There are no translations available.
Aprende a manejarte con el Vi, un editor de texto que fue creado para sistemas UNIX y que posteriormente se implantó en sistemas GNU/Linux.

EDITOR VI

1.- Introducción

Vi es un editor de texto que fue creado para sistemas UNIX y que posteriormente se implantó en sistem

Vi fue escrito originalmente en la Universidad de California en Berkeley por William Joy en 1976, con la

Es posiblemente el único editor que se encuentra en la mayoría de sistemas UNIX y GNU/Linux.

Aunque Vi no es muy fácil de usar ni muy autoexplicativo, es tan común en Linux y Unix que muchas ve

Es importante saber que también existe una versión mejorada de Vi (VIM VI Improved).

Además, se ha de señalar que en gran parte de las distribuciones GNU/Linux Vi no es más que un enla

2.- ¿Tenemos instalado Vi?

Normalmente tendremos instalado este editor en nuestra distribución GNU/Linux, sea cual sea. De toda

alex@alexDebian :~$vi

~

~ VIM - VI Mejorado

~ versión 6.3.82

~	por

~ Vim es código abierto y se puede distribuir libremente

~ ¡Patrocine el desarrollo de Vim!

~ escriba :help sponsor<Intro> para más información

~ escriba «:q<Intro>» para salir

~ escriba «:help<Intro> o <F1>» para obtener ayuda

~ escriba «:help version6<Intro>» para información de la versión

o, también con

alex@alexDebian :~$which vi

/usr/bin/vi

Que nos dice donde está ubicado el ejecutable de Vi (por convenio, todo el software adicional

del sistema se almacena en /usr/, y en bin/ sus archivos ejecutables)

En el hipotético caso de no tener instalado Vi, lo instalaremos en nuestro sistema de una forma

u otra dependiendo del gestor de paquetes con el que trabaja.

Para sistemas con gestor de paquetes tipo APT (sistemas Debian)

* Instalación:

Como root ejecutamos:

root@alexDebian :#apt-get install vim

Para sistemas con gestor de paquetes tipo RPM (sistemas Red-Hat, Fedora, Suse, OpenSuse, Mandriva, etc...)

- RPM (sistemas Red-Hat)

Primero, conseguimos el paquete RPM adecuado para nuestra distribución (hay que tener en cuenta la versión del sistema operativo), sabiendo que la tipología de un paquete RPM es:

<name>-<version>-<release>.<arch>.rpm

* ¿Donde conseguirlo?:

http://rpmfind.net/ .

http://rpm.pbone.net/

http:// www.rpmhelp.net/

* Instalación:

Como root ejecutamos:

root@alexRedHat :#rpm -i vi-version-arquitectuta.rpm

- Yum (Yellow Dog Updater, Modifier para Fedora Core). Yum es un gestor de paquetes

