百万人のMutt ――Mutt活用講座

滝澤隆史
taki@cyber.email.ne.jp

第3回　フック

Muttの連載も3回目になりました。国内のユーザー数が非常に少ないMuttですが、本誌の特集記事や連載記事のおかげか、増加傾向にあるようです*1。

今回は、パターンと並び、Muttの代表的な特徴の1つである「フックコマンド」について説明を行います。

その前にMuttの最新状況についてお知らせしましょう。執筆時点での最新バージョンは1.3.19です。5月下旬に開発者メーリングリストで1.4の話がちらっと出ました。コミッターのThomas Roessler氏が1.3系列も十分安定しているし、1.2に対しても十分なアドバンテージがあるので、S/MIMEのパッチを取り入れたら1.4にしたいと発言しています。そのため、具体的なスケジュールは決まっていませんが、それほど遠くないうちに1.4が出ると思います。何か進展がありましたら、この場でお知らせします。

フックの基礎

フックコマンドは、「hook-type A B」の形式で、表1に示すものがあります。「A」にはメールボックスやメールアドレスを示す正規表現やパターンを指定します。「B」には設定コマンドやメールボックスを指定します。

まず、個別のフックコマンドについて説明をする前に、共通事項を説明しておきましょう。

【リスト1】$default_hookを内部的に展開したパターン

```
~f spam@example.com !~P | (~P ~C spam@example.com)
```

【表1】フックコマンド一覧

hook-type	動作／作用
folder-hook	特定のメールボックスを開くときにコマンドを実行
send-hook	メッセージの作成時にコマンドを実行
message-hook	メッセージを整形して表示する前にコマンドを実行
mbox-hook	スプールと見なすメールボックスと読み込んだメッセージを保存するメールボックスを指定
save-hook	保存時のデフォルトのメールボックスを指定
fcc-hook	作成時のFcc:メールボックスを指定
fcc-save-hook	fcc-hookとsave-hookを一緒に指定
pgp-hook	宛先のPGP鍵を選択
charset-hook	文字符号化方式の別名を定義
iconv-hook	システムの文字符号化方式を定義

●デフォルトのコマンドの指定

設定コマンドを実行する「folder-hook」、「send-hook」、「message-hook」では、コマンドを実行して設定を変更した場合、そのフックから抜けた後でも変更した設定が継続して有効になります。そのため、パターンにデフォルトを示す「.」を指定したフックコマンドを先に指定しておかなければなりません。例えば、次のように設定すると、$recordフォルダ*2ではインデックスを送信日順に表示しますが、それ以外のフォルダではスレッド表示するようになります。

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

フックコマンドは、記述した順に順次実行されるので、デフォルトのコマンドは他のフックコマンドより前に記述しなければなりません。なお、Muttの設定を変えるようなコマンドでなければ、デフォルトのコマンドを指定する必要はありません。

●default_hook

後述しますが、「save-hook」と「message-hook」は、その性質上、送信者を示すパターンを指定することが多いでしょう。また、「send-hook」と「fcc-hook」は、逆に宛先を示すパターンを指定することが多いでしょう。そこでMuttでは、パターンではなく単にメールアドレスを(正規表現で)記述すれば、送信者や宛先を表すパターンに展開するようになっています。このパターンは変数$default_hookで設定でき、デフォルト値は「~f %s !~P | (~P ~C %s)」です。「%s」の部分がメールアドレスに置き換えられます。

例として、送信者がspam@example.comであるメッセージの保存時のデフォルトのメールボックスを「$folder/spam」にしたい場合を考えてみます。これを行うために、フックコマンドsave-hookを用いて、次のようにパターンとして単にメールアドレス「spam@example.com」を記述します。

```
save-hook spam@example.com =spam
```

このspam@example.comは、Muttの内部処理的には、リスト1に示すパターンに展開されて実行されます。

これは「送信者(From)がspam@example.comである(あなたが

*1　日本語版パッチのダウンロード数から判断すると100人以上増えている気がします。
*2　送信したメッセージを保存するフォルダを指定する変数。ショートカットは「<」。詳細は本誌2001年6月号（44ページ）を参照。

送ったものを除く)。あるいは、あなたからspam@example.com宛に送ったもの」という意味になり、意図したパターンを含んでいることが分かります。なお、正規表現として厳密に書けば「^spam@example\.com$」の方が間違いがないのかもしれませんが、ここでは簡略化して、単にメールアドレスそのものを書いています。

