Começando com VIM: O Editor de Texto

Primeiros passos com VIM. Dicas de VIM para iniciantes. Como começar a usar o VIM.

22/Mai/2016 — 12 minutos de leitura

Editar este artigo

A tarefa que um Desenvolvedor de Software precisa cumprir com a maior agilidade é a escrita/edição de texto. Depois de pensar por algumas horas, planejar o que vai fazer e como fazer (quais funções, métodos, classes criar), nós partimos para a escrita do código fonte no nosso editor de texto favorito ou em uma IDE .

Existem várias técnicas para você ficar fera no teclado, porém hoje vou apresentar outra forma de agilizar as coisas. Vou apresentar (pela minha perspectiva) o **VIM**.

O VIM é um editor de texto muito amado por alguns e odiado por outros. A primeira vez que você abre o VIM, você pensa: *como é que eu faço pra sair daqui.*

Existe até uma piada (velha) que diz:

- Por que o programador usa VIM?

- Por que ele entrou no VIM uma vez e nunca descobriu o comando pra sair.

> *HUE HUE HUE.*

Eu sou apaixonado pelo Sublime Text a alguns anos e você pode sentir isso nesse post , onde eu falo sobre o uso do Sublime Text no dia-a-dia, nesse outro post , onde eu ensino instalar e configurar o Sublime no Ubuntu, nesse outro post , que eu mostrei alguns plugins que uso diariamente e no meu .dotfiles , onde você encontra mais plugins e configurações legais para o Sublime.

Mas, se o Sublime Text atende hoje 100% meu Workflow e eu sou apaixonado por esse editor, então por que eu fui inventar de aprender usar o VIM?

A resposta é simples: **fui experimentar**.

Prólogo

Eu sempre vi algumas pessoas aclamando o editor como se ele fosse um deus de alguma religião, e de fato é. Existe até uma Igreja do VIM , onde você pode aprender como exorcizar um computador!

editor de texto e fui conhecer melhor. Em São Paulo existe até um grupo de Meetup, o VIM-SP . Gostei bastante da galera que usa o editor e acabei me interessando mais em aprender a usa-lo.

