XEmacs FAQ

Frequently asked questions about XEmacs
Last Modified: $Date: 1995/09/21 17:03:34 $

Ben Wing <wing@netcom.com>
Chuck Thompson <cthomp@cs.uiuc.edu>
Anthony Rossini <rossini@stat.sc.edu>

XEmacs FAQ

Archive-name: xemacs-faq Posting-frequency: monthly Last-modified: $Date: 1995/09/21 17:03:34 $ URL1: http://xemacs.cs.uiuc.edu/ URL2: http://www.stat.sc.edu/~rossini/doc/xemacs

Current-Editor: Anthony Rossini <rossini@stat.sc.edu>

This is a list of frequently asked questions for XEmacs users and maintainers.

You can obtain the latest version of this file by anonymous FTP from site ‘`ftp.cs.uiuc.edu:/pub/xemacs/faq/` where FORMAT is text, texinfo (the master copy), postscript, dvi, html, info. Note that the html version has 2 files.

Thanks to the many who contributed, especially to JWZ <jwz@netscape.com> for starting this whole wonderful mess... Special thanks to Tim Geisler <tmgeisle@faui80.informatik.uni-erlangen.de> for the initial texinfo version and to Giacomo Boffi <sboff@hp735.stru.polimi.it> for the initial HTML version. Other contributors (many left out, unfortunately, due to an old mail-purge) include:

- Juergen Nickelsen <nickel@prz.tu-berlin.de>
- Evelyn Ginsparg <ginsparg@adra.com>
- d3h554@foghorn.pnl.gov
- Marty Hall <hall@aplcenmp.apl.jhu.edu>
- Richard Caley <rjc@cogsci.ed.ac.uk>
- fcg@philabs.Philips.COM
- Stig <stig@hackvan.com>
- Arup Mukherjee <arup+@cmu.edu>
- Kevin R. Powell <powell@csl.ncsa.uiuc.edu>
- Eric Eide <eeide@cs.utah.edu>
- William G. Dubuque <wgd@martigny.ai.mit.edu>
- Chris Flatters <cflatter@nrao.edu>
- John A. Turner <turner@lanl.gov>

1 1. Introductory Questions

1.1 1.1. Preface

This is the introduction to a list of frequently asked questions (FAQ's) about XEmacs with answers. This article contains a listing of the questions; subsequent articles contain the questions and answers.

This is not a substitute for general Emacs questions, and actually includes nothing that might help a novice learn to use Emacs or XEmacs. For that, check out the regular Emacs lists, the tutorial inside of XEmacs (look on the Help menu), or the O'Reilly book *Learning GNU Emacs*. This FAQ focuses on specific issues regarding XEmacs. If you don't find the answer here, perhaps it really is a more general question, and check the GNU Emacs FAQ for more information, as well.

The FAQ list is posted to reduce the noise level in the '`comp.emacs.xemacs`' newsgroup which results from the repetition of frequently asked questions, wrong answers to these questions, corrections to the wrong answers, corrections to the corrections, debate, name calling, etc. Also, it serves as a repository of the canonical "best" answers to these questions. However, if you know a better answer or even a slight change that improves an answer, please tell me!

If you know the answer of a question is in the FAQ list, please reply to the question by e-mail instead of posting. Help reduce noise!

Please suggest new questions, answers, wording changes, deletions, etc. The most helpful form for suggestions is a context diff (i.e. the output of '`diff -c`'). Include '`FAQ`' in the subject of messages sent to us about the FAQ list.

Please do not send questions to us just because you do not want to disturb a lot of people and you think we would know the answer. We do not have time to answer questions individually and keep up with everything else we have to get done.

Many thanks need to go to all contributors on the old alt.lucid-emacs.* and current comp.emacs.xemacs groups. This would never have been collected without you.

Any directories given are usually in reference to the base directory formed by unpacking XEmacs.

1.2 1.2. What is XEmacs?

An alternative to GNU Emacs, originally based on an early alpha version of FSF's version 19. XEmacs was known as Lucid Emacs through version 19.10. Almost all features of GNU Emacs are supported in XEmacs (the ones that aren't supported are generally implemented in a better way in XEmacs). The maintainers of XEmacs actively track changes to GNU Emacs while also working to add new features never before seen in Emacs.

1.3 1.3. What is the current version of XEmacs?

The current version is 19.13, released on September 1, 1995.

1.4 1.4. Where can I find it?

The canonical source and binaries is found via anonymous FTP at '`ftp.cs.uiuc.edu:/pub/xemacs`'.

1.5 1.5. Why Another Version of Emacs?

For a detailed description of the differences between GNU Emacs and XEmacs, and a detailed history of XEmacs, see the file '`etc/NEWS`' in the source distribution. However, here is a list of some of the reasons why we think you might consider using it:

- It looks nicer
- The XEmacs maintainers are generally more receptive to suggestions than the GNU Emacs maintainers
- Many more bundled packages than GNU Emacs, all properly integrated with XEmacs
- Binaries are available for many common operating systems
- Face support on TTY's
- A built-in toolbar
- Better Motif compliance
- Some internationalization support (full MULE support starting with 19.14)
- Variable-width fonts
- Variable-height lines

- Marginal annotations
- ToolTalk support
- XEmacs can be used as an Xt widget, and can be embedded within another application
- Horizontal and vertical scrollbars (using real toolkit scrollbars)
- Better APIs (and performance) for attaching fonts, colors, and other properties to text
- The ability to embed arbitrary graphics in a buffer
- Completely compatible (at the C level) with the Xt-based toolkits

1.6 1.6. What do `M-x`, GNU, etc. mean?

This is really a general Emacs question, but a table of some common terms/acronyms is provided here for your convenience. See the *GNU Emacs FAQ* for a more complete list.

‘`BLAT FOOP`’	Historical XEmacs error message
‘`C-x`’	*C-x* means that the `CONTROL` key and the `X` key should be pressed together, like when you use the `SHIFT` key.
‘`E-Lisp`’	Same as *Emacs-Lisp*.
‘`Emacs-Lisp`’	The dialect of Lisp supported by Emacs and XEmacs. You use this to customize XEmacs.
‘`FAQ`’	Frequently asked question(s).
‘`FSF`’	The Free Software Foundation.
‘`GNU`’	*GNU* refers to products written by the Free Software Foundation.
‘`JWZ`’	Jamie Zawinski <jwz@netscape.com>, the former maintainer of Lucid Emacs (which XEmacs evolved from).
‘`M-x`’	*M-x* means that the `META` key and the `X` key should be pressed together, like when you use the `SHIFT` key. Sometimes the `META` key is labelled ‘`Alt`’ or with a diamond. Some keyboards don’t have a `META` key at all; then you can get the equivalent by typing the `ESC` key followed by the `X` key.
‘`RMS`’	Richard Stallman <rms@gnu.ai.mit.edu>, the author of GNU Emacs.
‘`VI`’	An editor used by those heretics that don’t subscribe to the Emacs religion.

1.7 1.7. Where can I get help for using XEmacs?

Probably the easiest way, if everything is installed, is to use info, by pressing `C-h i`, or selecting '`Emacs Info`' from the Help Menu. Also, `M-x apropos` will look for commands for you.

