```
ARGS=$@
kterm -fg white -bg black -xim -geometry 80x40 \
-e sh -c '/usr/local/bin/mutt $ARGS' &
```

Muttには、メッセージの一覧を表示する「インデックス画面」(画面5)と、メッセージの内容を表示する「ページャ画面」(画面6)があります。デフォルトの状態では、このように、イ

COLUMN 3

メールボックスの形式

ここでは、Muttで利用できるメールボックスの形式の特徴を説明します。

●mbox

1つのファイルにすべてのメッセージを格納する形式で、各メッセージは以下のように「From 」で始まる行(From 行)で始まり、空白行で終わります。From 行の形式は、大抵「From 送信者(envelope sender) 配送時間」となっています。

このため、このFrom 行が各メッセージの区切りになりますが、本文中に「From 」で始まる行があると区別できなくなって問題になります。この対策として、本文中の「From 」の前に「>」などを挿入して「>From 」のようにする方法と、次のメッセージの「From 」行を示すために、ヘッダにメッセージサイズを記録する「Content-Length:」フィールドや、メッセージの行数を記録する「Lines:」フィールドを挿入する方法とがあります。前者の方法では、メッセージを変更しているため電子署名の検証などで問題が生じます。また後者の方法では、記録された数値が何らかの原因で間違っている場合に、メーラーがそのメッセージの全部を表示できないことがあります。なお、Muttは後者の方法で処理しています。

●MMDF

mbox形式の変種の1つで、各メッセージを「^A^A^A^A」(Ctrl+Aを4つ並べた行)で囲みます。このため、前述したFrom 行関係の問題は一切ありません。

●MH

ディレクトリ内に、1メッセージ1ファイルで格納する形式です。ファイル名はそれぞれ1から始まる連続した番号になります(Muttのインデックスで表示する番号とは無関係です)。例えば、次のようになります。

```
$ ls inbox
1  2  3
```

mboxやMMDFと違って、既存のメッセージファイルそのものにはロックをかける必要はありませんが、MDAから配達されるメッセージに関しては、ロックの必要性があります。

●Maildir

qmailというMTAで使われているメールボックスの形式です。MHと同様、ディレクトリ内へ1メッセージ1ファイルで格納する形式です。しかし、「tmp」,「new」、「cur」という3つのサブディレクトリの使用と、ユニークなファイル名の付け方により、ファイルのロックが一切必要なく安全に操作できるようになっています。このため、NFSの環境でも安心して使えます。それぞれのサブディレクトリの役割は次のようになります。

tmp:安全な配送を行なうために一時的に使われる
new:スプール
cur:メーラがnewから読み取ったメールを扱う

ディレクトリに格納されているファイルは、以下のようになっています。

```
$ ls Maildir/*
Maildir/cur:
982378966.3456.deathstar:2,S
          ⋮
```

これを見ると、ファイル名が「time.pid.host」の形式であるのが分かります。これがユニークであるための必用条件です。さらにcurディレクトリにあるファイルは、ファイル名の後尾に以下のような情報を付け加えることができます。

