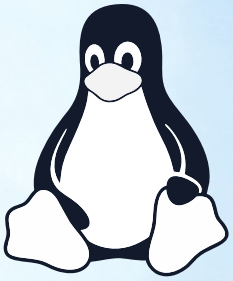
COLEÇÃO ESPECIAL DE 20 ANOS DA UTAH

E-BOOKS - PREPARATÓRIO PARA CERTIFICAÇÃO

Linux



Linux Professional Institute

LPIC-1

LINUX ADMINISTRATOR

E-Book 01

VIM - EDITOR DE ARQUIVO

Caminhe com
os gingantes

UTAH TREINAMENTOS

grupoutah grupoutah utah_networxs Grupo Utah

www.utah.com | cursos@utah.com.br

Sumário

Peso	3
Descrição	Candidatos devem ser capazes de editar arquivos de texto utilizando o vi.
Áreas de conhecimento	▫ Navegar em um documento de texto utilizando o vi. ▫ Utilizar os modos básicos do vi. ▫ Inserir, editar, excluir, copiar e pesquisar texto.
Itens relacionados	▫ vi ▫ / e ? ▫ h, j, k e l ▫ j, o e a ▫ c, d, p, y, dd e yy ▫ ZZ, :w!, :q!, e :e!

Sobre o editor Vi

O editor vi é mais que certamente o editor padrão na maioria dos sistemas Linux atuais. De fato, se um sistema possui apenas um editor, então provavelmente é o vi, e não é por acaso que existe um objetivo só para ele definido pela LPI (e não para o emacs).

Se criarmos uma lista com os 10 comandos e utilitários que qualquer administrador Linux deve obrigatoriamente conhecer, o vi com certeza estaria entre os primeiros, pois editar arquivos de texto (configuração, scripts, listagens, etc) faz parte do dia-a-dia de um administrador de sistemas Linux.

Iniciando o Vi

Muitas distribuições Linux atuais já disponibilizam uma versão melhorada do vi original, o vim (de Vi Improved). O vim é retroativamente compatível com o vi e algumas vezes também possui um modo gráfico (gvim) disponível. Em alguns sistemas que possuem o vim instalado, o comando vi é na verdade um alias para o comando vim.

A seguir, temos a tela inicial do editor vim:

