# The Elm Reference Guide

*A comprehensive list of all commands,*
*options and such to the* **Elm** *mail system*

The Elm Mail System
(Version 2.4)

Syd Weinstein, Elm Coordinator
Datacomp Systems, Inc.
3837 Byron Road
Huntingdon Valley, PA 19006-2320

email: elm@DSI.COM or dsinc!elm

# The Elm Reference Guide

(The Elm Mail System, Version 2.4)

October 1, 1992

Syd Weinstein
Datacomp Systems, Inc.
3837 Byron Road
Huntingdon Valley, PA 19006-2320

email: elm@DSI.COM or dsinc!elm

Derived from
"The Elm Mail System, Version 2.0"
by
Dave Taylor
Intuitive Systems
Mountain View, California
email: taylor@intuitive.com or limbo!taylor

## 1. Introduction

There are many parts to a complex software system and **The Elm Mail System** is no different. This document describes fully all the options available in the mailer, including the command line options, the commands (in considerably more detail than in *The Elm Users Guide*) and the *.elm/elmrc* file.

To be more explicit, this document covers: a discussion of the *.elm/elmrc* file, command line options of *elm*, outgoing mail processing, responses of various commands, mail archive folders, the Alias system, system aliases etc, more on the **Elm** utilities, and a section for expert mail users.

Without any further ado, then, let's get this show on the road!!

## 2. The .elm/elmrc File

Elm, like lots of other software on the UNIX® system, has the ability to automatically read in a configuration file at each invocation. The file must be called *elmrc* and reside in the *.elm* directory located in your home directory. It can have any of the entries below, in any order. If you are missing any entries, or if you don't have an *.elm/elmrc* file, the default values (listed below for each option) or those values specified by your systems administrator in a system wide *elm.rc* file will be used. Note that those options below designated with * can be altered using elm via the o)ptions screen while those designated with a + can be placed on the options screen using the *configoptions* variable. Also note that when you save a new *.elm/elmrc* file via the '>' command of the options screen, it will be (re)created only including those options that you have changed via your original *.elm/elmrc* file or via the o)ptions screen.

## String Variables

**aliassortby*** You can have your alias display sorted by any of the following ways:

- alias — This will sort according to *aliasname* for each address.
- name — This will sort according to *username* for each address.
- text — This will present the aliases in the order found in the *aliases.text* file.

Each of these fields can also optionally be prepended with the sequence "reverse-" to reverse the order of the sort. This doesn't imply anything about the order of the message in the *aliases.text* file itself and affects only their order on the display screen. (The default is *name* order.)

**alteditor+** The editor to use when mailing to a message that already includes text, as the "builtin" editor cannot handle that situation. Messages that already include text are forwarded messages and replies where the original message is included in the reply. This value is not needed if the *editor* is not "builtin".

**alternatives** This is a list of other machine/username combinations that you receive mail from (forwarded). This is used when the *group reply* feature is invoked to ensure that you don't send yourself a copy of the outbound message. (The default is a list of no alternatives.)

**attribution** When you *forward* a message or *reply* to it, you can optionally attribute the quoted text to its original author. Defining the attribution string here allows you to indicate the form that the attribution should take. The sequence '%s' in the attribution will be replaced by the name of the original author. Examples are:

```
attribution = According to %s:
attribution = %s writes:
```

**builtinlines** This is used to determine if the builtin pager should be used on some messages even if you would usually use an external pager program. There are two ways of determining whether the builtin pager should be used. If you want any message that is shorter than "n" lines to use the internal pager, set the parameter to "n". If you want the builtin pager to be used if the message is "m" lines shorter than the number of lines on your screen, set the parameter to be "-m". Setting the parameter to zero will result in the message always being sent through your external pager. This variable is used only if the pager is not set to the builtin pager.

**calendar*** This is used in conjunction with the '<' *scan message for calendar entries* command, as the file to append any found calendar entries to. (The default is *calendar* in your home directory.)

**charset** This is the Character Set used for Messages with Content-Type: text/plain. The default depends on your site's installation, but is usually *US-ASCII*. *charset* is only recognized if *MIME* support is configured. You could set charset to your preferred national Character Set. But be aware, that there is currently no way to change it online. Also you must be aware that Elm probably needs metamail to display messages with charset=US-ASCII, if you change charset. Elm tries to know which charset could display *US-ASCII* too, but its list of compatible charsets is probably not complete.

**displaycharset** This is the Character Set which is supported by your terminal. The default depends on your sites installation but is usually *US-ASCII*. For Sites with X-Windows support, *ISO-8859-1* would be a reasonable default.

**compatcharsets** This is the list of Character Sets, which are more or less a superset of US-ASCII. This enables Elm to display messages with charset=US-ASCII with the builtin pager, or your preferred pager instead of calling metamail. *compatcharsets* is only recognized if MIME support is configured. Currently the ISO-8859-X character sets are default.

**configoptions** This is a list of letters that indicate which of the run-time configurable options you desire placed on the o)ptions screen. There are 22 run-time configurable options, but only room for 15 on the normal sized screen. The default list is *^_cdefsopyv_am_un*. Two additional characters can be specified for formatting. Those are '_', which adds a blank line, and '^', which places the title message on that line instead of the bottom of the screen. The letters 'i', 'q', and 'x' are reserved for "return to index", "quit", and "exit" respectively and are not listed as part of the configoptions list. The options controlled by each letter are:

| | |
|---|---|
| a | A)rrow cursor *(arrow)* |
| b | B)order on copy *(prefix)* |
| c | C)alendar file *(calendar)* |
| d | D)isplay mail using *(pager)* |
| e | E)ditor (primary) *(editor)* |
| f | F)older directory *(maildir)* |
| h | H)old sent message *(copy)* |
| j | J) reply editor *(alteditor)* |
| k | K) pause after pager *(promptafter)* |
| l | A(l)ias Sorting *(aliassortby)* |
| m | M)enu display *(menu)* |
| n | N)ames only *(names)* |
| o | O)utbound mail saved *(sentmail)* |
| p | P)rint mail using *(print)* |
| r | R)eply copies msg *(autocopy)* |
| s | S)orting criteria *(sortby)* |
| t | T)ext editor (~e) *(easyeditor)* |
| u | U)ser level *(userlevel)* |
| v | V)isual Editor (~v) *(visualeditor)* |
| w | W)ant Cc: prompt *(askcc)* |
| y | Y)our full name *(fullname)* |
| z | Z) signature dashes *(sigdashes)* |

**easyeditor+** The editor to be used by the ~e escape within the "builtin" editor. The default value is the value of the configuration option emacs_editor.

