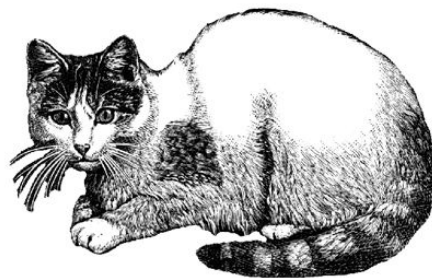


Living in the Shell

#2 - Editores de Texto

Why Vim?

Graphical Environment not required



Why Vim?

Porque VIM parar aqui

Porque Falar sobre Editores de Texto?

De acordo com a filosofia do Unix de:

"Faça programas que façam apenas uma coisa bem feita, que consigam trabalhar em conjunto um com os outros e que usem texto como interface de troca de dados".

Na primeira aula desta série, aprendemos como usar programas que trabalham em conjunto através de pipes, agora temos q aprender a trabalhar bem com a interface universal que é o texto.

Porém minha visão eh que a maioria das pessoas não têm compreensão de como usar corretamente os editores de texto no linux, principalmente o VIM.

Quais são os tipos de editores de texto?

Existem dois paradigmas de edição de texto no mundo.

- **Editores Modais (Vim, ed):** Eles permitem a edição de textos através do uso de COMANDOS sobre o texto, e o editor tem vários MODOS de operação, e na qual a EDIÇÃO do texto é apenas um desses modos. Esse tipo de editor é o mais antigo dos editores e antecede a existência de telas para a edição de textos e programas. Por serem muito antigos, geralmente funcionam sem a necessidade de teclas especiais do teclado e/ou periféricos como o Mouse.
- **Editores Nao Modais (Emacs, Notepad, Nano):** Os Editores não-modais apenas tem um modo de execução: EDIÇÃO, e para executar comandos sobre todo o texto com salvar, procurar, substituir, copiar, colar, exige que se use outro periférico como o Mouse, ou teclas especiais.

Editores Modais (Vim, Ed)

Como falado anteriormente, os editores modais tem varios modos de operacao, no caso do Vim, temos os seguintes modos:

Vantagem:

- Alta expressividade e produtividade, uma vez que com os comandos podemos fazer grandes alterações com poucos toques no teclado, muitas vezes dispensando periféricos como o Mouse.
- Comandos podem ser facilmente definidos como macros e repetidos N vezes de uma só vez, fazendo em segundos,

Desvantagem:

- Necessidade de aprender um paradigma não natural, e de se praticar uma série de comandos que geralmente são apenas algumas poucos toques no teclado, e de forma nao visual (sem cliques de mouse ou seleção de menus).

Editores Modais (Vim, Ed)



Editores Modais
antecedem os editores
visuais, até porque foram
criados numa época em
que não havia monitores.

Na foto ao lado vemos
Ken Thompson, e Dennis
Ritchie, que inventaram a
Linguagem C e
co-criaram o Unix

Editores Nao Modais (Emacs, Notepad, Nano)

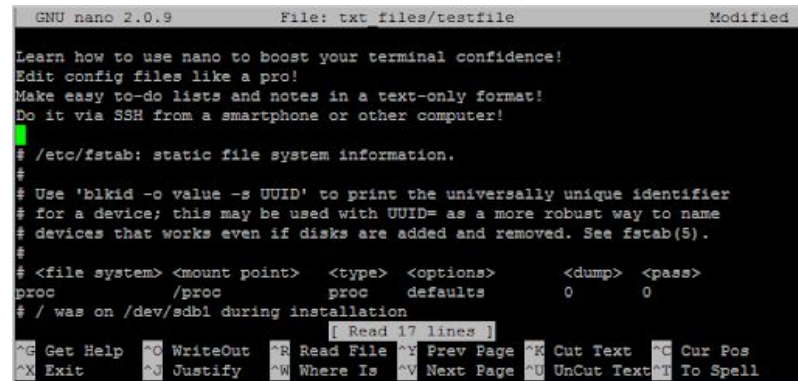
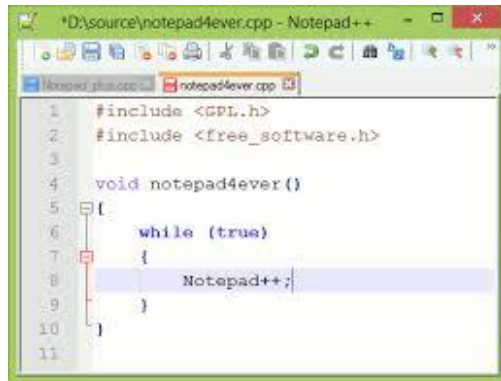
Os Editores Não Modais têm apenas um modo de operação que é o da edição de texto, logo