Try reading this FAQ, examining the regular GNU Emacs FAQ (which can be found with the Emacs 19 distribution) as well as at '`http://www.eecs.nwu.edu/emacs/faq/`' and reading the Usenet group '`comp.emacs.xemacs`'. If that does not help, try posting your question to '`comp.emacs.xemacs`'. If you cannot post or read Usenet news, there is a corresponding mailing list which is available. It can be subscribed to by sending mail to

```
xemacs-request@cs.uiuc.edu
```

for subscription information and

```
xemacs@cs.uiuc.edu
```

to send messages to the list.

To cancel a subscription, YOU MUST use the xemacs-request address.

Bug reports should be sent to the same locations.

1.8 1.8. Where is the mailing list archived?

The mailing list is archived in the directory '`ftp.cs.uiuc.edu:/pub/xemacs/mlists/`'.

1.9 1.9. What is InfoDock, how does it relate to XEmacs, and how can I obtain it?

InfoDock is an integrated productivity toolset, mainly aimed at technical people. It is built atop the XEmacs variant of GNU Emacs and so has all of the power of Emacs, but with an easier to use and more comprehensive menu-based user interface. The next section describes how it differs from XEmacs and GNU Emacs from the Free Software Foundation.

The quickest way to get a feel for InfoDock is to browse the InfoDock Manual, especially the section on tools. This will help you decide whether or not to download InfoDock for local use. This manual is available in gzipped Postscript form, alongside the InfoDock distribution. (See below for FTP retrieval instructions.)

InfoDock is aimed at people who want a free, turn-key productivity environment. Although InfoDock is customizable, it is not intended for people who like basic versions of Emacs which need to be customized extensively for local use; standard Emacs distributions are better for such uses. InfoDock is for those people who want a complete, pre-customized environment in one package, which they need not touch more than once or twice a year to update to new revisions.

InfoDock is pre-built for SPARCstations running SunOS V4 or V5 (Solaris). It is intended for use on a color display, although most features will work on monochrome monitors. Simply unpack InfoDock according to the instructions in the ID-INSTALL file and you are ready to run.

The InfoDock Manual is concise, yet sufficient as a user guide for users who have never used an Emacs-type editor before. For users who are already familiar with Emacs, it supplements the information in the GNU Emacs Manual.

InfoDock menus are much more extensive and more mature than standard Emacs menus. Each menu offers a '`Manual`' item which displays documentation associated with the menu's functions.

Three types of menubars are provided:

1. An extensive menubar providing access to global InfoDock commands.
2. Mode-specific menubars tailored to the current major mode.
3. A simple menubar for basic editing to help novices get started with InfoDock.

Most modes also include mode-specific popup menus. Additionally, region and rectangle popup menus are included.

- '`Hyperbole`', the everyday information manager, is a core part of InfoDock. This provides context-sensitive mouse keys, a rolodex-type contact manager, programmable hypertext buttons, and an autonumbered outliner with embedded hyperlink anchors.
- '`PIEmail`', the prototype Personalized Information Environment Mail Tool, is included.
- The '`OO-Browser`', a multi-language object-oriented code browser, is a standard part of InfoDock.

InfoDock saves a more extensive set of user options than other Emacs versions.

InfoDock inserts a useful file header in many file types, showing the author, summary, and last modification time of each file. A summary program can then be used to summarize all of the files in a directory, for easy MANIFEST file creation.

Your working set of buffers is automatically saved and restored (if you answer yes to a prompt) between InfoDock sessions.

Refined color choices for code highlighting are provided for both dark and light background display frames.

The `C-z` key prefix performs frame-based commands which parallel the `C-x` key prefix for window-based commands.

The Smart Menu system is included for producing command menus on dumb terminals. (InfoDock does not yet run on dumb terminals but will in 1995.)

Lisp libraries are better categorized according to function.

Extensions and improvements to many areas of Emacs are included, such as: paragraph filling, mail reading with Rmail, shell handling, outlining, code highlighting and browsing, and man page browsing.

InfoDock questions, answers and discussion should go to the mail list '`infodock@hub.ucsb.edu`'. Use '`infodock-request@hub.ucsb.edu`' to be added or removed from the list. Always include your InfoDock version number when sending help requests.

InfoDock is available across the Internet via anonymous FTP. To get it, first move to a directory into which you want the InfoDock archive files placed. We will call this <DIST-DIR>.

```
cd <DIST-DIR>
```

FTP to ftp.cs.uiuc.edu (Internet Host ID = 128.174.252.1):

```
prompt> ftp ftp.cs.uiuc.edu
```

Login as '`anonymous`' with your own <user-id>@<site-name> as a password.

```
Name (ftp.cs.uiuc.edu): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password: -<your-user-id@your-domain>
230 Guest login ok, access restrictions apply.
```

Move to the location of the InfoDock archives:

```
ftp> cd pub/xemacs/infodock
```

Set your transfer mode to binary:

```
ftp> bin
200 Type set to I.
```

Turn off prompting:

```
ftp> prompt
Interactive mode off.
```

Retrieve the InfoDock archives that you want, either by using a '`get <file>`' for each file you want or by using the following to get a complete distribution, including all binaries:

```
ftp> mget ID-INSTALL
ftp> mget id-*
```

Close the FTP connection:

```
ftp> quit
221 Goodbye.
```

Read the '`ID-INSTALL`' file which you just retrieved for step-by-step installation instructions.

2 2. Compiling XEmacs

2.1 2.1. What is the best way to compile XEmacs with the netaudio system, since I have got the netaudio system compiled but installed at a weird place, I am not root. Also in the READMEs it does not say anything about compiling with the audioserver?

You should only need to add some stuff to the configure command line. To tell it to compile in netaudio support:

```
--with-sound=both
```

(Or '`--with-sound=nas`' if you don't want native sound support for some reason.) To tell it where to find the netaudio includes and libraries:

```
--site-libraries=WHATEVER
--site-includes=WHATEVER
```

Then (fingers crossed) it should compile and it will use netaudio if you have a server running corresponding to the X server. The netaudio server has to be there when XEmacs starts. If the netaudio server goes away and another is run, XEmacs should cope (fingers crossed, error handling in netaudio isn't perfect).

BTW, netaudio has been renamed as it has a name clash with something else, so if you see references to NAS or Network Audio System, it's the same thing. It also might be found at

```
ftp.x.org:/contrib/audio/nas/
ftp.ncd.com:/pub/ncd/technology/src/nas/
```

3 3. Problems running XEmacs / weird messages

3.1 3.1. Help! XEmacs just crashed on me!

First of all, don't panic. Whenever XEmacs crashes, it tries extremely hard to auto-save all of your files before dying. (The main time that this will not happen is if the machine physically lost power or if you killed the XEmacs process using '`kill -9`'.) The next time you try to edit those files, you will be informed that a more recent auto-save file exists. You can use `M-x recover-file` to retrieve the auto-saved version of the file.

Now, XEmacs is not perfect, and there may occasionally be times, or particular sequences of actions, that cause it to crash. If you can come up with a reproducible way of doing this (or even if you have a pretty good memory of exactly what you were doing at the time), the maintainers would be very interested in knowing about it. Post a message to '`comp.emacs.xemacs`' or send mail to '`xemacs@cs.uiuc.edu`'.

