GNU Emacs

Rodrigo Decuir
ojo@ciencias.unam.mx

Semestre 2025 - 1

Resumen

Notas para el curso propedéutico dirigido a los estudiantes de primer ingreso. El objetivo principal es capacitar a los chicos para ser autosuficientes con *emacs*, es decir, que una vez terminado el propedéutico puedan encontrar las respuestas a las preguntas que les vayan surgiendo por su cuenta.

Temas a tratar: filosofía del proyecto GNU, terminología de emacs como la base de su funcionamiento, comandos básicos para comenzar con su *uso*, movimiento, edición de archivos y compilación de programas.

Índice

1. Introducción

1.1. ¿Qué es un editor de texto?

Un editor de texto es un programa que nos permite **crear** y **manipular** archivos de texto plano.

1.2. IDEs...

El motivo principal por el que se les hace hincapié en no usar IDEs recae en que no se puede estudiar ni cambiar su código fuente y parte de la ideología que se les busca transmitir en la carrera se encuentra fuertemente guiada por la idea de que **no somos usuarios finales**, somos los que hacemos y mantenemos los programas que usan los *normies*. En este sentido, lo correcto es usar programas diseñados "para programadores", aquellos en los que se puede contribuir a su código abierto estudiando y cambiando su código fuente.

Por otro lado, al usar software propietario (aquel que no es de código abierto) se corre el riesgo de ser víctima de acciones maliciosas por parte de sus desarrolladores. El mes pasado salió un artículo en medium, en el que mediante una investigación preliminar usando google OSV scanner (una herramienta de google que encuentra vulnerabilidades afectando dependencias de proyectos) se descubrieron 1,283 extensiones con dependencias maliciosas en el marketplace de VSCode.

1.3. Breve historia

El primer programa del proyecto del proyecto GNU (aquel que estableció la ideología que seguimos, la ideología del software libre) fue el editor de texto **GNU Emacs** o cómo lo conocemos hoy en día Emacs.

El desarrollo de Emacs comenzó en 1976 en el laboratorio de inteligencia artificial del MIT (CSAIL) y se le atribuye a principalmente a Richard Stallman.

El que este editor de texto sea de software libre también se lo debemos a Stallman, pues James Gosling (el padre de java) estuvo a nada de volverlo software propietario.

1.4. La guerra entre vi y emacs

También llamada la *guerra santa* para hacer referecia a la rivalidad que ha habido entre ambos editores practicamente desde la creación de vi.

A manera de broma Richard Stallman decidió fundar La Iglesia de Emacs (una religión paródica) y se autonombró San IGNUcius, un santo de carácter evangelizador. En las conferencias que ha dado, para su interpretación suele vestirse con una túnica negra, colocarse en la cabeza el plato de un antiguo disco duro, a modo de aureola dorada, y sostener su computadora portátil que asegura sólo contiene software libre.

Para la Iglesia de Emacs es claro que Vi es el editor del infierno y se refiere a él por su nombre completo: **vi vi vi** (666, en romano), el editor de la bestia. Sin embargo, no se opone al uso de vi, pues ve su uso no como un pecado, sino como una forma de sufrimiento autoimpuesto.

Richard Stallman como San IGNUcio (Emacs vs Vi)

Para los que no se hayan dado cuenta, están en tierra santa, de hecho la Facultad de Ciencias es una de las arquidiócesis de La Iglesia de Emacs y nuestro arzobispo es Canek.

2. Comandos básicos

2.1. Abrir emacs

Hay dos formas de abrir emacs desde la terminal, de forma normal o en un proceso distinto (independiente de la terminal):

```
$ emacs        $ emacs &
```

Lo única diferencia es que el proceso distinto les permite seguir usando la terminal que abrieron.

Lo primero que vemos al iniciar Emacs es una pantalla de ayuda, si pasan su mouse por encima de los links (sin hacer click) pueden ver que nos aparecen dos opciones para seguir el link: mouse-1 que es hacer *click* o **RET** (por return) que es apretar Enter, Emacs se desarrolló pensando en solo usar el teclado, pues es más rápido editar así, una vez que tengan Emacs pueden reafirmar lo que vamos a ver hoy (viernes) y el martes dando click o presionando Enter en alguno de los links.

2.2. Ctrl y Alt

La idea básica detrás del funcionamiento de Emacs es: que **todo es un comando** (las combinaciones de teclas ejecutan comandos escritos en Lisp).

Cuando Richard Stallman desarrolló Emacs, su teclado tenía una tecla llamada Meta, es por esto que en la notación de Emacs aparece una M haciendo referencia a la tecla Meta. Cómo nuestros teclados no tienen Meta, por defecto Alt la sustituye.

