# Shell 流程控制

## if elif else

* **格式**

**if** *condition1***then** *command1***elif** *condition2***then** *command2***else** *commandN***fi**

* **示例**

a=10

b=20

if [ $a == $b ]

then

echo "a 等于 b"

elif [ $a -gt $b ]

then

echo "a 大于 b"

elif [ $a -lt $b ]

then

echo "a 小于 b"

else

echo "没有符合的条件"

fi

* **示例2**

写成一行（适用于终端命令提示符）：

if [ $(ps -ef | grep -c "ssh") -gt 1 ]; then echo "true"; fi

## For循环

* **格式**

**for** var **in** item1 item2 ... itemN   
**do** *command1   
 command2   
 ...   
 commandN***done**

写成一行：

**for** var **in** item1 item2 ... itemN; **do** *command1*; *command2*… **done**;

[注] in结构后面可以使用函数来构造范围，比如$()、``这些

**for((**i=1;i<=10;i++**))**;   
**do**echo **$***(expr $i \\* 3 + 1)*;   
**done**

**无线循环**

**for ((** ; ; **))**

* **示例 1实现文件和目录遍历**

#!/usr/bin/env bash  
for file in **/usr/local/myTest/\***  
do  
 if [ -d "$file" ]  
 then echo "$file is directory"  
 elif [ -f "$file" ]  
 then echo "$file is file"  
 fi  
done

for file in **$(ls \*.sh)**

do

echo $file is file path \! ;

done

* **示例 2 遍历 脚本输入参数**

for i in **$\*** ;

do

echo $i is input chart\! ;

done

* **示例 3 遍历 字符串 按空格 分割的每个单词**

list="rootfs usr data data2"

for i in **$list**;

do

echo $i is appoint ;

done



* **示例 3 数字循环**

#!/bin/bash

for**((i=1;i<=10;i++))**;

do

echo $(expr $i \\* 3 + 1);

done

#!/bin/bash

for i in **$(seq 1 10)**

do

echo $(expr $i \\* 3 + 1);

done

#!/bin/bash

for i in **{1..10}**

do

echo $(expr $i \\* 3 + 1);

done

* **示例 for循环批量修改文件名**



"\_linux"去掉

方法1

cd /test

for i in **`ls|grep .\*.jpg`**

do

mv $i `echo $i |cut -d '.' -f1|sed 's#\_linux##'`.jpg

done。

方法2 使用cut、awk工具

cd /test

for i in **`ls|grep .\*.jpg`**

do

mv $i `echo $i|cut -d '.' -f1|awk -F '\_' '{print $1"\_"$2"\_"$3}'`.jpg

done

方法3 使用rename工具

cd /test

for i in **`ls|grep .\*.jpg`**

do

echo $i|rename "\_linux" "" \*.jpg

done

## while循环

* **格式**

**while** *condition***do** command   
**done**

* **无限循环**

while :

do

command

done

或者

while true

do

command

done

或者

for (( ; ; ))

* **循环计算1到100的和**

int=1

while**(( $int <= 5 ))**

do

echo $int

**let int++**

done

echo "final res:${int}"

OR

int=1

while **[ $int -le 5 ]# OR [[ $int -le 5 ]]**

do

echo $int

let int++

done

echo "final res:${int}"

* **read结合while循环读取文本文件**

file=$1 #将位置参数1的文件名复制给file  
if [ $# -lt 1 ];then #判断用户是否输入了位置参数  
 echo "Usage:$0 filepath"  
 exit  
fi  
**while read -r line** #从file文件中读取文件内容赋值给line（使用参数r会屏蔽文本中的特殊符号，只做输出不做转译）  
do  
 echo $line  
done **< $file** #输入重定向

## case

* **格式**

**case** 值 **in**值1) *command1* **;;**

值2|值3|值4) *command2* **;;**

\*) *commandN #无匹配*

**;;**   
**esac**

**如果无一匹配模式，使用星号 \* 捕获该值，再执行后面的命令。**

## 跳出循环break 和continue

break命令用于跳出循环，使用break可以跳出任何类型的循环：for、while、until

* **跳出单循环**

a=1  
while [ $a -le 5 ]  
do  
 if [ $a -eq 3 ]  
 then  
 **break**  
 fi  
done

* **跳出内循环**

a=1  
**while [ $a -le 5 ]**  
do  
 echo "Outer loop:a=$a"  
 a=$[$a+1]  
 **for val in 1 2 3 4 5**  
 do  
 if [ $val -eq 3 ]  
 then  
 **break**  
 fi  
 echo " Inner loop:val=$val"  
 val=$[$val+1]  
 done  
done

* **跳出多层循环**

break n，n表示要跳出的循环层数，默认情况下 n=1，代表只跳出当前循环

**continue命令用于中止本次循环，重新判断循环条件，开始下一次循环**

a=1  
for val in 1 2 3 4 5  
do   
 if [ $val -le 3 ]  
 then  
 **continue**  
 fi  
 echo "val=$val"  
done

# 输入输出重定向 和 >/dev/null 2>&1

## 文件描述符

| **类型** | **文件描述符** | **默认情况** | **对应文件句柄位置** |
| --- | --- | --- | --- |
| 标准输入（standard input） | 0 | 从键盘获得输入 | /proc/slef/fd/0 |
| 标准输出（standard output） | 1 | 输出到屏幕（即控制台） | /proc/slef/fd/1 |
| 错误输出（error output） | 2 | 输出到屏幕（即控制台） | /proc/slef/fd/2 |

## 输出重定向

| **命令** | **介绍** |
| --- | --- |
| command >filename | 把标准输出重定向到新文件中 |
| command 1>filename | 同上 |
| command >>filename | 把标准输出追加到文件中 |
| command 1>>filename | 同上 |
| command 2>filename | 把标准错误重定向到新文件中 |
| command 2>>filename | 把标准错误追加到新文件中 |

## 输入重定向

| **命令** | **介绍** |
| --- | --- |
| command <filename | 以filename文件作为标准输入 |
| command 0<filename | 同上 |
| command <<delimiter | Here Document从标准输入中读入，直到遇到delimiter分隔符,将**两个 delimiter 之间的内容**(Here Document Content 部分) 传递给command作为输入参数 |

* **示例 command <<delimiter 用法**

command *<<* **delimiter** Here Document Content  
**delimiter**

[注]

delimiter只是一个标识而已，可以替换成任意的合法字符

作为结尾的delimiter一定要**顶格**写，**前面**不能有任何字符

作为结尾的delimiter**后面**也不能有任何的字符（包括空格）

作为起始的delimiter前后的空格会被省略掉

* 普通字符

*cat <<* **EOF** *> output.sh* echo "hello"  
 echo "world"  
**EOF**

* 变量

*cat <<* **EOF** *> output.sh* echo "This is output"  
 echo $1  
**EOF**

commond >/dev/null 2>&1 等价于 commond 1>/dev/null 2>&1