●unhook

定義したフックを取り除きたい場合には「unhook」を用います。文法は次の通りです。

```
unhook [ * | hook-type ]
```

例えば、Muttを起動している途中で、設定した「send-hook」をすべて取り除きたいときは、設定コマンドを入力する「:」を入力後、次のように「unhook」コマンドを入力します。

```
unhook send-hook
```

すべてのフックを取り除くときは次のようなコマンドを入力します。

```
unhook *
```

●push

pushは、フックとは直接には関係ない設定コマンドですが、フックと一緒に用いると非常に便利なのでここで紹介します。

pushは、メールボックスやメッセージを開いたときに、連続したキー操作をキーボードバッファに加えるコマンドです。言い換えれば、メールボックスやメッセージを開いたときに実行される「キーボードマクロ*3」のようなものです。文法は次の通りです。

```
push <string>
```

「string」の部分は、マクロのキーシーケンスと同じ記述法です。このコマンドは、通常は起動時に開くメールボックス(スプールフォルダ)に対して働きます。設定コマンドを実行するフックコマンド「folder-hook」、「message-hook」と組み合わせれば、その対象のメールボックスやメッセージに対しても働くようになります。

例として、$recordフォルダを開いたときに、インデックスを送信日順にソートするようにしてみましょう。

```
folder-hook < 'push od'
```

「o」はsort-mailboxで、「d」はdate(送信日順)です。ただし、あくまでもキー操作であるため、メールボックスの読み込みが終わった後に実行されることに注意してください。

*3 本誌2001年7月号を参照してください。

なおpushコマンドは、setコマンドでMuttの設定を変えているわけではないので、そのコマンドを実行した影響は後には出てきません。そのため、デフォルトの設定は必要ありません。

M 設定コマンドを実行する

ここでは設定コマンドを実行するフックコマンドについて解説します。先に述べたデフォルトのコマンドを設定することを忘れないでください。

●folder-hook

メールボックスごとに設定などを変えたい場合は、「folder-hook」を使います。文法は次の通りです。

```
folder-hook <regexp> <command>
```

メールボックスを開くときに、そのメールボックスが正規表現「regexp」に一致していれば、設定コマンド「command」を実行します。コマンドが実行されるタイミングは、メールボックスを読み込む前です。

正規表現「regexp」は、メールボックスのファイル名(ディレクトリ名)だけでなく、絶対パスで比較されます。例えば、次の例のように$folder以下にサブディレクトリmagazineを設け、その下に複数のメールボックスを置いたとします。

```
$folder + magazine + mag1
        |          + mag2
        |          + mag3
```

この場合、メールボックスmag1のパスは「=magazine/mag1」と認識されます。mag2とmag3も同様です。そのため、正規表現として「magazine」を記述すればmagazine以下のすべてのメールボックスに一致します。

次にインデックスの表示に関する例を示します。まず、デフォルトの設定を行います。通常はスレッドで表示する場合は次のようにします。

```
folder-hook . 'set sort=threads'
```

続いて、個別のメールボックスの設定を行います。$spoolfile、$mboxを受信日順に、$record、$folder/magazine以下のメールボックスを送信日順にソートする場合は次のようにします。

```
folder-hook '!' 'set sort=date-received'
folder-hook > 'set sort=date-received'
folder-hook < 'set sort=date-send'
folder-hook magazine 'set sort=date-send'
```

なお、$spoolfileのショートカット「!」を使うときは必ず引用符で囲ってください。囲まないと、論理否定演算子と判断されます。

●send-hook

メッセージを作成するときに、宛先に応じて設定を変えたい場合は「send-hook」を使います。文法は次の通りです。

```
send-hook ＜pattern＞ ＜command＞
```

作成するメッセージがパターン「pattern」に一致したときに、設定コマンド「command」を実行します。コマンドが実行されるタイミングは、メッセージの作成時にラインエディタ上で「To」や「Subject」などを入力した後です。編集した後に「To」や「Subject」を追加したり編集したりしても、コマンドがさらに実行されることはありません。

使用できるパターンとして、プログラムの仕様的には「~h」、「~b」、「~B」以外のものがすべて使えるのですが、実質的には、作成するメッセージが持っている情報(To、Cc、Fromなどのメールアドレスと Subject)が利用できるものだけです。通常は、単に正規表現を記述して「$default_hook」で展開させるか、Toに一致する「~t」を使うくらいでしょう。

