Wanderlust User's Manual

Yet another message interface on Emacsen
for Wanderlust version 2.15.9

Yuuichi Teranishi
Fujikazu Okunishi
Masahiro Murata
Kenichi Okada
Kaoru Takahashi
Bun Mizuhara
Masayuki Osada
Katsumi Yamaoka
Hiroya Murata
Yoichi Nakayama

1 Introduction of Wanderlust

Wanderlust is an mail/news management system on Emacsen. It supports IMAP4rev1(RFC2060), NNTP, POP and local message files.

The main features of Wanderlust:

- Pure elisp implementation.
- Supports IMAP4rev1, NNTP, POP(POP3/APOP), MH and Maildir format.
- Unified access method to messages based on Mew-like Folder Specification.
- Mew-like Key-bind and mark handling.
- Manages unread messages.
- Interactive thread display.
- Folder Mode shows the list of subscribed folders.
- Message Cache, Disconnected Operation.
- MH-like FCC. (Fcc: %Backup and Fcc: $Backup is allowed).
- MIME compliant (by SEMI).
- Transmission of news and mail are unified by Message transmitting draft.
- Graphical list of folders (XEmacs and Emacs 21).
- View a part of message without retrieving the whole message (IMAP4).
- Server-side message look up (IMAP4). Multi-byte characters are allowed.
- Virtual Folders.
- Supports compressed folder using common archiving utilities.
- Old articles in folders are automatically removed/archived (Expiration).
- Automatic re-file.
- Template function makes it convenient to send fixed form messages.

1.1 Environment

We confirm Wanderlust works on following Emacsen:

- Mule 2.3 based on Emacs 19.34
- Emacs 20.2 or later
- XEmacs 20.4 or later
- Meadow 1.00 or later
- NTEmacs 20.4 or later
- PMMule

IMAP4 connectivity with following imapd are confirmed to work with Wanderlust:

- UW imapd 4.1–4.7, 4.7a, 4.7b, 4.7c, 2000 or later
- Cyrus imapd 1.4, 1.5.19, 1.6.22–1.6.24, 2.0.5 or later
- Courier-IMAP 1.3.2 or later
- AIR MAIL (AIRC imapd release 2.00)
- Express Mail

- Microsoft Exchange Server 5.5
- Sun Internet Mail Server 3.5, 3.5.alpha, 4.0

LDAP connectivity with following LDAPd are confirmed to work with Wanderlust:

- OpenLDAP 2.0.6 or later

2 Start up Wanderlust

The necessary procedure for starting Wanderlust is explained in steps here.

(Of course, you need a mail/news readable environment in advance)

2.1 Installing MIME modules

You must install packages named APEL, FLIM and SEMI beforehand to use Wanderlust.

For Emacs 23 or later, downloading from below URLs is highly recommended. SEMI-EPG is corresponding package to SEMI.

```
APEL:     https://github.com/wanderlust/apel
FLIM:     https://github.com/wanderlust/flim
SEMI-EPG: https://github.com/wanderlust/semi
```

You can download original APEL, FLIM and SEMI from following URLs:

```
APEL:   http://git.chise.org/elisp/dist/semi/
FLIM:   http://git.chise.org/elisp/dist/apel/
SEMI:   http://git.chise.org/elisp/dist/semi/
```

You have to install APEL, FLIM and SEMI[-EPG] in this order. Generally, '`make install`' will do the job. (In XEmacs 21, '`make install-package`'.)

Refer to the documents of each package for detailed installation procedure[1].

Recommended combination of APEL, FLIM and SEMI are following:

– APEL 10.8, FLIM 1.14.9, SEMI-EPG 1.14.7

You can also use many other FLIM/SEMI variants. Combination of the latest versions should work. For example, the following combination are confirmed to work.

– APEL 10.6, SLIM 1.14.9, SEMI 1.14.5
– APEL 10.6, CLIME 1.14.5, EMIKO 1.14.1

You have to re-install Wanderlust if you upgraded APEL, FLIM or SEMI.

2.2 Download and Extract the Package

You can download Wanderlust package from following sites:

Original site:

```
https://github.com/wanderlust/wanderlust
```

Mirrored ftp/http sites:

```
http://www.jpl.org/ftp/pub/github-snapshots/
```

Extract the obtained package to your working directory:

```
% cd ~/work
% tar zxvf wl-version.tar.gz
% cd wl-version
```

[1] If you want to use SEMI on Emacs 19.34.
http://www.jpl.org/ftp/attic/INSTALL-SEMI-ja.html (In Japanese) may help you.

2.2.1 To use SSL (Secure Socket Layer)

SSL (Secure Socket Layer) can be used for SMTP, IMAP, NNTP and POP connections in Wanderlust.

There are two ways to use SSL. One is to start SSL negotiation just after the connection establishment (generic way). The other one is to start SSL negotiation by invoking STARTTLS command in the each session.

For the formal SSL (generic SSL), Emacs 24 or later uses built-in GnuTLS if available. if not available, Emacs try to use '`tls.el`'. In this case, GnuTLS's `gnutls-cli` is needed. if neither avialable, '`ssl.el`' of '`utils`' directory and OpenSSL's `openssl`.

For the latter SSL (STARTTLS), you need '`starttls.el`'. Moreover, GnuTLS or starttls package is needed according to `starttls-use-gnutls` variable's value. If your Emacs doesn't have '`starttls.el`' neither '`starttls.el`' doesn't have the defninition of `starttls-use-gnutls`, you need starttls package.

You can download starttls package from the following site.

`ftp://opaopa.org/pub/elisp/`

2.3 Byte-compile and install

2.3.1 Installation

Edit `LISPDIR` and `EMACS` in '`Makefile`'. Set the Emacs's command name to `EMACS`. Set package installation directory to `LISPDIR`. Then, please execute following commands.

```
% make
% make install
```

Destination directory is auto-probed if you leave `LISPDIR` in '`Makefile`' as is. (That is, leave it as '`NONE`')

If you are using an Emacs variant which does not merge specified directory to `load-path` (e.g. Mule 2.3 based on Emacs 19.28), then you will see the error message:

```
Cannot open load file: mime-setup
```

In this case, either add destination directories of custom, APEL, FLIM and SEMI to environmental variable `EMACSLOADPATH`, or define `load-path` in '`WL-CFG`' in extracted directory.

If you want to handle shimbun folders or to use BBDB, add directory where emacs-w3m or BBDB is installed to `load-path`. Then necessary modules will be byte-compiled and installed. See Section 3.8 [Shimbun Folder], page 15 , See Section 13.1.2 [BBDB], page 93 .

2.3.2 'WL-CFG'

Contents of the file '`WL-CFG`' is loaded under installation if a file with that name exists in extracted directory. You can use '`WL-CFG`' to configure `load-path` to extra packages such as SEMI if needed.

If you want to specify the install directory of Wanderlust related files, then set following variables in '`WL-CFG`'

`WL_PREFIX`

A directory to install WL modules. This directory is relative directory from `LISPDIR`. WL modules include '`wl*.el`', '`wl*.elc`' files.

`ELMO_PREFIX`
: A directory to install ELMO modules. This directory is relative directory from `LISPDIR`. ELMO modules include '`elmo*.el`', '`elmo*.elc`' files.

Default value of `WL_PREFIX` and `ELMO_PREFIX` are '`wl`'.

If you want to install ELMO related files under a sub-directory such as "elmo" then add following to '`WL-CFG`':

```
(setq ELMO_PREFIX "elmo")
```

2.3.3 Install as a XEmacs package

It is possible to install Wanderlust as one of packages of XEmacs (21.0 or later). Configuration for autoload and icon-path in local '`~/.emacs`' files are no longer necessary, if you install Wanderlust as a package.

Follow the next example to install Wanderlust as an XEmacs package.

```
% vi Makefile
% make package
% make install-package
```

package directory is auto-probed, if SEMI is installed. (you can also specify it with `PACKAGEDIR` in '`Makefile`')

2.3.4 Run in place

If wl and elmo directories are defined in `load-path`, then byte-compilation and installation are not necessary to start Wanderlust. For example, if package is extracted in '`~/work`', Wanderlust can be invoked with following setting in '`~/.emacs`'.

```
(add-to-list 'load-path "~/work/wl-version/wl")
(add-to-list 'load-path "~/work/wl-version/elmo")
```

2.3.5 Manual

Manual is described in Info format. Please do following.

```
% make info
% make install-info
```

If you install Wanderlust as a XEmacs package, Info file is already installed too, so there are no need of these commands.

Manual directory is automatically detected. Of course, it can be configured by `INFODIR` in '`Makefile`'.

You can read manual at the following URL:

http://wanderlust.github.io/wl-docs/wl.html

2.4 Set up .emacs

The Wanderlust package contains two module groups.

'`ELMO (elmo-*.el)`'
: These modules show everything as folders. This is the back-end for WL.

‘WL (wl-*.el)’
: These modules controls the behavior of main body of Wanderlust. They are also the front-end for ELMO.

You can customize the behavior of Wanderlust by changing the value of environmental variables which begins with `elmo-` and `wl-`.

The minimal requirement for settings is as the following.

```
;; autoload configuration
;; (Not required if you have installed Wanderlust as XEmacs package)
(autoload 'wl "wl" "Wanderlust" t)
(autoload 'wl-other-frame "wl" "Wanderlust on new frame." t)
(autoload 'wl-draft "wl-draft" "Write draft with Wanderlust." t)

;; Directory where icons are placed.
;; Default: the peculiar value to the running version of Emacs.
;; (Not required if the default value points properly)
(setq wl-icon-directory "~/work/wl/etc")

;; SMTP server for mail posting. Default: nil
(setq wl-smtp-posting-server "your.smtp.example.com")
;; NNTP server for news posting. Default: nil
(setq wl-nntp-posting-server "your.nntp.example.com")
```

‘`~/.wl`’ is automatically loaded when Wanderlust starts up (if such a file exists). So it is convenient to gather Wanderlust specific settings in ‘`~/.wl`’. Settings for "face" must be written in ‘`~/.wl`’, because you can't write them in ‘`.emacs`’ (if you write it to ‘`.emacs`’, you'll get an error). See Section 13.2 [Highlights], page 95 .

All above described settings except autoload configuration can be written in ‘`~/.wl`’).

2.4.1 mail-user-agent

If you write following setting in your ‘`~/.emacs`’, you can start Wanderlust draft mode by typing *C-x m* (`compose-mail`). This means it enables you to run Wanderlust as a default mail composer of Emacsen.

It is effective only when your Emacs can define `mail-user-agent`. See Section “Mail Methods” in *The Emacs Editor* .

```
(autoload 'wl-user-agent-compose "wl-draft" nil t)
(if (boundp 'mail-user-agent)
    (setq mail-user-agent 'wl-user-agent))
(if (fboundp 'define-mail-user-agent)
    (define-mail-user-agent
      'wl-user-agent
      'wl-user-agent-compose
      'wl-draft-send
      'wl-draft-kill
      'mail-send-hook))
```

2.5 Folder Definition

You can skip this section because it is possible to add/edit the subscribe folders from the buffer for list of folders. See Section 4.2 [Folder Manager], page 28 .

Define the folders you want to subscribe in file '`~/.folders`'. The contents written in '`~/.folders`' become the folders which you subscribe to as it is.

Format for '`~/.folders`' is very simple. Here is an example:

```
#
# Lines begin with '#' are comment.
# Empty lines are ignored
#
# folder name  "folder nickname"
# (nicknames are not necessary)
#
%inbox  "Inbox"
+trash  "Trash"
+draft  "Drafts"
%#mh/Backup@my.imap.example.com "Sent"
# Folder Group
Emacsen{
    %#mh/spool/wl            "Wanderlust ML"
    %#mh/spool/elips         "ELIPS ML"
    %#mh/spool/apel-ja       "APEL Japanese ML"
    %#mh/spool/xemacs-beta   "XEmacs beta"
    -fj.news.reader.gnus@other.nntp.example.com "Gnus Net news"
    *-fj.editor.xemacs,-fj.editor.mule,-fj.editor.emacs "fj's Emacsen"
}
#
# If folder name ends with '/', that means an 'access group',
# all subfolders automatically included in one folder group.
#
%#mh/expire@localhost /
# All MH folders are included in one folder group.
+ /
```

Each line contains one folder you want to read. The definition of folders will be explained in detail in the next section.

The part surrounded by '*group name{*' and '`}`' will become one folder group. One folder group is treated as a directory which can be opened and closed in folder mode. It is convenient for collecting some folders and putting them in order.

Please note that '*group name{*' and '`}`' occupies one line and you have to write it that way (It is because the parser sucks).

There are two types of groups. One is like '`Emacsen`' from above example which the user chooses his favorite folders as a group.

The other one is *access group* like '`+ /`' from above example. It collects all sub-folders in the folder to make a group. (Its behavior differs by the type of the folder. For example, '`+`' followed by '`/`' makes entire MH sub-directories to one group)

This behavior is better understood if you try it and confirmed the function first. You can write and try a small folder definition, so you will know the idea of the folder function before writing the real one.

2.6 Start Wanderlust

If installation and configuration worked well, you can invoke Wanderlust by typing following command in Emacs.

```
M-x wl
```

After initialization, Folder Mode which shows the list of folders will appear. That means the folders you defined in the '`~/.folders`' are listed.

If you start Wanderlust with prefix argument like *`C-u M-x wl`*, you can skip folder checking.

2.7 Overview

Basically, you will handle messages in wanderlust while you come and go from/to each of the following buffers. Details of each ones are explained in following chapters.

'`Folder Buffer`'
: You can see the list of folders. You can select some folder and go into the summary of it. You can subscribe new folder or edit subscription list.

'`Summary Buffer`'
: You can see the list of messages in the folder. You can select message and view its contents, and reply to some message. You can delete ones or move ones to another folder.

'`Message Buffer`'
: You can see the contents of the message. You can save part to disk or open in external programs.

'`Draft Buffer`'
: You can edit message.

3 Wanderlust's folders

This chapter describes the folder types which Wanderlust is able to handle.

Wanderlust uses ELMO as it's interface, so you can use every folder types supported by ELMO.

As of version 2.15.9, 15 types of folders are predefined. These are IMAP, NNTP, LocalDir(MH), Maildir, News Spool, Archive, POP, Shimbun, Search, Multi, Filter, Pipe, File, Access and Internal folder types.

3.1 IMAP Folder

A folder to access e-mails via IMAP4rev1 protocol (RFC 2060).

Format:

'%' mailbox ['':' username ['/' authenticate-type]]['@' hostname][':' port]['!']

You can specify `login` (encoded password transmission), `cram-md5` (CRAM-MD5 authentication), `digest-md5` (DIGEST-MD5 authentication) or `clear` (or `nil`, plain password transmission) as *authenticate-type*.

default:

username -> *The value of* elmo-imap4-default-user.
Initial setting is USER *environment variable or* LOGNAME *environment variable or return value of* (user-login-name).

authenticate-type -> *The value of* elmo-imap4-default-authenticate-type.
Initial setting is "*auth*".

hostname -> *The value of* elmo-imap4-default-server.
Initial setting is "*localhost*".

port -> *The value of* elmo-imap4-default-port.
Initial setting is 143.

You can omit the *hostname* from folder names if you set `elmo-imap4-default-server` as your main IMAP server. For example, you can specify a folder as '`foo%imap@gateway`' even if you have to go through a firewall.

```
;; Example: imap4.exaple.org as main IMAP server
(setq elmo-imap4-default-server "imap4.example.org")
```

SSL (Secure Socket Layer) connection will be used if a folder name ends with '`!`'. If a folder name ends with '`!!`', STARTTLS connection will be established.

If the value of `elmo-imap4-default-stream-type` is `ssl`, SSL will be the default connection, i.e. you can omit '`!`'. If it is is `starttls`, STARTTLS will be the default connection. To use normal connection in these cases, add '`!direct`' at the end of folder name.

```
;; Example: Use SSL connection
(setq elmo-imap4-default-stream-type 'ssl)
```

If you specify `login`, `cram-md5` or `digest-md5` as authentication method, the password is sent in encoded form. But, if your server is unable to receive an encoded password, authentication will fall back to `clear` (that is, sending password in raw format) after confirmation to user. If `elmo-imap4-force-login` is non-nil, authentication will fall back to `clear` without confirmation (default value is `nil`).

```
;; Example: password in raw format
(setq elmo-imap4-default-authenticate-type 'clear)
```

Example:

```
%inbox     -> IMAP mailbox "inbox"
%#mh/inbox -> IMAP mailbox "#mh/inbox"

%inbox:hoge -> IMAP mailbox "inbox" of user "hoge".
%inbox:hoge/clear@server1
            -> server1's IMAP mailbox "inbox"
               of user "hoge", with plain password authentication
               ('clear).
```

3.1.1 International mailbox names (Modified UTF7)

You can use international mailbox names in *mailbox* part, if you are using Emacs with UTF-7 support and `elmo-imap4-use-modified-utf7` is set to non-nil value (default value is `t` on Emacs 23 or later and `nil` on other Emacsen).

Emacs 23 or later doesn't need additional package. Other Emacsen needs Mule-UCS package to use UTF-7. Mule-UCS works on following Emacsen.

- Emacs 20.3 or later
- XEmacs 21.2.37 or later

You can obtain Mule-UCS package (snapshot of 2004) from following URL.

http://www.jpl.org/ftp/pub/tmp/

3.2 NNTP Folder

A folder to access USENET news via NNTP protocol (RFC 977). One newsgroup is treated as a folder.

Format:

'-' newsgroup [':' username]['@' hostname][':' port]['!']

default:

```
hostname  -> The value of elmo-nntp-default-server.
             Initial setting is 'localhost'.
username  -> The value of elmo-nntp-default-user.
             Initial setting is nil.
port      -> The value of elmo-nntp-default-port.
             Initial setting is 119.
```

AUTHINFO is used as authentication method if the *username* is non-nil. SSL connection will be used if a folder name ends with '`!`'. If a folder name ends with '`!!`', STARTTLS connection will be established.

If the value of `elmo-nntp-default-stream-type` is `ssl`, SSL will be the default connection, i.e. you can omit '`!`'. If it is is `starttls`, STARTTLS will be the default connection. To use normal connection in these cases, add '`!direct`' at the end of folder name.

Example:

```
-fj.rec.tv            -> Newsgroup 'fj.rec.tv'.
-fj.rec.tv@newsserver -> Newsgroup 'fj.rec.tv' on 'newsserver'.
```

3.3 MH Folder

A folder to access MH format mail (1 file is 1 mail).

Format:

'+' *directory-name*

Normally, *directory-name* is an relative path to the variable `elmo-localdir-folder-path` (default is '`~/Mail`'), but if it starts with '`/`' or '`~`', then it is treated as an absolute path (this is also true for drive-letters).

Message number is used for the name of the message file.

Example:

```
+inbox          -> '~/Mail/inbox'
+from/teranisi -> '~/Mail/from/teranisi'
+~/test         -> '~/test'
```

3.4 Maildir Folder

A folder to access Maildir format (1 file is 1 mail).

Format:

'.' [*directory-name*]

Normally, *directory-name* is a relative path to the variable `elmo-maildir-folder-path` (default is '`~/Maildir`'), but if it starts with '`/`' or '`~`', then it is treated as an absolute path (this is also true for drive-letters).

Maildir contains '`cur`', '`new`' and '`tmp`' subdirectories. Messages are contained in the '`cur`' directory. All message files in the '`new`' directory are moved to '`cur`' directory when you access the folder. All message files contained in the '`tmp`' directory and not accessed for 36 hours are deleted.

This behavior conforms to the http://cr.yp.to/proto/maildir.html.

Example:

```
.               -> '~/Maildir'
.inbox          -> '~/Maildir/inbox'
.from/teranisi -> '~/Maildir/from/teranisi'
.~/test         -> '~/test'
```

3.5 News Spool Folder

This folder handles locally saved news articles which are proposed by Mew/IM. You can also read articles directly from a spool-file which is retrieved by an utility like `gnspool`.

Format:

'=' *directory-name*

directory-name is a sub-directory to the directory defined by variable `elmo-localnews-folder-path` (default is '`~/News`') You can use '`.`' as directory delimiter as well as '`/`'.

Example:

```
=fj/os/os2          -> '~/News/fj/os/os2'
=fj.os.bsd.freebsd -> '~/News/fj/os/bsd/freebsd'
```

3.6 Archive Folder

This method can handle archive files, which are compressed by utilities such as Info-ZIP or LHA, as one folder.

Format:

'$' path-name ['/' archiver-type '/' prefix]

path-name is the relative path from `elmo-archive-folder-path` (initial setting is '`~/Mail`'). If *path-name* begins with '/' or '~' or 'drive-letter of DOS', *path-name* is treated as absolute path. ange-ftp format is also permitted under the environment of ange-ftp, efs.

The actual file name of the archive folder is `elmo-archive-basename` (Initial setting is '`elmo-archive`') under the *path-name*. If a file named *path-name* exists, it is treated as folder. The suffix is automatically decided for *archiver-type*.

If *archiver-type* is omitted, `elmo-archive-default-type` (Initial setting is `zip`) is referred.

prefix specifies the internal directory structure of the archive. For example, if the ML server is fml, '`msend.tar.gz`' has a structure like '`spool/1`', so you have to specify '`spool`' as *prefix*.

Example:

```
$teranisi         -> '~/Mail/teranisi/elmo-archive.zip'
$bsd/freebsd;lha  -> '~/Mail/bsd/freebsd/elmo-archive.lzh'
$/foo@server:~/bar;zoo     -> '~/bar/elmo-archive.zoo' on ftp server
$d:/msend.tar.gz;tgz;spool -> 'd:/msend.tar.gz'
$ml;zip/        -> Access group consists of archive folders
                   under '~/Mail/ml'
```

3.6.1 Supported Archives

By default, following archives are supported.

```
LHA, Info-ZIP/UNZIP, ZOO, RAR  ;; full-access
GNU TAR('tgz, 'tar)            ;; read-only
```

If your archiver can include multiple files in one archive, you have a possibility use it as an archiver of Wanderlust (ARJ/UNARJ, ARC is one of the candidate. TAR is supported read-only because it cannot delete file in the archive (`mv`)).

`gzip`, `bzip`, `bzip2` cannot be used as an archiver of Wanderlust because they cannot include multiple files. Archivers that cannot extract files to standard output are also not supported.

3.6.2 OS specific information about archiver.

Behaviors of the following archivers are confirmed by further experiences. ('`*`' mark means recommended archiver).

```
[OS/2]  Warp4.0J(w/o VoiceType)+Fx00505/emx0.9c(fix04)/PMMule,EmacsPM
         LHA  OS/2 version Rel.2.06b    Feb 18, 1998
        *UnZip 5.32 of 3 November 1997, by Info-ZIP.
        *Zip 2.2 (November 3rd 1997).
```

```
            Zoo archiver, zoo 2.1 $Date: 91/07/09 02:10:34 $
            GNU tar version 1.10 - AK 2.58 (DBCS/SJIS) 981216(homy) version
            gzip 1.2.4 (18 Aug 93) + bzip2 patch(by Iida-san)

[UN|X]  FreeBSD 2.2.7-RELEASE, Linux 2.0.30, Solaris2.6, HP-UX 9.07
            LHa for UNIX  V 1.14c
            UnZip 5.32 of 3 November 1997
            Zip 2.2 (November 3rd 1997)
            GNU tar 1.12 (1.11.x is no good)
            gzip 1.2.4 (18 Aug 93)

[Win32] Win.98/Meadow
            Lha32 version 1.28
            Zip 2.2
            UnZip 5.40
            GNU tar 1.11.8 + 1.5(WIN32)
            GZIP 1.2.4
            RAR 2.06
```

* Caution about LHA

If you are an OS/2 user, Peter Fitzsimmons's LH/2 is not supported. Hiramatsu version of LHA is only supported. In Win32, LHa32 is only supported (DOS version is no good).

* Caution about GNU tar

You have to take care about GNU tar's version because many version has problem on deleting file from archive.

Please test '`--delete`' '`-f`' options work. Otherwise, your archive will be destroyed. No problem is reported on above versions of GNU tar.

3.6.3 TIPS

For comfortable migration, usage of `wl-summary-archive` (see Section 9.5 [Archive], page 76) or Expire (see Section 9.1 [Expire], page 70) is recommended. To treat archive folders created by expiration, you must set non-nil value to `elmo-archive-treat-file`.

On the OS/2, there is a great difference between Mule2.3(19.28) and Emacs20.2 in processing speed. For comfortable use, Emacs20 is recommended. (If re-search's performance is the problem, 19.3x or later may be okay.)

If many files are included in one archive, it takes long time to access the archive folder because archiver starting overhead is increased (especially LHA). 150-200 messages in one archive is recommended.

Of course, following is possible `:-)` (meanings of these variables are described later.)

```
(setq wl-fcc "$backup")
(setq wl-trash-folder "$trash;lha")
```

3.6.4 Variables About Archive Folder

`elmo-archive-default-type`

The initial setting is `zip`. Set archiver type by symbol.

`elmo-archive-`*`type`*`-method-alist`
: Define archiver *type*'s methods. (*type* is '`lha`', '`zip`', '`zoo`', '`tgz`' etc) Each element of the alist is following.

```
(action . (exec-name options))   ;; external program and its option.
(action . function)              ;; function
```

Currently available actions are following.

```
'ls, 'cat ('cat-headers)        ;; Minimal setting(read-only)
'mv ('mv-pipe), 'rm ('rm-pipe)  ;; full-access (with above)
'cp ('cp-pipe)                  ;;
```

In above actions, actions enclosed with braces are optional (They are used for better performance).

`elmo-archive-suffix-alist`
: An alist of archiver-type (symbol) and suffix.

`elmo-archive-file-regexp-alist`
: An alist of a regexp to get file number from list output of archiver and archiver-type (symbol).

`elmo-archive-method-list`
: A list of elmo-archive-*type*-method-alist (*type* is a symbol of archiver-type).

`elmo-archive-lha-dos-compatible`
: The initial setting is `t` on OS/2 and Win32. If non-nil, LHA is DOS (Mr. Yoshizaki original) compatible.

`elmo-archive-cmdstr-max-length`
: The initial setting is 8000.

Max length of command line argument for external archiver program. Emacs does not have a limit of command line byte length, but some OS (e.x OS/2) have. It depends on the OS. Archive folder is affected by this limit because it calls external archiver program directly (not called via shell). For example, you cannot delete messages if archiver program must receive larger bytes of arguments to delete. OS/2 have a command line argument limit of 8190 bytes, so we defined default as 8000 with some margin.

However, you don't have an influence of command line argument limit if the archiver has 'actions' to receive target file information from standard input (`rm-pipe`, `mv-pipe`, `cat-headers` action).

