

# Una guía para principiantes a Vim



Lucy Mitchell

[Seguir](#)

Una introducción al editor de texto de línea de comandos y algunos consejos útiles para comenzar.



Esta publicación cubre

- ¿Qué es vim?

- Por qué comencé a aprenderlo y lo que estoy disfrutando al respecto
- Consejos para noobs y para comenzar

No cubre

- Complementos
- .vimrc
- Atajos específicos
- Cómo salir de vim \*

---

## **¿Qué es vim?**

vim es un editor de texto que se ejecuta en la terminal en sistemas tipo Unix.

La primera versión de vim ahora tiene 27 años (!)

Y fue creada por Bram Moolenaar. vim significa

VI iMproved, ya que vim se basa en vi. también

existen otros sucesores (como nvi y elvis).

**Oh.... Entonces, ¿qué es vi?**

vi (pronunciado VEE-EYE) es el editor de texto

original en Unix. Todo en vi está disponible en

vim, más algunos extras.

**¡Frio! ¿Puedo usar vi entonces?**

¡No tan fácilmente como podrías pensar! Si

intenta abrir un archivo usándolo `vi`

<nombrearchivo> simplemente ... se abre en vim.

Este es el por qué:

```
Last login: Thu Jan  3 16:41:08 on ttys003
[16:47:54] vim
♥ cd
[16:48:01] ~
♥ which vi
/usr/bin/vi
[16:48:07] ~
♥ l /usr/bin/vi
lrwxr-xr-x  1 root  wheel    3B  9 Dec 15:24 /usr/bin/vi -> vim
[16:48:18] ~
♥ █
```

Veamos esto:

En los sistemas Unix, `cd` significa Cambiar directorio y, sin argumentos posteriores, lo reubicará en el directorio de inicio. (No fue necesario hacer esto, solo quería demostrar que lo

que sucedió después fue independiente de la ubicación).

`which` devuelve la ruta del comando solicitado (en este caso `vi`),. Como puede haber múltiples versiones del mismo comando ubicadas en diferentes directorios, `which` mostrará la que está utilizando actualmente en el terminal. Incluso si existen múltiples coincidencias, carga el primer partido en el entorno LOOKUP PATH del shell e ignora el resto, razón por la cual solo un resultado regresa.

`ls` es un alias que existe en `my ~/.bash_profile` y es la abreviatura de `ls -l,, o list the contents of this directory, in long format, on one line.`

Básicamente, esto es "muéstrame todas las cosas".

(Puede encontrar una tonelada más banderas [aquí](#) para `ls` que se deberán enviar con prontitud por el rabbithole de banderas en general. Si usted va a hacer esto, por lo menos lo hacen en la línea de comandos en lugar de buscando en Google y escriba `man ls` donde los `man` espectáculos de mando tu información del manual del programa y los imprime. ¡Genial!)

Y ahora a la parte interesante. La primera sección de lo que se muestra muestra los [permisos](#) del [archivo](#) , pero debido a que comienza con un `l`, muestra que en realidad es un enlace simbólico (es un 'enlace suave' a otro archivo o ubicación, de la misma manera que puede crear un acceso directo en el escritorio a un programa o ubicación

de archivo en otro lugar cuando se utiliza una interfaz gráfica). Así que cuando escribo en `vim` la ubicación real al que apunta es ... .. `/usr/bin/vim`. Y por eso se `vim` abre.

## **¿Por qué debería aprenderlo?**

Lo más probable es que si estás leyendo este artículo ya estás al menos parcialmente invertido. Avanza si no necesitas convencerte.

Las personas tienen diferentes experiencias de aprendizaje de `vim` (algunos argumentan que hay una curva de aprendizaje enorme y muy empinada para entender los conceptos básicos) pero todos están de acuerdo: una vez que alcanzas un nivel moderado de uso, te hace más rápido



editar tus código. Como dice un recurso, puedes aprender a [editar a la velocidad del pensamiento](#) .

**Serás más eficiente.** La cuestión es que debes esforzarte en aprender lo *suficiente como* para alcanzar un umbral mínimo para realmente cosechar esta eficiencia, y eso no es tarea fácil.

Hasta este punto, solo será lento (más lento de lo normal, lo que puede ser frustrante). Pero como con todas las habilidades, cuanto más practiques,

1. cuanto más rápido reconozca su cerebro y comience a automatizar qué teclas hacen qué (a menudo erróneamente llamado "memoria muscular")
2. cuanto más competente se convierta en combinaciones y resolución de problemas

*(Por ejemplo, en el primer obstáculo, solía encontrarme muy confuso que (ispoilers!)  
jEstaba abajo y k arriba porque mi cerebro esperaba que fueran al revés. Tomó algunas horas pero ahora es totalmente inconsciente y  
Estoy muy feliz de haber persistido).*

Además, es rápido. Vengo de un entorno de uso de Atom, que consume muchos recursos ya que se ejecuta en Electron, y carga un navegador web completo para cada instancia de Atom que está abierta, por lo que puede ser un poco lento y lento de cargar. No es así para vim!