Nota. Para mac si no tienen Alt, ESC debería de funcionar.

Para **llamar un comando** solo hay que mantener presionado Alt seguido de x y en el **minibuffer** (la parte de abajo de la pantalla en la que aparecen los comandos que vamos tecleando) deberían de ver M-x, que es la abreviación de lo que acabamos de teclear en la notación de emacs.

Ahora pueden llamar el comando doctor y presionar Enter para abrir un programa que mediante procesamiento de lenguaje natural simula una conversación con un psicoterapeuta.

Los comandos de Emacs usualmente involucran el uso de las teclas Ctrl y Alt.

`M-`	mantener presionada la tecla Alt
`C-`	mantener presionada la tecla Ctrl
`C-a`	mantener presionado Ctrl mientras se presiona a
`C-a b`	se presiona Ctrl y a, se sueltan, y se presiona b
`C-a C-b`	se presiona Ctrl y a, se sueltan, y se presiona Ctrl y b

Nota. Para **abreviar los comandos** que usan Ctrl dos veces lo que pueden hacer es ejecutar el comando sin soltar Ctrl, por ejemplo: en lugar de teclear C-x C-f pueden teclear C-x-f.

2.3. Me equivoqué escribiendo un comando, ¿qué hago?

No entres en pánico.

Tanto C-g como ESC ESC ESC limpian el minibuffer (cancelan todas las teclas presionadas).

2.4. Ventanas

Puede parecer confuso, pero en emacs una *ventana* es parte de lo que se conoce como *frame* (marco).

El *frame* es la ventana que se abrió cuando iniciaron Emacs.

Un *frame* se puede dividir en ventanas.

`C-x 2`	abre una ventana horizontalmente
`C-x 3`	abre una ventana verticalmente
`C-x o`	cambia a la siguiente ventana
`C-x 0`	cierra la ventana actual
`C-x 1`	cierra todas las ventanas excepto la enfocada

Ejercicio 1. Divide el frame en 9, de modo que queden 3 filas y 3 columnas.

Nota. Se puede cambiar el tamaño de las ventanas:

`C-x ^`	hace la ventana más alta
`C-x }`	engorda la ventana
`C-x {`	achata la ventana

2.5. Buffers

El **texto** que editamos dentro de una ventana se encuentra contenido en un objeto llamado *buffer*. En emacs inicialmente se podría pensar que un buffer es un archivo, sin embargo, un buffer se **asocia** a más que solo archivos, por ejemplo: cuando se llama al comando animate, emacs carga una animación ASCII en un buffer especial. Este buffer no está asociado a un archivo en disco, sino que es un buffer que se crea dinámicamente (dependiendo de que nombre le pases al programa).

Abajo de cada ventana se despliega el nombre del *buffer*.

Un *buffer* es desplegado en un *frame*.

`C-x C-b`	lista los buffers
`C-x b`	cambia de buffer, pueden usar las flechas (arriba/abajo) para elegir uno
`C-x k`	mata el buffer enfocado

Nota. Los buffers de lectura se cierran con C-x 1 en caso de que el cursor no se encuentre activo en dicho buffer y con q en caso de que su cursor se encuentre activo en el buffer. El buffer que aparece al *iniciar emacs* es de lectura.

Ejercicio 2. Crea un nuevo buffer haciendo una llamada al comando tetris, cambia entre los buffers que despliegue la lista de buffers y mata los buffers `*tetris*` y `*doctor*`.

2.6. Cerrar emacs

El comando para cerrar Emacs es C-x C-c y siguiendo la idea de "abreviar los comandos", pueden simplemente hacer **C-x-c**. En caso de no haber guardado el contenido de algún buffer, el minibuffer lanza un mensaje para saber si deseas guardar los cambios.

2.7. Manejo de archivos

`C-x C-f`	abrir un archivo
`C-x C-s`	guardar un archivo
`C-x C-w`	guardar como
`C-x s`	guarda todos los archivos abiertos

Nota. Automáticamente cada que guardan un archivo se crea una copia en el mismo directorio, se les llama **archivos de respaldo**. Dichos archivos terminan con tilde.

2.7.1. Autocompletado

(TAB) les permite autocompletar texto en el minibuffer. Si por ejemplo, tienen un archivo con un nombre muy largo pueden autocompletar la busqueda de dicho archivo escribiendo las primeras letras de el nombre del archivo y después presionando (TAB).

