熟悉 VI 是学习 UNIX 系统的一个关口

vi 是 visual edit 的缩写

前言

文本编辑器是所有计算机系统中最常用的一种工具。UNIX 下的编辑器有 ex,sed 和 vi 等，其中，使用最为广泛的是 vi，而 vi 命令繁多，论坛里好像这方面的总结不多，以下稍做总结，以资共享！渴望更正和补充！

进入 vi 的命令

vi filename :打开或新建文件，并将光标置于第一行首
vi +n filename ： 打开文件，并将光标置于第 n 行首
vi + filename ： 打开文件，并将光标置于最后一行首
vi +/pattern filename：打开文件，并将光标置于第一个与 pattern 匹配的串处
vi -r filename ： 在上次正用 vi 编辑时发生系统崩溃，恢复 filename
vi filename....filename ： 打开多个文件，依次进行编辑

移动光标类命令

h ： 光标左移一个字符
l ： 光标右移一个字符
space：光标右移一个字符
Backspace：光标左移一个字符
k 或 Ctrl+p：光标上移一行
j 或 Ctrl+n ： 光标下移一行
Enter ： 光标下移一行
w 或 W ： 光标右移一个字至字首
b 或 B ： 光标左移一个字至字首
e 或 E ： 光标右移一个字至字尾
) ： 光标移至句尾
(： 光标移至句首
}：光标移至段落开头
{：光标移至段落结尾
nG：光标移至第 n 行首
n+：光标下移 n 行

n-：光标上移 n 行
n$：光标移至第 n 行尾
H ：光标移至屏幕顶行
M ：光标移至屏幕中间行
L ：光标移至屏幕最后行
0：（注意是数字零）光标移至当前行首
$：光标移至当前行尾

屏幕翻滚类命令

Ctrl+u：向文件首翻半屏
Ctrl+d：向文件尾翻半屏
Ctrl+f：向文件尾翻一屏
Ctrl＋b；向文件首翻一屏
nz：将第 n 行滚至屏幕顶部，不指定 n 时将当前行滚至屏幕顶部。

插入文本类命令

i ：在光标前
I ：在当前行首
a：光标后
A：在当前行尾
o：在当前行之下新开一行
O：在当前行之上新开一行
r：替换当前字符
R：替换当前字符及其后的字符，直至按 ESC 键
s：从当前光标位置处开始，以输入的文本替代指定数目的字符
S：删除指定数目的行，并以所输入文本代替之
ncw 或 nCW：修改指定数目的字
nCC：修改指定数目的行

删除命令

ndw 或 ndW：删除光标处开始及其后的 n-1 个字
do：删至行首
d$：删至行尾
ndd：删除当前行及其后 n-1 行

x 或 X：删除一个字符，x 删除光标后的，而 X 删除光标前的
Ctrl+u：删除输入方式下所输入的文本

搜索及替换命令

/pattern：从光标开始处向文件尾搜索 pattern
?pattern：从光标开始处向文件首搜索 pattern
n：在同一方向重复上一次搜索命令
N：在反方向上重复上一次搜索命令
：s/p1/p2/g：将当前行中所有 p1 均用 p2 替代
：n1,n2s/p1/p2/g：将第 n1 至 n2 行中所有 p1 均用 p2 替代
：g/p1/s//p2/g：将文件中所有 p1 均用 p2 替换

选项设置

all：列出所有选项设置情况
term：设置终端类型
ignorance：在搜索中忽略大小写
list：显示制表位(Ctrl+I)和行尾标志（$)
number：显示行号
report：显示由面向行的命令修改过的数目
terse：显示简短的警告信息
warn：在转到别的文件时若没保存当前文件则显示 NO write 信息
nomagic：允许在搜索模式中，使用前面不带“”的特殊字符
nowrapscan：禁止 vi 在搜索到达文件两端时，又从另一端开始
mesg：允许 vi 显示其他用户用 write 写到自己终端上的信息

保存退出命令

：n1,n2 co n3：将 n1 行到 n2 行之间的内容拷贝到第 n3 行下
：n1,n2 m n3：将 n1 行到 n2 行之间的内容移至到第 n3 行下
：n1,n2 d ：将 n1 行到 n2 行之间的内容删除
：w ：保存当前文件
：e filename：打开文件 filename 进行编辑
：x：保存当前文件并退出
：q：退出 vi
：q!：不保存文件并退出 vi