**editor*** The editor to use when typing in new mail. If you select ‘‘none’’ or ‘‘builtin’’ you'll get a Berkeley Mail style interface for all mail that doesn't already have text in the buffer (e.g. a reply, mail with a ‘‘signature’’, etc). There are two possible formats for it, either a command that can have a filename affixed to (as a suffix) before being executed, or a string that contains the meta-sequence ‘%s’ which will be replaced by the name of the file before being executed. Examples of each are:

```
editor = emacs -nw
editor = emacs -nw %s -f text-mode -f turn-on-auto-fill
```

(The default is to use the value of $EDITOR in your current environment, and if not set, an editor selected by the person who configured *elm* for your system.)

escape — The character used with the ‘‘builtin’’ editor (see above) to escape from text entry to input a command. When a line begins with this character, the editor interprets it as a command rather than as text to add. (The default is ‘‘~’’.)

fullname* — This is the name the mailer will use in messages you send. It is highly recommended that you use your full name and nothing strange or unusual, as that can appear extremely rude to people receiving your mail. (The default is to use the ‘‘gcos’’ field from the */etc/passwd* file on systems that use this field to store full names, and to use the contents of *.fullname* in your home directory on other systems.)

hostdomain — This is the domain name of your system. This variable is only valid in the system wide *elm.rc* file. It is only necessary if the value returned by the getdomainname system call is incorrect for your mail use or if that system call is unavailable on your system. If this variable is specified, then the hostfullname variable must also be specified.

hostfullname — This is the full "Fully Qualified Domain Name" of your system. This variable is only valid in the system wide *elm.rc* file. It is only necessary if the value returned by the getdomainname and gethostname system calls are incorrect for your mail use or if those system calls are unavailable on your system. It is also required if either hostdomain or hostname are used within the system wide *elm.rc* file.

hostname — This is the local node name of your system. This variable is only valid in the system wide *elm.rc* file. It is only necessary if the value returned by the gethostname system call is incorrect for your mail use or if that system call is unavailable on your system. If this variable is specified, then the hostfullname variable must also be specified.

localsignature — See signature.

maildir* — This is your folder directory. When you specify a folder name beginning with the ‘=’ metacharacter[1], it stands for this directory name. That is, if you save a message to folder =*stuff* the ‘=’ will be expanded to the current value of *maildir*. (The default is the directory *Mail* in your home directory.)

pager* — This is the program to be used to display messages. You can specify ‘‘builtin’’ or the name of any standard pager. If you use ‘‘builtin+’’, each screenfull of displayed message is ‘‘paged’’ from the top of your screen with a title line, while ‘‘builtin’’ simply ‘‘scrolls up’’ subsequent screenfulls once it has ‘‘paged’’ the first screenfull. (The default is to use the value of $PAGER in your current environment, and if not set, a pager selected by the person who configured *elm* for your system, quite likely ‘‘builtin+’’.)

1. Note that ‘%’ and ‘+’ are synonymous with ‘=’ throughout *elm*

precedences — Some mail transports will look at a "Precedence" header in outbound mail messages to determine how to deliver the message. The Elm header editing menu allows you to place a precedence on your mail messages. By default, Elm will allow any value to be specified as the message precedence. This option may be used to restrict the allowed precedences to a particular list. For example, you might say:

```
precedences = special-delivery air-mail first-class bulk junk
```

Exactly what precedences your mail transport supports and what they do (if anything at all!) will vary from site to site.

The distinction between the "Precedence" and "Priority" headers is subtle: the precedence tells the mail system how to handle the message and the priority tells the recipient how important the message is. Although these are quite different things, they are often related. This option will also allow you to associate message priorities with precedences. For example, you might say:

```
precedences = special-delivery:urgent air-mail:urgent first-class bulk junk
```

In this example, if you select an "air-mail" precedence then the message priority will default to "urgent". If you select a "first-class" precedence then no special priority is implied. The priorities given in this field will be used only if you have not already assigned a priority to your message, and even if one is assigned you can always go back and change it.

prefix+ — When you *reply* to a message or *forward* a message to another person, you can optionally include the original message. Defining the prefix value here allows you to indicate what the prefix of each included line should be. (The default is "> " and is standard in the Unix community.)

print* — This is the command used for printing mail messages. There are two possible formats for the command, depending whether or not the command contains the sequence '%s'. In the first form the mail message(s) to be printed are fed into the specified command. An example of this form is:

```
print = print -formfeed
```

In the second form the mail message(s) to be printed are dumped into a temporary file, and a '%s' in the command string is replaced with the name of the temporary file. An example of this form is:

```
print = pr %s | lpr
```

Another significant difference between the two forms is that in the second form all output from the command is discarded, but in the first form the command all output will be displayed on the terminal. For example, if you wish to use the "pass-through" feature to print on the aux port of a terminal, you must use the first form. The second form is provided for backward compatibility with old versions of Elm. It is depreciated and might be removed in a future release. (The default is set by the person who configured *elm* for your system.)

receivedmail — This is the folder to which incoming mail is saved after you've read it. When you answer *no* ('n') to the "keep unread messages in your incoming mailbox?" prompt or *yes* ('y') to the "store read messages in your "received" folder?", this is where the messages go. (The default is "=received", that is, a folder called *received* in your *maildir* directory).

remotesignature See signature.

sentmail* This is the folder to which a copy of outgoing mail is automatically saved. This will only be done if the *copy* flag is turned on (see below). Also note that if the *savename* feature (see below) is enabled then this folder may be ignored since the program may save to a folder that has the same name as the login of the person you're sending to. Whether or not a copy is saved, and to what folder, can be changed just prior to sending a message, see below. (The default is "=sent", that is, a folder called *sent* in your *maildir* directory).

shell This defines the shell to use when doing "!" escapes and such. (The default is to use the value of $SHELL in your current environment, and if not set, a shell selected by the person who configured *elm* for your system.) Note: the "!" escape is optional and may not be enabled in your version of Elm.

signature This file, if defined, will be automatically appended to all outbound mail before the editor is invoked. Furthermore, if you'd like a different ".signature" file for *local* mail and *remote* mail (remote being via other hosts), you can alternatively define two variables, *localsignature* and *remotesignature*, to have the same functionality. (The default is to not have signatures appended to your message.)

sortby* You can have your folder sorted by any of the following ways:

| | |
|---|---|
| from | This will sort according to whom each message is *from*. |
| lines | This will sort *shortest* to *longest* by message. |
| mailbox | This will leave the messages in the order found in the folder. |
| received | This will sort *least recently received* to *most recently received*. |
| sent | This will sort *least recently sent* to *most recently sent*. |
| status | This will sort by priority, action, new, tagged, then deleted. |
| subject | This will sort according to the *subject* of each message. |

Each of these fields can also optionally be prepended with the sequence "reverse-" to reverse the order of the sort. This doesn't imply anything about the order of the message in the folder itself and affects only their order on the index screen. (The default is *mailbox* order.)

tmpdir Use this if you want to define your own directory for the temporary file Elm creates while running. This is only necessary if using the system temporary directory could cause problems, such as when not all NFS clients mount the common temporary directory, or when the temporary directory is prone to being cleared periodically. The default entry of the system temporary directory is normally OK.

visualeditor+ The editor to be used by the ~v escape within the "builtin" editor. The default value is the value of the configuration option vi_editor.

weedout When specifying this option, you can list headers that you *don't* want to see when you are displaying a message. This list can continue for as many lines as desired, as long as the continued lines all have leading indentation. All headers in this entry append to the default weedout list. There are two special header flags. The first, *clear-weed-list*, will clear the default list. The second, *end-of-user-headers*, will terminate the entry, in case the following lines look like they might be more headers for the list. (The default is ‘‘Received:’’ and others.)

## Numeric Variables

bounceback — This is a hop count threshold value and allows you to set up the mailer so that when you send mail more than *n* machines away, it'll automatically include a ‘‘Cc:’’ to you through the remote machine. In practice this should be very rarely used. (Note: this refuses to bounce mail off an Internet address. The default is to have it set to zero, which disables the function)

timeout — On more advanced systems, it's nice to start up the mailer in a window and let it sit in background unless new mail arrives (see *wnewmail* for another window based program) at which point it can be brought up to the forefront of the system and read. In this case, it would be quite convenient to have the mailer internally resynchronize every so often. This option specifies the number of seconds that this occurs.

