百万人のMutt

～Mutt活用講座～

滝澤隆史
taki@cyber.email.ne.jp

第8回　LDAP

相も変わらずMutt 1.4はまだ出てなくて、執筆時点でのバージョンは1.3.24iです。それで、現在何が行われているかというと、スレッド表示回りで大きな変更が加えられたところです。新しいバージョンでスレッド表示を行うと、ツリーに「?」や「=」が出てくることがあるので驚かないでください。この件については今月のコラム「スレッドツリー」で簡単に紹介します。

筆者はMutt日本語版のページだけではなく、「MUAs for Windows」*1（記事末のResource [1] を参照）というページも運営しているのですが、評価したメーラの中でLDAPを用いたメールアドレスの検索機能が付いているものがいくつかあります。我らがMuttも外部プログラムと連携することで、LDAPでの検索を行うことができます*2。

LDAP検索の機能は、使い始めるとかなり重宝するもので、会社などの組織では、アドレス帳などと連携して利用されている方もいるでしょう。しかし、当然のことながらLDAPの機能は、LDAPサーバが用意されていなければ使えません。「ではLDAPサーバでも立ち上げてみるか」と思って、OpenLDAPをインストールしてみたものの、最初は「はて、どうやってメールアドレスを登録すればいいの？」と途方に暮れてしまうことでしょう。そこで、今回はLDAPを利用したアドレス帳の作り方を紹介します。また、筆者が作っている検索スクリプトも紹介します。

前準備

●OpenLDAP

今回の記事ではLDAP関連のプログラムとして、LDAPv3対応の「OpenLDAP」（[2]）のバージョン2.0を利用することを前提として話を進めていきます。OpenLDAPのインストール方法や初期設定に関しては、稲地稔氏のサイト「OpenLDAP管理者ガイド」（[3]）に詳しく掲載されているので、そちらを参照してください。また、RPMなどでパッケージングしたものも多く提供されているので、それを導入してもいいでしょう。

●Perlのモジュール

今回紹介する検索スクリプトや関連するスクリプトはPerlで記述されているので、Perlがインストールされていなければインストールしてください。また、スクリプトはPerlの次のモジュールを必要とします。

```
Jcode
MIME::Base64
Net::LDAP
```

これらは「CPAN」（[4]）からダウンロードできます。今月のコラム「Perlのモジュールのインストール方法」で紹介した方法でインストールを行うのが楽でしょう。

●LDAPの基礎知識

LDAPについてまじめに説明すると、連載1回分では書ききれないので省略します。その代わりといってはなんですが、参考サイトとして「OpenLDAP管理者ガイド」（[3]）と「LDAP Linux HOWTO」（[5]）を一通り読んでみてください。LDAPに関する最低限必要な知識が身に付くはずです。

slapdの設定

ここではアドレス帳としてのLDAPサーバを構築するためのslapd（OpenLDAPにおけるLDAPデーモン）の設定ファイルslapd.confに最低限記述するべき項目について説明します。

●スキーマの登録

リスト1に示した4つのスキーマが登録されていないようであれば登録してください。パスはインストールした状況に応じて変えてください。

●rootdnなどの設定

まず、使用するディレクトリ階層を決めます。ここでは「dc=example,dc=org」とします。バックエンドデータベースに問い合わせる際、ディレクトリ階層を接尾語として付加する指示をする「suffix」に続けてディレクトリ階層を記述します。

次に、LDAPデータベースに対して「何でもできる権限」を持つDN（Distinguished Name：識別名）であるroot DNの情報を指定します。ここではroot DNを「cn=Manager,dc=example,dc=org」とします。rootdnにはroot DNを記述し、rootpwにはそのパスワードを記述します。このパスワードはslappasswdの出力を記述してください。なお、SASL認証の話は本記事では省略します。以上の設定を行った例をリスト2に示します。

ここで設定したroot DNは、後ほどエントリの登録で使用します。実際にはroot DNとは別に登録用のDNを別途作成した方がよいと思います。

●属性の選択

検索を行うためには、属性（attribute）ごとに索引を作ります。この「索引を作る」指定はslapd.confに記述します。アドレス帳に必要な属性は何かを考えると、一般的には表1のようなものが挙げられると思います。この属性はすべてobjectClassの「inetOrgPerson」およびその継承元の「organizationalPerson」、「person」で定義*3されたものです。特に、メー

【リスト1】スキーマの登録（slapd.conf）

```
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/nis.schema
```

