12

A gentle introduction to vi
Una Elegante introducción a vi

There is an old joke about a visitor to New York City asking a passerby for directions to the city's famous classical music venue.
Hay un viejo chiste sobre un visitante de la ciudad de Nueva York que pregunta a un transeúnte cómo llegar al famoso lugar de música clásica de la ciudad.

Visitor: Excuse me, how do I get to Carnegie Hall?
Visitante: Disculpe, ¿cómo llego al Carnegie Hall?

Passerby: Practice, practice, practice!
Passerby: ¡Practica, practica, practica!

Learning the Linux command line, like becoming an accomplished pianist, is not something that we pick up in an afternoon. It takes years of practice. In this chapter, we will introduce the vi (pronounced "vee eye") text editor, one of the core programs in the Unix tradition. vi is somewhat notorious for its difficult user interface, but when we see a master sit down at the keyboard and begin to "play," we will indeed be witness to some great art. We won't become masters in this chapter, but when we are done, we will know how to play the equivalent of "Chopsticks" in vi.
Aprender la línea de comandos de Linux, como convertirse en un pianista consumado, no es algo que aprendamos en una tarde. Se necesitan años de práctica. En este capítulo, presentaremos el editor de texto vi (que se pronuncia "ojo en u"), uno de los programas centrales de la tradición Unix. vi es algo notorio por su difícil interfaz de usuario, pero cuando vemos a un maestro sentarse al teclado y comenzar a "tocar", ciertamente seremos testigos de un gran arte. No nos convertiremos en maestros en este capítulo, pero cuando terminemos, sabremos cómo tocar el equivalente de "Chopsticks" en vi.

Why We Should Learn vi
Por qué deberíamos aprender vi

In this modern age of graphical editors and easy-to-use text-based editors such as nano , why should we learn vi ? There are three good reasons.
En esta era moderna de editores gráficos y editores basados en texto fáciles de usar como nano, ¿por qué deberíamos aprender vi? Hay tres buenas razones.

- vi is almost always available. If we have a system with no graphical interface, such as a remote server or a local system with a broken X configuration, this can be a lifesaver. nano , while increasingly popular, is still not universal. POSIX, a standard for program compatibility on Unix systems, requires

that vi be present.
vi casi siempre está disponible. Si tenemos un sistema sin interfaz gráfica, como un servidor remoto o un sistema local con una configuración X rota, esto puede ser un salvavidas. nano, aunque es cada vez más popular, todavía no es universal. POSIX, un estándar para la compatibilidad de programas en sistemas Unix, requiere que vi esté presente.

- vi is lightweight and fast. For many tasks, it's easier to bring up vi than it is to find the graphical text editor in the menus and wait for its multiple megabytes to load. In addition, vi is designed for typing speed. As we will see, a skilled vi user never has to lift their fingers from the keyboard while editing.
vi es ligero y rápido. Para muchas tareas, es más fácil abrir vi que encontrar el editor de texto gráfico en los menús y esperar a que se carguen sus múltiples megabytes. Además, vi está diseñado para escribir con rapidez. Como veremos, un usuario experto de vi nunca tiene que levantar los dedos del teclado mientras edita.

- We don't want other Linux and Unix users to think we are cowards.
No queremos que otros usuarios de Linux y Unix piensen que somos cobardes.

Okay, maybe two good reasons.
Bien, quizás dos buenas razones.

A Little Background
Un poco de historia

The first version of vi was written in 1976 by Bill Joy, a University of Cali fornia at Berkeley student who later went on to cofound Sun Microsystems.
La primera versión de vi fue escrita en 1976 por Bill Joy, un estudiante de la Universidad de Cali fornia en Berkeley que luego fue cofundador de Sun Microsystems.

vi derives its name from the word "visual," because it was intended to allow editing on a video terminal with a moving cursor. Previous to visual editors, there were line editors that operated on a single line of text at a time. To specify a change, we tell a line editor to go to a particular line and describe what change to make, such as adding or deleting text. With the advent of video terminals (rather than printer-based terminals like teletypes), visual editing became possible. vi actually incorporates a powerful line editor called ex , and we can use line editing commands while using vi.
vi deriva su nombre de la palabra "visual", porque estaba destinado a permitir la edición en un terminal de video con un cursor en movimiento. Antes de los editores visuales, había editores de línea que operaban en una sola línea de texto a la vez. Para especificar un cambio, le decimos a un editor de línea que vaya a una línea en particular y describa qué cambio hacer, como agregar o eliminar texto. Con la llegada de los terminales de video (en lugar de los terminales basados en impresoras como los teletipos), la edición visual se hizo posible. vi en realidad incorpora un poderoso editor de línea llamado ex, y podemos usar comandos de edición de línea mientras usamos vi.

Most Linux distributions don't include real vi ; rather, they ship with an enhanced replacement called vim (which is short for "vi improved") written by Bram Moolenaar. vim is a substantial improvement over traditional Unix vi and is usually symbolically linked (or aliased) to the name vi on Linux systems. In the discussions that follow, we will assume that we have a program called vi that is really vim.
La mayoría de las distribuciones de Linux no incluyen vi real; más bien, se envían con un reemplazo

mejorado llamado vim (que es la abreviatura de “vi mejorado”) escrito por Bram Moolenaar. vim es una mejora sustancial sobre Unix vi tradicional y generalmente está vinculado simbólicamente (o con alias) al nombre vi en los sistemas Linux. En las discusiones que siguen, asumiremos que tenemos un programa llamado vi que es realmente vim.