宛先に応じて署名ファイルを変える例をリスト2に示します。

【リスト2】宛先に応じて署名ファイルを変える

```
send-hook . 'set signature=~/.signature-ja'
send-hook '~t @example.org' 'set signature=~/.signature-en'
```

デフォルトでは$signatureに日本語の署名ファイル「~/.signature-ja」を指定し、example.orgドメイン宛には英語*4の署名ファイル「~/.signature-en」を指定します。

●message-hook

メッセージごとに設定を変えて表示させたい場合はmessage-hookを使います。

文法は次の通りです。

```
message-hook ＜pattern＞ ＜command＞
```

表示されるメッセージがパターン「pattern」に一致したとき、設定コマンド「command」を実行します。コマンドが実行されるタイミングは、メッセージを読み込んだ後で、表示したり整形したりする前です。なお、このコマンドはバージョン1.3.2から「display-hook」として追加され、バージョン1.3.5で「message-hook」に変わりました。message-hookの使用例は「フックコマンドの応用」で紹介します。

M メールボックスを指定する

ここではメールボックスを指定するフックコマンドについて

Column 1.2系列Mutt日本語パッチについて

Muttの系列は、Linuxカーネルのように、「安定版」と「開発版」に分類できます。識別方法もLinuxカーネルと同じで、バージョン番号「x.y.z」のマイナーバージョン番号「y」が偶数ならば「安定版」、奇数であれば「開発版」です。「安定版」とは文字通りその時点でもっとも安定している（であろう）バージョン、「開発版」とは次期安定版に取り込まれる予定の新機能を実装／テストするためのバージョンです。

現在精力的に開発が進められているMuttの1.3系列は「開発版」と位置付けられているものの、最近のバージョンでは十分に安定して動作しているため、日常的に使用しても全くといっていいほど問題はありません。1.3系列の一番の特徴は国際化が進められていることにあり、オリジナルの状態で基本的な日本語を扱うことができるようになっています。しかしその反面、内部で文字コード変換が頻繁に発生するためか、環境によっては1行スクロール動作などに若干の重たさを感じる場合もあるようです。もっとも、CPUパワーのあり余っているような最新のマシンではあまり問題にならないと思われますが。

一方、2001年6月中旬の時点で「安定版」とされている1.2系列では十分な国際化がなされておらず、マルチバイト文字はサポートされていません。そのため、オリジナルの状態では日本語を通さないので、日本語メールリーダとして使うためには日本語パッチ（通称jpパッチ）を当てることが必須になります。しかし1.2系列は1.3系列と比較して動作が軽いという特徴がありますから、Mutt 1.3系列でのスクロール速度に不満がある場合は1.2系列を試されてみてはいかがでしょうか。ここでは1.2系列のMutt日本語パッチについて簡単にご紹介いたします。

2001年6月中旬の時点での1.2系列の最新版は、2000年7月28日にリリースされた1.2.5iです。「最新版」といってもリリースされてすでに1年ほどになりますから、1.2系列としてのリリースはそろそろ打ち止めかもしれません。この1.2.5iで日本語を読み書きするには、前述のように日本語パッチを当てる必要があります。現在このバージョンに対応した日本語パッチとして1.2.5i-jp2(mutt-1.2.5i-jp2-diff.gz)が提供されおり、「Mutt Japanese Edition」のページなどからダウンロードすることができます([4]、[5])。

1.2.5i-jp2の基本的な機能は、1.3系列のMuttにオプショナルな日本語パッチ（通称jpパッチ）を当てた場合とほぼ同等ですが、jaパッチを開発する際に、廃止された機能を含んでいたり、同等の機能であってもオプション名が異なる場合があります。オリジナルのMutt 1.2.5iに対して、日本語パッチが提供する機能としては、

- 日本語の表示(表示コードはEUCのみサポート)
- Muttの表示するメッセージの日本語化
- 日本語の添付ファイル名対応
- JIS罫線文字によるスレッド表示対応
- メーリングリストなどの「Subject:」に付加されている通し番号の非表示
  （通し番号部分のユーザーカスタマイズは不可）
- メール本文表示時に本文非表示指定や文字置き換えが可能
- APOP対応

などが挙げられます。ここでは各機能の詳細な説明やインストール方法は省略させていただきますが、興味のある方は、mutt-1.2.5i-jp2-diff.gzに含まれるリリースノートをご一読ください。