3.7 POP Folder

A folder to access e-mails via POP3 protocol (RFC 1939).

Format:

```
'&' [username]['/' authenticate-type][':' numbering-method]['@' hostname][':' port]['!
```

You can specify '`user`' (plain password transmission) or '`apop`' (APOP authentication) as *authenticate-type*.

You can specify '`uidl`' (use UIDL command for message numbering) or '`list`' (use LIST command for message numbering) as '`numbering-method`'.

default:

```
username   -> The value of elmo-pop3-default-user.
              Initial setting is USER environment variable or
            LOGNAME environment variable or return value of
            (user-login-name).
authenticate-type -> The value of elmo-pop3-default-authenticate-type.
              Initial setting is 'user'.
numbering-method -> Follow the value of elmo-pop3-default-use-uidl.
              If t, use UIDL for numbering. Initial setting is t.
hostname   -> The value of elmo-pop3-default-server.
              Initial setting is 'localhost'.
port       -> The value of elmo-pop3-default-port.
              Initial setting is 110.
```

Example:

```
&hoge@localhost     -> access localhost as user 'hoge'.
&hoge@popserver:109 -> access the server "popserver" on port 109
                         as user 'hoge'.
```

To use apop as an *authenticate-type*, 'md5.el' is needed (XEmacs doesn't need 'md5.el'). 'md5.el' is included in 'utils/sasl/lisp/' or Emacs/W3 package (http://www.cs.indiana.edu/elisp/w3/docs.html) or LCD archive (GPL2).

If the last character of the folder name is '!', Wanderlust connects to the POP server via SSL (Secure Socket Layer). If a folder name ends with '!!', STARTTLS connection will be established.

If the value of elmo-pop3-default-stream-type is ssl, SSL will be the default connection, i.e. you can omit '!'. If it is is starttls, STARTTLS will be the default connection. To use normal connection in these cases, add '!direct' at the end of folder name.

3.8 Shimbun Folder

A folder for watching "shimbun" (means "newspaper" in Japanese), news site and mailing list archives on WWW by using emacs-w3m (http://emacs-w3m.namazu.org/).

You should possess w3m and emacs-w3m to use this.

Format:

```
'@' module-name '.' folder-name
```

Admissible values of *module-name* and *folder-name* are described in 'README.shimbun.ja' distributed with emacs-w3m.

Example:

```
@airs.wl  -> archive of wanderlust ML (using module 'sb-airs.el')
@asahi/   -> access group of all folders in module 'sb-asahi.el'
```

3.8.1 Variables About Shimbun Folder

elmo-shimbun-update-overview-folder-list

The initial setting is all. Specify a set of folders to update overview when messages are fetched. Specify all to update overview in all shimbun folders. You can specify a list of regular expressions of shimbun folder names to restrict affected folders.

Example:

```
(setq elmo-shimbun-update-overview-folder-list
      '("^@airs\\." "^@namazu\\."))
```

Update summary view automatically after fetching.

3.9 Search Folder

A folder to access messages found by an external program with some condition.

Format:

'[' search condition ']' [search target ['!' search engine]]

The format of the *search condition* and *search target* depend on the *search engine*.

3.9.1 Supported search engines

Supported search engines are following ones. Default search engine can be assigned by `elmo-search-default-engine`.

3.9.2 namazu

The messages registered in the namazu-index is found by using namazu (http://www.namazu.org/).

search condition is a query of namazu. Please refer to the document of the attached to namazu for details.

search target is a namazu-index used for search. The directory with the index or the alias that explain in the following can be specified. Default value of the path of namazu index can be assigned by `elmo-search-namazu-default-index-path`.

Example:

```
[wanderlust]           -> search messages matched with
                          "wanderlust" from the default index
[semi flim]~/Mail/semi -> search "semi flim" from the index
                          in the directory "~/Mail/semi"
```

3.9.2.1 Enter space to separate keywords

If you want to use space in folder entry, *C-q SPC* will help you.

3.9.2.2 Alias name for index

You can define an alias name for index.

Example:

```
(setq elmo-search-namazu-index-alias-alist
      '(("cache" . "~/.elmo/cache")
        ("docs" . "~/documents")))
```

Above definition defines two index aliases. You can specify

```
[wanderlust]cache
```

to execute a namazu search with keyword '`wanderlust`' using a index in the directory '`~/.elmo/cache`'.

3.9.2.3 Multiple indices

You can specify a list for `elmo-search-namazu-default-index-path` and `elmo-search-namazu-index-alias-alist`. When list is specified, all index contained in the list is used as the namazu indices.

Example:

```
(setq elmo-search-namazu-index-alias-alist
      '(("all" . ("~/.elmo/cache" "~/documents"))
        ("cache" . "~/.elmo/cache")))
```

Using above alias setting, you can specify

```
[wanderlust]all
```

to execute a namazu search with keyword '`wanderlust`' using indices in the directory '`~/.elmo/cache`' and '`~/documents`'.

3.9.3 grep

The files that exists in the directory specified with the *search target* are found by using grep.

search condition is a regular expression of grep. The directory as *search target* cannot be omitted.

Example:

```
[wanderlust]~/Mail/inbox!grep
    -> search messages matched with "wanderlust"
       from the directory "~/Mail/inbox"

["[sr]emi"]~/Mail/semi!grep
    -> If ']' is included in regular expression,
       search condition should be enclosed with '"'.
```

3.9.4 mu

Search mail using mu (http://www.djcbsoftware.nl/code/mu/).

Examples (from the mu cheatsheet)

```
[Helsinki]
    -> messages about Helsinki (in message body, subject, sender, ...)

[to:Jack subject:jellyfish tumbleweed]
    -> messages to Jack with subject jellyfish containing the word tumbleweed

[size:2k..2m date:20091201..20093112 flag:attach from:bill]
    -> messages between 2 kilobytes and a 2Mb, written in December 2009 with an attach

[subject:soc* flag:unread]
    -> unread messages about things starting with 'soc' (soccer, society, socrates, ..

[mime:image/*]
    -> messages with images as attachment

['foo bar']
    -> search for messages with the phrase "foo bar"
```

3.9.5 notmuch

Search mail using notmuch (http://notmuchmail.org/).

Examples (from the notmuch manual)

```
[term1]
    -> messages that contain 'term1'

[-term1]
    -> messages that DO NOT contain 'term1'

[term1 term2]
    -> messages containing term1 and term2

['foo bar']
    -> search for messages with the phrase "foo bar"
```

3.10 Multi Folder

A folder to access virtual folder which collects messages from multiple folders.

Format:

'' folder-1 [',' folder-2] ... [',' folder-N]*

After '*' character, specify multiple folders you want to collect separated by ',' like '*folder-1,folder-2,...,folder-N*'.

Example:

```
*-fj.editor.xemacs,-fj.editor.mule,-fj.editor.emacs
-> -fj.editor.xemacs, -fj.editor.mule and -fj.editor.emacs are
   treated as one folder.

*+inbox,-fj.rec.tv,%inbox
-> +inbox, -fj.rec.tv and %inbox are treated as one folder.
```

3.11 Filter Folder

A folder to access virtual folder which collects all messages that satisfy a condition.

Format:

'/' condition '/' target-folder

In the *condition* part, you can specify following.

1. Partial filter: '`first:`*number*', '`last:`*number*'

   first: *number* messages are picked from top of folder. last: *number* messages are picked from bottom of folder.

   Example:

   ```
   /last:10/-fj.os.linux -> Latest 10 messages from -fj.os.linux are picked.
   /first:20/%inbox      -> First 20 messages from %inbox are picked.
   ```

2. Date filter: '`since:`*date*', '`before:`*date*'

   since: only messages arrived since *date* are picked (*date* is included). before: only messages arrived before *date* are picked (*date* is not included).

   You can specify following as *date*.

   ```
   yesterday ->  a day before today.
   lastweek  ->  same day of last week.
   lastmonth ->  same day of last month.
   lastyear  ->  same day of last year.
   numberdaysago -> number days ago. (e.x. '3daysago')
   day-month-year -> specify date directly (ex. 1-Nov-1998)
   ```

   Example:

   ```
   /since:3daysago/+inbox -> messages arrived since 3 days ago in +inbox
                             are picked.
   /before:yesterday/+inbox -> messages arrived before yesterday in +inbox
                               are picked.
   ```

3. Field filter: '*field:value*'

   All messages that have *field* and its value is *value* are picked. *field* and *value* are case insensitive.

   Example:

   ```
   /from:teranisi/+inbox -> In +inbox, messages which have From: field
                            and its value includes "teranisi" string are picked.
   /body:foo/%inbox      -> In %inbox, messages which have "foo" text
                            are picked.
   ```

4. Flag filter: '`flag:`*flag-name*'

   Pick up messages with flag specified by *flag-name*.

You can specify following flag names:

```
unread    -> unread
important -> important
answered  -> replied
forwarded -> forwarded
digest    -> unread or important
any       -> unread or replied or forwarded or global-flag.
```

You can also use flags which you have set as 'global-flag'. global-flag is a flag which has arbitrary name. You can put global-flag on messages by invoking `wl-summary-set-flags` (Key `F`). By default, '`important`' flag is prepared. You can view messages with global-flag by visiting the subfolder of '`'flag`' folder.

See Section 3.13 [Internal Folder], page 22 .

Example:

```
/flag:digest/%inbox     -> a folder consist of unread or important
                           message in %inbox.
/flag:wl/+ML/Wanderlust -> a folder consist of messages with global flag
                           wl in +ML/Wanderlust.
```

5. Compound condition

   A condition starting with '`!`' indicates a negation. If you combine conditions by character '`|`', it is considered as OR condition. '`&`' is considered as AND condition, likewise. Condition can be grouped by parentheses ('`(`', and '`)`').

   '`/tocc:xxxx/`' is an abbreviation of '`/to:xxxx|cc:xxxx/`'. '`/!tocc:xxxx/`' is an abbreviation of '`/!to:xxxx&!cc:xxxx/`'.

   Example:

```
/from:teranisi&!to:teranisi/+inbox
                        -> In +inbox, messages are picked if the message's
                           From: field includes "teranisi" and
                           To: field doesn't include "teranisi".

/tocc:"Yuuichi Teranishi"/+inbox
                        -> In +inbox, messages are picked if the
                           message's To: field or Cc: field includes
                           "Yuuichi Teranishi".

/(from:yt|from:teranisi)&subject:report/+inbox
                        -> In +inbox, messages are picked if the message's
                           From: field includes "yt" or "teranisi", and
                          Subject includes "report".
```

Tip for string description:

Space character, '`"`', '`/`','`)`','`|`' and '`&`' should be enclosed with '`"`' in *value* string. ('`"`' should be escaped with '`\`' in it). You can enclose the string with '`"`' even it does not contain these characters.

Advanced example:

```
*%inbox,/from:teranisi/%inbox@server
       -> Messages in %inbox or
          message is in the %inbox@server folder and it's From field
          includes "teranisi" are collected.

/last:100//to:teranisi/*+inbox,%inbox
       -> Latest 100 messages which is in the +inbox or %inbox folder
          and To: field matches "teranisi".

/from:hogehoge//last:20//tocc:teranisi/%#mh/inbox@localhost
       -> Pick messages which have From: field and it includes "hogehoge"
          from latest 20 messages in the %#mh/inbox@localhost
          and To: or Cc: field includes "teranisi".
```

3.12 Pipe Folder

In the pipe folder, messages are automatically transferred from the source folder to destination folder.

Format:

'|' *source-folder* '|' *destination-folder*

When you access the pipe folder, messages are automatically transferred from *source-folder* to *destination-folder*. It is convenient if you want to download messages to local disk via POP. For example, if you specify following

```
|&username@popserver|+inbox
```

and access it, Wanderlust downloads messages from '`&username@popserver`' to '`+inbox`' automatically.

On the other hand, if you put '`|:`' instead of second '`|`', then messages are copied to the destination folder (not deleted from source-folder). At the next time you access that folder, copies new messages only.

'|' *source-folder* '|:' *destination-folder*

If you want to copy messages from POP server and view them, specify the folder as follows:

```
|&username@popserver|:+inbox
```

where messages will be kept on the server.

Example:

```
|%inbox|%myinbox   -> Download %inbox to %myinbox.
|*&user@popserver1,&user@popserver2|+inbox
         -> Download from &user@popserver1 and &user@popserver2 to +inbox.
|-gnu.emacs.sources|:+sources
         -> Copy messages from -gnu.emacs.sources to +sources.
```

After messages are moved, a hook `elmo-pipe-drained-hook` is called.

3.13 Internal folder

A folder to access internal messages of Wanderlust.

Format:

```
'flag' ['/' global-flag]
'sendlog'
'cache/00' - 'cache/1F'
```

A folder named 'flag' is a special virtual folder which collects messages which have *global-flag*.

There is 'important' flag defined as *global-flag* by default. You can review important messages at once after you put important marks on the messages in the different folders. If *global-flag* is omitted, it is treated as 'important' flag is specified.

In addition, in summary mode, to be described later, you can freely define global flags and put them on messages. See Section 5.1 [Usage of Summary Mode], page 32 .

In this folder, if you delete message, *global-flag* put on the message is removed. If you append messages to this folder, the message will have *global-flag*.

A folder named 'sendlog' is a virtual folder which collects cached messages which are recoded on '~/.elmo/sendlog'. It might be useful when you forgot to add cc for yourself. To use this, you should set `wl-draft-use-cache` to non-nil so that sent messages are cached.

You can access cached messages fetched via network by accessing folders named 'cache/00' - 'cache/1F'. 00 - 1F are the name of the subdirectories of the cache directory ('~/.elmo/cache').

3.14 File folder

File Folder gives the view for local file system. The one File Folder corresponds to the one directory.

Format:

```
'file:' Path-of-the-directory
```

Example:

```
file:~/work     -> '~/work'
file:/etc       -> '/etc'
```

3.15 Access folder

A folder to access virtual folder which collects messages from a root folder and subfolders of one. The add and remove of the subfolder is automatically reflected.

Format:

```
'access:' root-folder
```

Example:

```
access:%INBOX -> All subfolders of IMAP mailbox "inbox".
access:'cache -> All of 'cache folder
```

3.16 RSS Folder

An RSS folder presents the messages contained in an RSS or Atom feed:

```
rss:https://github.com/wanderlust/wanderlust/commits/master.atom
```

This folder type attempts to automatically handle all known versions of RSS and Atom. Should you need more configurability, please use a Shimbun folder (see Section 3.8 [Shimbun Folder], page 15) instead.

Since this folder doesn't do any persistent caching, messages will disappear as soon as they expire from the feed. Should you want persistent messages, combine this with a pipe folder:

```
|rss:http://lwn.net/http://lwn.net/headlines/newrss|+lwn
```

You may also use an RSS, Atom or OPML feed as an access group, in which case related feeds will appear as subfolders.

4 Folder mode

After you start Wanderlust, folder mode is appeared firstly. It contains folder list you subscribed. You can select and edit folders in this mode.

4.1 Selecting Folder

4.1.1 Usage (TIPS)

4.1.1.1 Check new, unread number

Folder mode looks like this. (In XEmacs, it looks much nicer `;-)`)

```
[-]Desktop:14186/35580/67263
   Inbox:3/10/10
   Trash:2/7/10
   Drafts:0/0/3
   Sent:0/9/348
   [-]Emacsen:0/34/4837
     Wanderlust ML:0/0/558
     ELIPS ML:0/0/626
     tm:0/0/821
     XEmacs Beta:0/29/255
     Mew:0/0/998
     Mule-Win32:0/0/1491
     fj's Emacsen:0/5/88
```

Each line means:

folder-name:new-number/unread-number/all-number

`s` key on the folder line updates these numbers. It changes its color if it has many new messages.

The whole folder mode is a folder group named '`Desktop`'. Folder group open/close by return key. A operation to a folder group is treated as operations on the children folders. For example, when you type `s` on '`[-]Emacsen`', six children folders update its unread number status.

4.1.1.2 Select Folder

To enter summary mode of the folder, type return (or space) key on the folder line. If the variable `wl-stay-folder-window` has non-nil value, summary window appears on the right of the folder mode window.

4.1.2 Key bindings

Folder mode's key binding (related to selecting folders) is following.

`SPC`
`RET` Enter to the summary mode of the folder at the current cursor point. With prefix argument, enter the sticky summary. If the cursor is on the top of folder group line, the folder group is opened or closed. When the cursor is on the access

group and this command is called with prefix argument, folder children list is updated to the newest one. (Children list is updated recursively if the access folder has hierarchical structure.) (`wl-folder-jump-to-current-entity`)

M-RET Folder children list of the access group at the current cursor point is updated to the newest one. (Children list is updated recursively if the access folder has hierarchical structure.) (`wl-folder-update-recursive-current-entity`)

w Create a new draft message. (`wl-draft`)

W If the current cursor point is on the NNTP folder, create a new draft message which already has '`Newsgroups:`' field. If the current cursor point is on the folder for mailing list (refile destination), create a new draft message which already has '`To:`' field with guessed mailing list address (If `wl-subscribed-mailing-list` is valid list). (`wl-folder-write-current-folder`)

C-c C-o Move to the draft buffer if available. If multiple draft buffer exists, moved to one after another. If prefix argument is specified, load draft folder's message to the draft buffer and jump to it. (`wl-jump-to-draft-buffer`)

s Update new and unread number information of the folder at the current cursor point. (`wl-folder-check-current-entity`)

S Update summary information of the folder at the current cursor point. (`wl-folder-sync-current-entity`)

r s Update new and unread number information of the folders in the currently selected region. (`wl-folder-check-region`)

r S Update summary information of the folders in the currently selected region. (`wl-folder-sync-region`)

Z Sync up address book status with '`~/.addresses`''s content. (`wl-status-update`)

P Jump cursor to the folder which have unread messages on the upward from current cursor point. (`wl-folder-prev-unread`)

N Jump cursor to the folder which have unread messages on the downward from current cursor point. (`wl-folder-next-unread`)

p Move cursor to the folder on the previous line. (`wl-folder-prev-entity`)

n Move cursor to the folder on the next line. (`wl-folder-next-entity`)

J Jump to the folder specified by the user input. (`wl-folder-jump-folder`)

I Prefetch new messages of the folder at the current cursor point by `wl-summary-incorporate`. If the cursor is on the folder group, it is executed recursively. (`wl-folder-prefetch-current-entity`)

c Mark all unread messages of the folder at the current cursor point as read. If the cursor is on the folder group, it is executed recursively. (`wl-folder-mark-as-read-all-current-entity`)

f Enter summary mode of the first unread folder. (`wl-folder-goto-first-unread-folder`)

E Empty trash. (`wl-folder-empty-trash`)

F Flush queue. (`wl-folder-flush-queue`)

V Move to the virtual folder (filter folder) with the condition specified. (`wl-folder-virtual`)

? Search the folders with the condition specified. (`wl-folder-pick`)

o All unread folder is opened. (`wl-folder-open-all-unread-folder`)

x Execute marks in summary buffers. See Section 5.5 [Sticky Summary], page 36 . (`wl-execute-temp-marks`)

/ Folder group is opened/closed. (`wl-folder-open-close`)

[All folder group is opened. (`wl-folder-open-all`)

] All folder group is closed. (`wl-folder-close-all`)

q Quit Wanderlust. (`wl-exit`)

z Suspend Wanderlust. (`wl-folder-suspend`)

M-s Save current folder status. (`wl-save`)

M-t Toggle Wanderlust's offline/online status. (`wl-toggle-plugged`)

C-t Start Wanderlust's plug-status manager. (`wl-plugged-change`)

4.1.3 Customize variables

`wl-folders-file`
: The initial setting is '`~/.folders`'. Subscribed folders are described (saved) in this file.

`wl-folder-info-save`
: The initial setting is `t`. If non-nil, unread information is saved and used in the next Wanderlust session.

`wl-stay-folder-window`
: The initial setting is `nil`. If non-nil, summary window is appeared on the right side of the folder buffer.

`wl-folder-window-width`
: The initial setting is 20. Folder mode's window width when `wl-stay-folder-window` is non-nil.

`wl-folder-use-frame`
: The initial setting is `nil`. If non-nil, use new frame for the folder window.

`wl-folder-many-unsync-threshold`
: The initial setting is 70. If the number of unread messages is more than this value, folder color is changed.

`wl-highlight-folder-by-numbers`
: This option controls how to highlight each line in the folder buffer. The default value is `t`, highlighting with various colors based on the message numbers. If it

is `nil`, highlighting with various colors based on the folder status. In addition, if it is a number (e.g. 1), highlighting will be done based on both the message numbers and the folder status.

`wl-folder-desktop-name`
: The initial setting is '`Desktop`'. The name of top folder group.

`wl-folder-petname-alist`
: The initial setting is `nil`. An alist of folder's realname and its nickname.

`wl-folder-access-subscribe-alist`
: The initial setting is `nil`.

Control automatic subscribing and unsubscribing of the children list of access groups.

Each element is:

```
(regexp-of-access-folder . (subscribe-flag regexp-of-folders ...))
```

If *subscribe-flag* is non-nil, folders which have name matched to *regexp-of-folders* are displayed. Otherwise, hidden. However, already unsubscribed folder is not displayed even when the *subscribe-flag* is non-nil. Multiple *regexp-of-folders* can be specified.

Example:

```
'(("^-fj$" . (t   "^-fj\\.\\(comp\\|editor\\|mail\\)"
                  "^-fj\\.\\(net\\|news\\|os\\|rec\\)"))
  ("^-$" . (t   "^-\\(fj\\|tnn\\|japan\\|gnu\\|comp\\)"))
  ("^\\+ml$" . (nil "^\\+ml$" "^\\+ml/tmp")))
```

`wl-folder-hierarchy-access-folders`
: A list of regular expressions for access groups which creates children folder list hierarchically.

For example, if you specify `wl-folder-hierarchy-access-folders` like following,

```
(setq wl-folder-hierarchy-access-folders
      '("^-[^\\.]*$" "^-comp.unix$" "^-comp.unix.bsd$"))
```

you obtain the access group hierarchy as follows.

```
[-]-:912/912/3011
  [-]-fj:674/674/1314
    -fj.comp.announce:0/0/2
    -fj.comp.dev.cdrom:0/0/0
    ...
  [+]-japan:238/238/1688
  [-]-comp:0/0/4
    [-]-comp.unix:0/0/0
      -comp.unix.admin:0/0/0
      -comp.unix.dos-under-unix:0/0/0
      -comp.unix.programmer:0/0/0
      [-]-comp.unix.bsd:0/0/23
        -comp.unix.bsd.freebsd.announce:0/0/0
  ...
```

If you opened '`-`' in this example, only the direct children is created ('`-fj`', '`-japan`', '`-tnn`', ...). second hierarchy ('`-fj.comp.announce`', ..., '`-comp.unix`', ...) is not created until the children access group is opened.

4.2 Editing Folders

As described before, subscribed folder list is saved in '`~/.folders`' file. But you don't have to edit '`~/.folders`' directly. You can append, delete, edit folders from folder mode.

4.2.1 Usage (Tips)

4.2.1.1 Append Folder

m a appends new folder to your folder list. If you enter non-existent folder, it will ask you to create a new one. *m g* appends new folder group. To append new folder to this group, firstly open it, then execute append command in the next line.

4.2.1.2 Edit Folder

You can cut folder by *C-k*, paste by *C-y*. Thus, you can change folder position as if you were editing a normal file.

4.2.1.3 Create Multi Folder.

1. Type *m q* to clear `wl-fldmgr-cut-entity-list`.
2. Cut folder by *C-k* or copy folder by *M-c*.
3. Type *m m*, then you can create multi folder.

4.2.1.4 Delete Nickname, Filter

You can delete nickname or filter by putting "" (*NULL*) from the minibuffer while appending.

4.2.1.5 Append Folder to Empty Group

To append new folder to the empty folder group (after you create folder group by typing *m g*), firstly open it, then execute append command in the next line. If it is closed, folder is appended on the same level with the folder group above. It is difficult to explain by words so try it. In other words, appended position depends on the open/close status of the upper one.

4.2.1.6 Charset of the Folders File.

`wl-mime-charset` is used for saving `wl-folders-file`.

4.2.1.7 Create Filter

m f adds filter to the folder at the current cursor point. To create new filter folder and leave the current folder unchanged, copy it *M-c*, make filter *m f* and paste it *C-y*. Multiple filter can be specified while appending filter. If you put "" (*NULL*), filter is deleted.

4.2.1.8 Sort Folders

Sorting of the folders is executed by the function specified by `wl-fldmgr-sort-function`. The initial setting is `wl-fldmgr-sort-standard`, which sorts alphabetically. Sorting affects only on the current folder group. It does not affect on the child groups.

4.2.1.9 Hiding Folders in the Access Group

Usually, access group displays all children folders, but you can set some folders hidden. Following operations are only available on access group.

Command `wl-fldmgr-unsubscribe` (*u*) toggles the visibility (subscribe/unsubscribe) of the folder at current cursor point. Against this, `wl-fldmgr-unsubscribe-region` (*U*) hides folders in the specified region.

Note that `wl-fldmgr-unsubscribe-region` does not toggle while `wl-fldmgr-unsubscribe` toggles. These two commands accept prefix argument and if the argument has positive number, the unsubscribe it. If the prefix argument has negative value, folder becomes visible and if zero, folder visibility is toggled.

The other commands, `wl-fldmgr-subscribe` and `wl-fldmgr-subscribe-region` are also prepared (not binded to the key).

Moreover, if `wl-fldmgr-cut` or `wl-fldmgr-cut-region` is executed in the access group, they have a same effect with `wl-fldmgr-unsubscribe` and `wl-fldmgr-unsubscribe-region`. The difference is that cut commands deletes folders from the current buffer.

4.2.1.10 Operations in the Access Group

You can insert and delete folders in the access group like usual folder group. But insert and delete commands can be only available for the children folders of the access group and they only sets the subscribe status. In other words, insertion of the folder means subscribing, deletion means unsubscribing.[1]

To update the access group when children folders are inserted or deleted by other way (other than Wanderlust), open the access group by typing *C-u RET*. See Section 4.1 [Selecting Folder], page 24 .

The order of children folders of access group is saved after insertion/deletion/sorting. If you set `wl-force-fetch-folders` to non-nil or open access group by typing *C-u RET*, disappeared folders are deleted and newly created folders are inserted on the top of the access group.

