Shell 脚本语法

By： sunny

Create： 2008-11-14

Version： 1.0

Copyright © All Winner Limited. All right reserved.

版本：

版本	人员	内容
	Sunny	First Version

Copyright © All Winner Limited. All right reserved.

Copyright © All Winner Limited. All right reserved.

一　关键字

1．保留字：

（1）注释：　#

（2）命令分隔符：　分号‘;’，　换行符‘\n’

（3）逻辑控制：　&&　||

（4）引号：　“”　　‘’

（5）反斜杠号：　\

（8）if　　then　　elif　　else　　fi

（9）循环控制:

　　while　　until　　for　　do　　done　　break

　　case　　in　　;;　　*　　esac

2．命令字

echo　var　set　expr　test　　run

二　变量

变量名规定：可以说大小写字母、数字、'_',第一个符号不可以为数字;

变量定义：　　var　　var　name;

变量赋值：　　set　　set　name file.txt;

变量值引用：变量名前加$　　$name;

区别：变量引用　　变量名

　　变量值引用　$变量名

示例：

Copyright © All Winner Limited. All right reserved.

```
echo "test var and set begin";
var name
var var1
set name file.txt
set var1 20
echo "value of name is :$name, value of var1 is:$ var1"
echo "test var and set end"
```

执行结果：

echo : test var and set begin

echo : value of name is :file.txt, value of var1 is:20

echo : test var and set end

三　语法说明

1．命令分隔符 ：

分号 ‘;’ 或是 换行符 ‘\n’。一行有多条语句，除了最后一条语句外其他的所有语句都必须在语句结尾加上‘;’。（注：如果一行只有一条命令，在命令结尾最好不要再加‘;’）

2．注释：

‘#’后面内容表示脚本注释说明，注释最好单独占一行

3．条件判断语句 if then elif else

判断条件：命令执行结果判断，变量值

测试返回值或者为真(0)，或者为假(1)。

一般形式

if 语句：

Copyright © All Winner Limited. All right reserved.

if 条件 1 如果条件 1 为真

then 那么 （必不可少）

命令 1 执行命令 1

elif 条件 2 如果条件 1 不成立

then 那么 （必不可少）

命令 2 执行命令 2

else 如果条件 1，2 均不成立

命令 3 那么执行命令 3

fi 完成 （必不可少）

示例：

```
echo "if test begin"
var var1 = 1
echo "var1 is $var1"
set var1 2
echo "var1 is $var1"
if (test $var1 == 1)
then
echo "var1 is 1"
elif (test $var1 == 0)
then
echo "var1 is 0"
else
echo "var1 is other value"
fi
echo "if test end"
```

执行结果：

echo : if test begin

echo : var1 is 1

echo : var1 is 2

Copyright © All Winner Limited. All right reserved.

echo : var1 is other value

echo : if test end

4. 用{ }和（）将命令结合在一起

{}与（）用法一致，可以将一组命令结合在一起，执行一组命令，可以用命令分隔符隔开每一个命令。如果一行有多个命令，除最后一条命令外其他的所有命令结束都必须加上‘;’。

一般形式为：

{命令 1;命令 2;. . . }

（命令 1；命令 2；…）

示例：

```
echo "test {} begin"
{
echo "0";echo "1"
echo "2"
echo "3"
}
echo "test {} end"echo "test () end"
```

执行结果：

echo : test {} begin

echo : 0

echo : 1

echo : 2

echo : 3

echo : test {} end

Copyright © All Winner Limited. All right reserved.

5. 使用 &&

一般形式为：

命令 1 && 命令 2

说明：&&左边的命令（命令 1）返回真(成功被执行)，&&右边的命令（命令 2）才能被执行；

即“如果这个命令执行成功& &那么执行这个命令”。

示例：

```
test 1 < 2 && echo "1 < 2 right"
test 1 > 2 && echo "1 > 2 wrong"
```

执行结果：

echo : 1 < 2 right

6. 使用 ||

一般形式为：

命令 1 || 命令 2

说明：如果||左边的命令（命令 1）未执行成功，那么就执行||右边的命令（命令 2）；

即“如果这个命令执行失败了|| 那么就执行这个命令”。

示例：

```
test 1 < 2   ||   echo "1 < 2 right"
test 1 > 2   ||   echo "1 > 2 wrong"
```

执行结果：

echo : 1 < 2 wrong

Copyright © All Winner Limited. All right reserved.