If at all possible, include a stack backtrace of the core dump that was produced. This shows where exactly things went wrong, and makes it much easier to diagnose problems. To do this, you need to locate the core file (it's called '`core`', and is usually sitting in the directory that you started XEmacs from, or your home directory if that other directory was not writable). Then, go to that directory and execute a command like

```
gdb `which xemacs` core
```

and then issue the command '`where`' to get the stack backtrace. (You might have to use '`dbx`' or some similar debugger in place of '`gdb`'. If you don't have any such debugger available, complain to your system administrator.)

It's possible that a core file didn't get produced, in which case you're out of luck. Go complain to your system administrator and tell him not to disable core files by default. (If you explicitly disabled core files, then double shame on you!)

3.2 3.2. When I try to use some particular option of some particular package, I get a cryptic error in the minibuffer.

If you can't figure out what's going on, try typing `ESC ESC` and issuing the command

```
(setq debug-on-error t)
```

and then try and make the error happen again. This will give you a backtrace that may be enlightening. If not, try reading farther down in this FAQ; if that fails, you could try posting to '`comp.emacs.xemacs`' (making sure to include the backtrace) and someone may be able to help.

3.3 3.3. I get tons of translation table syntax error messages during startup. How do I get rid of them?

There are two causes of this problem. The first usually only strikes people using the prebuilt binaries. The culprit in both cases is the file '`XKeysymDB`'.

- The binary cannot find the XKeysymDB file. The location is hardcoded at compile time so if the system the binary was built on puts it a different place than your system does, you have problems. To fix, set the environment variable '`XKEYSYMDB`' to the location of the XKeysymDB file on your system or to the location of the one included with XEmacs which should be at:
  `<xemacs_root_directory>/lib/xemacs-19.13/etc/XKeysymDB`
- The binary is finding the XKeysymDB but it is out-of-date on your system and does not contain the necessary lines. Either ask your system administrator to replace it with the one which comes with XEmacs (which is the stock R6 version and is backwards compatible) or set your '`XKEYSYMDB`' variable to the location of XEmacs's described above.

3.4 3.4. How can I avoid the startup warnings about deducing proper fonts?

This is highly dependent on your installation, but try with the following font as your base font for XEmacs and see what it does:

```
-adobe-courier-medium-r-*-*-*-120-*-*-*-*-iso8859-1
```

More precisely, do the following in your resource file:

```
Emacs.default.attributeFont: -adobe-courier-medium-r-*-*-*-120-*-*-*-*-iso8859-1
```

3.5 3.5. Help! I can not get XEmacs to display on my Envizex X-terminal!

Try setting the DISPLAY variable using the numeric IP address of the host you are running XEmacs from.

3.6 3.6. Why do I get weird messages about giftoppm and ppmdither not being found?

Because they are not. You are probably trying to read some HTML files, or using W3-mode as your Web client. To display inline pictures, this requires the above programs (which can be found in the netpbm package or its older version, pbmplus). The netpbm package can be found at '`ftp.x.org`', or any of its mirror sites. It can also be found at '`ftp.cs.uiuc.edu:/pub/xemacs/aux`'.

3.7 3.7. How can I avoid those messages about deleting excess backup files?

Try the following Emacs-Lisp:

```
(setq version-control t)
(setq kept-old-versions 0)
(setq kept-new-versions 8)
(if (string-match "XEmacs\\|Lucid" emacs-version)
   (setq trim-versions-without-asking t)
   (setq delete-old-versions t))
```

3.8 3.8. Help! XEmacs just locked up my X server on my Linux box!

There have been several reports of the X server locking up under Linux. In all reported cases removing speedo and scaled fonts from the font path corrected the problem. This can be done with the command 'xset'.

It is possible that using a font server may also solve the problem.

4 4. Customization – Emacs Lisp and the .emacs file

4.1 4.1. How can '`.emacs`' determine which of the family of Emacsen I am using?

To determine if you are currently running GNU Emacs 18, GNU Emacs 19, XEmacs 19, or Epoch, and use appropriate code, check out the example given in '`etc/sample.emacs`'. There are other nifty things in there as well! Alternatively, there is a package, '`emacs-vers.el`', available at an Emacs-Lisp archive near you; try searching

```
archive.cis.ohio-state.edu:/pub/gnu/emacs/elisp-archive/misc/
```

4.2 4.2. How can I detect a color display?

You can test the return value of the function `(device-class)`, as in:

```
(if (eq (device-class) 'color)
    (progn
      (set-face-foreground  'font-lock-comment-face "Grey")
      (set-face-foreground  'font-lock-string-face "Red")
      ....
      ))
```

4.3 4.3. How can I evaluate Emacs-Lisp expressions without switching to the *scratch* buffer?

```
(put 'eval-expression 'disabled nil)
```

This sets it so that hitting `ESC ESC` lets you type a single expression to be evaluated. This line can also be put into your '`.emacs`'.

4.4 4.4. If you put (setq tab-width 6) in your '`.emacs`' file it does not work! Is there a reason for this. If you do it at the EVAL prompt it works fine!! How strange.

Use setq-default, since tab-width is all-buffer-local.

4.5 4.5. How can I add directories to the load-path?

Here are two ways to do that, one that puts your directories at the front of the load-path, the other at the end:

```
;;; Add things at the beginning of the load-path
(setq load-path (cons "bar" load-path))
(setq load-path (cons "foo" load-path))

;;; Add things at the end
(setq load-path (append load-path '("foo" "bar")))
```

4.6 4.6. How to check if a lisp function is defined or not?

Use the following elisp:

```
(fboundp 'foo)
```

It's always a mistake, under all circumstances, to test 'emacs-version' or any similar variables, in case they are not bound, unless you do the above.

Instead, use feature-tests, such as featurep or boundp or fboundp, or even simple behavioural tests, eg (defvar foo-old-losing-code? (condition-case nil (progn (losing-code t) nil) (wrong-number-of-arguments t)))

There is an incredible amount of broken code out there which could work much better more often in more places if it did the above instead of trying to divine its environment from the value of one variable.

4.7 4.7. Can I force the output of (face-list) to a buffer other than the minibuffer since it is too wide to fit?

Evaluate the expression in the "*scratch*" buffer with point on the rightmost paren and typing `C-j`.

5 5. Customization – X resources

5.1 5.1. Where is a list of X resources?

Search through the NEWS file for "X Resources". A fairly comprehensive list is given after it.

In addition, an app-defaults file is supplied, '`etc/Emacs.ad`', listing the defaults. The file '`etc/sample.Xdefaults`' gives a set of defaults that you might consider. It is essentially the same as '`etc/Emacs.ad`' but some entries are slightly altered. Be careful about installing the contents of this file into your '`.Xdefaults`' (.Xresources) file if you use FSF GNU Emacs under X11 as well.

6 6. Changing textual fonts and colors

6.1 6.1. How do I set the text, menu and modeline fonts?

Note that you should use '`Emacs.`' and not '`Emacs*`' when setting face values.

In '`.Xdefaults`':

```
Emacs.default.attributeFont:  -*-*-medium-r-*-*-*-120-*-*-m-*-*-*
Emacs*menubar*font:           fixed
Emacs.modeline.attributeFont: fixed
```

This is confusing because modeline is a face, and can be found listed with all faces in the current mode by using M-x set-face-font (enter) ?. It uses the face specification of "attributeFont", while menubar is a normal X thing that uses the specification "font". With Motif it may be necessary to use "fontList" instead of "font". (Please rewrite this if you understand this better than me ... A general description of faces would be very useful. I am just saying that menubar is a FAQ because it's not a face and not listed.)