4.2.2 Key bindings

Key bindings on the folder mode related to folder editing are shown below. All bindings starts with *m*, and primary commands are binded to one stroke key binding.

m a
: Add specified folder to your folder list . If you enter non-existent folder, create it after confirmation. (`wl-fldmgr-add`)

+

m g
: Create a folder group. (`wl-fldmgr-make-group`)

m A
: Create an access group. (`wl-fldmgr-make-access-group`)

m d
: Delete folder itself and msgdb. If the folder itself cannot be deleted like NNTP folder, only msgdb is deleted. (`wl-fldmgr-delete`)

[1] In the current implementation, it is faster to delete region than to unsubscribe region.

R
m R Change the name of folder or folder group. msgdb's path is also changed. (`wl-fldmgr-rename`)

***
m m Create a multi folders in the cutlist (cut, copied folders). (`wl-fldmgr-make-multi`)

|
m f Create a filter folder. (Put a filter on the folder). (`wl-fldmgr-make-filter`)

M-c
m c Copy folder (it is not available on folder group). (`wl-fldmgr-copy`)

M-w
m W Copy folders in the specified region. (`wl-fldmgr-copy-region`)

C-k
m k Cut folder. Folder itself is not deleted. (`wl-fldmgr-cut`)

C-w
m C-w Cut folders in the specified region. (`wl-fldmgr-cut-region`)

C-y
m y Paste folders that are copied or cut (folders in the cut-list). (`wl-fldmgr-yank`)

m p Put nickname on the folder. (`wl-fldmgr-set-petname`)

m q Clear the cut-list. (cut, copied folder information is cleared, you cannot paste after this) (`wl-fldmgr-clear-cut-entity-list`)

m s Sort folders in the current folder group. (`wl-fldmgr-sort`)

m C-s Save current folder view to the '`wl-folders-file`'. (`wl-fldmgr-save`)

[Following commands are only available on the access groups]

u
m u Set the visibility of folder (subscribe/unsubscribe). (`wl-fldmgr-unsubscribe`)

U
r u Set the visibility of the folders (subscribe/unsubscribe) in the specified region. (`wl-fldmgr-unsubscribe-region`)

l
m l List folders that are currently available. (`wl-fldmgr-access-display-normal`)

L
m L List all folders regardless of the subscription status. (`wl-fldmgr-access-display-all`)

4.2.3 Customize variables

`wl-interactive-save-folders`
The initial setting is `t`. If non-nil and folder view is modified, confirm saving it before Wanderlust or Emacs exits. If `nil`, save without confirmation.

`wl-fldmgr-make-backup`
: The initial setting is `t`. If non-nil, '`~/.folders.bak`' is created before saving the folder status.

`wl-fldmgr-sort-function`
: The initial setting is `wl-fldmgr-sort-standard`. A function to sort folders. By default function, folders are sorted alphabetically and folder group is put on top (when `wl-fldmgr-sort-group-first` is non-nil).

`wl-fldmgr-sort-group-first`
: The initial setting is `t`. If non-nil, `wl-fldmgr-sort-standard` precedes folder group. If `nil`, it does not care whether it is folder group or not.

`wl-folder-check-async`
: The initial setting is `t`. If non-nil, check folder's unread status asynchronously. It boosts newsgroup checking.

`wl-folder-check-fast`
: The initial setting is `nil`. If non-nil, it does not update folder status while checking.

`wl-folder-notify-deleted`
: The initial setting is `nil`. If non-nil, negative value is displayed when the message is deleted. If `sync`, folder is synchronized when the message is deleted. If `nil`, message deletion is ignored.

`wl-fldmgr-add-complete-with-current-folder-list`
: The initial setting is `nil`. Non-nil means call `elmo-folder-list-subfolders` and get completion candidate for `wl-fldmgr-add`.

4.2.4 Miscellanea

Following is a note for folder editing.

1. cut or copy stacks the folder in the `wl-fldmgr-cut-entity-list`. paste(yank) command pastes the folders on one cut or copy command (If copy command is executed by region, folders in the region are pasted by one paste command)
2. You cannot cut '`Desktop`' group. Also, you cannot paste folders at the outside of the '`Desktop`'.
3. You cannot copy folder group.
4. Operations on the access group are only available for the folders in the same access group.
5. You cannot create a folder which has same name with the folders already exist.
6. You cannot insert folders which have same name in one group. You can insert them in the different groups. You cannot put same nickname to the different folders.

5 Summary Mode

After you select the folder via folder mode, you enter to the summary mode.

5.1 Usage (Tips)

5.1.1 Summary Content

In the summary mode, messages are displayed like following.

```
377  09/16(Wed)11:57 [+1: Takuro Kitame  ] Bug?
381  09/17(Thu)00:16 [+3: Fujikazu Okuni ] elmo-lha.el -- LHA interface
384  09/17(Thu)01:32 [+1: Yuuichi Terani ] wl-0.6.2
389 N09/18(Fri)01:07 [+2: Yuuichi Terani ] wl-0.6.3
```

Each line displays:

Message number, Temporal mark, Persistent mark, Date, Sender, Subject

If you want to know how to change the format for this, please refer the section format of Summary lines. See Section 5.6 [Summary View], page 37 .

Message number is the message's unique number in the folder. In the NNTP folder, it is article number, in the IMAP folder, it is UID and in the MH folder, it is the filename of the message.

Temporal mark and *Persistent mark* are described later.

Date is displayed like '*Month/Day(Week Day)Hour:Minute*'. Default setting displays week day in Japanese, but if you want to display it in English, set the value of `wl-summary-weekday-name-lang` as '`en`'.

Sender's indentation corresponds to the depth of the thread. Sender name is displayed as nickname if it is defined in the address book. Set `wl-use-petname` as `nil`, if you want to quit displaying with nickname.

If number is printed at the head of *Sender* part like '`+2`', that means the message have 2 follow messages.

Subject is the '`Subject:`' header field of the message. If the message have same '`Subject:`' with the parent message, it is not displayed. Some mailing list puts its sequence number in the '`Subject:`' field, but it is ignored. `wl-summary-no-subject-message` is displayed when the message has empty subject field.

5.1.2 Temporary Marks

There are seven temporary marks, '`*`', '`d`', '`D`', '`o`', '`O`', '`i`' and '`~`'. Temporary marks indicates message operations.

'`*`' Target mark. You can execute a command on the all messages that have '`*`' mark, with the key bindings which begins with *m*.

'`d`' The mark to dispose. You can put '`d`' by typing *d* key.

'`D`' The mark to force delete. You can put '`D`' by typing *D* key.

'`o`' The mark to refile. After you type *o* key, prompt appears to input refile destination. Your answer is printed in the summary line.

‘O’ The mark to refile. You can put this mark by typing *O* key. The difference between this mark and refile mark is, this mark does not delete the message while latter does.

‘i’ The mark to prefetch reserved. You can put this mark by typing *i* key.

‘~’ The mark to resend reserved. After you type ~ key, prompt appears to input address to resend. Your answer is printed in the summary line.

x key executes action for temporary marks, respectively.

5.1.3 Persistent Marks

There are ten persistent marks, ‘!’, ‘N’, ‘n’, ‘U’, ‘u’, ‘A’, ‘a’, ‘F’, ‘f’ and ‘$’.

The persistent mark indicates the message’s status and it is saved. Each persistent mark indicates:

‘N’ It is new message.

‘n’ It is new message. It differs from ‘N’ that message with ‘n’ is already cached.

‘U’ It is unread message.

‘u’ It is unread message. It differs from ‘U’ that message with ‘u’ is already cached.

‘!’ It is message already read. It differs from message without mark that message with ‘!’ is not cached yet.

‘A’ It is already replied message.

‘a’ It is already replied message. It differs from ‘A’ that message with ‘a’ is already cached.

‘F’ It is already forwarded message.

‘f’ It is already forwarded message. It differs from ‘F’ that message with ‘f’ is already cached.

‘$’ It is a message with some global flag. It is convenient to put this mark on the messages to remember (If you want to remember to write a reply for the message, for example) because this mark remains after you exited Emacs. Messages with the ‘$’ mark can be reviewed in the ‘’flag’ folder even the message itself is deleted in the actual folder. You can put global flag by typing *$* or *F* key.

‘None’ If the message is read and cached (or local message),there are no persistent mark.

‘N’, ‘U’, ‘!’, ‘A’, ‘F’ indicates that the message have no cache. Messages with the marks other than these, you can read them in the offline status even they are in the IMAP folder or netnews folder.

Among messages with persistent marks, ones with marks specified by `wl-summary-expire-reserve-marks` are excluded from the expiration (as a function of wanderlust) explained later. See Chapter 9 [Expire and Archive], page 70 .

5.1.4 How To Read

Basically, you can read messages only typing space key again and again.

To update summary status to the newest status (synchronize), type *s* key.

You can jump to next unread message by typing *N* key, and *n* key moves cursor to the next message. Enter message buffer by typing *j* key. To operate multipart, you have to enter to the message buffer. See Chapter 6 [Message], page 52 .

5.1.5 Pack the Message Numbers

You can pack the message numbers in Summary by *M-x wl-summary-pack-number*. Note that only MH Folder, News Spool Folder and Maildir Folder are supported folder types.

5.2 Thread Operations

For example, the following line indicates one thread (a context of a topic).

```
384  09/17(Thu)01:32 [+1: Teranishi       ] wl-0.6.2
```

If you type `/` on this line, the thread is opened and it changes the appearance like following.

```
384  09/17(Thu)01:32 [ Teranishi          ] wl-0.6.2
388  09/17(Thu)22:34 +-[ Murata san         ]
```

(Message 388 is the replied message to the message 384.) If you type `/` key once again, the thread is closed. With prefix argument, `/` opens all children threads.

If you type *[*, opens all threads in summary. *]* closes all threads.

Commands with the key binding that begins with *t* executes commands on the messages in the thread. See Section 5.8 [Key Bindings of Summary], page 39 .

5.2.1 reconstruct thread by hand

You can reconstruct the thread manually. In Summary, *M-w* (`wl-summary-save-current-message`) at the corresponding message, and *C-y* (`wl-summary-yank-saved-message`) at the new parent message then you have the reconstructed thread.

5.3 Cache

5.3.1 Cache File

The messages which have to access via network (e.x. IMAP, NNTP folder) are cached as a local file so as to save network traffic or to enable off-line operation. The cache file is saved under the directory '`~/.elmo/cache`'. To expire cache, type *M-x elmo-cache-expire-by-size*. The command deletes cache files to the specified size by the order of last accessed time.

5.3.2 Buffer Cache and Prefetching

The messages that are read are kept in the cache buffer so as to make the behavior fast when you are to read the message again. It is called 'buffer cache'. The number of cache buffer is specified by `wl-message-buffer-cache-size`.

There are message prefetching mechanism in the Wanderlust that prefetches next message while you are reading.

You can control the message prefetching mechanism by these two variables.

`wl-message-buffer-prefetch-folder-type-list`
: The initial setting is `'(imap4 nntp)`. If it is a list of folder types, it specifies the folder types in which message prefetching is enabled. In initial setting, messages are prefetch only in the nntp and imap4 folders. In this case, multi folder that contains localdir and imap4 prefetches only imap4 messages. This variable precedes the value of `wl-message-buffer-prefetch-folder-list`. To prefetch messages in all folder types, specify `t`.

`wl-message-buffer-prefetch-folder-list`
: The initial setting is `nil`. A list of regexp of folders to enable message prefetching.

`wl-message-buffer-prefetch-depth`
: The initial setting is 1. The number of messages for automatical prefetch.

`wl-message-buffer-prefetch-idle-time`
: The initial setting is 1 (in seconds). The period of automatical prefetch.

`wl-message-buffer-prefetch-threshold`
: The initial setting is 30000 (bytes). If prefetching message has larger size than this value, Wanderlust does not prefetch automatically. If `wl-message-buffer-prefetch-threshold` is `nil`, the message is not checked for the size.

`wl-auto-prefetch-first`
: The initial setting is `nil`. If non-nil, first message is automatically prefetched to the buffer when you enter the folder.

5.4 Auto Refile

You can refile messages automatically, by typing *C-o* (`wl-summary-auto-refile`). It decides destination of refile by the content of the message header information (information in the msgdb).

By default, '`From:`', '`Subject:`', '`To:`' and '`Cc:`' is available. If you want to decide destination by other header fields, set the variable `elmo-msgdb-extra-fields` like following.

```
(setq elmo-msgdb-extra-fields
      '("x-ml-name"
        "reply-to"
        "sender"
        "mailing-list"
        "newsgroups"))
```

By this setting, Wanderlust saves extra fields in the msgdb. You have to type *s all* to get extra fields for the messages that are already exists in the summary.

Then, specify the refile rule. The refile target folder of auto refiling is decided by the value of `wl-refile-rule-alist`. `wl-refile-rule-alist` is a list of a rule:

```
(field (regexp . target)
       (regexp . target)
       ...)
```

Each rule means 'if *field* value matches *regexp*, then refile to *target* folder'. The rule matched first is applied.

field is a string of field name. You can specify a list of field name string, too. In this case, if one of these fields is matched, then the rule is fired (i.e. OR condition).

regexp is a regular expression for field value. *target* is a target folder string. You can specify a rule at *target* part, too. In this case, If the field value of the rule and the current rule is matched, then the current rule is fired (i.e. AND condition).

You can refer matched substring of *regexp* to specify *target* part. To refer substring, use following keys:

'`\&`' means substitute original matched text.

'`\`*N*' means substitute what matched the *N*th '`\(...\)`'. (*N* is a number.)

Following is an example of `wl-refile-rule-alist`.

```
(setq wl-refile-rule-alist
      '(("x-ml-name"
         ("^Wanderlust" . "+wl")
         ("^Elisp" . "+elisp"))
        (("To" "Cc")
         ("\\([a-z]+\\)@gohome\\.org" . "+\\1"))
        ("From"
         ("me@gohome\\.org" . ("To" ("you@gohome\\.org" .
                                     "+from-me-to-you"))))))
```

After these settings, refile marks are automatically put on the condition matched messages by typing *C-o* (`wl-summary-auto-refile`).

Messages which have `wl-summary-auto-refile-skip-marks` is skipped auto refiling. By default, '`N`', '`U`' and '`!`' is specified, so the messages with these persistent marks are not automatically refiled. It means Wanderlust does not execute auto refile on unread messages by the default setting. To execute auto refile on all messages, set following.

```
(setq wl-summary-auto-refile-skip-marks nil)
```

5.5 Sticky Summary

The buffer of the 'sticky summary' does not killed by typing *q*.

By entering the summary by typing *Shift RET* in Folder mode or *G* in some suummary sticky summary buffer is created. Also typing *M-s* (`wl-summary-stick`) on the normal summary makes current one sticky.

The buffer name of the sticky summary becomes like '`Summary:`*folder-name*'. You can visit the sticky summary at any time by *C-x b* (`switch-to-buffer`), or you can go round summary buffers by *C-c C-n* (`wl-summary-previous-buffer`) and *C-c C-p* (`wl-summary-next-buffer`) in summary mode.

In sticky summary, the summary buffer is preserved after *g* or *q*. To delete sticky summary, type *C-u q* to exit or move to another summary by *C-u g*. Other operations in the sticky summary are same as normal summary.

`wl-summary-always-sticky-folder-list` specifies the folders that are automatically stuck.

5.6 Format of summary lines

You can alter the format of lines in Summary mode.

Summary line format is specified by `wl-summary-line-format`. You can use control strings defined by `wl-summary-line-format-spec-alist`.

An example follows.

```
;; number temporary-mark persistent-mark date branch
;; [ (number-of-children) sender ] subject
(setq wl-summary-line-format "%n%T%P%M/%D(%W) %t%[%17(%c %f%) %] %s")
```

Where the number set the column number of the field. If negative value, the column is filled from right. If the number begins with '0', '0' is used for filling columns instead of ' '.

Example:

```
%5n   -> '1    '
%-05n -> '00001'
```

Major control strings defined by `wl-summary-line-format-spec-alist` are displayed in the following list.

```
%n  message number
%T  temporary mark
%P  persistent mark
%Y  year
%M  month
%D  day
%W  day of week
%h  hour
%m  minute
%t  branch of the thread
%[  [ (< for re-connected child)
%]  ] (> for re-connected child)
%f  sender
%s  subject
%S  size
%c  +number-of-children: (display only for opened thread)
%C  [+number-of-children] (display only for opened thread)
%#  mailing list information ('(' ML-name [ ' ' ML-number ] ')')
%l  number in the mailing list
%@ '@' only if the first MIME part is multipart/mixed
%~  ' ' only if previous column is not empty
```

The temporary mark ('%T') and persistent mark ('%P') must appear at the constant column. For example, if you specify '%T' or '%P' after the '%t', which changes its length by thread position, marks are not treated correctly.

If the format string is enclosed by '%number(' and '%)', the column of the enclosed region is justified to the 'number'. Multiple level '%number(' parenthesis can be defined. It is useful to justify the column of the multiple control strings. For example, in the above `wl-summary-line-format`,

```
%17(%c %f%)
```

means “Adjust number-of-children and sender string to the 17 column”.

You can specify the format by each folders with `wl-folder-summary-line-format-alist`. Please set regular expression for folder names and summary line format as the following example.

```
(setq wl-folder-summary-line-format-alist
      '(("^%" . "%T%P%M/%D(%W)%h:%m %t%[%17(%c %f%) %] %s")
        ("^+" . "%n%T%P%M/%D %[ %17f %] %t%C%s")))
```

5.6.1 on the format for sender name

The format string ‘`%f`’ displays the return value of the function specified by `wl-summary-from-function`. If you use the function `wl-summary-default-from` (default), it displays sender name ordinarily, while displays the recipient names if the folder name matches with `wl-summary-showto-folder-regexp` and the sender is yourself. If the value of `wl-use-petname` is Non-nil, it uses petname to display.

For example, to display recipient names for the message in ‘`+backup`’ if its sender is yourself, set up as follows.

```
(setq wl-summary-showto-folder-regexp "^\\+backup$")
```

5.7 Temporary marks and their effect

You can define temporary marks and corresponding procedure by `wl-summary-mark-action-list`. Initially, refile (‘`o`’), copy (‘`O`’), dispose (‘`d`’), delete (‘`D`’), prefetch (‘`i`’) and resend (‘`~`’) are defined.

Each element of `wl-summary-mark-action-list` consists of

```
(‘MARK’ ‘SYMBOL’
 ‘ARGUMENT-FUNCTION’ ‘SET-MARK-FUNCTION’ ‘EXEC-FUNCTION’
 ‘FACE’ ‘DOC-STRING’)
```

‘`MARK`’ is a temporary mark string, and ‘`SYMBOL`’ is the name of the action to be defined. ‘`ARGUMENT-FUNCTION`’ is a function to generate argument to be given to ‘`SET-MARK-FUNCTION`’, which will be described next, and it takes arguments:

```
(‘ACTION’ ‘NUMBER’)
```

Where ‘`ACTION`’ equals to ‘`SYMBOL`’, and ‘`NUMBER`’ is message number. ‘`SET-MARK-FUNCTION`’ is a function to be called when the mark is put. It takes arguments:

```
(‘NUMBER’ ‘MARK’ ‘DATA’)
```

Where ‘`NUMBER`’ is target message number, ‘`MARK`’ is a temporary mark string, and ‘`DATA`’ is given by ‘`ARGUMENT-FUNCTION`’.

‘`EXEC-FUNCTION`’ is a function to be called when the action is executed. Its argument is a list of ‘`MARK-INFO`’. Here ‘`MARK-INFO`’ means a list consists of

```
(‘NUMBER’ ‘MARK’ ‘DATA’)
```

‘`FACE`’ is a face to be used for highlighting.

5.8 Key bindings

Key bindings of the summary mode are shown below.

`SPC` Proceed reading a message at the current cursor point. (`wl-summary-read`)

`.` Redisplay a message at the current cursor point with default display type. If this command is called with prefix argument, reload and redisplay message regardless of the message cache. If this command is called with twice multiples *C-u* as *C-u C-u* `.`, reload and redisplay message with current display type regardless of the message cache. (`wl-summary-redisplay`)

`<` Display the top message in the folder. (`wl-summary-display-top`)

`>` Display the bottom message in the folder. (`wl-summary-display-bottom`)

`BS`
`DEL` Display the previous page of the message at the current cursor point. (`wl-summary-prev-page`)

`RET` Display the next line of the message at the current cursor point. Display the message at the current cursor point if it is not displayed yet. (`wl-summary-next-line-content`) If prefix argument is specified, message is scrolled up by one line. (`wl-summary-prev-line-content`) If prefix argument is numeric, cursor is jumped to the message with specified number.

`-`
M-RET Display the previous line of the message at the current cursor point. Display the message at the current cursor point if it is not displayed yet. (`wl-summary-prev-line-content`)

/ Toggle open or close the thread at the current cursor point. With prefix argument, open all children threads. (`wl-thread-open-close`)

[Open all threads. (`wl-thread-open-all`)

] Close all threads. (`wl-thread-close-all`)

g Go to other folder. (`wl-summary-goto-folder`)

c Mark all messages in the folder as read. (`wl-summary-mark-as-read-all`)

a Prepare a draft for reply the message at the current cursor point. (`wl-summary-reply`)

A Prepare a draft for reply the message at the current cursor point. (`wl-summary-reply-with-citation`)

C If the message at current cursor point is your own netnews article, cancel it. (`wl-summary-cancel-message`)

E Prepare a draft for re-editing the message at current cursor point. If the message at current cursor point is your own netnews article, a draft for 'supersedes message' for the message is prepared. (`wl-summary-reedit`)

M-E If the message at current cursor point is a bounced message, a draft for re-sending original message is prepared. (`wl-summary-resend-bounced-mail`)

f A draft for forwarding the message at current cursor point is prepared. (`wl-summary-forward`)

$ Put '`important`' flag on the message at current cursor point. If already flagged as '`important`', remove the flag. If it is called with prefix argument, ask global flag to put similarly to *F*. (`wl-summary-mark-as-important`)

F Put arbitrary global flag entered in the minibuffer. If you use Emacs 21 or later, you can specify multiple flags separated by '`,`' simultaneously. If it is called with prefix argument, remove existent global flags. (`wl-summary-set-flags`)

y
e Save the message at current cursor point. (`wl-summary-save`)

n Move cursor to the next message. If message is marked with a temporal mark in `wl-summary-skip-mark-list`, cursor is not moved to it. In the offline mode, cursor is not moved to the messages which are not cached yet. (`wl-summary-next`)

p Move cursor to the previous message. If message is marked with a temporal mark in `wl-summary-skip-mark-list`, cursor is not moved to it. In the offline mode, cursor is not moved to the messages which are not cached yet. (`wl-summary-prev`)

N Move cursor to the downward message which is unread or marked as '`$`'. In the offline mode, cursor is not moved to the messages which are not cached yet. If there are messages which have target mark '`*`' in the summary, cursor is moved to the downward message which have a target mark. This behavior is changed according to the value of `wl-summary-move-order`. (`wl-summary-down`)

P Move cursor to the upward message which is unread or marked as '`$`'. In the offline mode, cursor is not moved to the messages which are not cached yet. If there are messages which have target mark '`*`' in the summary, cursor is moved to the downward message which have a target mark. This behavior is changed according to the value of `wl-summary-move-order`. (`wl-summary-up`)

w Prepare a new draft. (`wl-summary-write`)

W Prepare a new draft. If the current folder is NNTP folder, '`Newsgroups:`' field is completed. If the current folder is mailing list folder (refile destination), guess '`To:`' field and completed (If `wl-subscribed-mailing-list` is valid list) (`wl-summary-write-current-folder`)

H Toggle display type between all and partial header fields and redisplay the message at current cursor point. If this command is called with prefix argument, reload and redisplay message regardless of the message cache. If this command is called with twice multiples *C-u* as *C-u C-u H*, set default display type of summary by current display type of header fields. (`wl-summary-toggle-all-header`)

M Toggle display type for MIME analysis and redisplay the message at current cursor point. A change is performed in the order set as `wl-summary-display-mime-mode-list`. If this command is called with numeric prefix argument, it switch directly as follows.

```
1: Enable MIME analysis.
2: Enable MIME analysis only for header fields.
3: Disable MIME analysis.
```

If this command is called with twice multiples *C-u* as *C-u C-u M*, set default display type of summary by current display type of MIME analysis. (`wl-summary-toggle-mime`)

C-c C-f Toggle header body narrowing of the message at current cursor point. (`wl-summary-toggle-header-narrowing`)

B If the message at current cursor point has encapsulates multiple messages using MIME, de-capsulate and extract them on the current folder. If it is invoked in non-writable folder or it is called with prefix argument, it asks the destination folder. (`wl-summary-burst`)

@ Append/change/delete the message's sender information to the address book '`~/.addresses`' interactively. If this command is called with prefix argument, arbitrary address can be edited. (`wl-summary-edit-petname`)

Z Sync up address book status with '`~/.addresses`''s content. (`wl-status-update`)

| Pipe current message's content to the external process. (`wl-summary-pipe-message`)

Print out current message's content. It uses `ps-print` module in Emacs 20.x. If you don't use color printer, you might want to set `wl-ps-print-buffer-function` to `ps-print-buffer`.

```
(setq wl-ps-print-buffer-function 'ps-print-buffer)
```

(`wl-summary-print-message`)

q Exit current folder. (`wl-summary-exit`)

j Jump cursor to the currently displayed message's window. (`wl-summary-jump-to-current-message`)

J Jump cursor to the other message. (`wl-summary-jump-to-msg`)

I Update summary status and prefetch all messages which have marks included in the `wl-summary-incorporate-marks`. (`wl-summary-incorporate`)

M-j Jump cursor to the message which have specified '`Message-Id:`'. If `elmo-use-database` is non-nil, other folder is also searched. (`wl-summary-jump-to-msg-by-message-id`)

^ Jump to parent message. (`wl-summary-jump-to-parent-message`)

! Mark as unread the message at current cursor point. (`wl-summary-mark-as-unread`)

s Synchronize summary view after prompting the update range. You can specify one of the follows.

all	Discard present msgdb and retrieve all informations. Do not retrieve killed messages.
all-entirely	Discard present msgdb and retrieve all informations. Retrieve killed messages, too.
update	Update the difference between informations in present msgdb and in current folder instance. Do not retrieve killed messages.
update-entirely	Update the difference between informations in present msgdb and in current folder instance. Retrieve killed messages, too.
rescan	Redisplay summary by rescanning present msgdb.
rescan-noscore	Redisplay summary by rescanning present msgdb. Display messages killed by score, too.
rescan-thread	Redisplay summary by rescanning present msgdb. Reconstruct thread, too.
cache-status	Sync the all marks with the real status of cache.
mark	Update marks.
no-sync	Do nothing.
first:NUM	Move to the filter folder(partial filter).
last:NUM	Move to the filter folder(partial filter).