Starting and Stopping vi
Iniciar y detener vi

To start vi , we simply enter the following:
Para iniciar vi, simplemente ingresamos lo siguiente:

```
[me@linuxbox ~]$ vi
```

A screen like the following should appear.
Debería aparecer una pantalla como la siguiente.

```
~
~
~                           VIM - Vi IMproved
~
~                            version 8.1.2269
~                       by Bram Moolenaar et al.
~                  Modified by team+vim@tracker.debian.org
~                Vim is open source and freely distributable
~
~                        Help poor children in Uganda!
~                type  :help iccf<Enter>       for information
~
~                type  :q<Enter>               to  xit
~                type  :help<Enter>  or  <F1>  for on-line help
~                type  :help version8<Enter>   for version info
~
~                        Running in Vi compatible mode
~                type  :set nocp<Enter>        for Vim defaults
~                type  :help cp-default<Enter> for info on this
~
~
~
```

Compatibility Mode
Modo de compatibilidad

In the example startup screen, we see the text “Running in Vi compatible mode.” This means that vim will run in a mode that is closer to the normal behavior of vi rather than the enhanced behavior of vim . For the purposes of this chapter, we will want to run vim with its enhanced behavior. To do

> this, you have a few options. Try running vim instead of vi . If that works, consider adding alias vi='vim' to your .bashrc file. Alternatively, use this command to add a line to your vim configuration file:
> En la pantalla de inicio de ejemplo, vemos el texto "Ejecutando en modo compatible con Vi". Esto significa que vim se ejecutará en un modo más cercano al comportamiento normal de vi en lugar del comportamiento mejorado de vim. Para los propósitos de este capítulo, queremos ejecutar vim con su comportamiento mejorado. Para hacer esto, tiene algunas opciones. Intente ejecutar vim en lugar de vi. Si eso funciona, considere agregar alias vi = 'vim' a su archivo .bashrc. Alternativamente, use este comando para agregar una línea a su archivo de configuración de vim:

```
echo "set nocp" >> ~/.vimrc
```

> Different Linux distributions package vim in different ways. Some distributions install a minimal version of vim by default that supports only a limited set of vim features. While performing the lessons that follow, you may encounter missing features. If this is the case, install the full version of vim.
> Diferentes distribuciones de Linux empaquetan vim de diferentes maneras. Algunas distribuciones instalan una versión mínima de vim de forma predeterminada que admite solo un conjunto limitado de funciones de vim. Mientras realiza las lecciones que siguen, puede encontrar funciones que faltan. Si este es el caso, instale la versión completa de vim.

Just as we did with nano earlier, the first thing to learn is how to exit. To exit, we enter the following command (note that the colon character is part of the command):
Al igual que hicimos anteriormente con nano, lo primero que debemos aprender es cómo salir. Para salir, ingresamos el siguiente comando (tenga en cuenta que el carácter de dos puntos es parte del comando):

```
:q
```

The shell prompt should return. If, for some reason, vi will not quit (usually because we made a change to a file that has not yet been saved), we can tell vi that we really mean it by adding an exclamation point to the command.
El indicador de shell debería regresar. Si, por alguna razón, vi no se cierra (generalmente porque hicimos un cambio en un archivo que aún no se ha guardado), podemos decirle a vi que realmente lo decimos en serio agregando un signo de exclamación al comando.

```
:q!
```

Tip
Sugerencias

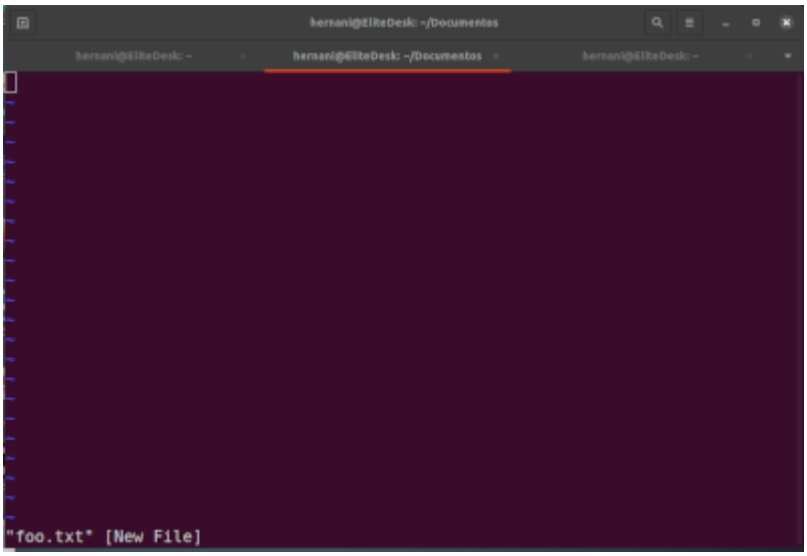
> If you get “lost” in vi , try pressing the esc key twice to find your way again.
> Si se “pierde” en vi, intente presionar la tecla esc dos veces para encontrar el camino nuevamente.

Editing Modes
Modos de edición

Let's start vi again, this time passing the name of a nonexistent file to it. This is how we can create a new file with vi:
Empecemos vi de nuevo, esta vez pasándole el nombre de un archivo inexistente. Así es como podemos crear un nuevo archivo con vi:

```
[me@linuxbox ~]$ rm -f foo.txt
[me@linuxbox ~]$ vi foo.txt
```

If all goes well, we should get a screen like this:
Si todo va bien, deberíamos tener una pantalla como esta:



The leading tilde characters (~) indicate that no text exists on that line. This shows that we have an empty file. Do not type anything yet!
Los caracteres de tilde iniciales (~) indican que no existe texto en esa línea. Esto muestra que tenemos un archivo vacío. ¡No escribas nada todavía!

The second most important thing to learn about vi (after learning how to exit) is that vi is a modal editor. When vi starts, it begins in command mode.
La segunda cosa más importante que debe aprender sobre vi (después de aprender a salir) es que vi es un editor modal. Cuando vi se inicia, comienza en modo comando.