6.2 6.2. How can I set the background/foreground colors when highlighting a region?

You can change the face 'zmacs-region' either in your '`.Xdefaults`':

```
Emacs.zmacs-region.attributeForeground: firebrick
Emacs.zmacs-region.attributeBackground: lightseagreen
```

or in your '`.emacs`':

```
(set-face-background 'zmacs-region "red")
(set-face-foreground 'zmacs-region "yellow")
```

6.3 6.3. How can I set the most commonly used color options from my '`.emacs`' instead of from my '`.Xdefaults`'?

```
(set-face-background 'default "bisque")          ; frame background
(set-face-foreground 'default "black")           ; normal text
(set-face-background 'zmacs-region "red")   ; When selecting w/
                                                 ; mouse
(set-face-foreground 'zmacs-region "yellow")
(set-face-font  'default "*courier-bold-r*120-100-100*")
(set-face-background 'highlight "blue")          ; Ie when selecting buffers
(set-face-foreground 'highlight "yellow")
(set-face-background 'modeline "blue")           ; Line at bottom of buffer
(set-face-foreground 'modeline "white")
(set-face-font  'modeline "*bold-r-normal*140-100-100*")
(set-face-background 'isearch "yellow")          ; When highlighting while
                                                 ; searching
(set-face-foreground 'isearch "red")
(setq x-pointer-foreground-color "black")        ; Adds to bg color,
                                                 ; so keep black
(setq x-pointer-background-color "blue")         ; This is color you really
                                                 ; want ptr/crsr
```

6.4 6.4. Is there anyway to limit the number of available colors in the color map?

XEmacs 19.13 doesn't have such a mechanism (unlike netscape, or other color-hogs).

7 7. The modeline

7.1 7.1. How can I make the modeline go away?

(set-specifier has-modeline-p nil)

7.2 7.2. How do you have XEmacs display the line number in the modeline?

Add the following line to your '`.emacs`' file:

```
(setq line-number-mode t)
```

7.3 7.3. How do I get XEmacs to put the time of day on the modeline?

```
(display-time)
```

7.4 7.4. How can one change the color of the modeline based on the mode used? i.e. red for C mode, green for TeX mode etc.?

You can use something like the following:

```
(add-hook 'lisp-mode-hook
        '(lambda () (set-face-background 'modeline "red" (current-buffer))
                    (set-face-foreground 'modeline "yellow" (current-buffer))))
```

Then, when editing a Lisp file (ie went into Lisp mode), the modeline colors change from the default set in your '`.emacs`'. The change will only be made in the buffer you just entered (which contains the Lisp file you are editing) and will not affect the modeline colors anywhere else.

Notes:

- The hook is the mode name plus "-hook". Ie c-mode-hook, c++-mode-hook, emacs-lisp-mode-hook (ie your '`.emacs`' or a xx.el file), lisp-interaction-mode-hook (the *scratch* buffer), text-mode-hook, etc.
- Be sure to use add-hook, not (setq c-mode-hook xxxx), otherwise you will erase anything that anybody has already put on the hook.
- You can also do (set-face-font 'modeline `font`), eg (set-face-font 'modeline "*bold-r-normal*140-100-100*" (current-buffer)) if you wish the modeline font to vary based on the current mode.

7.5 7.5. With AUC-TeX, fast typing is hard because the current chapter/section/... is given in the modeline. How can I turn this off?

it's not auc-tex, it is func-menu. from func-menu.el:

```
(defvar fume-display-in-modeline-p t
```

8 8. The keyboard

8.1 8.1. What is the difference in in key sequences between XEmacs and GNU Emacs?

The real question might be rephrased as "When should one use the quoted list, vector, or escaped string representations of key sequences?" Is there any particular advantage to one representation over another?

>From Richard Mlynarik <mly@adoc.xerox.com>:

(meta a)	is a convenience shorthand for the sequence `[(meta a)]`. `(global-set-key 'a 'foo)` means the same thing as `(global-set-key '[a] 'foo)`. It could be argued that allowing such a shorthand just leads to sloppiness and bugs, but it's there, and it isn't likely to go away.
[(meta a)]	is The Right Thing. It corresponds in a one-to-one way with the internal representation of key-sequences in keymaps.
[Meta-a]	is typical FSF Emacs 19 brain damage. As is usual, an existing, functional design is ignored (XEmacs) and an incompatible and technically worse kludge is used.
"\ea"	is compatible with Emacs 18, but suffers from ASCII Seven-Bit Brain Damage. I also find it harder to read. Use this if you're trying to write code which works in every Emacs, but be aware that you can not express all Possible key-sequences (`control-9`, `f1`, etc.) using this.

8.2 8.2. How can I make XEmacs recognize the Alt key of my HP workstation as a Meta key?

Put the following line into a file and load it with xmodmap(1) before starting XEmacs:

```
remove Mod1 = Mode_switch
```

8.3 8.3. How can I stop the down-arrow-key from adding empty lines at the bottom of my buffers?

Add the following line to your '`.emacs`' file:

```
(setq next-line-add-newlines nil)
```

8.4 8.4. I wonder if there is an interactive function that can generate "fake" keyboard events. This way, I could simply map them inside XEmacs.

This seems to work:

```
(defun cg--generate-char-event (ch)
  "Generate an event, as if ch has been typed"
  (dispatch-event (character-to-event ch)))

;;  Backspace and Delete stuff
(global-set-key '(backspace)
                '(lambda () (interactive) (cg--generate-char-event 127)))
(global-set-key '(unknown_keysym_0x4)
                '(lambda () (interactive) (cg--generate-char-event 4)))
```

8.5 8.5. I am trying to bind C-. to scroll up by one line and C-; to scroll down by one line...

Add the following (Thanks to Richard Mlynarik <mly@adoc.xerox.com> and Wayne Newberry <wayne@zen.cac.stratus.com>) to '`.emacs`':

```
(defun scroll-up-one-line ()
  (interactive)
  (scroll-up 1))

(defun scroll-down-one-line ()
  (interactive)
  (scroll-down 1))

(global-set-key [(control ?.)]   'scroll-up-one-line)       ; C-.
(global-set-key [(control ?;)]   'scroll-down-one-line)     ; C-;
```

The key point is that you can only bind simple functions to keys; you can not bind a key to a function that you're also passing arguments to. (See <A HREF="#ss8.7">8.7 How can I bind complex functions</A> for a better answer.)

8.6 8.6. I cannot manage to globally bind my `DELETE` key to something other than the default. How does one do this?

```
(defun Foo ()
  (interactive)
    (message "You hit DELETE"))

(global-set-key "\C-?" 'Foo)
```

However, some modes explicitly bind `DELETE`, so you would need to add a hook that does `local-set-key` for them.

8.7 8.7. How can I bind complex functions (or macros) to keys?