(`wl-summary-sync`)

S Sort summary order. You can sort by '`date`', '`from`', '`number`', '`subject`', '`size`' and '`list-info`'. With prefix argument, sort summary lines into descending order. (`wl-summary-sort`)

T Toggle the threading. The state will be preserved after exiting Wanderlust. You can alter default state for newly created summary by `wl-summary-default-view` or `wl-summary-default-view-alist`. Threading status is displayed on the modeline. '`{S}`' means threading is off (Sequence) and '`{T}`' means threading is on (Thread). (`wl-summary-toggle-thread`)

l Toggle displaying of folder window. (`wl-summary-toggle-disp-folder`)

v Toggle displaying of message window. (`wl-summary-toggle-disp-msg`)

V Move to the virtual folder (filter folder) with the condition specified. If called with prefix argument and current folder is virtual, exit it. (`wl-summary-virtual`)

TAB Jump to the message which is displayed last. (`wl-summary-goto-last-displayed-msg`)

? Put '`*`' mark on the messages that satisfies the specified condition. If messages already have '`*`' mark, new '`*`' marks are overridden. If prefix argument is specified, current '`*`' marks are removed and new '`*`' marks are appended.

(`wl-summary-pick`)

R Mark as read the message at the current cursor point. (`wl-summary-mark-as-read`)

`x` Execute action for all temporary marks in the summary buffer. (`wl-summary-exec`)

`*` Put target mark on the message at the current cursor point. (`wl-summary-target-mark-line`) See Section 5.7 [Mark and Action], page 38 .

`o` Put refile mark on the message at the current cursor point. (`wl-summary-refile`) See Section 5.7 [Mark and Action], page 38 .

`C-o` Execute auto refile. (`wl-summary-auto-refile`)

`O` Put copy mark on the message at the current cursor point. (`wl-summary-copy`) See Section 5.7 [Mark and Action], page 38 .

`M-o` Put refile mark on the message at the current cursor point with the destination previously specified. (`wl-summary-refile-prev-destination`)

`d` Put disposal mark on the message at the current cursor point. The result of disposal is controlled by `wl-dispose-folder-alist`, refiled to `wl-trash-folder` by default. (`wl-summary-dispose`) See Section 5.7 [Mark and Action], page 38 .

`D` Put force deletion mark on the message at the current cursor point. (`wl-summary-delete`) See Section 5.7 [Mark and Action], page 38 .

`i` Put prefetch reservation mark on the message at the current cursor point. (`wl-summary-prefetch`) See Section 5.7 [Mark and Action], page 38 .

`~` Put resend reservation mark on the message at the current cursor point. (`wl-summary-resend`) See Section 5.7 [Mark and Action], page 38 .

`u` Unmark the temporal mark on the message at the current cursor point. (`wl-summary-unmark`)

`U` Unmark all the temporal marks. (`wl-summary-unmark-all`)

`r R` Mark as read messages in the specified region. (`wl-summary-mark-as-read-region`)

`r $` Put '`important`' flag on messages in the specified region. If already flagged as '`important`', remove the flag. (`wl-summary-mark-as-important-region`)

`r F` Put arbitrary global flag entered in the minibuffer on messages in specified region. (`wl-summary-set-flags-region`)

`r !` Mark as unread messages in the specified region. (`wl-summary-mark-as-unread-region`)

`r x` Execute action for each temporary marks on the messages in the specified region. (`wl-summary-exec-region`)

`r *` Put target mark on the messages in the specified region. (`wl-summary-target-mark-region`) See Section 5.7 [Mark and Action], page 38 .

`r o` Put refile mark on the messages in the specified region. (`wl-summary-refile-region`) See Section 5.7 [Mark and Action], page 38 .

r O Put copy mark on the messages in the specified region. (`wl-summary-copy-region`) See Section 5.7 [Mark and Action], page 38 .

r d Put disposal mark on the messages in the specified region. (`wl-summary-dispose-region`) See Section 5.7 [Mark and Action], page 38 .

r D Put force deletion mark on the messages in the specified region. (`wl-summary-delete-region`) See Section 5.7 [Mark and Action], page 38 .

r i Put prefetch reservation mark on messages in the specified region. (`wl-summary-prefetch-region`) See Section 5.7 [Mark and Action], page 38 .

r u Delete temporal mark on the messages in the specified region. (`wl-summary-unmark-region`)

r y Save messages in the specified region. (`wl-summary-save-region`)

t R Mark as read messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-mark-as-read`)

t $ Put '`important`' flag on the messages which are the descendant of the current thread. If already flagged as '`important`', remove the flag. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-mark-as-important`)

t F Put arbitrary global flag entered in the minibuffer on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-set-flags`)

t ! Mark as unread messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-mark-as-unread`)

t x Execute action for temporary marks on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-exec`)

*t ** Put target mark '`*`' on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-target-mark`) See Section 5.7 [Mark and Action], page 38 .

t o Put refile mark on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-refile`) See Section 5.7 [Mark and Action], page 38 .

t O Put copy mark on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-copy`) See Section 5.7 [Mark and Action], page 38 .

t d Put disposal mark on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-dispose`) See Section 5.7 [Mark and Action], page 38 .

t D Put force deletion mark on the messages which are the descendant of the current thread. (`wl-thread-delete`) See Section 5.7 [Mark and Action], page 38 .

`t i` Put prefetch reservation mark on messages which are the descendant of the current thread. (`wl-thread-prefetch`) See Section 5.7 [Mark and Action], page 38 .

`t u` Unmark temporal mark on the messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-unmark`)

`t y` Save messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree. (`wl-thread-save`)

`m R` Mark as read all messages which have target mark '`*`'. (`wl-summary-target-mark-mark-as-read`)

`m $` Put '`important`' flag on all messages which have target mark '`*`'. If already flagged as '`important`', remove the flag. (`wl-summary-target-mark-mark-as-important`)

`m F` Put arbitrary global flag entered in the minibuffer on all messages which have target mark '`*`'. (`wl-summary-target-mark-set-flags`)

`m !` Mark as unread all messages which have target mark '`*`'. (`wl-summary-target-mark-mark-as-unread`)

`m o` Put refile mark on the messages which have target mark '`*`'. (`wl-summary-target-mark-refile`) See Section 5.7 [Mark and Action], page 38 .

`m O` Put copy mark on the messages which have target mark '`*`'. (`wl-summary-target-mark-copy`) See Section 5.7 [Mark and Action], page 38 .

`m d` Put disposal mark on the messages which have target mark '`*`'. (`wl-summary-target-mark-dispose`) See Section 5.7 [Mark and Action], page 38 .

`m D` Put force deletion mark on the messages which have target mark '`*`'. (`wl-summary-target-mark-delete`) See Section 5.7 [Mark and Action], page 38 .

`m i` Put prefetch reservation mark on messages which have target mark '`*`'. (`wl-summary-target-mark-prefetch`) See Section 5.7 [Mark and Action], page 38 .

`m y` Save messages which have target mark '`*`'. (`wl-summary-target-mark-save`)

`m u` Unmark all temporal marks. (`wl-summary-delete-all-temp-marks`)

`m a` Put target mark '`*`' on the all messages. (`wl-summary-target-mark-all`)

`m t` Put target mark '`*`' on the messages in the current thread. (`wl-summary-target-mark-thread`)

`m r` Put target mark '`*`' on the messages in the specified region. (`wl-summary-target-mark-region`)

`m A` Prepare a draft which cites all messages which have target mark '`*`'. (`wl-summary-target-mark-reply-with-citation`)

`m f` Prepare a draft which forwards all messages which have target mark '`*`'. (`wl-summary-target-mark-forward`)

m U Uudecode the messages which have target mark '*'. (`wl-summary-target-mark-uudecode`)

m ? Pick messages from the '*' marked messages. That is, '*' marks on the messages are remained if the specified condition is satisfied. (`wl-summary-target-mark-pick`)

m # Print out all messages which have target mark '*'. (`wl-summary-target-mark-print`)

m | Pipe content of each message with target mark '*' to some specified external process. (`wl-summary-target-mark-pipe`)

M-t Toggle offline/online status of Wanderlust. (`wl-toggle-plugged`)

C-t Start Wanderlust's plug-status manager. (`wl-plugged-change`)

C-c C-o Move to the draft buffer if available. If multiple draft buffer exists, moved to one after another. If prefix argument is specified, load draft folder's message to the draft buffer and jump to it. (`wl-jump-to-draft-buffer`)

M-w Save the message at the current cursor point. (`wl-summary-save-current-message`)

C-y Regard the message at the current cursor point as parent, connect the message saved by `wl-summary-save-current-message` to the thread. (`wl-summary-yank-saved-message`)

C-x C-s Save the current summary. (`wl-summary-save-status`)

5.9 Customiziable variables

`wl-summary-move-order`
: The initial setting is `unread`. Specify cursor moving policy. If you want to precede new messages, set `new`. If you want to precede unread messages, set `unread`. If `nil`, proceed to next message.

`wl-auto-select-first`
: The initial setting is `nil`. If non-nil, first message is automatically displayed when you enter the folder.

`wl-auto-select-next`
: The initial setting is `nil`. This controls behavior when there is no unread message in current summary.

```
nil: asks whether you want to go back to folder mode
'unread: asks whether you want to go to next unread folder
  If the next one comes to be possessing no unread message
  by treatment of cross-posted messages or Scoring, then
  asks whether you want to go to next to next folder.
'skip-no-unread: similar as unread
  But does not ask before going to next to next folder.
otherwise: asks whether you want to go to next unread folder
```

It might be useful to set `'skip-no-unread` for people who want to continue reading by just pressing and pressing space key.

`wl-thread-insert-opened`
: The initial setting is `nil`. If non-nil, thread is inserted as opened.

`wl-thread-open-reading-thread`
: The initial setting is `t`. If non-nil, reading thread is automatically opened though it is closed thread.

`wl-summary-exit-next-move`
: The initial setting is `t`. If non-nil, move to next folder at summary exit.

`wl-folder-move-cur-folder`
: The initial setting is `nil`. If non-nil, cursor position on the folder is moved.

`wl-summary-weekday-name-lang`
: Specify language of the weekday. '`en`' displays English, '`fr`' displays French, '`de`' displays Deutsch. You should rescan summary view after changing this value.

`wl-summary-fix-timezone`
: The initial setting is `nil`. Time zone of the date string in summary mode is adjusted using this value. If `nil`, it is adjust to the default time zone information (system's default time zone or environment variable '`TZ`').

`wl-use-petname`
: The initial setting is `t`. If non-nil, sender part displays nickname.

`wl-break-pages`
: The initial setting is `t`. If non-nil, message is split as pages by '`^L`'.

`wl-summary-from-function`
: Format function to display sender in summary. The initial setting is `wl-summary-default-from`.

`wl-summary-no-from-message`
: The initial setting is '`nobody@nowhere?`'. A string which is displayed when there's no '`From:`' field in the message.

`wl-summary-subject-function`
: Format function to display subject in summary. The initial setting is `wl-summary-default-subject` and it will cut the list name part etc. on the top of the subject. To display subject as it is, set as follows.

```
(setq wl-summary-subject-function 'identity)
```

`wl-summary-no-subject-message`
: The initial setting is '`(WL:No Subject in original.)`'. A string which is displayed when there's no '`Subject:`' field in the message.

`wl-summary-default-view`
: The initial setting is `'thread`. The default state for newly created summary. You can set either `'thread` for thread view or `'sequence` for sequential view.

`wl-summary-use-frame`
: The initial setting is `nil`. If non-nil, use new frame for the summary.

`wl-use-folder-petname`
: The initial setting is the list shown below:

```
(modeline)
```

A list of display policy (symbol) of folder nickname. Available symbols are:

`modeline` Display folder petname on modeline.

`ask-folder`
: Destination folder is notified as nickname if `wl-auto-select-next` is non-nil.

`read-folder`
: You can input folder name by nickname in the function `wl-summary-read-folder`.

`wl-summary-move-direction-toggle`
: The initial setting is `t`. If non-nil, last executed *p*, *P*, *n*, *N* toggles the direction of cursor move. If you want to aware of reading direction, set this to `t`.

`wl-summary-width`
: The initial setting is 80. Width of summary line. If `nil`, summary line's width is as is.

`wl-summary-print-argument-within-window`
: The initial setting is `nil`. If non-nil, the action argument is always printed on right side of window.

`wl-summary-indent-length-limit`
: The initial setting is 46. Specify the limit of thread indent level. `nil` means unlimited indent level. If you set this to `nil` you should set `wl-summary-width` to `nil`, too.

`wl-summary-max-thread-depth`
: The initial setting is 15. If thread depth of the message is larger than this value, the thread is divided.

`wl-summary-recenter`
: The initial setting is t. If non-nil, cursor point is moved to the center of the summary window.

`wl-summary-max-thread-depth`
: The initial setting is 30. If thread depth is larger than this value, divide it.

`wl-summary-divide-thread-when-subject-changed`
: The initial setting is `nil`. If non-nil, thread is split if the subject is changed.

`wl-summary-search-via-nntp`
: The initial setting is `confirm`.

If non-nil and `wl-summary-jump-to-msg-by-message-id` failed, call `wl-summary-jump-to-msg-by-message-id-via-nntp` and search message from the NNTP server `elmo-nntp-default-server`. The value of `elmo-nntp-default-user`, `elmo-nntp-default-port`, `elmo-nntp-default-stream-type` are used.

If `confirm`, server name can be specified. You can specify NNTP folder format like '`-:username@servername:119!`'.

`wl-summary-keep-cursor-command`
: The initial setting is the list shown below:

```
(wl-summary-goto-folder wl-summary-goto-last-visited-folder)
```

When you entered to summary by these commands and the target summary buffer already exists, summary status is not automatically updated and cursor position is saved.

`elmo-folder-update-threshold`
: The initial setting is 500. If updated message number is larger than this value, confirm whether drop them or not (in the case where the value of `elmo-folder-update-confirm` is non-nil).

`elmo-folder-update-confirm`
: The initial setting is `t`. If the value is non-nil, do check with `elmo-folder-update-threshold`.

`wl-summary-always-sticky-folder-list`
: The initial setting is `nil`. `wl-summary-always-sticky-folder-list` specifies the folders that are automatically stuck. Each element is regexp of folder name.

`wl-summary-reserve-mark-list`
: The initial setting is the list shown below:

```
("o" "O" "D" "d" "i")
```

If a message is already marked as temporal marks in this list, the message is not marked by any mark command.

`wl-summary-skip-mark-list`
: The initial setting is the list shown below:

```
("D" "d")
```

If a message is already marked as temporal marks in this list, the message is skipped at cursor move.

`elmo-message-fetch-threshold`
: The initial setting is 30000 (bytes). If displaying message has larger size than this value, Wanderlust confirms whether fetch the message or not (in the case where the value of `elmo-message-fetch-confirm` is non-nil).

`elmo-message-fetch-confirm`
: The initial setting is `t`. If the value is non-nil, do check with `elmo-message-fetch-threshold`.

`wl-prefetch-threshold`
: The initial setting is 30000 (bytes). If prefetching message has larger size than this value and `wl-prefetch-confirm` is non-nil, Wanderlust confirms whether prefetch the message or not. If `wl-prefetch-threshold` is `nil`, the message is prefetched without confirmation.

`wl-prefetch-confirm`
: The initial setting is `t`. If non-nil, Wanderlust confirms whether prefetch the message or not if the message has larger size than `wl-prefetch-threshold`.

`elmo-imap4-use-cache`
: The initial setting is `t`. If non-nil, messages read via IMAP4 are cached.

`elmo-nntp-use-cache`
: The initial setting is `t`. If non-nil, messages read via NNTP are cached.

`elmo-pop3-use-cache`
: The initial setting is `t`. If non-nil, messages read via POP3 are cached.

`elmo-shimbun-use-cache`
: The initial setting is `t`. If non-nil, messages read in Shimbun folders are cached.

`wl-summary-resend-use-cache`
: The initial setting is `nil`. If non-nil, messages are resend using cache even in the offline status. Note that if you use cache, the message identity is not guaranteed.

`wl-folder-process-duplicates-alist`
: The initial setting is `nil`. This list determines how to deal with duplicated messages in the same folder. Each item in the list is regexp of folder name and action; you can specify any one of the following in the place of action:

```
nil : do nothing for duplicated messages.
hide : hide duplicated messages from the summary.
read : set duplicated messages as read.
```

Following is an example (hide duplicated messages in multi folders)

```
(setq wl-folder-process-duplicates-alist
                  '(("^\\+draft$" . nil) ("^\\+trash$" . nil)
                    ("^\\*.*" . hide) (".*" . read)))
```

`wl-summary-flag-alist`
: The initial setting is as follows:

```
((important "orange"))
```

Specify the color and the mark of message in summary buffer with flag. If the mark are omitted, the mark specified in the variable `wl-summary-flag-mark` is assumed. If multiple global flags are on one message, the former flag in this list is preferred.

Example:

```
(setq wl-summary-flag-alist
      '((important "purple")
        (todo "red")
        (business "green" "B")
        (private "blue" "X")))
```

`wl-summary-display-mime-mode-list`
: The initial setting is the list shown below:

```
(mime as-is)
```

The function `wl-summary-toggle-mime` switch specification of MIME analysis in the order of this list. You can specify one of the follows.

```
mime        : Header and body are decoded.
header-only : Only header is decoded.
as-is       : Header and body are not decoded.
```

6 Message Buffer

Message Buffers utilize MIME-View mode of SEMI. For operational procedures and key bindings, refer to respective documents. See Section "MIME-View" in *a MIME user interface for GNU Emacs* . You can also see help by `?` in message buffer.

p at the top of a message or *n* at the bottom of a message brings you back to Summary mode. *l* toggles display of Summary mode buffer.

6.1 Key Bindings

l Toggles display of Summary buffer. (`wl-message-toggle-disp-summary`)

Button-2 Assumes '`Message-ID:`' at the mouse pointer, and shows the corresponding message if found. (`wl-message-refer-article-or-url`)

Button-4 (upward movement of a wheel)
Scrolls the message backwards. When the top of the message is hit, moves to the previous message. (`wl-message-wheel-down`)

Button-5 (downward movement of a wheel)
Scrolls the message forward. When the bottom of the message is hit, moves to the next message. (`wl-message-wheel-up`)

D Delete the part under cursor. In fact it appends modified message to the current folder then moves old one to trash folder. Therefore the message number will be changed. (`wl-message-delete-current-part`)

6.2 Customizable Variables

`wl-message-window-size`
Initial setting is `(1 . 4)`. It is a cons cell and the ratio of its car and cdr value corresponds to the ratio of Summary and Message windows.

`wl-message-ignored-field-list`
Initial setting is `nil`. All fields that match this list will be hidden in message buffer. Each elements are regexp of field-name. If `nil`, the value of `mime-view-ignored-field-list` is used.

`wl-message-visible-field-list`
Initial setting is `nil`. All fields that match this list will be display in message buffer. Each elements are regexp of field-name. This value precedes `wl-message-ignored-field-list`. If `nil`, the value of `mime-view-visible-field-list` is used.

`wl-message-sort-field-list`
Initial setting is '("Return-Path" "Received" "^To" "^Cc" "Newsgroups" "Subject" "^From"). Header fields in message buffer are ordered by this value. Each elements are regexp of field-name.

`wl-message-truncate-lines`
The initial value is the value of `default-truncate-lines`. If it is non-nil, truncate long lines in message buffer.

`wl-message-auto-reassemble-message/partial`
: The initial setting is `nil`. If non-nil, automatically reassemble fragments of the message on displaying when its MIME media type is message/partial.

7 Draft Buffer

At Summary mode, pressing `w` and the like creates a new draft buffer. You can edit and send mail and news articles in this buffer.

By pressing `W`, Wanderlust guess addressees and prepare draft buffer with those if possible.

7.1 Tips

Basically it is Emacs-standard mail mode.

7.1.1 Parameters for Sending

According to the information of servers to send messages, configure following variables.

`wl-smtp-posting-server`
: The name of the SMTP server used for mail transmission.

`wl-smtp-posting-port`
: The SMTP port number for mail transmission. Without configuration, use default SMTP port number (25).

`wl-nntp-posting-server`
: The name of NNTP server used for news submission. Without configuration, use `elmo-nntp-default-server`.

`wl-nntp-posting-port`
: The NNTP port number for news submission. Without configuration, use `elmo-nntp-default-port`.

You may configure following variables on demand. See section Variables of Draft Mode for detail See Section 7.3 [Variables of Draft Mode], page 60 .

`wl-smtp-posting-user`
: User name for authentication by SMTP AUTH.

`wl-smtp-authenticate-type`
: The authentication method for SMTP AUTH. Without configuration, authentication will not be carried out.

`wl-smtp-authenticate-realm`
: The authentication realm for SMTP AUTH. Without configuration, authentication realm will not be specified.

`wl-smtp-connection-type`
: Specify how to establish SMTP connections.

`wl-nntp-posting-user`
: User name for AUTHINFO authentication on news submission.

`wl-nntp-posting-stream-type`
: Specify how to establish NNTP connections.

7.1.2 Editing Message Header

You can freely edit header region above '`--text follows this line--`', until you invoke the sending operation.

Initially, the cursor is at the '`To:`' field. Fill in recipients addresses. `TAB` completes them.

You can use following headers to specify recipients. Add some of them by yourself. Field names can be completed by `TAB`.

'`Newsgroups:`'
: Specify newsgroups to which you post the news article.

'`Cc:`'
: Specify addresses to send a copy (Carbon Copy) of the message.

Following ones are removed from the header contents before sending.

'`Bcc:`'
: Specify addresses to send a copy (Blind Carbon Copy) of the message.

'`Fcc:`'
: Specify folders in which a copy of the message is saved.

'`Ecc:`'
: Specify recipients to send encapsulated copy of the message.

You can add initial headers by following variables.

`wl-fcc`
: The initial setting is `nil`. If non-nil, the value of this variable is inserted as a '`Fcc:`' of the draft when it is prepared. If function is specified, its return value is used.

`wl-bcc`
: The initial setting is `nil`. If non-nil, the value of this variable is inserted as a '`Bcc:`' of the draft when it is prepared.

7.1.3 Editing Messages and Sending

As a matter of course, editing message body can be performed in the same way as usual writing. You may write message body under '`--text follows this line--`' line. (NOTE: Be sure to leave the line '`--text follows this line--`' intact.)

Multi-part editing utilize MIME edit mode of SEMI. For procedures of editing, refer to respective documents. See Section "MIME-Edit" in *a MIME user interface for GNU Emacs* . You can also see help by *C-c C-x ?* in draft buffer.

If you save the draft buffer you are editing, it is appended to the folder specified by `wl-draft-folder`. You can leave draft buffer after storing it for future editing by *C-c C-z* (`wl-draft-save-and-exit`).

If you have finished editing, you can send message by *C-c C-c*.

7.1.4 Dynamic Modification of Messages

You can set `wl-draft-config-alist` so that header and body of the message will automatically modified depending on information of header and others.

The initial setting of `wl-draft-config-alist` is `nil`.

In the example below, the header is modified when `wl-draft-send-and-exit` or `wl-draft-send` is invoked. You can set `wl-interactive-send` to non-nil so as to confirm changes before sending the message.

```
(setq wl-draft-config-alist
      '(((string-match "aaa\\.example\\.com$" (system-name))
         ;; applied if the expression is non-nil
         (wl-smtp-posting-server . "mailserver-B")
         (wl-nntp-posting-server . "newsserver-B")
         ;; settings of temporary variables
         )
        ("^To: .*user@aaa\\.bbb\\.example\\.com"
         ;; applied if it matches the header of the draft buffer
         ("Organization" . (format "Go %s" my-webpage)))
                        ;; you can write elisp expressions here (eval only)
         (top . "Hello.\n")     ;; inserted at the top of the body
         (bottom . "\nBye.\n") ;; inserted at the bottom of the body
        ))
```

The format of `wl-draft-config-alist` is:

```
'(("regexp of the header" or elisp expression
  ("Field" . value(elisp expression))
   (variable . value(elisp expression))
   (sub-function . value(elisp expression))
   function
   ...)
  ("regexp of the header" or elisp expression
   ("Field" . value(elisp expression))
   ...))
```

Per default, there are 13 following sub-functions.

```
'header:      Inserts the specified string at the bottom of the header.
'header-top:  Inserts the specified string at the top of the header.
'header-file: Inserts the specified file at the bottom of the header.
'x-face:      Inserts 'X-Face:' field with the content of the specified file.
'top:         Inserts the specified string at the top of the body.
'top-file:    Inserts the specified file at the top of the body.
'body:        Replaces the body with the specified string.
              Specifying nil deletes the entire body string.
'body-file:   Replaces the body with the content of the specified file.
'bottom:      Inserts the specified string at the bottom of the body.
'bottom-file: Inserts the specified file at the top of the body.
'part-top:  Inserts the specified string at the top of the current part.
'part-bottom: Inserts the specified string at the bottom of the current part.
'template:    Applies the specified template.
              (refer to the next subsection)
```

These are defined in `wl-draft-config-sub-func-alist` and you can change them or add your own functions. If you read the code, you can easily find how to write the functions.

At the first of each item, *a regular expression of the header* or an *elisp expression* should be specified. In the case of an elisp expression, the item is applied when the expression is evaluated non-nil.

Per default, when multiple items match or are evaluated non-nil, all such items are applied, but if you set a variable `wl-draft-config-matchone` to `t`, only the first matching one is applied.

At the second of the item, a cons or a list of functions should be specified. The car part of the cons should be a header field, a variable, or a sub-function. When a header field is specified, the field will be modified. If a variable is specified, the value of the variable will be modified temporarily.

In the cdr part of a cons, not only a variable but also an elisp expression can be specified as is. If the car part is a header field and the cdr part is `nil`, the field will be deleted.

See the next example as well:

```
(setq wl-draft-config-alist
      '((reply                          ;; (1)
         "X-ML-Name: \\(Wanderlust\\|emacs-mime-ja\\|apel-ja\\)"
         ;; applied if it matches the header of the buffer being replied
         (body . "  Hello.\n")
         (template . "default")
         )))
```

As in the (1) above, if a header regexp is prepended with `reply`, it is applied when the draft is prepared by `wl-summary-reply` for example, and when it matches the header being replied. It is ignored when there is no buffer being replied, like after `wl-draft` was invoked.

If you want to use name of parent folder, you can refer the buffer local variable `wl-draft-parent-folder`. In the following example, Wanderlust changes From according to the folder name of the summary in which the draft was invoked.

```
(setq wl-draft-config-alist
      '(((string-match \".*@domain1$\" wl-draft-parent-folder)
         (\"From\" . \"user@domain1\"))
        ((string-match \".*@domain2$\" wl-draft-parent-folder)
         (\"From\" . \"user@domain2\"))))
```

Note that `wl-draft-config-alist` is applied only once when `wl-draft-send-and-exit` or `wl-draft-send` is invoked. Therefore, if you want to apply `wl-draft-config-alist` again after aborting transmission, execute *C-c C-e* (`wl-draft-config-exec`) explicitly.

