

システム工学概論

2. シェルプログラミングの実践

山田泰司

`taiji@aihara.co.jp`

株式会社あいはら 研究開発チーム

シェルプログラミングに必要なツール

シェルスクリプトを書くためのエディタ (例えば、GNU emacs)

Bourne shell(sh, ksh, bash, zsh)

その他、様々なコマンドラインプログラム (例えば、GNU coreutils)

shellutils(basename, expr, false, test, true など)

fileutils(chmod, mkdir, mv, rm, touch など)

textutils(cat, sort, uniq など) や GNU sed

対立するツール

csh(tcsh) ... これは学ぶ必要はありません

Microsoft command.com, cmd.exe, PowerShell など

シェルスクリプトの利点

オペレーション環境がシェルであれば、もし、定型的な作業が発生した場合に、その作業を自動化、かつ、一般化することがシェルプログラミングにより容易にできる。

また、正しく動作するシェルスクリプトを一度書いてしまえば、手作業で一連のコマンドを打鍵するよりも、間違いが無く目的を遂行できるようになる。

Bourne shell スクリプトは、ほとんどのプラットフォームで同一に動作する。但し、真のマルチプラットフォームのためには、多少の方言や外部コマンドラインプログラムの有無などに配慮する必要がある。

makefile, configure スクリプト、プログラミング言語 C からの外部アプリケーションの呼び出し等、シェルプログラミングが有効な場面が多く存在し、応用範囲が広い。

(例題 1) シェルプログラミングの実践 1

以下のような2つのファイルがあるものとする。

```
$ ls  
A.txt B.txt
```

これらのファイル名を「交換」するシェルスクリプトを書け。

(例題 1) の解説

例題は、通常、コマンドラインで以下のようにオペレーションすることに他ならない。

```
$ mv A.txt A.txt.tmp  
$ mv B.txt A.txt  
$ mv A.txt.tmp B.txt
```

GUI で操作する場合も、ファイルが同一フォルダに格納されている場合は、やはりファイル名の交換には、ひとつのファイルの退避を要する。

(例題1)の解答例

以下のようなシェルスクリプト「swapname」を作成する。

```
#!/bin/sh
tmpname="$1.tmp"
mv "$1" "$tmpname"
mv "$2" "$1"
mv "$tmpname" "$2"
```

そして、そのファイルに実行権を付ける。

```
$ chmod +x swapname
```

実行例は以下のようなになる。

```
$ ./swapname A.txt B.txt
```

(例題2) シェルプログラミングの実践2

以下のような n 個のファイル群があるものとする。

```
$ ls
```

```
A.txt B.txt C.txt D.txt E.txt ...
```

これらのファイルの内容を変えずにファイル名を「逆順」にするシェルスクリプトを書け。

(例題2)の解説

n が偶数の場合、

```
$ ls  
A.txt B.txt C.txt D.txt  
$ ./swapname A.txt D.txt  
$ ./swapname B.txt C.txt
```

n が奇数の場合、

```
$ ls  
A.txt B.txt C.txt D.txt E.txt  
$ ./swapname A.txt E.txt  
$ ./swapname B.txt D.txt
```

のように、先ほどの「swapname」シェルスクリプトをオペレーションすることに他ならない。

つまり、ファイル名交換が、引数の数の半分の回数の繰り返しとなる。

(例題2)の解答

以下のようなシェルスクリプト「swapnames」を作成する。

```
#!/bin/sh
swapapair(){
  tmpname="$1.tmp"
  mv "$1" "$tmpname"; mv "$2" "$1"; mv "$tmpname" "$2"
}
l=1
u="$#"
c=`expr $# / 2`
while true; do
  eval lfile="\${$l}"
  eval ufile="\${$u}"
  swapapair "$lfile" "$ufile"
  if [ "$l" = "$c" ]; then
    break
  fi
  l=`expr "$l" + 1`
  u=`expr "$u" - 1`
done
```

実行例は以下のようなになる。

```
$ ./swapnames A.txt B.txt C.txt D.txt E.txt
```

問題の一般化とシェルプログラミング

よいプログラミングのコツ

問題を切り分けよ（ここではファイル名交換の手続き）

問題を一般化せよ（ここでは任意のファイル数へ拡張）

想定される使用場面に応じて、例外処理で事故を防げ

シェルプログラミングのコツ

テスト環境（想定するファイル群）を用意して、気軽に実行されるであろうコマンドを出力しながらシェルスクリプトを書いていく

(例題2)の解答の補足

先のシェルスクリプト「swapnames」を以下のように修正する。

```
#!/bin/sh
swapapair(){
  if [ -f "$1" -a -f "$2" ]; then
    name=`basename $1`
    tmpname=`mktmp \"./$name.XXXXXX\"`
    mv "$1" "$tmpname"; mv "$2" "$1"; mv "$tmpname" "$2"
  elif [ -f "$1" -a ! -f "$2" ]; then
    mv "$1" "$2"
  elif [ ! -f "$1" -a -f "$2" ]; then
    mv "$2" "$1"
  fi
}
```

すると、万が一存在しないファイルを指定したとしてもエラーとはならない。

また、「~.tmp」が既に存在した場合、それが失われずに正常に動作する。

* (例題 2) の解答の補足

先のシェルスクリプト「swapnames」をさらに以下のように修正する。

```
      :
c="`expr $# / 2`"
if [ "$#" = 0 ]; then
    echo <<EOF
$0 - swap filename(s)
usage: $0 file1 file2 ...
EOF
exit
fi
while true; do
    :
```

すると、万が一、引き数を指定しない場合に無限ループにならず、簡単な使い方が表示される。