GNU Emacs Reference Card
(for version 20)

Starting Emacs

To enter GNU Emacs 20, just type its name: emacs
To read in a file to edit, see Files, below.

Leaving Emacs

suspend Emacs (or iconify it 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
version control checkin/checkout           C-x C-q

Getting Help

The help system is simple. Type C-h (or F1) and follow the directions. If you are a first-time user, type C-h t for a tutorial.

remove help window                        C-x 1
scroll help window                        C-x C-v
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 mode-specific 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
undo an unwanted change                   C-x u or C-_n
restore a buffer to its original contents M-x revert-buffer
redraw garaged screen                     C-l

Incremental Search

search forward                             C-s
search backward                            C-x
regular expression search                  C-w-s
reverse regular expression search          C-w-x
select previous search string              M-p
select next search string                   M-n
exit incremental search                    RET
undo effect of last character              DEL
abort current search                       C-g

Use C-s or C-x again to repeat the search in either direction. If Emacs is still searching, C-g cancels only the part not done.

© 1987 Free Software Foundation, Inc. Permission on back, v4.9
### Motion

<table>
<thead>
<tr>
<th>entity to move over</th>
<th>backward</th>
<th>forward</th>
</tr>
</thead>
<tbody>
<tr>
<td>character</td>
<td>C-b</td>
<td>C-f</td>
</tr>
<tr>
<td>word</td>
<td>M-b</td>
<td>M-f</td>
</tr>
<tr>
<td>line</td>
<td>C-p</td>
<td>C-n</td>
</tr>
<tr>
<td>go to line beginning (or end)</td>
<td>C-a</td>
<td>C-e</td>
</tr>
<tr>
<td>sentence</td>
<td>N-a</td>
<td>N-a</td>
</tr>
<tr>
<td>paragraph</td>
<td>M-{</td>
<td>M-{</td>
</tr>
<tr>
<td>page</td>
<td>C-x [</td>
<td>C-x ]</td>
</tr>
<tr>
<td>sexp</td>
<td>C-M-b</td>
<td>C-M-f</td>
</tr>
<tr>
<td>function</td>
<td>C-M-a</td>
<td>C-M-a</td>
</tr>
<tr>
<td>go to buffer beginning (or end)</td>
<td>M-&lt;</td>
<td>M-&gt;</td>
</tr>
<tr>
<td>scroll to next screen</td>
<td>C-v</td>
<td></td>
</tr>
<tr>
<td>scroll to previous screen</td>
<td>M-v</td>
<td></td>
</tr>
<tr>
<td>scroll left</td>
<td>C-x &lt;</td>
<td></td>
</tr>
<tr>
<td>scroll right</td>
<td>C-x &gt;</td>
<td></td>
</tr>
<tr>
<td>scroll current line to center of screen</td>
<td>C-u C-l</td>
<td></td>
</tr>
</tbody>
</table>

### Killing and Deleting

<table>
<thead>
<tr>
<th>entity to kill</th>
<th>backward</th>
<th>forward</th>
</tr>
</thead>
<tbody>
<tr>
<td>character (delete, not kill)</td>
<td>DEL</td>
<td>C-d</td>
</tr>
<tr>
<td>word</td>
<td>M-DEL</td>
<td>M-d</td>
</tr>
<tr>
<td>line (to end of)</td>
<td>N-O C-k</td>
<td>C-k</td>
</tr>
<tr>
<td>sentence</td>
<td>C-x DEL</td>
<td>M-k</td>
</tr>
<tr>
<td>sexp</td>
<td>M- C-M-k</td>
<td>C-M-k</td>
</tr>
<tr>
<td>kill region</td>
<td>C-w</td>
<td></td>
</tr>
<tr>
<td>copy region to kill ring</td>
<td>M-w</td>
<td></td>
</tr>
<tr>
<td>kill through next occurrence of char</td>
<td>M-z char</td>
<td></td>
</tr>
<tr>
<td>yank back last thing killed</td>
<td>C-y</td>
<td></td>
</tr>
<tr>
<td>replace last yank with previous kill</td>
<td>M-y</td>
<td></td>
</tr>
</tbody>
</table>

### Marking

<table>
<thead>
<tr>
<th>action</th>
<th>key</th>
</tr>
</thead>
<tbody>
<tr>
<td>set mark here</td>
<td>C-@ or C-SPC</td>
</tr>
<tr>
<td>exchange point and mark</td>
<td>C-x C-x</td>
</tr>
<tr>
<td>set mark arg words away</td>
<td>N-s</td>
</tr>
<tr>
<td>mark paragraph</td>
<td>M-b</td>
</tr>
<tr>
<td>mark page</td>
<td>C-x C-p</td>
</tr>
<tr>
<td>mark sexp</td>
<td>C-M-b</td>
</tr>
<tr>
<td>mark function</td>
<td>C-N-b</td>
</tr>
<tr>
<td>mark entire buffer</td>
<td>C-z b</td>
</tr>
</tbody>
</table>

### Query Replace

- interactively replace a text string: M-%
- using regular expressions: M-z query-replace-regexp

Valid responses in query-replace mode are:

- replace this one, go on to next: SPC
- replace this one, don't move: ,
- skip to next without replacing: DEL
- replace all remaining matches: !
- back up to the previous match: -
- exit query-replace: RET
- enter recursive edit (C-M-c to exit): C-r
Multiple Windows