This is also useful for normal terminals, for example you can leave *elm* running at night (I usually do) and when you come in in the morning it'll be all ready to read your mail!

(The default is a 300 second (5 minute) timeout period).

userlevel* — This is what the program uses to determine the relative level of sophistication of the user — the values are 0 for a new user (the default), 1 for someone familiar with *elm* user, and 2 for experts. Some advanced features are hidden from novice users, while experts get less verbose prompt messages.

## Boolean Variables

alwaysdelete — When set, this changes the default answer of the prompt ‘‘Delete messages?’’ to the indicated value. The value for *alwaysdelete* also applies to deletions from the alias system. (The default is to have the answer be *No* (i.e. alwaysdelete = OFF).)

alwayskeep — This sets the default answer on the ‘‘keep unread mail in incoming mailbox’’ However, if you answered *No* to the ‘‘store read mail in "received" folder’’ it is presumed that you'd also want to keep your unread mail in the incoming mailbox, too, and the value of alwayskeep is ignored. (The default is to have the default answer be *Yes* (i.e., alwayskeep = ON).)

alwaysstore — This sets the default answer on the ‘‘store read mail in "received" folder’’ prompt to the value indicated. (The default is to have the default answer be *No* (i.e., alwaysstore = OFF).)

arrow* — Sometimes you are forced to use a slow, or ‘‘dumb’’ terminal. In this case, you can force the current message pointer to be the ‘‘->’’ sequence rather than the inverse bar. (Note that this is overridden by the similar ‘-a’ command line option, see below.) (The default is OFF.)

ask — This allows you to tell the *elm* system that you'd rather not be asked ‘‘Delete Mail?’’ and such each time you quit, resynchronize a folder or change folders, and instead it should just use the values of ‘‘alwaysdelete’’, ‘‘alwaysstore’’, and ‘‘alwayskeep’’ without prompting. (Note that when you quit *elm*, if you use ‘Q’ instead of ‘q’, you will never be questioned, regardless of how you have *ask* set.) (The default is ON, i.e. to ask the questions.)

askcc+ If turned off, this will allow you to send mail without being presented the ‘‘Copies to:’’ prompt for each message. This still allows you to explicitly include addresses in the ‘‘cc’’ list via either ‘‘~c’’ in the builtin editor, or via using the header editor. (The default is ON, i.e. to ask about copies.)

autocopy+ If turned on, this will automatically copy the text of each message being replied to into the edit buffer. Otherwise you will be prompted as to whether you want the message included in yours. (See the *prefix* option above.) (The default is OFF.)

confirmappend When set to ON this will force elm to ask for permission to append messages to the end of any file that already exists. If the file is a mail folder in the user’s mail directory or an ordinary file makes no difference. (The default is OFF.)

confirmcreate When set to ON elm will ask for permission before it creates a new file to store any messages in it. It makes no difference whether the new file would be a mail folder in the user’s mail directory or an ordinary file. (The default is OFF.)

confirmfiles This will allow you to have some last resort control over elm when a message would be appended (by copying, saving or auto-cc) to an existing file which is not a folder in your mail directory. (See maildir above.) When set to ON elm will ask you for permission to append a message to the end of an ordinary file. When set to OFF elm will operate just as before and silently add your message to the end of the specified file whether it is a folder or not. (The default is OFF.)

confirmfolders When set to ON this will allow you to be asked about creation of new mail folders in your mail directory. (See maildir above.) When set to OFF elm will silently create new mail folders whenever a copy of a message is going to be stored in a folder that does not already exist. (The default is OFF.)

copy+ This, in combination with the *sentmail* folder, will allow you to have silent copies of all outgoing mail made on the outbound step. Whether a copy is saved and to which folder can be set prior to sending a message, see below. (The default is OFF.)

forcename This, in combination with the *savename* option, governs how a copy of an outgoing message will be saved. (See *savename* below for details.) (The default is OFF.)

forms Enables the generation of and replies to forms type messages. See the *Elm Forms Mode Guide* for further information about mail forms.

keepempty The mail system has a habit of deleting folders when you’ve removed everything from them. With this option turned on, it will instead preserve them as zero-byte files. This option does not apply to your incoming mailbox. (The default is OFF.)

keypad If on, this tells *elm* that you have an HP terminal and enables the <NEXT>, <PREV>, <HOME> and <SHIFT-HOME> keys. (The default is OFF.)

menu* If turned off, this will inhibit the Menu display on all of the screen displays within the *elm* program. (Note that this is overridden by the similar ‘-m’ command line option, see below.) (The default is ON.)

metoo If turned on, a user will get a copy of a message he sends to a mailing list he is on. Otherwise the user will not get a copy of message sent to a mailing list he is on. (The default is OFF.)

names* If turned off, the primary recipients' address is displayed on your screen with their full names when you send a message. Otherwise, only the full names are given. (The default is ON.)

movepage If this is enabled then commands that move through the folder by pages (the '+', '-', <right-arrow>, and <left-arrow> keys) will also move the current message pointer to the top of that page of messages. If this is turned off then moving through the pages doesn't alter the current message pointer location. (The default is OFF.)

noheader This boolean flag tells the mailer not to include the headers of messages when copying a message into a file buffer for replying to or forwarding. (The default is ON.)

pointnew If this is turned on, the mailer will be automatically pointing to the first new message in your folder when started, instead of at message #1. This will only be effective for the incoming mailbox since other folders are not expected to have 'new' mail. (The default is ON.)

promptafter+ If this flag is ON and you use an external pager, when the pager exits, you will be prompted for a command rather than returned directly to the index screen. If the external pager you are using exits when it reaches end of file (i.e. end of the message), you should have this flag ON, otherwise the last screen of the displayed message will not be held but instead be immediately replaced by the index screen. If the external pager you are using does not exit until you command it to exit, you have a choice. If you usually want to see the index screen before issuing a command, having the flag OFF will cut down out the extra keystroke needed to return to the index screen. If you usually don't need to see the index screen to before issuing the next command, having the flag ON will allow you to proceed with your next command without having to wait for the redrawing of the index screen. (The default is ON.)

resolve This is a boolean flag that defines the behavior of the program for such actions as deletion, saving a message and so on. Specifically, with this option enabled, as soon as mail is 'dealt with' the program moves you to the next message in the folder, with deletion, undeletion, saving a message and forwarding a message all being treated as dealing with email in this manner. (The default is ON.)

savename One of the problems with electronic mail systems is that one tends to get very large, one-dimensional (flat) files that contain lots of completely unrelated mail. If this option is turned on, *elm* will use a more intelligent algorithm — on incoming mail, when you *save* or *copy* it, the default mailbox to save to (changeable by pressing anything other than <return> of course) is a folder that is the *login name* of the person who sent you the message. Similarly, when sending mail out, instead of just blindly saving it to the *sentmail* folder, *elm* will save it to a folder that is the login name of the person who is to receive the mail[2].