As an example, say you want the PASTE key on a Sun keyboard to insert the current Primary X selection at point. You can accomplish this with:

```
(define-key global-map 'f18 'x-insert-selection)
```

However, this only works if there is a current X selection (the selection will be highlighted). The functionality I like is for the PASTE key to insert the current X selection if there is one, otherwise insert the contents of the clipboard. To do this you need to pass arguments to x-insert-selection. This is done by wrapping the call in a 'lambda form:

```
(define-key global-map 'f18
  (function (lambda () (interactive) (x-insert-selection t nil))))
```

This binds the 'f18 key to a "generic" functional object. The interactive spec is required because only interactive functions can be bound to keys. Also take a look at the doc for "function".

For the FAQ example you could use:

```
(global-set-key [(control ?.)]
```

```
(function (lambda () (interactive) (scroll-up 1))))
(global-set-key [(control ?;)]
        (function (lambda () (interactive) (scroll-up -1))))
```

This is fine if you only need a few functions within the lambda body. If you're doing more it's cleaner to define a separate function as in the original FAQ example (<A HREF="#ss11.3">question 11.3</A>).

8.8 8.8. Can the cursor keys scroll the screen a line at a time, rather than the default half page jump? I tend it to find it disorienting.

Try this:

```
(defun scroll-one-line-up (&optional arg)
  "Scroll the selected window up (forward in the text) one line (or N lines)."
  (interactive "p")
  (scroll-up (or arg 1)))

(defun scroll-one-line-down (&optional arg)
  "Scroll the selected window down (backward in the text) one line (or N)."
  (interactive "p")
  (scroll-down (or arg 1)))

(global-set-key 'up 'scroll-one-line-up)
(global-set-key 'down  'scroll-one-line-down)
```

The following will also work but will affect more than just the cursor keys (i.e. C-n and C-p):

```
(setq scroll-step 1)
```

8.9 8.9. How to map "Help" key alone on Sun type4 keyboard?

The following works in GNU Emacs 19:

```
(global-set-key [help] 'help-command)   ;; Help
```

The following works in XEmacs 19.13 with the addition of shift:

```
(global-set-key [(shift help)] 'help-command)   ;; Help
```

But it doesn't work alone. This is in the file '`PROBLEMS`' which should have come with your XEmacs installation:

Emacs ignores the `HELP` *key when running OLWM.*

OLWM grabs the `HELP` key, and retransmits it to the appropriate client using XSendEvent. Allowing Emacs to react to synthetic events is a security hole, so this is turned off by default. You can enable it by setting the variable x-allow-sendevents to t. You can also cause fix this by telling OLWM to not grab the help key, with the null binding '`OpenWindows.KeyboardCommand.Help:`'.

8.10 8.10. How can you type in special characters in XEmacs?

One way is to use the package '`x-compose`'. Then you can use sequences like `Compose " a` to get d (a-umlaut), etc.

8.11 8.11. Why does `(define-key global-map [ delete-forward ] 'delete-char)` complain of not being able to bind an unknown key?

Try this instead:

```
(define-key global-map [delete_forward] 'delete-char)
```

and it will work.

What you are seeing above is a bug due to code that is trying to check for FSF Emacs bogosity like

```
(define-key global-map [C-M-a] 'delete-char)
```

which otherwise would cause no errors but would not result in the expected behavior.

9 9. The cursor

9.1 9.1. Is there a way to make the bar cursor a little thicker than one-pixel thick since I lose it?

For a 1 pixel bar cursor, use:

```
(setq bar-cursor t)
```

For a 2 pixel bar cursor, use:

```
(setq bar-cursor 'anything-else)
```

You can use a color to make it stand out better:

```
Emacs*cursorColor:  Red
```

9.2 9.2. On the same subject – is there a way to get back the old "block" cursor where the cursor covers the character in front of the point?

```
(setq bar-cursor nil)
```

10 10. The mouse; cutting and pasting

10.1 10.1. I keep hitting the middle mouse button by accident and getting stuff pasted into my buffer so how can I turn this off?

Here is an alternative binding, whereby the middle mouse button selects (but does not cut) the expression under the mouse. Clicking middle on a left or right paren will select to the matching one. Note that you can use `define-key` or `global-set-key`.

```
(defun Mouse-Set-Point-and-Select (event)
  "Sets the point at the mouse location, then marks following form"
  (interactive "@e")
  (mouse-set-point event)
  (mark-sexp 1)
  )
(define-key global-map 'button2 'Mouse-Set-Point-and-Select)
```

(Editor's Note – there is a problem with texinfo/text/html conversion, so the double at-sign should only be a single, above. I'll fix it one of these days – AJR)

10.2 10.2. How do I set control/meta/etc modifiers on mouse buttons?

Use, for instance, `[(meta button1)]`. For example, here is a common setting for Common Lisp programmers who use the bundled ilisp package, whereby meta-button1 on a function name will find the file where the function name was defined, and put you at that location in the source file.

[Inside a function that gets called by the lisp-mode-hook and ilisp-mode-hook]

```
(local-set-key [(meta button1)] 'edit-definitions-lisp)
```

10.3 10.3. I do "^x ^b" to get a list of buffers and the entries get highlighted when I move the mouse over them but clicking the left mouse does not do anything.

Use the middle mouse button.

10.4 10.4. How can I get a list of buffers to popup when I hit button 3 on the mouse?

The following code will actually replace the default popup on button3:

```
(defun cw-build-buffers ()
  "Popup buffer menu."
  (interactive "@")
  (run-hooks 'activate-menubar-hook)
  (popup-menu (car (find-menu-item current-menubar '("Buffers")))))

(define-key global-map [(button3)] 'cw-build-buffers)
```

(Editor's Note – there is a problem with texinfo/text/html conversion, so the double at-sign should only be a single, above. I'll fix it one of these days – AJR)

10.5 10.5. Why does cut-and-paste not work between XEmacs and a cmdtool?

We don't know. It's a bug. There does seem to be a work-around, however. Try running xclipboard first. It appears to fix the problem even if you exit it. (This should be mostly fixed in 19.13, but we haven't yet verified that).

10.6 10.6. How I can set XEmacs up so that it pastes where the cursor is _not_ where the pointer lies?

Try adding the following to your '`.emacs`':

```
(define-key global-map 'button2 'x-insert-selection)
```

This comes from the '`sample.emacs`' file in '`etc/`', which has lots of goodies.

10.7 10.7. How do I select a rectangular region?

Just select the region normally, then use the rectangle commands (e.g. `kill-rectangle`) on it. The region does not highlight as a rectangle, but the commands work just fine.

To actually sweep out rectangular regions with the mouse do the following:

```
(setq mouse-track-rectangle-p t)
```

10.8 10.8. Why does M-w take so long?

It actually doesn't. It leaves the region visible for a second so that you can see what area is being yanked. If you start working, though, it will immediately complete its operation. In other words, it will only delay for a second if you let it.

11 11. Highlighting

11.1 11.1. How can I highlight selections?

Use zmacs mode. This mode allows for what some might call gratuitous highlighting for selected regions (either by setting mark or by using the mouse). To use, add the following line to your '`.emacs`' file:

```
(setq zmacs-regions t)
```

This is the default behavior.

11.2 11.2. How do I get a pending-delete type of behavior?

Pending delete is what happens when you select a region (with the mouse or keyboard) and you press a key to replace the selected region by the key you typed. Usually backspace kills the selected region.

