Pues resulta que me encuentro administrando unos servidores solaris, toda una novedad para mi.

Lo cierto es que muchas veces me dijeron "aprende vi... aprende vi..." pero la verdad es que yo me sentía basatante cómodo utilizando nano, así que nunca me propuse a aprenderlo.

Pues bien, resulta que solaris 10 (ninguna versión de solaris realmente) trae nano por defecto. Todos los Unix-Like traen vi.

No es muy dificil aprender este poderoso editor de texto. Sin embargo, la documentación que mas me gustó es esta que a continuación expongo.

Tomado de la página: www.emezeta.com

Vim es un editor de ficheros de textos muy versátil, que dispone de una gran flexibilidad a la hora de escribir scripts, modificar ficheros de texto, etc... pero sobretodo, a la hora de **pro gramar**
.

A mi criterio, es el mejor **editor** que existe, sin embargo, a sus usuarios normalmente le suele pasar como a la Pepsi: o te encanta, o lo odias.

Sus detractores engloban las desventajas de vim en dos, muy comunes:

- **Poco «amistoso»**: Es cierto. En cuanto a primeras impresiones, Vim tiene las de perder. Muchos usuarios se asustan sin darle ni una oportunidad. A pesar de su **muy** dura curva de aprendizaje, una vez se aprende sus funciones básicas, la productividad asciende de una manera notable.

- **Atajos dificiles**: Vim es muy potente, pero carece de menús o botones. Todo se hace a través de comandos y atajos que son secuencias de letras y signos. Esto resulta muy frustrante para muchos usuarios que olvidan o no recuerdan con facilidad. Para ello recomiendo alguna chuleta como VIMRefCard .

Las ventajas, sin embargo, son múltiples. **VIM** ocupa muy poco y existe en prácticamente todos los **Linux** o **Unix** disponible. Al ser un programa que se ejecuta en entorno de texto es útil para accesos remotos y edición vía terminal.

Así pues, vamos a darle un breve repaso a su uso básico desde cero, para perder ese miedo y comprobar lo útil que es aprender a usarlo.

VIM: Introducción

Arrancar el vim es muy sencillo. Sólo hay que escribir en una terminal **vim**, seguido del nombre del fichero a editar. Nos aparecerá una ventana en negro, donde nos aparecerá el contenido del fichero (*o en negro si está vacío*). En la parte inferior, nos aparecerán los mensajes o comandos que escribamos para manejar el editor, así como la línea en la que estamos, porcentaje del fichero, etc.

Lo primero que hay que aprender de **Vim** (*muy importante*) es que tiene varios modos de uso:

- Nada más entrar en vim a editar un fichero, estamos en el **modo normal**, en el que podremos introducir **atajos** para realizar operaciones (*borrar linea, deshacer, etc...*).

IMPORTANTE
: En este modo no podemos escribir en el fichero. Las teclas que pulsemos probablemente estarán asociadas a una operación determinada. Muchos de estos comandos (
no todos
) comenzarán por
:
.

- Para escribir texto en el fichero tendremos que entrar en el **modo edición**, que es tan fácil como pulsar la tecla
insert
(
o
i
). Sabrás que has entrado en este modo porque abajo aparecerá el texto
-- INSERTAR --
. Ahora todo lo que tecleemos se estará escribiendo en el fichero de texto. Para volver al
modo normal
sólo hay que pulsar la tecla
ESC
(
Escape
).

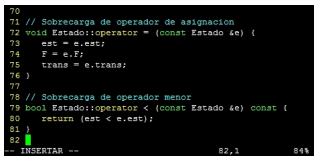
Todo esto puede parecer muy lioso al principio, pero conforme comiences a utilizarlo con frecuencia, verás que resulta cómodo y lo haces de forma automática.

Primeras impresiones

Vim reconoce automáticamente por la extensión del fichero, el lenguaje en el que estamos programando (*.C, .sql, .pl, .latex, .php...*), por lo tanto nos hará un **resaltado de sintaxis** con colores, que nos resultará bastante agradable.