R(replied): 返信済
S(seen): 既読
T(trashed): 削除
F(flagged): ユーザー定義のフラグ

このため、メッセージを解析しなくてもそのメッセージのステータス情報が分かるようになっています。

逆にmbox、MMDF、MHなどは、ヘッダに「Status:フィールド」を挿入することで、こうした情報を保持しています。

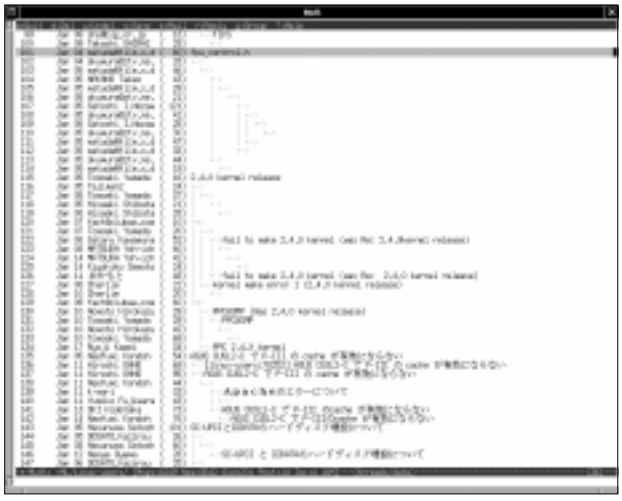
●お勧めのメールボックスの形式

特に特定のメールボックスを使いたいという明確な理由がないのでしたら、Maildir形式をお勧めします。利点としては

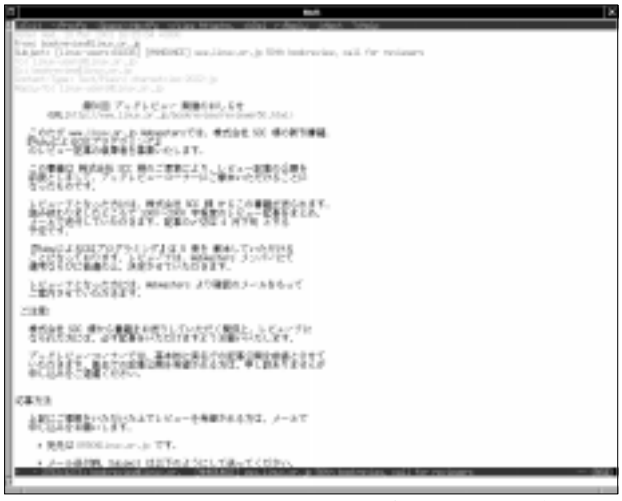
・ファイルの破壊があった場合でも、局所的な被害で済む
・メッセージの移動、コピーのコストが小さい
・メッセージをオリジナルのまま扱える
・未読、既読などのステータス情報の取得コストが小さい

といったことが挙げられます。逆に、欠点としては、ファイルの数が多くなり、それに伴う問題が生じる可能性があることくらいです。

(滝澤隆史)



画面5　インデックス画面(スレッド表示は折り畳むことも可能)



画面6　ページャ画面(ページャの表示もさまざまにカスタマイズ可能)

ンデックス画面とページャ画面の表示はともに全画面表示となりますが、変数「$pager_index_lines」を

```
set pager_index_lines=10
```

のように指定すれば、その行数だけインデックス画面を表示したままページャー画面を表示することができます(画面7)。

ページャでは、

```
set tilde=yes
```

のように$tilde変数をセットすれば、メッセージの終わりを示すために空白行をチルダで埋めて表示させることができます(画面8)。

今度は、ページャでヘッダ表示をカスタマイズしてみます。まず、次のコマンドで、表示させるヘッダをいったんクリアします。

```
ignore *
```

なお、カスタマイズしたヘッダ表示は、メッセージを表示し直すことで反映されます。

こうした上で次に、表示させたいヘッダを

```
unignore date from to cc reply-to subject
```

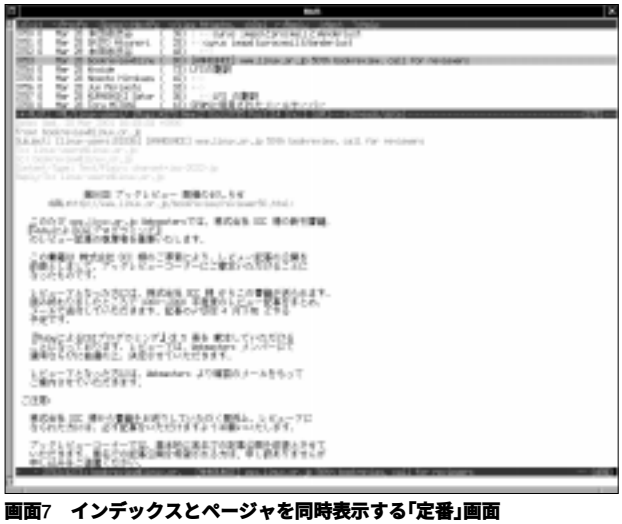
としてセットし

```
hdr_order date from subject to cc reply-to
```

