
Platform: Unix

Level of Difficulty: Beginner

This document will familiarize the user with the basic operation of the Emacs, Pico, and VI, text editors.

Introduction

A text editor is a program that manipulates plain text (ASCII). Most text editors are full screen editors (they display a full screen of text) as opposed to line editors (which display one line at a time). Emacs is a full screen text editor supported by RUCS. On RUCS UNIX machines the default version of Emacs is the GNU release of Emacs. There are a couple of different UNIX text editors available for use in creating and editing files. These UNIX text editors are applications that can be run either in a telnet session or an X-windows session.

The three most popular UNIX text editors here at Rutgers are Emacs, Pico, and VI. Using these UNIX text editors allows you to program in any computer language and translate it into code that the computer will be able to read and execute. There are also non-UNIX based text editors that are available as well. On Macintoshes you can use BB Edit and on PC's you can use Wordpad or Notepad. It suggested that you stay away from using MS Word or Word Perfect because of their auto spell check and syntax check functions. These functions create difficulties in having the code being interpreted properly and affect how smoothly your program will be executed. The difference between UNIX and non-UNIX based text editors is that by using the non-UNIX based text editors you will have to use Fetch or FTP in order to save the file to your Eden account. By using the UNIX based text editors, you can save yourself a couple of extra steps since the files will be directly saved to Eden. In this document we will only be dealing with UNIX based text editors.

Starting Emacs

To start emacs from the UNIX system prompt type:

```
er2%> emacs <filename>
```

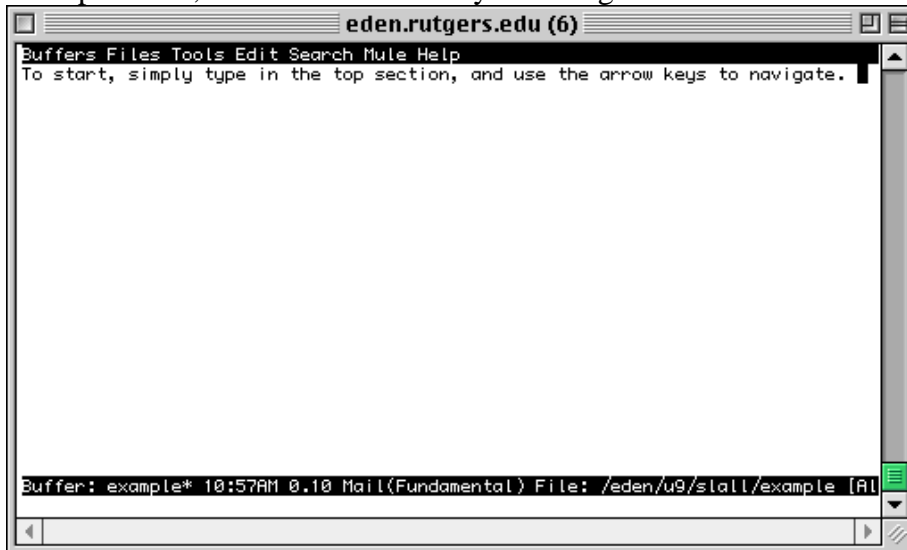
The filename is the name of the file you wish to edit or create. If you are using an x-terminal and wish to start Emacs in the same window you are currently using:

```
er2%> emacs -nw <filename>
```

The `-nw` stands for no new window.

Emacs Basics

A window now appears, with a bar across the bottom displaying information about the file, and a line below it where commands you type will appear. To start, simply type in the top section, and use the arrow keys to navigate.



All commands are keystroke based, and usually are `<Ctrl>` and a letter, or `<Esc>` and a letter. (Note that Emacs calls the `<Esc>` key Meta, and abbreviates it with a capital M.)

For example, to save your buffer, type: `<Ctrl>x <Ctrl>s`

In Emacs a buffer is an area of memory where editing is done. These changes are not reflected in files until that buffer is saved to a file. In Emacs white space is blank lines, spaces, or tabs. Indented or blank lines separate Emacs paragraphs. Emacs sentences are terminated by a paragraph separator or one of the following punctuation marks followed by two spaces: a period, an exclamation point, or a question mark.

Saving, Loading, and Exiting:

Exiting and suspending:

`<CTRL>x <CTRL>c` exits Emacs. If you have not saved all the buffers you have changed, it will ask if you want to save those buffers before you exit. NOTE: if a buffer is not associated with a file, this exit will not ask to save your work.

<CTRL>z suspends Emacs and returns to command prompt. In UNIX you can have multiple sessions suspended and resume one with either the % or fg command.

<CTRL>g exits currently executing command.

Saving and inserting files:

<CTRL>x <CTRL>s saves the current buffer into a file.

<CTRL>x s prompts you to save every modified buffer.

<CTRL>x <CTRL>f finds a file for editing, creates a new buffer.

<CTRL>x <CTRL>v visits a file for editing, discards current buffer contents.

<CTRL>x <CTRL>w writes the current buffer into a file and prompts you for a filename.

<CTRL>x i prompts you for a file to insert at the cursor.

