誰にでも使える m4 講座
第 1 回
「置き換えの置き換え」

安岡孝一

yasuoka : root さん、 root さん。

root : 何だい？

yasuoka : こういうスクリプトもらったんですけど、どうもよくわからないんです。

```
~/bin% cat malias （ぽこ）
#! /bin/sh
# "malias" Version 1.0
( awk '
$1=="a"{
  printf("define(`%s%c,`%s",$2,39,$3);
  for(i=4;i<=NF;i++)
    printf(" %s",$i);
  printf("%c)dnl\n",39);
}' $HOME/.mailrc
  echo $*
) | m4
exit 0
~/bin% 
```

root : どういうコマンドなんだい？

yasuoka : ~/.mailrc を見て、アドレスの別名を展開してくれます。例えば

```
~/bin% cat ~/.mailrc （ぽこ）
set folder=/home/yasuoka/Mail
set record=/home/yasuoka/mbox
set VISUAL=vi
set crt=24 PAGER=more
a taka takahash
a matu matukawa
a tm taka matu
ig message-id received date from status
~/bin% 
```

だったら、 tmっていう別名にメールを送ると

```
~/bin% malias tm （ぽこ）
takahash matukawa
~/bin% 
```

実際には takahash と matukawa にメールが送られるっていうのを、教えてくれます。

root : .mailrc に書いてあるアドレスの別名を展開してくれるんだね。で、これがどうしたの？

yasuoka : 仕掛けがさっぱりわかりません。 m4ってコマンドに色々入力してるみたいなんですけど…。

root : じゃあまず m4 から説明しなきゃいけないな。 m4 を立ち上げてごらん。

> m4 は Unix システム上で動作するマクロ・インタプリタである。

yasuoka : m4 ですね。

```
~/bin% m4 （ぽこ）

```

root : 基本的に m4 は、入力されたものをそのまま出力する。何か適当な文字列を入力してごらん。

yasuoka : じゃあ tekitoっと。

```
tekito （ぽこ）
tekito

```

tekito が出力されました。

root : そういうこと。でもこれだけだと何にもできないから、文字列を他の文字列に置き換える機能が備わってる。 define(`tekito',`otiket') してごらん。

> define(文字列,文字列)
>
> 左文字列をマクロ名とし、右文字列をその値としてマクロ定義する。何も返さない。右文字列中に $1、 $2、…、 $9 が含まれている場合は、その部分はマクロの第 1、第 2、…、第 9 パラメータで置き換えられる。 $0 が含まれている場合は、マクロ名自身で置き換えられる。

yasuoka : define(‘tekito’,‘otiket’) ですね。

define(‘tekito’,‘otiket’) （ぽこ）

これでどうなったんですか？

root : tekito が otiket に置き換わるようになったんだ。もう一度 tekito を入力してごらん。

yasuoka : tekito ですね。

tekito （ぽこ）
otiket

うーん、でもこれ sed の s とどう違うんですか？

root : tekitona はどうなる？

yasuoka : tekitona ですか？

tekitona （ぽこ）
tekitona

root : tekito と tekitona は別の文字列だとみなされるんだ。それから m4 では置き換えの置き換えができる。 define(‘hoka’,‘tekito’) してごらん。

yasuoka : はい。

define(‘hoka’,‘tekito’) （ぽこ）

root : そこで hoka はどうなる？

yasuoka : えっと

hoka （ぽこ）
okitet

okitet になりました。

root : hoka が tekito に置き換えられて、 tekito が okitet に置き換えられたんだ。さらに undefine(‘tekito’) して、もう一度 hoka してごらん。

yasuoka : undefine(‘tekito’) して、もう一度 hoka ですね。

undefine(‘tekito’) （ぽこ）

hoka （ぽこ）
tekito

undefine(文字列) 　　文字列のマクロ定義を取り消す。何も返さない。

yasuoka : hoka が tekito になりました。

root : tekito のマクロ定義は取り消されたからね。それ以上置き換えは起こらない。もう一つ m4 の define のすごいところは、パラメータを処理できることかな。 define(‘ohce’,‘$9$8$7$6$5$4$3$2$1’) してごらん。