If *forcename* is off (see above), the copy will be saved to that folder only if the folder already exists. In practice, this means that important people that you communicate with (those that you tend to save mail from) have folders that are actually *a recorded log of the discussion in both directions* and those others (random mailings) are all stuffed in the *sentmail* folder for easy perusal and removal.

Otherwise, if you always want to save copies of outgoing messages in separate folders by recipient login name, you'll want to set *forcename* to on. (The default for *savename* is ON.)

2. When sending to a group, it's saved to the first person in the list only.

sigdashes+ If on, this tells *elm* that you wish to follow the convention of prefixing your signature with newline dash dash blank newline. This will be placed in your message before your signature file. If off, the signature file is placed at the end of the message without any prefix.

softkeys If on, this tells *elm* that you have an HP terminal with the HP 2622 function key protocol and that you'd like to have them available while in the program. (The default is OFF.)

titles This flag allows you to have the first line of a message titled with:

`Message` *N/M* `from` *username* *date* `at` *time*

where all the information has been previously extracted from the message. This is especially useful if you weed out all the headers of each message with a large 'weedout' list. (The default is ON.)

usetite This flag allows you to enable/disable use of the *termcap/terminfo* **ti/te** capabilities. Many terminal emulators require it (not the least of which is the OpenLook *cmdtool*). Some terminal emulators clear the screen on **te** (some *xterm*s). (Note that this is overridden by the similar '-t' command line option, see below.) (The default is ON.)

weed This is a boolean flag that, in combination with the ''weedout'' list, allows you to custom define the set of headers you would like to not have displayed while reading messages. (The default is ON.)

One more thing: the format for each of the lines is:
`variable = value`
and for boolean variables, *value* can be 'ON' or 'OFF' only.

For a better idea of how this all works, here's my *.elm/elmrc* file. While looking through it, notice that you can have lots of comments and blank lines for readability and that you can also use 'shell variables' and the '~' metacharacter for your home directory, and they are expanded accordingly when read in by the mailer. (Note that this was automatically saved by the *elm* program on the fly from the o)ptions screen)

```
#
# .elm/elmrc - options file for the ELM mail system
#
# Saved automatically by ELM 2.4 for Dave Taylor
#

# For yes/no settings with ?, ON means yes, OFF means no

# where to save calendar entries
calendar = ~/.Agenda

# what editor to use ("none" means simulate Berkeley Mail)
editor = none

# the character to use in the builtin editor for entering commands
escape = ~

# the full user name for outbound mail
fullname = Dave Taylor
```

```
# where to save received messages to, default file is "=received"
receivedmail = $HOME/Mail/received

# where to save my mail to, default directory is "Mail"
maildir = /users/taylor/Mail

# program to use for displaying messages ('builtin' is recommended)
pager = builtin

# prefix sequence for indenting included message text in outgoing messages...
prefix = >_

# how to print a message ('%s' is the filename)
print = lpr -Plw2 %s

# where to save copies of outgoing mail to, default file is "=sent"
sentmail = /users/taylor/Mail/mail.sent

# the shell to use for shell escapes
shell = /bin/csh

# local ".signature" file to append to appropriate messages...
localsignature = localsig

# remote ".signature" file to append to appropriate messages...
remotesignature =  remotesig

# do we want dashes above signatures? (News 2.11 compatibility and convention)
sigdashes = ON

# how to sort folders, "Mailbox" by default
sortby = Reverse-Received

# how to sort the alias list, "Name" by default
aliassortby = Name

# should the default be to delete messages we've marked for deletion?
alwaysdelete = ON

# should the default be to store read messages to the "received" folder?
alwaysstore = ON

# should the default be to keep unread messages in the incoming mailbox?
alwayskeep = ON

# should we use the "->" rather than the inverse video bar?
arrow = OFF

# should the message disposition questions be displayed(ON) or
# auto-answered(OFF) with the default answers when we resync or change folders?
ask = ON

# would you like to be asked for Carbon-Copies information each msg?
```

```
askcc = ON

# automatically copy message being replied to into buffer?
autocopy = OFF

# threshold for bouncing copies of remote uucp messages...
# zero = disable function.
bounceback = 0

# save a copy of all outbound messages?
copy = ON

# do we want to be able to mail out AT&T Mail Forms?
forms = OFF

# should we keep folders from which all messages are deleted?
keepempty = OFF

# we're running on an HP terminal and want HOME, PREV, NEXT, etc...
keypad = OFF

# should we display the three-line 'mini' menu?
menu = ON
# would you like a copy of a message you send to an alias you are on???
metoo = OFF

# when using the page commands (+ - <NEXT> <PREV>) change the current
# message pointer...?
movepage = ON

# just show the names when expanding aliases?
names = ON

# when messages are copied into the outbound buffer, don't include headers?
noheader = ON

# start up by pointing to the first new message received, if possible?
pointnew = ON

# prompt for a command after the external pager exits?
promptafter = ON

# emulate the mailx message increment mode (only increment after something
# has been 'done' to a message, either saved or deleted, as opposed to
# simply each time something is touched)?
resolve = ON

# save messages, incoming and outbound, by login name of sender/recipient?
savename = ON

# save outbound messages by login name of sender/recipient even if the
# associated folder doesn't already exist?
forcename = OFF
```

```
# are we running on an HP terminal and want HOME, PREV, NEXT, etc...?
# (this implies "keypad=ON" too)
softkeys = OFF

# set the main prompt timeout for resynching...
timeout = 60

# display message title when displaying pages of message?
titles = ON

# are we good at it?  0=beginner, 1=intermediate, 2+ = expert!
userlevel = 2

# enable the weedout list to be read?
weed = ON

# what headers I DON'T want to see, ever.
weedout = "Path:" "Via:" "Sent:" "Date" "Status:" "Original" "Phase"
            "Subject:" "Fruit" "Sun" "Lat" "Buzzword" "Return" "Posted"
            "Telephone" "Postal-Address" "Origin" "X-Sent-By-Nmail-V" "Resent"
            "X-Location" "Source" "Mood" "Neuron" "Libido" "To:" "X-Mailer:"
            "Full-Name:" "X-HPMAIL" "Cc:" "cc:" "Mmdf" "Network-" "Really-"
            "Sender:" "Post" "Message-" "Relay-" "Article-" "Lines:"
            "Approved:" "Xref:" "Organization:" "*end-of-user-headers*"

# alternative addresses that I could receive mail from (usually a
# forwarding mailbox) and don't want to have listed...
alternatives = hplabs!taylor  hpldat!taylor  taylor@hplabs  taylor%hpldat
```

## 3. The Command Line Options

There are a number of command line options to the *elm* program, with only one that needs to be remembered: ‘‘-?’’or ‘‘-h’’ for help.

The flags are:

-a Arrow. This allows you to have the ‘‘->’’ arrow pointer rather than the inverse bar. This can also be set in the *.elm/elmrc* file — check the boolean variable *arrow*).

-c Check only. This is useful for expanding aliases without sending any mail. The invocation is similar to invoking *elm* in send-only mode: `elm -c` *list-of-aliases*

-d *n* Set debug level to *n*. Useful for debugging the *elm* program, this option will create a file in your home directory called *ELM:debug.info*, then output a running log of what is going on with the program. Level *n* can be 1 through 11, where the higher numbers generate more output. This option might be disabled by the the person who configured *elm* for your system.