Getting Help in Emacs

These are a few of the informational commands available within Emacs.

<CTRL>h starts the help command.

<CTRL>h a searches for commands that contain the supplied keyword.

<CTRL>h c when followed by a command keystroke, shows command name.

<CTRL>h i starts the "info" information menu of online Emacs documentation.

<CTRL>h k when followed by a command keystroke, describes the command.

<CTRL>h t starts the Emacs tutorial. Recommended method for learning Emacs.

<CTRL>h ? lists the help sub-commands.

<CTRL>h <CTRL>h lists and describes the help subcommands.

Running the Emacs tutorial:

Emacs comes with an excellent tutorial which goes through each Emacs command step by step.

<CTRL>h t starts the Emacs tutorial from within Emacs.

Or from the command line:

er2%> teach-emacs

Cursor Movement

Moving your cursor allows you to select where in the displayed text the current action is to take place. On some screens the cursor is a flashing underscore character (_), on others

it is a colored rectangle. The arrow keys work well to navigate Emacs on most systems, but there are other options for advanced users which can be more time efficient.

- <CTRL>f moves the cursor forward one character.
- <CTRL>b moves the cursor backward one character.
- <ESC> f moves the cursor forward one word.
- <ESC> b moves the cursor backward one word.
- <ESC> e moves the cursor forward one sentence.
- <ESC> a moves the cursor backward one sentence.
- <ESC>] moves the cursor forward one paragraph.
- <ESC> [moves the cursor backward one paragraph.
- <CTRL>e moves the cursor to the end of the current line.
- <CTRL>a moves the cursor to the beginning of the current line.
- <CTRL>p moves the cursor to the previous line.
- <CTRL>n moves the cursor to the next line..

Controlling the display allows you to select what part of the buffer you are looking at.

- <CTRL>l clears screen and redisplay with the line the cursor is on in the center.
- <CTRL>v scrolls to the next screen.
- <ESC> v scrolls to the previous screen.
- <ESC> < moves the cursor to the beginning of the buffer
- <ESC> > moves the cursor to the end of the buffer
- <CTRL>x [moves the cursor to the preceding <CTRL>l, or beginning of file.
- <CTRL>x] moves the cursor to the next <CTRL>l, or end of file.

Text Handling

Changing existing text requires the text handling commands.

- <ENTER > ends current line and starts new line
- <CTRL>o opens a new line at the cursor.
- <ESC> q adjusts current paragraph to fill out to the fill column (70 by default).
- <ESC>l <ESC>q justifies text. (the numeric “one”, not the letter “l”)
- deletes the character to the left of the cursor.
- <CTRL>d deletes the character at the cursor.
- <ESC> d kills (deletes) from the cursor to the end of the current word
- <ESC> kills from the cursor to the beginning of the current word.
- <CTRL>u repeats a command multiple times. To go down 10 lines type:
 - <CTRL>u 10 <CTRL>n
- <CTRL>k kills from the cursor to the end of the current line.

<ESC> k kills from the cursor to the end of the current sentence.
<CTRL>x kills from the cursor to the beginning of the current sentence.
<CTRL>t transposes the character at the cursor with the one to the left of it
<ESC> t transposes the words on either side of the cursor.
<CTRL>x <CTRL>t transposes the current line with the one above, place cursor on the next line.
<CTRL>_ undoes the previous action. Repeated undos will undo preceding actions.
<CTRL>@ places a mark at the cursor. A region is the text between a mark and the cursor.
<CTRL><SPACE> places a mark at the cursor
<CTRL>x <CTRL>x exchanges point (cursor) and mark, allows you to see both ends of a region.
<CTRL>w kills region.
<ESC> w copies region.
<CTRL>y yanks back (restore) the most recent deletion done by any of the above kill commands or by a copy.
<ESC> y yanks back (restore) a previous deletion done by any of the above kill commands or by a copy. MUST be preceded by a <CTRL>y.

HINT: an effective way to move whole regions of text from place to place is to set a mark (with <CTRL>@) at the top of the region you wish to move, then move the cursor down to the bottom of the region and then delete the region (with <CTRL>w). Then move the cursor to the place where you want this region moved to, and restore (yank) the region back in (with <CTRL>y).

Buffers

“Buffer” is the term Emacs uses to refer to the portion of memory that has a body of text in it for manipulation. A much simpler way of putting this is that a “Buffer” is the space where you type in text. For the most part this is the general term you would use to describe the space you are typing text into. Whether you are using VI, or Pico the general term for the space where you type in text is called a buffer. Often buffers contain copies of the contents of specific files. The below commands are the basic commands for working with buffers.

<CTRL>x <CTRL>b lists current buffers.
<CTRL>x b switches to an existing buffer. You will be prompted for a buffer name.
<CTRL>x 4 b splits the window, prompt for a buffer name, bring that buffer into the second window, and place the cursor in the new buffer.

