Shell概述

shell的作用：

Shell是操作系统的最外层，Shell可以合并编程语言以控制进程何文件，以及

启动何控制其他程序。

shell的种类

Bourne Shell /urs/bin/sh : /bin/sh

Bourne Again Shell /bin/bash

C Shell /usr/bin/csh

K Shell /usr/bin/ksh

Shell for Root /sbin/sh

不同的Shell语言的 语法有所不同，所以不能交换使用。

\*.sh的执行

1.chmod o+x \*.sh

2./bin/bash \*.sh

shell格式

#!/bin/bash

#文件名

#功能

#作者 日期

1.shell编程之变量详解

变量无需定义，赋值就是定义

分类

局部变量

环境变量

2.shell常见的系统变量解析

$0 当前程序的名称

$n 当前程序的第n个参数（n=1，2，...9）

$\* 当前程序的所有参数（不包括程序本身）

$\*   
所有参数列表。如"$\*"用「"」括起来的情况、以"$1 $2 … $n"的形式输出所有参数。   
$@   
所有参数列表。如"$@"用「"」括起来的情况、以"$1" "$2" … "$n" 的形式输出所有参数

$# 当前程序的参数个数（不包括程序本身）

$? 命令或程序执行完后的状态，一般返回0表示执行成功

$UID 当前用户ID

$PWD 当前所在目录

$$ Shell本身的PID

3.shell编程常用命令

pwd

passwd

ls

cd

\*

?

[]

cat

file

more

clear

cp

mv

chown

chgrp

chmod

rm

mkdir

rmdir

grep

find

egrep

>>

>

<

|

||

&&

&

date

echo

sleep

head

tail

diff

spell

lp

cal

who

talk

write

df

ps

netstat

basename

man

su

cut

vi

emacs

awk

sed

pr

4.常见输入输出重定向命令格式

命令1 > 文件1 ：将命令1的输出结果重定向到文件1

命令1 &> 文件1 ：将命令1的输出结果和标准错误输出一起重定向到文件1

命令1 >> 文件1 ：将命令1的输出结果追加到文件1中

命令1 2> 文件1 ：将命令1的标准错误输出的结果重定向到文件1中

命令1 < 文件1 ：将文件1做为命令1的标准输入

命令1 << 字符串1 ：允许连续输入数据，直到收到字符串1

附加输入重定向

mail yangm\_mh@bit.edu.com << exit

> this is a test.

> test

//定义的结束字符串是exit

5.命令执行顺序

1.顺序执行，命令之间用分号隔开

2.有条件执行，&& ：前一个执行成功后才执行后一个命令

|| ：前一个执行不成功才执行后一个命令

6.shell通配符

? ：任意的单个字符

\* ：任意长度的任意字符串

[] ：匹配放在[]中的字符集中的任意一个字符

{} ：将大括号中的字符以及前导字符串和后继字符串作为匹配条件

7.单引号、双引号及反引号

双引号： 除 "$" "、" "`" "\" 都被解释成字符本身

单引号： 所有的特殊字符（ "$" "、" "`" "\"）都失去特殊意义，成为普通字符

反引号：``：在反引号中的字符被解释成命令

8.shell变量类型

环境变量

HEME PATH TERM PWD PS1 PS2 SHELL MAIL LOGNAME UID

位置变量

$0 ：当前执行的命令名

$n ：对应的第n个位置参数

预定义的特殊变量

$# ： 实际位置参数个数

$\* ： 命令行中所有位置参数组成的字符串

$! ：上一个后台命令对应的进程号

$? ：表示最近一条命令执行后的退出状态（返回值）为十进制数，一般为0

$$ ：当前进程号PID

用户自定义变量

9.变量的操作

变量名=变量值 //=前后无空格

变量的删除使用unset

变量赋值

1.使用read命令赋值

read name

echo name

2.直接给变量赋值

AGE=35

3.使用命令行参数赋值

varfz

echo $0

echo $1

echo $2

chmod u+x varfz

./varfz 1 2

4.利用命令的的输出结果赋值

CURDIR=`pwd`

RESULT=`find /etc -name rc.local`

echo $CURDIR

echo $RESULT

5.从文件读入数据实现赋值

ls > execfile

while read LINE

do

chmod +x $LINE

done<execfile

变量的输出

echo 命令：直接输出变量的值

echo $name

printf 命令：格式化树吹变量

\" ：输出双引号

\oNNN ：按八进制输出

\\ ：输出反斜杠

\a ：响铃

\b ：退格

\f ：填表

\n ：换行

\r ：回车

\t ：Tab 制表符

\v ：垂直制表符

\xNNN ：按十六进制输出

%% ：输出%

%b ：对带有"\"的字符进行转义

10.数组变量

echo ${name[2]} //使用{}是为了确保在解析时，将name[2]视为一个变量名

name=(elemnt1,elemnt2...)

11.函数

[function] 函数名

{

命令表

[return]

}

function showwho()

{

date

echo "当前登录的用户名："

echo $LOGNAME

}

函数调用

showwho

通过命令行定义的函数在用户退出时将自动被删除，如果希望函数总是可用，

而不是每次登录都要输入，可将函数的定义放在"~/.bash\_profile"中，然后

通过.命令或source .bash\_profile 文件，使修改能够立即生效

函数参数传递

$函数名 参数1 参数2 参数3 ...

$1：对应第一个位置参数

$2：对应第二个位置参赛

show()

{

echo $a $b $c $d

echo $1 $2 $3 $4

}

a=111

b=222

c=333

d=444

echo "Function Begin"

show a b c d

echo "Function End"

函数返回值

return 的返回值只能时0～256之间的一个整数

并将返回值保存到$?中

载入函数

source 函数所在文件

删除函数

unset -f 函数名

函数的作用域

没有使用关键字local进行特别声明的变量都具有全局作用域

全局变量与局部变量相同时，就近原则

函数的嵌套