In this mode, almost every key is a command, so if we were to start typing, vi would basically go crazy and make a big mess.
En este modo, casi todas las teclas son un comando, por lo que si comenzáramos a escribir, vi básicamente se volvería loco y haría un gran lío.

Entering Insert Mode
Entrar en el modo de inserción

To add some text to our file, we must first enter insert mode. To do this, we press the i key. Afterward, we should see the following at the bottom of the screen if vim is running in its usual enhanced mode (this will not appear in vi compatible mode):
Para agregar algo de texto a nuestro archivo, primero debemos ingresar al modo de inserción. Para hacer esto, presionamos la tecla i. Luego, deberíamos ver lo siguiente en la parte inferior de la pantalla si vim se está ejecutando en su modo mejorado habitual (esto no aparecerá en el modo compatible con vi):

```
-- INSERT --
```

Now we can enter some text. Try this:
Ahora podemos ingresar algo de texto. Prueba esto:

```
The quick brown fox jumped over the lazy dog.
```

To exit insert mode and return to command mode, press the esc key.
Para salir del modo de inserción y volver al modo de comando, presione la tecla esc.

Saving Our Work
Salvando Nuestro Trabajo

To save the change we just made to our file, we must enter an ex command while in command mode. This is easily done by pressing the : key. After doing this, a colon character should appear at the bottom of the screen.
Para guardar el cambio que acabamos de hacer en nuestro archivo, debemos ingresar un comando ex mientras estamos en modo comando. Esto se hace fácilmente presionando la tecla:. Después de hacer esto, debería aparecer un carácter de dos puntos en la parte inferior de la pantalla.

```
:
```

To write our modified file, we follow the colon with a w and then press enter.
Para escribir nuestro archivo modificado, seguimos los dos puntos con una w y luego presionamos enter.

```
:w
```

The file will be written to the hard drive, and we should get a confirmation message at the bottom of the screen, like this:
El archivo se escribirá en el disco duro y deberíamos recibir un mensaje de confirmación en la parte inferior de la pantalla, como este:

```
"foo.txt" [New] 1L, 46C written
```

> # Tip
> # Sugerencias

> If you read the vim documentation, you will notice that (confusingly) command mode is called normal mode and ex commands are called command mode. Beware.
> Si lee la documentación de vim, notará que (de manera confusa) el modo de comando se llama modo normal y los comandos ex se llaman modo de comando. Tener cuidado.

Moving the Cursor Around
Mover el cursor alrededor

While in command mode, vi offers a large number of movement commands, some of which it shares with less . Table 12-1 lists a subset.
Mientras está en modo comando, vi ofrece una gran cantidad de comandos de movimiento, algunos de los cuales comparte con menos. La Tabla 12-1 enumera un subconjunto.

> Table 12-1: Cursor Movement Keys
> Tabla 12-1: Teclas de movimiento del cursor

Key	Moves the cursor
l or right arrow	Right one character. Derecho un caracter.
h or left arrow	Left one character. Izquierda un caracter.
j or down arrow	Down one line. Bajar una linea
k or up arrow	Up one line. Subir una linea
0 (zero)	To the beginning of the current line. al principio de la linea corriente
^	To the first non-whitespace character on the current line. Al primer carácter que no es un espacio en blanco en la línea actual.
$	To the end of the current line. Hasta el final de la línea actual.
w	To the beginning of the next word or punctuation character. Al principio de la siguiente palabra o signo de puntuación
W	To the beginning of the next word, ignoring punctuation characters. Al principio de la siguiente palabra, ignorando los caracteres de puntuación.
b	To the beginning of the previous word or punctuationcharacter. Al comienzo de la palabra anterior o el carácter de puntuación.
B	To the beginning of the previous word, ignoring punctuation characters. Al principio de la palabra anterior, ignorando los caracteres de puntuación.

Key	Moves the cursor
ctrl-F or page down	Down one page. Abajo una página.
ctrl-B or page up	Up one page. Subir una página.
numberG	To line number. For example, 1G moves to the first line of the file. Al número de línea. Por ejemplo, 1G se mueve a la primera línea del archivo.
G	To the last line of the file. Hasta la última línea del archivo.

Why are the h , j , k , and l keys used for cursor movement? When vi was originally written, not all video terminals had arrow keys, and skilled typists could use regular keyboard keys to move the cursor without ever having to lift their fingers from the keyboard.
¿Por qué se utilizan las teclas h, j, kyl para el movimiento del cursor? Cuando vi se escribió originalmente, no todos los terminales de video tenían teclas de flecha, y los mecanógrafos expertos podían usar las teclas normales del teclado para mover el cursor sin tener que levantar los dedos del teclado.

Many commands in vi can be prefixed with a number, as with the G command listed in Table 12-1. By prefixing a command with a number, we may specify the number of times a command is to be carried out. For example, the command 5j causes vi to move the cursor down five lines.
Muchos comandos en vi pueden tener un prefijo con un número, como con el comando G listado en la Tabla 12-1. Al prefijar un comando con un número, podemos especificar el número de veces que se ejecutará un comando. Por ejemplo, el comando 5j hace que vi mueva el cursor cinco líneas hacia abajo.

Basic Editing
Edición básica

Most editing consists of a few basic operations such as inserting text, deleting text, and moving text around by cutting and pasting. vi , of course, supports all of these operations in its own unique way. vi also provides a limited form of undo. If we press the u key while in command mode, vi will undo the last change that you made. This will come in handy as we try some of the basic editing commands.
La mayor parte de la edición consiste en algunas operaciones básicas, como insertar texto, eliminar texto y mover texto cortando y pegando. vi, por supuesto, soporta todas estas operaciones de una manera única. vi también proporciona una forma limitada de deshacer. Si presionamos la tecla u mientras está en modo comando, vi deshará el último cambio que realizó. Esto será útil mientras probamos algunos de los comandos de edición básicos.