-f *folder* Folder. Read specified folder rather than the default incoming mailbox. Note that you can use the same metacharacters (e.g. ‘=’) as when you *change folders* from within the program. You can also use the same abbreviatory symbols (‘!’, ‘>’ and ‘<’), but remember to ‘‘single quote’’ them in case they have special meaning in the shell you use.

-h or -? Help. Gives a short list of all these options and exits.

-i *file* Include. Include prepared file in edit buffer before sending. This facilitates using *elm* with other programs that interface with mail (like news readers, for example). There is an example of how to set up the **rn** news reading program to use *elm* in *The Elm Users Guide*. The file specified is copied into the temporary file just before the signature file.

-k Keypad — This option, when used, lets the *elm* program know that you're on an HP terminal, and it can then interpret the <PREV>, <NEXT> and <HOME>/<SHIFT>-<HOME> keys accordingly. If you are not on an HP terminal, it is recommended that you do NOT use this option. (See the *keypad* option in the *.elm/elmrc* section.)

-K Keypad + softkeys. The *elm* mailer can use the HP softkeys as an alternative form of input. If you specify this option be sure that you're on an HP terminal that can accept the standard 2622 terminal escape sequences! (See the *softkeys* option in the *.elm/elmrc* section for more information.)

-m Inhibit display of the 3-line menu when using the mailer. This, of course, gives you three more message headers per page instead. (See also the `menu` option in the *.elm/elmrc* section.)

-s *subject* In send-only and batch mode, this is how to indicate the subject of the resulting message. Please see the section on "Non-Interactive Uses of Elm" in *The Elm Users Guide* for more information.

-t Disable use of the *termcap/terminfo* **ti/te** capabilities. Many terminal emulators require it (not the least of which is the OpenLook *cmdtool*). Some terminal emulators clear the screen on **te** (some *xterm*s). (See also the `usetite` option in the *.elm/elmrc* section.)

-z Zero. This causes the mailer not to be started if you don't have any mail. This emulates the behavior of programs like *Berkeley Mail*.

All the above flags default to reasonable options, so there is usually no need to use them. Furthermore, the most used flags are available through the *.elm/elmrc* file. See above.

## 4. Multi-Media Capabilities of Elm

Elm optionally provides you with some Multi-Media features, which are compliant to the MIME (Multipurpose Internet Mail Extension) IETF draft.

If the support is compiled into Elm, on the receiving side Elm accesses Metamail from Nathaniel Borenstein of Bellcore. If you receive a MIME compliant Message, Elm calls Metamail automatically on displaying the message. Metamail asks you for each part of the message, if you want to display it. Metamail uses the display programs available at your site. This is controlled through the mailcap file.

On the sending side, there is a simple mechanism integrated in elm to compose MIME compliant messages. If you have one or more key lines `[include file contenttype/subtype encoding]` in the message body, at each of these key lines, a file would be included, and will be a part of the message. The text lines before, between and after the *include* lines go into extra Message parts of type Text.

As an example, say you want to include the file foo.gif into your message, which is a GIF Image, and you want to use *base64* encoding, use the following line:

```
[include foo.gif image/gif base64]
```

Or you want to include a text file which contains plain ASCII:

```
[include foo.txt text/plain]
```

The *encoding* parameter is optional and the default is 7bit.

Refer to RFC1341 for valid *contenttype/subtype* and *encoding* parameter values.

## 5. Special Outgoing Mail Processing

There are a few extra features that the mailer offers on outgoing mail that are worthy of mention:

The first, and probably the most exciting feature[3], is the ability to send *encrypted* mail! To do this is extremely simple: You need merely to have two key lines `[encode]` and `[clear]` in the message body.

Consider the following outgoing message:

```
Joe,
Remember that talk we had about Amy?  Well, I talked to my manager
about it and he said...
uhh...better encrypt this...the usual ‘key’...
[encode]
He said that Amy was having family problems and that it had been
affecting her work.
Given this, I went and talked to her, and told her I was sorry for
getting angry.  She said that she understood.
We’re friends again!!
[clear]
Exciting stuff, eh?
                                             Mike
```

While this is obviously quite readable while being typed into the editor, as soon as the message is confirmed as wanting to be sent, the *elm* mailer prompts with:

```
Enter encryption key: @
```

and accepts a key (a series of 8 or less characters) without echoing them to the screen. After entry, it will ask for the same key again to confirm it, then *poof* it will encrypt and send the mail.

If you have the *copy* option enabled, the program will save your copy of the message encrypted too. (This is to ensure the privacy and security of your mail archive, too.)

If the mailer doesn’t ask for the encryption key, it’s because you don’t have the `[encode]` entered as the first 8 characters of the line. It MUST be so for this to work!!

On the other end, a person receiving this mail (they must also be using *elm* to receive it, since this mailer has a unique encryption program) will be reading the message and then suddenly be prompted:

```
Enter decryption key: @
```

and will again be asked to re-enter it to confirm. The program will then on-the-fly decrypt the mail and display each line as it is decoded. The `[clear]` line signifies that the block to encrypt is done.

---

3. Unfortunately, at many non-US sites, it’s quite probable that you won’t be able to use this feature since you won’t have the *crypt()* library available due to licensing restrictions.

Note that it is not possible currently to *pipe* or *print* encrypted mail.

The other option on outgoing mail is the ability to specify what section of the message you want to have archived (assuming *copy* is enabled) and what section you don't. This is most useful for sending out source file listings and so on.

To indicate the end of the section that should be saved in the archive, you need merely to have the line

```
[nosave]
```

or

```
[no save]
```

appear by itself on a line. This will be removed from the outgoing mail, and will indicate the last line of the message in the saved mail. Other than this, the saved mail is identical to the outgoing mail.

## 6. Customized header lines

The mailer provides a facility for including customized header lines in the messages you send. If you have an *.elm/elmheaders* file, the mailer will include its contents immediately after the regular headers of all outbound mail. The mailer supports use of the backquote convention in this file. Here's a typical *.elm/elmheaders* file.

```
Organization: Hewlett-Packard Laboratories
Phone: (415)-555-1234
Operating-System: `uname -srv`
```

These lines will appear after all other header lines in the message.

## 7. Commands

This section will discuss each command in the *elm* program in more detail than above, including the prompts the user can expect upon executing the command, the meaning of different options, etc.

? Help. This command used once puts you in the *help* mode, where any key you press will result in a one-line description of the key. Pressed again at this point will produce a two page summary listing each command available. '.' will leave the help mode and return you to the main menu level.

<space> Display the current message. <space> is useful for reading through a mail folder. When issued from the index screen, it will display the first screen of the current message; and then when issued while in the builtin pager, it will page through the message to the end; and then when issued at the end of a message (with either the builtin pager or an external pager), it will display the first screen of the next message not marked for deletion.

<return> Display the current message. <return> behaves somewhat differently from <space>. When issued while in the builtin pager, it will scroll the current message forward one line, and then when issued at the end of a message (with either the builtin pager or an external pager), it will redisplay the first screen of the the *current* message. The latter is useful in case you have issued a non-pager command while in the builtin pager and want to restart the display of the current message.

! Shell. This allows you to send a command to the shell without leaving the program. (Note it is possible for the person who installed Elm on your system to disable this feature.)

| Pipe. This command allows you to pipe the current message or the set of *tagged* messages through other filters as you desire. The shell used for the entire command will be either the one specified in your *.elm/elmrc* file, or the default shell (see above).