【リスト2】サフィックスとroot DNの設定（slapd.conf）

```
suffix "dc=example,dc=org"
rootdn "cn=Manager,dc=example,dc=org"
rootpw {SSHA}kRIEpgTXxTRTy45ehf87fO7+PXFfJWv3
```

＊1　1年くらい更新していないので、いくぶん情報は古いものとなっています。　＊2　この概要については2001年7月号で紹介しています。

ルアドレスを調べるという目的と、スキーマの定義からすると、cn、sn、mailが最低限必要なものです。

表1にobjectClassはありませんが、定義上必要なものです。その他の属性は必要に応じて追加します。また、表1にはなくてもinetOrgPersonクラスで定義されている属性であれば使用できます。ここではsn、givenName、oを追加することにします。この場合のslapd.confの例を示します。

```
index objectClass            eq
index cn,mail,sn,givenName,o eq,sub
```

以上で最小限の設定が完了しました。それではslapdを起動してみましょう。

LDIF

●LDIFの形式

LDAPデータベースにエントリを登録したり取り出したりする際のデータ形式には、「LDIF」という形式が使われます。エントリの登録を行う前には、このLDIF形式で記述したファイルをエントリごとに作成しておく必要があります。

LDIFの基本的な形式は次の通りです。

```
dn: ＜識別名＞
＜属性記述子＞: ＜属性値＞
＜属性記述子＞: ＜属性値＞
＜属性記述子＞:: ＜base64符号化値＞
＜属性記述子＞:< ＜URL＞
        ⋮
```

＜識別名＞にはエントリを識別する名前を指定します。単に「DN」と記述しているものはこの識別名のことを示しています。＜属性記述子＞には、cn、mailなど属性の名前を指定します。「cn;lang-ja」などのように、属性記述子と「;」に続けてオプションを記述することができます。＜属性値＞は属性の値を指定します。＜base64符号化値＞は、属性値がコロン「:」やスペース「 」や小なり「<」で始まる場合や、非表示文字、ASCII以外の文字を含んでいる場合に、属性値をbase64で符号化して表すためのものです。＜URL＞には、属性値をファイルから取得する場合に、ファイルを「file://」の形式で指定します。「#」で始まる行はコメントとみなされます。エントリが複数あるときは空行で区切ります。

LDIFファイルの例をリスト3に示します。リスト3には2つのエントリが記述されています。1つ目はドメイン「dc=example,dc=org」のエントリです。2つ目はドメイン「dc=example,dc=org」に所属するユーザー「Taro Itsutsubashi」のエントリです。

●LDIFと日本語

次に日本語を扱う場合の処理について説明します。

属性値に日本語が含まれる場合は、先ほどLDIFの形式の説明で触れた＜base64符号化値＞の形式で指定する必要があります。処理としては、日本語を含んだ属性値の文字列の文字コードをUTF-8に変換してからbase64に符号化するようにします。例えば、「五橋 太郎」という名前をcnに記述したい場合は、リスト4の(2)のようになります。この変換にはリスト5に示したスクリプトを用いると簡単にできます。

```
$ echo "五橋 太郎" | utf8b64.pl
5LqU5qmLIOWkqumDjg==
```

また、属性値がどの言語で記述されているかを示すのに、言語コードを含んだ言語オプションを属性記述子に付けることもできます。言語オプションは「lang-」に続けて言語コードを記述します。例えば日本語の場合、言語コード「ja」または「ja-JP」を指定して、「lang-ja」または「lang-ja-JP」のようになります。先ほどの「五橋 太郎」さんの例では、リスト4の(3)のようになります。ローマ字で表記する場合はリスト4の(4)のように「lang-en」を付けて記述することもできます。

さらに、属性値が読み(読み仮名)であることを示したい場合は「phonetic」を付けます。「いつつばし たろう」という読みを付ける場合は、リスト4の(5)のようになります。ただし、執筆時点で最新であるOpenLDAP 2.0.18では対応していないので、このオプションは実際には使いません。

リスト3の「Taro Itsutsubashi」のエン

【表1】アドレス帳に必要な属性

属性	意味
cn	共通名
mail	メールアドレス
sn	姓
givenName	名
ou	所属部
o	所属組織
telephoneNumber	電話番号
facsimileTelephoneNumber	FAX番号
description	備考

