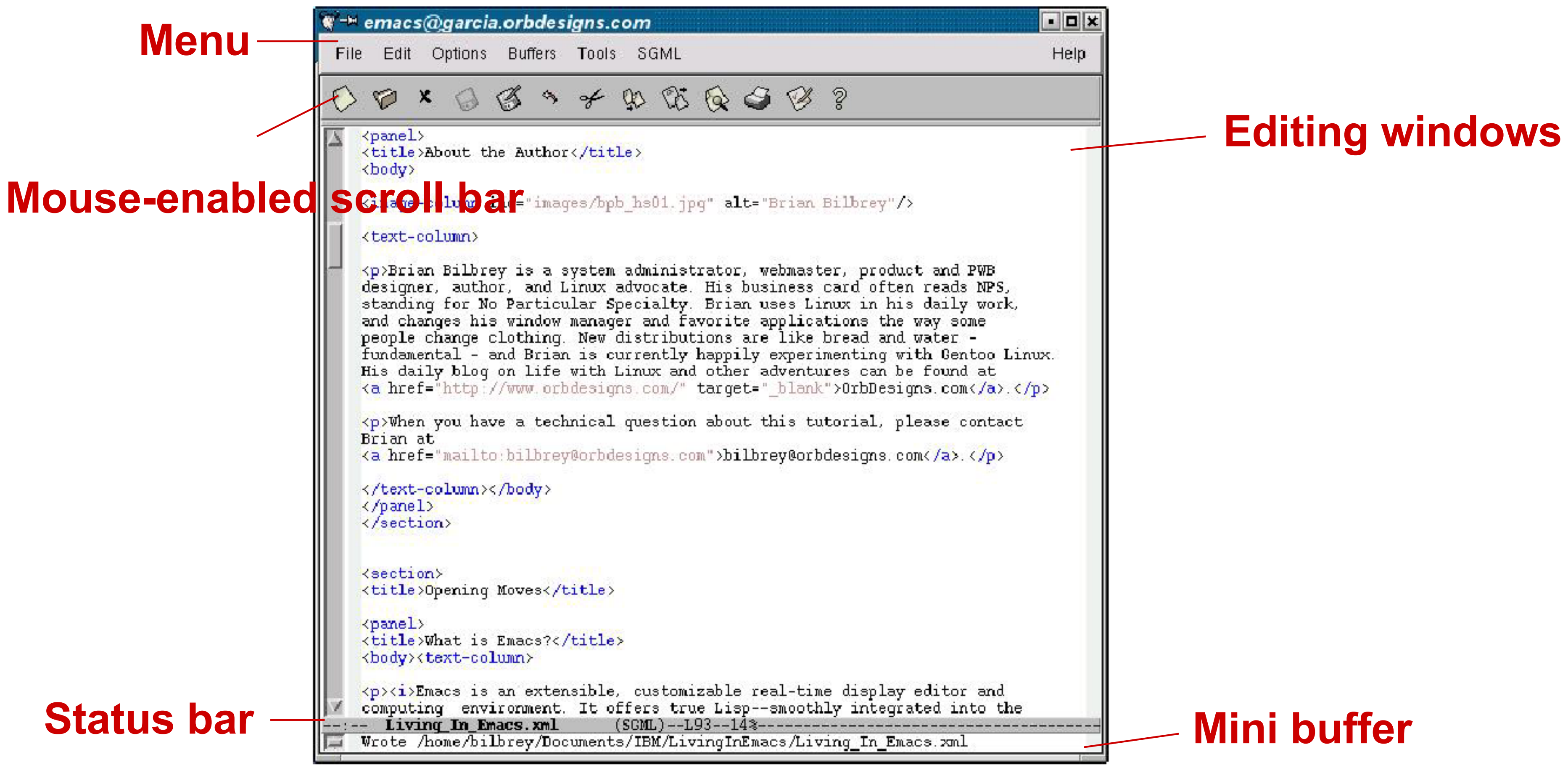
Emacs

What is Emacs?

- Emacs is the extensible, customizable, self-documenting real-time display editor.
- Emacs can do so very many different things so well that it would make a fine operating system.

