Lafite

The Interlisp Mail System

Documentation: {Phylum}<Lisp>Library>Lafite.Press
Program: {Phylum}<Lisp>Library>Lafite.Dcom
Revised: February 28, 1984 by Bill van Melle

This document presupposes knowledge of Interlisp-D and its display system. Familiarity with the Laurel message system is also helpful.

General comments

Mail systems are notorious for inspiring *''It would be nice ...''*, *''Why doesn't it ...''*, etc. thoughts. Please don't feel bashful about sending such messages to LafiteSupport (using Lafite, of course).

Lafite is the Interlisp system for reading and sending mail. Lafite retrieves inbound mail from the user's *inboxes* on one or more mail servers. Mail is retrieved into one or more *mail folders*, which can be any Interlisp-accessible file. Although much of Lafite is modeled after Laurel, it differs from Laurel in the following ways:

> Laurel can only access mail folders on the local disk; Lafite can access folders on remote file servers. Thus, it is not necessary to transfer mail folders back and forth between your local disk and your file servers.
>
> Laurel can only ''browse'' one mail folder at a time; Lafite can have several mail folders ''opened'' at the same time. Utilizing the Interlisp window system, you can view the table of contents of many mail folders and refer to messages in them independently.
>
> You may have multiple windows displaying messages and multiple windows for sending messages and you may move text freely among them.

Lafite can read mail files written by the Laurel and Hardy mail programs; the files it writes are in Laurel format.

Restrictions:

Currently, Lafite only communicates with Grapevine and MTP (used by MAXC) mail servers. Interfaces to other kinds of mail servers (e.g., Star mail) will be added as protocols for them are published and users desire them.

Mail files must be randomly accessible, ruling out FTP-only file servers.

Lafite operation

When Lafite is loaded, it can be started by calling `(LAFITE 'ON` *MAILFOLDER* `)`. Lafite will attempt to establish the user's mail server identity, read stored user data from the file `LAFITE.PROFILE` (if it exists) and bring up the Lafite Status window. It will then establish a browser for the file *MAILFOLDER*, creating an empty mail folder of that name if one does not exist. If *MAILFOLDER* is not supplied then `ACTIVE.MAIL` will be used (see `DEFAULTMAILFILENAME` below). If *MAILFOLDER* is supplied but is the atom `NIL`, then no mail folder is opened but you are free to send mail and open any mail folder at a later time.

There are three major types of windows used by Lafite: the Status Window; Browser Windows, which are views on particular mail folders; and Message Composition Windows. Each type of window has its own fixed menu of commands. In general, while a command is ''in progress'', the menu item that invoked it is greyed out. Commands may be selected with the LEFT or MIDDLE mouse buttons; in some cases, the MIDDLE button provides some sort of special treatment, described in the documentation of such commands.

The Lafite Status Window

The Lafite Status window contains a small, fixed menu and a region for Lafite status information. While Lafite is in operation, it runs a background process `LAFITEMAILWATCH` that polls the user's inboxes periodically and reports in the status region if there is new mail. Clicking in the status region causes the background process to wake up and report status immediately, instead of waiting its normal interval. The commands in this window's menu are as follows:

Browse pops up a menu of the mail folders that Lafite is aware that you have. Selecting 'Another Folder' prompts you for a folder name in the prompt window. After a folder is selected, a *browser* window onto that mail folder is opened.

Selecting the **Browse** command with the MIDDLE button brings up a menu of Browse-related commands. In addition to simple **Browse** there are these commands:

Browse Laurel Folder Browses a file that was produced by the Laurel mail reader version 6.1 or later. Laurel 6.1 files are almost the same as Lafite files, but contain some line-formatting information that is stripped out by this command. After you have applied this command once to a file, you can subsequently browse the file with the normal **Browse** command (until you use Laurel on it again, of course).

Forget Folder Removes a folder from the list of known mail folders.

Forget Message Form Removes a message form from the list of known message forms (see **Save Form** command).

Send Mail brings up an Interlisp text editor window on a message form. The form is a canonical ''empty message'' if **Send Mail** is selected with the LEFT button. If **Send Mail** is selected with the MIDDLE button, a menu is presented with the following choices:

Standard Form provides an empty message template (same as using the LEFT button).

Last Message recalls the text of the last message edited.

Lisp Report provides a message template to report an Interlisp bug or make a suggestion.

Lafite Report provides a message template similar to **Lisp Report** but sent to Lafite maintainers.

Private Form prompts for a form name, which can be any text file, or a form created by the **Save Form** command (below).

Also in the menu are any names of known user-defined message forms created by the **Save Form** command. Each message form runs in its own process, so you can have several in progress at once. When you have finished composing the message, click **Deliver** in the message's menu.

