XEmacs Reference Card
(for version 21.0+)

Starting Emacs
To enter XEmacs, just type its name: xemacs
To read in a file to edit, see Files, below.

Leaving Emacs
suspend Emacs (or iconify frame under X) C-z
exit Emacs permanently C-x C-c

Files
read a file into Emacs C-x C-f
save a file back to disk C-x C-s
save all files C-x s
insert contents of another file into this buffer C-x i
replace this file with the file you really want C-x C-v
write buffer to a specified file C-x C-w

Getting Help
The Help system is simple. Type C-h and follow the directions.
If you are a first-time user, type C-h t for a tutorial;
quit Help window q
scroll Help window space
apropos: show commands matching a string C-h a
show the function a key runs C-h c
describe a function C-h f
get modesopecific information C-h m

Error Recovery
abort partially typed or executing command C-g
recover a file lost by a system crash M-x Recover-file
recover files from a previous Emacs session M-x Recovery-session
undo an unwanted change C-x u or C-_r
restore a buffer to its original contents M-x revert-buffer
redraw garbed screen C-l

Incremental Search
search forward C-s
search backward C-r
regular expression search C-x w
reverse regular expression search M-x w
select previous search string M-p
select next later search string M-n
exit incremental search RET
undo effect of last character DEL
abort current search C-g
Use C-s or C-r again to repeat the search in either direction,
If Emacs is still searching, C-g cancels only the part not done.

Marking
set mark here C-s or C-spc
exchange point and mark C-x C-x
set mark at words away M-q
mark paragraph M-h
mark page C-x C-p
mark sexp C-x M-s
mark function C-x M-f
mark entire buffer C-x h

Query Replace
interactively replace a text string M-x
using regular expressions M-x query-replace-regexp
Valid responses in query-replace mode are
replace this one, go on to next SPC or y
replace this one, don't move ,
skip to next without replacing DEL or n
replace all remaining matches !
back up to the previous match -
exit query-replace SSC
enter recursive edit (C-x 0 to exit) C-x r
delete match and enter recursive edit C-x w

Multiple Windows
delete all other windows C-x 1
delete this window C-x 0
split window in two vertically C-x 2
split window in two horizontally C-x 3
scroll other window C-x v
switch cursor to another window C-x o
shrink window shorter M-x shrink-window
grow window taller C-x t
shrink window narrower C-x { 
grow window wider C-x }
select buffer in other window C-x 4 b
display buffer in other window C-x 4 C-o
find file in other window C-x 4 f
find file read-only in other window C-x 4 r
run Dired in other window C-x 4 d
find tag in other window C-x 4 .

Motion
entity to move over backward forward
character C-b C-f
word M-b M-f
line C-p C-n
sentence M-a M-e
paragraph M-{ M-
page C-x [ C-x ]
sexp C-x M-b C-x-M-f
function C-x-M-a C-x-M-s
go to buffer beginning (or end) M-< M->
scroll to next screen C-v
scroll to previous screen M-v
scroll left C-x <
scroll right C-x >
scroll current line to center of screen C-u C-i

Killing and Deleting
entity to kill backward forward
character (delete, not kill) DEL C-d
word M-DEL M-d
line (to end of) M-0 C-k
sentence C-x DEL M-k
sexp M-< C-M-k C-M-k
kill region C-w
copy region to kill ring M-w
kill through next occurrence of char M-z char
yank back last thing killed C-y
replace last yank with previous kill M-y

© 1998 Free Software Foundation, Inc. Permissions on back, v2.0 XEmacs
### XEmacs Reference Card

#### Buffers
- select another buffer: \texttt{C-x b}
- list all buffers: \texttt{C-x C-b}
- kill a buffer: \texttt{C-x k}

#### Transposing
- transpose characters: \texttt{C-t}
- transpose words: \texttt{M-t}
- transpose lines: \texttt{C-x C-t}
- transpose sexps: \texttt{C-M-t}

#### Spelling Check
- check spelling of current word: \texttt{M-$}
- check spelling of all words in region: \texttt{M-x impell-region}
- check spelling of entire buffer: \texttt{M-x impell-buffer}

#### Tags
- find a tag (a definition): \texttt{M-.}
- find next occurrence of tag: \texttt{C-u M-.}
- specify a new tags file: \texttt{M-x visit-tags-table}
- regexp search on all files in tags table: \texttt{M-x tags-table}
- run query-replace on all the files: \texttt{M-x tags-query-replace}
- continue last tags search or query-replace: \texttt{M-}

#### Shells
- execute a shell command: \texttt{M-!}
- run a shell command on the region: \texttt{M-!}
- filter region through a shell command: \texttt{C-u M-!}
- start a shell in window \texttt{*shell*}: \texttt{M-x shell}

#### Rectangles
- copy rectangle to register: \texttt{C-x r r}
- kill rectangle: \texttt{C-x r k}
- yank rectangle: \texttt{C-x r y}
- open rectangle, shifting text right: \texttt{C-x r o}
- blank out rectangle: \texttt{M-x clear-rectangle}
- prefix each line with a string: \texttt{M-x string-rectangle}
- select rectangle with mouse: \texttt{M-x button}

#### Abbrevs
- add global abbrev: \texttt{C-x a g}
- add mode-local abbrev: \texttt{C-x a i}
- add global expansion for this abbrev: \texttt{C-x a i g}
- add mode-local expansion for this abbrev: \texttt{C-x a i l}
- explicitly expand abbrev: \texttt{C-x a e}
- expand previous word dynamically: \texttt{M-}
Keyboard Macros

- start defining a keyboard macro: C-x
- end keyboard macro definition: C-x
- execute last-defined keyboard macro: C-x e
- edit keyboard macro: C-x C-k
- append to last keyboard macro: C-u C-x
- name last keyboard macro: M-x name-last-kbd-macro
- insert Lisp definition in buffer: M-x insert-kbd-macro

Commands Dealing with Emacs Lisp

- eval sexp before point: C-x C-e
- eval current defun: C-M-x
- eval region: M-x eval-region
- eval entire buffer: M-x eval-current-buffer
- read and eval minibuffer: M-ESC
- re-execute last minibuffer command: C-x ESC ESC
- read and eval Emacs Lisp file: M-x load-file
- load from standard system directory: M-x load-library

Simple Customization

Here are some examples of binding global keys in Emacs Lisp:

(global-set-key [control c] [super t 'goto-line])
(global-set-key [control z] (control k) [control k] 'kill-region)
(global-set-key [meta #] [super t 'query-replace-regexp)

An example of setting a variable in Emacs Lisp:

(setq backup-by-copying-when-linked t)

Writing Commands

(defun command-name (args)
  "documentation"
  (interactive "template")
  body)

An example:

(defun this-line-to-top-of-window (line)
  "Reposition line point is on top of window."
  With ARG, put point on line ARG. Negative counts from bottom."
  (interactive "p")
  (recenter (if (null line)
                0
                (prefix-numeric-value line))))

The argument to interactive is a string specifying how to get the arguments when the function is called interactively. Type C-h i interactive for more information.

Copyright © 1998 Free Software Foundation, Inc.
designed by Stephen Gadea, April 1998 v2.0 Xemacs
By GNU Emacs version 19 on Unix systems
Updated for XEmacs in February 1995 by Ben Wang

Permission is granted to make and distribute copies of this card provided the copyright notice and this permission notice are preserved on all copies.
For copies of the GNU Emacs manual, write to the Free Software Foundation, Inc., 50 Temple Place, Suite 330, Boston, MA 02111-1307, USA.