

OFFICE OF INFORMATION TECHNOLOGY

Getting Started with Emacs

Kenny Burns

Scott Cole



©1996 BOSTON UNIVERSITY

Permission is granted to make verbatim copies of this document provided copyright and attribution are maintained.

Office of Information Technology
111 Cummington Street
Boston, Massachusetts 02215

READ ME FIRST...

- This handout accompanies the Information Technology Tutorial, “An Introduction to Emacs.” Additional copies are available from the Office of Information Technology, 1111 Cunningham Street, 9a–5p Monday through Friday.
- Consider your familiarity with the following topics before attending this tutorial:
 - Required knowledge...** **Available tutorial...**
 - Basic UNIX experience “An Introduction to UNIX” (see IT’s tutorial schedule)
 - Using a mouse See Appendix A of this handout
- If you have suggestions or comments for the authors of this handout please send electronic mail to emacs-tutorial-handout@bu.edu.
- The text file `sample.txt` is used throughout this handout. It provides the user with a file to use for experimentation during the tutorial. This file is available via anonymous ftp for users who are not in the tutorial. See “`Sample.txt`” on page one for more information.

CONTENTS

Read Me First.....	i
Contents	i
Introduction.....	1
Emacs Command Format.....	2
The Emacs Display	3
Getting Started (Lab 1, Part 1)	4
Moving Around (Lab 1, Part 2)	5
Killing Text (Lab 1, Part 3).....	6
Moving Text & Undoing Changes (Lab 2, Part 1)	7
Searching & Replacing (Lab 2, Part 2)	8
Buffers & Frames (Lab 3)	9
More Help (Lab 4)	10
Appendix A: Using the Mouse.....	11
Appendix B: Lab Command Summary	12



INTRODUCTION



This tutorial applies to both the text-based Emacs, where all commands are executed with keystrokes, and the newer version which supports a graphical user interface. The graphical version supports a mouse interface, menu bar, scroll bars, and other features. These are available only on X displays. If you are not familiar with this sort of interface refer to appendix A.

Emacs Defined

Emacs is a self-documenting, customizable, extensible, real-time display editor. Emacs is classified as a display editor because it displays the contents of a file as it's typed rather than requiring a special command to view the contents. It is real-time because the screen updates as each character is typed. Changes can be seen immediately as they are input. We call Emacs self-documenting since its complete documentation is built-in and is available at any time with just a keystroke (**C-h**). Emacs is customizable and extensible because commands can be re-mapped to different keystrokes, and new commands can be created and added to its basic functionality. Because the source to GNU Emacs is freely-distributed, its flexibility is further enhanced.

Emacs is found on a variety of operating systems and architectures, most notably on UNIX and VMS machines. The particular variety of Emacs we will be discussing in this document is GNU Emacs (version 19), a widely-installed distribution of Emacs from the Free Software Foundation.

Conventions Used in this Document

Each section has two accompanying labs. Labs that are labeled with a  (text icon) describe procedures and commands for the text-based version of Emacs while labs marked by a  (mouse icon) describe the mouse-based version. We suggest working through the text lab first then the mouse-based lab second in each of the sections. Emacs commands are shown in *bold italic* type throughout this document.

How Emacs Interacts with Files

Emacs, as with editors in general, does not directly access a file's contents. Instead, Emacs copies the contents of the file into a buffer (see right side bar). A buffer is a temporary work area where Emacs allows you to edit a text file. The real file is left untouched as a backup until Emacs is given the explicit command to save the buffer upon which time the changes are saved permanently.

Emacs also supports the "version" capability of the operating system on which it resides. For instance, on UNIX, the previous edition of a file exists as the filename appended with a tilde (~). On VMS, there can be many backup versions of a file, such as FILE.TXT;1, FILE.TXT;2, and FILE.TXT;3, where version 3 is the most recent.

SAMPLE.TXT

Retrieving the sample.txt Text File

If you already have a file named sample.txt you may want to rename it before continuing.