：!command：

执行 shell 命令 command
：n1,n2 w!command：将文件中 n1 行至 n2 行的内容作为 command 的输入并执行之，若不指定 n1，n2，则表示将整个文件内容作为 command 的输入
：r!command：将命令 command 的输出结果放到当前行

寄存器操作

"?nyy：将当前行及其下 n 行的内容保存到寄存器？中，其中?为一个字母，n 为一个数字
"?nyw：将当前行及其下 n 个字保存到寄存器？中，其中?为一个字母，n 为一个数字
"?nyl：将当前行及其下 n 个字符保存到寄存器？中，其中?为一个字母，n 为一个数字
"?p：取出寄存器？中的内容并将其放到光标位置处。这里？可以是一个字母，也可以是一个数字
ndd：将当前行及其下共 n 行文本删除，并将所删内容放到 1 号删除寄存器中。

进入 vi

vi test

离开 vi

:q! 离开 vi,并放弃刚在缓冲区内编辑的内容。
:wq 将缓冲区内的资料写入磁盘中，并离开 vi。
:ZZ 同 wq
同 wq
:w 将缓冲区内的资料写入磁盘中，但并不离开 vi。
:q 离开 vi,若文件被修改过，则会被要求确认是否放弃修改的内容，此指令可与: w 配合使用。

Vi 的操作模式

Vi 提供两种操作模式：
输入模式（insert mode）
指令模式（command mode）

当使用者进入 vi 后，既处于指令模式下，此刻键入任何字元皆被视为指令。

输入模式：

a(append) 游标之后加入资料。
A 该行之末加入资料
i (insert) 游标之前加入资料
I 该行之首加入资料
o (open) 新增一行与该行之下供输入资料
O 新增一行与该行之上供输入资料

指令模式：B 移动

移至该行第一个字符，若光标在该行第一字符则光标移至上一行第一字符。
b 由游标所在位置之前一个字串的第一个字元
cc 删除整行，修改整行的内容。
D 以行为单位，删除游标在内后面的所有字符。
db 删除该行光标前字符
dd 删除该行
de 删除自光标开始后面的字符
d 加字符 删除光标所在位置至字符之间的单
E 移至该行最后字符，若光标在该行最后字符则光标移至下一行最后字符
e 由游标所在位置至该字串的最后一个字元
G 移至该档案的最后一行
h 向前移一个字元
j 向下移一个字元
k 向上移一个字元
0 移至该行之首
M 移至视窗的中间那行
L 移至视窗的最后一行
l 向后移一个字符
0 由游标所在位置该行的第一个字元
nG 移至该档案的第 n 行
n+ 自游标所在位置向后移 n 行至该行的第一字符
n- 自游标所在位置向前移 n 行至该行的第一字符
R 进入取代状态，直到《ESC》为止
s 删除游标所在字元，并进入取代模式直到《ESC》
S 删除游标所在之该行资料，并进入输入模式直到《ESC》

w 由游标所在位置之下一个字串的第一个字元
x 删除游标所在该字元。
X 删除游标所在之前一字元。
r 用接于此指令之后的字元取代（replace）游标所在字元
yy yank 整行，使游标所在该行复制到记忆体缓冲区
<ctrl><g> 显示该行之行号、档案名称、档案中最末之行号、游标所在行号占

总行号之百分比

$ 由游标所在位置至该行的最后一个字元。
） 由游标所在位置至下一个句子的第一个字元。
（ 由游标所在位置至该句子的第一个字元。
{ 由游标所在位置至该段落的最后一个字元。
} 由游标所在位置至该段落的第一个字元

yank 和 delete 可将指定的资料复制到记忆体缓冲区，而藉有 put 指令可将缓冲区内的资料复制到荧幕上
例如：搬移一行 ：在该行执行 dd
游标移至目的地