Appending Text
Agregar texto

vi has several different ways of entering insert mode. We have already used the i command to insert text.
vi tiene varias formas diferentes de ingresar al modo de inserción. Ya hemos utilizado el comando i para

insertar texto.

Let's go back to our foo.txt file for a moment.
Volvamos a nuestro archivo foo.txt por un momento.

```
The quick brown fox jumped over the lazy dog.
```

If we wanted to add some text to the end of this sentence, we would discover that the i command will not do it, because we can't move the cursor beyond the end of the line. vi provides a command to append text, the sensibly-named a command. If we move the cursor to the end of the line and type a , the cursor will move past the end of the line and vi will enter insert mode. This will allow us to add some more text.
Si quisiéramos agregar algo de texto al final de esta oración, descubriríamos que el comando i no lo hará, porque no podemos mover el cursor más allá del final de la línea. vi proporciona un comando para agregar texto, el comando sensiblemente llamado. Si movemos el cursor al final de la línea y escribimos a, el cursor se moverá más allá del final de la línea y vi entrará en el modo de inserción. Esto nos permitirá agregar más texto.

```
The quick brown fox jumped over the lazy dog. It was cool.
```

Remember to press the esc key to exit insert mode.
Recuerde presionar la tecla esc para salir del modo de inserción.

Because we will almost always want to append text to the end of a line, vi offers a shortcut to move to the end of the current line and start appending. It's the A command. Let's try it and add some more lines to our file.
Debido a que casi siempre querremos agregar texto al final de una línea, vi ofrece un atajo para ir al final de la línea actual y comenzar a agregar. Es el comando A. Intentémoslo y agreguemos algunas líneas más a nuestro archivo.

First, we'll move the cursor to the beginning of the line using the 0 (zero) command. Now we type A and add the following lines of text:
Primero, moveremos el cursor al principio de la línea usando el comando 0 (cero). Ahora escribimos A y agregamos las siguientes líneas de texto:

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
Line 5
```

Again, press the esc key to exit insert mode.
Nuevamente, presione la tecla esc para salir del modo de inserción.

As we can see, the A command is more useful as it moves the cursor to the end of the line before starting insert mode.
Como podemos ver, el comando A es más útil ya que mueve el cursor al final de la línea antes de iniciar el modo de inserción.

Opening a Line
Abrir una línea

Another way we can insert text is by “opening” a line. This inserts a blank line between two existing lines and enters insert mode. This has two variants, as described in Table 12-2.
Otra forma de insertar texto es "abriendo" una línea. Esto inserta una línea en blanco entre dos líneas existentes y entra en el modo de inserción. Tiene dos variantes, como se describe en la Tabla 12-2.

> Table 12-2: Line Opening Keys
> Tabla 12-2: Teclas de apertura de línea

Command	Opens
o	The line below the current line La línea debajo de la línea actual
O	The line above the current line La línea sobre la línea actual

We can demonstrate this as follows: place the cursor on line 3 and then type o.
Podemos demostrar esto de la siguiente manera: coloque el cursor en la línea 3 y luego escriba o.

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3

Line 4
Line 5
```

A new line was opened below the third line, and we entered insert mode.
Se abrió una nueva línea debajo de la tercera línea y entramos en el modo de inserción.

Exit insert mode by pressing the esc key. Press u to undo our change.
Salga del modo de inserción presionando la tecla esc. Presione u para deshacer nuestro cambio.

Type O to open the line above the cursor.
Escriba O para abrir la línea sobre el cursor.

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2

Line 3
Line 4
Line 5
```

Exit insert mode by pressing esc , and undo our change by typing u.
Salga del modo de inserción presionando esc, y deshaga nuestro cambio escribiendo u.

Deleting Text
Eliminar texto

As we might expect, vi offers a variety of ways to delete text, all of which contain one of two keystrokes. First, the x command will delete a character at the cursor location. x may be preceded by a number specifying how many characters are to be deleted. The d command is more general purpose. Like x , it may be preceded by a number specifying the number of times the deletion is to be performed. In addition, d is always followed by a movement command that controls the size of the deletion. Table 12-3 provides some examples.
Como era de esperar, vi ofrece una variedad de formas de eliminar texto, todas las cuales contienen una de dos pulsaciones de teclas. Primero, el comando x eliminará un carácter en la ubicación del cursor. x puede ir precedido de un número que especifique cuántos caracteres se eliminarán. El comando d tiene un propósito más general. Al igual que x, puede ir precedido de un número que especifique el número de veces que se realizará la eliminación. Además, d siempre va seguida de un comando de movimiento que controla el tamaño de la eliminación. La Tabla 12-3 proporciona algunos ejemplos.

> Table 12-3: Text Deletion Commands
> Tabla 12-3: Comandos de eliminación de texto

Command	Deletes
x	The current character El personaje actual
3x	The current character and the next two characters El personaje actual y los dos personajes siguientes
dd	The current line La linea actual
5dd	The current line and the next four lines La línea actual y las siguientes cuatro líneas
dW	From the current cursor position to the beginning of the next word Desde la posición actual del cursor hasta el comienzo de la siguiente palabra
d$	From the current cursor location to the end of the current line Desde la ubicación actual del cursor hasta el final de la línea actual
d0	From the current cursor location to the beginning of the line Desde la ubicación actual del cursor hasta el comienzo de la línea
d^	From the current cursor location to the first non-whitespace character in the line Desde la ubicación actual del cursor hasta el primer carácter que no sea un espacio en blanco en la línea

Command	Deletes
dG	From the current line to the end of the file Desde la línea actual hasta el final del archivo
d20G	From the current line to the twentieth line of the file Desde la línea actual hasta la vigésima línea del archivo

Place the cursor on the word It on the first line of our text. Press the x key repeatedly until the rest of the sentence is deleted. Next, press the u key repeatedly until the deletion is undone.
Coloque el cursor sobre la palabra It en la primera línea de nuestro texto. Presione la tecla x repetidamente hasta que se elimine el resto de la oración. A continuación, presione la tecla u repetidamente hasta que se deshaga la eliminación.

> Note
> Nota.

> Real vi supports only a single level of undo. vim supports multiple levels.
> Real vi admite solo un nivel de deshacer. vim admite varios niveles.

Let's try the deletion again, this time using the d command. Again, move the cursor to the word It and type dW to delete the word.
Intentemos la eliminación de nuevo, esta vez con el comando d. Nuevamente, mueva el cursor a la palabra It y escriba dW para borrar la palabra.

