誰にでも使える m4 講座
第 3 回
「マクロを定義するマクロの定義」

安岡孝一

```
sed "1s?^.*\$?define(\‘argv’,\‘\‘$*’’)dnl?" $0 | m4 ; exit
dnl ##### "malias" Version 2.0 for BSD #####
divert(-1)
define(‘tT’,‘translit(‘$1’,
  ‘abcdefghijklmnopqrstuvwxyz@%!.-’,
  ‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcde’)’)
define(‘xny’,‘define(‘i’,index(‘$1’,‘ ’))ifelse(i,0,
  ‘$0(substr(‘$1’,1))’,i,-1,‘‘x$1y’’,
  ‘‘x’substr(‘$1’,0,i)‘y’ $0(
  substr(‘$1’,incr(i)))’)’)
define(‘mal’,‘define(‘x’tT(‘$1’)‘y’,xny(tT(‘$2’)))’)
define(‘tf’,maketemp(‘/tmp/maliasXXXXX’))
changequote({,})
syscmd({expand $HOME/.mailrc | sed -e ’/^a /!d’ -e \
  ’s/^a *\([^ ]*\) \(.*\)$/’"mal(\‘\1’,\‘\2’)/" >} tf)
changequote
include(tf)syscmd(‘rm’ tf)
define(‘Tt’,‘translit(‘$1’,
  ‘edcbaxyABCDEFGHIJKLMNOPQRSTUVWXYZ’,
  ‘-.!%@‘’abcdefghijklmnopqrstuvwxyz’)’)
divert(0)Tt(substr(xny(tT(argv)),0))
```

root ： yasuoka くん、 yasuoka くん。
yasuoka ： 何ですか？
root ： 前々回の malias の m4 版を作ってみたよ。

（間）

yasuoka ： これですか？ うーん、何がなんだかさっぱりわかりませんね。

```
~/bin% malias tm unix （ぽこ）
takahash matukawa unix
~/bin% egrep ’^a ’ ~/.mailrc （ぽこ）
a taka takahash
a matu matukawa
a tm taka matu
~/bin% 
```

でもちゃんと動いてますね。

root ： 最初の行はシェル・スクリプトで、スクリプトに渡されたパラメータを argv に定義するためのものだ。2 行目以降が m4 のスクリプト。実際に malias tm unix を実行した時には

```
define(‘argv’,‘‘tm unix’’)dnl
dnl ##### "malias" Version 2.0 for BSD #####
divert(-1)
define(‘tT’,‘translit(‘$1’,
  ‘abcdefghijklmnopqrstuvwxyz@%!.-’,
  ‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcde’)’)
define(‘xny’,‘define(‘i’,index(‘$1’,‘ ’))ifelse(i,0,
  ‘$0(substr(‘$1’,1))’,i,-1,‘‘x$1y’’,
  ‘‘x’substr(‘$1’,0,i)‘y’ $0(
  substr(‘$1’,incr(i)))’)’)
define(‘mal’,‘define(‘x’tT(‘$1’)‘y’,xny(tT(‘$2’)))’)
define(‘tf’,maketemp(‘/tmp/maliasXXXXX’))
changequote({,})
syscmd({expand $HOME/.mailrc | sed -e ’/^a /!d’ -e \
  ’s/^a *\([^ ]*\) \(.*\)$/’"mal(\‘\1’,\‘\2’)/" >} tf)
changequote
include(tf)syscmd(‘rm’ tf)
define(‘Tt’,‘translit(‘$1’,
  ‘edcbaxyABCDEFGHIJKLMNOPQRSTUVWXYZ’,
  ‘-.!%@‘’abcdefghijklmnopqrstuvwxyz’)’)
divert(0)Tt(substr(xny(tT(argv)),0))
```

っていうのが m4 のスクリプトとして順次実行されるわけだ。

yasuoka ： うーん、なかなかすごいですね。順に教えていって下さい。

root : 1 行目は argv の定義。2 行目はコメントだ。3 行目の divert(-1) だけど、これは m4 の出力を全部捨てるっていう命令だ。

divert(-1) 　出力先を /dev/null に変更する。何も返さない。 divert(0) 　出力先を標準出力に戻す。何も返さない。 divert(数) 　出力先を「数」番目のバッファファイルに変更する。何も返さない。「数」は 1 桁の自然数のみ。 undivert(数) 　「数」番目のバッファファイルの内容を現在の出力先に出力し、そのバッファファイルを消去する。何も返さない。「数」は、現在の出力先以外の 1 桁の自然数。 divnum 　現在の出力先番号を返す。

root : ちょっとためしてみようか。m4 を立ち上げてごらん。
yasuoka : はい。

```
~/bin% m4 （ぽこ）
```

root : divert(-1) してごらん。
yasuoka : divert(-1) ですね。

```
divert(-1) （ぽこ）
```

root : で、適当な文字列を入力する。
yasuoka : tekitoっと。

```
tekito （ぽこ）
```

何も出ませんね。
root : でも define とかはできるよ。
yasuoka : えっと

```
define(‘S’,‘ifelse($1,1,1,‘$0(eval($1-1))+$1’)’) （ぽこ）
S(10) （ぽこ）
```