The Emacs view


Menu
Editing windows
Mouse-enabled scroll bar
Status bar
Mini buffer
emacs@garcia.orbdesigns.com
File Edit Options Buffers Tools SGML Help

```
<panel>
<title>About the Author</title>
<body>

<image-column id="images/bpb_hs01.jpg" alt="Brian Bilbrey"/>

<text-column>

<p>Brian Bilbrey is a system administrator, webmaster, product and PWB
designer, author, and Linux advocate. His business card often reads NPS,
standing for No Particular Specialty. Brian uses Linux in his daily work,
and changes his window manager and favorite applications the way some
people change clothing. New distributions are like bread and water -
fundamental - and Brian is currently happily experimenting with Gentoo Linux.
His daily blog on life with Linux and other adventures can be found at
<a href="http://www.orbdesigns.com/" target="_blank">OrbDesigns.com</a>.</p>

<p>When you have a technical question about this tutorial, please contact
Brian at
<a href="mailto:bilbrey@orbdesigns.com">bilbrey@orbdesigns.com</a>.</p>

</text-column></body>
</panel>
</section>


<section>
<title>Opening Moves</title>

<panel>
<title>What is Emacs?</title>
<body><text-column>

<p><i>Emacs is an extensible, customizable real-time display editor and
computing  environment. It offers true Lisp--smoothly integrated into the
--:--  Living_In_Emacs.xml      (SGML)--L93--14%--------------------------------
Wrote /home/bilbrey/Documents/IBM/LivingInEmacs/Living_In_Emacs.xml
```

Emacs keystroke conventions

- C-<chr> == Control + character, pressed at the same time.
- M-<chr> == Meta + character, pressed at the same time.

p.s. Meta might be the Alt key, or the Esc key

Commands and key-bindings

- Emacs implements a version of LISP to build its commands and extensions.
- All commands have names, like `Buffer-menu-bury`, `backward-char`.
- Many of the commands are bound to key combinations, prefaced with the Control and Meta keys.

First instructions

- C-x C-c ：Quit the Emacs
- C-x C-f ：open a file
- C-x C-s：save my work and continue typing
- C-x C-w： save as
- C-space: set mark
- C-w: cut text to the cursor position after marking
- M-w: copy text to cursor position post marking
- C-y" paste txt

Emacs cut and paste

- C-d： deletes the character under the cursor
- M-d： kill-word
- M-Delete： backward-kill-word
- M-k： kill-sentence
- C-x： Delete backward-kill-sentence
- C-k： kill-line
- C-y ： *yank* the most recent block

The universal argument

- C-u： `universal-argument`. It can be used as a prefix for a great number of other actions

- Ex：type `C-u 3 C-k` kills three lines.

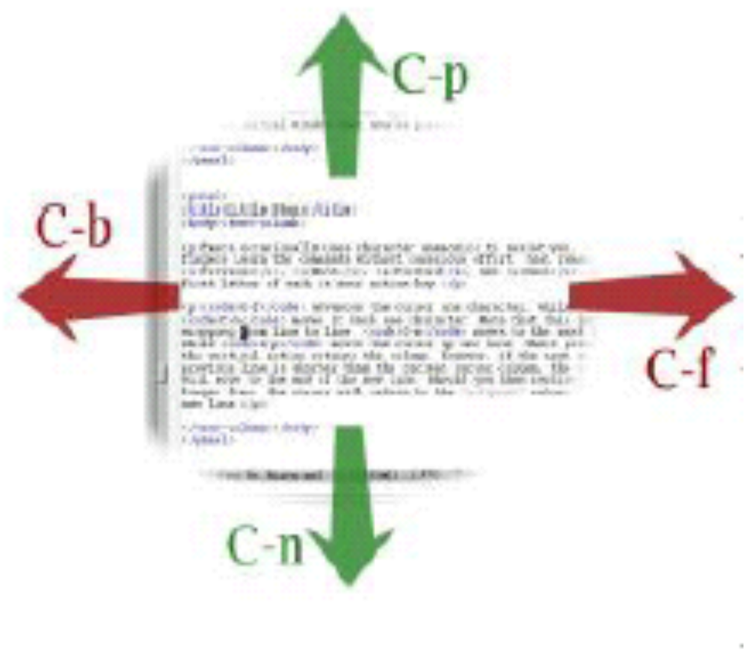
Basic operations in review

Key-binding	**Action (command)**
`C-g (Esc Esc Esc)`	`keyboard-quit` to get out of a command that's been started
`Backspace`	`backward-delete-char`
`Delete (C-d)`	`delete-char`
`C-x u (C-_)`	`advertised-undo`
`M-d`	`kill-word`
`M-Delete`	`backward-kill-word`

Basic operations in review (conti.)