```
The quick brown fox jumped over the lazy dog. was cool.
Line 2
Line 3
Line 4
Line 5
```

Type d$ to delete from the cursor position to the end of the line.
Escriba d $ para eliminar desde la posición del cursor hasta el final de la línea.

```
The quick brown fox jumped over the lazy dog.
Line 2
Line 3
Line 4
Line 5
```

Type dG to delete from the current line to the end of the file.
Escriba dG para eliminar desde la línea actual hasta el final del archivo.

Type u three times to undo the deletion.
Escriba u tres veces para deshacer la eliminación.

Cutting, Copying, and Pasting Text
Cortar, copiar y pegar texto

The d command not only deletes text, it also “cuts” text. Each time we use the d command, the deletion is copied into a paste buffer (think clipboard) that we can later recall with the p command to paste the contents of the buffer either before or after the cursor.
El comando d no solo elimina texto, también lo “corta”. Cada vez que usamos el comando d, la eliminación se copia en un búfer de pegado (piense en el portapapeles) que luego podemos recuperar con el comando p para pegar el contenido del búfer antes o después del cursor.

The y command is used to “yank” (copy) text in much the same way the d command is used to cut text. Table 12-4 provides some examples of combining the y command with various movement commands.
El comando y se usa para "tirar" (copiar) texto de la misma manera que el comando d se usa para cortar texto. La Tabla 12-4 proporciona algunos ejemplos de combinación del comando y con varios comandos de movimiento.

> Table 12-4: Yanking Commands
> Tabla 12-4: Comandos de arrastre

Command	Copies
yy	The current line La linea actual
5yy	The current line and the next four lines La línea actual y las siguientes cuatro líneas
yW	From the current cursor position to the beginning of the next word Desde la posición actual del cursor hasta el comienzo de la siguiente palabra
y$	From the current cursor location to the end of the current line Desde la ubicación actual del cursor hasta el final de la línea actual
y0	From the current cursor location to the beginning of the line Desde la ubicación actual del cursor hasta el comienzo de la línea
y^	From the current cursor location to the first non-whitespace character in the line Desde la ubicación actual del cursor hasta el primer carácter que no sea un espacio en blanco en la línea
yG	From the current line to the end of the file Desde la línea actual hasta el final del archivo
y20G	From the current line to the twentieth line of the file Desde la línea actual hasta la vigésima línea del archivo

Let's try some copy-and-paste. Place the cursor on the first line of the text and type yy to copy the current line. Next, move the cursor to the last line (G) and type p to paste the line below the current line.
Intentemos copiar y pegar. Coloque el cursor en la primera línea del texto y escriba yy para copiar la línea actual. A continuación, mueva el cursor a la última línea (G) y escriba p para pegar la línea debajo de la línea actual.

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
Line 5
The quick brown fox jumped over the lazy dog. It was cool.
```

Just as before, the u command will undo our change. With the cursor still positioned on the last line of the file, type P to paste the text above the current line.
Al igual que antes, el comando u deshará nuestro cambio. Con el cursor aún posicionado en la última línea del archivo, escriba P para pegar el texto sobre la línea actual.

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
The quick brown fox jumped over the lazy dog. It was cool.
Line 5
```

Try some of the other y commands in Table 12-4 and get to know the behavior of both the p and P commands. When you are done, return the file to its original state.
Pruebe algunos de los otros comandos y de la tabla 12-4 y conozca el comportamiento de los comandos py P. Cuando haya terminado, devuelva el archivo a su estado original.

Joining Lines
Uniendo Líneas

vi is rather strict about its idea of a line. Normally, it is not possible to move the cursor to the end of a line and delete the end-of-line character to join one line with the one below it. Because of this, vi provides a specific command, J (not to be confused with j , which is for cursor movement), to join lines together.
vi es bastante estricto sobre su idea de línea. Normalmente, no es posible mover el cursor al final de una línea y borrar el carácter de final de línea para unir una línea con la que está debajo. Debido a esto, vi proporciona un comando específico, J (que no debe confundirse con j, que es para el movimiento del cursor), para unir líneas.

If we place the cursor on line 3 and type the J command, here's what happens:
Si colocamos el cursor en la línea 3 y escribimos el comando J, esto es lo que sucede:

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3 Line 4
Line 5
```

Search-and-Replace
Buscar y reemplazar

vi has the capability to move the cursor to locations based on searches. It can do this either on a single line or over an entire file. It can also perform text replacements with or without confirmation from the user.
vi tiene la capacidad de mover el cursor a ubicaciones basadas en búsquedas. Puede hacer esto en una sola línea o en un archivo completo. También puede realizar reemplazos de texto con o sin confirmación del usuario.