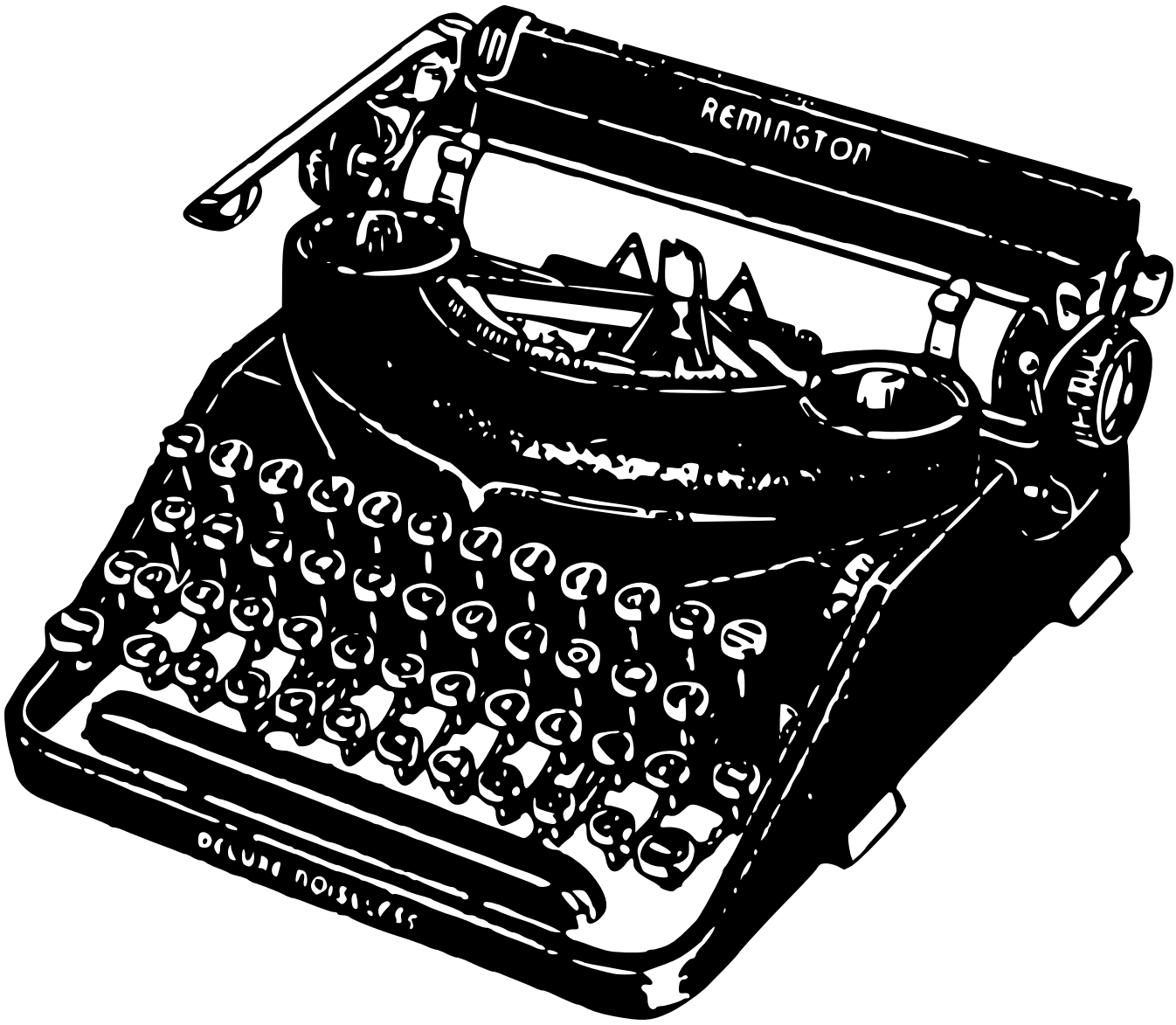
Também foi quando eu vi o site, extremamente futurista, do VIM , que logo eu decidi que deveria testar de verdade esse negócio!

Então, sem mais delongas, vamos a minha visão da coisa.

Primeiros contatos com o VIM

O primeiro contato com o VIM pode ser bem esquisito. Você pensa: Pra que que eu vou sair do meu super editor pra vir pra esse negócio, dentro do Terminal, cheio de atalhos que eu tenho que decorar???

A primeira vez que eu abri o VIM para editar um texto grande, me senti usando uma dessas:



Mas a primeira impressão não bastou pra mim.

Acabei descobrindo que estava colocando o carro na frente dos bois .

A tal da agilidade usando VIM

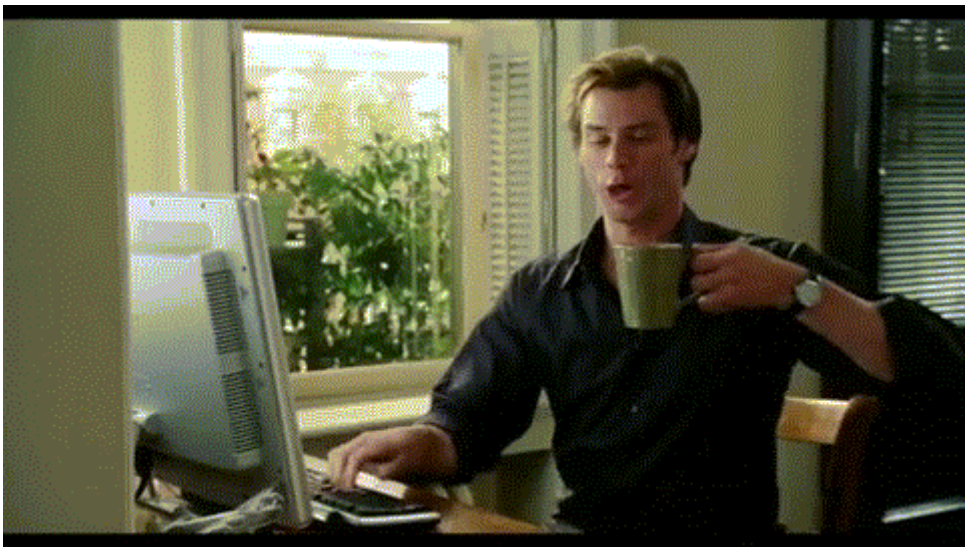
A primeira coisa que você precisa entender, quando começa a usar o VIM,