【リスト3】LDIFファイルの例

```
# example.org
dn: dc=example,dc=org
dc: example
objectClass: dcObject

# Taro Itsutsubashi
dn: cn=Taro Itsutsubashi,dc=example,dc=org
cn: Taro Itsutsubashi
sn: Itsutsubashi
givenName: Taro
mail: t5b@example.org
o: Example Co.
objectClass: inetOrgPerson
```

【リスト4】LDIFの記述例

```
cn: Taro Itsutsubashi                                    …… (1)
cn:: 5LqU5qmLIOWkqumDjg==                                …… (2)
cn;lang-ja:: 5LqU5qmLIOWkqumDjg==                        …… (3)
cn;lang-en: Taro Itsutsubashi                            …… (4)
cn;lang-ja;phonetic:: 44GE44Gk44Gk44Gw44GXIOOBn+OCjeOBhg==  … (5)
```

【リスト5】utf8b64.pl

```
#!/usr/bin/perl -w
use Jcode;
use MIME::Base64 qw(encode_base64);

while(<STDIN>){
  chomp;
  print encode_base64(Jcode->new($_,'euc')->utf8);
}
```

【リスト6】リスト3の「Taro Itsutsubashi」のエントリを日本語で記述した例

```
# Taro Itsutsubashi
dn: cn=Taro Itsutsubashi,dc=example,dc=org
cn: Taro Itsutsubashi
cn;lang-ja:: 5LqU5qmLIOWkqumDjg==
cn;lang-ja:: 44GE44Gk44Gk44Gw44GXIOOBn+OCjeOBhg==
sn;lang-ja:: 5LqU5qmL
sn;lang-en: Itsutsubashi
sn;lang-ja:: 44GE44Gk44Gk44Gw44GX
givenName;lang-ja:: 5aSq6YOO
givenName;lang-en: Taro
givenName;lang-ja:: 44Gf44KN44GG
mail: t5b@example.org
o;lang-ja:: 4peL4peL5Lya56S+
o;lang-en: Example Co.
objectClass: inetOrgPerson
```

＊3 /etc/openldap/schemaディレクトリにあるcore.schemaとinetorgperson.schemaを参照。

トリを日本語で記述した場合の例をリスト6*4に示します。見ての通り、いくつも同じ属性を記述することができます。検索時にはすべて検索の対象になりますが、検索結果を取得した際、LDAPクライアントは同じ属性のうち最初のものだけを結果として表示する傾向があります(後述する検索スクリプトも同様です)。そのため、同じ属性の中では、検索結果として表示したいものを先に記述した方がよいでしょう。例えば、リスト6のsnとgivenNameでは「日本語（漢字)」、「ローマ字」、「日本語（平仮名)」という順に記述しています。もちろん言語コードを選択して結果を表示してくれるクライアントがあれば話は別です(後述する検索スクリプトでは対応しています)。

●LDIF生成スクリプト

先の項目の例のようなLDIFファイルをいちいち手動で作成していたら非常に大変です。実際にはLDIFファイルを自動生成するスクリプトなどを使うことになります。それほど難しくはないのでお好きなプログラミング言語で作ってみてはどうでしょうか？

参考までに、筆者が作成したPerlのスクリプトcsv2ldif.plを、本誌の付録CD-ROMに収録しましたので使ってみてください。同じファイルは「Mutt Japanese Edition - Download」([6])からダウンロードも可能です。

【表2】LDIFの作成方針

属性	作成方針
cn	漢字、英数字
sn	漢字、平仮名、英数字
givenName	漢字、平仮名、英数字
mail	英数字
o	漢字

【リスト7】dc.ldif

```
# example.org
dn: dc=example,dc=org
dc: example
objectClass: dcObject
```

【リスト8】user.ldif

```
# Taro Itsutsubashi
dn: cn=Taro Itsutsubashi,dc=example,dc=org
cn;lang-ja:: 5LqU5qmLIOWkqumDjg==
cn: Taro Itsutsubashi
sn;lang-ja:: 5LqU5qmL
sn;lang-ja:: 44GE44Gk44Gk44Gw44GX
sn;lang-en: Itsutsubashi
givenName;lang-ja:: 5aSq6YOO
givenName;lang-ja:: 44Gf44KN44GG
givenName;lang-en: Taro
mail: t5b@example.org
o:: 4peL4peL5Lya56S+
objectClass: inetOrgPerson
```