If you don't want to apply `wl-draft-config-alist` when `wl-draft-send-and-exit` or `wl-draft-send` is invoked, do the following:

```
(remove-hook 'wl-draft-send-hook 'wl-draft-config-exec)
```

If you want to apply `wl-draft-config-alist` when a draft buffer is prepared, do the following:

```
(add-hook 'wl-mail-setup-hook 'wl-draft-config-exec)
```

If you want to apply `wl-draft-config-alist` when you re-edit a mail from summary mode by typing *E*(`wl-summary-reedit`), do the following:

```
(add-hook 'wl-draft-reedit-hook 'wl-draft-config-exec)
```

7.1.5 Inserting Templates

Set a variable `wl-template-alist`, and type *C-c C-j* or *M-x wl-template-select* in the draft buffer.

The format of `wl-template-alist` is almost the same as `wl-draft-config-alist`. See Section 7.1.4 [Dynamical Message Re-arrangement], page 55 .

```
(setq wl-template-alist
      '(("default"
         ("From" . wl-from)
         ("Organization" . "Example Co.Ltd.")
         (body . "Hello.\n"))
        ("report"
         (template . "default")                ;; (a)
         ("To" . "boss@example.com")
         ("Subject" . "Report")
         (body-file . "~/work/report.txt")
         )
        ))
```

As you can see, the only difference is item (template) names such as '`default`' and '`report`', instead of a regexp of header. Because definition of each item is the same as `wl-draft-config-alist`, you can call another template, like (a).

Executing the command `wl-template-select` results in template selection, but the result differs depending on variable `wl-template-visible-select`.

If `wl-template-visible-select` is `t` (default), a buffer window is shown below the draft buffer. You can select a template by *n* and *p* seeing the buffer window.

Press the `RET` key and the template is actually applied to the draft buffer. If you press *q*, nothing is applied. In addition, you can adjust the window size by `wl-template-buffer-lines`.

If `wl-template-visible-select` is `nil`, you should type the name of the template in the mini buffer.

If `wl-template-select` is executed with prefix argument, inversed value of `wl-template-visible-select` is used.

As shown in the example in `wl-draft-config-alist`, you can select '`default`' template by writing:

```
(template . "default")
```

7.1.6 Sending mail by POP-before-SMTP

You can send mail by POP-before-SMTP. Necessary setting is

```
(setq wl-draft-send-mail-function 'wl-draft-send-mail-with-pop-before-smtp)
```

to change mail posting function from its default value `wl-draft-send-mail-with-smtp`. Also you would configure following variables on demand.

`wl-pop-before-smtp-user`
: The POP user name for POP-before-SMTP authentication. If unset, `elmo-pop3-default-user` is used.

`wl-pop-before-smtp-server`
: The POP server name for POP-before-SMTP authentication. If unset, `elmo-pop3-default-server` is used.

`wl-pop-before-smtp-authenticate-type`
: The POP authentication method for POP-before-SMTP authentication. If unset, `elmo-pop3-default-authenticate-type` is used.

`wl-pop-before-smtp-port`
: The POP port number for POP-before-SMTP authentication. If unset, `elmo-pop3-default-port` is used.

`wl-pop-before-smtp-stream-type`
: If `ssl`, POP connection is established using SSL. If `starttls`, STARTTLS (RFC2595) connection will be established. If unset, `elmo-pop3-default-stream-type` is used.

If variables for POP-before-SMTP (`wl-pop-before-smtp-*`) are unset, settings for POP folders (`elmo-pop3-default-*`) are used. Therefore, if SMTP server and POP server are actually the same, and if POP folder per default (such as '`&`') is available, no settings are required.

Refer to the following URL about POP-before-SMTP.

http://www.iecc.com/pop-before-smtp.html

7.2 Key Bindings

C-c C-y Cites the content of the current message buffer (the part under cursor). If the region is active, cites the region (it affects only if `transient-mark-mode` (on GNU Emacs) or `zmacs-regions` (on XEmacs) is Non-nil). If the command is called with prefix argument, the text inserted by yank command (the text content of clipboard) is cited. (`wl-draft-yank-original`)

C-c C-p Previews the content of the current draft. This is useful for previewing MIME multi-part messages. (`wl-draft-preview-message`)

C-c C-s Sends the content of the current draft. Does not erase the draft buffer. This is useful for sending multiple messages, which are a little different from each other. (`wl-draft-send`)

C-c C-c Sends the content of the current draft and erases the draft buffer. (`wl-draft-send-and-exit`)

C-x C-s Save the current draft. (`wl-draft-save`)

C-c C-k Kills the current draft. (`wl-draft-kill`)

C-x k Kills the current draft. (`wl-draft-mimic-kill-buffer`)

C-c C-z Saves the current draft, and erases the draft buffer. This is useful if you want to suspend editing of the draft. You can resume editing by pressing *E* (`wl-summary-reedit`) in the '`+draft`' folder. (`wl-draft-save-and-exit`)

C-c C-r Encodes or decodes the specified region in Caesar cipher. (`wl-caesar-region`)

C-l Recenter and rehighlight current draft. (`wl-draft-highlight-and-recenter`)

M-t Toggles off-line/on-line states of Wanderlust. (`wl-toggle-plugged`)

`C-c C-o` Jumps to the other draft buffer, if exists. With prefix argument, reads a file (if any) from the draft folder when there is no such buffer. (`wl-jump-to-draft-buffer`)

`C-c C-e` Applies `wl-draft-config-alist`. (`wl-draft-config-exec`)

`C-c C-j` Selects a template. (`wl-template-select`)

`C-c C-a` Enter Address Manager. See Section 11.2 [Address Manager], page 84 . (`wl-addrmgr`)

`C-c C-d` Elide the text between point and mark (`wl-draft-elide-region`). The text is killed and replaced with the contents of the variable `wl-draft-elide-ellipsis`. The default value is to use an ellipsis ('`[...]`').

7.3 Customizable Variables

`wl-subscribed-mailing-list`
: The initial setting is `nil`. Mailing lists to which you subscribe. If any of these are contained in '`To:`' or '`Cc:`' field of a reply draft, removes your own address from '`Mail-Followup-To:`' and '`Cc:`'. And if any of these are contained in '`To:`' or '`Cc:`' field of a message to be automatically re-filed, the destination folder will be leaned in connection with the address.

  Example:

```
(setq wl-subscribed-mailing-list
      '("wl@ml.gentei.org"
        "apel-ja@m17n.org"
        "emacs-mime-ja@m17n.org"))
```

`wl-insert-mail-followup-to`
: The initial setting is `nil`. If non-nil, '`Mail-Followup-To:`' field is automatically inserted in the draft buffer.

`wl-insert-mail-reply-to`
: The initial setting is `nil`. If non-nil, '`Mail-Reply-To:`' field is automatically inserted in the draft buffer.

`wl-auto-insert-x-face`
: The initial setting is `t`. If non-nil and there is an encoded X-Face string in a file '`~/.xface`' (the value of the variable `wl-x-face-file`), inserts it as an '`X-Face:`' field in the draft buffer. If `nil`, it is not automatically inserted.

`wl-insert-message-id`
: The initial setting is `t`. If non-nil, '`Message-ID:`' field is automatically inserted on the transmission.

`wl-message-id-use-message-from`
: The initial setting is `t`. If non-nil, the value of '`From:`' field or `wl-from` will be used as the domain part of '`Message-ID:`'.

`wl-local-domain`
: The initial setting is `nil`. If `nil`, the return value of the function `system-name` will be used as the domain part of '`Message-ID:`'.

If `system-name` does not return FQDN (i.e. the full name of the host, like '`smtp.gohome.org`'), you **must** set this variable to the string of the local domain name without hostname (like '`gohome.org`'). That is, a concatenation of `system-name` '`.`' `wl-local-domain` is used as domain part of the '`Message-ID:`'.

If your terminal does not have global IP, set the value of `wl-message-id-domain`. (You might be beaten up on the Net News if you use invalid '`Message-ID:`'.)

Moreover, concatenation of `system-name` '`.`' `wl-local-domain` will be used as an argument to the HELO command in SMTP.

`wl-message-id-domain`
: The initial setting is `nil`. If non-nil, this value is used as a domain part of the '`Message-ID:`'. If your terminal does not have global IP address, set unique string to this value (e.x. your e-mail address).

`wl-unique-id-suffix`
: The initial setting is '`.wl`'. You can specify the string in generated Message-ID which appear just before '`@`' or '`%`'.

`wl-draft-config-alist`
: The initial setting is `nil`. Modifies the draft message just before the transmission. The content of `wl-draft-config-alist` will be automatically applied only once on the transmission. If you want to apply it manually, use *`C-c C-e`*. This command can be used many times.

`wl-template-alist`
: The initial setting is `nil`. This variable specifies the template to be applied in the draft buffer.

`wl-draft-config-matchone`
: The initial setting is `nil`. If non-nil, only the first matching item is used when `wl-draft-config-alist` is applied. If `nil`, all matching items are used.

`wl-template-visible-select`
: The initial setting is `t`. If non-nil, you can preview the result of the template selection in another window.

`wl-template-confirm`
: The initial setting is `nil`. If non-nil, asks for confirmation when you press the enter key to select template while previewing.

`wl-template-buffer-lines`
: The initial setting is 7. If `wl-template-visible-select` is non-nil, this variable specifies the size of the preview window.

`wl-draft-buffer-style`
: The initial setting is `full`. Style of draft buffer window (except for replying and forwarding). `keep` is to use current window, `full` is to use full frame window, `split` is to split current window and use it. If some function is specified, it is called with the draft buffer as an argument.

`wl-draft-reply-buffer-style`
: The initial setting is `split`. Style of draft buffer for replying and forwarding. `keep` is to use message buffer window, `full` is to use full frame window, `split` is to split message buffer window and use it. If some function is specified, it is called with the draft buffer as an argument.

`wl-draft-use-frame`
: The initial setting is `nil`. If non-nil, use new frame for the draft.

`wl-draft-reply-default-position`
: The initial setting is `body`. Specify initial cursor position on draft buffer for reply. `body` is to move cursor to the top of the message body, `bottom` to the bottom of the message body, and `top` to the top of the header.

`wl-draft-truncate-lines`
: The initial value is the value of `default-truncate-lines`. If it is non-nil, truncate long lines in draft buffer.

`wl-from`
: The initial setting is the value of the variable `user-mail-address`. The value of this variable is inserted as a '`From:`' field of the draft when it is prepared.

`wl-envelope-from`
: The initial setting is `nil`. The value of this variable is used for envelope from (MAIL FROM). If `nil`, the address part of `wl-from` is used.

`wl-user-mail-address-list`
: The initial setting is `nil`. This is the User's address list. If you have multiple addresses, set this variable.

`wl-reply-subject-prefix`
: The initial setting is '`Re: `'. In the '`Subject:`' of the reply draft, this string is prepended to the '`Subject:`' of being replied. You can specify a function to be message buffer of the reply target.

`wl-forward-subject-prefix`
: The initial setting is '`Forward: `'. In the '`Subject:`' of the forwarding draft, this string is prepended to the '`Subject:`' of being forwarded. You can specify a function to be message buffer of the forward target.

`wl-draft-reply-use-address-with-full-name`
: The initial setting is `t`. If non-nil, insert her full name with address when prepare a draft for reply a message. If it is `nil`, insert her address only.

`wl-draft-enable-queuing`
: The initial setting is `t`. This flag controls off-line transmission. If non-nil, the draft is sent off-line.

`wl-draft-use-cache`
: The initial setting is `nil`. If the value is non-nil and `wl-insert-message-id` is nil, cache the message which is sent.

`wl-fcc-force-as-read`
: The initial setting is `nil`. If the value is non-nil, Mark as read the message saved by '`Fcc:`'.

`wl-auto-flush-queue`
: The initial setting is t. This flag controls automatic transmission of the queue when Wanderlust becomes on-line. If non-nil, the queue is automatically transmitted (with confirmation by `y-or-n-p`). If you want to transmit it manually, press *F* in the folder mode.

`wl-ignored-forwarded-headers`
: Initial setting is '`\\(received\\|return-path\\|x-uidl\\)`'. All headers that match this regexp will be deleted when forwarding a message.

`wl-ignored-resent-headers`
: Initial setting is '`\\(return-receipt\\|[bdf]cc\\)`'. All headers that match this regexp will be deleted when resending a message.

`wl-draft-always-delete-myself`
: If non-nil, always removes your own address from '`To:`' and '`Cc:`' when you are replying to the mail addressed to you.

`wl-draft-delete-myself-from-bcc-fcc`
: If any of `wl-subscribed-mailing-list` are contained in '`To:`' or '`Cc:`' field, do not insert '`Bcc:`' or '`Fcc:`' field.

`wl-draft-send-mail-function`
: The initial setting is `wl-draft-send-mail-with-smtp`. This is the function to post mails. To use POP-before-SMTP, set this to `wl-draft-send-mail-with-pop-before-smtp`.

`wl-smtp-posting-server`
: The initial setting is `nil`. This is the SMTP server name for mail transmission.

`wl-smtp-posting-port`
: The initial setting is `nil`. This is the SMTP port number for mail transmission. If `nil`, default SMTP port number (25) is used.

`wl-smtp-posting-user`
: The initial setting is `nil`. This is the user name for SMTP AUTH authentication.

`wl-smtp-authenticate-type`
: The initial setting is `nil`. This string-valued variable specifies the authentication method for SMTP AUTH authentication. You may specify `plain`, `cram-md5`, `digest-md5`, `login`, etc. If `nil`, authentication will not be carried out.

`wl-smtp-authenticate-realm`
: The initial setting is `nil`. This string-valued variable specifies the authentication realm for SMTP AUTH authentication. You have to set this variable for DIGEST-MD5 authentication and so on. If `nil`, authentication realm is not specified in the authentication.

`wl-smtp-connection-type`
: The initial setting is `nil`. This symbol-valued variable specifies how to establish SMTP connections. If `nil`, use default connection type. If it is `starttls`, use STARTTLS (RFC3207). If it is `ssl`, use SSL.

`wl-nntp-posting-server`
: The initial setting is `nil`. This is the NNTP server name used for news submission. If `nil`, `elmo-nntp-default-server` is used.

`wl-nntp-posting-user`
: The initial setting is `nil`. This is the user name for AUTHINFO authentication on news submission. If `nil`, `elmo-nntp-default-user` is used. If it is still `nil`, AUTHINFO authentication will not be carried out.

`wl-nntp-posting-port`
: The initial setting is `nil`. This is the port number of the NNTP server used for news submission. If `nil`, `elmo-nntp-default-port` is used.

`wl-nntp-posting-stream-type`
: The initial setting is `nil`. If `nil`, `elmo-nntp-default-stream-type` is evaluated. If `ssl`, SSL is used for news submission. If `starttls`, STARTTLS (RFC2595) connection will be established.

`wl-nntp-posting-function`
: The initial setting is `elmo-nntp-post`. This is the function to post NNTP message.

`wl-nntp-posting-config-alist`
: The initial setting is `nil`. The value takes an alist to define NNTP server like following example. It takes precedence over `wl-nntp-posting-{server|user|port|function}`.

```
(setq wl-nntp-posting-config-alist
      '((",?gmane\\." . "news.gmane.org")
        (",?comp\\." .
         ((server . "news-server")
          (user . "newsmaster")
          (port . 119)
          (function . elmo-nntp-post)))
        (".*" . "default-news-server")))
```

`wl-pop-before-smtp-user`
: The initial setting is `nil`. This is the POP user name for POP-before-SMTP. If it is `nil`, `elmo-pop3-default-user` is used.

`wl-pop-before-smtp-server`
: The initial setting is `nil`. This is the POP server name for POP-before-SMTP. If it is `nil`, `elmo-pop3-default-server` is used.

`wl-pop-before-smtp-authenticate-type`
: The initial setting is `nil`. This is the authentication method for POP-before-SMTP authentication. If it is `nil`, `elmo-pop3-default-authenticate-type` is used.

`wl-pop-before-smtp-port`
: The initial setting is `nil`. This is the POP port number for POP-before-SMTP. If it is `nil`, `elmo-pop3-default-port` is used.

`wl-pop-before-smtp-stream-type`
: The initial setting is `nil`. This flag controls the use of SSL for POP-before-SMTP. If it is `nil`, `elmo-pop3-default-stream-type` is used. If `ssl`, SSL is used. If `starttls`, STARTTLS (RFC2595) connection will be established.

`wl-draft-queue-save-variables`
: Specifies a list of variable to which queued messages are saved on the off-line transmission.

`wl-draft-sendlog`
: The initial setting is `t`. If `t`, transmission log is written in '`~/.elmo/sendlog`'. It is written when:

  - drafts are sent by smtp or qmail
  - saved into folders by fcc
  - saved into folders by queuing

  (it is written even if the transmission fails). But transmission by '`im-wl.el`' is not written in the '`sendlog`' and left to the logging function of `imput`.

`wl-draft-sendlog-max-size`
: The initial setting is 20000 (in bytes). If `wl-draft-sendlog` is `t`, the log is rotated when it grows beyond the size specified by this variable.

`wl-use-ldap`
: The initial setting is `nil`. If non-nil, address completion uses LDAP.

`wl-ldap-server`
: The initial setting is '`localhost`'. LDAP server name for address completion.

`wl-ldap-port`
: The initial setting is `nil`. If non-nil, the value is used as port number.

`wl-ldap-base`
: The initial setting is '`c=US`'. LDAP search starting point (base) for address completion.

`wl-draft-remove-group-list-contents`
: The initial setting is `t`. If non-nil, remove the group-lists' members in the recipients when sending the message (group-list means the description such as '`Group: foo@gohome.org, bar@gohome.org;`' in the recipients).

8 Off-line Management

Wanderlust has on-line and off-line states.

8.1 Off-line State

Wanderlust has on-line and off-line states. In the off-line state, you cannot access messages via network, unless they are cached.

'`[ON]`' in the mode line indicates the on-line state. '`[--]`' in the mode line indicates the off-line state. In folder or summary modes, press *`M-t`* to switch between off- and on-line.

You can invoke Wanderlust in the off-line state by setting `wl-plugged` to `nil` in '`~/.wl`' or anything appropriate.

In the off-line mode, *`n`* and *`p`* command in the summary mode ignores uncached messages.

8.2 Enable Disconeected Operations

Even in the off-line state, provided that relevant messages are cached, and the variable `elmo-enable-disconnected-operation` (described later) is non-nil, you can following operations: See Section 8.3 [Plugged Mode], page 67 , See Section 8.4 [Off-line State settings], page 68 .

As soon as Wanderlust becomes on-line, such operations invoked off-line are reflected in the servers via network.

If the variable `elmo-enable-disconnected-operation` is `nil`, these off-line operations are not executed and causes an error on re-file or copy operations.

8.2.1 Transmission of Messages

You can proceed sending operation for mail/news messages while you are off-line, then it will be reserved for sending (if you are using '`im-wl.el`', it is irrelevant). Messages reserved for sending while off-line are accumulated in the queue folder, '`+queue`'. These messages are transmitted at once when Wanderlust becomes on-line.

You can visit '`+queue`' in the off-line state and confirm content of messages in the queue. You can also remove messages. Removed messages are not transmitted even in the on-line state.

8.2.2 Re-file and Copy (IMAP4)

Re-file and copy operations to IMAP folders invoked during the off-line state are accumulated in the queue, and reflected in the server side when Wanderlust becomes on-line. If you visit destination folders after off-line re-file or copy, it looks as if messages were appended even in off-line state.

For the safety reasons, messages re-filed off-line are removed from source folders only if their '`Message-ID:`' match messages on the servers. While the queue is processed, messages that failed to be re-filed or copied to the specified folders are appended to the folder '`+lost+found`'.

8.2.3 Creation of Folders (IMAP4)

You can create IMAP folders off-line. The creation of folders are reflected in the servers when Wanderlust becomes on-line. If the creation of those folders fails at that time for some reasons, messages to be re-filed into those are appended to the folder '`+lost+found`' instead.

8.2.4 Marking (IMAP4)

Off-line changes in unread/read and importance mark '`$`' information are also reflected in the servers when Wanderlust becomes on-line.

8.2.5 Pre-fetching

You can make reservations for pre-fetching messages in networking folders (IMAP, NNTP, POP3, shimbun). Reserved messages are marked with '`u`' but not cached yet. When Wanderlust becomes on-line, they are pre-fetched from servers.

8.3 Switching On-line/Off-line per Server/Port

M-t described above switches networking states as a whole, but you can switch on-line/off-line per server/port.

Pressing *C-t* in the folder or summary modes brings you in wl-plugged-mode shown below, in which you can change the plugged state for each port.

```
Queuing:[ON] AutoFlushQueue:[--] DisconnectedOperation:[ON]
[ON](wl-plugged)
  [--]hosta
    [--]smtp        +queue: 2 msgs (1,2)        ...sending queue
    [--]nntp(119)   +queue: 1 msg (3)           ...sending queue
  [ON]hostb
    [--]imap4/cram-md5(143) %#mh/wl(prefetch-msgs:3,mark-as-important:1)
                            %inbox(delete-msgids:1)    ...dop queue
    [ON]nntp(119)
    [ON]smtp
```

The first line indicates status of the following three variables, and simply pressing `SPC` or `RET` in each labeled column modifies the values of these variables.

```
"Queuing"               wl-draft-enable-queuing
"AutoFlushQueue"        wl-auto-flush-queue
"DisconnectedOperation" elmo-enable-disconnected-operation
```

where '`[ON]`' means its value is `t`, and '`[--]`' means `nil`.

The second and after lines indicate on-line/off-line states of servers and ports, where '`[ON]`' stands for on-line and '`[--]`' for off-line (in XEmacs or Emacs 21, they are shown with icons). Pressing `SPC` or `RET` in each line switches its state.

sending queue means messages accumulated in the folder '`+queue`' for off-line transmission, and *dop queue* means off-line operations when `elmo-enable-disconnected-operation` is `t`.

They are displayed if there are any of them. In the example above, in the sending queue there are two messages (the first and the second in the queue folder) for smtp to hosta and

one (the third) for nntp to hosta, and in the dop queue there are one for '`%inbox`' and two for '`%#mh/wl`'.

If you change '`(wl-plugged)`' in the second line, the variable `wl-plugged` is changed, so that the mode line indicator and plugged states of all ports are affected. If you change plugged states of any servers or ports, '`(wl-plugged)`' in the second line is affected depending on `elmo-plugged-condition` settings and the plugged state of each port.

8.4 Invoking Wanderlust in the Off-line State

As described before, if you set `wl-plugged` to `nil` in '`~/.wl`' or anything appropriate, you can invoke Wanderlust in the off-line state. You can specify off-line state on a per server or port basis. Refer to `wl-reset-plugged-alist` also.

Usually, when Wanderlust starts up, the plugged state of each port is read from '`~/.folders`' and `wl-smtp-posting-server`, `wl-nntp-posting-server` and so on. If you want to change the plugged state of these ports or to add other ports, configure `wl-make-plugged-hook` with a function.

```
(add-hook 'wl-make-plugged-hook
          '(lambda ()
             (elmo-set-plugged plugged-value(t/nil) server port)
                 ;; add or change plugged states of the port of the server
             (elmo-set-plugged plugged-value(t/nil) server)
                 ;; if the port is omitted, all ports are affected
                 ;; (you cannot omit the port if you newly add the server)
             ))
```

8.5 Customizable Variables

`wl-plugged`
: If this variable is set to `nil`, Wanderlust starts up in off-line mode from the beginning.

`wl-queue-folder`
: The initial setting is '`+queue`'. This is the folder in which messages in the transmission queue are accumulated.

`wl-auto-flush-queue`
: The initial setting is `t`. This flag controls automatic transmission of the queue when Wanderlust becomes on-line. If non-nil, the queue is automatically transmitted (with confirmation by `y-or-n-p`). If you want to transmit it manually, press *F* in the folder mode.

`elmo-enable-disconnected-operation`
: The initial setting is `t`. Controls off-line operations regarding networking folders. If non-nil, off-line operations are carried out.

`elmo-lost+found-folder`
: The initial setting is '`+lost+found`'. This is the folder to which messages are saved when they fails to be appended while the off-line re-file/copy queue is processed.

`elmo-plugged-condition`

The initial setting is `one`. The value of `wl-plugged` reflects the return value of the function `elmo-plugged-p` (without arguments). This variable `elmo-plugged-condition` specifies the condition on which the return value of `(elmo-plugged-p)` should be t depending on the plugged state of each port.

```
'one         : plugged if one or more ports are plugged.
'all         : plugged if all ports are plugged.
'independent : reflects wl-plugged (elmo-plugged) regardless of plugged
               states of the ports.
function     : reflects the return value of the function
 functions available per default
 'elmo-plug-on-by-servers
             : reflects the plugged state of the servers spec-
ified by the
               variable elmo-plug-on-servers.
 'elmo-plug-on-by-exclude-servers
             : reflects the plugged state of the servers that are not
               in elmo-plug-on-exclude-servers.
                  elmo-plug-on-exclude-servers defaults to
                   '("localhost"
                     (system-name)
                     (system-name)without the domain part)
Example 1:
 (setq elmo-plugged-condition 'all)
Example 2:
 (setq elmo-plug-on-servers '("smtpserver" "newsserver"))
 (setq elmo-plugged-condition 'elmo-plug-on-by-servers)
Example 3:
 (setq elmo-plug-on-exclude-servers '("localhost" "myname"))
 (setq elmo-plugged-condition 'elmo-plug-on-by-exclude-servers)
```

`wl-reset-plugged-alist`

The initial setting is `t`. If non-nil, plugged states are initialized on a per server or port basis when Wanderlust starts up.

If `nil`, plugged states are retained while Emacs is running. In other words, they are initialized when Emacs is restarted even if the value is `nil`.

9 Automatic Expiration and Archiving of Messages

9.1 Expiration

Expiration means deletion of old messages which have outlasted a certain period of time.

`wl-expire` supports not only simple deletion, but also moving to specified archiving folders.

9.2 How to Use

Configure `wl-expire-alist` and press `e` in the folder mode, or *M*-`e` in the summary mode.

