Vi

¿Que es Vi?

Dispersos alrededor de su computadora hay miles de archivos de texto. A un nuevo usuario, esto le puede parecer intrascendente, pero casi todo en Slackware Linux usa un archivo de texto plano para la configuración. Esto permite a los usuarios realizar cambios en el sistema de forma rápida, fácil e intuitiva.

En el capítulo 5, vimos algunos comandos como ***cat*** y ***less*** que pueden usarse para leer estos archivos, pero ¿qué pasa si queremos modificarlos?. Para eso, necesitamos un editor de texto, y ***vi*** está a la altura de la tarea.

En resumen, ***vi*** es uno de los editores de texto más antiguos y masivos utilizados hoy en día. Es amado por administradores de sistema, programadores y entusiastas en todo el mundo. De hecho, casi todo este libro fue escrito usando ***vi*** ; solo el siguiente capítulo sobre ***emacs*** fue escrito con ese editor.

Sin embargo, se necesita una explicación más detallada para saber exactamente qué es ***vi*** hoy en día, ya que Slackware Linux técnicamente no incluye ***vi*** . Más bien, Slackware incluye dos vi *"clones"* , ***elvis*** (1) y ***vim*** (1). Estos clones agregan muchas características adicionales a vi, como el resaltado de sintaxis, los modos de edición binarios y el soporte de red. No vamos a profundizar demasiado en todos estos detalles. De manera predeterminada, si ejecuta ***vi*** en Slackware Linux, utilizará ***elvis*** , por lo que todos los ejemplos en este capítulo asumirán que es lo que está usando. Si ha usado otra distribución de Linux anteriormente, puede que esté más familiarizado con ***vim*** . Si es así, puede cambiar el enlace simbólico de `/ usr / bin / vi` para que apunte a `/ usr / bin / vim` , o agregar un alias a los scripts de inicio de su shell. ***vim*** generalmente se considera que tiene más funciones que ***elvis*** , pero ***elvis*** es un programa mucho más pequeño y contiene más funciones que la mayoría los usuarios necesitaran.

vi es muy poderoso, pero también algo engorroso y desafiante para un nuevo usuario. Sin embargo, dominar ***vi*** es una habilidad importante para cualquier administrador de sistema que se respeta ya que ***vi*** se incluye en casi todas las distribuciones de Linux, todos los sistemas BSD y todos los sistemas UNIX existentes. Incluso está incluido en Mac OS X. Una vez que haya aprendido ***vi*** , no tendrá que aprender a utilizar ningun otro editor de texto para trabajar en ninguno de estos sistemas. De hecho, los clones ***vi*** incluso se han portado a sistemas Microsoft Windows, por lo que también puede usarlos allí.

Los distintos modos de vi

Los nuevos usuarios a menudo se sienten frustrados al usar ***vi*** por primera vez.

Cuando se invoca sin ningún argumento, ***vi*** mostrará una pantalla similar a esta.

```
~
~
```

```
~
~
~
~
~
~
~
~
~
                                                Command
```

En este punto, el usuario estará escribiendo y esperará que las teclas que presiona aparezcan en el documento. En cambio, algo realmente extraño sucede. La razón de esto es simple. ***vi*** tiene diferentes *"modos"* . Hay un modo de comando y un modo de inserción.

El modo de comando es el predeterminado; en este modo, cada pulsación realiza una acción particular, como mover el cursor, borrar texto, copiar, buscar, etc.

Abriendo, Guardando, y Saliendo.

Ok, entonces has decidido que quieres aprender a usar ***vi*** . Lo primero que debes hacer es aprender a abrir y guardar archivos.

Abrir archivos es bastante fácil. Simplemente escriba el nombre del archivo como un argumento en la línea de comandos y ***vi*** lo cargará con gusto. Por ejemplo, vi chapter_11.xml abrirá el archivo `chapter_11.xml` y cargará su contenido en la pantalla, suficientemente simple. Pero, ¿qué pasa si hemos terminado con un documento y deseamos guardarlo? Podemos hacerlo en modo de comando usando el comando *: w* . Cuando se encuentra en el modo de comando, al presionar la tecla `:` se ubica temporalmente el cursor en la línea inferior de la ventana y le permite ingresar comandos especiales. (Esto se conoce técnicamente como ex-mode después de la venerable aplicación ***ex*** que no documentaremos aquí.) El comando para guardar su trabajo actual es *: w* . Una vez hecho esto, ***vi*** volverá a escribir sus cambios en el búfer dentro archivo. Si desea abrir otro documento, simplemente use el comando *: e other_document* y ***vi*** lo abrirá para usted. Si ha realizado cambios en el búfer pero aún no lo ha guardado, *: e* fallará e imprimirá un mensaje de advertencia en la línea inferior. Puede omitir esto con el comando *: e!* . La mayoría de los comandos de modo ex en ***vi*** pueden ser *"forzados"* agregando `!` a ellos. Esto le indica a ***vi*** que desea abandonar cualquier cambio que haya realizado en el búfer y abrir el otro documento inmediatamente.