登録

●LDIFファイルの作成

まず、ベースとなるディレクトリ階層のエントリのLDIFファイルを作成します。例を示すとリスト7のようになります。

次にそのディレクトリ階層に登録する人ごとのエントリのLDIFファイルを作成します。表2のような作成方針を基にするとリスト8のようになります。

●登録

先ほど作成したLDIFファイルをldapaddを使って登録します。登録するためには書き込みの権限が必要です。とりあえずは全知全能たるroot DNでバインドしてみましょう*5。

登録したときの様子を実行例1に示します。特に何もエラーが出なければ成功です。

【実行例1】ldapaddでLDIFファイルを登録する

```
$ ldapadd -D 'cn=Manager,dc=example,dc=org' -h localhost -f dc.ldif -x -W
Enter LDAP Password:
adding new entry "dc=example,dc=org"

$ ldapadd -D 'cn=Manager,dc=example,dc=org' -h localhost -f user.ldif -x -W
Enter LDAP Password:
adding new entry "cn=Taro Itsutsubashi,dc=example,dc=org"
```

【実行例2】実行例1で登録したデータの検索

```
$ ldapsearch -h localhost -b 'dc=example,dc=org' -x '(objectClass=*)' dn
version: 2

# filter: (objectClass=*)
# requesting: dn

# emaillab, org
dn: dc=emaillab,dc=org

# Taro Itsutsubashi, emaillab, org
dn: cn=Taro Itsutsubashi,dc=emaillab,dc=org

# search result
search: 2
result: 0 Success

# numResponses: 3
# numEntries: 2
```

【リスト9】検索スクリプトの設定例

```
# set your display character encoding
my $charset = 'euc-jp';

# LDAP bind parameters
my $host = 'localhost';
my $port = '389';
my $anonymous = 1;
my $binddn = 'cn=query,dc=example,dc=org';
my $bindpassword = 'secret';
my $version = 3;
my $timeout = 10;
# LDAP search parameters
my $scope = 'sub';
my $basedn = 'dc=example,dc=org';
my $filter =
'(&(|(cn=%s)(sn=%s)(givenName=%s)(o=%s)(mail=%i))(objectClass=inetOrgPerson))';
my @ref_attrs = ['mail', 'cn', 'sn', 'givenName', 'o'];
# output parameters
my $mail_attr = 'mail';
my @name_attr = ('sn', 'givenName');
my @comment_attr = ('o');
```

＊4　ここまでやるとやりすぎのような気がします。　＊5　実運用上は登録用のエントリを作成し、そのエントリのDNに対してアクセス制御で書き込みを許可させた方がいいでしょう。

●確認

先ほど登録したデータが検索できるか確認します。実行例2では、検索して見つかったエントリのDNを表示します。

以上でサーバ側の設定は完了です。

検索スクリプト

●インストール

まず、検索スクリプトmutt_ldap_query_ja.plを、本誌の付録CD-ROM（または[6]に示すサイト）から入手し、$HOME/binなどにコピーして実行ビットを立てます。

●設定

スクリプトの中でいくつか設定を記述する必要があるので、スクリプトをエディタで開いてください。まず、設定例をリスト9に示します。設定項目の中にはLDAPの知識を必要とするものもあるので、いくつか説明を行います。

●LDAPサーバへの接続の設定

$host、$portには、LDAPサーバのホスト名とポート番号を記述します。

anonymous bind（匿名接続）でない場合は、$anonymousを「0」に設定し、$binddnと$bindpasswordを設定してください。ただしSASL認証には対応していません。

●検索の起点の設定

$basednには検索対象の起点となるディレクトリ階層のDNを指定します。

●フィルタの設定

$filterには検索条件を記述します。デフォルトのままでもよいのですが、検索対象に別の属性を加える場合はフィルタの設定を変更する必要があります。

ここでフィルタの記述方法を簡単に説明します。フィルタの基本式は次の4つです。

```
(&＜フィルタ＞＜フィルタ＞…)
(|＜フィルタ＞＜フィルタ＞…)
(!＜フィルタ＞)
(＜条件＞)
```

それぞれ、上の3つはそれぞれAND、OR、NOT演算子です。この4つの式自体が＜フィルタ＞となり、それぞれ入れ子にできます。4つ目の＜条件＞は次の形式です。

```
＜属性記述子＞=＜属性値＞
```

＜属性値＞には、「部分一致」を示すためにワイルドカードである「*」を前後に置いたり、「存在」を示すために属性値そのものを「*」にしたりできます。例えば、次のフィルタは「cnがtaroを含んでいる」あるいは「givenNameがtaroと一致する」ことを示しています。

```
(|(cn=*taro*)(givenName=taro))
```

