

WhyFLOSS Conference

Madrid, Julio 2007

Organiza: **neurowork™**

Colaboran:  **CINDETEC** **UNED** 

Media sponsors: **LINUX+** **LINUX** 

Todo LINUX **MUNDO LINUX**

Desarrollo con GNU/Emacs

David Arroyo, UNED

<http://www.whyfloss.com/es/conference/madrid07/>





J/Emacs

Desarrollo con GNU/Emacs **Desarrollo basado en comunidad**

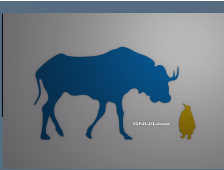
David Arroyo Menéndez

UNED - Innova (<http://www.innova.uned.es>)



UNED





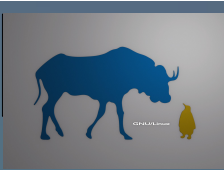
Emacs: Historia

1956: John McCarthy invita a 10 personas a Darmouth durante 2 meses a estudiar una materia que llama Inteligencia Artificial

1960: John McCarthy escribe "Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I". Poco después Russell implementa esas ideas. Ese lenguaje se llama Lisp.

1975-1978: "Lambda the Ultimate Papers" son una serie de artículos acerca de Lisp, dónde se escriben cosas como la especificación de un nuevo dialecto Lisp: Scheme. Son escritos por Gerald Jay Sussman y Guy Steele Jr.



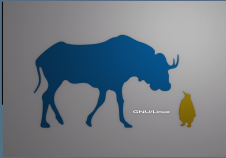


1976: EMACS es escrito por Richard Stallman, Guy Steele and Dave Moon. Si bien Stallman es quien hace casi todo el trabajo.

1977: Stallman publica "AI truth maintenance system called dependency-directed backtracking". El artículo era coescrito con Gerald Jay Sussman.

1978: Multics Emacs por Bernie Greenberg. Escrito in MacLisp y también usa Lisp como su lenguaje de extensión.

1980 Richard Greenblatt, un hacker del AI Lab, fundó Lisp Machines, Inc. (LMI) para vender máquinas Lisp, que él y Tom Knight diseñaron en el laboratorio.



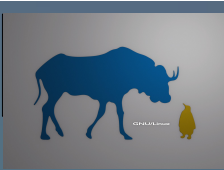
1991: Lucid comienza a hacer una serie de mejoras a GNU Emacs que no son devueltas a la comunidad dando lugar a XEmacs.

1994: Guy Steele entra en Sun Microsystems

1996: James Gosling, Guy Steele y Bill Joy escriben "The Java Language Specification"

2007: Java es liberado bajo la licencia GPL tras una carta de James Gosling

2007: GNU/Emacs ya va por su versión 22



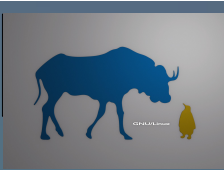
I think it would be a tragic statement of the universe if Java was the last language that swept through. -James Gosling

Changing the programming language is like changing the underlying genetic structure of an organism. It's about as deep in the core of the craft of programming as you get, so it really is difficult to change it. -James Gosling

"One can even conjecture that Lisp owes its survival specifically to the fact that its programs are lists, which everyone, including me, has regarded as a disadvantage." - John McCarthy, "Early History of Lisp"

"Programs must be written for people to read, and only incidentally for machines to execute." - Abelson & Sussman, SICP, preface to the first edition



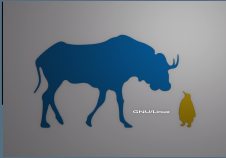


"We were not out to win over the Lisp programmers; we were after the C++ programmers. We managed to drag a lot of them about halfway to Lisp."
- Guy Steele, Java spec co-author

"People sometimes ask me if it is a sin in the Church of Emacs to use vi. Using a free version of vi is not a sin; it is a penance. So happy hacking."
- Richard Stallman

If you give someone Fortran, he has Fortran. If you give someone Lisp, he has any language he pleases.
-- Guy Steele

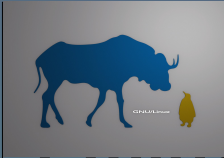




Lisp is worth learning for the profound enlightenment experience you will have when you finally get it; that experience will make you a better programmer for the rest of your days, even if you never actually use Lisp itself a lot. Eric Raymond, "How to Become a Hacker"

I have a lot of stuff in Emacs, including the VC mode that front-ends for RCS/SCCS/CVS and the Grand Unified Debugger mode that lets you drive GNU gdb and other symbolic debuggers from within Emacs. According to RMS's credit list, I appear to have more Emacs Lisp code in the standard Emacs distribution than anyone else but him. Eric Raymond, (<http://www.catb.org/~esr/software.html>)





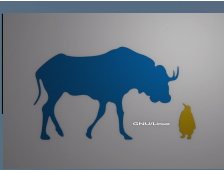
Emacs: Un editor que se aprende en auto

Échale un vistazo!! ;-)