yasuoka : define(‘ohce’,‘$9$8$7$6$5$4$3$2$1’) ですか？

define(‘ohce’,‘$9$8$7$6$5$4$3$2$1’) （ぽこ）

root : で、 ohce(te,ki,to) したらどうなる？

yasuoka : ohce(te,ki,to) ですね。

ohce(te,ki,to) （ぽこ）
tokite

パラメータを逆順に並べるんですね。

root : そういうこと。この場合は第 4 パラメータ以降はないから、 $4 から $9 はヌルになる。さらにおもしろいのは ohce(ka,ho) だ。やってごらん。

yasuoka : はい。

ohce(ka,ho) （ぽこ）
tekito

え？ どういうことですか？

root : まず ohce(ka,ho) が hoka に置き換えられるだろ。それがさらに tekito に置き換えられたんだ。

yasuoka : あ、 define(‘hoka’,‘tekito’) してあったんでしたね。

root : dumpdef(‘hoka’) で確かめてごらん。

yasuoka : dumpdef ですか？

```
dumpdef(‘hoka’) （ぽこ）
‘hoka’ ‘tekito’

```

これは「hoka は tekito に置き換えられる」っていう意味ですか？

root : そうだよ。

dumpdef(文字列) 　　文字列のマクロ定義をエラー出力に出力する。何も返さない。 dumpdef 　　マクロ定義を全てエラー出力に出力する。何も返さない。 errprint(文字列) 　　文字列をエラー出力に出力する。何も返さない。

yasuoka : さっきから気になってたんですけど、どうして‘ ’で囲むものと囲まないものがあるんですか？

root : ‘ ’は、置き換えを抑止するんだよ。試しに‘hoka’を入力してごらん。

‘文字列’ 　　文字列の置き換えを抑止する。文字列中に‘を含む場合はそれと同数の’を含んでいなければならない。 # 　　改行までの文字列の置き換えを抑止する。

yasuoka : ‘hoka’ですか？

```
‘hoka’ （ぽこ）
hoka

```

root : tekito には置き換えられなかったろ。 dumpdef(hoka) はどうなる？

yasuoka : dumpdef(‘hoka’) の‘ ’を取るんですね。

```
dumpdef(hoka) （ぽこ）

```

あれ？ 何も出ない。

root : hoka が tekito に置き換えられてから、 dumpdef(tekito) が実行されるからね。 tekito はさっき undefine したから、何も定義されてない。さて、これで m4 の基本は説明したから、さっきのスクリプトに戻ってみようか。 m4 の終了はコントロール D だよ。

yasuoka : エンドオブファイルで終了っと。

```
~/bin%
```

root : さっきのスクリプトを見せてくれるかい？

yasuoka : はい。

```
~/bin% cat malias （ぽこ）
#! /bin/sh
# "malias" Version 1.0
( awk ’
$1=="a"{
  printf("define(‘%s%c,‘%s",$2,39,$3);
  for(i=4;i<=NF;i++)
    printf(" %s",$i);
  printf("%c)dnl\n",39);
}’ $HOME/.mailrc
  echo $*
) | m4
exit 0
~/bin%
```

root : このスクリプトの m4 の部分を cat に変えたのを作ってみて。

yasuoka : えっと

```
~/bin% sed s/m4/cat/ malias （ぽこ）
#! /bin/sh
# "malias" Version 1.0
( awk ’
$1=="a"{
  printf("define(‘%s%c,‘%s",$2,39,$3);
  for(i=4;i<=NF;i++)
    printf(" %s",$i);
  printf("%c)dnl\n",39);
}’ $HOME/.mailrc
```

```
  echo $*
) | cat
exit 0
~/bin% ■
```

これで大丈夫かな。

```
~/bin% !! > malias.tmp （ぽこ）
sed s/m4/cat/ malias > malias.tmp
~/bin% chmod 755 malias.tmp （ぽこ）
~/bin% rehash （ぽこ）
~/bin% ■
```

できました。

root : それじゃ、 malias.tmp tm してみて。

yasuoka : はい。

```
~/bin% malias.tmp tm （ぽこ）
define(‘taka’,‘takahash’)dnl
define(‘matu’,‘matukawa’)dnl
define(‘tm’,‘taka matu’)dnl
tm
~/bin% ■
```

あ、そういうことなんですね。

```
~/bin% !! | m4 （ぽこ）
malias.tmp tm | m4
takahash matukawa
~/bin% ■
```

でもこの dnlって何ですか？

root : じゃあ、 dnl をなくしてみてごらん。

yasuoka : はい。

```
~/bin% malias.tmp tm | sed s/dnl// （ぽこ）
define(‘taka’,‘takahash’)
define(‘matu’,‘matukawa’)
define(‘tm’,‘taka matu’)
tm
~/bin% ■
```