今回使用する検索スクリプトでは、検索文字をどう扱うかを表すために、表3のような変数が設定できます。この変数を使って、先ほどの「taro」という文字列を検索する例のフィルタを書き直すと、次のようになります。

```
(|(cn=%s)(givenName=%e))
```

【表3】検索スクリプトで使える変数

変数	意味	パターン
%e	完全一致	query
%s	部分一致	*query*
%i	前部一致	query*
%f	後部一致	*query

Column　スレッドツリー

Mutt 1.3.23iからMutt 1.3.24iにかけて、スレッド表示回りが大きく変更され、スレッドツリーで用いられる記号も増えました。しかし、マニュアルにはスレッドツリーで用いられる記号についてはあまり記述がありません。記号の意味は、使っていれば何となくわかると思いますが、整理も兼ねてここで紹介します。

■スレッドの生成方法

スレッドはの生成には、通常Message-IDフィールドとReferencesフィールドが用いられます。In-Reply-Toフィールドは原則として用いられません。これらのフィールドでスレッドを構築できないものに対しては、Subjectフィールドが同じであるものをつなげて「疑似スレッド」が構築されます。

■スレッドツリーの記号

スレッドツリーに用いられる記号には、表Aのようなものがあります。どのように表示されるのかを確認しながら、それぞれについて見ていきましょう。

リストAに示したのは、fooとbarによるメッセージのやり取りのスレッド表示例です。リストAの状態からfooのメッセージだけにlimit機能で絞り込みを行なったものをリストBに示します。

「*」は、Subjectフィールドによる疑似スレッドが構築されていることを示します。リストAの6と7の関係がこれに該当します。設定変数$sort_reが設定されている場合（デフォルト）には、Subjectが$reply_regexpで設定した文字列（通常は「Re: 」）で始まる場合にのみ疑似スレッドが構築されます。$sort_reが設定されていない場合は、$reply_regexpの文字列の有無に関わらず、疑似スレッドが構築されます。ただし、設定変数$strict_threadsを設定すると、疑似スレッドの構築が行なわれなくなるため、この表示は行われなくなります。

「?」は、その位置のメッセージが欠落していることを示します。リストAの5がこれに該当します。設定変数$hide_missingは、そのメッセージの祖先のメッセージすべてが欠落しているとき、そのことを表示しないようにさせる設定です。そのため、$hide_missingの設定を解除すると、リストAの8は、メッセージの祖先が欠落しているため、表示はリストCのようになることがあります。

「=」は、メッセージが重複していることを示します。つまり「Message-IDフィールドの内容が同じであること」を表しています。リストAの3と4の関係がこれに該当します。設定変数$duplicate_threadsの設定を解除すると、この表示は行なわれなくなります。

「&」はlimit機能などによって絞り込みを行なった際、その位置のメッセージが隠されていることを示します。リストBの3には2が隠されていて、7には6が隠されています。（滝澤隆史）

【表A】スレッドツリーの記号

記号	意味
*	疑似スレッドを構築している
?メッ	セージが欠落している
=	メッセージが重複している
&	メッセージが隠されている

【リストA】fooとbarのメッセージのやり取りの画面表示例

```
1     May 20 foo      parent
2     May 20 bar      +->child
3     May 20 foo      | +->grandchild
4     May 20 foo      |   +=>grandchild (duplicate)
5     May 20 foo      +-?->grandchild (missing)
6     May 20 bar      the same subject
7     May 20 foo      +*>
8     May 20 bar      missing
```

【リストB】リストAからfooのメッセージだけを絞り込んだもの

```
1     May 20 foo      parent
3     May 20 foo      +-&->grandchild
4     May 20 foo      |   +=>grandchild (duplicate)
5     May 20 foo      +-?->grandchild (missing)
7     May 20 foo      &*>the same subject
```

【リストC】メッセージの祖先が欠落している場合

```
8     May 20 bar      ?-?-?->missing
```

Column

Muttも歩けば棒に当たる

塩崎と申します。今回は、生来のものぐさである私が、どのようにMuttを使っているかをご紹介します。ですから、目新しい使い方はないかと思いますが、みなさんが「ああ、こんなに手軽にMuttに乗り換えられるんだ」と感じて、実際にMuttを使い始めてみるきっかけになれば幸いです。