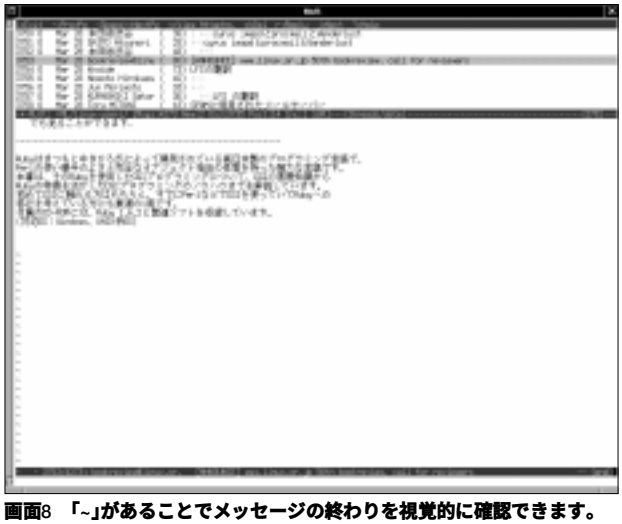
で、その順番を指定します(画面9)。

色の設定では、まず、環境変数「COLORFGBG」でデフォルトの色を設定しましょう。これは、フォアグランドとバックグランドをセミコロンで区切り、

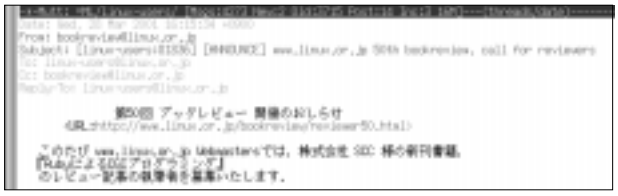
```
$ set COLORFGBG="white;black"
```



画面7　インデックスとページャを同時表示する「定番」画面



画面8　「~」があることでメッセージの終わりを視覚的に確認できます。



画面9　ページャに表示されるヘッダは自在にカスタマイズ可能

```
$ export COLORFGBG
```

のように設定します。後は、Muttや日本語パッチに含まれているサンプルファイルを例にして、マニュアルの色属性の項目を見ながらカスタマイズしてみてください。

●作成に関するカスタマイズ

メッセージ作成に関しては、表1のようなカスタマイズが可能です。

●メーリングリスト用の設定

Muttでは、フォルダごとに表示方法を変えることもできます。以下のように設定すれば、通常のフォルダではインデックス画面でスレッド表示し、$recordフォルダでは送信日順に、$spoolfileや$mboxフォルダでは日付順にそれぞれ表示されるようになります。

```
folder-hook '.' 'set sort=threads'
folder-hook '<' 'set sort=date-sent'
folder-hook '!' 'set sort=date-received'
```

日本のメーリングリストでは

```
Subject: [prefix:0123] hogohoge
```

のように、Subjectフィールドに、メーリングリスト名と投稿番号などのプレフィックスをつける場合が多く見受けられます。しかし、プレフィックスが長いと肝心のSubjectの内容が行からはみ出てしまって読めません。そのため、次の設定を行うとこのプレフィックスをインデックス画面では表示しなくなります。また、返信や転送する際も除去するようになります。

```
set delete_prefix=yes
```

このプレフィックスのパターンは次のように正規表現で設定できます。なお、この例はデフォルト値です。

```
set delete_regexp="^(\[[A-Za-z0-9_.: \-]*\][ ]*)"
```

この$delete_prefixと$delete_regexpは、日本語パッチによって追加される拡張機能です。

MuttのIn-Reply-Toフィールドは、デフォルトのままだと、スレッドを構築できないメーラ―があります。そのため、次のように設定してあげると相手にやさしいかもしれません。

```
set in_reply_to="%i"
```

未読のメッセージがあるままMuttを終了すると、未読メッセージに自動的に付けられていたフラグ(N)が、未読の古いメッセージとして「O」(Old)フラグに変更されます。すると、次回起動時に続きの未読のメッセージを読もうとした場合に、多少不便になります。これを防ぐためには、

```
set mark_old=no
```

のようにして、$mark_oldを解除します。

メーリングリストのメールなどを振り分けて複数のメールボックスを使い分けている人は、Mailboxesコマンドで、それらのメールボックスを指定すると便利でしょう。

```
Mailboxes =list/mutt-j =list/
mutt-dev
```

これをしておくと、cキーを入力してメールボックスを変えるときに、新着メールのあるメールボックスのパスを自動的に表示して誘導してくれるようになります。

さらに、スペースキーで次のメールボックスのパスが表示されますし、ファイルブラウザで新着メールがある

表1　メッセージ作成に関する変数

変数	説明、設定例
edit_headers	ヘッダの編集を可能にします。 set edit_headers=yes
attribution	返信時に用いる文を設定します。 set attribution="On %d,\n %n wrote:\n"
abort_nosubject	Subjectフィールドを記入しなくてもメッセージを作成できるようにします。 set abort_nosubject=no
mime_forward	転送時に添付ファイルとして送るか尋ねるようにします。 set mime_forward=ask-yes
forward_format	転送時のSubjectの形式を設定します。 set forward_format="Fwd: %s"
signature	送信するメッセージの最後に付ける署名ファイルを指定します。$signatureで指定したファイルに署名を作成しておきましょう。なお、「-- \n」は署名の先頭に自動的に挿入されるので記述する必要はありません。 set signature="~/.signature"
alias_file	aliasファイルを設定します。予めaliasファイルをtouchで作成しておきましょう。 set alias_file="~/.mutt/alias" source ~/.mutt/alias

メールボックスには「N」が付くようになります。

●外部プログラムの利用

Muttは、外部プログラムと連携することにより非常に便利になります。特にMIMEタイプと外部プログラムを対応付ける「mailcap」と、メッセージからURLを抽出してプログラムに渡す「urlview」は、設定、導入の価値は十分にあるでしょう。詳しくはコラム「mailcap」と「urlview」をご覧ください。

以上、ここに示した設定例はほんの一部です。使いながら徐々にカスタマイズしてください(編集部注：次号からのMutt活用連載では、豊富なMuttのカスタマイズをどしどし紹介していく予定です。ご期待ください)。

Muttを使ってみよう

ここでは、Muttを実際に使ってみるための基本的な操作について解説します。

M 基本操作

ターミナル上で「mutt」と入力すると、Muttが起動します。Muttのインデックス画面でqキーを押すと終了します。

基本的には、最上行の「メニュー」に表示してあるキーで操作します。ラインエディタはEmacsライク、ページャはlessライクであるとういことを覚えておけば、すぐに使えるでしょう。また、キー操作で分からないことがあったら、?キーを押してヘルプで確認してください。

