使用特殊环境变量

内容提要

1. 了解 Shell 变量的详细分类
2. 学会使用位置变量
3. 学会使用进程状态变量

Shell 变量的详细分类

- 用户自定义变量
- 环境变量
  - 可写的环境变量（约定用大写字母命名）
  - 只读的环境变量
    - 参数（位置）变量
    - 进程状态变量

通常将可写的环境变量就称作环境变量。环境变量和用户自定义变量的使用已经在 Shell 变量和 Shell 环境 介绍过了，本节重点介绍在 Shell 脚本中经常使用的只读环境变量。 在shell脚本中通常使用条件语句与只读的环境变量结合进行判断，再根据判断结果进行不同的操作。

只读的环境变量

只读的环境变量也称特殊的环境变量，常用的特殊的环境变量及其含义列于下表。

1、 参数（位置）变量

特殊变量	含义
$0	脚本名称
$n	n是大于等于1的整数，表示第n个位置参数，当n大于9时，要使用 ${n} 的形式
$#	位置参数的个数
"$*"	所有的位置参数(作为单个字符串)
"$@"	所有的位置参数(每个都作为独立的字符串)

2、 进程状态变量

特殊变量	含义
$$	当前进程的 PID
$?	在此之前执行的命令或脚本的返回值，0表示成功，非0表示不同原因的失败
$!	运行在后台的最后一个作业的 PID
$_	在此之前执行的命令或脚本的最后一个参数

位置变量的使用

首先编写如下的脚本：

```
$ vi abc.sh
```

```
#!/bin/bash
echo $0                    # 显示当前进程或脚本的名称
echo $1,$2,$3,$4 -- $#     # 显示前4个位置参数和位置参数的个数
echo "$@"                  # 显示所有的位置参数
shift                      # 向左移动所有的位置参数
echo $1,$2,$3,$4 -- $#
echo "$@"
```

```
$ chmod +x abc.sh
```

在当前 Shell 中使用位置参数

```
# 可以使用 set 命令在当前 Shell 中设置位置参数
$ set aa bb cc dd
$ . abc.sh
bash
aa,bb,cc,dd -- 4
aa bb cc dd
bb,cc,dd, -- 3
bb cc dd
# 也可以在调用脚本的同时传递位置参数
$ . abc.sh 1 2 3 4 5 6 7
bash
1,2,3,4 -- 7
1 2 3 4 5 6 7
2,3,4,5 -- 6
2 3 4 5 6 7
# 在调用脚本的传递的位置参数优先于 set 设置的位置参数
```

在子 shell 中使用位置参数

```
# 在调用脚本的同时传递位置参数
$ ./abc.sh 1 2 3 4 5 6 7
./abc.sh
1,2,3,4 -- 7
1 2 3 4 5 6 7
2,3,4,5 -- 6
2 3 4 5 6 7
```

进程状态变量的使用

命令或脚本的退出状态码

每个命令都会返回一个 退出状态码 (也称 返回状态)。 成功的命令返回0，而不成功的命令返回非零值。非零值通常都被解释成一个错误码。行为良好的 Linux 命令、程序和工具都会返回0作为退出码来表示成功。

同样的，脚本中的函数和脚本本身也会返回退出状态码。在脚本或者是脚本函数中执行的最后的命令会决定退出状态码。在脚本中，exit n 命令将会把 n 退出状态码传递给父 shell（n必须是十进制数, 范围是0 - 255)。

常用的错误码：

- 1 -- 通用错误，如 0 做除数等
- 126 -- 命令或脚本没有执行权限
- 127 -- 命令没找到

进程状态变量的使用举例

```
# 当前进程的 PID
$ echo $$
9245
# 调用子shell
$ bash
# 当前进程的 PID
$ echo $$
9474
# 返回父shell
$ exit
# 在此之前执行的命令或脚本的返回值
$ echo $?
0
$ list
bash: list: command not found
$ echo $?
127
$ touch bbb.sh
$ ./bbb.sh
bash: ./bbb.sh: Permission denied
$ echo $?
126
# 运行后台进程任务
$ sleep 1000 &
[1] 9494
# 显示运行在后台的最后一个作业的 PID
$ echo $!
9494
# 运行 abc.sh 脚本
$ ./abc.sh 1 2 3 4 5 6 7
./abc.sh
1,2,3,4 -- 7
1 2 3 4 5 6 7
2,3,4,5 -- 6
2 3 4 5 6 7
# 显示在此之前执行的命令或脚本的最后一个参数
$ echo $_
7
$ ls
abc.sh bin  server
# 因为 ls 是 ls --color=tty 的别名
$ echo $_
--color=tty
$
```

在 Shell 脚本中经常会判断某进程运行后的状态变量值并做相应处理，以免因为一个进程由于没有正确运行而导致后续脚本运行失败。

- 显示源文件
- 登录