The following boldfaced commands illustrate the process to transfer the sample.txt file from the bu.edu archive to your home directory. Note that the password should contain your login name as well as the current host name.

```
host% ftp bu.edu
Connected to bu.edu.
220 bu.edu FTP server ready.
Name (ftp.bu.edu:tutor1): anonymous
331 Guest login ok
Password: your_login_name@host_name
230 Guest login ok restrictions apply.
ftp> cd /examples/emacs
250 CWD command successful.
ftp> get sample.txt
200 PORT command successful.
```

The machine will transfer the sample.txt file to your account now. After transferring the file logout from the ftp server as shown below.

```
226 Transfer complete.
915 bytes received in 0.4 seconds
ftp> bye
```

EMACS COMMAND FORMAT

The Meta Key

The ASCII character set is a standardized set of 128 individual character values which include letters, numbers, symbols, and control characters. Emacs uses an 8-bit extended ASCII, which means that all 128 characters of the ASCII set can be prefixed by a meta key, thus making a total of 256 characters possible. The meta-key is denoted in both this handout and most Emacs references by a capital M. *Although it can be defined to be just about any key, we have chosen the escape key since it is available on most keyboards.* The escape key is usually labeled as ESC on the keyboard. Here is an example of a meta command: *M-x*. This means press and release the meta key (which we have defined to be the ESC key) and then type an x character. *Do not hold down the meta key*, this will inadvertently invoke a lisp evaluation function. If this happens by mistake, press *C-g* to exit.

The Control Key

Control characters are formed by pressing and holding down the keyboard key labeled Control then pressing and releasing another character. The control key is denoted by the capital letter C. A control sequence, such as *C-g*, is considered one keystroke. Here is an example of a control sequence: *C-g*. This means press and hold the control key and type a g character. *Do not hold down the g.*

In a two-keystroke key sequence, the first keystroke is referred to as the prefix key. It is important to remember that keys in Emacs have no intrinsic meanings themselves. Emacs assigns the actual commands to functions, which are in turn bound to the keys. With things arranged in this manner, it is possible to bind functions to any keys desired, and thus customize the user interface. See “More Help” for more information on these advanced features. See the Emacs reference card, available from Information Technology, for a list of function names.

A Note for VT220 Users

The VT220's method for mapping keys can cause problems when using Emacs. If the escape key produces a “23~” and moves the cursor to an inappropriate location, changing the hardware setup of your terminal to a VT100 should resolve the problem.



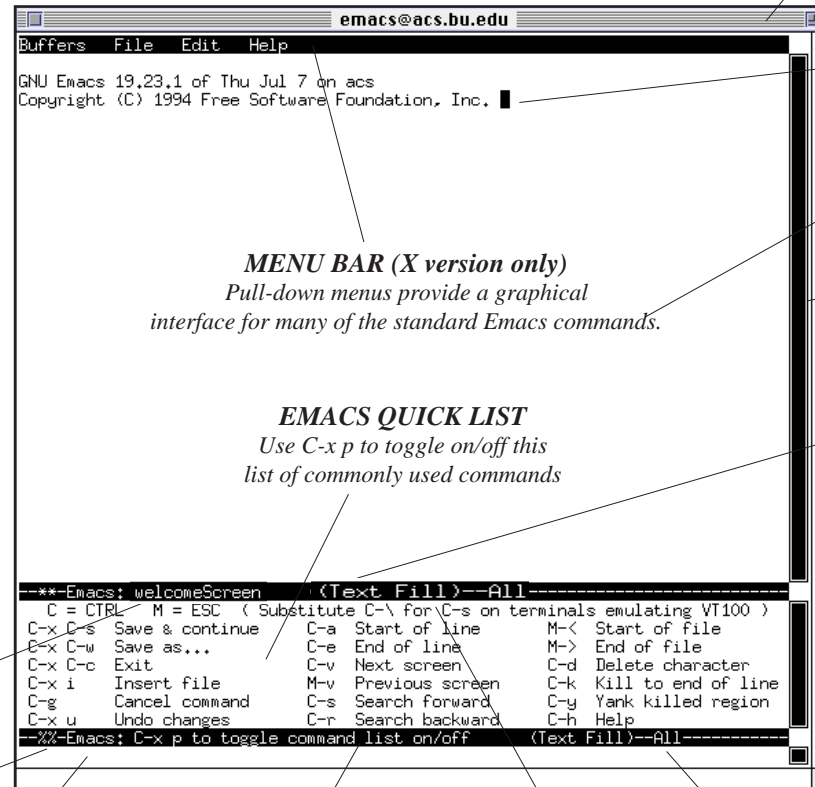
The meta key can be defined to be just about any key on the keyboard. We have chosen the escape key, which is labeled as ESC on most keyboards. A capital m (M) is used in this handout when referring to the Meta Key.