Searching Within a Line
Buscando dentro de una línea

The f command searches a line and moves the cursor to the next instance of a specified character. For example, the command fa would move the cursor to the next occurrence of the character a within the current line. After performing a character search within a line, the search may be repeated by typing a semicolon.
El comando f busca una línea y mueve el cursor a la siguiente instancia de un carácter especificado. Por ejemplo, el comando fa movería el cursor a la siguiente aparición del carácter a dentro de la línea actual. Después de realizar una búsqueda de caracteres dentro de una línea, la búsqueda puede repetirse escribiendo un punto y coma.

Searching the Entire File
Buscando en todo el archivo

To move the cursor to the next occurrence of a word or phrase, the / command is used. This works the same way as we learned earlier in the less program. When you type the / command, a / will appear at the bottom of the screen. Next, type the word or phrase to be searched for, followed by the enter key. The cursor will move to the next location containing the search string. A search may be repeated using the previous search string with the n command. Here's an example:
Para mover el cursor a la siguiente aparición de una palabra o frase, se usa el comando /. Esto funciona de la misma manera que aprendimos anteriormente en el programa less. Cuando escriba el comando /, aparecerá un / en la parte inferior de la pantalla. A continuación, escriba la palabra o frase que desee buscar, seguida de la tecla Intro. El cursor se moverá a la siguiente ubicación que contiene la cadena de búsqueda. Una búsqueda puede repetirse usando la cadena de búsqueda anterior con el comando n. Aquí tienes un ejemplo:

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
Line 5
```

Place the cursor on the first line of the file. Type this and press enter:
Coloque el cursor en la primera línea del archivo. Escriba esto y presione enter:

```
/Line
```

The cursor will move to line 2. Next, type n and the cursor will move to line 3. Repeating the n command will move the cursor down the file until it runs out of matches. While we have so far used only words and phrases for our search patterns, vi allows the use of regular expressions, a powerful method of expressing complex text patterns. We will cover regular expressions fully in Chapter 19.
El cursor se moverá a la línea 2. Luego, escriba ny el cursor se moverá a la línea 3. Si repite el comando n, el cursor se moverá hacia abajo en el archivo hasta que se agoten las coincidencias. Si bien hasta ahora solo hemos utilizado palabras y frases para nuestros patrones de búsqueda, vi permite el uso de expresiones regulares, un método poderoso para expresar patrones de texto complejos. Cubriremos las expresiones regulares por completo en el Capítulo 19.

Global Search-and-Replace
Búsqueda y reemplazo global

vi uses an ex command to perform search-and-replace operations (called substitution in vi) over a range of lines or the entire file. To change the word Line to line for the entire file, we would enter the following command:
vi usa un comando ex para realizar operaciones de búsqueda y reemplazo (llamado sustitución en vi) en un rango de líneas o en todo el archivo. Para cambiar la palabra Línea a línea para todo el archivo, deberíamos ingresar el siguiente comando:

```
:%s/Line/line/g
```

Let's break down this command into separate items and see what each one does (see Table 12-5).
Dividamos este comando en elementos separados y veamos qué hace cada uno (consulte la Tabla 12-5).

> Table 12-5: An Example of Global Search-and-Replace Syntax
> Tabla 12-5: Un ejemplo de sintaxis global de búsqueda y reemplazo

Item	Meaning
:	The colon character starts an ex command. El carácter de dos puntos inicia un comando ex.

Item	Meaning
%	This specifies the range of lines for the operation. % is a shortcut meaning from the first line to the last line. Alternately, the range could have been specified 1,5 (since our file is five lines long) or 1,$, which means “from line 1 to the last line in the file.” If the range of lines is omitted, the operation is performed only on the current line. Esto especifica el rango de líneas para la operación. % es un atajo que significa desde la primera línea hasta la última línea. Alternativamente, el rango podría haberse especificado 1,5 (ya que nuestro archivo tiene cinco líneas de largo) o 1, $, que significa "desde la línea 1 hasta la última línea del archivo". Si se omite el rango de líneas, la operación se realiza solo en la línea actual.
s	This specifies the operation. In this case, it’s substitution (search-and-replace). Esto especifica la operación. En este caso, es sustitución (buscar y reemplazar).
/Line/line/	This specifies the search pattern and the replacement text. Esto especifica el patrón de búsqueda y el texto de reemplazo.
g	This means “global” in the sense that the search-and-replace is performed on every instance of the search string in the line. If omitted, only the first instance of the search string on each line is replaced. Esto significa "global" en el sentido de que la búsqueda y reemplazo se realiza en cada instancia de la cadena de búsqueda en la línea. Si se omite, solo se reemplaza la primera instancia de la cadena de búsqueda en cada línea.

After executing our search-and-replace command, our file looks like this:
Después de ejecutar nuestro comando de búsqueda y reemplazo, nuestro archivo se ve así:

```
The quick brown fox jumped over the lazy dog. It was cool.
line 2
line 3
line 4
line 5
```

We can also specify a substitution command with user confirmation.
También podemos especificar un comando de sustitución con la confirmación del usuario.

This is done by adding a c to the end of the command. Here’s an example:
Esto se hace agregando una c al final del comando. Aquí tienes un ejemplo:

```
:%s/line/Line/gc
```

This command will change our file back to its previous form; however, before each substitution, vi stops and asks us to confirm the substitution with this message:
Este comando cambiará nuestro archivo a su forma anterior; sin embargo, antes de cada sustitución, vi se detiene y nos pide que confirmemos la sustitución con este mensaje:

```
replace with Line (y/n/a/q/l/^E/^Y)?
```

Each of the characters within the parentheses is a possible choice, as described in Table 12-6.
Cada uno de los caracteres entre paréntesis es una opción posible, como se describe en la Tabla 12-6.

> Table 12-6: Replace Confirmation Keys
> Tabla 12-6: Reemplazo de teclas de confirmación

Key	Action
y	Perform the substitution. Realice la sustitución.
n	Skip this instance of the pattern. Omita esta instancia del patrón.
a	Perform the substitution on this and all subsequent instances of the pattern. Realice la sustitución en esta y todas las instancias posteriores del patrón.
q or esc	Quit substituting. Deja de sustituir.
l	Perform this substitution and then quit. This is short for “last.” Realice esta sustitución y luego salga. Esta es la abreviatura de "último".
ctrl-E, ctrl-Y	Scroll down and scroll up, respectively. This is useful for viewing the context of the proposed substitution. Desplácese hacia abajo y hacia arriba, respectivamente. Esto es útil para ver el contexto de la sustitución propuesta.

If you type y , the substitution will be performed. n will cause vi to skip this instance and move on to the next one.
Si escribe y, se realizará la sustitución. n hará que vi omita esta instancia y pase a la siguiente.

Editing Multiple Files
Editar varios archivos

It’s often useful to edit more than one file at a time. You might need to make changes to multiple files, or you might need to copy content from one file into another. With vi we can open multiple files for editing by specifying them on the command line.
A menudo, es útil editar más de un archivo a la vez. Es posible que deba realizar cambios en varios archivos o puede que necesite copiar el contenido de un archivo a otro. Con vi podemos abrir varios archivos para editarlos especificándolos en la línea de comando.

```
vi file1 file2 file3 . . .
```

Let’s exit our existing vi session and create a new file for editing. Type :wq to exit vi , saving our modified text. Next, we’ll create an additional file in our home directory that we can play with. We’ll create the file by capturing some output from the ls command.
Salgamos de nuestra sesión vi existente y creemos un nuevo archivo para editar. Escriba: wq para salir de vi,

guardando nuestro texto modificado. A continuación, crearemos un archivo adicional en nuestro directorio de inicio con el que podemos jugar. Crearemos el archivo capturando algunos resultados del comando ls.

```
[me@linuxbox ~]$ ls -l /usr/bin > ls-output.txt
```

Let's edit our old file and our new one with vi.
Editemos nuestro archivo antiguo y el nuevo con vi.

```
[me@linuxbox ~]$ vi foo.txt ls-output.txt
```

vi will start, and we will see the first file on the screen.
vi se iniciará y veremos el primer archivo en la pantalla.

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
Line 5
Switching Between Files
```

To switch from one file to the next, use this ex command:
Para cambiar de un archivo al siguiente, use este comando ex:

```
:bn
```

To move back to the previous file, use the following:
Para volver al archivo anterior, utilice lo siguiente:

```
:bp
```

While we can move from one file to another, vi enforces a policy that prevents us from switching files if the current file has unsaved changes. To force vi to switch files and abandon your changes, add an exclamation point (!) to the command.
Aunque podemos pasar de un archivo a otro, vi impone una política que nos impide cambiar de archivo si el archivo actual tiene cambios sin guardar. Para obligar a vi a cambiar de archivo y abandonar sus cambios, agregue un signo de exclamación (!) Al comando.

In addition to the switching method already described, vim (and some versions of vi) provides some ex commands that make multiple files easier to manage. We can view a list of files being edited with the :buffers command. Doing so will display a list of the files at the bottom of the display.
Además del método de conmutación ya descrito, vim (y algunas versiones de vi) proporciona algunos comandos ex que facilitan la administración de varios archivos. Podemos ver una lista de archivos que se están editando con el comando: buffers. Al hacerlo, se mostrará una lista de los archivos en la parte inferior de la pantalla.

```
:buffers
1 %a    "foo.txt"           line 1
2       "ls-output.txt"     line 0
Press ENTER or type command to continue
```

To switch to another buffer (file), type :buffer followed by the number of the buffer you want to edit. For example, to switch from buffer 1 containing the file foo.txt to buffer 2 containing the file ls-output.txt, we would type this:
Para cambiar a otro búfer (archivo), escriba: búfer seguido del número del búfer que desea editar. Por ejemplo, para cambiar del búfer 1 que contiene el archivo foo.txt al búfer 2 que contiene el archivo ls-output.txt, escribiríamos esto:

```
:buffer 2
```

Our screen now displays the second file. Another way we can change buffers is to use the :bn (short for buffer next) and :bp (short for buffer previous) commands mentioned earlier.
Nuestra pantalla ahora muestra el segundo archivo. Otra forma en que podemos cambiar los búferes es usar los comandos: bn (abreviatura de búfer siguiente) y: bp (abreviatura de búfer anterior) mencionados anteriormente.

Opening Additional Files for Editing
Abrir archivos adicionales para editarlos

It's also possible to add files to our current editing session. The ex command :e (short for "edit") followed by a filename will open an additional file. Let's end our current editing session and return to the command line.
También es posible agregar archivos a nuestra sesión de edición actual. El comando ex: e (abreviatura de "editar") seguido de un nombre de archivo abrirá un archivo adicional. Terminemos nuestra sesión de edición actual y regresemos a la línea de comandos.