Por último, esta es su oportunidad de aprender algo nuevo, tanto la sintaxis como los elementos

conceptuales y abstractos. La cultura de programación se ha apropiado de la palabra `grok` de su etimología de la ciencia para significar que no solo puede regurgitar las palabras o los caracteres de una manera práctica, sino que comprende el espíritu del lenguaje. Las diferencias entre los lenguajes de programación de computadoras son más que solo la sintaxis y la aplicación. Existe una visión del mundo profundamente diferente subyacente en ellos, y esta es su oportunidad de aprender una nueva. Para una buena introducción a vim, lea [esto](#) , escrito por la persona que lo hizo.


Para mí, aprender a usar vim ha sido *increíble* .

¿Recuerdas cómo se sintió cuando comenzaste a

aprender a programar? ¿Cuando algo funcionó y te sentiste como un mago? Todavía soy relativamente nuevo en VIM, pero constantemente me sorprende lo expresivo que puede ser con tanta franqueza; Estos comandos concisos ejercen poder y eficacia. Todavía tengo esa sensación de mago cuando aprendo un nuevo comando o aprecio una nueva combinación en vim, y todavía tengo mucho que aprender.

Actualmente ayudo a enseñar en un [campamento de programación](#) y en las primeras dos semanas de mis alumnos les animo a que no tengan miedo de buscar nuevos atajos usando [este repositorio](#).de conceptos básicos (por favor, póngase en contacto si tiene alguna adición). La apreciación que tengo de vim es como la

sensación que tuve por primera vez cuando

aprendí - `control-up/down` en Atom,

**Bien, cállate, estoy dentro. ¿Qué hago?**

1.

Tipo de **vimtutor completo** `vimtutor` en

la terminal. En cualquier sitio.