THE EMACS DISPLAY

The Emacs display consists of several parts: the text window, mini-buffer (also called the echo area), mode line, menu bar (X version only) and the frame. The Emacs image on your particular screen might not look exactly like the diagram below because of your particular environment. If you are using Emacs in an X environment rather than via an ascii terminal then your display will more closely resemble the diagram. In either case, X environment or ascii terminal, all of the Emacs commands are available from the keyboard. The text window is the area which displays the current section of the open buffer. The last line of the text window is called the mode line and is displayed in reverse video.

Directly underneath the mode line, on the last line of the entire Emacs display, is the mini-buffer. When the first keystroke of a key sequence that requires a second keystroke has been entered, the cursor moves to the mini-buffer to prompt for the rest of the command. To cancel a command in progress, type **C-g**.

The entire Emacs display is contained within a frame. When using the X version multiple frames can be activated.



FRAME
Each Emacs window is called a frame. Multiple frames are only available when working in an X environment.

CURSOR
The point of insertion and deletion.

TEXT WINDOW
Displays the current section of the open file.

SCROLL BAR (X version only)
Left click to scroll to the top of a buffer, right click to scroll to the bottom, and middle click/drag for elevator scrolling.

MAJOR MODES
Displays the current major modes as described below:
Fundamental mode the default mode
Text mode for writing text
Indented text mode indents all text
Picture mode for creating simple line drawings
C mode for writing C programs
FORTRAN mode for writing FORTRAN programs
Emacs LISP mode for writing EMACS LISP functions
LISP mode for writing LISP programs
LISP interaction mode writing/evaluating expressions
nroff mode for formatting files in nroff
TeX mode for formatting files in TeX
LaTeX for formatting files in LaTeX
Scribe mode for formatting files in Scribe
Outline mode for writing outlines
View mode for viewing files but not editing

MENU BAR (X version only)
Pull-down menus provide a graphical interface for many of the standard Emacs commands.

EMACS QUICK LIST
Use C-x p to toggle on/off this list of commonly used commands

BUFFER NAME
Displays the name of the current buffer.

FILE STATUS
Indicates the status of the current file and buffer:
** modifications
-- no modifications
%% read-only access

MODE LINE
Displays information about the buffer name and the major and minor modes.

MINI-BUFFER
Displays & prompts for commands requiring two keystrokes.

MINOR MODES
Displays the current minor mode as listed below:
Abbrev mode allows for use of word abbreviations
Fill mode enables word wrap
Overwrite mode replaces characters rather than inserting
Auto-save mode saves file every 300 keystrokes (default)

BUFFER DISPLAY
Shows what percentage of the buffer is displayed:
All the entire buffer is visible on screen
Top cursor is at the beginning of the buffer
Bot cursor is at the end of the buffer
50% cursor is at middle of the buffer

GETTING STARTED

Lab 1 shows the process used for creating and saving an Emacs file. See “Appendix A: Using the Mouse” if you are not familiar with the operation of a graphical user interface before completing this lab.



Lab 1, Part 1 Creating and Saving a File

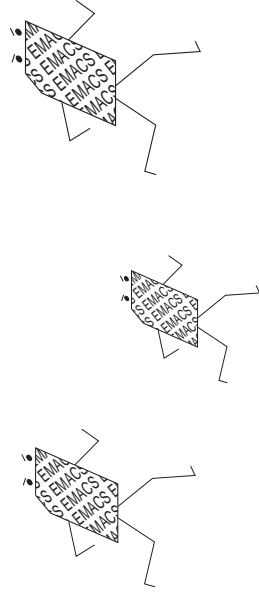
- Start up Emacs with a new file called test by entering *emacs test* from the UNIX prompt and *press the return key*.
- An empty Emacs buffer will appear on your screen. *Type a sentence into the text buffer* and save it with the command *C-x C-s*.
- Enter *C-x C-c* to exit Emacs.



- Start up Emacs with a new file called test by entering *emacs test* from the UNIX prompt and *press the return key*.

When you are using Emacs in an X environment it may be more convenient to put the Emacs session into the background by appending an ampersand (&) when starting Emacs. For example, *emacs test &* invokes an Emacs frame and allows you to continue entering additional commands at the UNIX prompt, such as *emacs test2 &*. See “Buffers & Frames” for more information.