（吉田行範）

＊4 最近はorgやcomドメインだから英語を使うというわけではないような状況なので、本誌2001年7月号のキーボードマクロの例を使うしかないのかもしれません。

解説します。

●mbox-hook

複数のスプールフォルダを使いたい場合にmbox-hookを使います。文法は次の通りです。

```
mbox-hook ＜regexp＞ ＜mailbox＞
```

メールボックスが正規表現「regexp」に一致したときに、このメールボックスをスプールフォルダとして扱い、終了時やメールボックスを変えるときに既読のメッセージを「mailbox」に移動します。一時的に$spoolfileと$mboxを設定したようなものと考えると分かりやすいかと思います。ただし、実際に$spoolfileと$mboxの値が変更されている訳ではありません。

mbox-hookの使用例は「フックコマンドの応用」で紹介します。

●save-hook

「s」(save-message)や「C」(copy-message)などでメッセージを保存するときのデフォルトのメールボックスを指定したいときにsave-hookを使います。文法は次の通りです。

```
save-hook pattern ＜mailbox＞ *5
```

メッセージを保存するときに、保存するメッセージがパターンpatternに一致したら、保存先のデフォルトのメールボックスをmailboxにします。

次の例は、freemail.example.comドメインから来たメッセージを保存するときのデフォルトのメールボックスを$folder/trashにする例です。

```
save-hook freemail.example.com =trash
```

この場合は、パターンとして単に正規表現を記述しただけなので、$default_hookで設定したパターンに展開されます。

なお、通常のデフォルトのメールボックスとしては、From(あなたが送信者の場合はTo)に記述されたメールアドレスの「@」の左側の文字列が使われます。例えば、foo@example.comから来たメッセージの場合は「=foo」になります。

●fcc-hook

作成したメッセージを宛先に応じてFccメールボックスを指定したい場合にfcc-hookを使います。文法は次の通りです。

```
fcc-hook ＜pattern＞ ＜mailbox＞
```

作成したメッセージがパターン「pattern」に一致したときに、Fccメールボックスとしてmailboxに保存します。

使用できるパターンとしてはsend_hookと同じで　プログラムの仕様的には「~h」、「~b」、「~B」以外のものをすべて使えるのですが、実質的には作成したメッセージが持っている情報(To、Cc、Fromなどのメールアドレスと Subject)を利用できるものだけです。通常は、単に正規表現を記述して$default_hookで展開させるか、Toに一致する「~t」を使うくらいでしょう。

例として、foo@example.org宛のメッセージの複製を$folder/fooに保存する場合を示します。

```
fcc-hook foo@example.org =foo
```

●fcc-save-hook

ある特定の人とのメールのやり取りを同じメールボックスに保存したい場合は「fcc-save-hook」を使います。文法は次の通りです。

```
fcc-save-hook ＜pattern＞ ＜mailbox＞
```

これは「fcc-hook」と「save-hook」を一緒に設定したのと同じ効果があります。

M PGP関連のフックコマンド

●pgp-hook

PGPでメッセージの暗号化を行うときに、相手のメールアドレスに従って、公開鍵が選ばれますが、相手が複数の公開鍵を持っていたりする場合など、特定の公開鍵を使いたい場合には「pgp-hook」を使います。文法は次の通りです。

```
pgp-hook ＜regexp＞ ＜keyid＞*6
```

メールアドレスが正規表現「regexp」と一致するときに公開鍵の鍵ID「keyid」を選びます。

例えば、foo@example.orgの公開鍵に鍵ID「0x01234567」を使いたい場合には、次のように設定します。

```
pgp-hook foo@example.org 0x01234567
```

M 文字符号化方式関連

MIMEで使用する文字符号化方式(Content-Typeヘッダフィールドのcharsetパラメータなど)には、IANA(Internet Assigned Numbers Authority、記事末のRESOURCE [1]を参照)に登録された文字符号化方式の名前([2])が使われます。しかし、使用する環境(OS)のiconv関数*7によっては、IANAに登録されていない名前を使ったりするものもあります。例え

＊5　Muttのマニュアルでは「filename」と書いてありますが、意味的には「mailbox」の方が分かりやすいと考え、「mailbox」に変えました。
＊6　Muttのマニュアルでは「pattern」と書いてありますが、意味的には「regexp」の方が分かりやすいと考え、「regexp」に変えました。
＊7　文字符号化方式の変換関数

【リスト3】返信時の署名の選択

```
message-hook . 'set signature=~/.signature-ja'
message-hook '~h "^content-type:.*(us-ascii|iso-8859-1)"' 'set signature=~/.signature-en'
```