Quit Stops Lafite, closes all browser windows and updates the associated mail folders. It also saves the names of known mail folders and form files on the file `LAFITE.PROFILE` so that this information will be available when you next run Lafite. You may achieve the same result under program control by calling `(LAFITE 'OFF)`, rather than buttoning this menu item. Most users find that they never invoke 'Quit', but rather keep Lafite always active in the background.

Browser Windows

A Browser window is a view onto a mail folder. The main part of the window displays the table of contents, a one-line summary of each message. This window is scrollable in both dimensions. Above the table of

contents is a horizontal menu (called the *browser menu*), containing commands specific to the mail folder associated with the browse window. Above the menu is a prompt window, in which various status information related to the browser is printed, and where some information is prompted for.

Browser comands operate on the currently selected set of messages. A selected message is indicated by a black triangle to the left of its message number. When the cursor is in the browser window it changes to a right-pointing arrow. When the cursor is this right-pointing arrow, messages can be selected by the following button operations:

left button selects just this single message, deselecting any other selected message.

middle button adds a message to the current selection.

right button extends the selection up or down. Deleted messages are not included in this extension unless the control key (CTRL) is down.

shift key (SHIFT) and any button removes this message from the current selection.

The commands in the browser menu are as follows:

Display displays the selected message. If the selected message is already displayed, the selection is advanced to the next undeleted message, and this message is displayed. If there is more than one message selected, buttoning **Display** cycles through the messages.

If you select **Display** with the LEFT button, the message is displayed in the primary message display window for the browser, replacing any previously displayed message. If you select **Display** with the MIDDLE button, the message is displayed in a newly created window, for which you will be prompted. Using the MIDDLE button you can make multiple windows containing messages for further reference (e.g., to use in composing your own message).

Delete deletes the selected messages. A deleted message is indicated by a black line through its summary line. The message is not actually removed from the mail folder until you Expunge (see Update, below).

Undelete undeletes the selected messages.

Answer constructs a Message Composition Window containing an *answer template* for the current message. After you deliver the answer, an ''**a**'' will appear in the browser window as the message's mark.

Forward similar to 'Answer' but the message form is a *forward template* for the currently selected messages. After you deliver the forwarded the messages, an ''**f**'' will appear as the message's mark.

Hardcopy prints the selected messages on your local printing device. When the hardcopy is complete, the message's mark is changed to ''**h**'' if the message didn't already have a more interesting mark.

Move To pops up a menu of known mail folders and moves the selected messages to the chosen folder. A new mail folder can be created by selecting 'Another Folder' and typing in the mail folder name in the prompt window. You will then be asked to confirm the move. When the move is successful, the messages are marked deleted in the source browser window, and given the ''**m**'' mark.

The name of the mail folder you most recently moved messages to appears in the title bar of the browser window as the ''Default 'Move To':'' folder. You can ''accelerate'' subsequent Move operations by selecting the 'Move To' command with the MIDDLE button. This will perform the move to the Default 'Move To' folder without asking for confirmation.

Update The changes that you make to your mail folder (deletions, changes of message marks, etc) are not actually transmitted to the physical mail file until you perform the Update command. There are two subcommand choices for Update:

> **Write out Changes Only** makes the browser and the mail file completely consistent with one another: if you were at this point to logout from Lisp, run Lafite in another incarnation of Lisp, and browse the same mail folder, you would get a browser that was in exactly the same state, deleted messages and all. This command involves writing out to your mail file and its table of contents the information about what has changed: new marks, deletions, newly parsed messages.
>
> **Expunge Deleted Messages** in addition to making the browser and the mail file completely consistent with one another, this command compacts your physical mail file so as to remove all messages marked deleted.

Which to choose? You eventually want to Expunge, so as to reduce the amount of file space your mail requires, but Expunge is not always fast. **Expunge** requires rewriting your mail file starting at the first deleted message, so is somehow ''proportional'' to the number of undeleted messages beyond that point. **Write out Changes Only** is proportional to the number of changes, and is thus usually faster than **Expunge**, except when the changes consist primarily of a string of deleted messages at the end of the folder.

If you Close or Shrink a Browser window that has had changes to it, you are prompted with a menu offering to **Write out Changes**, **Expunge**, or make no change before the window is closed/shrunk.

Get Mail brings new mail into the folder that this browser is viewing.

Changing the Message Mark

Each message in a mail folder has a ''message mark'', which is an additional tidbit of information about the message displayed in the summary line in the browser. Messages originally come in with the mark ''**?**'', meaning they are unexamined. Some Lafite commands change the mark automatically. For example, displaying a message changes its mark from ''**?**'' to a blank; answering a message changes its mark to ''**a**''. You can change the mark directly by selecting with the mouse in the narrow area immediately to the left of the message number, where the mark is printed. Simply click in the mark position, and type the new mark (a single character).