でも、何も出ないんじゃ、 define しても仕方ないですよ。
root : じゃ、元に戻そう。 divert(0) してごらん。
yasuoka : divert(0) ですね。

```
divert(0) （ぽこ）
```

戻ったのかな？

```
tekito （ぽこ）
tekito
S(10) （ぽこ）
1+2+3+4+5+6+7+8+9+10
```

あ、ちゃんと表示されるようになりました。
root : divert にはもう一つ機能があって、出力先をバッファファイルに変えることができるんだ。まずは divert(1) してごらん。
yasuoka : divert(1) ですね。

```
divert(1) （ぽこ）
```

root : で、適当な文字列を入力する。
yasuoka : tekitoっと。

```
tekito （ぽこ）
```

やっぱり表示されませんよ。
root : 次に divert(3) してごらん。
yasuoka : divert(3) ですか？

```
divert(3) （ぽこ）
```

root : で、さっきと違う文字列を入力する。
yasuoka : chigau mojiretsu ですね。

```
chigau mojiretsu （ぽこ）
```

やっぱり表示されません。

root : さらに divert(2) して、また違う文字列を入力する。
yasuoka : はい。

```
divert(2)（ぽこ）
mata chigau mojiretsu（ぽこ）
```

root : それから m4 を終了する。
yasuoka : コントロール D ですね。

```
tekito

mata chigau mojiretsu

chigau mojiretsu
~/bin%
```

あれ、さっきの文字列が今、表示された。

root : divert(1)ってのは、1 番目のバッファファイルに出力先を変更する、っていう意味なんだよ。divert(1) から divert(9) が使えるんだけど、これらのバッファファイルに入った内容は、m4 が終了する時に 1 番から順に標準出力に出力されるんだ。そういう順番になってるだろ？
yasuoka : そういえば mata chigau mojiretsu の方が chigau mojiretsu より先になってますね。
root : そういうこと。だいぶ脱線しちゃったな。どこまで話したっけ。
yasuoka : 3 行目の divert(-1) までです。でも、どうしてここで divert(-1) なんですか？
root : define の後にいちいち dnl を書くのがめんどくさかったからだよ。さて、4 行目から 11 行目までは tT と xny と mal のマクロ定義だけど、これは後回しにして 12 行目の define(‘tf’,maketemp(‘/tmp/maliasXXXXX’)) を説明しようか。

maketemp(文字列)
文字列中の XXXXX に、プロセス番号から生成される文字列を埋めて返す。

root : maketempってのは、テンポラリファイルの名前を作るためのマクロだと思っていい。ここでの maketemp(‘/tmp/maliasXXXXX’) だけど、例えばプロセス番号が 10000 だったら /tmp/malias10000 が返ってくる。最近の m4 では、XXXXX にプロセス番号から生成した別の文字列を埋めるみたいだけどね。とにかく、これでテンポラリファイル名を作ることができる。
yasuoka : どうしてテンポラリファイルが要るんですか？
root : syscmd の結果を一旦入れておいて読み込むためだよ。

syscmd(文字列)
文字列を Unix コマンドとみなして、シェルを通じて実行する。何も返さない。

root : syscmd の中の文字列は、Unix コマンドとして実行されるんだ。ここでは 14 行目から 15 行目に続く syscmd で expand $HOME/.mailrc | sed -e ’/^a /!d’ -e ’s/^a *\([^]*\) \(.*\)$/’"mal(\‘\1’,\‘\2’)/" > /tmp/malias10000 を実行して /tmp/malias1000 を作る。それを 17 行目の include(tf)、つまり include(/tmp/malias10000) で読み込む。

include(ファイル名)
ファイルの中身をそのまま返す。ファイルがなかった場合はエラー終了。
sinclude(ファイル名)
ファイルの中身をそのまま返す。ファイルがなかった場合は何も返さない。

yasuoka : 読み込むって？
root : /tmp/malias10000 の中は

```
mal(‘taka’,‘takahash’)
mal(‘matu’,‘matukawa’)
mal(‘tm’,‘taka matu’)
```