Esta opción puede no estar disponible en algunos linux con versiones minimalistas de **Vim**. Sólo tenemos que instalar la versión completa de vim con
apt-get install vim-common
y escribir (
en el modo normal del Vim
)
:syntax on

.



Sin duda, el resaltado de sintaxis es algo muy valioso para el programador.

Operaciones básicas del editor

Una vez tengamos nuestro texto escrito, necesitaremos saber como realizar algunas operaciones como **guardar fichero**, **salir del editor**, etc...

Como hemos dicho antes, para realizar operaciones que no son de escribir en el fichero, necesitamos entrar en el **modo normal** (*pulsando ESC si estamos en el modo edición*) y a continuación los atajos que queramos:

Secuencia	Significado	¡Mnemotécnica!
:q	Salir del editor sin guardar	quit
:q!	Salir del editor sin guardar y pedir confirmación	quit y pedir confirmación
:wq!	Salir del editor guardando y pedir confirmación	write & quit y confirmación
:w *f2.txt*	Guardar en un fichero llamado **f2.txt**	y s
:e *f1.txt*	Cierra el fichero actual y abre **f1.txt**	ed

Operaciones básicas de texto

En **Vim** como en cualquier editor, necesitaremos manipular rapidamente texto, y algo que enseguida se echa en falta en VIM, son las famosas opciones **Cortar, Copiar y Pegar**.

Con los cursores nos desplazamos por el contenido del fichero hasta llegar al inicio de la zona que queremos copiar. Pulsamos **ESC** (*si estamos en el modo edición*) y la tecla **V** para entrar

en el **modo visual** y nos desplazamos hacia el final de la zona que queramos copiar. Se verá que se remarca en otro color la zona seleccionada.

Una vez tengamos la zona a copiar seleccionada, sólo tenemos que pulsar **C** (*para cortar*) o **Y** (*para copiar*). Nos aparecerá abajo un mensaje **X lineas copiadas**.

Ahora sólo tenemos que desplazarnos a donde queramos pegar ese fragmento y pulsar (*como siempre, en el modo normal, no en el modo edición*) la tecla **P** (*pegar*).

Existen otras formas de copiar (*por método de buffer, por ejemplo*) con la secuencia **"***buffer***y** **y** para copiar y **"***buffer***p** para pegar, pero suelen ser más complejas.

Veamos más operaciones de texto:

Secuencia	Significado	¡Mnemotécnica!
dd	Suprimir línea actual al buffer (*pegar*)	*buffer*
u	Deshacer el último cambio en el fichero	undo
CTRL+R	Rehacer el último cambio en el fichero	redo
guu	Convertir a minusculas la línea actual	lowercase
gUU	Convertir a mayúsculas la línea actual	UPPERCASE
: *num*	Posicionarse en la línea *num*	
gg	Posicionarse al principio del fichero	
G	Posicionarse al final del fichero	
ga	Muestra código ASCII, hex y octal del caracter actual	

Operaciones de búsqueda y sustitución

Otra función que solemos echar de menos enseguida es la de buscar algún texto, reemplazar, etc. En **vim** no puede faltar esa opción, con sus respectivas mejoras y añadidos:

Para buscar un texto, escribimos (*en modo normal, pulsando antes **ESC** si estamos en modo edición*) la secuencia /*palabra*. Veremos que se resalta la palabra encontrada (*o nos avisa de que no existe*). Entonces podemos seguir buscando la próxima coincidencia pulsando **n** o buscarla hacia detrás pulsando **N**.

Para sustituir un texto debemos escribir la secuencia **:%s**/***texto1***/***texto2***/**g**, donde **texto1** es el texto a buscar y **texto2** el texto que será reemplazado. Si incluimos la **g** final (*global*), sustituirá todas las coincidencias que encuentre, sino sólo la primera que encuentre.

Preferencias

Hay ciertas opciones que denominamos preferencias porque son detalles que podemos fijar permanentemente (*o no*) como por ejemplo que queremos mostrar el numerado de línea (*como en los ejemplos de imagen*
), el resaltado de sintaxis, etc...