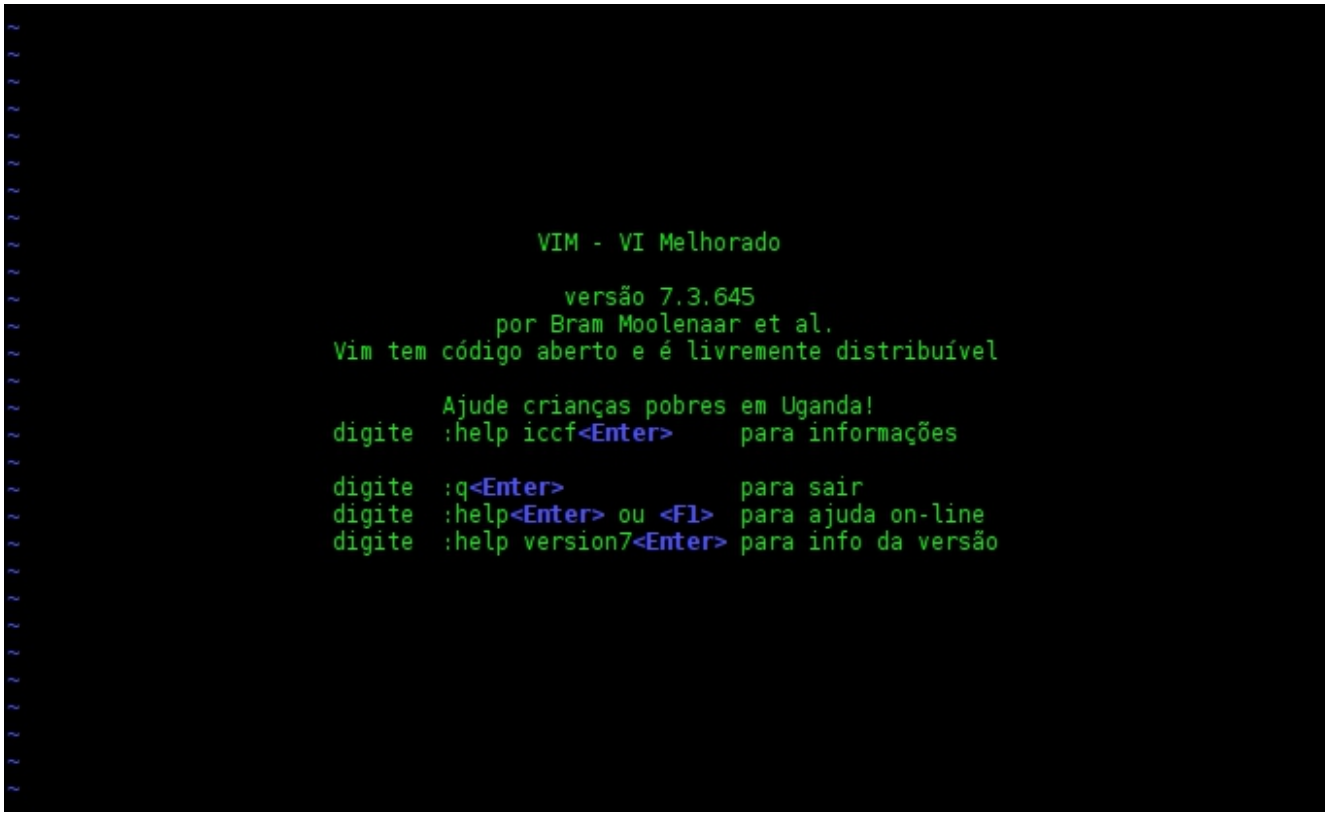


Imagem 01.

Vamos utilizar o vi para editar o script /home/aluno/count1.sh:

$ vi /home/aluno/count1.sh

Você deverá visualizar uma tela semelhante a esta:

```
x="$1"
echo "$2" $(date)
while [ $x -gt 0 ]; do let x=$x-1;done
echo "$2" $(date)
~
~
~
~
~
~
"count1.sh" 4L, 82C
```

Modos do Vi

O editor vi possui dois modos de operação:

Modo de comandos	Neste modo é possível utilizar as teclas para movimentar-se pelo arquivo e executar operações básicas de edição e pesquisa de texto. É neste modo que o vi inicia quando abrimos um arquivo.
Modo de inserção (edição)	No modo de inserção, as teclas que digitamos são interpretadas como caracteres e inseridas no arquivo de texto (daí o nome do modo). Existem diversas maneiras de entrarmos no modo de inserção (que veremos a seguir), mas para sair deste modo e voltar ao modo inicial de comandos utilizamos a tecla [Esc].

Estes dois modos alternam completamente o comportamento do editor. O vi foi criado numa época em que nem todos os teclados possuíam tantas teclas como conhecemos hoje, então tudo que fazemos no vi geralmente é feito utilizando as teclas principais do teclado e mais um conjunto minimalista de teclas como o [Esc] ou [Insert], e isso faz com que o vi tenha uma má fama de não ser intuitivo e difícil de usar.

Saindo do Vi

Uma das coisas mais importantes a se aprender em um editor de textos (principalmente o vi) é como sair dele antes que façamos algo que não deveríamos de feito. Existem diversas maneiras de sair do vi, dentre as quais podemos

:q!	Fecha imediatamente o vi e ignora todas as alterações não salvas.
:w!	Salva o arquivo (modificado ou não). É possível especificar um nome de arquivo como parâmetro, indicando ao vi que deve salvar as alterações no arquivo especificado (e não no arquivo atual). É geralmente mais seguro omitir o sinal de !, a não ser que tenha certeza do que está fazendo, pois a ! no final de um comando do vi força a operação sem solicitar confirmação do usuário.
ZZ	Salva o arquivo que tenha sido modificado e então fecha o editor.
:e!	Edita a cópia atual do arquivo no disco. É como recarregar o arquivo, abandonando as alterações feitas.
:!	Executa um comando do shell.

Nota
Quando digitamos :, o cursor irá mover para o canto inferior esquerdo da tela, onde será possível digitar os comandos.