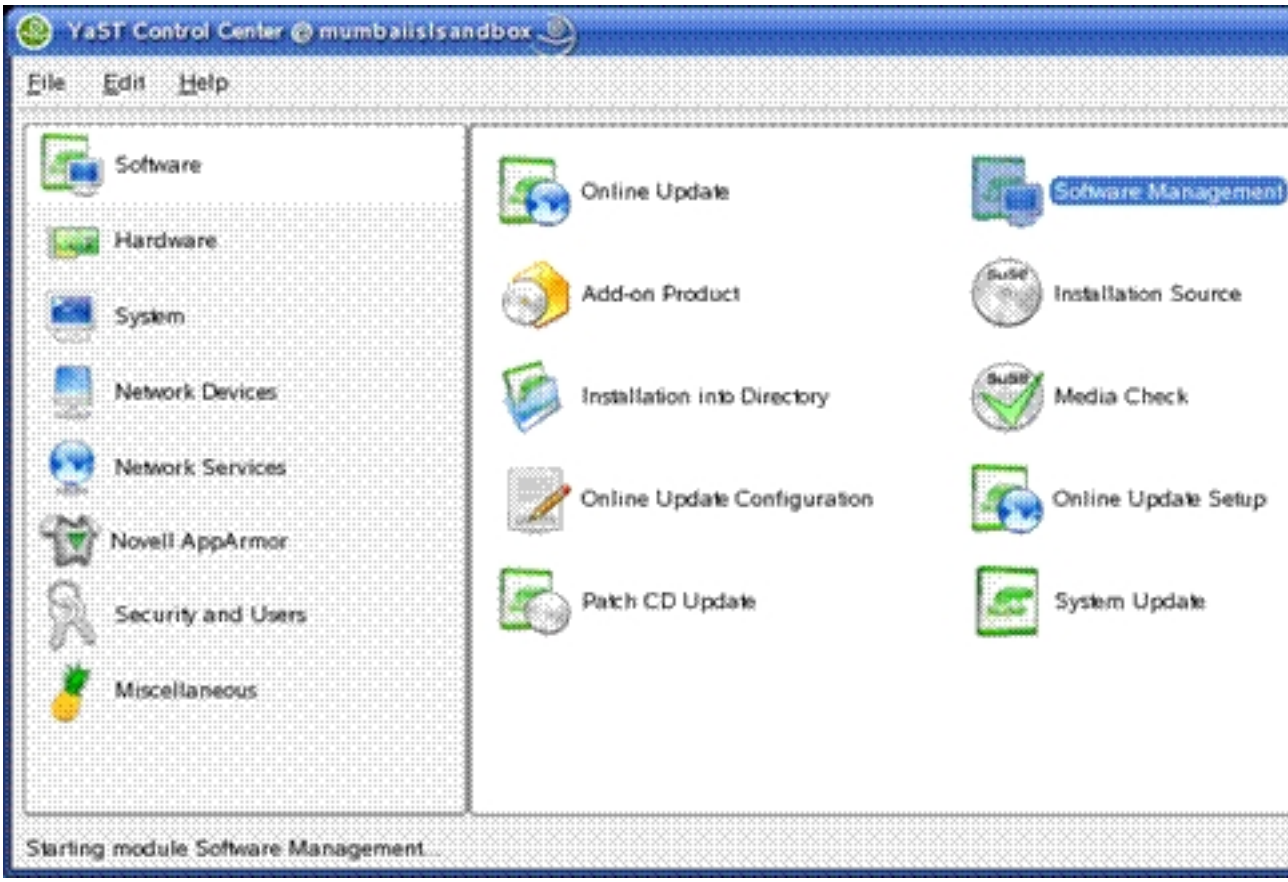
que deriva de RPM, pero que le hace más potente ya que utiliza repositorios (como Apt-get) y se encarga de las dependencias.

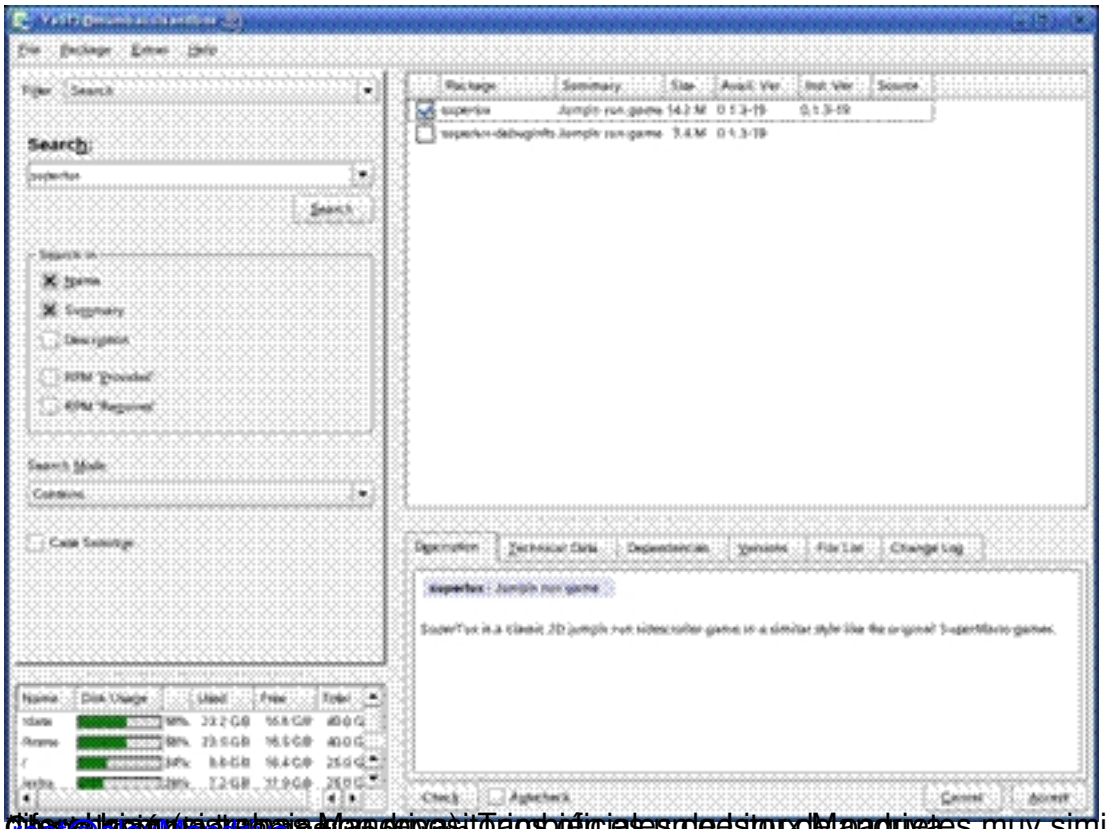
* Instalación:

Como root ejecutamos:

root@alexFedora :#yum install vim

- Yast (sistemas Suse) Yast es una herramienta gráfica de instalación y configuración para sistemas SuSE





[illegible] es muy similar a Yum, con la

Instalación por código fuente

Escogeremos esta opción si queremos instalar Vi o Vim por medio de sus fuentes.

Nos bajamos de la página oficial (http://www.vim.org/) el fichero fuente, por ejemplo

vim-7.0.tar.bz2. Lo vemos:

Nota	: el fichero fuente

tar -xvzf vim-7.0.tar.gz

```
alex@alexDebian :~$wget ftp://ftp.vim.org/pub/vim/unix/vim-7.0.tar.bz2

--19:43:07-- ftp://ftp.vim.org/pub/vim/unix/vim-7.0.tar.bz2

=> `vim-7.0.tar.bz2'

Resolviendo ftp.vim.org... 192.87.102.36

Conectando con

Identificándose como

==>

==> TYPE I ... hecho.

==>

Longitud: 6,575,205 (probablemente)

100%[==========================================================================
```

19:44:00

Lo descomprimimos:

alex@alexDebian :~$bunzip vim-7.0.tar.bz2

alex@alexDebian :~$tar -xvf vim-7.0.tar

vim70

vim70/README.txt

vim70/runtime

vim70/runtime/bugreport.vim

vim70/runtime/doc

vim70/runtime/doc/makehtml.awk

.....

Hacemos la precompilación:

```
alex@alexDebian :~$cd vim70

alex@alexDebian :~/vim$ ./configure

configure: creating cache auto/config.cache

checking whether make sets $(MAKE)... yes

checking for gcc... gcc

checking for C compiler default output file name... a.out

checking whether the C compiler works... yes

checking whether we are cross compiling... no

checking for suffix of execu

.....
```

Lo compilamos:

```
alex@alexDebian :~/vim70$ make

Starting make in the src directory.

If there are problems, cd to the src directory and run make there

cd src && make first

make[1]: Entering directory `/download/vim70/src'

mkdir objects

CC="gcc -Iproto -DHAVE_CONFIG_H " srcdir=. sh ./osdef.sh

gcc -c - I. -Iproto -DHAVE_CONFIG_H -g -O2 -o objects/buffer.o buffer.c

gcc -c - I. -Iproto -DHAVE_CONFIG_H -g -O2 -o objects/charset.o charset.c

gcc -c - I. -Iproto -DHAVE_CONFIG_H -g -O2 -o objects/diff.o diff.c

.....
```

Y por último, como root lo instalamos:

root@alexDebian :~/vim70#make install

```
Starting make in the src directory.

If there are problems, cd to the src directory and run make there

cd src && make install

make[1]: Entering directory `/download/vim70/src'

if test -f /usr/local/bin/vim; then

mv -f /usr/local/bin/vim /usr/local/bin/vim.rm;

rm -f /usr/local/bin/vim.rm;

fi

cp vim /usr/local/bin

strip /usr/local/bin/vim
```

```
chmod 755 /usr/local/bin/vim
```

```
/bin/sh ./mkinstalldirs /usr/local/share/vim
```

```
.....
```

Y ya tendríamos instalado VIM en nuestra máquina.

NOTA	: El directorio por

NOTA	: la instalación se

3.- Modos de vi

Vi tiene tres modos de utilización :

- **El modo normal**. En este modo Vi interpretará todo lo que escribamos para realizar acciones determinadas. Es el modo predeterminado de Vi cuando arranca.

- **El modo inserción.** Este modo se utiliza para modificar el contenido de un archivo, es decir, escribir texto, eliminarlo o desplazarnos por él. Es el modo que más se utiliza. Para volver al modo normal desde este modo, presionaremos siempre la tecla ESC.