3. Movimiento

3.1. Ejercicio

Dirigete a decuir.co/notes y descarga movimiento.txt

Entra al directorio de descargas ($ cd ./Download) y pasa el archivo al directorio $HOME

Abre el archivo directamente con emacs: $ emacs miarchivo.txt &

3.2. Lista de comandos usados en el ejercicio

3.2.1. Movimiento

`C-n`	línea de abajo
`C-p`	línea de arriba
`C-a`	inicio de línea
`C-e`	fin de la línea
`C-b`	siguiente carácter
`M-f`	siguiente palabra
`C-v`	scroll para arriba, el texto sube, el cursor baja
`M-v`	scroll para abajo, el texto baja, el cursor sube
`C-l`	mueve la posición del cursor junto con el texto

3.2.2. Búsqueda

`C-s <palabra>`	sirve para buscar una palabra hacia adelante
`C-r <palabra>`	sirve para buscar una palabra hacia atrás

3.2.3. Seleccionar una región

`C-SPC`	iniciar selección
`M-w`	copiar
`C-w`	cortar
`C-y`	pegar
`M-%`	reemplazar

3.2.4. Comentar

`M-x comment-region`	(autodescriptivo)
`M-x uncomment-region`	(autodescriptivo)
`M-;`	(comenta y descomenta)

3.2.5. Deshacer

Hay varias formas de deshacer una acción:

`C-/`	comando de undo
`C-_`	comando de undo
`C-x u`	comando de undo

3.2.6. Eliminar

`DEL | Backspace`	elimina un carácter hacia atrás
`C-d`	elimina un carácter hacia adelante
`M-DEL | M-Backspace`	elimina una palabra hacia atrás
`M-d`	elimina una palabra hacia adelante

4. Compilación

`M-x compile`	Ejecuta un comando y despliega cualquier error

5. Ejecución

Para ejecutar programas es necesario abrir una terminal y ejecutar el programa directamente.

`M-x eshell`	Abre una terminal nativa de emacs
`M-x shell`	Abre nuestra terminal

Ejercicio 3. Dirigete a decuir.co/notes y copia el código del .sh, pégalo en un archivo desde emacs, guárdalo y ejecutalo en un eshell mediante bash miarchivo.sh

6. Cómo hacerle preguntas a emacs

De acuerdo con un libro que chequé, los gurús de emacs saben resolver sus dudas de emacs desde emacs. Esto gracias el sistema de ayuda que proporciona el editor.

`C-h C-h`	abre un buffer de ayuda
`C-v & M-v`	para moverse en el buffer de ayuda
`C-h a <cadena>`	muestra comandos que coincidan con una cadena
`C-h k <key-sequence>`	describe la función que ejecuta una combinación de teclas
`C-h f <function-name>`	describe lo que hace una función

7. Repaso

Hice un archivo que les puede servir para reforzar algunas cosas que vimos. Lo pueden descargar en decuir.co/notes, se llama repaso.txt

8. Instalación

8.1. Linux

En cualquier distribución de linux hay paquetes de emacs, por lo que basta usar su manejador de paquetes favorito para instalarlo:

- *sudo dnf install emacs* (fedora)
- *sudo apt install emacs* (debian)
- *sudo pamac install emacs* (manjaro)

8.2. Otros sistemas operativos

Si cuentan con otro sistema operativo (que no debería de ser así):

- Mac OS S: http://emacsformacosx.com/ o con homebrew: *brew install emacs*
- Windows: no tengo idea

9. Configuración mínima de emacs (autor: Emiliano Galeana)

Para poder confgiurar Emacs, es necesario editar su archivo de configuración, en el momento en que instalas emacs, el archivo de configuración no existe (o se encuentra vacío) y no hace nada (en caso de existir) hasta que decides modificarlo.

La forma más fácil de comenzar a configurarlo es creando un directorio ∼/.emacs.d/ con un archivo ∼/.emacs.d/init.el (verifica si ya existe antes de crear el archivo, en cuyo caso basta con comenzar a editarlo). Escribe ahí tu configuración y la siguiente vez que abras emacs se verá reflejada.

9.1. Configuraciones iniciales

Estas configuraciones solamente editan la visualización de emacs como editor de textos, no agregan paquetes.