- An Emacs buffer containing your sentence will appear on screen. *Type another sentence into the text buffer* and save it by choosing *Save buffer..* from the *File menu*.
- Choose *Exit Emacs* from the *File menu* to end this session.



MOVING AROUND

Lab 1, part 2 shows how to move around within the Emacs text buffer. See “Appendix A: Using the Mouse” if you are not familiar with the operation of a graphical user interface before completing this lab. The cursor, sometimes called a point, is the point of insertion and the point of deletion. In other words, when characters are typed they appear at the location of the cursor and when the backspace key is typed, characters will disappear from the location immediately left of the cursor.

To insert and delete characters, it is necessary to move the cursor to the desired location in the text buffer. The following lab utilizes the most commonly used commands for changing the location of the cursor. Lisp function names, which have been bound to the keystrokes are also listed. To notify Emacs that a Lisp function name has been input rather than a keystroke type *M-x* before the Lisp function name. For example, to move forward one character using the Lisp function name rather than *C-f*, type: *M-x forward-char*. For a more complete list of Lisp function names obtain a copy of the Emacs Reference Card from the Office of Information Technology or refer to appendix B of this document.



Lab 1, Part 2 Moving Around

- Edit the file sample.txt by entering *emacs sample.txt* at the UNIX prompt and **press the return key**.
- Practice moving forward and backward, from one line to the next, and to the beginning and end of the buffer. Use the table of commands below:

	<i>Description</i>	<i>Lisp Function Name</i>
<i>C-f</i>	<i>next character</i>	<i>forward-char</i>
<i>C-b</i>	<i>previous character</i>	<i>backward-char</i>
<i>C-n</i>	<i>next line</i>	<i>next-line</i>
<i>C-p</i>	<i>previous line</i>	<i>previous-line</i>
<i>C-e</i>	<i>end of line</i>	<i>end-of-line</i>
<i>C-a</i>	<i>beginning of line</i>	<i>beginning-of-line</i>
<i>M-></i>	<i>end of buffer</i>	<i>end-of-buffer</i>
<i>M-<</i>	<i>beginning of buffer</i>	<i>beginning-of-buffer</i>
<i>C-v</i>	<i>forward one screen</i>	<i>scroll-up</i>
<i>M-v</i>	<i>back one screen</i>	<i>scroll-down</i>



- Edit the file sample.txt by entering *emacs sample.txt* at the UNIX prompt and **press the return key**.
- Using the mouse position the on-screen I-beam some place in the text buffer.
- **Click the mouse button** to move the cursor to this new location in the buffer.
- Practice moving the cursor to various locations in the buffer.
- Move the mouse to position the on-screen pointer over the scroll bars and practice scrolling through the text buffer:

Left click scrolling (scroll down)
Right click scrolling (scroll up)
Center drag scrolling (elevator scroll)

```
Buffers File Edit Help
source /usr/local/examples/SystemDotFi
#####
# The first line of this file causes c
# from /usr/local/examples/SystemDotFi
#####
# If you delete or comment out the lin
# you will need to reproduce, in this
# already provided in the system .cshrc
#####
# By copying the system .cshrc file, i
# done here, you will fail to pick up
# to the system .cshrc file in the fut
#####
# Before deciding to customize this .c
# and preferences please review the sy
#####
# Add your own commands following the
#####
unalias pushd
unalias popd
unalias cd

umask 077
unset prompt
unset autologout

set path=(. /bin /usr/local/bin /usr/

set prompt "%h %M:%d >"
set history = 200
setenv PRINTER north
set notify

limit core... 0

alias l ls -Fs
-----Emacs: .cshrc (Text Fi
```

Emacs contains many commands allowing for easy and quick traversal of the text buffer.

MOVING TEXT & UNDOING CHANGES

This lab demonstrates how to move a marked block of text to a new location within the buffer. Moving text from one location to another is accomplished in a two-set process using the kill and yank commands. The kill command removes the marked text placing it in the kill ring where it can be accessed by the yank command and pasted back into the buffer. Text is pasted into the buffer at the location of the cursor when the yank command is executed.

For a summary of moving and undoing commands see appendix B.