- **El modo "ultima línea" o "línea de comandos".** Modo para dar órdenes extendidas a Vi. Mientras se escriben las órdenes, éstas aparecen en la última línea de la pantalla, de ahí su nombre. Entraremos en este modo pulsado la tecla ":" y seguidamente escribimos las ordenes.

4.- Teclas para cambiar entre modos

ESC

Activa el modo normal. También sirve para interrumpir una acción que se está ejecutando

i

Activa el modo inserción

:

Activa el modo ultima línea

5.- Objetos de texto

Vi reconoce objetos de texto estándar: palabras, frases, párrafos y bloques.

Es importante saber que para cada objeto se usa una letra identificativa del mismo:

w

Para	palabras

s

Para	frases

P

Para	párrafos

b

Para texto encerrado entre paréntesis ()

B

Para texto encerrado entre corchetes []

6.- Sintaxis de Vi

vi [opciones] [fichero(s)]

- Opciones: aquí pondremos las opciones admitidas por Vi. Veremos todas en su página man (ejecutamos *man Vi*)

- Fichero: Si el archivo tecleado existe, se mostrará, sino crearemos un archivo en blanco con el nombre especificado.

Podemos especificar más de un fichero separados por espacios en blanco

Ejemplo. En este caso el fichero *pedro* no existe

```
alex@alexDebian :~/vi pedro
```

```
1|
```

```
~
```

```
~
```

```
"pedro" [Fichero nuevo] 0,0-1 Todo
```

El símbolo □ |□ le indica donde está el cursor.

7.- Trabajar con Vi

Inserción de texto.

Estando en el modo normal, presionamos:

a

Inserta después del carácter donde estamos situados

i

Inserta antes del carácter donde estamos situados

A

Añade al final de la línea actual

I

Añade al principio de la línea actual

R

Reemplazar texto

o

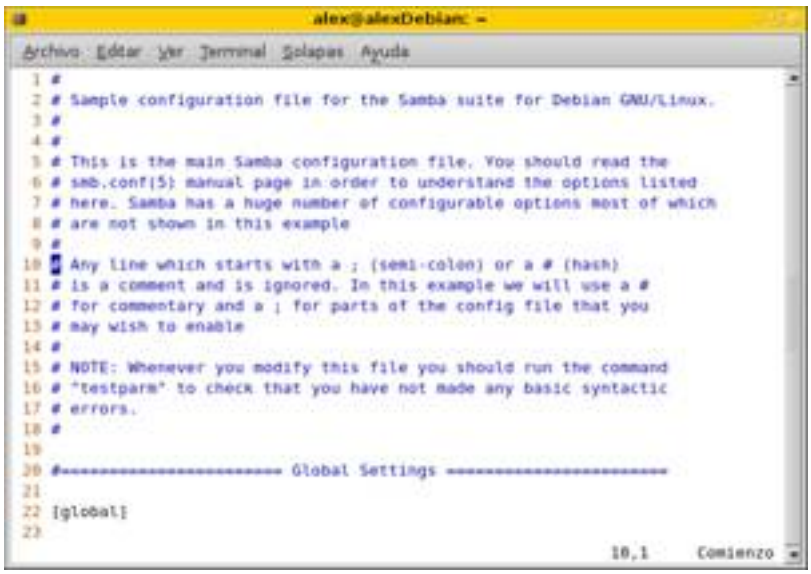
Añade una línea en blanco debajo de la línea donde estemos posicionados y pasa al modo inserción

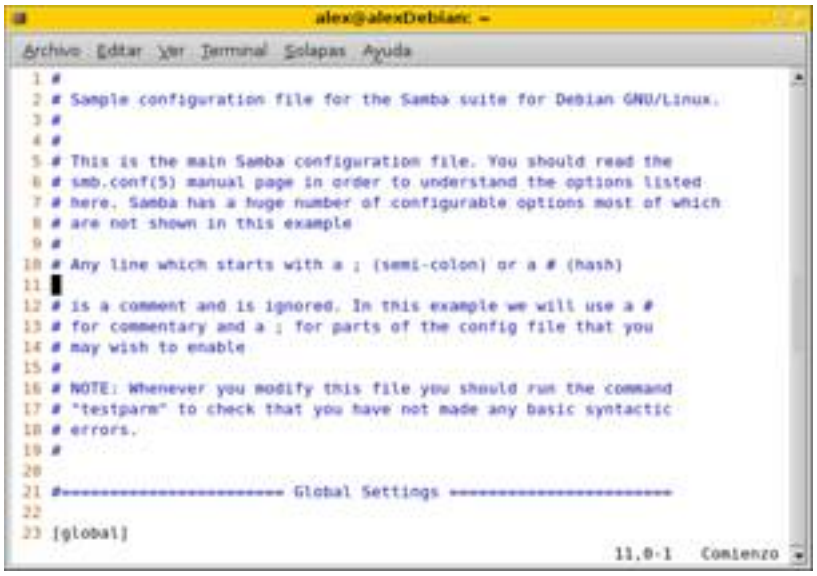
O

Añade una línea en blanco encima de la línea donde estemos posicionados y pasa al modo inserción