To get this behavior, add the following line to your '`.emacs`' file:

```
(require 'pending-del)
```

11.3 11.3. I do not like my text highlighted while I am doing isearch as I am not able to see whats underneath. How do I turn it off?

```
(setq isearch-highlight nil)
```

Note also that isearch-highlight affects query-replace and ispell. Instead of disabling isearch-highlight you may find that a better solution consists of customizing the 'isearch' face.

11.4 11.4. The text gets highlighted when I do C-x C-p (mark-page). Is there a way to turn this feature off?

```
(setq zmacs-regions nil)
```

12 12. The menubar and toolbar

12.1 12.1. How do I get rid of the menu (or menubar) ?

Answer: To get rid of the menubar, add to '`.emacs`':

```
(set-menubar nil)
```

12.2 12.2. Can I customize the basic menubar?

For an extensive menubar, add the line

```
(load "big-menubar")
```

to your '`.emacs`' file. If you'd like to write your own, this file provides a good set of examples to start from:

```
lisp/packages/big-menubar.el
```

(starting from your system XEmacs installation directory).

12.3 12.3. What controls how many buffers are listed in the menu "Buffers" list?

Add the following to your '`.emacs`', modified as needed:

```
(setq buffers-menu-max-size 20)
```

If you do not want a limit, try

```
(setq buffers-menu-max-size nil)
```

12.4 12.4. I am trying to use a resource like `Emacs*menubar*font` to set the font of the menubar but it's not working.

If you are using the real Motif menubar, this resource is not recognized; you have to say

```
Emacs*menubar*fontList: FONT
```

If you are using the Lucid menubar, the former resource will be recognized only if the latter resource is unset. This means that the resource

```
*fontList: FONT
```

will override

```
Emacs*menubar*font: FONT
```

even though the latter is more specific.

12.5 12.5. How can I bind a key to a function which will turn the toolbar on and off?

Bind keys to do this, in order to turn it off:

```
(set-specifier top-toolbar-height (cons (selected-frame) 0))
```

and to turn it back on:

```
(remove-specifier top-toolbar-height (selected-frame))
```

13 13. Scrollbars

13.1 13.1. How can I disable the scrollbar?

To turn disable them for all frames, add the following line to your '`.Xdefaults`':

```
Emacs.scrollBarWidth:  0
```

To turn the scrollbar off on a per-frame basis, use the following function:

```
(set-specifier scrollbar-width (cons (selected-frame) 0))
```

You can actually turn the scrollbars on at any level you want by substituting for (selected-frame) in the above command. For example, to turn the scrollbars off only in a single buffer:

```
(set-specifier scrollbar-width (cons (current-buffer) 0))
```

13.2 13.2. How can one use resources to change scrollbar colors?

Here's a recap of how to use resources to change your scrollbar colors:

```
! Motif scrollbars

Emacs*XmScrollBar.Background: skyblue
Emacs*XmScrollBar.troughColor: lightgray

! Athena scrollbars

Emacs*Scrollbar.Foreground: skyblue
Emacs*Scrollbar.Background: lightgray
```

Note the capitalization of '`Scrollbar`' for the Athena widget.

13.3 13.3. When I move the scrollbar in an XEmacs window, it moves the point as well, which should not be the default behavior. Is this a bug or a feature? Can I disable it?

The current behavior is a feature, not a bug. Point remains at the same buffer position as long as that position does not scroll off the screen. In that event, point will end up in either the upper-left or lower-left hand corner.

13.4 13.4. How can I get automatic horizontal scrolling? By the same token, how can I turn it off in specific modes?

To do this, add to your .emacs file:

```
(require 'auto-show)
```

Then do `(setq truncate-lines t)` in the mode-hooks for any modes in which you want lines truncated.

14 14. Frame Geometry

14.1 14.1. In Lucid Emacs 19.6 I did `(set-screen-width` *characters*`)` and `(set-screen-height` *lines*`)` in my '`.emacs`' instead of specifying '`Emacs*EmacsScreen.geometry`' in my '`.Xdefaults`' but this does not work in XE

macs 19.13.

These two functions now take frame arguments:

```
(set-frame-width (selected-frame) characters)
(set-frame-height (selected-frame) lines)
```

14.2 14.2. In XEmacs 19.11 I specified '`Emacs*EmacsScreen.geometry`' in my '`.emacs`' but this does not work in XEmacs 19.13.

We have switched from using the term 'screen' to using the term 'frame'. The correct entry for your '`.Xdefaults`' is now:

```
Emacs*EmacsFrame.geometry
```

15 15. Window/icon title; window manager problems

15.1 15.1. How can I get the icon to just say '`XEmacs`' and not include the name of the current file in it?

Add the following line to your '`.emacs`' file:

```
(setq frame-icon-title-format "XEmacs")
```

15.2 15.2. How can I have the window title area display the full directory/name of the current buffer file and not just the name?

Add the following line to your '`.emacs`' file:

```
(setq frame-title-format "%S: %f")
```

A more sophisticated title might be:

```
(setq frame-title-format
  '("%S: " (buffer-file-name "%f" (dired-directory dired-directory "%b"))))
```

That is, use the file name, or the dired-directory, or the buffer name.

15.3 15.3. When I run '`xterm -name junk`' I get an xterm whose class name according to xprop, is '`junk`'. This is the way it's supposed to work, I think. When I run '`xemacs -name junk`' the class name is not set to '`junk`'. It's sti

ll '`emacs`'. What does '`xemacs -name`' really do? The reason I ask is that my window manager (fvwm) will make a window sticky and I use XEmacs to read my mail. I want that XEmacs window to be sticky, without having to use the window manager's f unction to set the window sticky. What gives?

'`xemacs -name`' sets the application-name for the program (that is, the thing which normally comes from `argv[0]`.) Using '`-name`' is the same as making a copy of the executable with that new name. The WM_CLASS property on each frame is set to the frame-name, and the application-class. So, if you did `xemacs -name FOO` and then created a frame named `BAR`, you'd get an X window with WM_CLASS = `( "BAR", "Emacs")`. However, the resource hierarchy for this widget would be

```
Name:    FOO  .shell.       .pane        .BAR
Class:   Emacs.TopLevelShell.XmMainWindow.EmacsFrame
```

instead of the default

```
Name:    xemacs.shell.       .pane        .emacs
Class:   Emacs .TopLevelShell.XmMainWindow.EmacsFrame
```

It is arguable that the first element of WM_CLASS should be set to the application-name instead of the frame-name, but I think that's less flexible, since it does not give you the ability to have multiple frames with different WM_CLASS properties. Another possibility would be for the default frame name to come from the application name instead of simply being `emacs`. However, at this point, making that change would be troublesome: it would mean that many users would have to make yet another change to their resource files (since the default frame name would suddenly change from `emacs` to `xemacs`, or whatever the executable happened to be named), so we'd rather avoid it.

To make a frame with a particular name use:

```
(make-frame '((name . "the-name")))
```

16 16. Editing source code (C mode; Lisp mode; etc.)

16.1 16.1. How can I do source code highlighting using font-lock?

For most modes, font-lock is already set up and just needs to be turned on. This can be done by

```
M-x font-lock-mode
```

or by having XEmacs automatically start it by adding lines like

```
(add-hook 'emacs-lisp-mode-hook 'turn-on-font-lock)
(add-hook 'dired-mode-hook 'turn-on-font-lock)
```

to your '`.emacs`'. See the file `etc/sample.emacs` for more examples.

