BASH
シェルスクリプト

条件,ステートメントブロックとif文

2006年07月07日（金）　南　慶典

exit

ホームページ製作者より

exit文で終了しよう!!

『自分でスクリプトやプログラムを作成する際にはexit文で終わるように心がけるべきである。』

なぜなら、、

UNIXの思想の一つにシンプルで小さいものを作るという思想があるらしい。

例えば「ls」と「less」で『ls | less』ができるね、ということである。各プログラムの機能が小さくシンプルになっているからそれらの組み合わせでいろんなことができる。そうすれば非常に美しく、そしてその作成したスクリプトを他と組み合わせるためにexit文が必要なのである。

終了ステータス

条件文ではcommandを終了したときに返される値、終了（返却）ステータスがかなり重要になる。

終了ステータス（終了コード）

コマンドを終了した時点で、シェルは終了ステータスという返り値を返す。

正常終了なら
ステータス0（真）を返す

異常終了なら
コマンドが見つからない、展開やリダイレクトの際のエラーのためにコマンドが失敗、実行中にエラーが起こった場合などは 0 でない値 (偽)を返す

条件（演算子&&、||）

if文を実行するためには、条件をチェックするための仕組みが必要である。

演算子&&、||

```
command1 && command2
```

command1 が成功した場合（終了ステータス0を返した時）に限り、
command2 が実行される。

```
command1 || command2
```

command1 が 成功しない場合（0 以外の終了ステータス）を返した場合に限り
command2 が実行される。

また、このリストの終了ステータスは、リスト中で最後に実行されたコマンドの終了ステータスとなる。

ステートメントブロック

&&や||で連結するコマンドはそれぞれ一つずつしか書けない。

```
command1 && command2 ; command3
```

この場合、command1 && command2に関係なくcommand3は実行されてしまう。なのでcommand1が終了してからcommand2とcommand3を実行したい場合は、ステートメントブロック{ }で囲う。

```
command1 && { command2 ; command3 }
```

commandを；で区切った場合には、これらは順番に実行される。シェルはそれぞれのコマンドが終了するのを順番に待ち、返される返却ステータスは、最後に実行したコマンドの終了ステータスになる。

test コマンド（ [or [ ] ）

test とは、引数で渡された条件式が成り立つなら (真なら) 0、成り立たないなら (偽なら) 1 を返すコマンドである。

例）

```
test 1 = 1 && echo $?
0
test 1 = 2 || echo $?
1
```

1 は 1 と等しいので真、1 は 2 と等しくないので偽となり、$?は最後に実行した commandの終了ステータスである。

test と [は同じなのだが、異なるのは [で起動すると、引数の最後に] を要求することである。また[ ]は間にスペースが必要なので注意する。つまり下の 2 つのコマンドは等価である。

```
test 1 = 1 && echo true
true
[ 1 = 1 ] && echo true
true
```

補足　条件式

testコマンドでよく使う条件式について列挙する。全てではない。

n1 = n2	整数 n1 と n2 が等しければ真
n1 != n2	整数 n1 と n2 が等しくなければ真
n1 -eq n2	整数 n1 と n2 が等しければ真
n1 -ne n2	整数 n1 と n2 が等しくなければ真
n1 -gt n2	整数 n1 が n2 がより大きければ真
n1 -ge n2	整数 n1 が n2 より大きいか等しければ真
n1 -lt n2	整数 n1 が n2 より小さければ真
n1 -le n2	整数 n1 が n2 より小さいか等しければ真

bash の変数はすべて文字列であるが、ただ [コマンドが文字列としての "1" やらを数値に置き換えて解釈してくれているので、数値としての比較が可能になる。

-n ${string}	string の長さが 0 より大きければ真
-z ${string}	string の長さが 0 であれば真
${string1} = ${string2}	二つの文字列が等しければ真
${string1} != ${string2}	二つの文字列が等しくなければ真

補足　条件式

file関係についての条件式

-r file　　fileが存在し、読込み可であれば真
-w file　fileが存在し、書込み可であれば真
-x file　　fileが存在し、実行可であれば真
-f file　　fileが存在すれば真
-d dir　　ディレクトリdirが存在すれば真
-s file　　fileが存在しサイズがゼロでなければ真

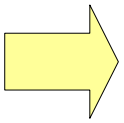
if, then, elif, else, fi

if文は複雑な条件分岐を表現するのに使用する.また,条件分岐があるコードの可読性が増す。

```
test -d hoge && echo true || echo false
```

上の文をif文で書くと

```
if test -d hoge
then echo true
else echo false
fi
```



ディレクトリ『hoge』があればtrue
を表示し、なければfalseを表示

またif文のキーワードはif、elif、else、then、fi である。testコマンドを[ ]に変えてelif を使った例を書くと

```
if [ -d hoge ]
then echo hoge is a directory
elif [ -f hoge ]
then echo hoge is a file
else echo hoge : no such a file
fi
```

1. ディレクトリ『hoge』があればhoge is a directory
2. ファイル『hoge』があればhoge is a file
3. どちらでもなければ、hoge : no ~ を表示する

if, then, elif, else, fi

if文は、testコマンド（[ ]）を使わなくても使用できる。
⇒ifとelifは行の最後のcommandが終了コード0を返していれば真とみなし、
　次のthen行を実行

以下にtestコマンドを使わない簡単な例を示す

```
if ls
then echo my directory
else echo not my dir
fi
```

```
hoge.txt
hogehoge.txt
juukaiki.dat
word.dat
my directory
```

```
if if ls; then echo my directory; else echo my dir ;fi
then echo succeeded
else echo failed
fi
```

練習問題

問題　1　以下の働きをするスクリプトmyif.shを作成せよ
juukaiki.datとword.datが存在　⇒　2つのファイルをsort
juukaiki.datしか存在しない　⇒　juukaiki.datを広さの大きい順にsort
word.datしか存在しない　⇒　word.datの中の一番少ないものを表示
両方存在しない　⇒　任意の文字列を表示

問題2
問題1のmyif.shを&&、||、[ ]は使わないで作り直したmyif2.shを作成せよ。([ ]ではなくtestで)