Lab 2, Part 1 Moving Text & Undoing Changes





- Move the first 2 lines of text to the 150th line in the buffer. You will need to mark and kill the first 2 lines. See the previous lab for assistance.
 - Move the cursor to the correct location using ***M-149*** then ***C-n*** to repeat the next-line command 149 times.
 - Yank back the last killed text (the 2 lines which you killed from the beginning of the buffer) using the ***C-y*** command.
 - Practice using the undo command, ***C-x u***, to cancel the changes.
- Move the first 2 lines of text to the 150th line in the buffer. You will need to mark and kill the first 2 lines. See the previous lab for assistance.
 - Move the cursor to the correct location by scrolling down the text buffer. It may help to locate the correct line by turning on line-number-mode with the command ***M-x line-number-mode***. However, this is an example of a situation when keyboard commands are better suited to the task than the mouse.
 - Yank back the last killed text (the 2 lines which you killed from the beginning of the buffer) by selecting ***Paste*** from the ***Edit menu***.
 - Practice using the ***Undo*** command. It is located in the ***Edit menu***.




It is a good idea to save your files (C-x C-s) every 10 or 15 minutes to help prevent losing important data.

SEARCHING & REPLACING

Emacs supports simple searches, incremental searches, and global query search and replace.

The most basic search, a simple search, is demonstrated in the  lab. A simple search locates the search string only after it has been completely specified. A more powerful search command called incremental search allows initiation of a search as the search string characters are typed. As each character of the search string is entered, Emacs will find the first occurrence of that string and move the cursor to its location. *See the left most  lab for a demonstration of an incremental search.*

Emacs provides both global and query search and replace. Global replace swaps the search string with the desired replacement throughout the entire buffer. Query replace prompts for confirmation each time the search string is located before any replacement takes place. *See the center  lab for a demonstration of the query replace search.*

For a summary of all Emacs' searching commands see appendix B.



Lab 2, Part 2 Incremental Search

- Enter the command **C-s** to start an incremental search for the string: instruments.
- The mini-buffer prompts for a search string.
- Enter only the 1st character of the search string. Notice that Emacs moves the cursor to the location of the first occurrence of the string "i".
- Enter the 2nd character of the search string. Now Emacs has moved the cursor to the point where it finds the next occurrence of the string "in".
- As the search string is entered Emacs continues its on-the-fly search. Enter the remaining characters of the search string. Notice that by the third character Emacs has located the search string.
- **Press the return key** to exit the incremental search at any time.



Lab 2, Part 2 Query Replace

- Perform a query-replace by entering the command **M-%**.
 - The mini-buffer prompts for a "string-to-be-replaced"; type **GNU** and **press the return key**.
 - Next, the mini-buffer prompts for a "string-to-replace-old-string"; type **GNU's Not Unix** and **press the return key**.
 - Emacs stops at the 1st occurrence of the string and the mini-buffer waits for one of the following commands:
 - **y** or **space bar** to replace this occurrence and go on to the next one
 - **n** or **delete key** to skip this occurrence and go on to the next one
 - **(period)** to replace this occurrence then stop the search
 - **!** to replace all remaining occurrences
 - **Press the return key** to exit query replace
- For this lab, press **delete key** twice to skip the first two occurrences without replacement then **press the space bar** to replace the third occurrence.
- The mini-buffer prints "Done" when there are no more occurrences of the search string.



Lab 2, Part 2 Simple Search

- Find the first occurrence of "GNU" using the search command. Select **Search** from the **Edit menu**.
- The mini-buffer will prompt for the search string. **Enter the search string** and **press the return key**.
- If this search string is located then Emacs will move the cursor to that location. If the search does not find the string then the mini-buffer will print a message that the search failed for your string.
- Notice that the simple search does not examine any text which precedes the cursor. There are two methods which can be used to ensure that the entire buffer is examined when using simple searches: either move the cursor to the beginning of the buffer before executing the search or, after searching, select **Search Backwards** from the **Edit menu** and enter the same search string.

BUFFERS & FRAMES

Frames

Every Emacs session is displayed within a frame. This is the “window” on-screen which displays the Emacs application (text buffer, menu, mini-buffer, mode line, etc.). When using Emacs in an X environment it is possible to edit several files at the same time by invoking the frames with an ampersand (&), for example, at the UNIX prompt enter *emacs filename &* and *press the return key*, then enter *emacs filename &* again and *press the return key*. You will see two separate frames, each editing the same file. Each frame is dis-

played in its own X window. Only a single frame is allowed in the text based version of Emacs, however, there is a method which allows for editing more than one file in the single frame.

