Aprendiendo Vim Progresivamente

Departamento de Ingeniería de Computadoras
Facultad de Informática - Universidad Nacional del Comahue

Este tutorial fue pensado como una guía mínima, para aprender a utilizar el editor Vim [1] en el menor tiempo posible. La idea es la siguiente : primero, se debe aprender lo mínimo para sobrevivir, y luego, de a poco, aprender nuevos comandos útiles.

Aprender Vim es difícil, pero increíblemente eficiente cuando se lo utiliza.

1. Introducción

Vim es un editor de texto de gran utilidad para los administradores de sistemas ya que está disponible en cualquier plataforma UNIX. Esto significa que podrá usar Vim en cualquier sistema de tipo UNIX. Además, resulta recomendable su utilización, porque requiere de muy pocos recursos para su funcionamiento, lo que permite utilizar Vim en maquinas remotas o cuando no pueda ejecutar otros programas debido a problemas con el hardware o el sistema.

> **Advertencia**. Aprender Vim es muy pero muy difícil al principio. Tomará tiempo. Será como intentar aprender a tocar un instrumento de música. Por lo tanto, no sueñe con ser más eficiente con Vim, en menos de tres días, que con cualquier otro editor que ya conozca. Es más, conocer Vim de manera eficiente puede tomar dos o más semanas en vez de sólo tres días.

El tutorial está estructurado en una serie de pasos, como una guía secuencial para el aprendizaje de Vim. Los cuatro pasos que debe seguir son :

1. Sobrevivir
2. Sentirse cómodo
3. Sentirse mejor, con confianza, y mas rápido
4. Utilizar los súper poderes de Vim

Y cuando esté finalizando este camino será un ¡Vim superstar!.

2. Nivel 1 - Sobrevivir

Pasos:

1. Instalar Vim
2. Ejecutar Vim

[1] http://www.vim.org

3. ¡NO HAGA NADA!. Sólo lea.

Luego de instalar el editor Vim inicie su ejecución. Puede hacerlo escribiendo vim[ENTER] en una terminal.

Aquí está la primer lección difícil de entender si es su primer contacto con Vim : cualquier intento de mover el cursor o de escribir, como lo haría con cualquier otro editor, no funcionará.

Si intenta moverse por la pantalla comprobaremos además, que las teclas del cursor no hacen nada, y que el ratón tampoco responde. ¿Qué ocurre?

Lo que ocurre es que Vim tiene diferentes modos de funcionamiento. Y, de manera predeterminada, inicia en *Modo Normal*.

2.1. Modos Fundamentales de Vim

Los dos modos fundamentales son los llamados modos normal y de inserción. El **Modo Normal** es aquel en el que Vim se encuentra al iniciar y en el que se supone que debemos dejarlo cuando no estemos haciendo otra cosa (por eso se llama normal). En él cualquier pulsación de teclas que realicemos se interpretará como comando. Por el contrario en el **Modo de Inserción** Vim se comporta como cualquier otro editor de texto, es decir: las teclas que presionamos se irán mostrando en pantalla en el lugar donde estaba el cursor, siendo interpretado como texto que hay que introducir en el documento que estamos editando.

Debemos hacer notar que Vim llama modo normal a aquel en el que menos se parece a otros editores de texto. Tal vez por ello a veces a este modo se le llama también *Modo de Comando*, aunque el nombre de modo normal es bastante indicativo de la filosofía de Vim. La idea es que siempre que no estemos haciendo otra cosa activemos el modo normal: mientras revisamos el texto escrito, mientras reflexionamos sobre qué añadir, mientras contestamos al teléfono, si hacemos una pausa para tomar un mate... se supone que habremos activado el modo normal.

Para ingresar al Modo Inserción, el cual es muy parecido al funcionamiento de cualquier otro editor, simplemente debe presionar la letra “i”. Ahora debería sentirse un poco mejor, ya que en Modo Inserción puede escribir de manera similar a un editor típico.

Para salir del Modo Inserción, y volver al Modo Normal sólo necesita presionar la tecla ESC.

Recuerde: con las teclas i y ESC, puede cambiar entre el Modo Normal y el Modo Inserción. Detectará que está en modo Inserción porque sobre la esquina inferior izquierda de la ventana de Vim se leerá `INSERT` o `INSERCION`. Al presionar ESC y volver al Modo Normal no habrá ninguna leyenda.