Pico

Pico is a simple and user friendly text editor with an interface very similar to Pine. It's similar appearance is not a coincidence. Pico was actually created by the same people that created Pine. Consequently, Pico is the default editor for Pine. To start up pico, just type *pico* at your eden prompt and the editor will open up to a new untitled buffer. Its the same syntax as when starting emacs. If you want to open up a specific file in Pico, just type at the prompt:

```
toolbox>pico <filename>
```

The file named <filename> will then open up in the Pico editor. To edit text in Pico, simply begin typing. The arrow keys will allow you to move around the window.

Pico commands:

Pico relies heavily on keyboard commands. The mouse is still useful for highlighting text, however you have to use keyboard commands to execute cut and paste. Most of the important commands in Pico are listed at the bottom of the Pico window (the '^' symbol is used to represent the <CTRL> key.) Some of the most useful commands are highlighted here.

- <CTRL>o saves the file that you are editing
- <CTRL>r opens a new file and inserts all its contents at the cursor
- <CTRL>k cuts a line of text
- <CTRL>u pastes a line of text that was just cut
- <CTRL>x exits pico (it will prompt you to save if you've made changes that you haven't saved)
- <CTRL>g brings up the help screen
- <CTRL>j formats (justifies) the current paragraph
- <CTRL>t brings up the spell checker
- <CTRL>r inserts an external file at the current cursor position
- <CTRL>w brings up a finder (where is) for text, neglecting case

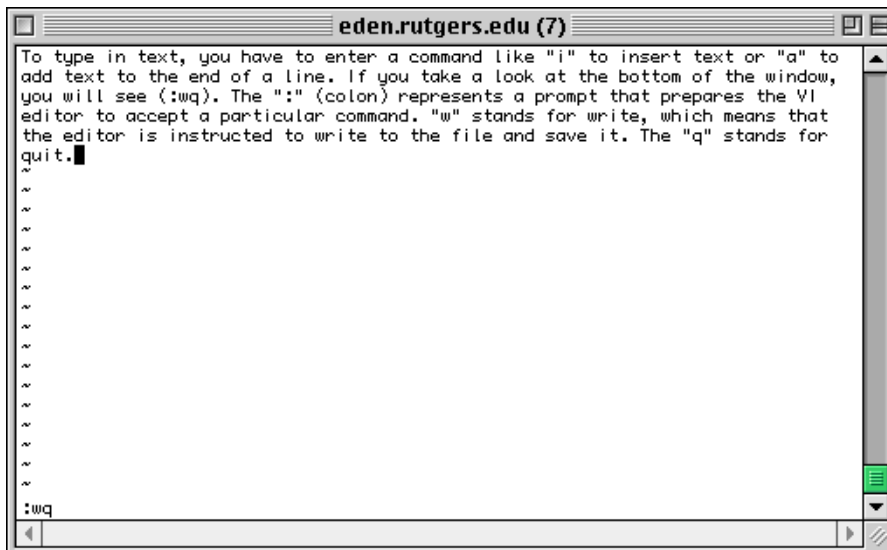
<CTRL>v moves forward a page of text
<CTRL>y moves backward a page of text

There are more commands in Pico than those that are listed at the bottom of the screen. These commands are explained in the help screen that appears when you press <CTRL>g

VI

VI is an older text editor from the late 70's. While it might seem a bit complicated, VI has become a very popular text editor here at Rutgers. The benefit to knowing VI is that it is an editor that is used on every UNIX system. So while some UNIX systems might have Emacs or Pico, because it is older, they all have VI. The commands in VI are slightly more difficult to grasp. Once you have some practice with it, however, you will find that it is a very useful text editor. To open up VI, just type *vi* at the command prompt. If you want to open up a specific file, type:

```
remus>vi <filename>
```



VI has two main modes: command mode and input mode. In command mode only commands can be entered (no text can be added) and in input mode, plain text can be typed in. When you start VI, you start in command mode. To type in text, you have to enter a command like either *i* to insert text or *a* to add text to the end of a line. After you enter this command, you can type text. When you're done typing, hit the <esc> key to return to command mode. It can be very complicated to keep track of which mode you are in. Some important commands in command mode are:

i	insert text (to get out of inserting text you have to hit <esc>)
a	inserts text to already existing lines (again you have to use <esc> to get out of it)
G	will take you to the end of a file
1G	will take you to line one of the file
r	replace one character

:f show filename, current line, and size
u will undo the last change made
:w<ENTER> will save changes to the file
ZZ will exit VI and save the file
:q!<ENTER> will exit VI without saving the file
x will delete a single character
dd will delete a whole line of text
J to join two lines
5J will join five lines (any number in front of the J command will join that many lines together).

Moving around using the h,j,k, and l, commands:

One of the more complicated ideas to grasp is how to move through text in VI. In order to scroll up and down, left and right in VI you have to use certain letter keys, as opposed to the arrow keys.

h move to the left
j move down
k move up
l move to the right

Where to get more help:

<http://www.gnu.org/manual/emacs-20.3/emacs.html> – The GNU online manual for Emacs.

<http://www.washington.edu/pine/> - The Pine information center

<http://www.pitt.edu/~document/pico/pico.html> – Intro. to Pico

<http://www.eng.hawaii.edu/Tutor/vi.html> – Intro. to VI

Or ask a consultant on duty.