Pattern match. This command, on the top level, allows the user to search through all the *from* and *subject* lines of the current folder starting at the current message and continuing through the end. If the first character of the pattern is a '/', then the program will try to match the specified pattern against *any* line in the folder. Again, this works from one after the current message through the end. Both searches are case insensitive.

- or <left> Display the previous page of the message index.

+ or <right>
Display the next page of the message index.

<number><return>
Specify new current message. When you type in any digit key *elm* will prompt "Set current to : n", where 'n' is the digit entered. Enter the full number and terminate with <return>. Note that changing the current message to a message not on the current page of headers will result in a new page being displayed.

< Scan message for calendar entries. A rather novel feature of the *elm* mailer is the ability to automatically incorporate calendar/agenda information from a mail message into the users calendar file. This is done quite simply; any line that has the pattern

-> *calendar entry*

will be automatically added to the users *calendar* file (see the *calendar* option of the *.elm/elmrc* file) if the '<' command is used.

For example, let's say we had a message with the text:

```
Regardless of that meeting, here's the seminar stuff:
-> 8/03 3:00pm: AI Seminar with Ira Goldstein of HP Labs
```

then using the '<' command would add the line:

```
8/03 3:00pm: AI Seminar with Ira Goldstein of HP Labs
```

to the users *calendar* file.

a Alias. The alias system is a way by which more complex mail addresses can be shortened for the mail user. For example:

```
joe, bleu   =  Joe Bleu  =   joe@hpfcla.SSO.HP.COM
```

which allows mail to 'joe' or 'bleu' with the system expanding the address properly. As is obvious, this saves remembering complex addresses. A more detailed discussion can be found in either the section entitled *The Alias System* in this document or *The Elm Alias System Users Guide*.

b Bounce mail. This "remails" mail to someone else in such a way as to make the return address the original sender rather than you. (The *forward* command is similar, but it makes the return address *you* rather than the original sender.)

C Copy to folder. This command copies the current message or set of tagged messages to a folder. If there is anything in the folder currently the message or messages are appended to the end, otherwise the folder is created containing only the newly copied message. The prompt for this command is ‘Copy to folder: ’. A response of <return> cancels the command and returns the user to the system prompt. The usual filename metacharacters are available, too. That is, this command expands filenames with ‘~’ being your home directory and ‘=’ being your *maildir* directory, if defined. This command also allows you to use ‘>’ for your *receivedmail* folder, ‘<’ for your *sentmail* folder, ‘.’ for the last folder you saved or copied a message to and ‘@alias’ for the default folder for ‘alias’. If you use a shell wildcard in the file or folder name, you will be given a list of all files or folders which match the wildcard. Elm uses your shell to find the names, so whatever wildcards you are used to can work. Finally, you can also enter ‘?’ at the prompt to get some detailed help.

c Change folder. Specifying this command allows the user to change the folder that is currently being read. This is intended for perusal and reply to previously archived messages. The prompt is ‘Name of new folder: ’ and entering <return> cancels the operation, while entering a filename causes the program to read that file as the new folder, if possible. This command expands filenames with ‘~’ being your home directory and ‘=’ being your *maildir* directory, if defined. This command also allows you to use ‘!’ as an abbreviation for you incoming mailbox, ‘>’ for your *receivedmail* folder, ‘<’ for your *sentmail* folder, ‘.’ for the last folder you saved or copied a message to and ‘@alias’ for the default folder for ‘alias’. If you use a shell wildcard in the file or folder name, you will be given a list of all files or folders which match the wildcard. Elm uses your shell to find the names, so whatever wildcards you are used to can work. Finally, you can also enter ‘?’ at the prompt

d, u Delete and Undelete. Neither of these two commands have any prompts and indicate their action by either adding a ‘D’ to the current message index entry (indicating deletion pending) or removing the ‘D’ (indicating that the message isn’t set for deletion).

<control>-D
This command allows you to easily mark for deletion all messages that have a specific pattern. After <control>-D is pressed, the program will prompt for the string to match (currently it only matches either the *from* or *subject* lines of the message).

<control>-U
This is the direct opposite command to the previous — all messages that match the specified pattern can have any mark for deletion removed by using this command.

e Edit mailbox. This allows you to modify the current mail file at a single keystroke. This is mostly useful for editing down messages before saving them. Modifying headers should be done with extreme caution, as they contain routing information and other vital stuff for full functionality.

f Forward. Allows the user to forward the current message to another user. This copies the message into the edit buffer and allows the user to add their own message too. The prompt is ‘Forward to:’ and will expand an alias if entered. (See also *bounce*, above.)

Elm will ask you if you want to edit the message before sending it. If you answer ‘yes’, Elm will prepend your prefix string to each line of the message, and let you edit the result. If you do not want the prefix string on each line, answer ‘no’; you will have another chance to edit the message when you get to the ‘send’ menu. (See also the ‘elmrc’ section, under *prefix*.)

g Group reply. Identical to *reply* below, except that the response is mailed to *all recipients* of the original message (except yourself — see the *alternatives* option for your *.elm/elmrc* file above).

h Display the current message with all headers intact. When you display a message with other commands, certain header lines are formatted and others discarded (according to the *weedlist* parameter in your *.elm/elmrc* file).

i Return to the index screen, when issued in the builtin pager or at the end of a message (with either the builtin pager or an external pager).

j or <down>, k or <up>
These four keys work similarly to what they would do in *vi* or any of the other (precious few) screen oriented programs. The 'j' and <down> keys move the current message pointer down to the next message skipping over any marked deleted (going to the next page if needed) and the 'k' and <up> keys move the current message pointer back to the previous message skipping over any marked deleted (also changing pages if needed)

J, K These two keys work similarly to their lower case counterparts, except that they don't skip over deleted messages.

l Limit. This feature allows you to specify a subset of the existing messages to be dealt with. For example, let's say we had a folder with four hundred messages in it, about four or five different subjects. We could then limit what we're dealing with by using the *limit* command. Pressing 'l' would result in the prompt:

```
Criteria:
```

to which we could answer `subject` *string*, `from` *string* or `to` *string*. In our example, we could use `subject programming` as a criteria for selection. Once we've limited our selections down, the screen will be rewritten with just the selected messages and the top line will change to have a message like:

```
Folder is "=elm" with 92 shown out of 124 [Elm 2.4]
```

We can further limit selections by entering further criteria, each time using the *limit* option again.