■ Muttを使い始めたきっかけ

くどいようですが私はものぐさなので、何か新しいソフトを使い始めるときは、「これまでと同じことができること」しかも「それが同じ操作体系でできるように簡単にカスタマイズできること」、さらに「何か便利になっていること」という条件を満たしていなければなりません。……という訳で、例えばシェルはcshからtcshへ、エディタはviからjvimへと移ってきています。ちなみに、OSはSunOS 4からFreeBSDへと移ってきています。

メールソフトに関しては、会社に勤め出した直後には、UCB Mailを使っていたものの、じきに周りの人に教えられてMush（Mail User's Shell）を使い始めるようになりました。Mushもかなり便利だったので10年近くお世話になりましたが、「日本語化が個人の手で行われていた」、「もともとのソースがconfigure形式ではなく#ifdefの嵐だった」ということもあって、バージョンアップもままならないのが残念でした。なので、Muttの存在には気づいてはいたのですが、当時は「S-Lang（しかも日本語版）って何？」という状態だったので、そのままほったらかしにしていました。

そのうち時代は進んで、添付ファイルやPGPが普及してくるようになりました。ところがこれまたMushは対応していなかったので、metamailなどでなんとかしのいでいました。しかしFreeBSDを4系列に上げたとき、curses周りが変更になったためかMushのコンパイルが通らなくなってしまい、さすがの私もとうとう重い腰を上げて2000年5月末にMuttに乗り換えることにしたのでした。

実際にMutt（当時のportsの1.0.1i-jp0）を使い始めてみると、初めに挙げた「新しいソフトを使い始めるときの必須条件」の3つすべてを満たしていたのです！　なぜもっと早く乗り換えなかったのかと後悔しました。

その後、新しいものを「いじめる」人が少しでも多い方がお役に立てるだろうと思い、2000年9月から開発版（本誌2001年11月号のコラムを書かれた岩下さん作のmutt-1.3.8i-ja2-beta3）を使い出し、滝澤さんがパッチを作られ、さらに岩下さんがそのportsを作られるたびに追いかけて、現在1.3.23.2i-ja.1.betaに至っています。

■ Muttの開発とのかかわり

と書くと大袈裟ですね。しかしほとんど何もしていません。パッチとも言えない「ちゃちゃ」を入れることくらいです。ただし、新しいバージョンが出ると「すかさず使う」、「気になったことは何でも（ソフト自体でも付属文書でも）メーリングリストに報告する」、ということで皆さんの努力に少しでも恩返ししたいと思っています。ちなみに、報告する際には「読む人が超能力者でなくても状況が分かる」ようにすることは心がけてます。

■ Muttの使い方

長い前置きはさておき、私の実際の使い方をご紹介します。

私は、Muttに限らず、新しいソフトが使えない環境でもできるだけ問題を起こさずに自分好みの設定が使えるように、古いソフトで対応できる部分は古いソフトの設定ファイルに書き、新しいソフトはその設定ファイルを読み込んだ上で独自の設定を加えるようにしています。

例えば、標準の`mail`コマンドでも解釈できる変数の設定などは`.mailrc`に書いておき、Mutt独自のキーの割り当てなどは`.muttrc`に書いておく、といった次第です。そのため、`alias`の設定も毎回`.mailrc`に書き込んでいるわけですが、そんなに頻繁に行う作業でもないので、未だに手作業を続けています。同じような理由で`envelope from`の設定も、Muttではなくsendmailの方（cf）で設定しています。

メールの受信はfetchmailで行い、それをprocmailに渡しています。ここでは私が読まなくても自動的に何か処理しなければならない手続きだけが書いてあり、メーリングリストの振り分けなどは行っていません。というのも、入っているメーリングリストの増減や、各メーリングリストのヘッダ仕様の変更に応じた書き換えを行うたびにメール消失に怯えなければならず、小心者の私には耐えられないからです。

メールが届くと、まずMuttを立ち上げ手動でメールを振り分けます。そのために、**リストA**のようなマクロ群を設定しています。例えば、受信メールの一覧画面でmutt-jメーリングリストのメールを見つけたら、Ctrl＋Mキーを押します。その時点でそれらしいメールにタグが付いていて、下に正しい保存先が表示されていればEnterキーを押します。うまくいかないようならMuttをいったん抜けて、マクロを設定し直して再び試せばよいので、精神衛生上も大変よろしい方法です。重要そうなSubjectのものは覗き見しつつ、この作業を振り分けの必要な分だけ繰り返します。具体的にはjキー（next-undeleted）と振り分け用のキー＋Enterキーとを繰り返す感じになります。