Movimentando-se

Os seguintes comandos nos ajudam a movimentar o cursor pelo arquivo:

h	Desloca o cursor para a esquerda.
j	Desloca o cursor para baixo.
k	Desloca o cursor para cima.
l	Desloca o cursor para a direita.
0	Posiciona o cursor no início da linha atual.
$	Posiciona o cursor no final da linha atual.
+	Move o cursor para o primeiro caractere da próxima linha
-	Move o cursor para o primeiro caractere da linha anterior.
w	Move o cursor para o primeiro caractere da próxima palavra.
W	O mesmo que o comando w, mas não diferencia caracteres de acentuação.
b	Move o cursor para o primeiro caractere da palavra atual ou anterior mais próxima.
e	Move o cursor para o último caractere da palavra mais próxima.
{ou}	Move o cursor para o início ou final do parágrafo.
H,MeL	Move o cursor para o primeiro caractere visível no topo, meio e rodapé da tela.
G	Move o cursor para a última linha do arquivo.
nG	Posiciona o cursor no início da linha n . Por exemplo, o comando 1G posicionaria o cursor na primeira linha do arquivo.

Os comandos do vi podem ser quantificados, o que significa que se você digitar um número antes dos comandos, isso fará com que o vi repita o comando pelo número de vezes especificado. Observe alguns exemplos:

- 3w move o cursor 3 palavras adiante.
- 2k move o cursos 2 linhas acima.
- 12l move o cursor 12 caracteres a direita.
- 2+ move o cursor 2 linhas abaixo e o posiciona no inicio da linha.

> *Dica*
> *Tente familiarizar-se bastante com os comandos quantificados do vi antes de fazer a prova LPI101.*

Pesquisando

A possibilidade de pesquisar o conteúdo de arquivos de texto é tão importante quanto editá-los. As pesquisar no vi sempre levam em consideração a posição atual do cursor, e podem transcorrer em duas direções: da posição atual do cursor até o final do arquivo (de cima para baixo) ou da posição atual do cursor até o início do arquivo (de baixo para cima).
Os comandos de pesquisa são os seguintes:

/	Inicia a pesquisa da posição atual do cursor para baixo no arquivo. Observe que se o vi chegar ao final do arquivo e não encontrar nada ele irá continuar a pesquisa a partir do início do arquivo.
?	É o inverso de /, pois inicia a pesquisa a partir da posição atual do cursor para cima.
n	Repete a última pesquisa efetuada na mesma direção.
N	Repete a última pesquisa efetuada na direção oposta.

Assim que digitar um dos comandos de início de pesquisa (/ ou ?), o prompt de pesquisa será aberto no canto inferior esquerdo da tela. Basta digitar a string a ser pesquisada e pressionar **[Enter]**. A partir deste momento, o vi armazena a pesquisa efetuada na memória, de modo que podemos utilizar os comandos **n** ou **N** para repetir a pesquisa em ambos os sentidos, o que possibilita navegar pelos resultados.

Editando texto

Alterar o texto inclui ações como inserir, excluir ou alterar o texto. A maior parte das alterações maiores que fizermos utilizando o vi serão no modo de inserção, onde tudo que digitamos é inserido no arquivo a partir da posição do cursor, e teclas como **[Del]**, **[Enter]** e **[Backspace]** funcionam do mesmo modo que em qualquer editor de textos plano por aí. Vejamos alguns comandos que podemos utilizar para entrar no modo de inserção do vi:

i	Inicia o modo de edição na posição atual do cursor.
I	Inicia o modo de edição no início da linha atual.
a	Inicia o modo de edição no próximo caractere em relação a posição atual do cursor.
A	Inicia o modo de edição no final da linha atual.
o	Adiciona uma nova linha abaixo do cursor e inicia o modo de edição nela.
O	Adiciona uma nova linha acima do cursor e inicia o modo de edição nela.

Porém, para algumas edições menores existem comandos muito úteis que permitem editar partes menores do texto (como alterar um caractere) sem necessariamente ter que recorrer ao modo de inserção do vi:

x	Apaga o caractere sob o cursor.
dw	Exclui a palavra inteira a partir da posição atual do cursor.
dd	Exclui a linha atual inteira.
D	Exclui todo o conteúdo a partir do cursor até o final da linha.
cw	Exclui (recorta) a palavra atual e entra no modo de edição.
cc	Exclui a linha atual e abre o modo de edição.
C	Exclui da posição do cursor até o final da linha e abre o modo de edição.
yw	Copia a palavra para o buffer.
yy	Copia a linha inteira para o buffer.
Y	Copia da posição do cursor até o final da linha para o buffer.
p	Cola o conteúdo armazenado no buffer na posição atual do cursor.