【リスト4】メッセージ中の見出しの表示

```
message-hook '~s "Newsletter"' 'push "/(^^[IVX]+\\..*$|^^[0-9]+\\..*$)\n"'
```

ば、あるOSでは「ISO-8859-1」を「ISO8859-1」という名前で扱っています。このような不一致を補正するために「charset-hook」と「iconv-hook」が用意されています。

●charset-hook

受け取ったメッセージの文字符号化方式の名前をMuttが理解できない(厳密に言えばiconv関数が知らない)場合に「charset-hook」を使います。文法は次の通りです。

```
charset-hook ＜alias＞ ＜charset＞
```

Muttが知らない「alias」に指定した文字符号化方式の名前を、それと同じか、あるいはほぼ同じ内容を示す文字符号化方式である「charset」に対応付けます。

例として、IANAに登録されていない「ISO-2022-JP-3」を部分的に表示可能な「ISO-2022-JP」の別名として定義した例を示します。

```
charset-hook iso-2022-jp-3 iso-2022-jp
```

●iconv-hook

IANAで登録されている文字符号化方式の名前とiconv関数で使用している名前が異なるときには「iconv-hook」を使います。文法は次の通りです。

```
iconv-hook ＜charset＞ ＜local-charset＞
```

IANAで登録されている文字符号化方式の名前「charset」を、iconv関数で使用している名前「local-charset」に対応付けます。なお、このコマンドは、バージョン1.3.16から追加されたものです。GNU libiconvライブラリやglibc-2.1.3以上を使っている場合は必要ありませんが、/usr/local/doc/mutt/samples/iconvに、OSごとの設定ファイルがあるので、必要であればsourceコマンドで読み込むようにしてください。

Mフックコマンドの応用

●返信時の署名の選択

返信するメッセージを作成するときに、返信元のメッセージが英語であれば英語の署名を、日本語であれば日本語の署名を付けたいでしょう。この場合、すぐに思い浮かべるのは作成時に使う「send-hook」の利用です。しかし、send-hookの項目で説明したように、「send-hookは、作成するメッセージが持っている情報しか扱えない」ので、宛先のメールアドレスで判断するならばともかく、このようなケースでは全く使えません。そこで、返信時にメッセージが読み込まれることを利用して、メッセージを読み込むときに実行される「message-hook」を使います。

リスト3に示す例では、デフォルトでは日本語の署名ファイルを設定し、返信元のメッセージのContent-Typeヘッダフィールドのcharsetパラメータが「US-ASCII」あるいは「ISO-8859-1」であれば、英語の署名ファイルを設定します。

●メッセージ中の見出しの表示

ニュースレターやメールマガジンで送信されるメッセージは長いものが多く、とても全部は読んでいられないと思うことがよくあります。このときは、内容の見出しを拾い、興味のある記事だけを読みたくなります。このような場合には、リスト4のように「message-hook」を用いて見出し行を検索させるコマンドを実行するようにします。

この例では、Subjectが「Newsletter」であるメッセージの場合に、「I.」のようなローマ数字や「1.」のような数字で始まる見出し行をページャの先頭行に表示します。「n」(search-next)で次の見出しが表示されるので*8、記事をざっと眺めるのには役立つと思います(画面1)。

【画面1】見出しの表示の例

＊8 「search-opposit」のキー定義をすると、前の見出しに戻れるので、さらに便利になります。定義の例は本誌7月号を参照してください。

M 月ごとにメールボックスの名前を変える

Muttは、メールボックスにメッセージがたくさん(数千通*9)溜ると、そのメールボックスを開くときのメッセージのスキャンに時間がかかるようになります。そこで、リスト5のように、「mbox-hook」を使用し、月ごとに既読メッセージの保存先を変えて、メッセージが溜まり過ぎるのを防いでみます。

この例では、procmailなどで$folder/mutt/spoolに配送されたメッセージで既読になったものを「$folder/mutt/2001-08」のような、月ごとのメールボックスに保存するようになっています。これは、Muttの設定ファイルでは「`」で囲まれたコマンドの出力が展開されることを利用しています。そのため、「‘date +%Y-%m‘」は今日が2001年8月であれば「2001-08」のように展開されます。

この例は「The Mutt FAQ」([3])の「Can Mutt be configured to rename folders at the end of the month, like Pine?」を応用したものです。