Conceptos:

- * Frame, Buffer, Window
- * Puntero y Marca
- * M-x





Algunas configuraciones simples

```
:: quito la barra de navegacion
```

```
(tool-bar-mode nil)
```

```
:: un atajo de teclado para goto-line
```

```
(global-set-key (kbd "C-v") 'goto-line)
```

```
:: colorines
```

```
(global-font-lock-mode)
```

```
:: metemos auto-fill en modo text
```

```
(setq text-mode-hook (cons 'auto-fill-mode text-mode-hook))
```



necesidades al

/ 2 atajos de teclado

- 1) Conoces el atajo de teclado navegando por el menú wysiwyg
- 2) Llegas a la función asociada al atajo de teclado con C-h k
- 3) Dado el atajo de teclado, llegas al código fuente de la función con C-h f
- 4) Ya puedes modificar el código fuente





n editor que lo

para ser productivo

código hay 2 atajos de teclado

El primer paso para aumentar la productividad utilizando un entorno es

aprender un atajo de teclado.

* C-h m: atajos del modo en el que estás

* C-h b: todos los atajos disponibles en ese momento

El siguiente paso para aumentar la productividad es que si te encuentras repitiendo las mismas acciones varias veces, las puedas grabar y ejecutarlas.

* C-x (: comienza la grabación

* insertas texto, ejecutas atajos, etc.

* C-x): fin de la grabación

* C-x e: ejecutas lo grabado



n editor que lo

para ser productivo

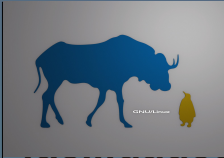
código hay 2 atajos de teclado

Otro paso aún mejor es escribir funciones para aquellas cosas que realizas con frecuencia. Un ejemplo:

```
(defun cp-dev ()  
  "Función para copiar al servidor de desarrollo"  
  (interactive)  
  (message "Copio un fichero local a dev")  
  (setq path  
    (progn (string-match "/home/username/cvs/rama-3.1/dotalf" (expand-  
file-name default-directory))  
          (replace-match "/web/dev" nil nil (expand-file-name default-  
directory))))  
  (start-process  
    "copying..." "*copying*" "scp" buffer-file-name (concat  
"username@dev.innova.uned.es:" path)))
```

Y luego puedo asociarle un atajo de teclado

```
(global-set-key "\C-c\C-d" 'cp-dev)
```

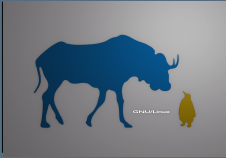


n editor con s para tu lenguaje

- * AdaMode
- * AnsysMode
- * AntlrMode
- * AutoconfMode - for autoconf configure scripts
- * AutoLispMode
- * AxxcessMode - also Netlinx
- * BisonMode
- * CamlMode - ObjectiveCaml
- * CPerlMode - improved PerlMode
- * CcMode - C, C++ (cpp), Objective C, Java, etc.
- * CgMode - Cg, nvidia language for graphics cards
- * ChillMode - CHILL (CCITT High Level Language) Mode for XEmacs
- * ChillModePackage - CHILL (CCITT High Level Language) Mode for Emacs
- * CheetahMode - Cheetah Templates Mode.
- * CobolMode
- * CSharpMode - C#
- * DMode - D language editing mode.
- * D4Mode - Alphora Dataphor's D4 relational data language.
- * DDLMode - Data Definition Language Mode
- * DelphiMode - for Borland Delphi's Object Pascal syntax
- * DevelockMode - highlight faulty formatted source
- * DoxyMacs - Doxygen
- * ECMAScriptMode - similar to JavaScriptMode
- * EIDoc - documentation for EmacsLisp
- * ElseMode - minor mode with code templates for various languages
- * EmacsCodeBrowser (ECB)
- * EmacsLispMode
- * EnformMode - Major mode for Tandem Enform files.
- * FlexMode

[...]



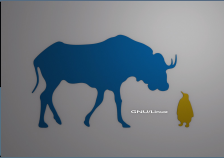


filosofía de extensibilidad de teclado para el escritorio

Algunos de los modos que uso que no son para programar:

bbdb	(unknown)
calendar	(unknown)
emacs-wiki	(unknown)
erc	5.0.2
emms	(unknown)
mailcrypt	3.5.8
mmm-mode	0.4.8
newsticker	(unknown)
remember	(unknown)
planner	(unknown)
tramp	2.0.39
w3m	(unknown)





: quiero más!

<http://www.gnu.org/software/emacs/>

<http://www.emacswiki.org/cgi-bin/wiki>

<http://savannah.gnu.org/projects/emacs>

<http://www.gnu.org/software/emacs/emacs-paper.html>

<http://www.gnu.org/gnu/rms-lisp.html>

<http://emacs.es.gnu.org>