Vantagem:

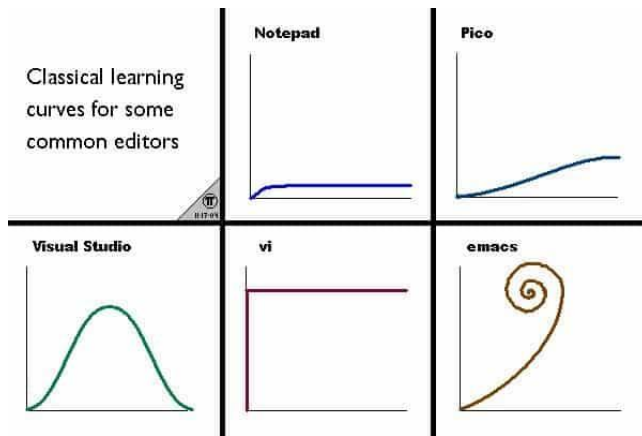
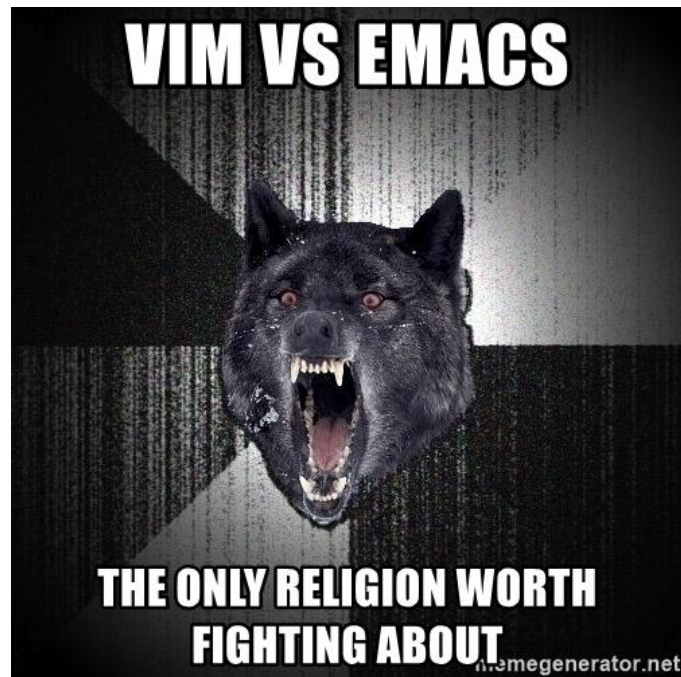
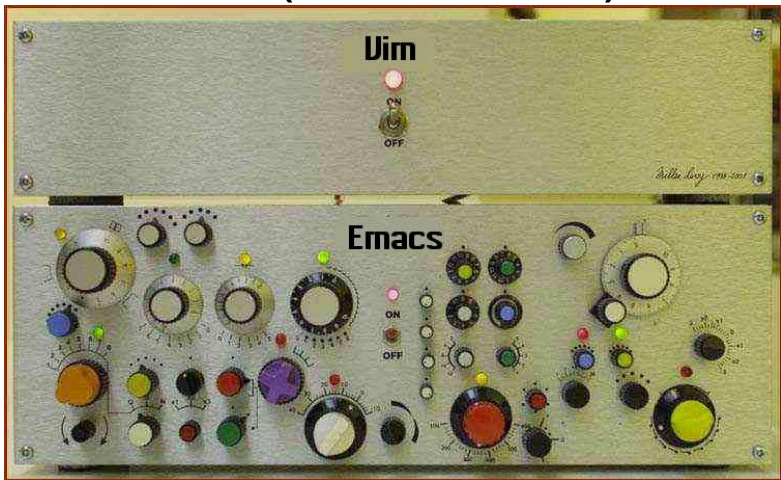
- São de fácil aprendizado, fáceis de implementar e leves, exige muito pouco treinamento
- Tem uma interface amigável e muitas vezes até menus para mostrar as operações a serem executadas de forma visual