これでいいかな。

```
~/bin% !! | m4 （ぽこ）
malias.tmp tm | sed s/dnl// | m4
```

```
takahash matukawa
~/bin% ■
```

あら？ 空行がいっぱい出た。

root : define それ自体は何も返さないけど、その後にある改行コードは m4 を素通りしてしまうからね。これを防ぐのが dnl だ。

dnl 　　改行までの文字列を読み飛ばす。

yasuoka : dnl で改行コードを取り除くことができるんですね。

root : そういうこと。でもこのスクリプト、ちょっとまずいんじゃないかな。

yasuoka : まずいって？

root : malias unix としてごらん。

yasuoka : unixっていうアドレスがどう展開されるかですね。

```
~/bin% malias unix （ぽこ）
```

```
~/bin% ■
```

何も出ない。

root : unix は define(‘unix’,‘’) で初期化されてるからね。それから malias tm-tm も試してごらん。

yasuoka : malias tm-tm ですか？

```
~/bin% malias tm-tm （ぽこ）
takahash matukawa-takahash matukawa
~/bin% ■
```

ちょっと変ですね。

root : メールのアドレスには色々な文字が使えるけど、 m4 でのマクロは英数字と _ からなる文字列で、 1 文字目が数字以外のものだけだからね。 tm-tm は 1 つの文字列とはみなされなくって、 tm と - と tm に分割されちゃう。

yasuoka : どうしたらいいんですか？

root : メールのアドレスで使える文字を、全部英数字に置き換えるようにするしかないんじゃないかな。

```
#! /bin/sh
# "malias" Version 1.1 for BSD
( tr 'a-z@%!.\055' A-Za-e < $HOME/.mailrc | awk '
$1=="A"{
  printf("define(‘z%s%c,‘z%s",$2,39,$3);
  for(i=4;i<=NF;i++)
    printf(" z%s",$i);
  printf("%c)dnl\n",39);
}'
  echo -n z
  echo "$*" | tr 'a-z@%!.\055' A-Za-e | sed 's/ / z/g'
) | m4 | tr A-Za-e 'a-z@%!.\055' | tr -d z
exit 0
```

（間）

root : メールのアドレスは大文字と小文字を区別しないから、それも入れておいたよ。もう出来たかい？

yasuoka : はい。

```
~/bin% malias unix （ぽこ）
unix
~/bin% malias tm-tm tm （ぽこ）
tm-tm takahash matukawa
~/bin% 
```

大丈夫みたいですね。どういう仕掛けなんですか？

root : m4 の入力を横取りしてごらん。

yasuoka : えーと

```
~/bin% sed 's/m4/tee tmp | m4/' malias （ぽこ）
#! /bin/sh
# "malias" Version 1.1 for BSD
( tr 'a-z@%!.\055' A-Za-e < $HOME/.mailrc | awk '
$1=="A"{
  printf("define(‘z%s%c,‘z%s",$2,39,$3);
  for(i=4;i<=NF;i++)
    printf(" z%s",$i);
  printf("%c)dnl\n",39);
}'
  echo -n z
  echo "$*" | tr 'a-z@%!.\055' A-Za-e | sed 's/ / z/g'
) | tee tmp | m4 | tr A-Za-e 'a-z@%!.\055' | tr -d z
exit 0
~/bin% !! > malias.tmp （ぽこ）
sed 's/m4/tee tmp | m4/' malias > malias.tmp
~/bin% malias.tmp tm-tm （ぽこ）
tm-tm
~/bin% cat tmp （ぽこ）
define(‘zTAKA’,‘zTAKAHASH’)dnl
define(‘zMATU’,‘zMATUKAWA’)dnl
define(‘zTM’,‘zTAKA zMATU’)dnl
zTMeTM
~/bin% 
```

こういうことですか。なかなか。

```
~/bin% rm malias.tmp tmp （ぽこ）
rm: remove malias.tmp? y （ぽこ）
rm: remove tmp? y （ぽこ）
~/bin% 
```

ところで root さん。 m4って、 define しかないんですか？

root : いや他にもいくつか命令があって、複雑なマクロ定義も書けるようになってる。次回はそういう命令を説明しよう。

yasuoka : はい。次回もよろしくお願いします。