Pero, ¿qué sucede si no me gustan mis cambios y quiero salir o comenzar de nuevo? Eso también puede hacerse fácilmente. La ejecución del comando *: e!* sin ningún argumento volverá a abrir el documento actual desde el principio. Salir de ***vi*** es tan simple como ejecutar el comando *: q* si no ha realizado ningún cambio en el búfer, o *: q!* si desea salir y abandonar esos cambios.

Desplazándose

Moverse en ***vi*** es quizás lo más difícil de aprender para un nuevo usuario. ***vi*** no usa tradicionalmente las teclas de flecha direccionales para el movimiento del cursor, aunque en Slackware Linux es una opción. Más bien, el movimiento es simplemente otro comando emitido en modo comando. La razón

de esto es bastante simple. ***vi*** en realidad es anterior a la inclusión de las teclas de flecha direccionales en los teclados. Por lo tanto, el movimiento del cursor tenía que realizarse utilizando las pocas teclas disponibles, por lo que se utilizan las teclas de la mano derecha *"fila de inicio"* de h, j, k y l. Estas teclas moverán el cursor cada vez que ***vi*** esté en modo de comando.

Aquí hay una tabla corta para ayudarte a recordar cómo funcionan.

Comando	**Resultado**
h	Mover el cursor un caracter a la izquierda.
j	Mover el cursor una linea abajo.
k	Mover el cursor una linea arriba.
l	Mover el cursor una linea a la derecha.

Moverse es un poco más poderoso que eso. Como muchas teclas de comando, estas teclas de movimiento aceptan argumentos numéricos. Por ejemplo, **10j** moverá el cursor hacia abajo 10 líneas. También puede moverse al final o al principio de la línea actual con $ y ^, respectivamente.

Editando un documento

Ahora que podemos abrir y guardar documentos, así como desplazarnos por ellos, es hora de aprender cómo editarlos. El medio principal de edición es ingresar al modo de inserción con las teclas de comando i o a. Estos pueden insertar texto en la ubicación actual del cursor, o agregarlo al final de la ubicación actual. Una vez en el modo de inserción, puede escribir cualquier texto normalmente y se colocará en su documento. Puede volver al modo de comando para guardar sus cambios presionando la tecla ESC.

Vi Cheat Sheet

Como vi puede ser difícil de aprender, preparé una pequeña hoja de trucos que debería ayudarlo con lo básico hasta que comience a sentirse cómodo.

Comando	**Resultado**
h	Mover el cursor un caracter a la izquierda.
j	Mover el cursor una linea abajo.
k	Mover el cursor una linea arriba.
l	Mover el cursor una linea a la derecha.
10j	Mover el cursor 10 lineas hacia abajo.
G	Mover el cursor al final del archivo.
^	Mover el cursor al principio de la linea.
$	Mover el cursor al final de la linea.
dd	Eliminar linea (y almacenar en el buffer del portapapeles).
5dd	Eliminar 5 lineas (y almacenar en el buffer del portapapeles).
dw	Eliminar una palabra (y almacenar en el buffer del portapapeles).
5dw	Eliminar 5 palabras (y almacenar en el buffer del portapapeles).
yy	Yank (copiar) una linea (y almacenar en el buffer del portapapeles).
yw	Yank (copiar) una palabra (y almacenar en el buffer del portapapeles).

Comando	Resultado
5yw	Yank (copiar) 5 palabras (y almacenar en el buffer del portapapeles).
p	Pegar texto en linea por debajo del cursor.
P	Pegar texto en linea por encima del cursor.
r	Reemplazar un caracter.
R	Reemplazar múltiples caracteres.
x	Eliminar un caracter.
X	Eliminar el caracter previo.
u	Deshacer la ultima acción.
:s'old'new'g	Reemplazar todas las ocurrencias de 'old' con 'new' (solo en la linea actual).
:%s'old'new'g	Reemplazar todas las ocurrencias de 'old' con 'new' (todas las lineas).
/asdf	Buscar la siguiente ocurrencia asdf.
:q	Salir (sin guardar cambios).
:w	Guardar el documento actual.
:w file	Guardar como 'file'.
:x	Guardar y salir.

Navegación del capítulo

Capítulo anterior: Trabajando con sistemas de archivos

Siguiente Capítulo: Emacs

Fuentes

- Original source: http://www.slackbook.org/beta
- Originally written by Alan Hicks, Chris Lumens, David Cantrell, Logan Johnson

slackbook, vi, text editor