Desvantagem:

- Como não tem o conceito de comandos, algumas tarefas mais complexas ou repetitivas podem ser muito trabalhosas.
 - *Por exemplo: Colocar um caractere de ; no final de cada linha, num editor de texto modal pode ser definido como um comando e rodado N vezes, enquanto que num editor Nao-Modal, voce tem q visitar todas as linhas e fazer a inserção manualmente.*



Emacs (NaoModal) vs Vim (Modais) memes.



Porque Aprender Vim?

- Existe em qualquer lugar:
 - *Desde IoT (Geladeiras, Carros) ate Mainframes da Decada de 80.*
- Editor Padrao do Unix
 - *(Vem em qualquer distribuição de Unix / Linux).*
- Nao precisa de plugins
 - *(80% da funcionalidade já vem compilado dentro dele).*
- Extremamente Leve
 - *(Ocupa em torno de 2-3 Mb de RAM, em contraste a centenas de Mb como o VSCode e outros).*

Demonstração dos modos do Vim:

Um dos maiores problemas que ocorre com os usuários comuns quando usam o VIM, é não entender que ele possui diversos modos de operação, com comandos distintos:

- **Modo Normal:** Nesse modo, mandamos comandos para o VIM para ele executar sobre o texto, em geral, linha a linha
- **Modo Edição / Inserção:** Permite inserir / remover caracteres.
- **Modo Visual:** Permite a Seleção de texto
- **Modo Comandos:** Permite a execução de comandos sobre o arquivo como um todo.

**NO MATTER HOW MUCH
EFFORT YOU PUT IN**



**IF YOU USE
THE WRONG
TOOLS, YOU WON'T
MAKE IT**

VIM é uma Linguagem:

Apesar do Vim ser um programa, ele na realidade eh um padrão de como definir e/ou usar comandos para edição de texto.

Dessa forma, praticamente qualquer outro editor de texto e/ou IDE tem plugins que oferecem a funcionalidade do VIM mesmo sem ter ele instalado, e até em sistemas não-unix.

A linguagem de comandos do VIM é uma linguagem composta por:

- **Verbos:** O que o Comando deve fazer?
- **Repetidores:** Quantas vezes?
- **Substantivos de Movimentacao:** Para aonde?
- **Substantivos de Objetos:** Aonde?

VIM - Movimentacao

- Movimentacao Basica
 - *(h,j,k,l)*
- Movimentação na Linha
 - *(0,\$,f,F,t,T)*
- Movimentação por palavra
 - *(w,W,b,B)*
- Movimentacao Relativa
 - *(<Mult><movimentacao>)*
 - *20j 10k*
- Movimentacao Arquivo
 - *(G, gg)*



Exemplos de Linguagem:

dw

apague (d) a palavra atual (w)

d3w

apague (d) 3 palavras (w) para a frente

ci{

mudar (apagar e colocar no modo de insercao), todo o conteúdo do bloco entre {} no cursor.

yggGp -

copiar (y) tudo do cursor ate o inicio do texto (G), ir para o final do texto (gg) e colar (p)