- Ahora es el momento de que aprenda lo necesario para sobrevivir en Modo Normal :

`:q[ENTER]`	salir del editor
`i`	cambia el modo al Modo Inserción. Presione ESC para volver al Modo Normal
`x`	borrar el caracter bajo el cursor
`:wq[ENTER]`	guardar y Salir (:w guardar, :q salir)
`dd`	borrar (y copiar) la línea actual
`p`	pegar

- Recomendados :

`h`	mueve el cursor a la izquierda
`j`	mueve el cursor hacia abajo
`k`	mueve el cursor hacia arriba
`l`	mueve el cursor a la derecha

- Obteniendo ayuda:

`:help <comando>[ENTER]`	Muestra la ayuda acerca del `<comando>`. Puede utilizar `:help` sin <

Sólo necesita aprender estos ocho comandos del Modo Normal para sobrevivir y comenzar. Luego, cuando se sienta cómodo con estos comandos (tal vez en un día o dos), puede continuar con el nivel 2.

Familiarizarse con el Modo Normal. En otros editores, para copiar texto se debe presionar la tecla Ctrl (generalmente Ctrl-C). Más aún, cuando presiona Ctrl en esos editores, es como si todas las teclas cambiaran de significado. Usando Vim en Modo Normal es similar a lo anterior, como si automáticamente Vim presionara la tecla Ctrl por usted.

Una última palabra acerca de las notaciones :

- En vez de escribir Ctrl-, escribiremos en este documento `<C- >`.
- Los comandos que comienzan con ":" finalizan con `<ENTER>`. Por ejemplo, cuando lea :q usted debe escribir :q y presionar `<ENTER>`

3. Nivel 2 - Sentirse Cómodo

Si ya se siente cómodo con los comandos necesarios para sobrevivir puede continuar y aprender unos pocos comandos más. Estos son los sugeridos:

- Variantes para ingresar al Modo Inserción:

`a`	ingresa al modo inserción y se posiciona en la siguiente posición a partir del cursor
`o`	agregar una nueva línea vacía e ingresa al modo inserción
`O`	agregar una nueva línea antes de la línea actual, e ingresa al modo inserción
`cw`	ingresa al modo inserción reemplazando la palabra actual

- Movimientos Básicos:

`0`	saltar al primer caracter de la linea
`^`	saltar al primer caracter no blanco de la linea
`$`	saltar al fin de linea
`g_`	saltar al último caracter no blanco de la linea
`/pattern`	buscar en el documento actual el texto `pattern`

- Copiar y Pegar:

`P`	pegar antes, ya que p es pegar después de la posición del cursor
`yy`	copiar la línea actual, mas fácil pero equivalente a `ddP`
`u`	undo
`<C-r>`	redo

- Cargar/Guardar/Salir/Cambiar Archivo(Buffer):

`:e <path/to/file>`	abrir
`:w`	guardar
`:saveas <path/to/file>`	guardar a `<path/to/file>`
`:x`, `ZZ` o `:wq`	guardar y salir (`:x` solamente guardar si es necesario)
`:q!`	salir sin guardar, también: `:qa!` para salir aun si hubieron buffers
`:bn` (o `:bp`)	mostrar el siguiente archivo (o el anterior) (buffer)

Tome el tiempo que sea necesario para aprender estos comandos, ya que una vez que los conozca podrá trabajar con Vim como lo hace con cualquier otro editor. Puede que aún se sienta un poco incómodo con la dificultad inicial y la manera en que Vim trabaja, pero verá que luego del siguiente nivel Vim le devolverá con creces el trabajo de haberlo aprendido.

4. Nivel 3 - Mejor. Más Fuerte. Más Rápido.

¡Felicitaciones por llegar hasta aquí!, ahora empieza la parte interesante. En el Nivel 3 veremos únicamente comandos compatibles con el viejo editor vi.

4.1. Mejor

- Vea como Vim lo puede ayudar a realizar tareas repetitivas:

`.`	(punto) repetirá el último comando
`N<command>`	repetirá el comando `<command>` N veces

4.1.1. Ejercitación

- Inicie Vim y escriba :

`2dd`	borrará 2 líneas
`3p`	pegará el texto 3 veces
`100idesu [ESC]`	escribirá

“desu “

- Repetimos el último comando recién ejecutado:

  `.` repite el último comando, por lo que escribirá nuevamente 100 “desu “
  `3.` escribirá 3 “desu” (y no 300, que inteligente ¿no?)

4.2. Más fuerte

Conocer como moverse eficientemente con Vim es muy importante. No saltee esta sección.