Simplemente hazlo.

```
=====
=   W e l c o m e   t o   t h e   V I M   T u t o r   -   V e r s i o n 1.7   =
=====

Vim is a very powerful editor that has many commands, too many to
explain in a tutor such as this.  This tutor is designed to describe
enough of the commands that you will be able to easily use Vim as
an all-purpose editor.

The approximate time required to complete the tutor is 25-30 minutes,
depending upon how much time is spent with experimentation.

ATTENTION:
The commands in the lessons will modify the text.  Make a copy of this
file to practice on (if you started "vimtutor" this is already a copy).

It is important to remember that this tutor is set up to teach by
use.  That means that you need to execute the commands to learn them
properly.  If you only read the text, you will forget the commands!

Now, make sure that your Shift-Lock key is NOT depressed and press
the  j  key enough times to move the cursor so that Lesson 1.1
completely fills the screen.
~~~~~
```

[DESCRIPCIÓN DE LA IMAGEN: pantalla de terminal que muestra la página de bienvenida de vimtutor]

¡Hurra! Ahora, ignorar las teclas de flecha y el uso

h, j, k y l en su lugar.

Cosas para recordar:

- vim se llama editor de texto modal  
  
porque tiene muchos modos, y los dos que se presentan primero son el modo de inserción y el modo de comando (o normal).
- cuando carga vim, comienza en modo normal. En este modo, los caracteres que escribe se interpretan como comandos, por ejemplo `x`, eliminarán el carácter que está actualmente debajo del cursor. Para volver a este modo, presione `esc`
- en modo normal, vim interpreta mayúsculas y minúsculas de manera diferente; `O` y `o` no son sinónimos (uno inserta una línea sobre el cursor, uno lo

inserta debajo). Otra buena ilustración de esto es `eso w y e` mueve el cursor hacia adelante hasta el comienzo de la siguiente palabra y el final de la palabra actual, respectivamente. `w` y `E` hacen los mismos trabajos, pero ignorar la puntuación. Una sutil, pero útil, distinción.

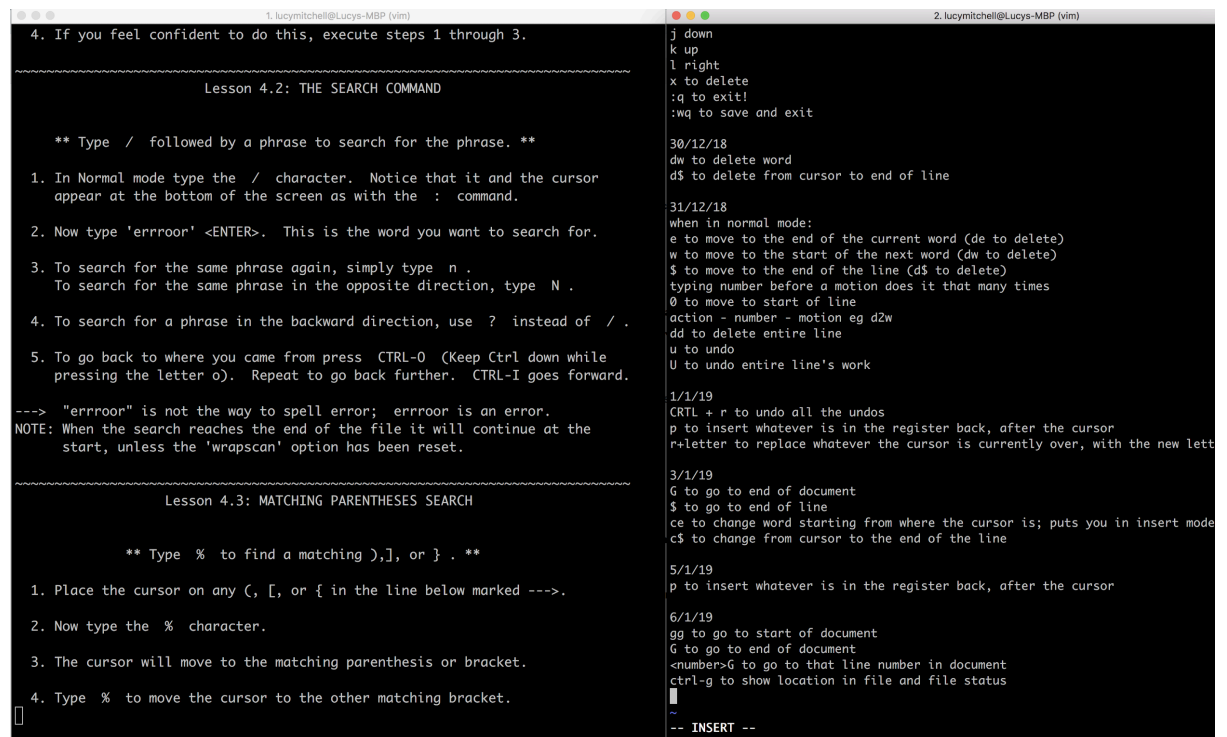
- en el modo de inserción, los caracteres que escribe forman palabras y oraciones como está acostumbrado a ver en un editor de texto. Para ingresar al modo de inserción, presione `i`.

Realmente no hay mucho más que pueda decir: vimtutor hace un gran trabajo explicando. Cubre muchos de los conceptos básicos, pero



definitivamente aumenta esto con la lectura de más información sobre vim en línea, o lee secciones de [este mega manual](#) .

## 2. Haga su propia lista de atajos ... en vim



[DESCRIPCIÓN DE LA IMAGEN: terminal que muestra el archivo vimtutor y los accesos directos uno al lado del otro]

Esto no es tanto sobre el contenido, los atajos reales. Más bien, esto le brinda una oportunidad

decidida y de bajo riesgo para probar lo que está aprendiendo; simplemente leer sobre cosas y probarlas una o dos veces no va a ser suficiente.

También le brinda cierta responsabilidad y se asegura de que realmente esté practicando. Tuve vimtutor y mi cheatsheet abiertos al mismo tiempo como se muestra arriba.

### **3. Practica tutoriales divertidos en línea**

Hay [muchos recursos](#) como era de esperar.

Tómese un tiempo en su calendario para hacer al menos media hora al día. ¡Vamos!

### **4. Desordenar y ser muy, muy lento**

Disfruto de ser lento y aprender a vim porque sigo haciendo algo y luego pienso que *ugh tiene que haber una manera más rápida de hacerlo* y luego investigo y ¡bam! Una nueva comprensión de cómo una pulsación de tecla puede salvarme cuatro.



**5. Vimtutor completo de nuevo ... y de nuevo**

Tiene el tamaño de un bocado, es gratis y cubre los cimientos sin asumir ningún conocimiento previo. Me gustaría llegar al punto en que pudiera enseñar vimtutor mientras duermo.

## **6. Desarrolle su propio procedimiento para aprender atajos**

. He estado haciendo referencia a [esta hoja de trucos](#) y viendo cómo puedo integrar cada atajo (un mínimo de uno nuevo por día) en mi práctica, solo para tener la oportunidad de probarlo. pocas veces. Recuerde, "*poco a poco, uno viaja lejos*".

## **7. Deja de esperar que las listas numeradas sean un sustituto para hacer el trabajo**

. Realmente solo necesitas practicar. Sin embargo, también soy culpable de esto y me encantó leer [esto](#) .

Soy consciente de que existen argumentos para usar neovim / nerdtree / otros complementos, pero para este artículo me estoy centrando en familiarizarme con la sintaxis y el paradigma como principiante. En caso de que comience a programar, al menos le recomiendo poner `syntax` `enable` para que tenga algo de resaltado de sintaxis.