To clear all the criteria and get back to the 'regular' display, simply enter *all* as the limiting criteria. It should be noted that the selection based on "to" isn't fully implemented for this version, so it is recommended that users stay with "subject" and "from" as the basis for their criteria.

m Mail. Send mail to a specified user. The prompt that is associated with this command is 'Send mail to :'. Entering an alias name results in the full address being rewritten in parenthesis immediately. This prompt is followed by 'Subject:' which allows the user to title their note. The final field is 'Copies to: ', which allows other people specified to receive "carbon copies" of the message. (See the *askcc* option of the *.elm/elmrc* file above.) Upon entering all three items the editor is invoked and the message can be composed.

n Next message that is not marked deleted: useful for displaying successive messages in a folder. When issued from the index screen, it displays the current message, and then when issued while in the builtin pager or at the end of a message (with either the builtin pager or an external pager), it will display the first screen of the next message not marked for deletion.

o Options. This full-screen display allows you to alter the settings of a number of parameters, including the current sorting method, the method of printing files, the calendar file, the save file, and so on. It's self-documenting (where have you heard *that* before?) so isn't explained in too much detail here.

p Print. This allows you to print out the current message or the tagged messages to a previously defined printer. (See the section on the *.elm/elmrc* discussing the *print* variable.)

q Quit. If you in the pager, you are returned to the index screen. If you are on the index screen, *elm* quits altogether. However, if you have the option *ask* set, *elm* first prompts you for the disposition of the messages in the current folder. If any messages are marked for deletion, it will ask if you want them deleted. If the current folder is your incoming mailbox, you will also be asked if read messages should be stored in your *receivedmail* folder, and if unread messages should be kept in the incoming mailbox. The default answers to these questions are set by the *.elm/elmrc* options *alwaysdelete*, *alwaysstore*, and *alwayskeep*. However, if you elect to not store your read messages (i.e. keep them) it is presumed you want to keep your unread messages, too.

Q Quick quit. This behaves similar to the 'q' command except that you are never prompted for answers to the message disposition questions. *Elm* will dispose of messages according to the values you have set for *alwaysdelete*, *alwaysstore*, and *alwayskeep* in your *.elm/elmrc* file.

r Reply. Reply to the sender of the current message. If the *autocopy* flag is OFF in your *.elm/elmrc* file, the program will prompt "Copy message? (y/n)" to which the user can specify whether a copy of the source message is to be copied into the edit buffer, or not. If copied in, all lines from the message are prepended with the *prefix* character sequence specified in your *.elm/elmrc* file.

s Save to folder. This command is like the 'copy' command, except that the saved messages are marked for deletion, and that if you are saving just the current message, the current message pointer is incremented afterwards (see the *resolve* option in the *.elm/elmrc* file above). This command expands folder names with '~' being your home directory and '=' being your *maildir* directory, if defined. This command also allows you to use '>' for your *receivedmail* folder, '<' for your *sentmail* folder, '.' for the last folder you saved or copied a message to and '@alias' for the default folder for 'alias'.

t Tag. Tag the current message for a later operation[4].

<control>-T
Tag all messages containing the specified pattern. Since *tagging* messages can occur on screens other than the one being viewed, the *elm* system will first check to see if any messages are currently *tagged* and ask you if you'd like to remove those tags. After that, it will, similar to the *<control>-D* function, prompt for a pattern to match and then mark all messages that contain the (case insensitive) pattern in either the *from* or *subject* lines.

x Exit. This leaves *elm* discarding any changes to the mailbox. If changes are pending (such as messages marked for deletion) a prompt is made to confirm discarding the changes. If confirmed, no messages are deleted and the statuses of messages are unchanged. That is, any messages that were new will remain new instead of being noted as old, and any messages that were read for the first time will be again noted as unread.

X Exit immediately. This leaves *elm* in the quickest possible manner without even prompting about discarding the changes to the mailbox. No messages are deleted and the statuses of messages are unchanged. That is, any messages that were new will remain new instead of being noted as old, and any messages that were read for the first time will be again noted as unread.

4. Currently only *pipe*, *print*, and *save* support this.

When you are about to send of a message under the *forward*, *mail*, or *reply* commands (see above), a small menu of the following options appears:

c Specify folder for saving a copy to. This allows you to override the *copy*, *forcename* and *savename* options of your *.elm/elmrc* file. It prompts you for the name of the folder where a copy of the outgoing message is to be saved. The default displayed is taken from those three *.elm/elmrc* options and can be changed. This command also allows you to use '>' for your *receivedmail* folder and '<' for your *sentmail* folder, and '=?' to mean ''conditionally save by name'' and '=' to mean ''unconditionally save by name''. (See the *savename* option above for details on saving by name.) (Since you could next enter the *edit headers* command and change the recipients of your message, the name of the folder under the two ''save by name'' options is not established until you enter the *send* command.) If you use a shell wildcard in the file or folder name, you will be given a list of all files or folders which match the wildcard. Elm uses your shell to find the names, so whatever wildcards you are used to can work. You can also enter '?' at the prompt to some help about saving.

e Edit message (or form). Entering this command will allow you to edit the text of your message or form.

f Forget. This gets you out of sending a message you started. If you are in send-only mode, the message is saved to the file *Canceled.mail* in your home directory. Otherwise it can be restored at the next *forward*, *mail*, or *reply* command during the current session of *elm*. After issuing one of those commands you will be prompted with ''Recall last kept message?''

h Edit headers. This puts you into the *header editing mode*, whereby you can edit to any of the various headers of your message. Like the options screen, it's self-documenting, so it isn't explained in too much detail here.

i Run ispell (or some other configured spelling correction program). The current message is run through an interactive spelling correction program if one is available. The default spelling program is the GNU ispell program unless changed by the person who installed Elm on your system.

m Make form. This converts the message you have edited into a form. (See *The Elm Forms Mode Guide* for more details.)

s Send. This sends the message as is without any further ado.

## 8. Using Elm with ''editor = none''

The *Elm* program also supports a builtin editor for simple message composition that is very (very) similar to the simple line editor available from the *Berkeley Mail* system.

To access it, you need merely to specify ''*editor=none*'' in your *.elm/elmrc* file. With that, any messages to be composed that don't already have text in the buffer (e.g. no reply with the text included, etc), will use this editor.

From the builtin editor, the following options are available for use. Each command here is prefixed with a '~'. You can specify a different ''escape'' character in your *.elm/elmrc* file, if you desire (see above).

~? Print a brief help menu.

~b Change the Blind-Carbon-Copy list.

~c Change the Carbon-Copy list.

~e Invoke the Emacs editor on the message, if possible.

~f Add the specified message or current message.

~h Change all the available headers (To, Cc, Bcc, and Subject).

~m Same as '~f', but with the current 'prefix'.

~o Invoke a user specified editor on the message.

~p Print out the message as typed in so far.

~r Include (read in) the contents of the specified file.

~s Change the Subject line.

~t Change the To list.

~v Invoke the Vi visual editor on the message.

~< Execute the specified UNIX command, entering the output of the command into the editor buffer upon completion. (For example "~< who" will include the output of the *who* command in your message.)

~! Execute a UNIX command if one is given (as in "~!ls") or give the user a shell (either from their shell setting in their *.elm/elmrc* file or the default).

~~ Add a line prefixed by a single '~' character.

A useful note is that the '~f' and '~m' commands invoke the *readmsg* command, so you can pass parameters along too. For example, if we wanted to include a message from Joe, without any headers, prefixed, we could use:

```
    ~m -n Joe
to accomplish the task.

To learn more about how they work, try 'em!
```

## 9. The Alias System

As mentioned previously, there exists in the *elm* system a set of aliases that associate an arbitrary word (such as a persons name) to a complex address or group. The advantages are readily apparent; rather than remembering an address of the form:

```
host1!host2! ... !hostN!user
```

the user merely has to remember a single word.

Two alias tables are available for a each user within *elm*, namely the system alias file and the user's alias file. The system alias file is created and maintained (by the system administrator) by editing the file *SYSTEM_ALIASES* as defined in the 'sysdefs.h' file (see *The Elm Configuration Guide*) and as described in the documentation with the *newalias* command, then running the *newalias* program.