M 送受信

●送信

まず、新規にメッセージを作成してみましょう。Muttを起動するとスプールメールボックスのインデックスが開きます。

ここでmキーを押してください。最下行のラインエディタ上で「To:」、「Subject:」の入力を促されるので、適当に入力します。送信試験の場合は、「To:」にあなたのアドレス、「Subject:」には「TEST」や「試験」とでも入力すればいいでしょう。当然ですが、ターミナル上で日本語の入力ができる環境になっていないと日本語を入力できません。

入力が終わると、エディタが起動します。署名ファイルがあれば自動的に付けられることを確認して、適当にメッセージを入力します。日本語の設定確認も兼ねて、日本語で入力してみましょう。また、先の設定で$edit_headersを設定していれば、ヘッダも編集できます。ここでSubject:フィールドを日本語で入力し直してもいいでしょう。編集し終わったら、保存してエディタを終了します。

すると、コンポーズ(compose)画面が開きますので(画面10)、宛先などを確認したらyキーを入力して送信します。nomailなどのように送信プログラムでキューに溜めておき、すぐに送信しないものを使っている場合は、キューに溜っているメッセージを送信するコマンド(Nomailの場合はnosend)を実行します。

COLUMN 4

urlview

urlviewは、テキストファイルからURLを抜き出してウェブブラウザなどの他のプログラムに渡して処理させるプログラムです。Muttの作者のMichael Elkins氏によって作られました。インストール方法は次の通りです。

```
$ gzip -dc urlview-0.9.tar.gz | tar xvf -
$ cd urlview-0.9
$ ./configure --with-slang
$ make
# make install
```

プロトコルによって使用するプログラムを使い分けるシェルスクリプト「url_hander.sh」が、このパッケージに含まれているので、パスの通ったディレクトリにコピーし、「Configurable section」をお好みに合わせて編集してください。

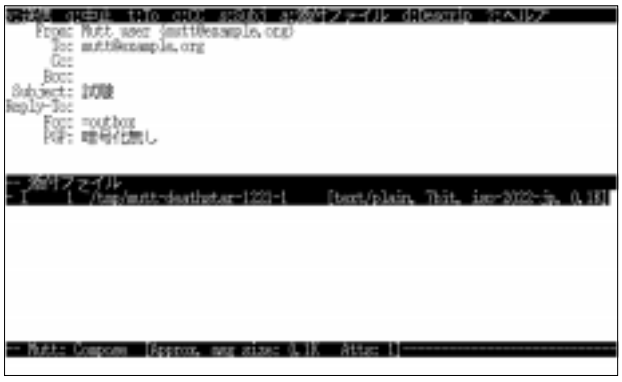
以下は筆者が設定している例です。

```
http_prgs="/usr/bin/w3m:VT"
mailto_prgs="/usr/local/bin/mutt:VT"
ftp_prgs="/usr/bin/ncftp:XT"
XTERM="/usr/X11R6/bin/kterm -rv"
```

次にMuttでの設定ですが、サイト共通の設定ファイルMuttrcに、次のmacroがデフォルトで記述されているため、ユーザごとの設定は必要ありません。しかし、何らかの理由で読み込まれなかったら、自分の設定ファイルに追加してください。このmacroにより「Ctrl＋B」でurlviewを起動できます。

```
macro index \cb |urlview\n
macro pager \cb |urlview\n
```

(滝澤隆史)



画面10　メッセージ送信の中心となるcompose画面

●受信

次に受信してみましょう。他のプログラムによってスプールに配送される場合は、ここで紹介するような操作を特に行う必要はありません。ここでは、POP3でメールを取得する例を示します。スプールメールボックスにいることを確認してGキーを入力してください。次のようなメッセージが出ますのでパスワードを入力してください。

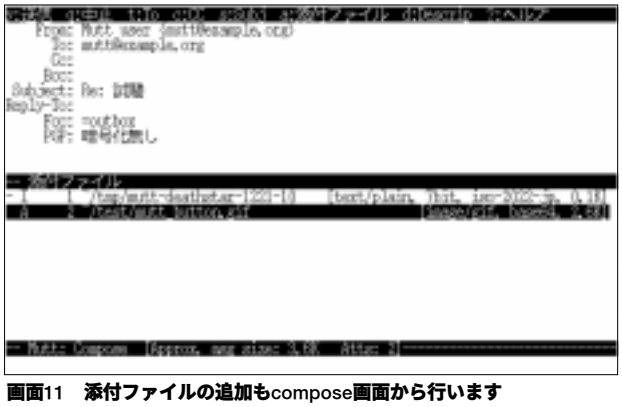
```
mutt@pop.example.org のパスワード:
```

メッセージをサーバ側で削除するかどうか尋ねられますので、受信が成功するのを確認できるまではとりあえず削除しないことにして「n」を入力します。

```
サーバからメッセージを削除して良いか? ([no]/yes):
```

これで、新しいメッセージがスプールメールボックスのインデックスに表示されれば成功です。

Enterキーを入力してページャを開き、早速その中身を読んでください。日本語で記述されたメッセージが問題なく読めたでしょうか？　読めなければ、日本語周りの設定を見直してください。なお、ページャを終了してインデックスに戻るにはiキーまたはqキーを入力します。