só vez. Você só precisa aprender os básicos e depois vai, gradualmente, aumentando seu nível de uso do editor e sua agilidade ao usar o mesmo.

É igual usar qualquer outro editor. Você não sai digitando extremamente rápido quando começa a usar uma IDE. Demora para você pegar todos os comandos, o mesmo funciona com os editores de texto e o VIM não é diferente disso.

Durante o aprendizado eu fui salvando os comandos que eu ia aprendendo em um repositório no meu GitHub, que acabou se tornando um GitBook chamado VIM para Noobs , onde você pode encontrar vários comandos, plugins úteis, temas e mais fonte de informação sobre o editor, mas não caia de cabeça nele agora. Continue a leitura.

Acho que o principal problema de quem começa com o VIM é a ideia de que, se outras pessoas dizem que escrevem texto super rápido no editor, logo no primeiro dia eu preciso escrever assim também.



Mas não é bem assim.

Se imagine aprendendo uma nova linguagem de programação, no primeiro dia você já sai dando aula? - Se você for um prodigio, não responda, por favor. ;P

Antes de começar a usar o VIM, saiba

O VIM trabalha com modos de operação. Onde, se você está em um modo ele possui determinados comandos e em outro modo os mesmos comandos podem fazer outra coisa. Os modos do VIM são os seguintes:

- **Modo normal:** quando você entra no VIM. Nesse modo você pode ler o texto, mas não vai conseguir inserir. Você consegue executar comandos de edição de texto, mas o VIM possui um modo somente para inserção. Quando você estiver em qualquer outro modo, você pode usar o `ESC` para voltar ao Normal Mode;
- **Modo de inserção:** quando você, de fato, vai inserir texto no arquivo. Você acessa esse modo pressionando `i`;
- **Modo visual: ** serve, principalmente, para seleção de grandes blocos de texto. Você acessa esse modo usando o `v`;
- **Modo de comandos (command mode): ** Onde você executa alguns comandos no VIM ou externos (como usar comandos do Git de dentro do VIM), consegue configurar o editor em tempo real, sair do editor, salvar arquivos, etc.

O modo em que você está fica marcado na parte de baixo do editor.

Ex.:

Ouça meu podcast

Alguns comandos do VIM aparecem nos tutoriais como letra maiúscula. Significa que você usa **`SHIFT` + `letra`** para executar o comando.

A maioria dos comandos é executado em Modo Normal.

Agora vamos conhecer os comandos que, realmente, vamos utilizar muito no começo.

Foque neles, nos primeiros dias, até aprender bem.

Abrir um arquivo

No Terminal, você pode cambiar até a pasta onde está o arquivo que você vai abrir e então basta digitar:

```
vim nome_do_arquivo
```

Mas você também pode usar o comando direto:

```
vim diretorio/outro_diretorio/nome_do_arquivo
```

Sintaxes

O VIM vai reconhecer a sintaxe automaticamente quando você abrir um arquivo com extensão específica, mas, se você começar a partir do VIM a

comando:

```
set syntax=sintaxe
```

Ex.:

```
set syntax=javascript
```

Para que o VIM passe a reconhecer a sintaxe que você está digitando.

Se movimentando dentro de um texto

Você pode usar as setinhas do teclado no começo, mas com o tempo vai aprender a usar mais as próprias letras, então você poderá usar:

- **h**: Esquerda
- **j**: Baixo
- **k**: Cima
- **l**: Direita

Isso no Modo Normal.

Uma maneira de se acostumar a usar os movimentos pelas letras é brincar

um pouco no VIM Adventures .