だから、include(/tmp/malias10000) は、これに置き換えられるんだ。
yasuoka : そういうことですか。
root : 続けようか。mal(‘taka’,‘takahash’) は、まず中のパラメータが置き換えられて mal(taka,takahash) になる。その次に mal が置き換えられて define(‘x’tT(‘taka’)‘y’,xny(tT(‘takahash’))) になる。
yasuoka : はい。
root : それから、また中のパラメータが置き換えられるんだけど、‘x’ はそのまま x に、tT(‘taka’) は tT(taka) に置き換えられた後

```
translit(‘taka’,
  ‘abcdefghijklmnopqrstuvwxyz@%!.-’,
  ‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcde’)
```

になる。ここでは、 t は T に、 a は A に、 k は K に置き換わるから、結果として TAKA になる。これで tT(‘taka’) の置き換えは終わりで、 ‘y’ はそのまま y になる。

yasuoka : 大文字にしてるだけですね。

root : まあそうかな。特殊文字もちょっとあるけどね。

yasuoka : でも ‘taka’, の後に改行がありますけど。

root : ああ、 m4 のマクロの置き換えでは (と, の後の改行や空白は無視されるんだよ。さて、次は xny(tT(‘takahash’)) だ。同じように xny(TAKAHASH) になってさらに

```
define(‘i’,index(‘TAKAHASH’,‘ ’))ifelse(i,0,
  ‘xny(substr(‘TAKAHASH’,1))’,i,-1,‘‘xTAKAHASHy’’,
  ‘‘x’substr(‘TAKAHASH’,0,i)‘y’ xny(
  substr(‘TAKAHASH’,incr(i)))’)
```

になる。ここで i が define されるんだけど、 index(‘TAKAHASH’,‘ ’) は -1 なので結局 i も -1 になる。

yasuoka : TAKAHASH の中に空白がないからですね。

root : そう。で、次の ifelse に移るんだけど、まず中のパラメータが全部置き換えられて

```
ifelse(-1,0,
  xny(substr(‘TAKAHASH’,1)),-1,-1,‘xTAKAHASHy’,
  ‘x’substr(‘TAKAHASH’,0,i)‘y’ xny(
  substr(‘TAKAHASH’,incr(i))))
```

になり、第 4・第 5 パラメータが同じなので、 ‘xTAKAHASHy’ になる。置き換えると xTAKAHASHy になるから、結局 mal(‘taka’,‘takahash’) は define(xTAKAy,xTAKAHASHy) となって、 xTAKAy に xTAKAHASHy が定義される。 mal(‘matu’,‘matukawa’) も同じで、 xMATUy に xMATUKAWAy が定義される。 mal(‘tm’,‘taka matu’) はちょっと動きが違うから、ちゃんと説明しようか。

yasuoka : お願いします。

root : まずは define(‘x’tT(‘matu’)‘y’,xny(tT(‘taka matu’))) となって、 ‘x’tT(matu)‘y’ の方は xMATUy になる。 xny(tT(‘taka matu’)) の方は xny(TAKA MATU) になってさらに

```
define(‘i’,index(‘TAKA MATU’,‘ ’))ifelse(i,0,
  ‘xny(substr(‘TAKA MATU’,1))’,i,-1,‘‘xTAKA MATUy’’,
  ‘‘x’substr(‘TAKA MATU’,0,i)‘y’ xny(
  substr(‘TAKA MATU’,incr(i)))’)
```

になる。ここで i は 4 に define されるから、 ifelse の方は

```
ifelse(4,0,
  xny(substr(‘TAKA MATU’,1)),4,-1,‘xTAKA MATUy’,
  ‘x’substr(‘TAKA MATU’,0,i)‘y’ xny(
  substr(‘TAKA MATU’,incr(i))))
```

となって、最後の

```
‘x’substr(‘TAKA MATU’,0,i)‘y’ xny(
  substr(‘TAKA MATU’,incr(i)))
```

に置き換わる。ここまではいいね。

yasuoka : 4 と 0 は同じじゃないし、 4 と -1 も同じじゃないからですね。

root : そういうこと。で、 ‘x’substr(‘TAKA MATU’,0,i)‘y’ の方は xTAKAy に、 xny(substr(‘TAKA MATU’,incr(i))) の方は incr(i) つまり incr(4) が 5 になるので、 xny(MATU) になる。

yasuoka : ちょっと待って下さい。 incr(4) がどうして 5 なんですか？

incr(文字列) 文字列の最初の数字の部分を取り出し、それに 1 を加えた値を返す。なお、文字列の最初の文字が数字でも - でもない場合は 1 を返す。

root : incr(整数) は eval(整数+1) だと考えたらいいよ。

yasuoka : わかりました。

root : それから xny(MATU) だけど、これは

```
define(‘i’,index(‘MATU’,‘ ’))ifelse(i,0,
  ‘xny(substr(‘MATU’,1))’,i,-1,‘‘xMATUy’’,
  ‘‘x’substr(‘MATU’,0,i)‘y’ xny(
  substr(‘MATU’,incr(i)))’)
```