画面11　添付ファイルの追加もcompose画面から行います

また、スプールにあるメッセージで既読となったメッセージは、メールボックスから抜けるときに$mboxで指定したメールボックスに移動されるので、次回起動時になくなったと慌てないようにしましょう。

成功したら、今度は返信してみましょう。インデックス上で返信したいメッセージにカーソルを合わせrキーを押します。To:やSubject:には、Muttが自動的に判断した値(返信先、元サブジェクトにRe:を付けた題名)が表示され、これでいいのかと尋ねられるので、編集したければ編集し、そのままでよければEnterキーを押してください。

すると、

```
返信にメッセージを含めるか? ([yes]/no):
```

というように、返信対象のメッセージを引用するかどうか尋ねられますので、引用する場合は「y」を入力してください。

以上の入力が終わると、新規メール作成時と同様にエディタが起動するので、文書を編集した後に保存、終了してコンポーズ画面に移ります。

今度はここでファイルを添付してみましょう。添付ファイルを付けるには「a」を入力します。

```
添付ファイル('?' で一覧):
```

ここで?キーを入力するとファイルブラウザ画面になります。ディレクトリを移動し、添付したいファイルにカーソルを合わせEnterキーを入力すると、「添付ファイル」の欄にそのファイルが追加されます。添付ファイルをすべて指定し終わったら、yキーを入力して送信します(画面11)。

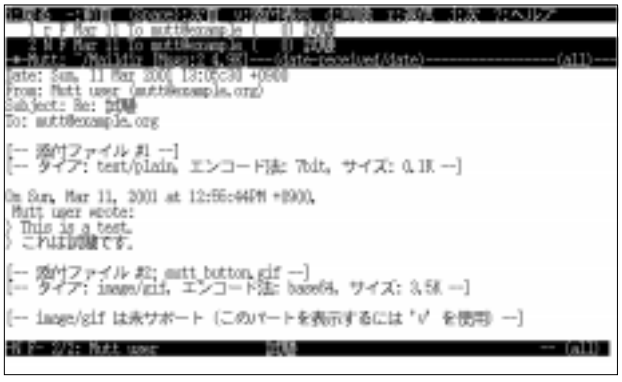
●MIMEマルチパート

先ほど返信したメッセージを受信して表示してみましょう。Muttは、マルチパートの各パートを対等に扱い、テキストなど表示できるものはそのまま表示します。表示できないものは、画面12の「添付ファイル #2」のように表示されます。

ここで、vキーを入力し、添付ファイル画面を表示します。表示したいパートにカーソルを合わせEnterキーを入力すると、MIMEのContent-Typeに対応してmailcapで指定したプログラムが起動し、その内容を見ることができます。また、sキーを入力するとそのパートを保存することができます。

●転送

今度は、先ほどの添付ファイルが付いたメールを転送をしてみます。インデックス上で返信したいメッセージにカーソルを合わせ、fキーを入力します。To:のプロンプトが出ますので、転送先を入力します。次に、Subject:が「Fw:」付きで



画面12　添付ファイル付きメッセージはこのように表示されます

自動生成され、これでいいのか尋ねられるので、編集したければ編集し、そのままでよければEnterキーを入力してください。すると、

```
添付ファイルとして転送? ([yes]/no):
```

のように、転送メッセージを添付ファイル(message/rfc822形式)として転送するかどうか尋ねられるので、お好きな方を選んでください。

後は、今まで説明した操作と同じように操作し、メッセージを送信します。受信してそのメッセージを確認してみてください。

メールボックスの移動

他のメールボックスを見るにはcキーを入力します。次のようなメッセージが出ます。

```
メールボックスをオープン('?' で一覧):
```

設定でmailboxesを指定している場合は、このときに新着メールのあるメールボックスのパスが表示されます。ここで、?キーを入力するとファイルブラウザに移るので、移動したいメールボックスを選択します。直接入力する場合は前章で説明したショートカットが使えるので利用してください。

メッセージの削除、移動、保存

メッセージを削除するにはdキーを入力します。この時点では削除マーク「D」が付くだけで、このメールボックスから抜けるとき、あるいは$キーを入力してメールボックス内の変更を明示的に書き込むときに実際に削除されます。削除マークを解除するには、まず、Jキー(Shift+j)とKキー(Shift+k)を使って削除マークのついたメッセージ上にカーソルを移動します(小文字のjキー、kキーだと、削除マークがついたメッセージは自動的にスキップしてしまいます)。そして、そこでuキーを入力します。

そのメールボックス内にあるDマークのすべてを解除するには、undelete-pattern(Uキー)という機能を使います。Uキー(Shift+u)を入力すると、パターンの入力を促されますので(パターンについては後述)、ここで「~D」を入力します。

```
メッセージの復活をするためのパターン: ~D
```