Start vi again with just one file.
Inicie vi de nuevo con un solo archivo

```
[me@linuxbox ~]$ vi foo.txt
```

To add our second file, enter the following:
Para agregar nuestro segundo archivo, ingrese lo siguiente:

```
:e ls-output.txt
```

It should appear on the screen. The first file is still present, as we can verify.
Debería aparecer en la pantalla. El primer archivo todavía está presente, como podemos verificar.

```
:buffers
1 #     "foo.txt"           line 1
2 %a    "ls-output.txt"     line 0
Press ENTER or type command to continue
```

Copying Content from One File into Another
Copiar contenido de un archivo a otro

Often while editing multiple files, we will want to copy a portion of one file into another file that we are editing. This is easily done using the usual yank and paste commands we used earlier. We can demonstrate as follows. First, using our two files, switch to buffer 1 (foo.txt) by entering this:
A menudo, al editar varios archivos, querremos copiar una parte de un archivo en otro archivo que estamos editando. Esto se hace fácilmente usando los comandos usuales de tirar y pegar que usamos anteriormente. Podemos demostrar lo siguiente. Primero, usando nuestros dos archivos, cambie al búfer 1 (foo.txt) ingresando esto:

```
:buffer 1
```

That command should give us this:
Ese comando debería darnos esto:

```
The quick brown fox jumped over the lazy dog. It was cool.
Line 2
Line 3
Line 4
Line 5
```

Next, move the cursor to the first line, and type yy to yank (copy) the line.
A continuación, mueva el cursor a la primera línea y escriba yy para tirar (copiar) la línea.

Switch to the second buffer by entering the following:
Cambie al segundo búfer ingresando lo siguiente:

```
:buffer 2
```

The screen will now contain some file listings like this (only a portion is shown here):
La pantalla ahora contendrá algunos listados de archivos como este (aquí solo se muestra una parte):

```
total 343700
-rwxr-xr-x 1    root    root    31316   2017-12-05 08:58    [
-rwxr-xr-x 1    root    root    8240    2017-12-09 13:39    411toppm
-rwxr-xr-x 1    root    root    111276  2018-01-31 13:36    a2p
-rwxr-xr-x 1    root    root    25368   2017-10-06 20:16    a52dec
-rwxr-xr-x 1    root    root    11532   2017-05-04 17:43    aafire
-rwxr-xr-x 1    root    root    7292    2017-05-04 17:43    aainfo
```

Move the cursor to the first line and paste the line we copied from the preceding file by typing the p command.
Mueva el cursor a la primera línea y pegue la línea que copiamos del archivo anterior escribiendo el comando p.

```
total 343700
The quick brown fox jumped over the lazy dog.
total 343700
-rwxr-xr-x 1    root    root    31316   2017-12-05 08:58    [
-rwxr-xr-x 1    root    root    8240    2017-12-09 13:39    411toppm
-rwxr-xr-x 1    root    root    111276  2018-01-31 13:36    a2p
-rwxr-xr-x 1    root    root    25368   2017-10-06 20:16    a52dec
-rwxr-xr-x 1    root    root    11532   2017-05-04 17:43    aafire
-rwxr-xr-x 1    root    root    7292    2017-05-04 17:43    aainfo
```

Inserting an Entire File into Another
Insertar un archivo completo en otro

It's also possible to insert an entire file into one we are editing. To see this in action, let's end our vi session and start a new one with just a single file.
También es posible insertar un archivo completo en uno que estamos editando. Para ver esto en acción, terminemos nuestra sesión vi y comencemos una nueva con un solo archivo.

```
[me@linuxbox ~]$ vi ls-output.txt
```

We will see our file listing again.
Volveremos a ver nuestra lista de archivos.

Move the cursor to the third line and then enter the following ex command:
Mueva el cursor a la tercera línea y luego ingrese el siguiente comando ex:

```
:r foo.txt
```

The :r command (short for "read") inserts the specified file below the cursor position. Our screen should now look like this:
El comando: r (abreviatura de "leer") inserta el archivo especificado debajo de la posición del cursor. Nuestra pantalla ahora debería verse así:

Saving Our Work
Salvando Nuestro Trabajo

Like everything else in vi , there are several different ways to save our edited files. We have already covered the ex command :w , but there are some others we may also find helpful.
Como todo lo demás en vi, hay varias formas diferentes de guardar nuestros archivos editados. Ya hemos cubierto el comando ex: w, pero hay algunos otros que también pueden resultar útiles.

In command mode, typing ZZ will save the current file and exit vi.
En el modo de comando, escribir ZZ guardará el archivo actual y saldrá de vi.

Likewise, the ex command :wq will combine the :w and :q commands into one that will both save the file and exit.

Del mismo modo, el comando ex: wq combinará los comandos: w y: q en uno que guardará el archivo y saldrá.

The :w command may also specify an optional filename. This acts like Save As. For example, if we were editing foo.txt and wanted to save an alternate version called foo1.txt, we would enter the following:
El comando: w también puede especificar un nombre de archivo opcional. Esto actúa como Guardar como. Por ejemplo, si estuviéramos editando foo.txt y quisiéramos guardar una versión alternativa llamada foo1.txt, ingresaríamos lo siguiente:

```
:w foo1.txt
```

> ## Note
> ## Nota
>
> While this command saves the file under a new name, it does not change the name of the file you are editing. As you continue to edit, you will still be editing foo.txt, not foo1.txt.
> Si bien este comando guarda el archivo con un nuevo nombre, no cambia el nombre del archivo que está editando. A medida que continúe editando, seguirá editando foo.txt, no foo1.txt.

Summing Up
Resumen

With this basic set of skills, we can now perform most of the text editing needed to maintain a typical Linux system. Learning to use vim on a regular basis will pay off in the long run. Since vi -style editors are so deeply embedded in Unix culture, we will see many other programs that have been influenced by its design. less is a good example of this influence.
Con este conjunto básico de habilidades, ahora podemos realizar la mayor parte de la edición de texto necesaria para mantener un sistema Linux típico. Aprender a usar vim de forma regular dará sus frutos a largo plazo. Dado que los editores de estilo vi están tan profundamente arraigados en la cultura Unix, veremos muchos otros programas que han sido influenciados por su diseño. menos es un buen ejemplo de esta influencia.