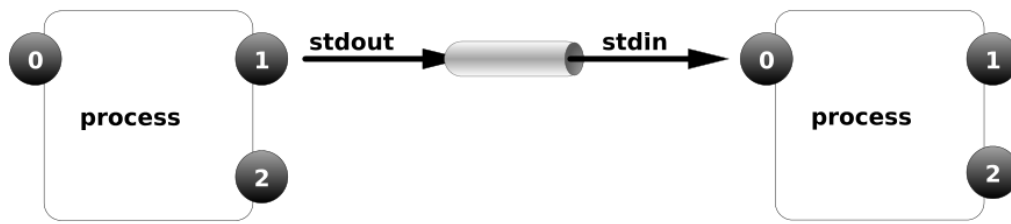
```
[root@tianyun ~]# cp -rf /proc/$$/fd/6 /file1
```

```
[root@tianyun ~]# exec 6<&-
```

```
[root@tianyun ~]# ll /proc/$$/fd
```

```
total 0
lr-x----- 1 root root 64 Sep 6 13:32 0 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 1 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 13:32 2 -> /dev/pts/0
lrwx----- 1 root root 64 Sep 6 15:38 255 -> /dev/pts/0
```

## 再谈管道



### 匿名管道

```
[root@tianyun ~]# rpm -qa |grep bash
```

### 命名管道

```
[root@tianyun ~]# mkfifo /tmp/tmpfifo
```

```
[root@tianyun ~]# file /tmp/tmpfifo
```

```
/tmp/tmpfifo: fifo (named pipe)
```

```
[root@tianyun ~]# tty
```

```
/dev/pts/0
```

```
[root@tianyun ~]# rpm -qa > /tmp/tmpfifo
```

```
[root@tianyun ~]# tty
```

```
/dev/pts/1
```

```
[root@tianyun ~]# grep bash /tmp/tmpfifo
```

```
bash-4.1.2-14.el6.x86_64
```

### 案例 1: 多文件处理

```
#!/bin/bash
```

```
exec 7<> /etc/hosts
```

```
exec 8<> /etc/sysconfig/network
```

```
while read -u 7 line
```

```
do
```

```
echo $line
```

```
read -u 8 line2
```

```
echo $line2
```

```
done
```

```
exec 7<&-
```

```
exec 8<&-
```

案例 2: 并发 ping\_multi\_thread1.sh

案例 3: 并发 ping\_multi\_thread2.sh