Message Composition Windows

On top of the text editor window is a horizontal menu for telling Lafite what to do with the message being created in the text editor window. When one has transformed the text to the desired message, select one of the following menu items:

Deliver sends your message. First the edit window ''grays out'', indicating Lafite is trying to send the message (this happens in its own process, so you can proceed with anything else you desire). If successful, the window shrinks to a small postal envelope with ***''x*** sent'' as the text of the address on the envelope, where ***x*** is the number of recipients of the message. If the delivery fails, for any of a variety of reasons (e.g. bad address fields, the grapevine timed out) which will be displayed in the prompt window, the message is redisplayed and you are back in the text editor to re-edit and then resend the message.

Save Form asks you for a file name on which to save this message for later use. This is the way to save

a message form for later repetitive use. It does *not* send it. If no extension is given for the file name, it defaults to the value of LAFITEFORM.EXT (initially LAFITE-FORM).

Abort closes the window and does nothing with the message. Note, however, that the text remains available as the ''last message'' (see 'Message Form' above) until you invoke another 'Message Form'.

Of course, you can always bring up the text editor command menu by MIDDLE buttoning the title bar of the editor window. You can then do any text editor command. If you select 'Quit' you will immmediately leave the text editor, bypassing the above commands.

When editing message forms, fields that should be filled in are enclosed in ''>>'' and ''<<''; e.g., >>Recipients<<, >>Subject<<. To make it easier to fill these in, the first field is highlighed in delete mode. Each successive field can be reached by typing the middle-blank key on the keyboard (or OPEN on the Dandelion keyboard, or whichever key you have assigned the TEdit ''Next'' syntax to). See the TEdit documentation for details.

Lafite System Functions and Variables

Below are the functions and global variables that control Lafite's behavior.

(LAFITE *ON/OFF MAILFOLDER*)

Used for starting and stopping Lafite with an optional mail folder. If *ON/OFF* is the atom ON then Lafite will start processing using *MAILFOLDER* . If *MAILFOLDER* is not supplied then Lafite will use your ACTIVE.MAIL mail folder (actually, the value of DEFAULTMAILFOLDERNAME). If *MAILFOLDER* is supplied but is the atom NIL then no browser will be created but you can send messages and start browsing mail files later using the 'Browse' menu item.

If *ON/OFF* is the atom OFF then Lafite will close all mail folders, expunging all messages marked for deletion; close all windows associated with Lafite; and remove the mail watching process from the active process list. This is the same as invoking 'Quit' in the Lafite Status Window.

LAFITEDEFAULTHOST&DIR

Lafite uses its own host and directory names for mail folders, LAFITE.PROFILE, etc., rather than the current connected directory because you may want to keep your mail folders someplace special (e.g., the local disk or your login directory), and the connected directory can change. LAFITEDEFAULTHOST&DIR is provided to tell Lafite where you generally keep your mail.

LAFITEDEFAULTHOST&DIR should be atomic, in the same form as LOGINHOST/DIR (e.g. {PHYLUM}<CARSTAIRS>). Lafite notices this variable only upon startup; to change it you need to Quit Lafite and restart it. If LAFITEDEFAULTHOST&DIR is NIL, then the value of LOGINHOST/DIR is used i.e., your login host and directory.

DEFAULTMAILFILENAME

If no mail folder is supplied to the function LAFITE, i.e., you call (LAFITE 'ON), then the value of this variable is used. Initially, ACTIVE.MAIL.

LAFITEMAIL.EXT

The default extension for names of mail folders. Initially, MAIL.

LAFITETOC.EXT

The string appended to the name of a mail folder to produce the name of its table of contents file. Initially, -LAFITE-TOC.

LAFITEFORM.EXT

The default extension for names of user-defined form files. Initially, LAFITE-FORM.

MAILWATCHWAITTIME

The number of minutes between polling for new mail from your mail servers. Initially set to 5.

LAFITEFLUSHMAILFLG

If NIL, Lafite won't flush your Grapevine mailbox when retrieving new mail, so the mail will still be there when you invoke Get Mail again. Initially, T.

LAFITENEWPAGEFLG

If T, then the Hardcopy command will start each message on a new page. Otherwise it will separate each message by a line of dashes. Intially, T.

LAFITEDISPLAYAFTERDELETEFLG

If T, Lafite will display the next message if you delete the one that is in the message display window and the next message has not been displayed. If ALWAYS, then it will display the next message even if it has already been seen. Initally, T. T is roughly Laurel semantics, ALWAYS is Hardy semantics.

LAFITEBROWSERREGION, LAFITEDISPLAYREGION, LAFITEEDITORREGION

These are REGIONs which are used to describe where the primary (i.e. first) window of each type is to be placed on the screen. Initally, they are set to something ''reasonable'' for the standard initial display configuration. If you set them to NIL then you will be asked to specify a region (via GETREGION) the first time any such window is created.