すると、~D(「Dマークがついたメッセージ」を表す)にマッチするメッセージが「undelete(復活)」されることになります。

メッセージを他のメールボックスに移動(保存)する場合はsキーを使います。このとき、

```
メールボックスに保存 : =myfriend
```

のように、分類先メールボックスとしてFrom:フィールドのアドレスに基づいたデフォルトのメールボックス名が表示されますが、これをそのまま受け入れる場合は稀でしょう。普通はこれを無視し、移動したいメールボックス名を入力します。既存のメールボックスの場合は?キーで選んでも構いません。

なお、移動を指定したメッセージには削除マークが付きますが、これは「このメールボックスからは削除される」という意味です。

移動ではなく、メッセージを他のメールボックスにコピーする場合は、Cキーを入力し、移動と同様にメールボックス名を入力します。

さらに、メッセージを特定のファイルとして保存したい場合は、いろいろやり方がありますが、メッセージをパイプで他のコマンドに渡すことができる、pipe-messageという機能を使うのが一番簡単でしょう。メッセージを表示しているところで「|」キーを押すと、

```
Pipe to command:
```

というプロンプトが表示されるので、そこに「cat - > message.txt」などのような、(パイプから)標準入力を受け取ってファイルに保存するコマンド入力します。

検索

Muttの検索機能は非常に強力で、POSIXの拡張正規表現が使え、検索パターンの指定方法は、何と36種類もあります。以下に、良く使われると思われるものを一部紹介します。詳しくはマニュアルの「4.2 Patterns」を参照してください。

~A	全てのメール
~T	タグ付けされたメール
~D	削除マークが付けられたメール
~t USER	USER宛のメール
~f USER	USERから出されたメール
~B EXPR	メッセージにEXPRを含むメール
~b EXPR	本文にEXPRを含むメール
~h EXPR	ヘッダーにEXPRを含むメール
~C EXPR	To:かCc:にEXPR を含むメール
~e EXPR	SenderフィールドにEXPRを含むメール
~i ID	Message-IDフィールドでIDに一致するメール
~s SUBJECT	SubjectフィールドにSUBJECTを持つメール

検索は、/キーで開始します。「ESC /」を入力すると、逆順検索ができます。ここでは、いくつかの検索方法の例を示します。

まずは文字列「mutt」を検索してみましょう。インデックス画面において/キーを入力すると、検索パターンの入力を促されるので、検索したい文字列(ここではmutt)を入力します。

```
検索パターン: mutt
```

単に文字列だけを入力した場合は「$simple_search」で設定したパターンに展開されます。この値は、デフォルトでは

```
~f %s | ~s %s
```

となっていますので、実際のパターンとしては、この場合、

```
~f mutt | ~s mutt
```

に展開されることになります。これはつまり、送信者とSubjectフィールドのどちらかに文字列「mutt」が含まれているメッセージを検索します。nキーを入力すると、次の検索結果に移動します。

では次に、Muttというメーラを使っているメッセージを検索してみましょう。この検索では、ヘッダ全体を検索しますので、検索パターンの指定は以下のようになります。

```
検索パターン: ~h "^X-Mailer: Mutt"| ~h "^User-
Agent: Mutt"
```

M 複数のメッセージに対する操作

Muttは、メッセージに「タグ」を付けることにより、複数のメッセージに対して同じ操作を適用することができます。

●タグの基本

タグを付ける方法は、手動で1つ1つ行なう方法と、パターンに一致したメッセージにタグを付ける方法とがあります。

手動で付ける方法としては、tキーと「ESC t」キーの2つのやり方があります。前者は単体のメッセージに、後者はカーソルがあるスレッド全体にタグを付けます。タグの解除もtキーで行います(タグが付いているメッセージ上でtキーを押せば、タグが解除される)。

パターンを指定してたタグ付けは、Tキーで行ないます。逆に、パターンをしてしていタグを解除するのは「Ctrl+T」です。

●タグに対する操作

タグが付いたメッセージすべてに同じ操作を適用するには、最初に;キーを入力してから、操作を行なうキーを入力します。

例として、複数のメッセージに対して1つの返信メッセージや転送メッセージを作成する方法を示します。

まず、同時に返信や転送を行ないたいメッセージに、手動でタグを付けます。次に;キーを入力します。すると、最下行のラインエディタに

```
tag-
```

が表示されるので、返信または転送を行なうキー(r、g、L、fなど)を入力します。例によってTo:やSubject:を確認あるいは編集すると、エディタが起動します。

返信の場合はタグを付けたメッセージがそれぞれ引用されているのを確認できます。添付ファイルとして転送する場合は、コンポーズ画面にてタグを付けたメッセージが添付されているのを確認できます。後は、通常通りです。

●複数メッセージの添付

転送とは別に、複数のメッセージを添付する場合にもタグを使います。この場合は、あらかじめタグを付けるのではなく、コンポーズ画面にて添付を行なうときにタグを付けるのがポイントです。まず、通常の作業通りにメッセージを作成してコンポーズ画面までたどり着きます。ここでAキーを入力すると、次のようなメッセージが出ます。

```
メッセージの添付のためにメールボックスを開く('?' で
一覧): ~/Maildir
```