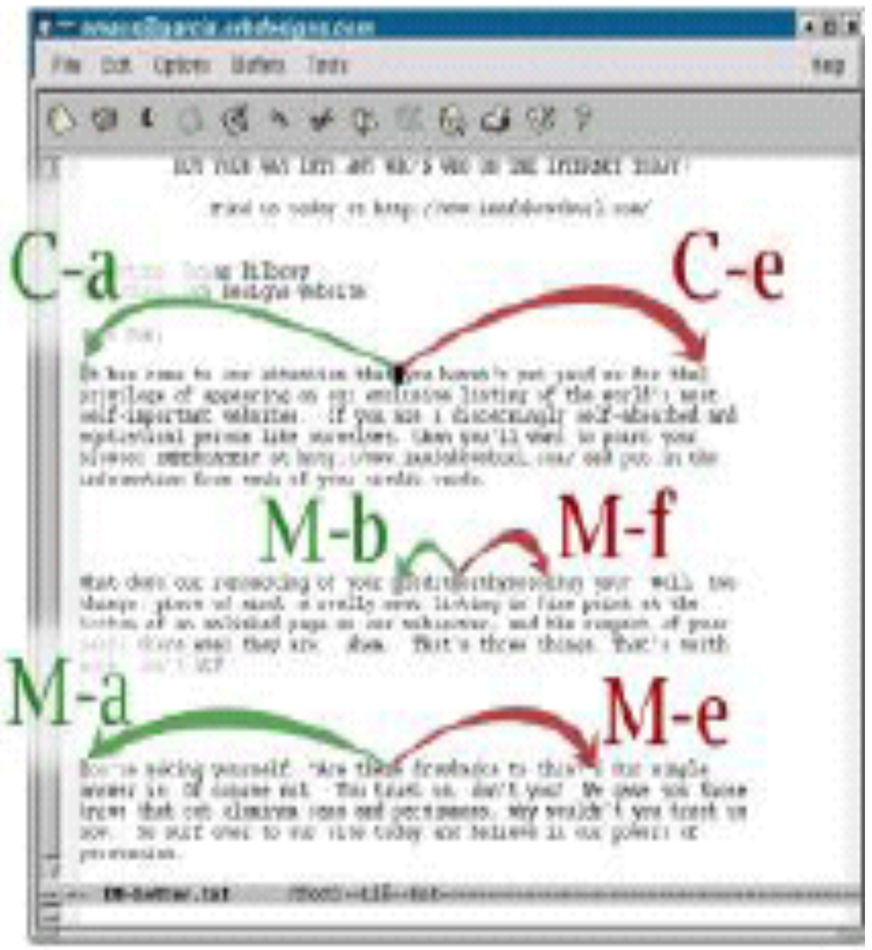
Key-binding	**Action (command)**
M-k	kill-sentence
C-x Delete	backward-kill-sentence
C-k	kill-line
C-y	yank is the paste equivalent
M-y	Traverse the kill ring, must follow C-y
C-u, C-u N	universal-argument, adds count prefix to commands