16.2 16.2. How do you arrange it so that XEmacs indents all the clauses of a Common Lisp `if` the same amount instead of indenting the 3rd clause differently from the first two?

One way is to add, to '`.emacs`':

```
(put 'if 'lisp-indent-function nil)
```

However, note that the package '`cl-indent.el`' that comes with XEmacs sets up this kind of indentation by default. '`cl-indent`' also knows about many other CL-specific forms. To use '`cl-indent`', one can do this:

```
(load "cl-indent")
(setq lisp-indent-function (function common-lisp-indent-function))
```

One can also customize '`cl-indent.el`' so it mimics the default '`if`' indentation ('`then`' indented more than the '`else`'). Here's how:

```
(put 'if 'common-lisp-indent-function '(nil nil &body))
```

Also, a new version (1.2) of '`cl-indent.el`' was posted to '`comp.emacs.xemacs`' on 12/9/94. This version includes more documentation than previous versions. This may prove useful if you need to customize any indent-functions. The post can be retrieved by searching the XEmacs mail archives.

16.3 16.3. I do not like cc-mode. How do I use the old c-mode?

Well, first off, consider if you really want to do this. cc-mode is much more powerful than the old c-mode. But if you still insist, add the following lines to your '`.emacs`':

```
(fmakunbound 'c-mode)
(makunbound 'c-mode-map)
```

```
(fmakunbound 'c++-mode)
(makunbound 'c++-mode-map)
(makunbound 'c-style-alist)
(load-library "old-c-mode")
(load-library "old-c++-mode")
```

This must be done before any other reference is made to either c-mode or c++-mode.

16.4 16.4. When I try to edit a postscript file it gets stuck saying: fontifying 'filename' (regexps....) and it just sits there. If I press ctrl-c in the window where XEmacs was started, it suddenly becomes alive again.

This was caused by a bug in the Postscript font-lock regular expressions. It should be fixed in 19.13. For earlier versions of XEmacs, have a look at your '`.emacs`' file. You will probably have a line like:

```
(add-hook 'postscript-mode-hook 'turn-on-font-lock)
```

Take it out, restart XEmacs, and it won't try to fontify your postscript files anymore.

16.5 16.5. Does anyone know how to get the "More" Syntax Highlighting on by default?

For C, C++, and Lisp, you can try adding the following to your '`.emacs`' file:

```
(setq c-font-lock-keywords c-font-lock-keywords-2)
(setq c++-font-lock-keywords c++-font-lock-keywords-2)
(setq lisp-font-lock-keywords lisp-font-lock-keywords-2)
```

17 17. Text mode

17.1 17.1. How can I enable auto-indent?

Put the following line in your '`.emacs`'.

```
(setq indent-line-function 'indent-relative-maybe)
```

If you want to get fancy, try '`filladapt`' and '`fa-extras`', available from the Emacs Lisp Archive at Ohio State University. Get them at URL:

```
ftp://archive.cis.ohio-state.edu/pub/gnu/emacs/elisp-archive/packages
```

17.2 17.2. How can I get XEmacs to come up in text mode (auto-fill) by default?

Try the following lisp in your '`.emacs`' file

```
(setq default-major-mode 'text-mode)
(setq text-mode-hook 'turn-on-auto-fill)
```

WARNING: note that changing default-major-mode from fundamental-mode can break a large amount of built-in code that expects newly created buffers to be in fundamental-mode. (Changing from fundamental-mode to auto-fill text-mode might not wreak too much havoc, but changing to something more exotic like a lisp-mode would break many Emacs packages.

Note that Emacs defaultly starts up in buffer *scratch* in initial-major-mode, which defaults to lisp-interaction-mode. Thus adding the following form to your Emacs init file will cause the initial *scratch* buffer to be put into auto-fill'ed text-mode.

```
(setq initial-major-mode
      (function (lambda ()
        (text-mode)
        (turn-on-auto-fill))))
```

Note that after your init file is loaded, if inhibit-startup-message is null (the default) and the startup buffer is *scratch* then the startup message will be inserted into *scratch*; it will be removed after a timeout by erasing the entire *scratch* buffer. Keep in mind this default usage of *scratch* if you desire any prior manipulation of *scratch* from within your Emacs init file. In particular, anything you insert into *scratch* from your init file will be later erased. Also, if you change the mode of *scratch* be sure that this will not interfere with possible later insertion of the startup message (e.g. if you put *scratch* into a nonstandard mode that has automatic font lock rules, then the startup message might get fontified in a strange foreign manner, e.g. as code in some programming language).

18 18. Shell mode

18.1 18.1. How do I start up a second shell buffer?

In the *shell* buffer:

```
M-x rename-buffer RET *shell-1* RET
M-x shell RET
```

This will then start a second shell. The key is that no buffer named *shell* can exist. It might be preferable to use `M-x rename-uniquely` to rename the *shell* buffer instead of `M-x rename-buffer`.

18.2 18.2. I'm using the Emacs `M-x shell` function, and I would like to invoke and use a telnet session within it. Everything works fine except that now all ^M's are filtered out by Emacs. Fixes?

Use `M-x rsh` or `M-x telnet` to open remote sessions rather than doing rsh or telnet within the local shell buffer.

19 19. Mail; VM; GNUS; BBDB; and related

19.1 19.1. How and where I am suppose to set the face attributes for customizing the appearance of messages (i.e. for VM)?

Suppose you want to use:

```
(set-face-font 'message-highlighted-header-contents
               "-adobe-courier-bold-r-normal--12*")
(set-face-foreground 'message-headers "darkslateblue")
(set-face-foreground 'message-header-contents "brown")
(set-face-foreground 'message-highlighted-header-contents "black")
(set-face-foreground 'message-cited-text "darkgreen")
```

Well, this should work, provided that `(require 'highlight-headers)` is executed first. Also, highlight-headers is self-contained and you don't need to turn on font-lock mode.

19.2 19.2. I seem to have heard that there's a package distributed with XEmacs which will use the echo area to notify you of incoming mail in a rather configurable way (multiple spool files, different actions for different files). What and where is

it?

It's called reportmail. Add the following to your '`.emacs`':

```
(load-library "reportmail")
```

19.3 19.3. Is there any way to add more faces and regexps to GNUS without hacking gnus.el?

You can try the folowing elisp:

```
(add-hook 'gnus-startup-hook
          '(lambda ()
             (font-lock-mode)
             (set-face-foreground 'message-headers "red")
             (set-face-foreground 'message-header-contents "orange")
             (set-face-foreground 'message-cited-text "blue")))
```

19.4 19.4. What is BBDB?

BBDB is the Big Brother Database, written by Jamie Zawinski <jwz@netscape.com>. It interfaces to VM, mh-e, and GNUS and conveniently snarfs information about people and other things from articles and messages that go by.

BBDB is available from the elisp archive

```
archive.cis.ohio-state.edu:/pub/gnu/emacs/elisp-archive
```

BBDB has a mailing list devoted to it:

```
info-bbdb@cs.uiuc.edu
```

This mailing list is archived at

```
ftp.cs.uiuc.edu:/pub/bbdb
```

19.5 19.5. I noticed that BBDB evokes an XEmacs bug; is there a fix?

This worked for me:

```
(setq bbdb-electric-p nil)
```

19.6 19.6. I'm getting the error 'Wrong type argument: listp :-pos' when I try to start BBDB. What do I do?