ここで目的のファイルがあるメールボックスを指定しても良いですが、?キーを押してファイルブラウザ画面にすると、より探しやすいでしょう。添付したいメッセージのあるメールボックスを指定(ファイルブラウザなら、選択)選択したら、Enterキーを入力します。するとインデックス画面になるので、添付したいファイルにタグを付けます。付け終わったらqキーを入力します。これで、複数のファイルを一気に選択し

て添付することができました。コンポーズ画面に戻るので、後は、添付したメッセージを確認し、送信します。

●複数メッセージの移動

最後に、パターンに一致したメッセージを他のメールボックスに移動する例を紹介します。具体的には、メーリングリスト「Linuxユーザーズメーリングリスト(linux-users)」のメッセージをメールボックス「=list/linux-users/」に移動してみます。

まずインデックス画面にてTキーを入力し、パターンを入力します。linux-usersのメッセージのヘッダには「X-ML-Name: linux-users」が付いているので、これを以下のようにパターンに指定して検索を行ないます。

```
メッセージにタグを付けるためのパターン: ~h "^X-ML-Name: linux-users"
```

検索結果が出てタグが付けられたら、;キーを入力します。最下行のラインエディタに「tag-」が表示されるので、移動を行なうキー、sキーを入力します。すると保存先を尋ねられるので、ここで「=list/linux-users」と入力します。

```
メールボックスにタグ付きメッセージを保存: =list/linux-users
```

これでEnterキーを入力すれば移動が完了です。このようなメーリングリストの振り分け操作は、量が少なければ、このように手動で対応しても何とかなりますが、数が多くなってくると、付録「MDAの活用」で説明するように、外部プログラムを使った方が楽です。

付録：MDAを活用しよう

Muttの本領は、procmailやmaildropなどの振り分け可能なMDA*9を一緒に使うことで発揮されます。そこでこのコラムでは、procmailとmaildrop、さらにPOP3ユーザーに必要なfetchmailについて説明します。

fetchmail

POP3ユーザーにとっては、procmailなどの振り分けツールを使うためには、Mutt内蔵のPOP3の機能ではなく、別のPOP3クライアントのプログラムでメールを取得する必要があります。ここでは、その代表的なプログラムである「fetchmail」について説明します。といっても、非常に有名なプログラムなので、特に細かい説明は必要ないと思いますので、最低限の設定だけにとどめます。

最低限設定すべき点は、「no mimedecode」を必ず付けることと、変数mdaに、使用するMDAを指定することです。残りの設定はマニュアルを参照してください。なお、振り分けの設定ができるまでは「keep」(メールをサーバに残す)にしておいたほうが無難かもしれません。

以下に、~/.fetchmailrcの例を2つ示します。

```
# 認証はAPOPで、MDAにprocmailを使う場合
# UIDLによる新着メールの判断
# サーバのメールは残したままにする
poll pop.example.org protocol APOP
    uidl
    username mutt
    password secret-password
    no mimedecode
    keep
    mda "procmail"

# 認証は平文，MDAにmaildropを使う場合，
# 受信後サーバのメールを削除
poll pop.example2.org protocol POP3
    username mutt
    password secret-password
    no mimedecode
    fetchall
    mda "maildrop"
```

procmail

procmailも有名なので細かい説明は省き、ここではMaildir形式のメールボックスに振り分けを行なう設定例を示します。なお、Maildir形式のメールボックスは、procmailのVer.3.14以降でサポートされました。これより古いバージョンを使っている場合は注意してください。

procmailの設定ファイルは「$HOME/.procmailrc」です。この設定ファイルの先頭に、いくつかの変数を設定します。MAILDIRとDEFAULTには、Muttの設定ファイルで設定した$folderと$spoolfileと同じものを設定します。LOGFILEに

*9 「Message Delivery Agent」の略。MTAなどが受け取ったメッセージを各ユーザーのメールボックスに配送するプログラム。

は、ログを記録するファイルを指定します。振り分けが設定通りに動くことが確認できるまではログをとった方がいいでしょう。

```
PATH=/bin:/usr/bin:/usr/local/bin
MAILDIR=$HOME/Mail/
DEFAULT=$HOME/Maildir/
LOGFILE=$HOME/var/log/procmail.log
```

次に、振り分け条件を記述します。記述法は次の通りです。

```
:0［フラグ］［ : ［ロックファイル］］
<条件>
<アクション>
```

フラグに関しては、マニュアルを参照してください。またロックファイルに関しては、Maildir形式を利用している場合は設定の必要はありません。

振り分けの条件は「*」で始まります。アクションには、単にメールボックスに振り分けるだけなら、振り分け先のメールボックスを指定します。なお、「#」で始まる行はコメントと見なされますが、条件やアクションの行には書いてはいけません。

さて、実際の記述を見てみましょう。まず、記述に失敗すると、メッセージがきれいさっぱりと消えてしまうことがあるので、.procmailrcの先頭でバックアップを取るように設定しましょう。