9.2.1 Configuring `wl-expire-alist`

An example configuration of `wl-expire-alist` is shown below. Everything in this `wl-expire-alist` makes a great difference in expiration, so be careful. I advise you to set `wl-expire-use-log` to `t`, especially in the initial stage.

```
(setq wl-expire-alist
      '(("^\\+trash$"   (date 14) remove)
                                ;; delete
        ("^\\+tmp$"     (date 7) trash)
                                ;; re-file to wl-trash-folder
        ("^\\+outbox$"  (number 300) "$outbox;lha")
                                ;; re-file to the specific folder
        ("^\\+ml/tmp$"  nil)
                           ;; do not expire
        ("^\\+ml/wl$"   (number 500 510) wl-expire-archive-number1 t)
                           ;; archive by message number (retaining numbers)
        ("^\\+ml/.*"    (number 300 310) wl-expire-archive-number2 t)
                           ;; archive by a fixed number (retaining numbers)
        ("^\\+diary$"   (date 30) wl-expire-archive-date)
                           ;; archive by year and month (numbers discarded)
        ))
```

Items in the list has the format of:

(regexp-for-folders specification-of-messages-to-be-deleted destination)

The folder is examined if it matches *regexp-for-folders* from the beginning of the list. If you invoke expiration on the folder that does not match any of them, nothing will happen. And if either the second or the third element of the item is `nil`, expiration will not take place.

You can use any one of the following for *specification-of-message-to-be-deleted*:

`(number` *n1 [n2])*

deletes messages depending on the number of messages in the folder.

n1 is the number of messages which should survive deletion, for example if its value is 500, the newest 500 messages survive and the rests are deleted.

n2 is the number of messages in the folder on which expiration should take place, which defaults to *n1* + 1. For example if its value is 510, folders with

510 or more messages are expired. If you configured automatic expiration, frequently used folders may expire every time it receive messages, and you may be annoyed with the long delay in reading mail. In that case, you can set a wide margin between *n2* and *n1*, so that expiration would not take place until a certain number of messages accumulate.

Messages with marks in `wl-summary-expire-reserve-marks` (marked with important/new/unread) are not deleted. If `wl-expire-number-with-reserve-marks` is non-nil, the folder will expire so as to have 500 messages including such ones. Otherwise, it will have 500 messages except such ones.

`(date` *d1*`)` deletes messages depending on the dates.

Messages dated *d1* or more days ago are deleted, for example if its value is seven, messages seven days old or more are deleted. Note that the date is the one in the '`Date:`' field of the message, not when the message entered the folder.

Messages with no or invalid '`Date:`' field does not expire; you might have to delete them by hand.

You can use any one of the following in the place of *destination*:

`remove` deletes the messages instantly.

`hide` hide the messages from summary (messages are not deleted).

`trash` moves the messages to `wl-trash-folder`.

string(folder)

moves the messages to the folder specified with *string*.

It would be useful for specifying an archiving folder, but because this does not move important messages, it might be better to use the standard functions described below.

function invokes the specified *function*.

To the *function*, three arguments are passed: a folder name, a list of messages to be deleted, and msgdb information of the summary. You can specify function-specific arguments after the name of the *function*. Note that the list contains messages with marks in `wl-summary-expire-reserve-marks`, be careful in writing your own function.

These are four standard functions; three of them move messages to an archive folder in the specified way. This means old messages can be compressed and saved in a file, being deleted from the original folder. The last one divides messages to some MH folders.

`wl-expire-archive-number1`

re-files to archiving folders corresponding to the message numbers of the messages being deleted. For example, a message numbered 102 will be re-filed to '`wl-00100.zip`', 390 to '`wl-00300.zip`', and so on. If `wl-expire-archive-files` is 200, messages will be re-filed to '`wl-00000.zip`', '`wl-00200.zip`', '`wl-00400.zip`', ...

The archiving folders to which messages are re-filed are determined by the name of the folder as follows (in this case, archiving folders are handled as if `elmo-archive-treat-file` were non-nil).

If the folder type is localdir:

'*ArchiveDir/foldername-xxxxx.zip*'

For example, '`+ml/wl`' corresponds to '`$ml/wl;zip`' ('`~/Mail/ml/wl-00100.zip`').

The folder type is other than localdir:

'*ArchiveDir/foldertype/foldername-xxxxx.zip*'

For example, '`%#mh/ml/wl`' corresponds to '`$imap4/#mh/ml/wl;zip`' ('`~/Mail/imap4/#mh/ml/wl-00100.zip`').

As you can see, in the case of localdir, the folder type is not included in the path name, but otherwise it is included. And you can control the prefix to the archiving folder name by `wl-expire-archive-folder-prefix`. Refer to `wl-expire-archive-folder-prefix` for details.

`wl-expire-archive-number2`

re-files every certain number of messages to archiving folders.

This differs from '`wl-expire-archive-number1`' in that this re-files to the folder up to the specified number regardless of message numbers. The archiving folders to which messages are re-filed are determined in the same way as `wl-expire-archive-number1`.

`wl-expire-archive-date`

re-files messages depending on its date (year and month) to archive folders.

For example, a message dated December 1998 is re-filed to `$folder-199812;zip`. The name of the archiving folders except the date part are determined in the same way as `wl-expire-archive-number1`.

You can set the first argument to these three standard functions to non-nil in `wl-expire-alist` so as to retain message numbers in the folder. For example, it can be specified just after the name of the function:

```
("^\\+ml/wl$" (number 300 310) wl-expire-archive-number1 t)
```

If you omit the argument, consecutive numbers from 1 are assigned for each archiving folder.

`wl-expire-localdir-date`

divedes messages depending on its date (year and month) to MH folders e.g. to '`+ml/wl/1999_11/`', '`+ml/wl/1999_12/`'.

9.2.2 Treatment for Important or Unread Messages

If you specify any of `remove`, `trash`, a folder name, or a standard function, messages with marks in `wl-summary-expire-reserve-marks` (which are called *reserved messages* thereafter) are retained.

Per default, this variable include the important, new, and unread marks, so that messages with these marks are not removed. Note that you cannot include the temporary mark (i.e.

temporary marks are removed anyway), and be sure to process temporary marks before you invoke expiration.

9.2.3 Auto Expiration

The following setup invokes expiration when you move into the summary mode. There will be no confirmation, so make sure you made no mistake in regexp and other settings before you set up this.

```
(add-hook 'wl-summary-prepared-pre-hook 'wl-summary-expire)
```

In the folder mode, you can invoke expiration per group as well as per folder. Therefore, if you specify '`Desktop`' group, all folders matching `wl-expire-alist` expire.

9.3 Tips

9.3.1 Treating archive folders.

To treat archive folders created by `wl-expire-archive-number1` and so on, you must set non-nil value to `elmo-archive-treat-file`.

9.3.2 Confirming

If you are to use `remove`, try `trash` at first and see messages move to `wl-trash-folder` as expected, then replace it with `remove`. It would be dangerous to use `remove` from the beginning.

If you are to use `wl-expire-archive-number1` and the like, try to make a folder of the archiver type (`zip` or `lha`) and see if you can append messages to it. Even if settings in `wl-expire-alist` and `elmo-archive` are correct, messages would not be saved anywhere and disappeared in case the archiver program fails.

After you make sure you can archive to the folder correctly, you can invoke expiration and utilize the log.

If you set `wl-expire-use-log` to `t`, '`~/.elmo/expired-log`' should contain the log, for example:

```
delete  +ml/wl  (593 594 595 596 597 598 599)
move    +ml/wl -> $ml/wl-00600;tgz;wl  (600 601 602)
```

The first column indicates the operation, i.e. '`delete`', '`copy`', or '`move`'. The next is the name of the folder that expired. In the case of '`copy`' and '`move`', the destination folder is recorded after '`->`'. The last is the list of message numbers that are actually deleted or moved (in the case of '`copy`' and '`move`', the number is the one in the source folder, rather than the destination folder).

9.3.3 Re-filing Reserved Messages

The three standard functions copy reserved messages to the archive folder, but do not delete them from the source folder. Because reserved messages and the like always remain, they are recorded in '`~/.elmo/expired-alist`' so that they are not copied over and over again. They are not recorded if copied by `wl-summary-archive`.

If you enabled logging, usually '`move`' is recorded for re-filing, but instead '`copy`' and '`delete`' are recorded separately if reserved messages are involved. This is because it actually copies messages including reserved, then deletes ones except reserved in that case.

9.4 Customizable Variables

`wl-expire-alist`
: The initial setting is `nil`. This variable specifies folders and methods to expire. For details, refer to `wl-expire-alist` settings above.

`wl-summary-expire-reserve-marks`
: The initial setting is the list below.

```
(list wl-summary-flag-mark
      wl-summary-new-uncached-mark
      wl-summary-new-cached-mark
      wl-summary-unread-uncached-mark
      wl-summary-unread-cached-mark)
```

Messages with these marks are retained in the folder, even after expiration. Only permanent marks can be listed, not temporary marks.

You can list marks one by one as in the default; you can use the following settings as well:

`all` All messages with permanent marks are retained, i.e. `wl-summary-read-uncached-mark` is included in addition to the defaults.

`none` All messages are handled as usual ones that are already read, no matter what marks they have; even important messages are deleted.

`wl-expire-archive-files`
: The initial setting is 100. This variable specifies the number of messages to be retained in one archiving folder.

`wl-expire-number-with-reserve-marks`
: The initial setting is `nil`. If non-nil, if expiring messages are specified by `number`, messages with `wl-summary-expire-reserve-marks` are also retained.

`wl-expire-archive-get-folder-function`
: The initial setting is `wl-expire-archive-get-folder`.

This variable specifies a function that returns the name of an archiving folder for standard functions in the place of *destination*. You can use the following three variables for simple modification of folder names; if you want more complex settings, define your own function in this variable.

`wl-expire-archive-get-folder` can be customized by these variables:

- `wl-expire-archive-folder-name-fmt`
- `wl-expire-archive-folder-type`
- `wl-expire-archive-folder-prefix`

`wl-expire-archive-folder-name-fmt`
: The initial setting is '`%s-%%05d;%s`'. This is a `format` string for archiving folders used in `wl-expire-archive-number1` and `wl-expire-archive-number2`. Note that you must specify the message number by '`%%d`', because it is parsed twice by `format`.

If you modify this, adjust `wl-expire-archive-folder-num-regexp` as well.

`wl-expire-archive-date-folder-name-fmt`
: The initial setting is '`%s-%%04d%%02d;%s`'. This is a `format` string for archiving folders used in `wl-expire-archive-date`. Note that you must specify the message number by '`%%d`', because it is parsed twice by `format`. There should be '`%%d`' twice, one for the year and the other for the month.

  If you modify this, adjust `wl-expire-archive-date-folder-num-regexp` as well.

`wl-expire-archive-folder-type`
: The initial setting is `zip`. This variable specifies an archiver type of the archiving folders.

`wl-expire-archive-folder-prefix`
: The initial setting is `nil`. This variable specifies the prefix (directory structure) to archiving folders. Exercise extreme caution in using this feature, as it has not been seriously tested. In the worst case, there is a fear of destructing archiving folders.

  `nil` — There will be no prefix.

  `short` — For example, '`+ml/wl`' will be prefixed by '`wl`', resulting in '`$ml/wl-00000;zip;wl`'.

  `t` — For example, '`+ml/wl`' will be prefixed by prefix '`ml/wl`', resulting in

  '`$ml/wl-00000;zip;ml/wl`'.

`wl-expire-archive-folder-num-regexp`
: The initial setting is '`-\\([-0-9]+\\);`'. This variable specifies the regular expression to be used for getting message numbers from multiple archiving folders specified by `elmo-list-folders`. Set it in accordance with `wl-expire-archive-folder-name-fmt`.

`wl-expire-archive-date-folder-num-regexp`
: The initial setting is '`-\\([-0-9]+\\);`'. This is the regular expression to be used for getting message numbers from multiple archiving folders specified by `elmo-list-folders`. Set it in accordance with `wl-expire-archive-date-folder-name-fmt`.

`wl-expire-delete-oldmsg-confirm`
: The initial setting is `t`. If non-nil, messages older than the one with the largest number will be deleted with confirmation. If `nil`, they are deleted without confirmation.

  This feature is valid only if non-nil is specified as a argument to the standard functions so as to retain numbers.

`wl-expire-use-log`
: The initial setting is `nil`. If non-nil, expiration logs are recorded in '`~/.elmo/expired-log`'. They are appended but not truncated or rotated automatically; you might need to remove it manually.

`wl-expire-add-seen-list`

The initial setting is `t`.

If non-nil, when messages are re-filed by expiration, read/unread information is passed to the destination folder.

However if you do not read the destination folder from Wanderlust, '`seen`' under '`~/.elmo/`' grows larger and larger, so you might want to set this to `nil` if you are simply saving to some archiving folders. Even if its value is `nil`, messages in the archiving folders are simply treated as unread; it does not affect expiration itself.

`wl-expire-folder-update-msgdb`

The initial setting is `t`. If `t`, in the folder mode, expiration is carried out after updating summary information. If you specified a list of regular expressions of folder names, summary information is updated for matching folders only.

9.5 Archiving Messages

9.5.1 Archiving Messages

M-x wl-summary-archive copies the whole folder to archiving folders. If there are the archiving folders already, only new messages are appended.

You can use `wl-archive-alist` in order to specify how messages are archived according to their folder names, as in `wl-expire-alist`. For example:

```
(setq wl-archive-alist
      '(("^\\+tmp$"     wl-archive-date)
        ("^\\+outbox$"  wl-archive-number2)
        (".*"           wl-archive-number1)))
```

Each item in the list has the following format:

(folders-regexp deleting-function)

As you can see, you can only use a function after *folders-regexp*. Per default, there are three functions:

- `wl-archive-number1`
- `wl-archive-number2`
- `wl-archive-date`

As inferred from their names, they work similarly to "expire" versions, other than the following points:

– No messages are deleted

– Message numbers are retained even if invoked without arguments

These functions are good to archive all messages in a folder by their numbers or by their dates. These are also useful for backup or confirmation purposes before expiration. If you try to re-file them after they are archived, they are deleted but not re-filed.

Per default, the archiving folders to which messages are copied are determined automatically by `wl-expire-archive-get-folder-function`. You can copy to a specific folder by invoking with a prefix argument, i.e. *C-u M-x wl-summary-archive*.

Note that this feature has not been seriously tested, because you can simply copy to an archiving folder, for example by `wl-summary-copy-region`.

The archiving folders are determined by the same logic as in `wl-summary-expire`; the following customizable variables are relevant:

- `wl-expire-archive-files`
- `wl-expire-archive-get-folder-function`
- `wl-expire-archive-folder-name-fmt`
- `wl-expire-archive-folder-type`
- `wl-expire-archive-folder-prefix`
- `wl-expire-archive-folder-num-regexp`

9.5.2 Customizable Variables

`wl-archive-alist`

The initial setting is the list shown below:

```
((".*" wl-archive-number1))
```

This variable specifies a function that copies to archiving folders. To the function, three arguments are passed: a folder name, a list of messages in the folder, and msgdb information of the summary. Needless to say, you can use your own function.

10 Score of the Messages

Scoring is the function that associates a score (value) with each message, and marks as read or deletes from the summary according to it.

You can put target or important marks on essential messages, or read marks on the ones you do not want to read, for example spam articles.

This scoring function has a capability and a format similar to the one that Gnus has, although there are some unsupported features and Wanderlust specifics. See Section "Scoring" in *The gnus Newsreader* .

10.1 Score Commands

10.1.1 Score File Specification

`wl-score-folder-alist` specifies score files or variables in which scores are defined, corresponding to folder names.

```
(setq wl-score-folder-alist
      '(("^-.*"
         "news.SCORE"
         "my.SCORE")
        (".*"
         "all.SCORE")))
```

If paths to the score files are omitted, the directory specified in the variable `wl-score-files-directory` is assumed.

No matter what you write in `wl-score-folder-alist`, the default score file `wl-score-default-file` (‘`all.SCORE`’) is always read (it does not have to exist). Therefore, in the example above, the three score files, ‘`news.SCORE`’, ‘`my.SCORE`’, and ‘`all.SCORE`’ are read for the folders that matches ‘`^-.*`’.

10.1.2 Scored Messages

Scores are attached to the messages that are specified by `wl-summary-score-marks` temporarily when the summary is updated; when you exit from the summary, the scores are removed and reverts to the defaults.

10.1.3 Creation of Score Files

In the summary buffer, move to an appropriate message and type *`L`*. Then type *`s`*, *`s`*, and *`p`* at a prompt in a mini-buffer. The string in Subject is presented. Edit it and press `RET`.

This makes −1000 are scored for messages with the same ‘`Subject:`’ as the string you entered. That is, such a score file is created automatically.

Then, try typing *`h`* and *`e`* in the same summary buffer. The score file you just made appears. This buffer is called *score editing buffer* thereafter. When you type *`C-c C-e`* in it, you are prompted in the mini-buffer as you are previously; type `a`. Then a score entry for "From" should be inserted. In this way, you can create a score file easily either in the summary buffer or in the score editing buffer.

By the way, you might be aware the numbers of key strokes are different between *`s s p`* and `a`. This is determined by `wl-score-header-default-entry`. This variable specifies the

default score entries corresponding to header fields. For example, for "subject" field, a type and a time limit are prompted, but for "from" field, they are fixed upon automatically as substring and permanent respectively. However, score values can be modified by the prefix argument. Typing `?` at the mini-buffer shows a help on keys and corresponding headers and types.

At last, type `C-c C-c` in the score editing buffer. This saves the score file and terminates the edit mode. Typing `C-c C-c` after erasing contents of the buffer deletes the score file being edited.

10.1.4 Tips

10.1.4.1 Selecting Score Files

You can change score files to which scores are appended by `wl-summary-increase-score` and `wl-summary-lower-score` by `wl-score-change-score-file`.

10.1.4.2 Summing Up the Score

If you add the same entries by `wl-summary-increase-score`, `wl-summary-lower-score`, and `wl-score-edit-insert-entry`, scores for the entry is summed up.

For example, if you create '`from`' entry with the score of −1000 by `L a` and again '`from`' with −200, one entry with the score of −1200 will be created as a result.

10.1.4.3 Creating Thread Key

Creating '`Thread`' key by `wl-summary-increase-score` or `wl-summary-lower-score` appends '`Message-ID`' of all children.

10.1.4.4 Creating Followup Key

Creating '`Followup`' key by `wl-summary-increase-score` or `wl-summary-lower-score` appends '`Message-ID`' of the message at the cursor to '`References`' key. If `wl-score-auto-make-followup-entry` is non-nil, '`Message-ID`' of all messages to be followed up within dates specified by `wl-score-expiry-days`.

10.1.5 Key Bindings

`K` Increases the score for the current message. And the score entry is appended to the score file at the same moment. You can specify the score value by a prefix argument.

`L` Decreases the score for the current message. And the score entry is appended to the score file at the same moment. You can specify the score value by a prefix argument.

`h R` Re-applies the scoring. However, already scored messages are not scored anew.

`h c` Changes the score file currently selected.

`h e` Edits the score file currently selected. If there are multiple score files, the previously specified one is selected.

`h f` Edits an arbitrary score file and selects it.

h F Erases caches associated to the score files that are read. If you modified score files directly (from other than Wanderlust), you need to re-read them after erasing the cache.

h m Specifies the criterion for scores to be marked as read. Messages with scores less than this value are marked as read.

h x Specifies the criterion for scores to be deleted from the summary. Messages with scores less than this value are deleted. "Deleted" means it is not shown; they are not removed from the summary information or the folder. The deleted messages can be shown by rescan-noscore again.

10.1.6 Key Bindings in the Score Editing Buffer

C-c C-k Abandons the file being edited.

C-c C-c Saves the file being edited, and quits from the edit mode.

C-c C-p Re-draws the score.

C-c C-d Inserts the number of dates from Dec. 31, 1 B.C. It is used for creating the third factor of time-limited scores.

C-c C-s Inserts the header of the message selected in the summary buffer.

C-c C-e Inserts the score entry of the message selected in the summary buffer.

10.1.7 Customizable Variables

`wl-summary-default-score`
: The initial setting is 0 (zero). This variable specifies the default value of the score. The score is increased or decreased based upon this value.

`wl-summary-important-above`
: The initial setting is `nil`. Messages with scores larger than this value are attached with the important mark (‘`$`’). If `nil`, no important marks are attached.

`wl-summary-target-above`
: The initial setting is `nil`. Messages with scores larger than this value are attached with the target mark (‘`*`’). If `nil`, no target marks are attached.

`wl-summary-mark-below`
: The initial setting is 0 (zero). Messages with scores smaller than this value are marked as read.

`wl-summary-expunge-below`
: The initial setting is `nil`. Messages with scores smaller than this value are deleted from the summary. If `nil`, they are not deleted.

`wl-summary-score-marks`
: The initial setting is the list shown below:

```
(list wl-summary-new-uncached-mark
      wl-summary-new-cached-mark)
```

Messages with these marks are scored.

`wl-use-scoring`
: The initial setting is t. If non-nil, scoring is enabled.

`wl-score-files-directory`
: The initial setting is '`~/.elmo/`'. The default directory for score files.

`wl-score-interactive-default-score`
: The initial setting is 1000. This value is used as a score when a score factor is `nil` in the score file. It is also used in `wl-summary-increase-score` and `wl-summary-lower-score`, on condition that the value of `wl-score-header-default-entry` is `nil`.

`wl-score-expiry-days`
: The initial setting is 7. This is the number of days before time-limited scores are deleted.

`wl-score-update-entry-dates`
: The initial setting is `t`. If non-nil, it enables deletion of time-limited scores.

`wl-score-header-default-entry`
: Specifies the default value for each header field for score entries created by `wl-summary-increase-score`, `wl-summary-lower-score`, and `wl-score-edit-insert-entry`.

`wl-score-simplify-fuzzy-regexp`
: In the case of a type of a score entry is `fuzzy`, this specifies a regular expression to be deleted from the string. Because this is usually used for Subject, the default is prefixes that are attached by mailing list programs.

`wl-summary-rescore-partial-threshold`
: The initial setting is 200. When sync-all or rescan is executed, if there are messages more than this value, only the last same number of messages as this value are scored.

`wl-summary-auto-sync-marks`
: If non-nil, unread/important marks are synchronized when the summary does. Unread marks reflect information on the IMAP4 server. Important marks reflect information on the IMAP4 server (flagged or not), and contents of '`'flag`' folder. The initial setting is `t`.

10.2 Score File Format

The format of score files are the same as Gnus, and basically you can use Gnus score files as they are. But they are not fully compatible because some keys are not supported and there are Wanderlust specifics. See Section "Score File Format" in *The gnus Newsreader* .

```
(("subject"
  ("for sale" -1000 nil s)
  ("profit" -1000 nil s))
 ("from"
  ("spam@spamspamspam" -10000 nil s))
 ("followup"
  ("my@address" 3001 nil s))
 ("chars"
  (1000000 -10 nil >))
 (important 5000)
 (target 3000)
 (mark 0)
 (expunge -3000))
```

`string` If the key is a string, it is the name of the header to be matched. The following keys are available: `Subject`, `From`, `Date`, `Message-ID`, `References`, `To`, `Cc`, `Chars`, `Lines`, `Xref`, `Extra`, `Followup`, `Thread` `Chars` and `Lines` mean the size and the number of lines of the message, respectively. `Extra`, `Followup`, `Thread` are described later. The rest corresponds the field of the same name.

Arbitrary numbers of core entries are specified after the key. Each score entry consists of these five factors:

1. A factor that matches header. This should be a number in the cases of `lines` and `chars`, otherwise a string.
2. A score factor. When the first item matches, the score of the message is increased or decreased by this value.
3. A time limiting factor. If `nil`, the score is permanent, and in the case of a number, the score is deleted if it does not match for days (`wl-score-expiry-days`) from the date specified by this. The date is since Dec. 31, 1 B.C.
4. A type factor. This specifies the way the first factor matches. Available types depend on keys.

   From, Subject, References, Message-ID
   : For these keys in string, `r` and `R` (regexp), `s` and `S` (substring), `e` and `E` (exact match), as well as `f` and `F` (fuzzy) can be used. `R`, `S`, `E`, and `F` are case sensitive.

   Lines, Chars
   : For these keys, the following five numerical relative operators can be used: `<`, `>`, `=`, `>=`, `<=`.

   Followup This key matches `From` header, and scores all follow-ups to the message. For example, it would be useful for increasing scores for follow-ups to you own article.

   You can use the same types as `From` except for `f`. And a '`Followup`' entry is automatically appended to the score file.

   Thread This key scores (sub-)threads beginning with `Message-ID` *x*. A '`Thread`' entry is automatically appended for each article

that has *x* in the `References` header. You can make sure the whole thread including messages that does not have all ancestors `Message-ID` in `References` is scored.

You can use the same types as `References` except for `f`. And a '`Thread`' entry is automatically appended to the score file.

5. A factor for extension header. This is meaningful only if the key is `Extra`. This specifies headers to be matched other than standard headers like `Subject` and `From`. Note that you should specify the header in `elmo-msgdb-extra-fields` also. Therefore it does not work in folders where extension headers cannot be retrieved.

The sum of these scores *after all factors are applied* becomes the score of the message.

`mark` Messages with a score less than this value is marked as read. The default is `wl-summary-mark-below`.

`expunge` Messages with a score less than this value is deleted from the summary. The default is `wl-summary-expunge-below`.

`mark-and-expunge`
Both `mark` and `expunge` are applied, i.e. messages with a score less than this value is marked as read and deleted from the summary.

`target` Messages with a score greater than this value is attached with temp marks. The default is `wl-summary-target-above`.

`important`
Messages with a score greater than this value is attached with important marks. The default is `wl-summary-important-above`.

10.2.1 Caveats

Not to mention the `extra` key, if `lines` or `xref` keys are used, you need to set `elmo-msgdb-extra-fields`.

```
(setq elmo-msgdb-extra-fields '("lines" "xref"))
```

There are other restrictions as shown below:

- Because '`References`' field in the summary information contains only the last '`Message-ID`', `references` key matches the last one only.

Keys that can be seen by folder of types:

	chars	lines	xref	extra
localdir,localnews	Y	E	E	E
nntp (supporting xover)	Y	E	E	N
(otherwise)	N	E	E	E
imap4	Y	E	E	E
pop3	N	E	E	E

```
Y: can be seen
N: cannot be seen (ignored)
E: can be seen with elmo-msgdb-extra-fields settings
```

11 Address Book

With address book, you can utilize address completion, and you have summary displayed with nicknames.

11.1 Address book

The file '`~/.addresses`' is a simple address book for Wanderlust. Make address file '`~/.addresses`', and edit to suit your requirement.

The data written in '`~/.addresses`' are used for address completion under draft editing mode. Furthermore, they are used when showing names in summary display mode. You can safely skip this section, if you don't want to customize address completion and summary display. It is possible to add/change/remove addresses from '`~/.addresses`' in summary buffer after Wanderlust is invoked.