- Saltos entre líneas:

  `NG` saltar a la linea N (equivalente a `:N`)
  `gg` lo mismo que `1G` o `:0` - saltar al inicio del archivo
  `G` saltar a la ultima línea

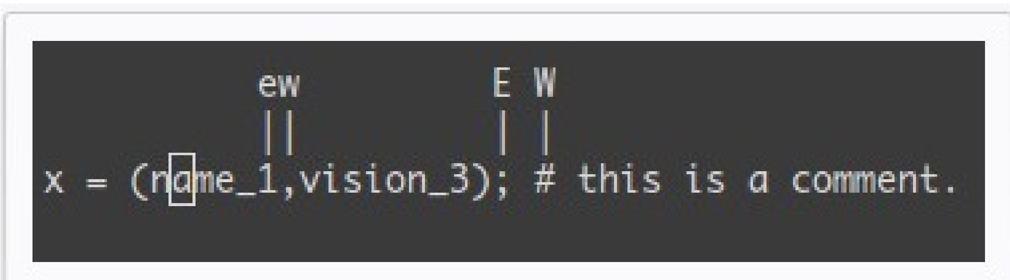
- Movimiento entre palabras:

  `w` ir al inicio de la siguiente palabra
  `e` saltar al final de la palabra actual
  `b` saltar al comienzo de la palabra actual

De manera predeterminada, para Vim, las palabras están compuestas de letras y del caracter “_”. Digamos que una PALABRA es un grupo de letras separadas por un caracter blanco.

- Si sólo desea trabajar con PALABRAS entonces utilice únicamente caracteres en mayúsculas:

  `W` saltar al inicio de la siguiente PALABRA
  `E` saltar al final de la PALABRA actual



- Ahora debe aprender a realizar movimientos muy eficientes:

  `%` saltar a la correspondencia de `(`, `{`, `[`
  `* (o #)` ir a la próxima (o previa) ocurrencia de la palabra bajo el cursor

4.3. Más Rápido

La razón de la importancia de los movimiento en vi tiene una razón : La mayoría de los comandos puede ser utilizado usando el formato general siguiente :

`<posición de inicio><command><posición final>`

- Por ejemplo : 0y$ significa

  `0` ir al comienzo de la linea
  `y` copiar desde aquí
  `$` hasta el fin de la linea actual

También puede ejecutar cosas como “ye”, copiar desde la posición actual hasta el final de la palabra. Y también puede ejecutar cosas como “y2/foo”, lo cual copia hasta la segunda ocurrencia de la palabra “foo”.

Además, lo que es cierto para `y` (copiar -en inglés yank-), es también cierto para `d` (borrar -en inglés delete-), gU (mayúsculas), gu (minúsuclas), etc.

5. Nivel 4 - Súper poderes de Vim

Si utiliza todos los comandos anteriores trabajará cómodamente con Vim. Y además, ahora tiene la preparación necesaria para aprender las características “matadoras”. Algunos de los siguientes comandos son la razón por la cual comencé a utilizar Vim.

- Movimientos en la linea actual: 0 ^$ g_ f F t T , ;

  `0` saltar a la columna 0
  `^` ir al primer caracter en la linea
  `$` saltar al fin de la linea
  `g_` saltar al ultimo caracter en la linea
  `fa` saltar a la próxima ocurrencia de la letra a en la linea.
  Adicionalmente “,” (o “;”) encontrará la próxima ocurrencia o la previa
  `t,` saltar al caracter anterior a “,”
  `3fa` encontrar la 3er ocurrencia de una a en la linea
  `F y T` actúan de la misma manera que “f” y “t” pero en reversa

Un tip útil es: `dt"` el cual borra todo hasta el caracter `"`.

5.1. Selección de una Zona <action>a<object>o <action>i<object>

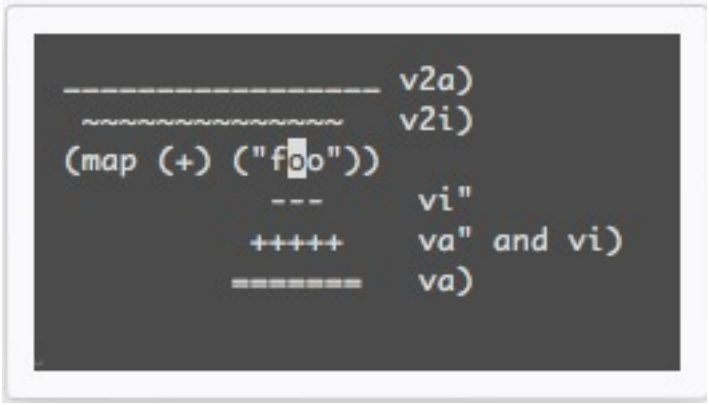
Estos comandos pueden sólo ser utilizados después de un operador en Modo Visual. Pero son muy poderosos. Su principal patrón es :

`<action>a<object>y <action>i<object>`

Donde action puede ser cualquier acción, por ejemplo, d (delete), y (copiar), v (seleccionar en modo visual). El objeto (object) puede ser : w para una palabra, W para una PALABRA (palabra extendida), s para una sentencia, p para un párrafo. Y también, caracteres naturales, tales como `", ', ), }, ]` .