Existe, ainda, algumas coisinhas legais que você vai usar bastante:

- Para ir ao final de um arquivo use: `G`
- Para voltar ao topo: `gg`
- Para mover ao **final** de uma palavra: `e`
- Para mover até o **começo** da próxima palavra: `w`
- Para mover ao próximo caractere específico use: `f[caractere]`. Ex.: `f{` para ir ao próximo `{`.
- Para mover ao começo de uma linha, use: `0`
- Para mover ao final de uma linha, use: `$`
- Para pular ao final de um parágrafo: `}`
- Para pular para o fechamento de um parentese/colchete/chaves use: `%`

Da para formar comandos com contadores para agilizar ainda mais as coisas.

Imagine que você precisa pular 10 linhas no texto, você pode usar `10j`.

Inserindo texto com VIM

Saber se movimentar vai servir para ler um texto ou para posicionar o cursor para executar um comando, mas como escrever um texto?

É bem simples, você pode pressionar o **i** para entrar no modo de inserção onde o cursor estiver posicionado e pode sair digitando a vontade.

Claro que, também, tem alguns comandinhos legais que você pode usar:

- **I** (maiúsculo) - Entra no modo de inserção no começo da linha
- **a** - Entra no modo de inserção na frente de onde o cursor se encontra
- **A** (maiúsculo) - Entra no modo de inserção no final da linha
- **o** - Adiciona uma linha abaixo de onde estiver o cursor e entra em modo de inserção
- **O** (maiúsculo) - Adiciona uma linha acima e entra em modo de inserção

Como apagar um texto

Você pode usar:

- **x** - Deleta o caractere onde o cursor estiver
- **X** - Deleta o caractere de trás do cursor
- **dw** - Deleta a partir do cursor até o começo da próxima palavra
- **d$** - Deleta do cursor até o fim da linha
- **dd** - Deleta a linha iteira, incluindo o [enter] que ouver no final
- **C** (maiúsculo) - Deleta de onde o cursor estiver até o final da linha e entra em modo de inserção a partir dali
- **ce** - Deleta do cursor até o final da palavra e entra em modo de inserção

Como copiar e colar ou recortar texto usando o VIM

Para copiar e colar você vai usar o comando **y**, para copiar, e o **p**, para colar.

Para facilitar a vida:

- Você pode usar **yy** para copiar uma linha inteira.
- Você pode entrar no modo visual (**v**), selecionar um texto e pressionar o **y**

Já para **recortar** um texto é, ainda, mais simples.

Toda vez que você deleta um texto usando **x**, **dd**, **dw**, etc, o VIM não joga fora de uma vez, ele armazena em cache e você pode usar o **p** para colar em outro lugar.

Ex.:

Você pode deletar uma linha inteira usando **dd** e depois colar em outro lugar usando o **p**.

Para **copiar para o Clipboard** e poder colar fora do VIM, você vai selecionar o texto usando o modo visual (**v**) e usar o comando **"+y**.

Para **colar a partir do Clipboard** você pode usar **"+p**.

E o mesmo esquema de montar comandos com contadores funciona para os de copiar e colar.

Ex.:

Copiar 10 linhas de um texto

```
10yy
```

Como desfazer alguma coisa

Quando fazemos algo errado, podemos usar o famoso CTRL+Z nos editores comuns, mas no VIM é um pouco diferente.

Você vai utilizar os comandos:

- **u** - Para desfaz a ultima alteração
- **U** - Para desfaz todas as mudanças efetuadas em uma linha inteira
- **CTRL+R** - Para refazer a ultima coisa que você desfez

Como localizar uma palavra dentro de um texto

Você pressionaria o CTRL+F em editores comuns, mas no VIM você vai usar somente RegEx!

No modo normal, você pode pressionar `/`, digitar o texto, pressionar enter e a busca vai acontecer.

Para pular de uma ocorrencia para outra você pode usar `n` para ir para a próxima e `N` para voltar.

Salvando ou saindo de um arquivo

Depois de editar o texto, conforme ensinado, como salvar as coisas agora?