Ello lo podemos hacer mediante comandos desde el modo normal o en el fichero ~/**.vimrc** (*sin el símbolo*
:
), donde se guardan las preferencias del usuario.

Comando	Significado
:set ts=3	Fija los tabulados a 3 espacios
:set sw=3	Fija los indentados a 3 espacios
:set number/nonumber	Activa/desactiva el numerado en los ficheros
:set backup/nobackup	Activa/desactiva la copia de seguridad automática
:set directory= *dir*	Fija la carpeta donde se harán las copias
:syntax on/off	Activa/desactiva el resaltado de sintaxis
:color *esquema*	Cambia color del vim (*evening, darkblue, desert, e*
:set cindent	Activa indentado automático
:set mouse=a/mouse=	Activa/desactiva el uso del ratón
:set paste/nopaste	Activa/desactiva el modo pegar texto literalmente
:spell	Activa el corrector ortográfico
:setlocal spell spelllang=es	Activa el idioma español del corrector ortográfico
:set spellfile=~/.vimdic	Fija diccionario de palabras desconocidas

Ejecución y comandos externos

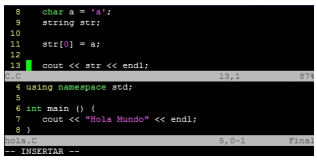
Es posible que mientras estamos editando un fichero, necesitemos ejecutar un comando (*por ejemplo, un ls para ver los archivos*
), esto se puede hacer escribiendo en el modo normal
:!
comando
, en nuestro ejemplo:
:!ls
.

También se puede hacer una pausa en la edición escribiendo **:shell** para realizar alguna

operación y cuando la terminemos, escribir
exit
y volver al editor. Incluso los comandos
:make
y
:cc
se pueden utilizar para ahorrarnos el estar saliendo del editor para hacer makes o ver el último error que nos dió.

Características avanzadas

En **Vim** podemos realizar una separación visual de forma que podamos editar dos (*o más*) ficheros simultaneamente. Para ello escribiremos
:split
fichero.ext
con lo que dividirá la pantalla mostrando los dos ficheros para edición.



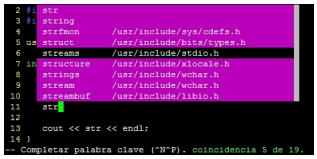
Es posible realizar un **:vsplit** *fichero.ext* para realizar una división vertical en lugar de horizontal, e incluso realizar más divisiones posteriormente, alternando para conseguir más divisiones, siempre y cuando nuestra resolución nos permita.

Para cambiar entre subdivisiones, pulsamos **CTRL+W** y presionamos la flecha de los cursores a donde queramos dirigirnos.

También es posible crear pestañas (*tal como lo hace Firefox*) para tener varios ficheros. Para ello escribimos **:tabnew** seguido del nombre del fichero a crear. Para movernos entre pestañas escribimos
:tabn
, pulsando
gt
o utilizando el ratón (

activando el previo set mouse=a
) para usar los iconos de texto superiores.

Como se puede ver, **Vim** tampoco carece de un sistema de autocompletado, por si no recordamos los nombres de nuestras variables o de las funciones que queremos utilizar. Basta con escribir el caracter inicial y pulsar **CTRL+N**. Nos aparecerá un panel de sugerencias coincidentes para elegir.



Otras operaciones avanzadas pueden ser:

Comando	Significado
=G	Indenta automáticamente todas las líneas de un fichero
{}	Detecta donde está la llave mal cerrada del párrafo actual
:g/^\s*$/d	Elimina las líneas en blanco de un fichero

Existen muchísimas operaciones útiles que no están en este manual, todo es cuestión de ir indagando un poco.

Para aquellos que les haya picado la curiosidad pero no se atreven a utilizar aplicaciones que no tienen entorno gráfico, **Vim** tiene un hermano gemelo, llamado **gVim** (*graphic vim*), una aplicación gráfica para windows, que ayudará a aquellos que quieran iniciarse con este magnífico editor.