7. 循环控制

循环控制目前支持 while，until，for。

(1) while 循环

while 循环用于不断执行一系列命令，也用于从输入文件中读取数据，其格式为：

while 命令

d o

命令 1

命令 2

. . .

d o n e

在 while 和 do 之间可以放一个或是几个命令。

只有当命令的退出状态为 0 时，do 和 done 之间命令才被执行，如果退出状态不是 0，则循

环终止。命令执行完毕，控制返回循环顶部，从头开始直至测试条件为假。

示例:

```
var var1 = 0      #var1 define
echo "test while loop control"
while test $var1 <= 10
do
{
      echo "var1 is : $var1"

      if test $var1 == 5
      then
      {
      echo "var1 equal to 5, loop break out"
      break
```

Copyright © All Winner Limited. All right reserved.

```
        }
        else
        expr $var1 ++ var1      #var1 + 1
}
done
echo "breakout"
echo "Test while end"
```

执行结果：

```
echo : test while loop control
echo : var1 is : 0
echo : var1 is : 1
echo : var1 is : 2
echo : var1 is : 3
echo : var1 is : 4
echo : var1 is : 5
echo : var1 equal to 5, loop break out
echo : breakout
echo : Test while end
```

(2)until 循环

until 循环执行一系列命令直至条件为真时停止。

until 循环与 while 循环在处理方式上刚好相反。

until 循环格式为：

```
until  条件
do
命令 1
```

Copyright © All Winner Limited. All right reserved.

命令 2

. . .

done

条件可为任意测试条件，测试发生在循环末尾，因此循环至少执行一次(注意这一点).

示例：

```
var var1 = 0
echo "test until loop control"
until test $var1 == 4
do
{
        echo "var1 is : $var1"
        expr $var1 ++ var1      #var1 + 1
}
done
echo "Test until end"
```

执行结果：

echo : test until loop control

echo : var1 is : 0

echo : var1 is : 1

echo : var1 is : 2

echo : var1 is : 3

echo : Test until end

(3) for 循环

for 循环一般格式为：

for 变量名 in 列表

d o

Copyright © All Winner Limited. All right reserved.

命令 1

命令 2

done

当变量值在列表里， for 循环即执行一次所有命令，使用变量名访问列表中取值。

示例：

```
echo "Test for begin"
for var3    in    1 2 3 4 5 6 7 8 9 10
do
{
        echo "Test $var3 time in for loop"
}
done
echo "Test for end"
```

执行结果：

echo : Test for begin

echo : Test 1 time in for loop

echo : Test 2 time in for loop

echo : Test 3 time in for loop

echo : Test 4 time in for loop

echo : Test 5 time in for loop

echo : Test 6 time in for loop

echo : Test 7 time in for loop

echo : Test 8 time in for loop

echo : Test 9 time in for loop

echo : Test 10 time in for loop

echo : Test for end

Copyright © All Winner Limited. All right reserved.

8 .case 语句

case 语句为多选择语句。可以用 case 语句匹配一个值与一个模式，

如果匹配成功，执行相匹配的命令。

case 语句格式如下：

case 值 in

模式 1)

命令 1;

. . .

;;

模式 2)

命令 2

. . .

;;

e s a c //结束标识

case 后面必须为关键词 in，每一模式必须以右括号结束。取值可以

为变量或常数。匹配发现取值符合某一模式后，其间所有命令开始执行直至;;

取值将检测匹配的每一个模式。一旦模式匹配，则执行完匹配模式相应命令后不再继续其他模式。*）表示默认匹配模式，如果前面模式均未匹配，则执行默认匹配模式后面的命令。

示例：

```
echo "test case begin"
var var_case = 20
case $var_case in
10)
{
    echo "case natch 10"
    echo "out"
}
```

Copyright © All Winner Limited. All right reserved.

```
;;
20)
{
    echo "case match 20"
    echo "Out"
}
;;
*) echo "match defult"
;;
esac    #end case
echo "Test Case end"
```

执行结果：

```
echo : test case begin
echo : case match 20
echo : Out
echo : Test Case end
```

9．反斜杠号

如果下一个字符有特殊含义，反斜线防止 s h e l l 误解其含义，即屏蔽其特殊含义。下述字

符包含有特殊意义： $ " 。在 e c h o 命令或是向命令传入参数中加入元字符，必须用反斜线起屏蔽作用。

示例：

```
var var1 = 1
echo "var1 is:$var1"
echo "var1 is:\$var1"
echo "\"hello\""
```

Copyright © All Winner Limited. All right reserved.

执行结果：

echo : var1 is:1

echo : var1 is:$var1

echo : "hello"

10．引号

（1）双引号

双引号可引用除字符‘$’‘\’外的任意字符或字符串。美元符号和反斜线，对解释器来说，有特殊意义。如果使用双引号将字符串赋给变量并反馈它，实际上与直接反馈变量并无差别。

（2）单引号

单引号与双引号类似，不同的是解释器会忽略任何引用值。会将引号里的所有字符，包括引号都作为一个字符串。

示例：

```
var var1 = 1
#duble quote
echo "var1 is:$var1"
echo "hello \"test\""
#single quote
echo 'var1 is:$var1'
echo 'hello \"test\" \"
```

执行结果：

echo : var1 is:1

echo : hello "test"

echo : var1 is:$var1

echo : hello \"test\" \'

Copyright © All Winner Limited. All right reserved.

11．注意事项：

注:与 bash 相似

最后一行最好为一空白行

执行成功：返回 0

执行失败：返回 1

Copyright © All Winner Limited. All right reserved.