TRABAJO PRÁCTICO

CASO DE ESTUDIO

GNU EMACS

[75.31] TEORÍA DE LENGUAJE

PRIMER CUATRIMESTRE DE 2020

CAMILA DVORKIN 101109

CDVORKIN@FI.UBA.AR

ÍNDICE

1 DESCRIPCIÓN

Emacs es un editor de texto con una gran cantidad de funciones, muy popular entre programadores y usuarios técnicos. GNU Emacs es parte del proyecto GNU y la versión más popular de Emacs con una gran actividad en su desarrollo. GNU Emacs es un editor extensible, personalizable, auto-documentado y de tiempo real. Se puede usar como un entorno de desarrollo integrado (IDE), que permite a los programadores editar, compilar y depurar su código con una única interfaz.

2 HISTORIA

Emacs nace en los laboratorios del MIT durante los años 70. El inicio original de Emacs fue como un conjunto de macros y combinaciones de teclas para el editor de texto TECO, un editor que realizaba de manera separada la introducción de texto, edición y la vista del mismo (lo que escribias, no aparecía adentro del documento, sino que se debía ejecutar diferentes instrucciones que le indiquen que escribir).

A mediados de 1970, Richard Stallman, agrega una nueva funcionalidad en donde permitía un modo de edición en tiempo real(novedad completa para la época) que atualizaba la visualización del texto que se está editando. Además agregó una característica al modo edición-muestreo de TECO, para que el usuario pueda redefinir cualquier atajo de teclado para ejecutar un programa TECO.

Por otro lado, Guy Steele coordinó entre los usuarios, creando un conjunto común de combinaciones de teclas y macros, que Richard Stallman desarrolló en la primera versión de Emacs. El diseño inicial de Emacs Lisp fue motivado por el requisito generalizado de extensibilidad: el usuario debe poder redefinir cada caracter. En consecuencia, Richard Stallman, inspirado en MacLisp y Common Lisp, crea Emacs Lisp, lenguaje de programación con todas las funciones con potentes funciones de abtracción.

En 1984, Stallman empezó a trabajar en GNU Emacs para producir una alternativa de software libre al Gosling Emacs. Inicialmente se basó en el Gosling Emacs, pero Stallman reemplazó el intérprete de Mocklisp con un intérprete de Lisp, lo que le obligó a sustituir casi todo el código. GNU Emacs se convirtió en el primer programa publicado por el emergente Proyecto GNU. GNU Emacs está escrito en Emacs Lisp (a su vez implementado en C) como lenguaje de extensión.

3 FEATURES

- **Altamente personalizable, usando código Emacs Lisp o una interfaz gráfica.**
- **Un sistema de empaquetado para descargar e instalar extensiones.**
- Modos de edición con reconocimiento de contenido, incluido el resaltado de sintaxis, para muchos tipos de archivos.
- Documentación incorporada completa, incluido un tutorial para nuevos usuarios. Soporte completo de Unicode para casi todos los scripts humanos.
- Una amplia gama de funcionalidades más allá de la edición de texto, incluyendo un planificador de proyectos, lector de correo y noticias, interfaz de depuración, calendario, cliente IRC, y más.

4 IMPLEMENTACIÓN

GNU Emacs está implementado como un intérprete de Emacs Lisp escrito en C, al cual se ha extendido con funciones en Lisp para editar texto; por lo que **casi todo el código de Emacs se puede modificar o extender en tiempo real**. El probelma del diseño de Emacs, es el rendimiento del editor resultante de cargar e interpretar el código de Lisp. Pero la realidad es que hoy en dia en computadoras modernas raramente se nota que es más lento que otros editores, incluso se inicia más rapidamente que la mayoría de procesadores de texto modernos.

5 FUNCIONALIDADES PERSONALIZABLES

Lo interesante de Emacs es la cantidad de caracterisiticas que brinda. Gracias a su diseño, casi toda la funcionalidad del editor, desde las operaciones básicas de edición (como la inserción de caracteres en un documento) hasta la configuración de la interfaz de usuario, es controlada por un dialecto del lenguaje de programación Lisp.

Emacs va a adaptar su comportamiento al tipo de texto que se esta editando mediante los *mayor modes*. Se van a modificar ciertas variables en Lisp para que Emacs se comporte de la forma más conveniente para ese tipo de texto.

Por otro lado, el comportamiento de Emacs puede mer más personalizado utilizando *minor modes*. En el entorno de Lisp, variables e incluso funciones enteras pueden ser modificadas, sin tener que recompilar o ni siquiera reiniciar el editor. Como resultado, el comportamiento de Emacs puede ser modificado casi sin límite, directamente por el usuario, o cargando fragmentos de código Emacs Lisp(librerías, paquetes, extensiones), con el objetivo de permitirle al usuario personalizar el editor para que se adapte a sus necesidades.

La aplicacion puede ser alterada por el usuario mientras aún se esta ejecuntando. Por lo tanto, no va a ser necesatio descargar el código fuente o compilar después de cada cambio. Basta con crear tu propio entorno con las modificaciones que uno desee. Por ende uno puede aprovechar el trabajo que otros han creado, se puede examinar el código y cambiarlo si no es concreto lo que se quiere. Incluso hay una gran cantidad de bibliotecas que fueron distribuidas a los usuarios a partir de código que fue personalizado por diferentes usuarios.

Algunos ejemplos de modos para que Emacs se comporte de la forma diferente a la predeterminada y logre trabajar de la manera que uno necesita: *Org mode* para organizar tus horarios y administrarte el calendario, *Deft mode* para tomar notas, *Fountain mode* para escribir guiones, *Muse* para escribir artículos y artículos, *SES* para hojas de cálculo, *Python mode* para código Python, *Sh mode* (shell script) para edición Bash , *nXML mode* para la edición de XML y muchos (*muchos!*) más.

En resumen, gracias a esta facilidad de extensión debido a su implementación, todo usuario va a poder hacer lo que quiera en Emacs, incluso probablemente ya exista un modo o un paquete que te lo facilitará.

6 ALGUNOS COMANDOS

A continuación, a modo de ejemplo, se enumeran algunos comandos sencillos que se pueden usar:

Comando	Tecla	Descripción
forward-word	M-f	Avanzar una palabra
search-word	C-s	Buscar una palabra en el buffer
undo	C-/	Deshacer el último cambio
keyboard-quit	C-g	Abortar el comando actual
find-file	C-x C-f	Visitar un fichero en su propio buffer de edición
save-buffer	C-x C-s	Guardar el buffer de edición actual en su fichero visitado
save-with-newname	C-x C-w	Guardar el buffer de edición actual con el nombre elegido
save-buffers-kill-emacs	C-x C-c	Preguntar al usuario si se quieren guardar los cambios
set-marker	C-[space]/C-@	Colocar un marcador en donde se quiere cortar o copiar
cut	C-w	Cortar todo el texto entre el marcador y el cursor
copy	M-w	Copiar todo el texto entre el marcador y el cursor
paste	C-y	Pegar texto del portapapeles de Emacs
kill buffer	C-x k	Cerrar el buffer actual

REFERENCIAS

GNU Emacs - www.gnu.org
https://www.gnu.org/software/emacs/
Introduction to Emacs Lisp - Stephan Monnier and Michael Sperber
https://www.iro.umontreal.ca/ monnier/hopl-4-emacs-lisp.pdf
Emacs - What is Emacs?
https://opensource.com/resources/what-emacs
6 things you should be doing with Emac
https://opensource.com/article/20/1/emacs-cheat-sheet
CommonLisp - AaronHawley
https://www.emacswiki.org/emacs/CommonLisp