【リスト5】メールボックスの名前を変える

```
mbox-hook =mutt/spool =mutt/‘date +%Y-%m‘
```

M 最後に

今回までの記事で、カスタマイズするために必要な内容をおおよそ説明したことになります。次回からは、個別の機能について説明したり、TIPSを紹介したりします。

Resource

[1] Internet Assigned Numbers Authority
http://www.iana.org/

[2] CHARACTER SETS
http://www.iana.org/assignments/character-sets

[3] The Mutt FAQ
http://www.fefe.de:80/muttfaq/faq.html.gz

[4] 1.2系のjaパッチダウンロード先
http://www.geocities.co.jp/SiliconValley-PaloAlto/2594/

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

Column

Muttも歩けば棒に当たる

■Muttとの出会い

はじめまして、安定系Mutt日本語パッチメンテナーの吉田です。といっても最近あまりメンテナンスしていないんですけど。

それは1997年春のことでした。当時の私はメールとニュースを読み書きするのにmnewsを愛用していましたが、ひょんなことからニュースリーダをslrnに乗り換えました。その後、しばらくはmnewsをメールリーダとして使い続けていましたが、mnewsの機能の半分(?)しか使わないのが悔しかったので、ついでにメールリーダも乗り換えることにしました。いま考えるとおかしな理由です。

そして、次期メールリーダを探すべくあちらこちらのFTPサイトからアーカイブを拾ってきてはせっせとmakeしていたある日、ついにMuttと出会いました。ピン！ と来ました。これだ！ と思いました。しかし、当時のMutt(確か0.69くらいでした)は、日本語を全く通さず、日本語メールリーダとしては実用になりませんでした。

あきらめかけたそのとき、ふとslrnのことが頭に浮かびました。「slrnの日本語パッチを参考にして日本語が通るようにしちゃえばいいじゃないか」と。幸いMuttもslrnと同じく「S-Lang」一族だったこともあり、さほど手間をかけることなく、翌日には最低限の日本語の読み書きができるようになっていました。これが最初のMutt日本語パッチです。

その後は、当時slrnの日本語パッチメンテナーだった菊谷さんを始めとする方々に助けていただきながら、さまざまな修正や機能追加を行ってきました。現在では、mutt-jメーリングリストの複数メンバーによって開発／保守が進められています。多くの人との素晴らしい出会いを与えてくれたMuttは、私にとって忘れることのできないソフトウェアです。

■私流Muttの使い方

最初に白状すると、私はそれほど凝った使い方をしていません。Muttの使いこなしについて書かれたWebページを見たりして、「へぇ、こんな機能があったのか」と感心することもしょっちゅうです。そういう訳で、あまり参考になるようなことは書けませんが、数年来重宝しているちょっとした工夫をご紹介します。

私は、割といろいろなメーリングリストに参加していることもあって、受信したメールの振り分けは必須の機能です。これには、Muttのフォルダ機能を使って分類したり、受信するそばからprocmailなどで振り分けてしまうのが一般的だと思いますが、量が多いと、いちいち手作業で分類するのも手間ですし、かといって未読メールを読むためにフォルダを移るのも面倒です。単に私が不精なだけなんですけど。そこで、受信したメールはすべてメールスプール上で読んで、既読のものをprocmailで各フォルダに振り分けるようにしています。

まずはfetchmailなどが直接procmailを起動しないようにしておきます。次に$HOME/.muttrcに以下の設定をします。

```
set mbox_type=mbox
set mbox=+inbox
set move=yes
```

これでMutt終了時にメールスプールの既読メールが$HOME/Mail/inboxにmbox形式で保存されます。あとはリスト6のシェルスクリプトをcronで定期的に走らせます。ご自分の環境に合わせてカスタマイズしてください。 (吉田行範)

【リスト6】メッセージ振り分けスクリプト

```
#!/bin/sh
MGbytesOX=$HOME/Mail/inbox
RCFILE=$HOME/.procmailrc
LOCKFILE=$MGbytesOX.lock

if [ -f $LOCKFILE ]; then
  exit 1
elif lockfile -! -r 1 $LOCKFILE >/dev/null 2>&1; then
  exit 2
else
  trap "rm -f $MGbytesOX" 1 2 3 13 15
  if [ -f $MGbytesOX -a -n $MGbytesOX ]; then
    formail -s procmail -m $RCFILE < $MGbytesOX && rm -f $MGbytesOX
  fi
  rm -f $LOCKFILE
  exit 0
fi
```

*9 ひたすら速いハードディスクドライブを使えば、1万通くらいまでは我慢できるようになります。