The format is very simple. Like this.

```
#
# Lines begin with '#' are comment.
# Empty lines are ignored
#
# Format of each line:
# email-address  "nickname "realname"
#
teranisi@gohome.org             "Yuuichi"      "Yuuichi Teranishi"
foo@bar.gohome.org              "Mr. Foo"    "John Foo"
bar@foo.gohome.org              "Mr. Bar"    "Michael Bar"
```

One line defines one persons description.

Actually, in default setup, *nickname* is used in summary-mode and *realname* is used in draft preparation mode. This behavior is better understood if you try it and confirmed the function first. You can write and try a small definition, so you will know the idea of the address book before writing a big one.

And, if MH alias file is specified in variable `wl-alias-file`, it is used as an address information in the draft preparation mode.

If variable `wl-use-ldap` is non-nil (initial setting is `nil`), address completion in draft mode uses LDAP information.

If you use LDAP, you have to set `wl-ldap-server`, `wl-ldap-port` and `wl-ldap-base` properly. If your emacs does not have LDAP feature as built-in feature (Currently only XEmacs can have built-in LDAP feature), you have to set command exec `PATH` to the program `ldapsearch`.

11.2 Address Manager

You can type *C-c C-a* to enter address manger mode. you can edit the address book and insert address to draft buffer.

11.2.1 Key Bindings

`t` Add '`To:`' mark.

`c` Add '`Cc:`' mark.

`b` Add '`Bcc:`' mark.

`u` Cancel the mark.

`x`

Insert '`To:`', '`Cc:`', or '`Bcc:`' marked addresses to draft buffer and quit address manager. When no draft buffer, make new draft with insert marked addresses. If no mark, quit address manager.

`q` Quit address manager.

`a` Add new entry.

`d` Delete entry.

`e` Edit entry.

12 Spam Filter

`wl-spam` provides an frontend to external spam filtering programs. You can register to or judge spam by the filtering program cooperateing with messages operations on Wanderlust.

12.1 Usage of Spam Filter

12.1.1 Initial Setting

To use `wl-spam`, write in '`~/.wl`' as follows:

```
;; Use 'bogofilter' as spam back end
;; Set 'scheme' here as the spam filter you will use.
;; See Section 12.2 [Spam Filter Processors], page 89 .
(setq elmo-spam-scheme 'bogofilter)
(require 'wl-spam)
```

12.1.2 spam mark

The spam mark ('`s`') will be provided as new temporary mark. Messages marked by this will be refiled into `wl-spam-folder` when the action is called for execution. Marked messages will be skipped by summary walking in ordinary way.

The spam mark is be put on by spam judgement described later, or by invoking *k m* at any time.

12.1.3 spam judgment

You can judge spam messages by following ways:

1. Make judgement on execution of auto-refile.

   Insert `wl-refile-guess-by-spam` to arbitrary position in `wl-auto-refile-guess-functions` as follows.

   ```
   (setq wl-auto-refile-guess-functions
         '(wl-refile-guess-by-rule
           wl-refile-guess-by-spam))
   ```

   In this example, judge spam if it could not decide refile destination by `wl-refile-rule-alist`.

2. Make judgement on entering the summary of specified folder.

   Specify the value of `wl-spam-auto-check-folder-regexp-list` as the list of regular expressions for folder names to be automatically judged by spam filter.

   ```
   (setq wl-spam-auto-check-folder-regexp-list '("\\+inbox"))
   ```

   In this example, judgement will be processed when you enter summary of the folder whose name contains '`+inbox`'.

3. Make judgement on splitting messages with `elmo-split`.

   It provides new function `spam-p` to be specified as '`CONDITION`' in `elmo-split-rule`. This function returns true when the message is judged as spam. See Section 13.5 [Split messages], page 100 .

   You can also process learning by the result of judgement. (You would better turn on this feature after learning to some extent)

Example follows:

```
(setq elmo-split-rule
      '(((spam-p) "+spam")
        ;; to learn by the judgement, use following instead
        ;((spam-p :register t) "+spam")
        (t "+inbox"))
```

12.1.4 spam learning

`wl-spam` automatically learn spam with refiling messages.

At first, `wl-spam` classifies the folders controlled by Wanderlust into following 4 domains by the class of containig messages

'`spam`' Folders containing messages judged as spam. (The folder specified by `wl-spam-folder`)

'`good`' Folders containing messages judged as non-spam.

'`undecide`'
Folders containing messages not yet judged. Folders without pre-distribution may belong to this domain e.g. '`+inbox`'. (specified by `wl-spam-undecided-folder-regexp-list`)

'`ignored`' Foldes have nothing to do with spam processing e.g. `wl-trash-folder` or `wl-draft-folder`. (specified by `wl-spam-ignored-folder-regexp-list`)

When you refile messages across different domains, it automatically learn messages as '`spam`' or '`non-spam`' according to domains it belongs before and after.

To put it concretely, it will learn by following rule:

'`undecide -> spam`'
learn as spam.

'`good -> spam`'
learn as spam and cancel previous study as non-spam.

'`undecide -> good`'
learn as non-spam.

'`spam -> good`'
learn as non-spam and cancel previous study as spam.

It do not learn anything in other cases.

12.1.5 Key Bindings

k m Put spam mark ('`s`') on current message.

k c Test current message and put spam mark if judged as spam. Remove spam mark if judged as non-spam.

k C Test messages with the mark in `wl-spam-auto-check-marks`, and put spam mark if judged as spam. If it is called with prefix argument, test all messages regardless of their marks.

`k s` Register current message as spam and put spam mark.

`k S` Register all messages in the folder as spam and put spam mark.

`k n` Register current message as non-spam and remove spam mark.

`k N` Register all messages in the folder as non-spam and remove spam mark.

`r k m` Put spam mark on messages in the specified region.

`r k c` Test messages in the specified region and put spam mark if judged as spam. Remove spam mark if judged as non-spam.

`r k s` Register messages in the specified region as spam and put spam mark.

`r k n` Register messages in the specified region as non-spam and remove spam mark.

`t k m` Put spam mark on messages which are the descendant of the current thread. With prefix argument, it affects on the all messages in the thread tree.

`t k c` Test messages which are the descendant of the current thread and put spam mark if judged as spam. Remove spam mark if judged as non-spam. With prefix argument, it affects on the all messages in the thread tree.

`t k s` Register messages which are the descendant of the current thread as spam and put spam mark. With prefix argument, it affects on the all messages in the thread tree.

`t k n` Register messages which are the descendant of the current thread as non-spam and remove spam mark. With prefix argument, it affects on the all messages in the thread tree.

`m k` Put spam mark (‘`s`’) on messages with the target mark ‘`*`’.

`m s` Register messages with the target mark ‘`*`’ as spam and put spam mark.

`m n` Register messages with the target mark ‘`*`’ as non-spam and remove spam mark.

12.1.6 Customizable Variables

`wl-spam-folder`
: Specify the name of destination folder for the spam messages. The initial setting is ‘`+spam`’.

`wl-spam-undecided-folder-regexp-list`
: Specify the list of regexp of folder names which contain messages not yet decided as spam or non-spam. The initial setting is `'("inbox")`.

`wl-spam-ignored-folder-regexp-list`
: The initial setting is as follows.

```
(list (regexp-opt (list wl-draft-folder
                        wl-trash-folder
                        wl-queue-folder)))
```

Folders of no effect against spam judgement, specified by the list of folder name regular expressions.

`wl-spam-auto-check-folder-regexp-list`
: Folders to make spam judgement on entering the summary of them, specified by the list of folder name regular expressions. The initial setting is `nil`.

`wl-spam-auto-check-marks`
: The initial setting is the following list:

```
(list wl-summary-new-uncached-mark
      wl-summary-new-cached-mark)
```

Messages with mark specified by this variable will be processed by whole-folder judgement including auto test by `wl-spam-auto-check-folder-regexp-list`. Persistent marks can be used in this method, but temporary marks cannot.

You can specify the list of marks as the initial setting, or you can specify follwing symbol:

`all` Process all messages regardless of persistent marks.

12.2 Supported Spam Filters

Supported spam filtering libraries are following ones.

12.2.1 bogofilter

bogofilter (http://bogofilter.sourceforge.net/) is a spam filter implemented by C language.

To use spam filter with bogofilter, write following setting in '`~/.wl`' or somewhere else.

```
(setq elmo-spam-scheme 'bogofilter)
```

12.2.1.1 Customizable Variables

`elmo-spam-bogofilter-program`
: The initial setting is '`bogofilter`'. Specify the name of executable of bogofilter. If the executable is not in your environmental variable `PATH`, you should set this by full path.

`elmo-spam-bogofilter-args`
: The initial setting is `nil`. Specify arguments to be supplied for bogofilter executable.

`elmo-spam-bogofilter-database-directory`
: Specify the directory for statistical database to be used. `nil` to use default directory ('`~/.bogofilter`'). The initial setting is `nil`.

`elmo-spam-bogofilter-max-messages-per-process`
: The initial setting is 30. This variable specifies the number of messages to be learned by one process.

`elmo-spam-bogofilter-debug`
: The initial setting is `nil`. If you specify non-nil, the output from `bogofilter` is stored in the buffer named `"*Debug ELMO SPAM Bogofilter*"`.

12.2.2 spamfilter.el

'spamfilter.el' (http://www.geocities.co.jp/SiliconValley-PaloAlto/7043/) is a spam filtering library implemented by Emacs Lisp.

Corresponding modules will be compiled/installed, if you have 'spamfilter.el' within load-path when you are to install wl. See Section 2.3 [Install], page 4 .

To use 'spamfilter.el', write following setting in '~/.wl' or somewhere else. (Of cource, you have to have settings for 'spamfilter.el' itself)

```
(setq elmo-spam-scheme 'spamfilter)
```

12.2.2.1 Customizable Variables

elmo-spam-spamfilter-corpus-filename
: The initial setting is '~/.elmo/.spamfilter'. It specifies the name of corpus file.

12.2.3 bsfilter

bsfilter (http://bsfilter.org/index-e.html) is a spam filter implemented by Ruby language.

To use spam filter with bsfilter, write following setting in '~/.wl' or somewhere else.

```
(setq elmo-spam-scheme 'bsfilter)
```

12.2.3.1 Customizable Variables

elmo-spam-bsfilter-program
: The initial setting is 'bsfilter'. Specify the name of executable of bsfilter. If the executable is not in your environmental variable PATH, you should set this by full path.

elmo-spam-bsfilter-args
: The initial setting is nil. Specify arguments to be supplied for bsfilter executable.

elmo-spam-bsfilter-database-directory
: Specify the directory for statistical database to be used. nil to use default directory ('~/.bsfilter'). The initial setting is nil.

elmo-spam-bsfilter-debug
: The initial setting is nil. If you specify non-nil, the output from bsfilter is stored in the buffer named "*Debug ELMO Bsfilter*".

elmo-spam-bsfilter-shell-program
: The initial setting is 'ruby'. Specify the shell to execute bsfilter. If the shell is not in your environmental variable PATH, you should set this by full path.

elmo-spam-bsfilter-shell-switch
: The initial setting is nil. Specify options to give to the shell executing bsfilter.

elmo-spam-bsfilter-update-switch
: The initial setting is "--auto-update". Specify options to give to bsfilter for learning messages.

12.2.4 SpamAssassin

SpamAssassin (http://spamassassin.org/) is one of the most popular spam filtering program implemented on Perl. SpamAssassin attempts to identify spam using text analysis and several internet-based realtime blacklists. SpamAssassin also uses a Bayesian learning filter which enables more accurate spam filtering.

To use '`SpamAssassin`' on Wanderlust, write following setting in '`~/.wl`' or somewhere else. (Of course, you have to install SpamAssassin beforehand.)

```
(setq elmo-spam-scheme 'sa)
```

12.2.4.1 Customize Variables

`elmo-spam-spamassassin-program`
: The initial setting is '`spamassassin`'. Specify the name of executable `spamassassin`. If the executable is not in your environmental variable `PATH`, you should set this by full path.

`elmo-spam-spamassassin-learn-program`
: The initial setting is '`sa-learn`'. Specify the name of the SpamAssassin's Bayesian filtering learner program, `sa-learn`. If the executable is not in your environmental variable `PATH`, you should set this by full path.

`elmo-spam-spamassassin-program-arguments`
: The initial setting is `'("-e")`. Specify the arguments to be supplied for `spamassassin` executable. You have to specify the argument to exit the program with an error exit code when the result is spam. For example, if you want to use `spamc` instead of `spamassassin`, you should specify `'("-c")`.

`elmo-spam-spamassassin-learn-program-arguments`
: The initial setting is `nil`. Specify the arguments to be supplied for `sa-learn`.

`elmo-spamassassin-debug`
: The initial setting is `nil`. If you specify `t`, the output from `spamassassin` is stored in the buffer named `"*Debug ELMO SpamAssassin*"`.

12.2.5 SpamOracle

SpamOracle (http://pauillac.inria.fr/~xleroy/software.html#spamoracle) is a spam filter implemented by Objective Caml language.

To use spam filter with '`spamoracle`', write following setting in '`~/.wl`' or somewhere else. (Of course, you have to install SpamOracle beforehand.)

```
(setq elmo-spam-scheme 'spamoracle)
```

12.2.5.1 Customizable Variables

`elmo-spam-spamoracle-program`
: The initial setting is '`spamoracle`'. Specify the name of executable of spamoracle. If the executable is not in your environmental variable `PATH`, you should set this by full path.

`elmo-spam-spamoracle-config-filename`
: Specify the name of config file. `nil` to use default file ('`~/.spamoracle.conf`'). The initial setting is `nil`.

`elmo-spam-spamoracle-database-filename`
: The initial setting is '`~/.elmo/.spamoracle.db`'. It specifies the name of database file.

`elmo-spam-spamoracle-spam-header-regexp`
: The initial setting is `"^X-Spam: yes;"`. It specifies the regular expression of the header that indicates spam mail. Use this setting when you change the `spam_header` parameter in the config file.

12.2.6 Regular Expressions Header Matching

Examine if regular expression matches corresponding field in message heaeder, and decide spam or not. To use this backend, add following setting to '`~/.wl`'.

```
(setq elmo-spam-scheme 'header)
```

If you want to check fields not included in the default overview information, add one into `elmo-msgdb-extra-fields`. Then it will do examination by the overview information and avoid loading whole message body as far as possible.

12.2.6.1 Customize Variables

`elmo-spam-header-good-alist`
: The initial setting is the following list:

```
'(("X-Spam-Flag" . "No"))
```

Specify a list of regular expressions to match with header field name for making non-spam decision. It takes precedence over `elmo-spam-header-spam-alist`.

`elmo-spam-header-spam-alist`
: The initial setting is the following list:

```
'(("X-Spam-Flag" . "Yes"))
```

Specify a list of regular expressions to match with header field name for making spam decision.

13 Advanced Issues

13.1 Living with other packages

Examples with other packages.

13.1.1 imput

Place '`util/im-wl.el`' on the `load-path` and do the following settings.

```
(autoload 'wl-draft-send-with-imput-async "im-wl")
(setq wl-draft-send-function 'wl-draft-send-with-imput-async)
```

13.1.2 bbdb.el

To use The Insidious Big Brother Database (http://bbdb.sourceforge.net/) with Wanderlust, place '`util/bbdb-wl.el`' on the `load-path` and do the following settings.

But, '`util/bbdb-wl.el`' is not compatible with BBDB 3.x (http://savannah.nongnu.org/projects/bbdb For BBDB 3.x, BBDBV3-Wl (https://gna.org/projects/bbdbv3-wl/) may be useful.

If BBDB is on the `load-path` at the installation, '`bbdb-wl.el`' is byte-compiled and installed. See Section 2.3 [Install], page 4 .

```
(require 'bbdb-wl)

(bbdb-wl-setup)
;; enable pop-ups
(setq bbdb-use-pop-up t)
;; auto collection
(setq bbdb/mail-auto-create-p t)
;; exceptional folders against auto collection
(setq bbdb-wl-ignore-folder-regexp "^@")
(setq signature-use-bbdb t)
(setq bbdb-north-american-phone-numbers-p nil)
;; shows the name of bbdb in the summary :-)
(setq wl-summary-from-function 'bbdb-wl-from-func)
;; automatically add mailing list fields
(add-hook 'bbdb-notice-hook 'bbdb-auto-notes-hook)
(setq bbdb-auto-notes-alist '(("X-ML-Name" (".*$" ML 0))))
```

You can complete address with BBDB by *M-TAB* in draft buffer.

13.1.3 lsdb.el

The following is an example setting to use The Lovely Sister Database (http://sourceforge.jp/projects/lsdb/) with Wanderlust.

```
(require 'lsdb)
(lsdb-wl-insinuate)
(add-hook 'wl-draft-mode-hook
          (lambda ()
            (define-key wl-draft-mode-map "\M-\t" 'lsdb-complete-name)))
```

In this example, bind *M-TAB* to `lsdb-complete-name` (complete address with LSDB).

13.1.4 sc.el(supercite), sc-register.el

The same setting as usual mailers should be OK. The following is an example of settings:

```
(autoload 'sc-cite-original "supercite" nil t)
(add-hook 'mail-citation-hook 'sc-cite-original)
```

13.1.5 mu-cite.el

The same setting as usual mailers should be OK. The following is an example of settings.

If you use mu-cite version 8.0 or earlier:

```
(autoload 'mu-cite/cite-original "mu-cite" nil t)
(setq mail-citation-hook 'mu-cite/cite-original)
```

If you use mu-cite version 8.1 or later:

```
(autoload 'mu-cite-original "mu-cite" nil t)
(add-hook 'mail-citation-hook (function mu-cite-original))
```

13.1.6 x-face

If you have installed one of the following, you can decode '`X-Face:`' field in message buffer and you will see face image.

If there is an encoded X-Face string in a file '`~/.xface`' (the value of the variable `wl-x-face-file`), it is inserted as a '`X-Face:`' field in the draft buffer (if `wl-auto-insert-x-face` is non-nil).

13.1.6.1 x-face-xmas (for XEmacs)

If you use '`x-face-xmas.el`' in x-face (ftp://jpl.org/pub/elisp/) 1.3.6.13 or later, do the following:

```
(autoload 'x-face-xmas-wl-display-x-face "x-face")
(setq wl-highlight-x-face-function 'x-face-xmas-wl-display-x-face)
```

13.1.6.2 x-face-mule (for Emacs)

If you use '`x-face-mule.el`' in bitmap-mule (ftp://ftp.jpl.org/pub/elisp/bitmap/) 8.0 or later, do the following:

```
(autoload 'x-face-decode-message-header "x-face-mule")
(setq wl-highlight-x-face-function 'x-face-decode-message-header)
```

13.1.6.3 x-face-e21 (for Emacs 21.x)

With Emacs 21.x, you can use '`x-face-e21.el`' (ftp://jpl.org/pub/elisp/) instead of '`x-face-mule.el`' to display X-Face. In this case, bitmap-mule is not required. Do as follows:

```
(autoload 'x-face-decode-message-header "x-face-e21")
(setq wl-highlight-x-face-function 'x-face-decode-message-header)
```

13.1.7 dired-dd(Dired-DragDrop)

If you embed '`dired-dd-mime.el`' in the dired-dd package, you can compose multi-part by simple Drag-and-Drop from dired to the draft buffer being edited in GNU Emacs (this feature is not Wanderlust specific, but general-purpose for SEMI).

```
;; dired-dd: http://www.asahi-net.or.jp/~pi9s-nnb/dired-dd-home.html
(add-hook
 'dired-load-hook
 (function
  (lambda ()
    (load "dired-x")
    ;; Set dired-x variables here.
    ;; To and flo...
    (if window-system
        (progn (require 'dired-dd)
               (require 'dired-dd-mime))))))
```

13.1.8 mhc.el

Message Harmonized Calendaring system (http://www.quickhack.net/mhc/)

By using MHC, you can make a calendar from the messages.

For mhc-0.25:

```
(setq mhc-mailer-package 'wl)
(autoload 'mhc-mode "mhc" nil t)
(add-hook 'wl-summary-mode-hook 'mhc-mode)
(add-hook 'wl-folder-mode-hook 'mhc-mode)
```

For mhc-current:

```
(autoload 'mhc-wl-setup "mhc-wl")
(add-hook 'wl-init-hook 'mhc-wl-setup)
```

13.1.9 wl-addrbook.el

Addrbook of Mew (http://www.mew.org/)

Place '`util/wl-addrbook.el`' and '`util/wl-complete.el`' on the `load-path` and do the following settings.

```
(require 'wl-addrbook)
(wl-addrbook-setup)
```

13.1.10 mime-w3m.el

You can display html part by using '`mime-w3m.el`' distributed with emacs-w3m (http://emacs-w3m.namazu.org/). You can find the usage in comment region at the head of '`mime-w3m.el`'. If you use SEMI-EPG, no additional setting is needed.

13.2 Highlights

13.2.1 Customizable Variables

`wl-summary-highlight`
: The initial setting is `t`. If non-nil, the summary is highlighted.

`wl-highlight-max-summary-lines`
: The initial setting is 10000. The summary is not highlighted if it has more lines than this value.

`wl-summary-highlight-partial-threshold`
: The initial setting is 1000. This is a threshold whether the whole summary is highlighted. If there are more lines of messages in the summary, it is partially highlighted.

`wl-summary-partial-highlight-above-lines`
: The initial setting is 30. If there are more lines of messages than `wl-summary-highlight-partial-threshold` in the summary, messages after the point that is the same number of lines as this value above the cursor line are highlighted partially. (If this value is `nil`, the last same number of lines as the value of `wl-summary-highlight-partial-threshold` are highlighted.)

`wl-highlight-body-too`
: The initial setting is `t`. If non-nil, bodies of drafts and messages are also highlighted.

`wl-highlight-message-header-alist`
: When highlighting headers of drafts and messages, this variable specifies which faces are allocated to important (`wl-highlight-message-important-header-contents`), secondly important (`wl-highlight-message-important-header-contents2`), and unimportant (`wl-highlight-message-unimportant-header-contents`) message headers. Similarly, it can be used for allocating arbitrary faces to arbitrary regular expressions.

`wl-highlight-citation-prefix-regexp`
: Specifies a regular expression to which quoted lines in bodies of drafts and messages match. Bodies matching to this regular expression are highlighted by the faces specified by (`wl-highlight-message-cited-text-*`).

`wl-highlight-highlight-citation-too`
: The initial setting is `nil`. If non-nil, the quoting regular expression itself given by `wl-highlight-citation-prefix-regexp` is also highlighted.

`wl-highlight-citation-header-regexp`
: Specifies a regular expression that denotes beginning of quotation. Bodies matching to this regular expression are highlighted by the face specified by `wl-highlight-message-headers`.

`wl-highlight-max-header-size`
: The initial setting is `nil`. If a header size is larger than this value, it will not be highlighted. If `nil`, always highlighted (ignore header size).

`wl-highlight-max-message-size`
: The initial setting is 10000. If a message is larger than this value, it will not be highlighted. With this variable, highlight is suppressed for uuencode or huge digest messages.

`wl-highlight-signature-separator`
: Specifies regular expressions that denotes the boundary of a signature. It can be a regular expression, or a list of ones. Messages after the place that matches this regular expression are highlighted by the face specified by `wl-highlight-message-signature`.

`wl-max-signature-size`
: The initial setting is 400. This is the largest size for a signature to be highlighted.

`wl-use-highlight-mouse-line`
: The initial setting is `t`. If non-nil, the line pointed by the mouse is highlighted in the folder mode, summary mode, and the like.

13.2.2 Setting Colors and Fonts of the Characters

If you want to change colors or fonts of the characters, you need to modify faces defined in Wanderlust. Use `set-face-font` if you want to change fonts, and `set-face-foreground` for colors, and so on. You cannot write face settings in '`.emacs`'; write in '`~/.wl`'.

For example, if you want to change the color for signatures to yellow, write

```
(set-face-foreground 'wl-highlight-message-signature "yellow")
```

in '`~/.wl`'.

Faces defined in Wanderlust:

`wl-highlight-message-headers`
: The face for field names of message headers.

`wl-highlight-message-header-contents`
: The face for field bodies of message headers.

`wl-highlight-message-important-header-contents`
: The face for important parts of message headers. Per default, this face is used for a body of '`Subject:`' field. You can change its value by editing `wl-highlight-message-header-alist`.

`wl-highlight-message-important-header-contents2`
: The face for secondly important parts of message headers. Per default, this face is used for bodies of '`From:`' and '`To:`' fields. You can change its value by editing `wl-highlight-message-header-alist`.

`wl-highlight-message-unimportant-header-contents`
: The face for unimportant parts of message headers. Per default, this face is used for bodies of '`X-`' fields '`User-Agent:`' fields. You can change its value by editing `wl-highlight-message-header-alist`.

`wl-highlight-message-citation-header`
: The face for headers of quoted messages.

`wl-highlight-message-cited-text-*`
: The face for texts of quoted messages. The last '`*`' is a *single figure* so that 10 different colors can be used according to citation levels.

`wl-highlight-message-signature`
: The face for signatures of messages. The initial settings are '`khaki`' for light background colors, and '`DarkSlateBlue`' for dark background colors.

`wl-highlight-header-separator-face`
: The face for header separators of draft messages.

`wl-highlight-summary-important-face`
: The face for message lines with important marks in the summary.

`wl-highlight-summary-new-face`
: The face for message lines with new marks in the summary.

`wl-highlight-summary-displaying-face`
: The face for the message line that is currently displayed. This face is overlaid.

`wl-highlight-thread-indent-face`
: The face for the threads that is currently displayed.

`wl-highlight-summary-unread-face`
: The face for message lines with unread marks in the summary.

`wl-highlight-summary-deleted-face`
: The face for message lines with delete marks in the summary.

`wl-highlight-summary-refiled-face`
: The face for message lines with re-file marks in the summary.

`wl-highlight-refile-destination-face`
: The face for re-file information part of message lines with re-file marks in the summary.

`wl-highlight-summary-copied-face`
: The face for message lines with copy marks in the summary.

`wl-highlight-summary-target-face`
: The face for message lines with target marks '`*`' in the summary.

`wl-highlight-summary-thread-top-face`
: The face for message lines that are on the top of the thread in the summary.

`wl-highlight-summary-normal-face`
: The face for message lines that are not on top of the thread in the summary.

`wl-highlight-folder-unknown-face`
: The face for folders that are not known to have how many unsync messages in the folder mode.

`wl-highlight-folder-zero-face`
: The face for folders that have no unsync messages in the folder mode.

`wl-highlight-folder-few-face`
: The face for folders that have some unsync messages in the folder mode.