- Suponga que el cursor esta en la primer "o" que aparece en (map (+) ("foo"))

`vi"`	seleccionará foo.
`va"`	seleccionará "foo"
`vi)`	seleccionará "foo"
`va)`	seleccionará ("foo")
`v2i)`	seleccionará map (+) ("foo")
`v2a)`	seleccionará (map (+) ("foo"))



5.2. Selección de bloques rectangulares: <C-v>

Los bloques rectangulares son muy útiles para comentar muchas líneas de código. Típicamente: `0<C-v><C-d>I-- [ESC]`

- Ejemplo:

`^`	ir al principio de la linea
`<C-v>`	Comenzar con la selección de un bloque
`<C-d>`	Mover hacia abajo la selección (también podría ser jjj o %, etc. . .)
`I-- [ESC]`	escribir – al comentario de cada linea

5.3. Completar: <C-n>y <C-p>

En Modo Inserción, sólo escriba el inicio de una palabra, y entonces presione `<C-p>`, Vim completará la palabra mágicamente con una sugerencia encontrada en el documento actual.

- Macros:

  `qa` hacer algo q, `@a, @@`

qa registra sus acciones en el registro a. Entonces @a ejecutará nuevamente la macro guardada dentro del registro a, como si la hubiese escrito.

Ejemplo: en una linea que contiene sólo números “1”, escriba lo siguiente:

- Ejemplo. En una linea que contiene sólo números “1”, escriba lo siguiente:

  `qaYp<C-a>q` recuerde que `<C-a>es Ctrl+a`

- El comando anterior realiza lo siguiente:

`qa`	inicia el registro
`Yp`	duplica la línea
`<C-a>`	incrementa el número
`q`	finaliza el registro de acciones

- Y podemos finalizar el ejemplo con:

`@a`	escribe 2 bajo el 1
`@@`	escribe 3 bajo el 2

Ahora escriba 100@@ y eso creará una lista incremental de números hasta el 103.

5.4. Insertar textos

Dos funciones muy útiles suelen ser poder insertar un archivo de texto en el lugar donde se encuentra el cursor e, insertar la salida de un comando en el lugar donde se encuentra el cursor. Para esto, en modo normal podemos ejecutar los siguientes comandos:

- El comando anterior realiza lo siguiente:

`:!<comando>`	Ejecuta `<comando>` en el shell y retorna sin insertar texto en el documento actual.
`!!<comando>`	Ejecuta `<comando>` en el shell y la salida del mismo se inserta en el documento actual.
`:r <archivo>`	Inserta el contenido de `<archivo>` en el documento actual.

5.5. Selección Visual: `v,V,<C-v>`

Anteriormente se especificó un ejemplo utilizando `<C-v>`. Hay también v y V.

- Una vez que se ha realizado una selección, puede:

`J`	unir todas las lineas juntas
`<(o >)`	realiza una sangría a la izquierda (o a la derecha)
`=`	auto realizar la sangría

- Agregar contenido al final de todas las líneas visualmente seleccionadas:

`<C-v>`	inicia el modo selección

- luego ir a la linea deseada (jjj o `<C-d>o /pattern o%` etc...). Finalmente:

`$`	ir al final de la linea
`A,`	escribir texto, ESC

Todas las líneas seleccionadas contendrán, al final, el texto recién ingresado.

5.6. Dividir las pantallas :split y vsplit

Además de aprender los comandos de esta sección se recomienda leer la ayuda con `:help split`.

- Estos son los comandos mas útiles para dividir la pantalla:

`:split`	crea una división (:vsplit crea una división vertical)
`<C-w><dir>`	donde dir es cualquiera de las teclas de movimiento hjkl o los cursores de
`<C-w>_ (o <C-w>|)`	maximiza el tamaño de la división (o la división vertical)
`<C-w>+ (o <C-w>-)`	Expande (o achica) la división

6. Conclusión

En este documento se explica el 90 % de los comandos que utilizo a diario. Se sugiere que no aprenda mas de uno o dos comandos por día. Después de dos o tres semanas comenzará a sentir el poder de Vim en sus manos.

Aprender a utilizar Vim es una tarea de entrenamiento, no de memorización. Afortunadamente para ello, Vim provee buenas herramientas y excelente documentación. Ejecute vimtutor hasta que esté familiarizado con los comandos básicos. Y también, debería leer esta ayuda cuidadosamente: `:help usr_02.txt`.

Ahí aprenderá acerca de !, directorios, registros, plugins y muchas otras cosas. Estudie Vim como si estuviese aprendiendo a tocar el piano y todo irá bien.

7. Licencia

[2] http://yannesposito.com/Scratch/en/blog/Learn-Vim-Progressively/