Outros comandos úteis

O vim também traz diversos comandos úteis à utilização, ao gerenciamento e à automatização do arquivo:

Comando	Descrição
CTRL + R	Refaz as últimas ações dos comandos desfeitos.
.	Repete o comando anterior.
N	Vai para o próximo resultado da pesquisa.
:![comando]	Executa um comando de nome [comando] e exibe o resultado no arquivo simplesmente, não inserindo tal resultado como conteúdo dele.
:r![comando]	Executa um comando de nome [comando] e insere o resultado dele no arquivo atual.
:set nu	Insere uma numeração de identificação das linhas no arquivo. Esta numeração é apenas visual, não sendo inserida no arquivo original.
:set nonu	Desabilita a numeração no arquivo. Inicializa a ajuda do vim.
:help	Inicializa a ajuda do vim.
:syntax off	Habilita a Sintaxe de coloração de instruções.
:syntax on	Habilita a Sintaxe de coloração de instruções.

O arquivo /etc/vimrc

O vim possui um arquivo de configuração global onde podemos ativar diversas opções para o funcionamento do editor. Todas as vezes que os usuários abrirem o editor vim algumas opções como set nu ou syntax on podem vir ativadas.

O arquivo /etc/vimrc

As modificações feitas neste arquivo afetam todos os usuários porém é possível criar configurações específicas para cada usuário. Para isso basta copiar este arquivo para dentro do home do usuário de forma oculta usando o comando abaixo:

 # cp /etc/vimrc ~/.vimrc

Editar o arquivo ~/.vimrc e inserir as instruções especificas do VIM para o usuário.

> *Dica free:*
>
> *É possível copiar um conjunto de linhas dentro do vi usando o conceito de marca. Esta opção é muito útil quando necessitamos copiar varias linhas dentro do arquivo e colar em outro local do próprio arquivo.*

Ação:

Coloque o cursos no inicio da primeira linha que será o inicio do trecho a ser copiado e pressione as teclas:
mp
Onde:
m – informa que a partir daquela linha será criada uma marca.
p – Nome dado para a marca (poderia ser aqui qq letra)

Feito isso, mova o cursor na ultima linha do trecho que deseja copiar. Feito isso pressione:
y'p
Onde:
y – Instrução para copiar
' – Instrução para chamar a marca.
p – Nome da marca que foi usada anteriormente.
Feito isso você terá marcado o trecho de linhas corresponde, para colar basta ir com o cursor até o local que deseja colar o conteúdo e pressionar:
p
Caso queira deletar um trecho basta trocar a letra ***"y"*** acima pela letra ***"d"*** que tem a função de recortar.

Questões – LPI

1. No editor Vi, qual dos seguintes comandos copiará a linha atual no buffer vi?
Escolha uma:
a. yy
b. 1y
c. 1c
d. cc
e. c

2. Qual das seguintes sequências no editor vi salva o documento aberto e sai do editor?
Selecione três opções corretas.

Escolha uma ou mais:
a. esc :x
b. esc :ZZ
c. esc ZZ
d. esc wq
e. esc :wq

3. O que acontece depois de emitir o comando vi sem parâmetros adicionais?
Escolha uma:
a. vi sai com uma mensagem de erro, pois não pode ser invocado sem um nome de arquivo para operar.
b. vi inicia e abre um novo arquivo que é preenchido com o conteúdo do buffer vi se o buffer contiver texto.
c. vi inicia e requer que o usuário crie explicitamente carregue um arquivo existente.
d. vi inicia no modo de comando e abre um novo arquivo vazio.
e. vi inicia e carrega o último arquivo usado e move o cursor para a posição em que o vi estava quando ele saiu.

4. Qual comando copia uma linha no vim?

a) y
b) yy
c) pp
d) dd
e) p

5. Qual comando é utilizado para forçar a saída de um arquivo de texto sem salvar as mudança e sair?

a) :q
b) :w!
c) :wq!
d) :wq
e) :q!

6. Qual comando é utilizado para desativar a sintaxe (Syntax Highlight) de cores de linhas?
a) :set disablenu
b) :set disable c) :set nu
d) :set nonu
e) :syntax off