É bem simples, você vai usar os comandos, no modo normal:

- **:q** - Sai sem salvar. Será solicitado confirmação se existirem alterações não salvas.
- **:q!** - Sai sem salvar descartando as alterações
- **:w** - Salva o arquivo (escreve)
- **:wq** - Salva e sai do arquivo

Se você quiser salvar o arquivo com outro nome, então pode utilizar o `:w novo_nome` e pronto!

Ainda se quiser salvar com um novo nome e sair do arquivo, pode usar:

```
:wq novo_nome
```

E se você não quiser ficar digitando `:wq`, pode ainda usar o `ZZ`, que também salva e sai do arquivo

História do VIM e mais comandos legais

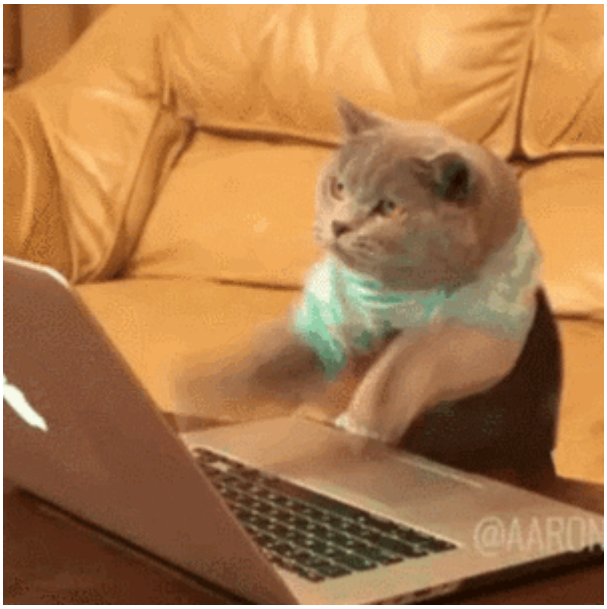
Se você quiser conhecer a história por trás dos comandos do VIM, da uma olhada nesse vídeo do Magnun Leno , no Grupylango.

Você vai entender o por que dos comandos específicos.

Ouça meu podcast

Grupylango - Vim - mais que um editor - com Magnun Le...

Dizem que, se usarmos o VIM por muito tempo, logo estaremos digitando extremamente rápido. Igual esse gatinho:



Plugins, temas, janelas, abas, sintaxes configurações especiais e magia negra

O VIM ainda possui MUUUUUITO mais coisas para se aprender.

O mais legal do VIM é que ele é extremamente "Hackeavel" (não que o Sublime também não seja). Você pode deixar o SEU VIM com sua cara, sua forma de digitar, suas manias, etc.

Isso você vai conseguir através de configurações no `.vimrc`, que é o arquivo de configuração do VIM, com plugins e até criando seus comandos e macros.

Como eu disse foque nos comandos principais na primeira semana, pelo menos, e depois vá se aprofundando, instalando plugins, criando comandos, etc.

Mas as primeiras configurações, como mudar um tema, você pode fazer logo de começo.

Você vai encontrar temas legais nesse site e, normalmente, a forma de instalar o tema, consta na documentação dos mesmos.

O esquema de cores que eu curto é esse aqui , parecido com o Monokai do Sublime.

Se quiser, pode até criar seu próprio tema rapidamente aqui .

Tudo isso e um pouco mais, você encontra no VIM para Noobs , depois de passar uma semana com o VIM, da uma olhada lá para aprender um pouco mais. :D

Ouça meu podcast

E ai, curtiu conhecer o VIM?

Eu ainda não utilizo ele 100%. O Sublime ainda é meu editor principal, mas, aos poucos, eu estou migrando para o VIM.

Logo logo postarei mais sobre minhas experiências com esse negócio!

Se curtiu esse artigo, compartilhe com os amigos nas Redes Sociais, espalhe nos grupos, mostra pro seu Gatinho.

Espalhe a palavra.

Tags: produtividade

William Oliveira

Inclusão social através do ensino de programação e front-end