VIM - Verbos

- Deletar (d)
- Copiar (y)
- Colar (p, P)

- Adicionar Texto (a, A)
- Inserir (i, o, O)
- Mudar (c)

Exemplos de Linguagem - Verbos + Movimentação

- CW
(Muda a Palavra atual)
- d\$
(Apaga tudo ate o final da linha)
- c\$
(Muda tudo ate o final da linha)
- ci{
(Muda o texto entre {})
- ci"
(Muda o texto entre "")
- cit
(Muda o texto entre <tag></tag>)
- dt(
(Apaga tudo ate ()



Exemplos de Linguagem - Macros

O Poder da linguagem do VIM, como qualquer linguagem é a capacidade de se criar comandos complexos a partir de comandos simples, para isso temos os macros que são super poderosos e práticos no vim, por exemplo, se queremos adicionar um comentário ao final de cada linha de texto, podemos fazer assim:

1. `qa` - Inicia a Macro
2. `A` - Para iniciar a adição no final da linha
3. `// Comentario` - Que eh o texto que queremos adicionar
4. `ESC` - Para terminar a edicao
5. `k0` - para ir para a linha de baixo e início da linha
6. `q` - Para terminar a gravacao da macro

Para rodar a macro, apenas digitamos: `@a`

Se quisermos fazer em 20 linhas, digitamos: `20@a`



Dica do Bencke:

Caso você queira apenas repetir um comando N vezes, apenas pressione `.` (ponto), então, você quer colar +20 vezes uma linhas, apenas pressione `p` uma vez para colar e `20.` (dois, zero e ponto) para colar 20 vezes a mesma linha.

Comandos

Para comandos que afetam o arquivo como um todo, podemos usar os comandos, eles sempre iniciam com o caracter “:”

- Salvar, manipular Arquivos / Sair do vim
(*:w, :wa, :wq, :saveas*)
- Mover entre arquivos
(*:ls :b*)
- Interacao com o programas externos / shell
:read !date
:!npm run fix
- Mover os diretorios
:cd
:pwd
- Comandos sobre todo o texto
:retab
:%s/Bradesco/BRADESCO/gc

VimScript, VimRC e Plugins

No Coração da experiência com o VIM está o VimRC que eh o arquivo de configurações pessoais do VIM, nesse arquivo estão:

- 1) **Configurações gerais de uso**
- 2) **Plugins e suas Configuracoes**
- 3) **Mapeamento Personalizado do Teclado**
- 4) **Funções pessoais para uso no VIM**

Todo Programador deve ter seu repositório de “DotFiles”, aonde ele guarda os arquivos de configuração de seus programas Unix de forma a ter eles em qualquer máquina e ter eles sobre controle de versão, por exemplo, meus dotfiles estão em:

<https://github.com/gbencke/dotfiles>

e os VIMRC em:

<https://github.com/gbencke/dotfiles/tree/master/vim>

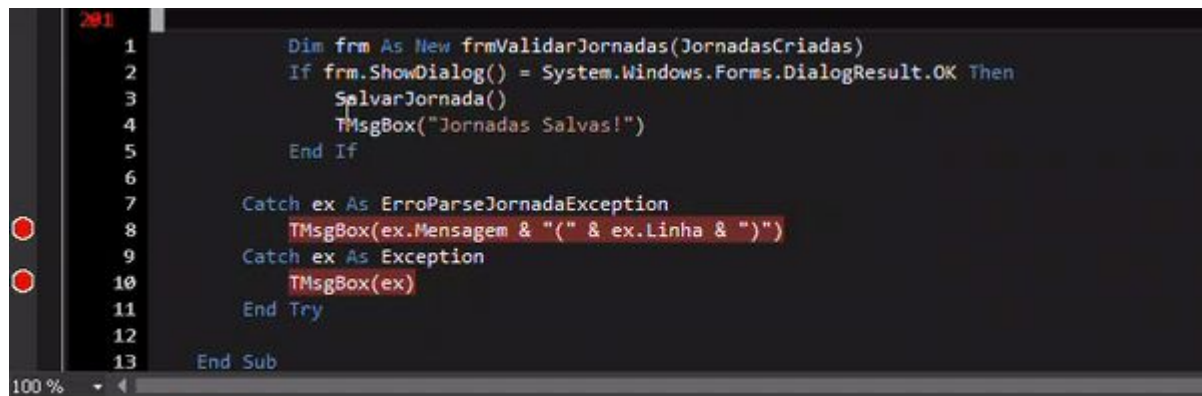
```
23 function! GetNERDTreeRoot()
22     if !empty($NERD_TREE_ROOT)
21         return $NERD_TREE_ROOT
20     else
19         return "~/git"
18     endif
17 endfunction
16
15 function CallNERDTree()
14     let dir_nerd = GetNERDTreeRoot()
13     echo dir_nerd
12     exe 'NERDTreeToggle' dir_nerd
11 endfunction
10
9 noremap <F2> :ALELint<CR>:echo "ALELint!Ok!"<CR>
8 noremap <F3> :ALEFix<CR>:echo "ALEFix!Ok!"<CR>
7 noremap <F4> :ALEFindReferences<CR>:echo "ALEFindReferences!Ok!"<CR>
6 noremap <F5> :ALEGoToDefinition<CR>:echo "ALEGoToDefinition!Ok!"<CR>
5 noremap <F6> :ALEHover<CR>:echo "ALEHover!Ok!"<CR>
4 noremap <F9> :Startify<CR>
3 noremap <F10> :call CallNERDTree()<CR>:echo "NERDTree!Ok!"<CR>
2 noremap <F11> :tab ball<CR>:echo "Tab Ball!Ok!"<CR>
1 noremap <F12> :ALEInfo<CR>:echo "ALEInfo!Ok!"<CR>
65
1 hi Visual term=reverse cterm=reverse guibg=Grey
2
```

Aonde usar Vim e sua Linguagem

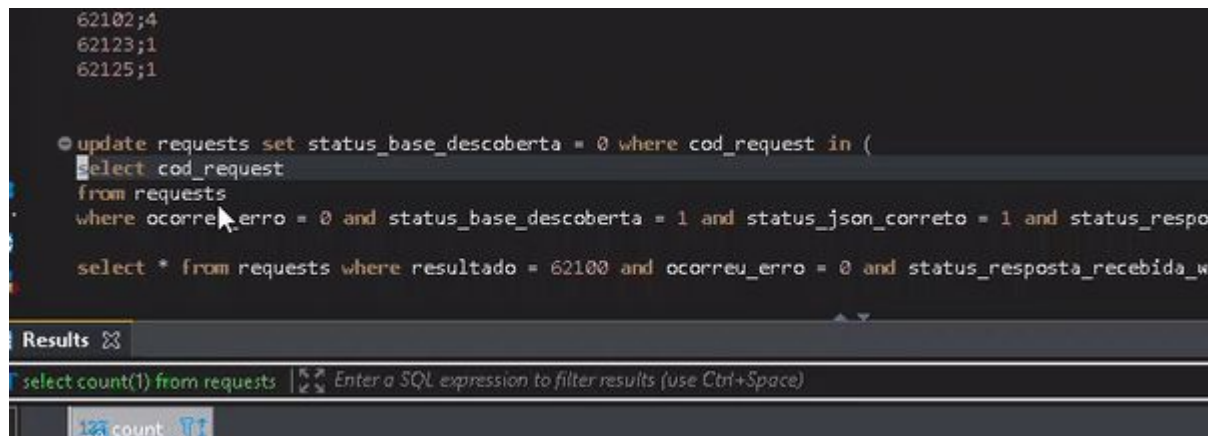
Apesar do VIM ser um programa, a sua linguagem e funcionalidade é amplamente disponível como um plugin em diversas ferramentas e IDEs como:

- **Visual Studio**
 - (Mostrado ao Lado)
- **Eclipse / dBeaver**
 - (Mostrado ao Lado)
- **VSCode**
- **IDEA**
- **Netbeans**

E praticamente qualquer IDE moderna



```
1 Dim frm As New frmValidarJornadas(JornadasCriadas)
2 If frm.ShowDialog() = System.Windows.Forms.DialogResult.OK Then
3     SalvarJornada()
4     MsgBox("Jornadas Salvas!")
5 End If
6
7 Catch ex As ErroParseJornadaException
8     MsgBox(ex.Mensagem & "(" & ex.Linha & ")")
9 Catch ex As Exception
10    MsgBox(ex)
11 End Try
12
13 End Sub
```



```
62102;4
62123;1
62125;1

update requests set status_base_descoberta = 0 where cod_request in (
select cod_request
from requests
where ocorreu_erro = 0 and status_base_descoberta = 1 and status_json_correto = 1 and status_respo

select * from requests where resultado = 62100 and ocorreu_erro = 0 and status_resposta_recebida_w

Results
select count(1) from requests Enter a SQL expression to filter results (use Ctrl+Space)
count
```

O ALUNO LENTO - <https://blog.sanctum.geek.nz/vim-koans/>

Mestre Wq estava almoçando quando um estudante irrompeu em seu quarto e se ajoelhou a seus pés. Lágrimas estavam em seus olhos e ele parecia profundamente frustrado.

Mestre Wq baixou sua tigela e perguntou:

"O que o perturba tanto, meu jovem?"

“Mestre,” ele disse. "Desisto. Eu nunca alcançarei o domínio do Vim! Jamais aprenderei os costumes dos grandes patriarcas! Eu nunca alcançarei a simplicidade brutal, o vazio divino do uso perfeitamente eficiente do Vim! ”

"Por que você diz isso?"

“Eu sou seu pior aluno, de longe. Quando estou tendo dificuldades para escrever uma macro simples, meus colegas alunos estão escrevendo macros recursivas com facilidade. Quando estou tentando lembrar a expressão regular para caracteres de espaço em branco, meus colegas alunos estão escrevendo testes de complexidade ciclomática em Vimscript. Sou muito lento, estou com vergonha e tenho medo de ter falhado. ”

O ALUNO LENTO - <https://blog.sanctum.geek.nz/vim-koans/>

Mestre Wq se levantou. “Venha comigo até a janela”, disse ele.

O aluno se levantou e seguiu Mestre Wq até a janela e olhou para o outro lado da rua, para a casa do vizinho de Mestre Wq. Pela janela, os dois puderam ver um jovem de terno e gravata, trabalhando em um documento.

"O que você vê?" perguntou Mestre Wq. O aluno observou por um tempo.

“Aquele jovem está usando o Microsoft Excel para gerar uma planilha. Ele está atualizando cada célula manualmente. Ele nem sabe usar fórmulas. Ele forma letras maiúsculas pressionando Caps Lock e, em seguida, pressionando-o novamente quando terminar. Ele é tão lento! Eu não entendo. Como ele pode estar tão contente? ”

“Vendo este jovem, como você pode não estar?” retornou Master Wq.

O aluno foi imediatamente esclarecido. Seu nome era Qa, e mais tarde ele se tornou um dos grandes mestres.

Conclusão

Editores Modais tem uma curva de aprendizado bastante acentuada e é normal levar 3-4 anos para se aprender a trabalhar bem com eles

Mas, a mente de um programador é um programa, então tendo um editor de texto que aceita comandos como apenas mais um programa e permite gerar comandos complexos a partir de simples, gera prazer e naturalidade a ele.

Além é claro, que qualquer programador deve customizar as suas ferramentas de forma a caber e a ser harmonioso com as suas preferências pessoais, como o Jedi que deve no final do seu treinamento criar seu próprio Sabre de Luz

E é nesse contexto que o Vim brilha, e continuara brilhando em suas diversas versões, por mais 50 anos.