になって、今度は i が -1 になるから

```
ifelse(-1,0,
  xny(substr(‘MATU’,1)),-1,-1,‘xMATUy’,
  ‘x’substr(‘MATU’,0,i)‘y’ xny(
  substr(‘MATU’,incr(i))))
```

になって、‘xMATUy’ つまり xMATUy が返る。

yasuoka : -1 と 0 は同じじゃないけど、 -1 と -1 が同じっと。

root : 結局 mal(‘tm’,‘taka matu’) は define(xTMy,xTAKAy xMATUy) になって、 xTMy に xTAKAy xMATUy が定義されるんだ。

yasuoka : うーん、だいたいわかりました。

root : これで include(/tmp/malias10000) は終わりだから、元のスクリプトの 17 行目の syscmd(‘rm’ tf) に戻って、 /tmp/malias10000 が rm される。最後から 2 行目までは Tt の定義。ここまでが済んで divert(0) で出力を標準出力に戻したら、最後の Tt(substr(xny(tT(argv)),0)) の置き換えに入る。

yasuoka : いよいよ佳境ですね。

root : まずは argv を置き換えて Tt(substr(xny(tT(tm unix)),0)) となり、tT と xny の置き換えで Tt(substr(xTMy xUNIXy,0)) となる。

yasuoka : tT が大文字変換で、 xny が x と y を付けるんでしたね。

root : substr(xTMy xUNIXy,0) だけど、これは xTMy xUNIXy を返すと同時に、再度 xTMy と xUNIXy を置き換えるものなんだ。 xTMy は xTAKAy xMATUy に置き換えられた後、さらに xTAKAHASHy xMATUKAWAy になる。 xUNIXy の方はもうこれ以上置き換えられないから、 substr(xTMy xUNIXy,0) は結局 xTAKAHASHy xMATUKAWAy xUNIXy になる。

yasuoka : うーん。

root : 続けるよ。 Tt(xTAKAHASHy xMATUKAWAy xUNIXy) は

```
translit(‘xTAKAHASHy xMATUKAWAy xUNIXy’,
  ‘edcbaxyABCDEFGHIJKLMNOPQRSTUVWXYZ’,
  ‘-.!%@‘’abcdefghijklmnopqrstuvwxyz’)’)
```

だから ‘takahash’ ‘matukawa’ ‘unix’ となって

```
takahash matukawa unix
```

が出力されておしまい。

yasuoka : 大変でしたね。

root : このスクリプトは BSD 用だから、最後に System V 用のスクリプトを付けておこう。 System V では m4 は相当拡張されてるから気を付けてね。

yasuoka : え、もう終わりなんですか。もうちょっと訊きたいこともあったのに。

root : それじゃ、 m4 はここまで。

```
dnl () {} ; exec m4 -Dargv="\‘$*’" $0
dnl ##### "malias" Version 2.0 for System V #####
divert(-1)
define(‘tT’,‘translit(‘$1’,
  ‘abcdefghijklmnopqrstuvwxyz@%!.-’,
  ‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcde’)’)
define(‘xny’,‘define(‘i’,index(‘$1’,‘ ’))ifelse(i,0,
  ‘$0(substr(‘$1’,1))’,i,-1,‘‘x$1y’’,
  ‘‘x’substr(‘$1’,0,i)‘y’ $0(
  substr(‘$1’,incr(i)))’)’)
define(‘mal’,‘define(‘x’tT(‘$1’)‘y’,xny(tT(‘$2’)))’)
define(‘tf’,maketemp(‘/tmp/maliasXXXXX’))
changequote({,})
syscmd({newform -i $HOME/.mailrc | sed -e ’/^a /!d’ -e \
  ’s/^a *\([^ ]*\) \(.*\)$/’"mal(\‘\1’,\‘\2’)/" >} tf)
changequote
include(tf)syscmd(‘rm’ tf)
define(‘Tt’,‘translit(‘$1’,
  ‘edcbaxyABCDEFGHIJKLMNOPQRSTUVWXYZ’,
  ‘-.!%@‘’abcdefghijklmnopqrstuvwxyz’)’)
divert(0)Tt(substr(xny(tT(argv)),0))
```