すると、あとは個人宛のものだけが残っているので、これを最優先で読みます。その後、時間に余裕があるときに、振り分けた各メーリングリストを

```
mail -f +mutt-j
```

のようにして読みます。実際には毎度こう打ち込むのは面倒臭いので、cshの`alias`で簡単に起動できるようにはしてあります。

メールボックス（`$mbox`）も、素の環境でも使えるように未だにmboxです。しかも、横着してためっぱなしにしていたもので、mutt-jメーリングリストは「`[Msgs:14275.4M]`」ですし、個人宛用のいわゆるメールボックスに関しては「`[Msgs:4639 33M]`」というありさまです。他のメーリングリストのものだと、未読のまま「`[Msgs:14997 48M]`」というのもあって、さすがに立ち上げるのも時間がかかり、ますます読まなくなるという悪循環に陥っています。

Mushのときは、コンパイル時にメッセージ数の制限を決めるようになっていたので、時々切り分けてはいたのですが、Muttになって便利になってしまったばかりに、ますます怠惰になっています。

利点としては、メールの分量が増えても`less`や`grep`で検索しやすいことですが、欠点としては（差分）バックアップが取りにくいことですね。

メッセージの並べ順（`$sort`）も`date-received`です。これも主に過去との互換性のためですが、それだけではありません。確かに`threads`や`date-sent`の方が便利な気もしますが、未読のメールがたまっている状態でも「とりあえず新しいのだけは消化したい」というようなとき、古いメールに関連付けられたメールや時間の狂った環境で送られたメールなどを見逃してしまうからです。

また、私は気が短くもあるので、頻繁に行う作業でいちいちyesかnoかを聞かれるものについては、できるだけどちらかに決めてしまっています。その他は、

```
set wait_key=no
set sendmail_wait=-1
```

というのも設定しています。これらの意味は、マニュアルなどを参照してみてください。

なお、キーの割り当ては次のようにvi風にしています。

```
bind index "g" first-entry
bind index "G" last-entry
bind index "h" previous-page
bind index "l" next-page
bind pager "k" previous-line
bind pager "j" next-line
bind pager "b" previous-page
bind pager "g" top
bind pager "G" bottom
```

その他、私にとって必要な変数は`$alternates`ですね。いくつかのメールアドレスへのメールをすべて同じところで読んでいるので、自分自身に返事を出さないように、アドレス群を列挙しておいています。

■ Muttへのお誘い

という訳で、私のような守旧派（？）でも、難儀な設定に苦労することなくMuttに乗り換えられたのがお分かりになると思います。ことに、他のメールソフト用の設定がある程度できている方なら、移行は本当に簡単にできると思います。

どこぞのメールソフトとは異なり、基本的な設定は安全側に倒してありますから、わずらわしいと思えばそこから少しずつ自分好みに変えていけばよいでしょう。少なくとも、このページに関心を持たれている方なら何も問題ないはずです。

それでは、mutt-jメーリングリストでお会いしましょう。　（塩崎毅彦）

【リストA】手動振り分けのためのマクロ

```
macro index \cam "T^~h X-ML-Name:\\ mutt-j$\n;s"
macro index \cax "T^~h X-ML-Name:\\ xml\n;s=xml"
macro index \cac "T^~h Sender:\\ Canna-request@\n;s"
macro index \cap "T(^~h Sender:\\ owner-pgsql-jp@sra.co.jp$|^~h Sender:\\ owner-jpug-users@|~h X-Sequence:\\ pgsql-jp)\n;s=pgsql-jp"
macro index \caf "T^~h Mailing-List:\\ contact\\ efont-help@ring.gr.jp\\;\n;s"
```

●検索結果から取得する属性

検索結果のエントリから取得する属性は、@ref_attrsに設定できます。これは「sn;lang-ja」のような言語オプションが付いたものでも設定可能です。デフォルトのままでも構いませんが、検索結果として別の属性を加えたり、言語オプションを指定したりする場合には変更します。出力として、mail、cn、sn、givenName、oが必要なときは、リスト10のように設定します。

●検索スクリプトの出力形式

Muttは、検索スクリプトの出力に対して、1行目は検索数などの応答メッセージの行があり、2行目以降は「メールアドレス＜tab＞名前＜tab＞その他の情報」という形式の行があることを期待しています。そのため、このスクリプトの設定では「# output parameters」のセクションに、$mail_attr、@name_attr、@info_attrそれぞれに対応する属性を記述します。例えば、名前をsnとgivenNameの組み合わせで出力する場合は次のようにします。

```
my @name_attr = ('sn', 'givenName');
```