`wl-highlight-folder-many-face`
: The face for folders that have many unsync messages in the folder mode. The boundary between 'some' and 'many' is specified by the variable `wl-folder-many-unsync-threshold`.

`wl-highlight-folder-unread-face`
: The face for folders that have no unsync but unread messages in the folder mode.

`wl-highlight-folder-killed-face`
: The face for folders that are deleted from the access group in the folder mode.

`wl-highlight-folder-opened-face`
: The face for open groups in the folder mode. It is meaningful when `wl-highlight-folder-by-numbers` is `nil` or a *number*.

`wl-highlight-folder-closed-face`
: The face for close groups in the folder mode. It is meaningful when `wl-highlight-folder-by-numbers` is `nil` or a *number*.

`wl-highlight-folder-path-face`
: The face for the path to the currently selected folder in the folder mode.

`wl-highlight-logo-face`
: The face for logo in the demo.

`wl-highlight-demo-face`
: The face for strings (for example, a version number) in the demo.

13.3 Notify Mail arrival

Following setting is to notify mail arrival of '`%inbox`' by the indicator on the modeline

```
(setq wl-biff-check-folder-list '("%inbox"))
```

13.3.1 Customizable Variables

`wl-biff-check-folder-list`
: The initial setting is `nil`. This is the list of folders to check mail arrival. If `nil`, wl doesn't check mail arrival.

`wl-biff-check-interval`
: The initial setting is 40 (in seconds). Check mail arrival in this period.

`wl-biff-use-idle-timer`
: The initial setting is `nil`. If it is `nil`, check mail arrival when the time specified by `wl-biff-check-interval` has passed. If it is non-nil, check mail arrival when idling time exceeds `wl-biff-check-interval`.

`wl-biff-notify-hook`
: This hook is run at the arrival of new mail. To beep with mail arrival(initial setting), set as follows.

  ```
  (setq wl-biff-notify-hook '(ding))
  ```

  For silence, set to `nil`.

13.4 Manage Passwords

If you input passwords to connect servers, they are stored in the variable `elmo-passwd-alist` per connection. You should be careful that others might read your passwords if they can touch your Emacs, since encoded plain passwords are there.

If you invoke *`M-x elmo-passwd-alist-save`* while you have stored passwords, then they are saved on the file, and it will save you to input passwords. In this case, the risk that

someone reads your keystroke might decrease, but please not that plain passwords are stored on a file. You should treat them very carefully. To remove saved passwords on file, invoke *M-x elmo-passwd-alist-clear* and then *M-x elmo-passwd-alist-save*.

`elmo-passwd-alist-file-name`
: The initial setting is '`passwd`'. This is the name of the file in which passwords are saved. `elmo-passwd-alist-save` saves current passwords to the file.

`elmo-passwd-life-time`
: The initial setting is `nil`. If the value is some number, timer is set to remove password entry after `elmo-passwd-life-time` seconds since you input the password. `nil` means never to remove passwords.

13.5 Message splitting

You can use `elmo-split` to split message in folder specified by the variable `elmo-split-folder` a la `procmail` according to some specified rules. To use this feature, set as follows in your '`~/.emacs`' at first.

```
(autoload 'elmo-split "elmo-split" "Split messages on the folder." t)
```

Set source folder like following.

```
(setq elmo-split-folder "%inbox")
```

And specify the rule in the variable `elmo-split-rule` (its format will be is described below). Then you can invoke *M-x elmo-split* to split messages according to `elmo-split-rule`. On the other hand, invoke *C-u M-x elmo-split* to do a rehearsal and show result (do not split actually).

We will describe how to specify the rule. First of all, see following example, please.

```
(setq elmo-split-rule
      ;; Store messages from spammers into '+junk'
      '(((or (address-equal from "i.am@spammer")
     (address-equal from "dull-work@dull-boy")
     (address-equal from "death-march@software")
     (address-equal from "ares@aon.at")
     (address-equal from "get-money@richman"))
 "+junk")
;; Store messages from mule mailing list into '%mule'
((equal x-ml-name "mule") "%mule")
;; Store messages from wanderlust mailing list into '%wanderlust'
;; and continue evaluating following rules
((equal x-ml-name "wanderlust") "%wanderlust" continue)
;; Store messages from Yahoo user into '+yahoo-{username}'
((match from "\\(.*\\)@yahoo\\.com")
 "+yahoo-\\1")
;; Store unmatched mails into '+inbox'
(t "+inbox")))
```

The basic unit of the rule is a combination like

```
(`CONDITION' `ACTION' [continue])
```

If '`CONDITION`' is true, '`ACTION`' is performed. The 1st element '`CONDITION`' is a condition represented by a balanced expression (sexp). Its grammar will be explained below. The 2nd element '`ACTION`' is the name of the folder to split messages into, or a symbol. When the 3rd element `continue` is specified as symbol, evaluating rules is not stopped even when the condition is satisfied.

The grammar for '`CONDITION`' is as follows. See example above to learn how to write the condition practically.

1. Functions which accept arguments '`FIELD-NAME`' and '`VALUE`'. ('`FIELD-NAME`' is a symbol that describes the field name)

   `equal` True if the field value equals to '`VALUE`'. Case of the letters are ignored.

   `match` True if the field value matches to VALUE. '`VALUE`' can contain `\&` and `\N` which will substitute from matching `\(\)` patterns in the previous '`VALUE`'.

   `address-equal`
   True if one of the addresses in the field equals to '`VALUE`'. Case of the letters are ignored.

   `address-match`
   True if one of the addresses in the field matches to '`VALUE`'. '`VALUE`' can contain `\&` and `\N` which will substitute from matching `\(\)` patterns in the previous '`VALUE`'.

2. Functions which accept an integer argument ('`SIZE`').

   `<` True if the size of the message is less than '`SIZE`'.

   `>` True if the size of the message is greater than '`SIZE`'.

3. Functions which accept any number of arguments.

   `or` True if one of the argument returns true.

   `and` True if all of the arguments return true.

4. A symbol.

   When a symbol is specified, it is evaluated.

You can specify followings as 2nd '`ACTION`'.

1. folder name

   If some string is specified, it will be regarded as the destination folder, and the message will be appended to it.

2. '`delete`'

   If the symbol '`delete`' is specified, delete the substance of the message in `elmo-split-folder`

3. '`noop`'

   If the symbol '`noop`' is specified, do nothing on the message and keep it as it is.

4. function

   If some function is specified, execute it.

If the message passes all rules, it will be dealed along '`ACTION`' specified by `elmo-split-default-action`.

13.6 Batch Processing

You can request wanderlust to do some job on the command line. For now, you can invoke prefetching new messages in specified folders.

Specify target folders in `wl-batch-prefetch-folder-list` then invoke as follows to execute prefetching:

```
% emacs -batch -l wl-batch -f wl-batch-prefetch
```

13.6.1 Customize Variables

`wl-batch-prefetch-folder-list`
: Target folders of prefetching by `wl-batch-prefetch`, specified as a list of folder names.

13.7 Advanced Settings

13.7.1 Draft for Replay

If you type `a` in the Summary Buffer, a draft for reply is prepared. The addressee for the draft is decided by following rules.

For example, you can set as follows:

```
(setq wl-draft-reply-without-argument-list
      '(("Mail-Followup-To" . (("Mail-Followup-To") nil ("Newsgroups")))
        ("Followup-To" . (nil nil ("Followup-To")))
        (("X-ML-Name" "Reply-To") . (("Reply-To") nil nil))
        ("From" . (("From") ("To" "Cc") ("Newsgroups")))))
```

Where each element of the list `wl-draft-reply-without-argument-list` is in the form

```
(key . (to-list cc-list newsgroup-list))
```

and if the field designated by '`key`' exist in the parent message, parent's field values designated by '`to-list`' are copied to '`To:`' in the draft. Similarly, parent's fields designated by '`cc-list`' and '`newsgroup-list`' are copied to '`Cc:`' and '`Newsgroups:`' in the draft respectively.

Examples:

```
("Mail-Followup-To" . (("Mail-Followup-To") nil ("Newsgroups")))
```

Match if the parent has '`Mail-Followup-To`' field. The components of parent's '`Mail-Followup-To`' and '`Newsgroups`' fields are copied to '`To`' and '`Newsgroups`' in the draft respectively.

```
(("X-ML-Name" "Reply-To") . (("Reply-To") nil nil))
```

Match if the parent has both '`X-ML-Name`' and '`Reply-To`' fields. Parent's '`Reply-To`' is copied to '`To`' in the draft.

```
("From" . (("From") ("To" "Cc") ("Newsgroups")))
```

Copy parent's '`From`' to '`To`' in the draft, parent's '`To`' and '`Cc`' to '`Cc`', parent's '`Newsgroups`' to '`Newsgroups`' respectively.

These are evaluated in order and first matched one is used.

Moreover, the behavior of `a` with prefix argument can be directed by `wl-draft-reply-with-argument-list` as well.

By the way, you can use some function (will be evaluated in the parent message buffer) in the place of '`key`' or '`to-list`' etc.

If you want to write a rule for replying to message written by yourself, specify function `wl-draft-self-reply-p` as '`key`'.

If you only want to reply to mailing lists in `wl-subscribed-mailing-list` if the parent has some of them, set as follows:

```
(defun wl-mailing-list-addresses ()
  (let (list-addrs)
    (dolist (to (mapcar
 (lambda (addr)
   (nth 1 (std11-extract-address-components addr)))
 (wl-parse-addresses
  (wl-concat-list
   (elmo-multiple-fields-body-list (list "To" "Cc"))
   ","))))
      (when (elmo-string-matched-member to wl-subscribed-mailing-list t)
(setq list-addrs (cons to list-addrs))))
    (nreverse list-addrs)))

(setq wl-draft-reply-with-argument-list
      '((wl-mailing-list-addresses . (wl-mailing-list-addresses nil nil))
("Reply-To" . (("Reply-To") nil nil))
("Mail-Reply-To" . (("Mail-Reply-To") nil nil))
("From" . (("From") nil nil))))
```

13.7.2 Appearance of Threads

```
  389  09/18(Fri)01:07 [ Teranishi         ] wl-0.6.3
  390  09/18(Fri)07:25 +-[ Tsumura-san       ]
  391  09/18(Fri)19:24 +-[ Murata-san        ]
  392  09/20(Sun)21:49 +-[ Okunishi-san      ]
  396  09/20(Sun)22:11 | +-[ Tsumura-san       ]
  398  09/21(Mon)00:17 |   +-[ Tsumura-san       ]
  408  09/21(Mon)22:37 |     +-[ Okunishi-san      ]
  411  09/22(Tue)01:34 |       +-[ Tsumura-san       ]
  412  09/22(Tue)09:28 |       +-[ Teranishi         ]
  415  09/22(Tue)11:52 |         +-[ Tsumura-san       ]
  416  09/22(Tue)12:38 |           +-[ Teranishi         ]
  395  09/20(Sun)21:49 +-[ Okunishi-san      ]
  397  09/21(Mon)00:15 +-[ Okunishi-san      ]
```

Settings to make appearance of threads like shown above:

```
(setq wl-thread-indent-level 2)
(setq wl-thread-have-younger-brother-str "+")
(setq wl-thread-youngest-child-str       "+")
(setq wl-thread-vertical-str             "|")
(setq wl-thread-horizontal-str           "-")
(setq wl-thread-space-str                " ")
```

If you do not want to see branches, do the following:

```
(setq wl-thread-indent-level 2)
(setq wl-thread-have-younger-brother-str " ")
(setq wl-thread-youngest-child-str       " ")
(setq wl-thread-vertical-str             " ")
(setq wl-thread-horizontal-str           " ")
(setq wl-thread-space-str                " ")
```

13.7.3 User-Agent Field

If you are eccentric enough to elaborate '`X-Mailer:`' or '`User-Agent:`' fields, define a function that generate appropriate strings as you like, and set it to variable `wl-generate-mailer-string-function`.

If you do not want verbose '`User-Agent:`' field, do the following:

```
(setq wl-generate-mailer-string-function
      'wl-generate-user-agent-string-1)
```

The following is a example:

```
(setq wl-generate-mailer-string-function nil)
(setq wl-draft-additional-header-alist
      (list
       (cons 'X-Mailer (lambda () (product-string-1 'wl-version)))))
```

13.8 Customizable Variables

Customizable variables that have not been described yet:

`wl-default-folder`
: The initial setting is '`%inbox`'. This is the default value for moving to a folder and the like.

`wl-draft-folder`
: The initial setting is '`+draft`'. It is the folder to which drafts are saved. It must be a writable folder. You can set IMAP remote folder, Maildir and so on. Note that variable settings applied by `wl-draft-config-exec` is saved under `elmo-msgdb-directory`. That is to say, if you specified remote folder as `wl-draft-folder`, variable settings which are applied by `wl-draft-config-exec` before saving the draft will not affect on the draft buffer on another host by invoking `wl-summary-reedit`.

`wl-trash-folder`
: The initial setting is '`+trash`'. It is the wastebasket folder. If you changed this variable, you had better restart Wanderlust.

`wl-interactive-exit`
: The initial setting is `t`. If non-nil, you are asked for confirmation when Wanderlust terminates.

`wl-interactive-send`
: The initial setting is `t`. If non-nil, you are asked for confirmation when mail is sent.

`wl-default-sync-range`
: The initial setting is '`update`'. Default update range of the summary. You can specify '`all`', '`update`', '`rescan`' or '`no-sync`'. See description of `wl-summary-sync` for the meaning of ranges.

`wl-folder-sync-range-alist`
: The initial setting is the alist shown below:

```
(("^&.*$" . "all")
 ("^\\+draft$\\|^\\+queue$" . "all"))
```

This is an associative list of regular expressions of folder names and update range of the summary. Update range is one of the '`all`', '`update`', '`rescan`' or '`no-sync`'. If the folder do not match any of them, the value of `wl-default-sync-range` is used ('`update`' by default). See description of `wl-summary-sync` for the meaning of ranges.

`wl-ask-range`
: The initial setting is `t`. If `nil`, the value of `wl-folder-sync-range-alist` is used for updating the summary when you changed folders.

`wl-mime-charset`
: The initial setting is `x-ctext`. This is the MIME charset for messages that are not MIME (e.g. without '`Content-Type:`'). This value also used as default charset for summary. (If you want to share Summary on Nemacs and other Emacsen, set this value as `iso-2022-jp`.)

`wl-highlight-folder-with-icon`
: This is meaningful for XEmacs or Emacs 21.. The initial setting depends on Emacsen (`t` for XEmacs or Emacs 21 with icons).

`wl-strict-diff-folders`
: This is a list of regular expressions of folders. Unread messages are checked, for example when you press `s` in the folder mode, usually in a brief way (rapidly processed but not accurate). The folders matching this variable are seriously checked. You may want to set this variable so as to match conditional filter folders for IMAP4 folders. The initial setting is `nil`.

`wl-folder-use-server-diff`
: When unread messages are checked, for example when you press `s` in the folder mode, usually (the number of messages on the server) − (the number of local messages) will be the number of unread messages. However, if this variable is non-nil, the number of unread messages on the server is checked. This affects IMAP4 folders only, but IMAP4 folders in mail boxes matching `elmo-imap4-disuse-server-flag-mailbox-regexp` are not checked for the number of un-

read messages on the server, even if they matches this variable. The initial setting is `t`.

`wl-auto-check-folder-name`
: The initial setting is `nil`. You can specify a folder or a group which is checked for unread message at the start. You can also specify a list of folders (groups) to be checked. If the value is `nil`, whole Desktop is checked at the start. If it is `none`, no folders are checked.

`wl-auto-uncheck-folder-list`
: The initial setting is the list shown below:

```
("\\$.*")
```

You can set a list of regular expressions to specify folders which are not automatically checked even if they are included in some groups assigned by `wl-auto-check-folder-name`.

`wl-auto-check-folder-list`
: The initial setting is `nil`. You can set a list of regular expressions to specify exceptions for `wl-auto-uncheck-folder-list`.

`wl-no-save-folder-list`
: The initial setting is the list shown below:

```
("^/.*$")
```

This is a list of regular expressions of folders not to be saved.

`wl-save-folder-list`
: The initial setting is `nil`. This is a list of regular expressions of folders to be saved. This takes precedence over `wl-no-save-folder-list`.

`wl-folder-mime-charset-alist`
: The initial setting is the alist shown below:

```
(("^-alt\\.chinese" . big5)
 ("^-relcom\\." . koi8-r)
 ("^-tw\\." . big5)
 ("^-han\\." . euc-kr))
```

This is an associative list of regular expressions of folder names and MIME charsets. If a folder do not match, `wl-mime-charset` is used.

`wl-folder-init-load-access-folders`
: The initial setting is `nil`. This is a list of access groups to be loaded specifically at the start. If it is `nil`, `wl-folder-init-no-load-access-folders` is referred.

`wl-folder-init-no-load-access-folders`
: The initial setting is `nil`. This is a list of access groups not to be loaded specifically at the start. It is ignored if `wl-folder-init-load-access-folders` is non-nil.

`wl-dispose-folder-alist`
: The initial setting is the alist shown below:

```
(("^-" . remove)
 ("^@" . remove))
```

This list determines disposition of messages with disposal marks. Each item in the list is a folder and destination; you can specify any one of the following in the place of destination:

```
remove or null : deletes the messages instantly.
string         : moves the messages to the specific folder.
trash or others  : moves the messages to wl-trash-folder.
```

`wl-x-face-file`
The initial setting is '`~/.xface`'. The name of the file that contains encoded X-Face strings. See Section 13.1.6.2 [x-face-mule], page 94 .

`wl-demo-display-logo`
If non-nil, bitmap image is shown on the opening demo. If you set `xpm` or `xbm`, (if possible) display selected image type logo.

`elmo-use-database`
This is meaningful for XEmacs only. The initial setting depends on XEmacs (`t` for XEmacs with dbm). If non-nil, Message-ID is controlled by dbm.

`elmo-nntp-list-folders-use-cache`
The initial setting is 600 (in seconds). This is period in seconds during which results of '`list`' and '`list active`' in NNTP are cached. If it is `nil`, they are not cached.

`elmo-nntp-max-number-precedes-list-active`
The initial setting is `nil`. If non-nil, the number of article obtained by '`list active`' in NNTP are used as the maximum article number of the folder. Set this to `t` if you are using for example INN 2.3 as an NNTP server, and if the number of read messages is not correct.

`elmo-nntp-default-use-listgroup`
The initial setting is `t`. If non-nil, '`listgroup`' is used for checking the total number of articles. If it is `nil`, '`group`' is used. In the latter case, the processing will be a little faster at the sacrifice of accuracy.

`elmo-pop3-send-command-synchronously`
The initial setting is `nil`. If non-nil, POP3 commands are issued synchronously. Some implementation of POP3 server fails to get summary information without this setting. You may have to set this variable to `t`, if the process hangs while looking up POP3.

`elmo-dop-flush-confirm`
The initial setting is `t`. If non-nil, you are asked for confirmation if accumulated off-line operations are executed.

`elmo-network-session-idle-timeout`
The initial setting is `nil`. Idle timeout of the network cache. Specified in seconds. If elapsed time since last access is larger than this value, cached session is not reused. If nil, network cache is reused.

13.9 Hooks

(Not yet written)

14 Switch from older version of Wanderlust

This chapter explains the important thing for the upgrade, or migration from the previous version. It includes the changes of the setup, limitations etc.

14.1 Migration from prior to the version 2.12.0

14.1.1 The conversion of msgdb

From version 2.12.0 on, the structure of msgdb is changed. The msgdb for newly created folder will use this new format when created and saved. But by writing following line, you may use the old format of the msgdb as it was.

```
(setq elmo-msgdb-default-type 'legacy)
```

With the default setup, the old msgdb format is converted to the new format automatically. You may change this behavior by writing following lines in '`~/.wl`'.

```
;; If the format of msgdb is different from elmo-msgdb-default-type,
;; the format will be converted automatically when
;; the msgdb is being loaded (default).
(setq elmo-msgdb-convert-type 'auto)

;; Convert msgdb when hitting s all in Summary mode
(setq elmo-msgdb-convert-type 'sync)

;; Inhibit conversion
(setq elmo-msgdb-convert-type nil)
```

As is explained in above section, you may continue to use the old format. But you will have following limitations.

1. You cannot use forwarded mark ('`F`', '`f`').
2. You may only use '`important`' flag. The other global flags may not be available.

14.1.2 Changes from ''mark' folder to ''flag'.

The folder ''`mark`' will be automatically converted to ''`flag`' folder when you first start the new version of Wanderlust. But there are some restrictions on this type of migrated folder.

1. '`important`' flag attached will not be removed by deleting the associated message in ''`flag`' folder.
2. The message won't be deleted by removing '`important`' flag in ''`flag`' folder.
3. help-echo will not show you the old message.

If you have problem with migrating from ''`mark`' folder to the ''`flag`' folder, invoking *M-x elmo-global-mark-upgrade* will transfer the message from ''`mark`' folder to the ''`flag`' folder. The duplicated message will not be processed, you may issue that command repeatedly.

15 Terminology around Wanderlust

Here we explain terminologies used in this manual.

'`folder`' A container in which messages are stored.

'`group`' A set consists of some folders.

'`access group`'
A special group consists of automatically collected folders under some specified path. See Section 2.5 [Folder Definition], page 7 .

'`summary buffer`'
A buffer for displaying list of messages in some folder.

'`sticky summary`'
Compared with ordinary summary buffer which will be destroyed after exiting from it, this type of summary will be remain even after exiting by *q* or *g*. See Section 5.5 [Sticky Summary], page 36 .

'`expire`' To delete or put into the archive expired messages. See Section 9.1 [Expire], page 70 .

'`score`' See Chapter 10 [Scoring], page 78 .

'`prefetch`'
To cache messages beforehand in order to read messages after you will be disconnected from the server.

16 Wanderlust Mailing List

Topics related to Wanderlust are discussed in following mailing lists. The latest version is also announced there.

Wanderlust Mailing List <wl@ml.gentei.org>

In this list Japanese is mainly used for discussion. We also have a list for discussion in English:

Wanderlust List in English <wl-en@ml.gentei.org>

(Messages posted to this list are also forwarded to the former one.)

A guide can be obtained automatically by sending mail to wl-ctl@ml.gentei.org (or to wl-en-ctl@ml.gentei.org for the English one) with the body

```
# guide
```

Please send bug reports or patches to one of those lists. You can post to the mailing list even though you are not a member of it.

If you send a bug report, please attach Backtrace with it.[1]

I would like to express my thanks to the members of the mailing list for valuable advice and many pieces of code they contributed.

16.1 Archive

You can read messages posted to the mailing list on the web or in NetNews.

Read messages posted to <wl@ml.gentei.org>

http://dir.gmane.org/gmane.mail.wanderlust.general.japanese
news://news.gmane.org/gmane.mail.wanderlust.general.japanese

Read messages posted to <wl-en@ml.gentei.org>

http://dir.gmane.org/gmane.mail.wanderlust.general
news://news.gmane.org/gmane.mail.wanderlust.general

[1] http://www.jpl.org/elips/BUGS-ja.html describes how to in Japanese.

17 Additional Information

17.1 Brief History

```
1998  3/05    Tried to make a prototype that displays MH messages in threads.
      3/10    Made a msgdb mechanism by elisp.
      3/26    IMAP and NNTP can be displayed in threads.
      4/13    Began to assemble thread display modules as elmo.
      5/01    Finished 0.1.0, initial version with many defects.
      6/12    I made a slip of the tongue and said I was writing elisp
              mailer supporting IMAP
      6/16    0.1.3 was announced at tm-ja, elisp ML.
      6/22    Thanks to Kitame-san, the mailing list started at northeye.org.
      7/01    Support for mm-backend (0.3.0).
      8/25    multi folder added (0.5.0).
      8/28    filter folder added (0.5.1).
      9/10    You can open/close threads (0.6.0).
      9/11    fldmgr by Murata-san made editing folders easy.
      9/18    lha folders added by Okunishi-san (0.6.3).
      9/24    Display of branches of threads (0.6.5).
      9/28    Compression folder supporting multiple archivers by Okunishi-
san.
     10/28    Off-line operations (0.7.4).
     12/09    Becomes beta version.
     12/21    wl-expire by Murata-san.
1999  2/03    auto-refile by Tsumura-san.
      4/28    wl-template by Murata-san.
      5/18    Released 1.0.0 stable.
      7/05    Scoring by Murata-san (2.1.0).
      9/26    New plugged system by Murata-san (2.2.2).
     12/20    Support Modified UTF7.
2000  3/24    Released 1.1.0 stable.
      4/03    CVS development started.
      5/07    Thread restoration & Its speed up with Murata-san.
      6/12    Address completion with LDAP with Chiba-san & Goto-san.
      7/11    killed message feature.
      7/18    Use UIDL in POP3.
      9/12    biff feature with Satata-san & Yamaoka-san.
     10/17    expire-hide by Okada-san.
     11/08    Released 2.4.0 stable.
2001  7/04    Released 2.6.0 stable.
      8/21    wl-addrmgr by Kitamoto-san.
     12/27    Released 2.8.1 stable.
2002 12/11    Released 2.10.0 stable.
2003  7/05    Released 2.10.1 stable.
      9/18    flag folder is added.
```

```
      9/20    New msgdb format (modb-standard) by H.Murata-san.
     10/20    Spam filter by H.Murata-san.
2004  1/06    Background color of the demo become configurable.
      2/09    'file' folder is added.
      9/12    forwarded mark.
              Default value of the mark strings are changed.
     12/24    Released 2.12.0 stable.
```

See '`ChangeLog`' for details.

17.2 The Name

According to a dictionary, Wanderlust has the meaning:

> wanderlust
> n eager longing for or impulse towards travelling in distant lands
> [Ger, fr wandern to wander + lust desire, pleasure]

but I had no profound intention. (if farfetched, IMAP ⇒ you can read mail anywhere ⇒ desire to wander ?)

Elmo is the abbreviation of '`Elisp Library for Message Orchestration`'. At first I meant the red puppet in the Sesame Street, but you may associate it with Wandering ⇒ Drifting ⇒ Guidepost ⇒ St. Elmo's fire ⇒ elmo.

17.3 Code Names

Each versions has code names (they are almost jokes). Currently they are picked up alphabetically from the top 40 hits of U.S. Billboard magazines in 1980s.

(http://ntl.matrix.com.br/pfilho/html/top40/)

Index

Concept Index

E

F

G

I

K

L

M

N

O

P

Q

R

S

T

U

W

X

Z

Key Index

!

#

$

*

+

-

.

/

<

>

?

@

[

]

^

|

~

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

Variable Index

E

M

W

Function Index

C

W

Short Contents

Table of Contents