それには、

```
# backup
:0 c
backup/
```

のように、メッセージのコピーを取るフラグ「c」を指定して、バックアップ先のメールボックスを記述します。この設定は、振り分け動作が問題なく動くようになったらコメントアウトしてもいいでしょう。

次に、実際の振り分け条件を記述します。

例えば、linux-usersメーリングリストを$HOME/Mail/list/linux-users/に振り分ける場合は、

```
# linux-users ML
:0
* ^X-ML-Name: linux-users
list/linux-users/
```

のようになります。ディレクトリ名の最後に「/」が付いていると、procmailは、そのメールボックスがMaildir形式であると認識するようになっています。なお、ここで指定しているlistというディレクトリは、あらかじめ作っておきます。

同様に、qmailメーリングリストをlist/djb-qmail/に振り分けるには、

```
# djb-qmail
:0
* ^Mailing-List: contact qmail-help@list.cr.yp.to
list/djb-qmail/
```

とします。

普通にメーリングリストのメールを振り分ける程度なら、このぐらいで十分です。試してみましょう。

maildrop

maildropは、Maildir形式をサポートしたMDAで、「Courier mail server」と一緒に配布されていますが、単体でも配布されています。AWKやPerlの文法に似たフィルタリング言語を使います。そのため、プログラミングに慣れている人には分かりやすいのではないかと思います。

設定ファイルは「$HOME/.mailfilter」で、このファイルは、パーミッションを「0600」にする必要があります。

まず、この設定ファイルの先頭に、

```
VERBOSE=6
PATH=/bin:/usr/bin:/usr/local/bin
DEFAULT="$HOME/Maildir/"
MAILDIR="$HOME/Mail"
```

のようにして、いくつかの変数を設定します。DEFAULTには、Muttの設定ファイルで設定した$spoolfileと同じものを設定します。また、振り分けが設定通りに動くことが確認できるまでは、エラーメッセージの確認のため「VERBOSE」を設定します。さらに、標準の変数ではありませんが、変数「MAILDIR」に$folderと同じものを設定します。

通常使う文法としては、if、to、ccさえ知っていれば十分でしょう。ifは他の言語と同様に次のような構文です。

```
if (expression)
{
  ……
}
else
{
```

```
  ……
}
```

toは、指定したメールボックスにメールを格納し、実行を終了します。ccは、メッセージのコピーを配送し、以降の文を実行します。

```
to expression
cc expression
```

さて、実際の記述を見てみましょう。

まず、念のためバックアップを取ることにします。ccを使いますが、メールボックスは絶対パスで記述する必要があります。先に設定した変数MAILDIRを利用するといいでしょう。ここの設定は振り分け動作が問題なく動いたらコメントアウトしても構いません。

```
# backup
cc "$MAILDIR/backup/"
```

なおmaildropは、自動的にMaildir形式のディレクトリ(cur、new、tmpというサブディレクトリを持つ)を作成してくれないので、一緒に提供されているmaildirmakeコマンドを使い、

```
$ maildirmake "$HOME/Mail/backup/"
```

などとして、あらかじめディレクトリを作成しておく必要があります。

ここでも、linus-usersを$HOME/Mail/list/linux-users/に振り分ける例を紹介します。これは、

```
# linux-users ML
if(/^X-ML-Name: linux-users/)
  to "$MAILDIR/list/linux-users/"
```

のようになります。ディレクトリ名の最後に「/」を書くことにより、Maildir形式であると認識されます。

同様に、qmailメーリングリストをlist/djb-qmail/に振り分ける例は、

```
# qmail ML
if(/^Mailing-List: contact qmail-help@list.cr.yp.to/)
  to "$MAILDIR/list/djb-qmail/"
```

となります。

普通にメーリングリストのメールを振り分けるだけなら、この程度で十分でしょう。実際に試してみてください。

RESOURCE

[1] Mutt
http://www.mutt.org/
ftp://ftp.mutt.org/pub/mutt/devel/

[2] Mutt Japanese Edition
http://www.emaillab.org/mutt/

[3] Ring Server
http://www.ring.gr.jp/
ftp://ftp.dnsbalance.ring.gr.jp/

[4] libiconv
http://clisp.cons.org/~haible/packages-libiconv.html

[5] S-Lang
http://www.s-lang.org/
http://www.actweb.ne.jp/k-yosino/slang-1.4.2jp0.tar.gz

[6] Nomail
http://www.ku3g.org/negi/nomail/

[7] nullmailer
http://www.em.ca/~bruceg/nullmailer/

[8] daemontools
http://cr.yp.to/daemontools.html
http://www.emaillab.org/djb/daemontools/
http://tanaka-www.cs.titech.ac.jp/%7Euske/doc/daemontools.html

[9] urlview
ftp://ftp.mutt.org/pub/mutt/contrib/

[10] w3m
http://ei5nazha.yz.yamagata-u.ac.jp/~aito/w3m/index.html
http://www.w3m.org/

[11] fetchmail
http://www.tuxedo.org/~esr/fetchmail/

[12] procmail
http://www.procmail.org/

[13] maildrop
http://www.flounder.net/~mrsam/maildrop/