执行 p

复制一行 ：在该行执行 yy
游标移至目的地
执行 p

视窗移动：

<ctrl><f> 视窗往下卷一页
<ctrl><b> 视窗往上卷一页
<ctrl><d> 视窗往下卷半页
<ctrl><u> 视窗往上卷半页
<ctrl><e> 视窗往下卷一行
<ctrl><y> 视窗往上卷一行
删除、复制及修改指令介绍：
d(delete)、c(change)和 y(yank)这一类的指令在 vi 中的指令格式为：
operation+scope=command
(运算子)（范围）
运算子：

d 删除指令。

删除资料，但会将删除资料复制到记忆体缓冲区。
y 将资料（字组、行列、句子或段落）复制到缓冲区。
p 放置（put）指令，与 d 和 y 配合使用。可将最后 delete 或 yank 的资料放置于游标所在位置之行列下。
c 修改（change）指令，类似 delete 于 insert 的组合。删除一个字组、句子等资料，并插入新键入的

该文章转载自网络大本营：http://www.xrss.cn/Info/15318.Html

set all 查看所有 set 选项
:set 显示当前 set 设置
:filetype on 通过$VIMRUNTIME/filetype.vim.打开文件类型检测
ai/noai 自动缩进，新行与前面的行保持—致的自动空格/不自动空格(缺省)
aw/noaw 自动写，转入 shell 或使用：n 编辑其他文件时，
当前的缓冲区被写入/不写
flash/noflash 在出错处闪烁但不鸣叫(缺省)/使用鸣叫而不闪烁
ic/noic 在查询及模式匹配时忽略大小写/不忽略大小写(缺省)
lisp/nolist 在编辑 lisp 程序时设置自动空格以便排列文本/按前一行自动设置空格
magic/nomagic 在进行模式匹配时使用全部的特殊宁符/将特殊宁符只限制于"^”和“$”
mesg/nomesg 允许/不允许其他用户向终端上写东西
nu/nonu 屏幕左边显示行号/不显示行号(缺省)
ro/noro 只读，除非明确设置为:w，否则不允许对文件改动/普通读／写模式
remap/noremap 允许将宏直接映射到已经编辑好的命令行中(缺省)／求宏定义明确
showmatch 显示括号配对，当键入“]”“)”时，
高亮度显示匹配的括号／缺省不高亮
showmode 处于文本输入方式时加亮按钮条中的模式指示器／缺省不指示当前模式
ts=n 将 TAB 键的跨度设置为 n 个宁符间距，缺省为 8
warn/nowarn 对文本进行了新的修改后，离开 shell 时系统给出显示(缺省)
ws/nows 在搜索时如到达文件尾则绕回文件头继续搜索
wrap/nowrap 长行显示自动折行
wm=n 保留空边，n 为显示的最右边留出的空白边的字符数
si smart indent 括号自动对齐
fe=prc 设置汉字整字处理
augroup 显示已存在 auto 命令组
nobackup 取消自动备份
encoding=prc 设置汉字处理，否则 backspace 删除半个汉字
cindent 设置 C 格式缩进，缩进量是通过 shiftwidth 的值，
而不是通过 tabstop 的值
cino=string string 定义了特殊需求的 cindent 行为，
参看:h cinoptions-values :h cinkeys 等

Linux 如何退出 VI 编辑器 2008-10-22 09:55:q 退出
:q!强行退出（不存盘）
:qw or ZZ 存盘退出
用 ESC 键只能切换到命令状态

更加详细内容：

在 linux 底下最常使用的文书编辑器为 vi ，请问如何进入编辑模式？
在一般模式底下输入： i, I, a, A 为在本行当中输入新字符；（出现 －Insert- ）
在一般模式当中输入： o, O 为在一个新的一行输入新字符；
在一般模式当中输入： r, R 为取代字符！（左下角出现 －Replace-）
如何由编辑模式跳回一般模式？
[Esc]
若上下左右键无法使用时，请问如何在一般模式移动光标？
h, j, k, l
若 [pagedown] [pageup] 在一般模式无法使用时，如何往前或往后翻一页？
[Ctrl] + [f]
[Ctrl] + [b]
如何到本档案的最后一行、第一行；本行的第一个字符、最后一个字符？
G, 1G, 0, $
如何删除一行、n 行；如何删除一个字符？
dd, ndd, x 或 X （dG 及 d1G 分别表示删除到页首及页尾）
如何复制一行、n 行并加以贴上？
yy, nyy, p 或 P
如何搜寻 string 这个字符串？
?string (往前搜寻)
/string (往后搜寻)
如何取代 word1 成为 word2，而若需要使用者确认机制，又该如何？
:1,$s/word1/word2/g 或
:1,$s/word1/word2/gc （需要使用者确认）
如何读取一个档案 filename 进来目前这个档案？
:r filename
如何另存新档成为 newfilename？
:w newfilename
如何存盘、离开、存盘后离开、强制存盘后离开？
:w； :q； :wq； :wq!
如何设定与取消行号？
:set nu
:set nonu
关于文本编辑 VI 的问题