Discard the copy of mail-extr.el that came with bbdb.

XEmacs 19.12 made the change that you cannot change the value of interned symbols whose names begin with a colon. The mail-extr.el that's distributed with 19.12+ is fixed to live with this restriction, but old copies of mail-extr.el will bomb.

19.7 19.7. I'm getting the error 'movemail: Permission denied' when I try and start VM. What do I do?

Execute the following two commands. movemail is normally located in the XEmacs install tree at '`.../lib/xemacs-19.13/${arch}/movemail`'.

```
          chgrp mail movemail
          chmod 2555 movemail
```

19.8 19.8. How can I get URLs sent to Netscape for VM and GNUS?

First, consider:

```
(setq highlight-headers-follow-url-function
   'highlight-headers-follow-url-netscape)
```

For VM, you have three choices. For the first, from within VM, click button3 and select netscape from menu or say :

```
(setq vm-url-browser "netscape")
```

and click button2 on the URL.

For the second, a better choice is:

```
(setq vm-url-browser 'vm-mouse-send-url-to-netscape)
```

If you set vm-url-browser to "netscape", it will launch a new version of netscape every time, but the two functions will direct an existing version of netscape to the new url, unless you aren't running netscape, in which case it will launch one.

For GNUS (at least for 5.0, though might work on 4.x.x):

```
(setq gnus-button-url 'gnus-netscape-open-url)
```

19.9 19.9. What can be done about this "integer constant overflow" error GNUS gives when reading newsgroups?

It is usually caused by too many headers in a news group. Look at the buffer *nntp* and do a search in that buffer for the integer that caused the overflow. When you find a match, put:

```
(setq gnus-ignored-newsgroups "newsgroup.you.found")
```

in your .emacs file.

20 20. Printing

20.1 20.1. Font-lock looks nice. How can I print (WYSIWYG) the highlighted document?

The package ps-print.el, which is now included with XEmacs, provides the ability to do this. It contains complete instructions on its use:

```
<xemacs_src_root>/lisp/packages/ps-print.el
```

20.2 20.2. My printer is a Postscript printer and `lpr` only works for Postscript files, so how do I get `M-x lpr-region` and `M-x lpr-buffer` to work?

```
(setq lpr-command "Command to print text files")
```

Please don't forget to replace *Command to print text files* with the actual command ('`enscript`', '`nenscript`', etc.)!

21 21. Gnuserv

21.1 21.1. How do I disable gnuserv from opening a new frame?

If you set the gnuserv-screen variable to the frame that should be used to display buffers that are pulled up, a new frame will not be created. For example, you could put

```
(setq gnuserv-screen (selected-frame))
```

early on in your '`.emacs`', to ensure that the first frame created is the one used for your gnuserv buffers.

21.2 21.2. What do I do to start the gnuserv server so that each subseqeuent XEmacs is a client?

Put the following in your '`.emacs`' file to start the server:

```
(gnuserv-start)
```

Start your first XEmacs as usual. After that, you can do

```
gnuclient randomfilename
```

from the command line to get your existing XEmacs process to open a new frame and visit randomfilename in that window. When you're done editing randomfilename, hit `C-x #` to kill the buffer and get rid of the frame.

22 22. Miscellaneous

22.1 22.1. How do I specify the paths that XEmacs uses for finding files?

You can specify what paths to use by using a number of different flags when running configure. See the section MAKE VARIABLES in the top-level file INSTALL in the XEmacs distribution for a listing of those flags.

Most of the time, however, the simplest fix is: DO NOT specify paths as you might for FSF GNU Emacs. XEmacs can generally determine the necessary paths dynamically at run time. The only path that generally needs to be specified is the root directory to install into. That can be specified by passing the `--prefix` flag to configure. For a description of the XEmacs install tree, please consult the NEWS file.

22.2 22.2. Why does edt emulation not work?

We don't know, but you can use tpu-edt emulation instead, which works fine and is a little fancier than the standard edt emulation. To do this, add the following line to your '`.emacs`':

```
(load "tpu-edt")
```

If you don't want it to replace Ctrl-h with edt-style help menu add this as well:

```
(global-set-key '(control h) 'help-for-help)
```

22.3 22.3. How can I emulate VI and use it as my default mode?

Our recommended VI emulator is viper. To put the current buffer into viper-mode, use the command:

```
M-x viper
```

To make viper-mode the default, add the following lines to your '`.emacs`':

```
(load-library "viper")
(setq term-setup-hook 'viper)
(setq find-file-hooks 'viper)
(setq find-file-not-found-hooks 'viper)
```

22.4 22.4. Is there some way to get the behavior so that if the current buffer has a file associated with it, the current buffer will use that files name else use the buffer name?

Just set frame-title-format from find-file-hooks. Alternatively, look at the answer to question 15.2.

In addition, one could set modeline-format.

22.5 22.5. I have no idea where this is coming from, but ever since I moved from 19.9 to 19.13 I have started seeing that all of my buffers will get a minor mode called '`Omit`'. I have no idea how it got there nor do I know what it does. What is

it?

It's part of dired. In dired, you can type M-o to get Omit mode and that will ignore uninteresting files (checkpoint files and backups, for example). You get Omit in the modeline everywhere because the variable 'dired-omit-files-p' is globally set to some non-nil value. If you want this functionality, it's probably best to use a hook:

```
(add-hook 'dired-after-readin-hook '(lambda () (dired-omit-toggle)))
```

Alternatively, since it seems odd to toggle the omit state with every readin, since readin can happen many times in a Dired buffer, you can try this hook to correct the "Omit" problem:

```
(add-hook 'dired-mode-hook
  (function (lambda ()
      ;; `dired-omit-files-p' is made buffer-local by "dired-x.el", but
      ;; maybe not soon enough.
            (make-local-variable 'dired-omit-files-p)
            (setq dired-omit-files-p t))))
```

This is only run once, when the Dired buffer is created.

22.6 22.6. How do I turn off the sound?

Add the following line to your '`.emacs`' file:

```
(setq bell-volume 0)
(setq sound-alist nil)
```

22.7 22.7. Can I have the end of the buffer delimited in some way? Say, with: [END] ?

```
(make-annotation "[END]" (point-max) 'text (current-buffer))
```

Note that you might want to put this in a hook.

You might also need:

```
(require 'annotations)
```

since `make-annotation` is not defined by default.

22.8 22.8. Can I insert today's date into buffer?

Use this lisp in a function:

```
(insert (current-time-string))
```

22.9 22.9. Are only certain syntactic character classes available for abbrevs? I didn't see any restrictions in the info.

Yes, abbrevs only expand word-syntax strings. So, in c-mode if you wanted to expand something to '`define `', you would be able to expand '`xd`' but not '`#d`'.

22.10 22.10. Filladapt used to work after I loaded it. Now in 19.13 it doesn't. What gives?

Filladapt 2.x is included in 19.13+. In it filladapt is now a minor mode and minor modes are traditionally off by default. The following added to your .emacs will turn it on for all buffers:

```
(setq-default filladapt-mode t)
```

Use `turn-on-filladapt-mode` to turn Filladapt on in particular major modes, like this:

```
(add-hook 'text-mode-hook 'turn-on-filladapt-mode)
```

Table of Contents