LAFITEDISPLAYFONT, LAFITEEDITORFONT, LAFITEHARDCOPYFONT

These are the fonts used for displaying messages, composing messages, and making hardcopy. You may change them individually. They should be FONTDESCRIPTOR's as returned by FONTCREATE. Initially, they are all TimesRoman12.

LAFITEBROWSERFONT

The font used for displaying the table of contents in the browser window. Initially, Gacha10.

LAFITESTATUSWINDOWPOSITION

Specifies where the status window appears when Lafite is invoked. It is a POSITION or NIL (in which case you will be asked to specify a position when Lafite starts).

LAFITENEWMAILTUNE, LAFITEGETMAILTUNE

(Dandelion only) These are lists of the form acceptable to the function PLAYTUNE, or NIL, in which case they are ignored. LAFITENEWMAILTUNE is played when Lafite discovers you have new mail waiting; LAFITEGETMAILTUNE is played when a Get Mail command completes.

LAFITEENDOFMESSAGESTR, LAFITEENDOFMESSAGEFONT

LAFITEENDOFMESSAGESTR is a string containing the text of the ‘‘End of Message’’ token displayed at the end of a message; LAFITEENDOFMESSAGEFONT is the font in which it is displayed. If LAFITEENDOFMESSAGESTR is NIL, then no ‘‘End of Message’’ token will appear.

LAFITEIFFROMMETHENSEENFLG

If true, then messages sent from you are considered ‘‘Seen’’ (and hence do not have the mark ‘?’), even though you have not yet displayed them. Initially T.

LAFITEBUFFERSIZE

The number of 512-character buffers used by the stream managing the file behind an open browser window. If you regularly receive very long messages, you might want to increase this to improve performance of Display followed by HardCopy or Move. Initially 20, which handles up to about 10,000-character messages.

Adding New Message Forms to Lafite

The normal way to add new message forms to Lafite is to edit an existing form (or build one from scratch) and save it away using the ‘Save Form’ menu item. You can also provide message forms that compute the text on the fly, as, for example, the ‘Lisp Support’ selection does. To add your own items to the ‘Message Forms’ menu, add a standard three-element menu item to the variable LAFITESPECIALFORMS and then set the variable LAFITEFORMSMENU to NIL (this is where the menu is cached). The three-element menu item should yield a LITATOM as its ‘‘value’’,that atom being interpreted as follows:

1. If the atom has a function definition, the function is called (with no arguments) and the returned value (a string or a TEdit TextStream) is used;

2. If the atom has a value, its value (a string or a TEdit TextStream) is used;

3. otherwise, a copy of the file by that name is used.

For example, if TEdit wanted to add a message form that contained the date the TEdit was made (similiar to Lafite’s bug report form) it could add

```
("TEdit Support" (QUOTE MAKETEDITFORM)
           "Make a form to report a problem with TEdit")
```

to LAFITESPECIALFORMS; MAKETEDITFORM could be defined to be

```
[LAMBDA NIL
  (PROG (OUTSTREAM)
        (SETQ OUTSTREAM (OPENTEXTSTREAM ""))
        (printout OUTSTREAM "Subject: TEdit: >>Subject<<" T)
        (printout OUTSTREAM "To: " TEDITSUPPORT T)
        (printout OUTSTREAM "cc: " (USERNAME) T)
        (printout OUTSTREAM "TEdit-System-Date: " TEDITSYSTEMDATE T T)
        (printout OUTSTREAM ">>Message<<" T)
        (RETURN OUTSTREAM]
```

where TEDITSUPPORT and TEDITSYSTEMDATE are variables set by TEdit. Lafite supplies one function to make this kind of message form easier to construct:

(MAKEXXXSUPPORTFORM SYSTEMNAME ADDRESS SYSTEMDATE)

Creates a message form (a TEdit stream) to be mailed to the maintainers of SYSTEMNAME. SYSTEMNAME is the name of the subsystem (a string); SYSTEMDATE, if non-NIL, is a date (string) of importance to include in the message; and ADDRESS is the mail system address of the intended recipient(s).

For example, if MAKETEDITFORM were defined as

```
[LAMBDA NIL
  (MAKEXXXSUPPORTFORM "TEdit" "TEditSupport" TEDITSYSTEMDATE)]
```

Then selecting ‘‘TEdit Support’’ in the Message Forms menu would produce a form such as the following:

Subject: TEdit: >>Subject<<
To: TEditSupport
cc: vanMelle.pa
TEdit-System-Date: 3-Feb-84 12:23:49
Lisp-System-Date: 3-Feb-84 18:13:22
Machine-Type: Dorado

>>Message<<