Little steps



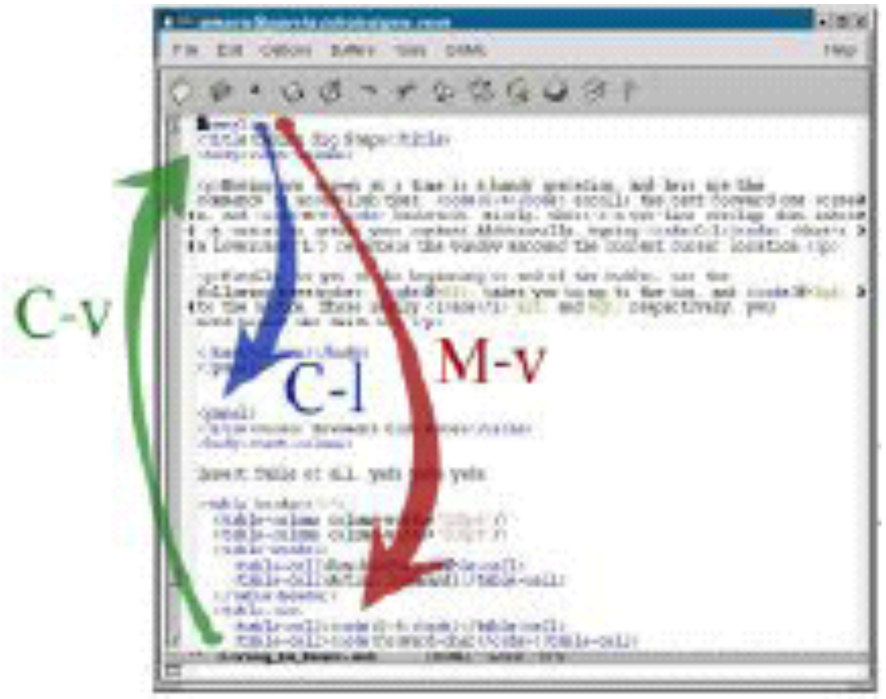
- `C-f` advances the cursor one character
- `C-b` moves it back one character
- `C-n` moves to the next line
- `C-p` moves the cursor up one line

Words, lines, and sentences



- `C-a` takes you to the first column in the current line
- `C-e` takes you to the line's end
- `M-b` moves back one word
- `M-f` moves the cursor forward one word
- `M-a` takes us backward to the beginning of the current sentence
- `M-e` moves forward in the same manner, relative to sentence ends

Taking big steps



- `C-v` scrolls the text forward one screen
- `M-v` scrolls the text backward one screen
- `C-l` re-centers the window around the current cursor location

Incremental searches

- C-s： isearch-forward
- C-r ： isearch-backward

Regexp searches

- ESC C-s： start a forward regexp search
- ESC C-r： start a backwards regexp search

Ex：I have the words *bartok* and *footok* someplace in my text. I want to find the closest instance of either one.

```
ESC C-r bar\|foo
```

Replacing text

- `M-X replace-string`: This is followed by the target string/expression and the replacement string. Replacement is unconditional and forward from the cursor location only.
- `M-%`: query-replace

Windows in Emacs

- `C-x 2`： split them horizontally
- `C-x 3`： split them vertically
- `C-x o`： switch between visible windows
- `C-x 1`： *maximize* the window that currently contains the cursor and close other windows.

Buffers in action

- Type `C-x C-b`. Your listing should resemble this:

```
 MR Buffer          Size Mode          File
 -- ------          ---- ----          ----
.*  practice1.txt 490  Text          ~/practice1.txt
    test2.txt     1    Text          ~/test2.txt
    test1.txt     0    Text          ~/test1.txt
*   *scratch*     191  Lisp Interaction
*   *Messages*    501  Fundamental
```

Buffers in action (conti.)

- The MR column reflects the "Modified" and "Read-Only" status of each buffer.
- Buffer (name), Size and File are self-explanatory.
- Switch to the buffer listing window (using C-x o), and press Enter to select it.

More about buffers

- `C-x b`: get a prompt in the mini-buffer, and then type the name of the destination buffer Press Enter to open that buffer in the current window.
- `C-x k`: kill the current buffer

Modes

- Modes are the methods by which Emacs features are expressed in the context of specific types of content.
- That is, indenting behaves differently in a C source code file than in an HTML file or in a letter to your boss.
- There are two different types of modes: major and minor.

Modes (conti.)

- `M-x valid mode name`：set the mode of a buffer

Ex：If I open a file named *bob.txt*, the buffer will open in text-mode. To start working in c-mode, I can type this:

```
M-x c-mode
```

Compiling code

- Type `M-x compile` and the prompt in the mini-buffer reads, `Compile command:`

  Type in

  ```
  gcc -o hello hello.c
  ```

- To see if my program works, I'll run it from inside Emacs:

  ```
  M-! ~/hello.
  ```

  There in the mini-buffer is my output:

  ```
  Hello, World!
  ```

Connectivity in Emacs

- `C-x m`: start a new e-mail message
- `M-x browse-url-lynx-emacs`: invoke Lynx, enter the URL