Working with Multiple Buffers

All Emacs frames can display multiple files by splitting the text buffer into sections. Emacs has several commands used to create and remove buffers. The active buffer is always determined by the location of the cur-

sor. Lab 3 demonstrates how to create, remove, and switch between buffers.

Inserting Files Into Buffers

Files can be combined inside the text buffer by using the insert-file command. The inserted file is placed at the location of the cursor when the command was executed. The mini-buffer will prompt for the name of the file to be inserted. Emacs leaves the cursor at its original location once the file is inserted.



Lab 3 Buffers & Frames

- Append the file */etc/motd* to the end of the file *sample.txt* with the command *C-x i*. The mini-buffer prompts for a filename, type */etc/motd* and *press the return key*.
- Edit a new file called *sample1.txt* by entering *C-x C-f sample1.txt* and *press the return key*.
- Exit Emacs by entering *C-x C-c* then restart with the file *sample.txt*, *emacs sample.txt*, and *press the return key*.
- Split the buffer into two halves by entering the command *C-x 2*.
- Edit a different file (*sample1.txt*) in this new buffer by reading the file with *C-x C-f*. Remember that the cursor's location dictates which window is active.
- Use *C-x o* to move the cursor from one buffer to the other.
- Make the current buffer consume the whole screen with the command *C-x I*.
- Use *C-x b* to switch back to the other buffer.
- Close all of the open buffers and exit Emacs by entering *C-x C-c*.



Most of Emacs' buffer commands are not available from the menu bar. However, the menu bar does allow for easy selection of various buffers.

- Select the desired buffer by selecting its name from the **Buffers** menu. To see a list of buffers select **List All Buffers** from the **Buffers menu**.
- To switch between buffers simply move the cursor to the desired buffer. All commands affect the active buffer, i.e., the buffer where the cursor is currently located.
- Select **Kill Buffer** from the **File menu** to remove a buffer from the screen.

MORE HELP

On-Line

Emacs provides extensive help features which revolve around a single character, **C-h**. **C-h** is the prefix key for all the help commands. The most useful help sequence is **C-h C-h**, which prints a list of all the possible help options and then prompts for which option you would like to select. Typing a third **C-h** prints a definition of each of the options. Lab 4 demonstrates how to use some of the help commands.

The very best way to learn any application program is by using it. Type **C-h t**, or select **Emacs Tutorial** from the **Help Menu**, to start a comprehensive, self-paced Emacs tutorial which will provide valuable hands-on experience using the commands described in this handout as many others.

Not On-Line

The *GNU Emacs Manual*, by Richard Stallman, is considered the definitive Emacs document. It is available at a nominal fee from the Free Software Foundation, 675 Massachusetts Avenue, Cambridge, MA, 02139. The manual also contains a handy quick reference card. For added convenience, Information Technology sells the manual for \$15, and the quick reference card is available from the IT Help Desk.

As with any UNIX application installed on an Information Technology supported machine, when you can no longer find help on your own, please contact your departmental consultant or send email to help@your machine name.



Lab 4

Getting On-line Help

- Find all commands which contain the word “window”. Type **C-h a** to invoke command-apropos. The mini-buffer will prompt for a keyword, type **window** and **press the return key**.
- Find the name and description of a function which is bound to a key sequence using the command, **C-h k**. The mini-buffer will prompt for a key, **enter a key from the keyboard** and **press the return key**.
- Use the command **C-x p** to display the quick list.



- Find all commands which contain the word “window”. Select **Command Apropos...** from the **Help Menu**. The mini-buffer will prompt for a keyword, type **window** and **press the return key**. All commands which contain that word will be displayed in a new buffer.
- Find the name and description of a function which is bound to a key sequence by selecting **Describe Key...** from the **Help Menu**. The mini-buffer will prompt for a key, **enter a key from the keyboard** and **press the return key**. The results will be displayed in a new buffer.
- Use the command **C-x p** to display the quick list.



Emacs has a large help system which includes complete documentation on each of its functions.

APPENDIX A: USING THE MOUSE

Dragging, Scrolling, & Selecting

Pointing & Clicking

The I-beam (right) is used for positioning the cursor (left) within the text buffer.