An individual user can also have an alias file which works in conjunction with the system aliases. To do this, he needs merely to enter the alias menu system and create aliases with the 'a' (*Alias current message*) or 'n' (*Make new alias*) commands. Alternatively, the user can peruse the documentation for the *newalias* command and create a file as indicated therein. After executing the program, the aliases will be available for use from within *elm*.

Please refer to *The Elm Alias Users Guide* for more helpful hints and so on.

Within *elm*, however, the alias system acts as an entirely different program, with its own display, own commands, and own mini-menu. The aliases are presented in a list similar to the main screen. The menu replaces the standard mini-menu with:

```
--------------------------------------------------------------------------------

          Alias commands:  ?=help, <n>=set current to n, /=search pattern
  a)lias current message, c)hange, d)elete, e)dit aliases.text, f)ully expand,
     l)imit display, m)ail, n)ew alias, r)eturn, t)ag, u)ndelete, or e(x)it

Alias: @
--------------------------------------------------------------------------------
```

The commands are:

? Help. This command used once puts you in the *help* mode, where any key you press will result in a one-line description of the key. Pressed again at this point will produce a two page summary listing each command available. '.' will leave the help mode and return you to the alias menu level.

<space> or <return> or v
Display the current alias address. The alias address is displayed below the alias menu. This command allows you to verify the address for a person or the contents of a group alias.

Pattern match. This command allows the user to search through all the *alias* and *username* entries for the alias list starting at the current alias and continuing through the end. If the first character of the pattern is a '/', then the program will also include the *comment* and the fully expanded *address* fields in the search. The search is case insensitive. This allows the user to find a specific alias in the situation where there are a large number of aliases.

a Alias current message. This allows the user to create an alias that has the return address of the current message as the address field of the alias. It prompts for a unique alias name. If the alias name is not unique, you will be asked if you wish to replace the existing alias. For further information, please see *The Elm Alias System Users Guide*.

c Change current user alias. This will prompt for changes to the current names and address. If other aliases are tagged you will be asked if you want to create a group alias from the tagged aliases. The original alias will be replaced with the new information in your individual aliases.text file (*$HOME/.elm/aliases.text*) and then added to the database (upon the next alias resync). Aliases that have been changed are marked with an 'N' (for new) until the database is updated.

d, u Delete or undelete an alias. This allows the user to mark an alias for deletion in the same fashion as on the main menu. The deletions are not actually made until the user *r)eturns* to the main menu or resyncs the display (with the '$' command). Deletions on system aliases are not allowed. These commands (plus the <control>-D and <control>-U versions) behave identically to their main menu counterparts.

e Edit the *.elm/aliases.text* file. The user alias file is edited using the editor defined in *editor* in your *.elm/elmrc* file. *newalias* is run after the edit.

f Display *fully expanded* alias. The currently highlighted alias is fully expanded and displayed to the user. This is most useful when working with group alias in the alias database.

l Limit the display. You can limit the display by alias type (person/group or user/system) or by search pattern on name or alias. Otherwise, this works exactly like the limit command on the main menu.

m Send mail to the current alias. A new mail message will be sent to the person or group specified by the selected alias. If aliases are currently tagged the message will be mailed to the person(s) and/or group(s) specified by the tagged aliases. Tags are cleared after mailing the message.

n Make a new user alias. This will prompt for a unique alias name and then for an address. If the alias name is not unique, you will be asked if you wish to replace the existing alias. If aliases are tagged you will be asked if you want to create a group alias from the tagged aliases. The information provided will be added to your individual aliases.text file (*$HOME/.elm/aliases.text*) and then added to the database (upon the next alias resync).

r or q or i
Return. Return to the main level of the *elm* program. Any pending deletions are processed and newalias is run to update the database. New additions are handled at this time as well.

R or Q
Quick return. This behaves like the 'r' command except that you are never prompted for answers to alias disposition questions. *Elm* will dispose of aliases according to the values you have set for *alwaysdelete* in your *.elm/elmrc* file.

t Tag. Tag the current alias for a later operation[5].

x Exit alias menu. Exits the alias menu without processing any deletions. Aliases marked for deletion are unmarked. Additionally, newalias is not run, even if alias additions have been made.

Additionally, the movement keys ('j', 'k', '+', '-', etc.) work in the same fashion as on the main menu. (See the previous section on *Commands* in this guide.)

## 10. Elm and Signals

*Elm* can handle several standard UNIX *signals* to do some special processing. *Signals* are interrupt messages sent from one program to another. No detailed messages are sent, but a properly configured program (such as *elm*) can watch for and handle these signals.

---

5. Currently only *mail*, *change*, and *new alias* support this.

In particular, *elm* watches for the following *signals* and takes these actions:

*ALRM* This is the alarm clock signal or time warning. Elm uses this to wake itself up periodically and check for new mail.

*HUP* This is the hangup notice. It means that the terminal/modem/whatever which you have been using with *elm* has become detached from the system where *elm* was running. When *elm* gets this signal, it aborts all the pending operations and exits leaving your mailbox unchanged.

*USR1* This is the first user-definable signal. When *elm* gets this signal, it receives any pending mail, performs all the pending operations (deletes), and exits leaving all unread mail marked as new. This is the same as giving the *$* and *X* commands.

*USR2* This is the second user-definable signal. When *elm* gets this signal, it receives any pending mail, performs all the pending operations (deletes), and exits leaving all unread mail marked as old. This is the same as giving the *$* and *Q* commands.

You would only use these signals yourself under the most unusual circumstances. For example, suppose you were using *elm* to read mail on *host_1*. You have many messages, most of which have been read and filed (and therefore deleted), or simply deleted. You have not yet resynchronized your mailbox (*$* command). Now you go to lunch. On your return, you're stopped at the door and told to take care of an emergency. You go to another part of the building, and want to read your mail to see what the emergency is – but you can't, because *elm* is still running at your desk.

What you really want is one of the scenarios given in the description of *HUP*, *USR1* or *USR2*. Use your local *ps* command to find out what the *process number* of your *elm* session is. Then give the command

```
kill -XXX process_number
```

where *XXX* is either *HUP*, *USR1* or *USR2* and *process_number* is the process number for your remote *elm* session. Your remote session will terminate with the actions noted above.

## 11. Expert Mail Users and Debugging the Mailer

There are some additional facilities available in the *elm* mailer for those people who are knowledgeable about mail protocols, or trying to debug/track down a problem.

The 'h' *headers* command at the outermost level of the mailer will display the current message ignoring the current setting of the 'weed' option. This is most useful for answering questions of the form "I wonder what this guy put in his header?" and such. This command does not show up on the mini-menu because it is somewhat esoteric, but it does appear on the '?' help screen (can you find it there, though?).

The '@' command at the outermost level of the mailer will output a screen of debugging information, including the number of lines and offsets of each of the messages in the current mailbox.

The '#' command at the outermost level of the mailer will display the entire stored 'record structure' for the current message.

The '%' command will display the full computed return address of the current message.

Starting up *elm* with the "-d" debug option will create a file called *ELM:debug.info* in your home directory and contain a wealth of useful information (to me, at least!) to aid in tracking down what errors are occurring and why.

If there are any problems with the mailer, please try to recreate the error with the debug option enabled and set to the highest level (11) before sending defect reports my way.

One final note: all error names reported by the program are documented in the AT&T System V Interface Definition Reference Manual in *errno*(2).