その他の情報に所属組織を出力する場合は次のようにします。

```
my @info_attr = ('o');
```

●検索の実行例

設定が終わったら動作確認をしてみましょう。「五橋」という文字列で検索を行った例を実行例3に示します。このような検索結果が表示されていれば成功です。

Muttでの検索

●設定

MuttでLDAP検索を行うための設定は、Muttの設定ファイルにリスト11の1行目のように記述するだけです。

また、第2引数として検索の始点となるDNを記述することもできます。これは、検索対象のディレクトリ階層を変えたい場合などに指定します。その場合$basednの設定を上書きします。利用規模が大きい場合は、組織や用途ごとにディレクトリ階層を使い分けて使うと便利でしょう。例えば、通常の階層「dc=example,dc=org」より1つ下の特定の階層「dc=eng,dc=example,dc=org」のエントリのみを検索したい場合は、リスト11の2行目のように記述します。

●使い方

Muttからの使い方は2つあります。インデックス画面にて「Q」(query)を入力してプロンプトを出して問い合わせる文字列を入力する方法とラインエディタ上でメールアドレスや名前を一部入力した後にCtrl＋T(complete-query)を入力する方法です。詳しくは本誌2001年7月号の記事をお読みください。

【実行例3】「五橋」で検索を行った例

```
$ mutt_ldap_query_ja.pl '五橋'
LDAP query: found 1
t5b@example.org 五橋 太郎          ○○会社
```

【リスト10】@ref_attrsの設定

```
my @ref_attrs = ['mail', 'cn', 'sn', 'givenName', 'o'];
```

【リスト11】Muttでの検索設定

```
set query_command="mutt_ldap_query_ja.pl '%s'"
set query_command="mutt_ldap_query_ja.pl '%s' 'dc=eng,dc=example,dc=org'"
```

最後に

実は、筆者も初めてOpenLDAPをインストールしたとき、どうしたらよいのかと途方に暮れてしまいました。当時のバージョンは1.2系列だったと思います。日本語の扱い方が分からず、調べてみると結局は対応していない状態だったのですが、「UTF-8を生で書けば一応使えるよ」というような情報があったりしました。2.0系列になった今では、本記事で紹介した通り日本語を正式に扱えます。しかし、LDAPに関する情報、特に日本語の扱い方に関する情報は依然として少ないままです。そういう状況の中で本記事がお役に立てれば幸いです。

Column Perlのモジュールのインストール方法

Perlに標準ではインストールされていないモジュールは、「CPAN」([5])からダウンロードできます。それだけでなく、Perlのモジュールには、自動的にダウンロードしてコンパイルし、そしてインストールまで行ってくれる便利な「CPANモジュール」があるので、その使い方を紹介しましょう。

まず次のコマンドを入力します。インストール作業を行う関係上、作業はrootの権限で行う必要があります。

```
# perl -MCPAN -e shell
```

初めて実行した場合は、いくつか質問されます。質問が一通り終わったら

```
cpan>
```

のようなプロンプトが表示されるので、インストールしたいモジュールの名前をinstallコマンドの後に続けて入力します。例えば、今回記事本文で使用するPerlのスクリプトで必要なモジュールは「Jcode」、「MIME::Base64」、「Net::LDAP」です。まず「Jcode」モジュールのインストールを行います。

```
cpan> install Jcode
```

同様にして「MIME::Base64」、「Net::LDAP」もインストールしてください。インストールが完了したら、「quit」と入力して終了します。

(滝澤隆史)

Resource

[1] MUAs for Windows
http://www.emaillab.org/win-mailer/

[2] OpenLDAP
http://www.openldap.org/

[3] OpenLDAP 管理者ガイド（邦訳）
http://www.interq.or.jp/earth/inachi/openldap/admin/index-ja.html

[4] CPAN
http://www.cpan.org/

[5] LDAP Linux HOWTO（邦訳）
http://www.linux.or.jp/JF/JFdocs/LDAP-HOWTO.html

[6] Mutt Japanese Edition - Download「LDAP検索プログラム」
http://www.emaillab.org/mutt/download.html#ldap

• OpenLDAP MAN ページ（邦訳）
http://www.interq.or.jp/earth/inachi/openldap/man/index.html