Para finalizar la inserción, pulsamos la tecla ESC (retorna al modo normal)

Ejemplo, insertamos una nueva línea en blanco debajo de la posición del cursor pulsando **o**:





Borrado de texto

Estando en el modo normal, presionamos:

x

Borra el carácter que está debajo del cursor

nº x

Borra el carácter que está debajo del cursor y todos los demás n-1 caracteres situados a la derecha del

D

Borra desde la posición actual al final de la línea

dd

Borra la línea actual

nº dd

Borra n líneas

Dw

Borra desde la posición actual al final de la palabra

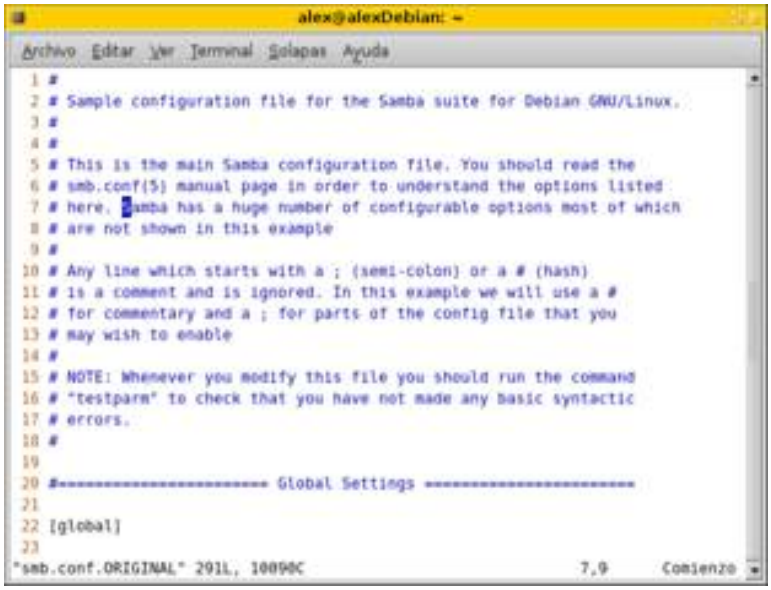
dnº$

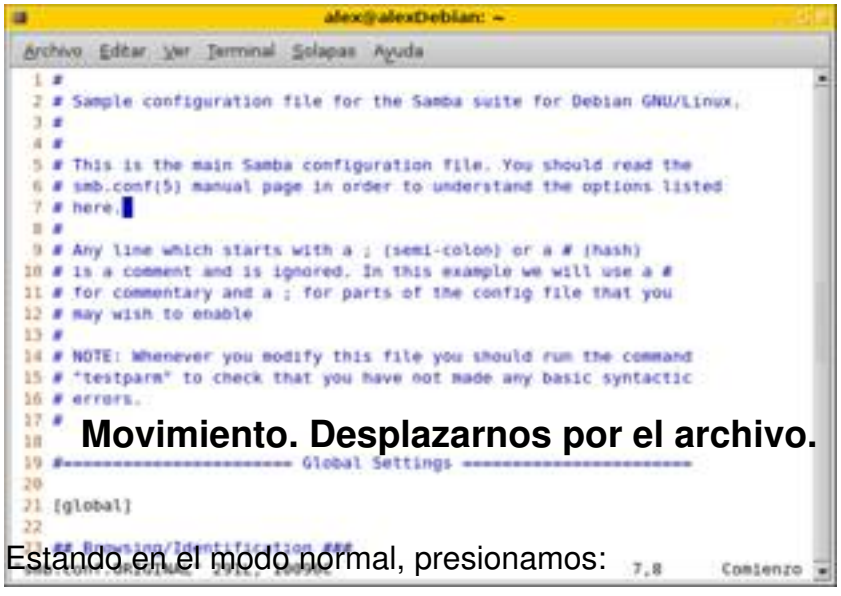
Borra desde la posición actual al final de la palabra y todas las n líneas que están debajo

dG

Borra desde la posición actual hasta el final del fichero

Ejemplo, borramos desde la posición actual del cursor hasta el final del párrafo, es decir, borramos la última frase pulsando la secuencia de teclas ***d2$***. Lo vemos:





Estando en el modo normal, presionamos:

h

Hacia la izquierda

j

Hacia abajo

k

Hacia arriba

l

Hacia la derecha

Esto se debe a que cuando nació vi, las teclas de flechas no existían. Lógicamente, utilizaremos ésta

w

Nos movemos al principio de la siguiente palabra

B

Nos movemos al principio de la palabra anterior

0

Nos movemos al principio de la línea actual

$

Nos movemos al final de la línea

E

Nos movemos al final de la siguiente palabra

E

Nos movemos al final de la siguiente palabra antes de un espacio

vi +10	***mi_fichero***

Abrimos	*mi_fichero*

[[

Nos movemos al inicio del archivo

]]

Nos movemos al final del archivo, también pulsando la tecla G

nºG

Nos movemos a la línea n

Ctrl+F

Pantalla siguiente (Tecla Av Pag)

Ctrl+B

Pantalla anterior (Tecla Re Pag)

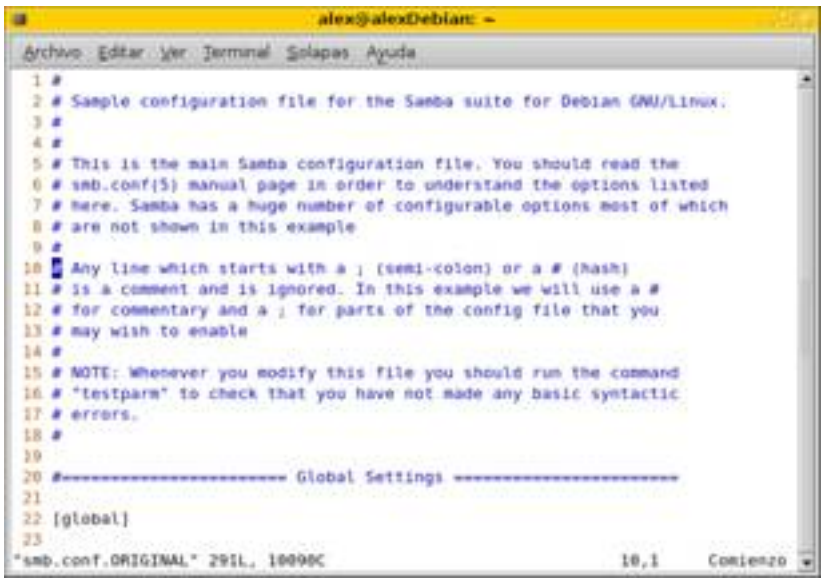
Ctrl+D

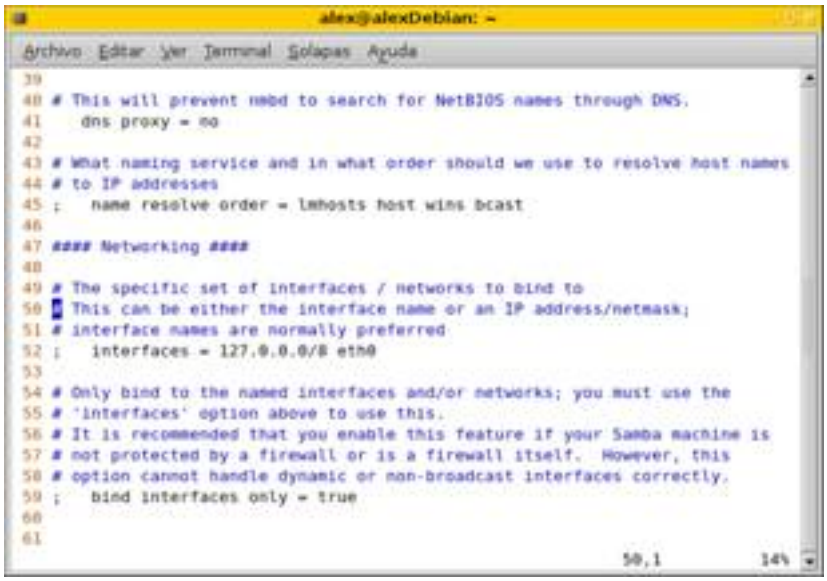
Media pantalla siguiente

Ctrl+U

Media pantalla anterior

Ejemplo, estando en la línea número 10, nos situamos en la 50 pulsando la secuencia de teclas ***50G***