9.1.1. Barras superiores

```
;; Quitamos la barra de File, Edit, Options, etc
(menu-bar-mode 0)
;; Quitamos la barra de abrir, guardar, undo, copiar, pegar
(tool-bar-mode 0)
```

9.1.2. Líneas a la izquierda

Emacs (malamente) por defecto no trae las famosas líneas a la izquierda para çontar.[en] qué linea estamos, esto puede ser frustrante al inicio.

```
;; Añadimos las líneas a la izquierda
(global-linum-mode 1)
```

9.1.3. Paréntesis

Otra cosa que emacs no trae por defecto es el autocompletado de paréntesis, lo cuál es de mucha ayuda a la hora de programar, esto igual autocompleta comillas simples y dobles, corchetes y llaves.

```
;; autocompletar paréntesis
(electric-pair-mode 1)
```

9.1.4. Configuración del teclado

Uno de los cambios más útiles en la configuración del teclado de emacs es hacer que la tecla "Ctrl"se comporte como la tecla Çaps Lockz viceversa:

```
(setq ctl-x-is-uppercase nil)
```

9.2. Archivos de respaldo

Para guardar los archivos de respaldo en .emacs.d

```
(custom-set-variables
  '(backup-directory-alist '((".*" . "~/.emacs.d/backups/"))))
```

9.3. Paquetes

Por defecto emacs no es tan personalizable como nos gustaría, hay que instalar paquetes de tercero, la mayoría de los paquetes se encuentran en MELPA (https://melpa.org). Aquí agregamos MELPA para poder descargar paquetes y además nos aseguramos de que los paquetes estén disponibles siempre que iniciemos emacs:

```
;; De donde vamos a descargar paquetes.
(require 'package)
(add-to-list 'package-archives
             '("melpa" . "https://melpa.org/packages/"))
(package-initialize)
(package-refresh-contents)
(unless package-archive-contents
```

```
  (package-refresh-contents))
(unless (package-installed-p 'use-package)
  (package-install 'use-package))
(require 'use-package)
(setq use-package-always-ensure t)
```

A partir de aquí necesitan haber agregado MELPA (para poder descargar paquetes):

9.3.1. Haskell

Haskell es un gran lenguaje de programación que pudiera parecer que solo se usa en la academia, como sea, por defecto emacs no tiene soporte para este lenguaje (Como para muchos otros), pero en primer semestre resulta un poco frustrante que en ICC usando java emacs /çoloree/ las palabras reservadas y en Estructuras Discretas no tengamos eso para nuestro laboratorio.

```
;; Haskell-mode
(use-package haskell-mode)
```

Se necesita la instalación de un compilador como GHC (https://www.haskell.org/ghc/) para poder trabajar con haskell.

9.3.2. Tema

Por defecto emacs está en blanco y aunque hay temas que vienen "predeterminados"hay una gran variedad en internet.

Nosotros recomendamos dracula-theme (https://github.com/dracula/emacs).

```
;; un tema oscuro (muy famoso), si quiren uno blanco borren esto
(use-package dracula-theme
  :ensure t
  :init
  (setq dracula-theme t)
  (load-theme 'dracula t))
```

9.3.3. Paréntesis

Un paquete que pareciera no tan interesante, pero nos ayuda a identificar qué llaves, paréntesis, corchetes cierran a cuál.

```
;; Colorea qué paréntesis cierra con cuál
(use-package rainbow-delimiters
  :hook (prog-mode . rainbow-delimiters-mode))
```

9.3.4. LaTeX

Durante toda la carrera se va a usar LaTeX, así que igual que haskell, queremos que nos resalte palabras reservadas, además para cuando usamos el modo matemático agregamos que cada que escribimos = $ = se autocomplete con otro.

```
;; Le metemos autocerrar pesitos a latex
(use-package latex-mode
  :ensure nil
  :defer t
  :hook (latex-mode . (lambda () setq electric-pair-pairs
                        '(?$ . ?$))))
```

9.3.5. Keys

Este paquete es útil para quien está empezando con emacs (y para quien tiene mala memoria) pues nos ayuda con recomendaciones de comandos al presionar ciertas teclas, por ejemplo si se quiere cambiar de ventana (=C-x o=) y no recordamos el comando, basta con "teclear-C-x= y se creará un buffer temporal con las posibles teclas que podemos escribir, cada una con el comando que invoca.

```
;; Ayuda para recomendar combinaciones de teclas que nos pueden servir.
(use-package which-key
  :config (which-key-mode))
```

10. Algunos sitios que pueden ser de ayuda

10.1. Configuracioń del init.el

https://github.com/pierre-lecocq/emacs4developers/

10.2. Artículos para ir explorando Emacs

https://www.masteringemacs.org/ (vienen al final de la página)

10.3. Jueguitos en emacs (para entretenerse)

https://www.emacswiki.org/emacs/CategoryGames

11. Bibliografía

Bibliografía consultada para la realización de estas notas:
Manual de supervivencia
Un repo de github muy ordenado
El cheat sheet
El libro de Mickey Petersen (lo pueden descargar en *libgen*)