Cursor movement is accomplished by a combination of pointing and clicking actions. The example below shows how to move the cursor from its current position (after the period in the sentence “Hello World.”) to a new location (between the two words).

- Move the on-screen I-beam pointer between the words “Hello” and “World” as shown below:

Hello I World.■

- Click the left mouse button to move the cursor to this new location.

Hello ■ World.

The cursor does not follow the I-beam nor does it move to the I-beam’s location until the mouse button is clicked. Be careful not to click the mouse button too soon while the I-beam is still moving. This is called dragging (moving the mouse and clicking at the same time) and is detailed in the following paragraph.

Dragging

Moving the mouse while simultaneously pressing a mouse button is called dragging. Dragging is used for selecting regions, pull-down menus, and scrolling.

Selecting Regions

Identify the region of text you wish to select, then move the on-screen arrow in front of the first character. Click and hold the left mouse button while dragging the arrow across the region. The region highlights (in reverse video) as it is selected but will return to normal when the mouse button is released. Release the mouse button when the region is highlighted.

Another way to select regions is to click the left button at the beginning and the right button at the end of a region. The selected text will also be copied to the kill ring.

Pull-Down Menus

Pull-Down menus are found along the top of the Emacs frame. Buffers, File, Edit and Help are all pull-down menus. Move the on-screen arrow to point to the name of a menu then click and drag the mouse downward until the desired command is selected. The commands will be “boxed” as the arrow passed over.

Some commands can only be executed at certain times; when not available they appear as greyed-out text in the menu.

Scrolling

Scrolling is necessary since only a portion of the buffer can be displayed on-screen in the Emacs window. The scroll bar and scroll box on the right side of the Emacs window allow easy traversal of the text buffer. There are 3 scrolling methods, each requiring the pointer to be located within the scroll bar.

- **Scrolling Up:** Click the left mouse button to scroll towards the beginning of the buffer.
- **Scrolling Down:** Click the right mouse button to scroll towards the bottom of the text buffer
- **Elevator Scrolling:** Click and hold the middle mouse button while dragging the mouse up and down within the scroll bar

SCROLL BAR
Located on the right side of the Emacs window. The pointer must be placed within the scroll bar to use the three different scrolling methods.

SCROLL BOX
Sometimes called an elevator, it is located within the scroll bar. The box’s location within the bar indicates the cursor’s location within the text buffer. A scroll box which is near the top of the bar means that the text window’s display is toward the beginning of the buffer.



APPENDIX B: LAB COMMAND SUMMARY

This is a summary of the commands used in the labs for this handout as well as some other helpful commands which were not demonstrated in the labs.

Working with Files

read a file C-x C-f
save a file C-x C-s
save all files C-x s
insert file into buffer C-x i

Getting Help

display quick list commands C-x p
apropos, show commands matching string C-h a
show the function bound to a key C-h c
describe a function C-h f
get mode-specific information C-h m

Incremental Search

search forward C-s
search backwards C-r
exit search return key
abort current search C-g

Query Replace

interactively replace a string M-%
Mini-buffer choices are:
replace this occurrence, go on to the next one space bar
replace this occurrence, don't move
skip to the next occurrence without replacing delete key
replace all remaining occurrences !
exit query replace escape key

Killing Text

kill one character backward/forward delete key/C-d
kill one word backward/forward M-delete key/M-d
kill text to end of line/beginning of line M-O C-k/C-k

kill sentence backward/forward C-x delete key/M-k
kill marked region C-w
copy region to the kill ring M-w
yank last thing killed C-y

Marking Regions

set a mark C-space bar
mark entire buffer C-x h
mark page C-x C-p
mark paragraph M-h
exchange point and mark C-x C-x

Moving Around

move one character backward/forward C-b/C-f
move one word backward/forward M-b/M-f
move one line backward/forward C-p/C-n
move to beginning/end of line C-a/C-e
move to beginning/end of sentence M-a/M-e
move to beginning/end of paragraph M-{/M-}
move to beginning/end of buffer M-</M->
scroll to previous/next screen M-v/C-v

Buffers

select another buffer C-x b
list all buffers C-x C-b
kill a buffer C-x k

Working in the Mini-Buffer

complete the command as much as possible tab key
complete and execute the command return key
show possible completions ?
abort command C-g