When two commands are shown, the second is for "other frame."
delete all other windows C-x 1
split window, above and below C-x 2 C-x 5 2
delete this window C-x 0 C-x 5 0
split window, side by side C-x 3
scroll other window C-M-v
switch cursor to another window C-x 0 C-x 5 0
select buffer in other window C-x 4 b C-x 5 b
display buffer in other window C-x 4 C-o C-x 5 C-o
find file in other window C-x 4 f C-x 5 f
find file read-only in other window C-x 4 r C-x 5 r
rune Dired in other window C-x 4 d C-x 5 d
find tag in other window C-x 4 t C-x 5 t

grow window taller C-x ~
shrink window narrower C-x \{ grow window wider C-x 

Formatting

indent: current line (mode-dependent) TAB
indent region (mode-dependent) C-M-\ indent sexp (mode-dependent) C-M-q indent region rigidly avg columns C-x TAB insert newline after point C-o move rest of line vertically down C-M-o delete blank lines around point C-x C-o join line with previous (with avg, next) M-\ delete all white space around point M-\ put exactly one space at point M-SPC fill paragraph M-q set fill column C-x f set prefix each line starts with C-x . set face M-g

Case Change

uppercase word M-u lowercase word M-l capitalize word M-c uppercase region C-x C-u lowercase region C-x C-l

The Minibuffer

The following keys are defined in the minibuffer.

complete as much as possible TAB
complete up to one word SPC
complete and execute RET
show possible completions ?
fetch previous minibuffer input M-p
fetch later minibuffer input or default M-n
regexp: search backward through history M-x
regexp: search forward through history M-s
abort command C-g

Type C-x ESC ESC to edit and repeat the last command that used the minibuffer. Type F10 to activate the menu bar using the minibuffer.
GNU Emacs Reference Card

Buffers

select another buffer: \texttt{C-x b}
list all buffers \texttt{C-x C-b}
kil 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 ispell-region}
check spelling of entire buffer \texttt{M-x ispell-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-search}
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{C-x r c}
prefix each line with a string \texttt{C-x r t}

Abbrevs

add global abbrev \texttt{C-x a g}
add mode-local abbrev \texttt{C-x a l}
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-/}
Regular Expressions

any single character except a newline . (dot)
zero or more repeats *
one or more repeats +
zero or one repeat ?
quote regular expression special character c \c
alternative ("or") |
grouping \( ... \)
same text as nth group \n
at word break \b
not at word break \B

text entity match start match end
line ^ $
word \(< \) \(> \)
buffer \; \;

class of characters match these match others
explicit set [ ... ] [^ ... ]
word-syntactic character \w \W
character with syntax c \sc \Sc

International Character Sets

specify principal language \texttt{M-x set-language-environment}
show all input methods \texttt{M-x list-input-methods}
enable or disable input method \texttt{C-\textbackslash}
set coding system for next command \texttt{C-x RET c}
show all coding systems \texttt{M-x list-coding-systems}
choose preferred coding system \texttt{M-x prefer-coding-system}

Info

enter the Info documentation reader \texttt{C-h i}
find specified function or variable in Info \texttt{C-h C-i}

Moving within a node:

scroll forward \texttt{SPC}
scroll reverse \texttt{DEL}
beginning of node . (dot)

Moving between nodes:

next node 
previous node 
move up 
select menu item by name 
select nth menu item by number (1-9) 
follow cross reference (return with 1) 
return to last node you saw 
return to directory node 
go to any node by name 

Other:

run Info tutorial \texttt{h}
quit Info \texttt{q}
search nodes for regexp \texttt{M-s}
Registers

save region in register: \texttt{C-x r s}
insert register contents into buffer: \texttt{C-x r i}
save value of point in register: \texttt{C-x r SP}
jump to point saved in register: \texttt{C-x r j}

Keyboard Macros

\texttt{start defining a keyboard macro: C-x (}
\texttt{end keyboard macro definition: C-x )}
\texttt{execute last-defined keyboard macro: C-x e}
\texttt{append to last keyboard macro: C-u C-x (}
\texttt{name last keyboard macro: M-x name-last-kbd-macro}
\texttt{insert Lisp definition in buffer: M-x insert-kbd-macro}

Commands Dealing with Emacs Lisp

\texttt{eval selp before point: C-x C-a}
\texttt{eval current defun: C-x M-x}
\texttt{eval region: M-x eval-region}
\texttt{read and eval minibuffer: M-:}
\texttt{load from standard system directory: M-x load-library}

Simple Customization

\texttt{customize variables and faces: M-x customize}

Making global key bindings in Emacs Lisp (examples):

\begin{itemize}
  \item \texttt{(global-set-key \textbackslash"C-cg\" \textquoteleft goto-line\textquoteright\)}
  \item \texttt{(global-set-key \textbackslash"M-\textnum\" \textquoteleft query-replace-regexp\textquoteright\)}
\end{itemize}

Writing Commands

\begin{verbatim}
(defun command-name (args)
 "documentation" (interactive "template")
 body)
\end{verbatim}

An example:

\begin{verbatim}
(defun this-line-to-top-of-window (line)
  \"Reposition line point is on to top of window.\n  With ARG, put point on line ARG.\""
  (interactive \"p\")
  (recenter (if (null line)
  0
  (prefix-numeric-value line))))
\end{verbatim}

The \texttt{interactive} spec says how to read arguments interactively.
Type \texttt{C-h f interactive} for more details.

\begin{footnotesize}
\begin{itemize}
  \item Copyright © 1987 Free Software Foundation, Inc.
  \item v2.1 for GNU Emacs version 26, June 1987
  \item designed by Stephen Gilmore
  \item Permission is granted to make and distribute copies of this card provided the copyright notice and this permission notice are preserved on all copies.
  \item For copies of the GNU Emacs manual, write to the Free Software Foundation, Inc., 68 Temple Place, Suite 330, Boston, MA 02111-1307 USA
\end{itemize}
\end{footnotesize}