Operaciones con los ficheros

Estando en el modo normal, presionamos **:** (entramos en el modo linea de comandos)

:w

Guardamos el fichero

:w	***Nombre de archivo***

Guardamos el fichero con el nombre indicado

:wq

Guardamos el fichero y salimos del editor (también :x)

:q

Salimos si no ha habido cambios

:	**q**	**!**

Salimos sin guardar los cambios

:e	***archivo***

Si no hemos realizado cambios o los hemos guardado, cerramos el archivo actual y editamos el *archivo*	

:e	**!**	***archivo***

Cerramos el archivo actual perdiendo los cambios si los hubiera y editamos el *archivo*	

:r	***archivo***

Añadimos el	*archivo*

:nºr	***archivo***

Añadimos el	*archivo*

:sh

Abrimos un intérprete de comandos sin salir del editor. Para salir de él escribimos exit

:N,Mw	***archivo***

Añadimos desde la linea N hasta la M (inclusive las dos) al	*archivo*

CTRL+G

Muestra el estado del fichero

Ejemplo, añadimos las 18 primeras líneas a un nuevo archivo llamado *cabeceraSamba.conf* .
Lo vemos:



Copiar, pegar, buscar y reemplazar

Estando en el modo normal, presionamos:

yy

Copiamos la linea actual (también con la tecla Y)

nºyy

Copiamos N líneas desde la posición actual y hacia abajo

yw

Copiamos la palabra situada en la posición actual

y$

Copiamos desde la posición actual hasta el final de la linea

p

Pegamos lo copiado debajo de la linea actual

P

Pegamos lo copiado encima de la linea actual

/	***cadena***

Buscamos la	*cadena*

?	***cadena***

Buscamos la	*cadena*

Utilizaremos la tecla “n” para ir hacia delante y “N” para ir hacia detrás

La	*cadena*

[]

Busca exactamente uno de los caracteres incluidos entre los corchetes

~

El carácter que sigue a “~” debe estar al principio de la linea

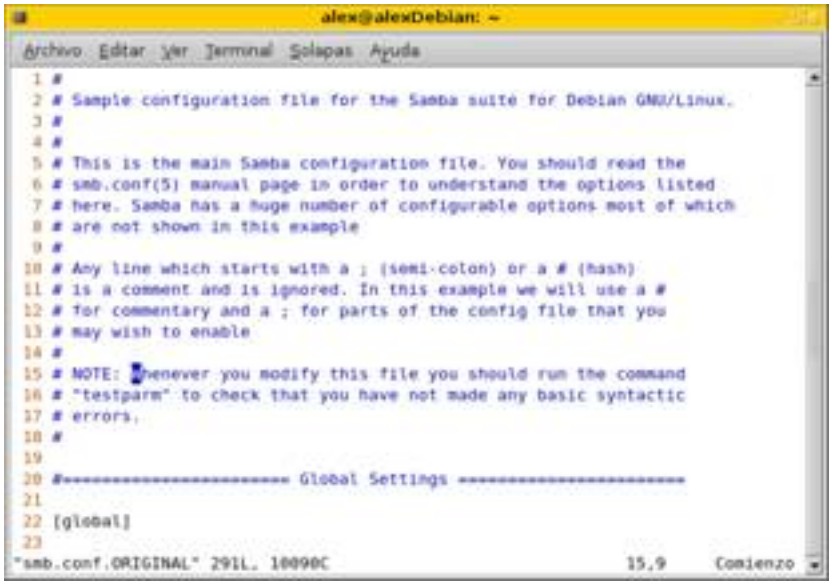
$

El carácter que precede a “$” se busca al final de la linea

[-]

Busca un rango de caracteres

Ejemplo, copiamos desde la posición actual hasta el final de la línea (secuencia de teclas ***y$***), y pegamos en la línea 19 (tecla
p
). Lo vemos:





Deshacer cambios, rehacer

Estando en el modo normal

U

Deshace todos los cambios que se han producido en la linea actual

u

Deshace el último cambio

Control+R

Rehacer un cambio

Ejemplo, deshacemos el pegado anterior. Lo vemos:



Comandos varios

Desde el modo normal

:help	***[argumento]***

Ayuda

:history

Mostrar histórico de comandos

Personalizar Vi

Estando en el modo normal

:set number

Numerar las líneas

:set nonumber

Quitar la numeración de las líneas

:syntax o n/off

Activar/Desactivar el coloreado de sintaxis en archivos de código fuente

:set ic

Realizamos búsquedas ignorando las mayúsculas/minúsculas

:set noic

Realizamos búsquedas con mayúsculas/minúsculas

Si queremos que algunas de estas sentencias estén siempre activas al ejecutar Vim, deberemos crear un archivo llamando *.vimrc* en nuestro directorio home (para saber cual es ejecutar *echo $HOME*)

Ejemplo de *.vimrc* para establecer color de sintaxis y numeración de las líneas



Ventanas

Con Vim podemos dividir la pantalla en diferentes zonas, de tal modo que cada una de ellas se edite un archivo diferente, o incluso, mostrar dos partes diferentes de un mismo fichero.

o Crear dos ventanas mostrando partes diferentes de un mismo archivo

Desde modo normal

:split

Crear una ventana horizontal

:vsplit

Crear una ventana vertical

o Crear dos ventanas una para cada archivo

Desde la linea de comandos

vi -o archivo1 archivo2

Ejemplo, abrimos dos archivos (ejecutamos ***vi o smb.conf.ORIGINAL .bash_profile***). Lo vemos:



Moviéndose entre ventanas

Ctrl+w, Ctrl+w

Para moverte entre vistas/ventanas

Ctrl+w, j

Para mover la vista hacia abajo

Ctrl+w, k

Para mover una vista hacia arriba

Ctrl+w, h

Para mover una vista hacia la izquierda

Ctrl+w, l

Para mover una vista hacia la derecha

Ctrl+w, =

Le indica a Vim que ponga las vistas al mismo tamaño

Ctrl+w, -

Reduce la vista activa en una línea

Ctrl+w,+

Aumenta la vista activa en una línea

Ctrl+w,q

Cierra la ventana activa

Ctrl+w,r

Gira la ventana hacia la derecha

Ctrl+w,R

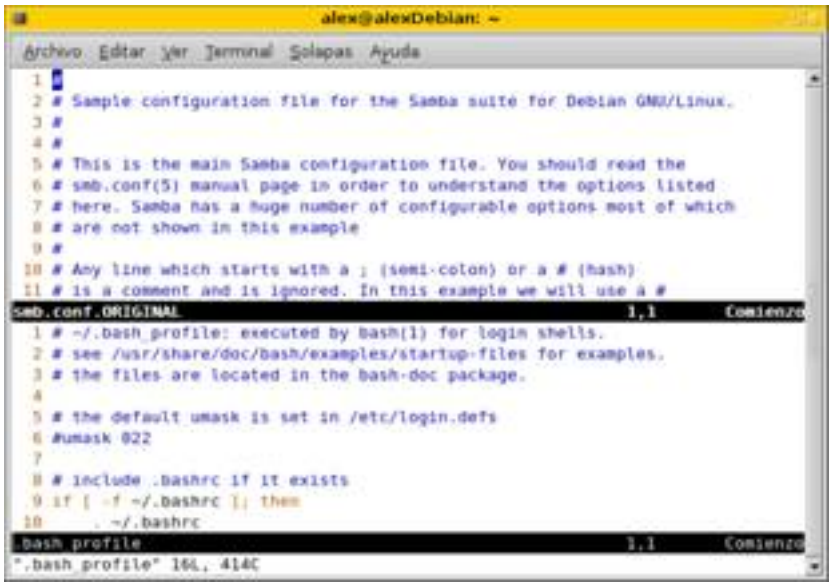
Gira la ventana hacia la izquierda

Teniendo de referencia el ejemplo anterior, nos movemos entre las ventanas teniendo pulsada la tecla ***Ctrl*** y pulsando dos veces la tecla ***w***. Lo vemos:

Ahora en la ventana de abajo



Ahora en la ventana de arriba



8.- Referencias

o http://www.vim.org

o http://www.wikipedia.es

o http://es.opensuse.org

o http://www.marblestation.com/

o http://www.FreeUOC.org Proyecto LPI

o Vim: Referencia rápida. Autor: Joaquín Ataz López. Murcia, Diciembre de 2004

o G.L.U.P. Guía de Linux Para El Usuario. Larry Greenfield

o GNU/Linux: Instalación y primeros pasos. Autores: Matt Welsh, Phil Hughes. Julio de 2002

o Sistema Operativo GNU/Linux básico. UOC. Autores: Roger Borg Viñas. Francesc Aulí Llinas

o Apuntes Sistema Operativo Linux Universidad Politécnica de Madrid. Noviembre 2004

;