(一）进入 vi

在系统提示字符(如$、#)下敲入 vi ＜档案名称＞，vi 可以自动帮你载入所要编辑的文件或是开启一个新

文件（如果该文件不存在或缺少文件名）。进入 vi 后萤幕左方会出现波浪符号，凡是列首有该符号就代

表此列目前是空的。

(二)、两种模式

如上所述，vi 存在两种模式：指令模式和输入模式。在指令模式下输入的按键将做为指令来处理：如输入

a，vi 即认为是在当前位置插入字符。而在输入模式下，vi 则把输入的按键当作插入的字符来处理。指令

模式切换到输入模式只需键入相应的输入命令即可（如 a,A），而要从输入模式切换到指令模式，则需在

输入模式下键入 ESC 键，如果不晓得现在是处於什麼模式，可以多按几次 [ESC]，系统如发出哔哔声就表

示已处于指令模式下了。

付：有指令模式进入输入模式的指令：

新增 (append)

a ：从光标所在位置後面开始新增资料，光标後的资料随新增资料向後移动。

A： 从光标所在列最後面的地方开始新增资料。

插入 (insert)

i： 从光标所在位置前面开始插入资料，光标後的资料随新增资料向後移动。

I ：从光标所在列的第一个非空白字元前面开始插入资料。

开始 (open)

o ：在光标所在列下新增一列并进入输入模式。

O: 在光标所在列上方新增一列并进入输入模式。

（三）、退出 vi

在指令模式下键入:q,:q!,:wq 或(注意:号)，就会退出 vi。其中:wq 和是存盘退出，而:q 是直接退出，

如果文件已有新的变化，vi 会提示你保存文件而:q 命令也会失效，这时你可以用:w 命令保存文件后再用:q

退出，或用:wq 或命令退出，如果你不想保存改变后的文件，你就需要用:q!命令，这个命令将不保存文件

而直接退出 vi。

（四）、基本编辑

配合一般键盘上的功能键，像是方向键、[Insert] 、[Delete] 等等，现在你应该已经可以利用 vi 来编辑文件

了。当然 vi 还提供其他许许多多功能让文字的处理更为方便。

何谓编辑？一般认为是文字的新增、修改以及删除，甚至包括文字区块的搬移、复制等等。先这里介绍 vi

的如何做删除与修改。（注意：在 vi 的原始观念里，输入跟编辑是两码子事。编辑是在指令模式下操作

的，先利用指令移动光标来定位要进行编辑的地方，然後才下指令做编辑。）

删除与修改文件的命令：

x： 删除光标所在字符。

dd ：删除光标所在的列。

r ：修改光标所在字元，r 後接著要修正的字符。

R： 进入取替换状态，新增文字会覆盖原先文字，直到按 [ESC] 回到指令模式下为止。

s： 删除光标所在字元，并进入输入模式。

S： 删除光标所在的列，并进入输入模式。

其实呢，在 PC 上根本没有这麼麻烦！输入跟编辑都可以在输入模式下完成。例如要删除字

元，直接按
[Delete] 不就得了。而插入状态与取代状态可以直接用 [Insert] 切换，犯不著用什麼指令模式的编
辑指令。不过就如前面所提到的，这些指令几乎是每台终端机都能用，而不是仅仅在 PC 上。在指令模式下移动光标的基本指令是 h, j, k, l 。想来各位现在也应该能猜到只要直接用 PC 的方向
键就可以了，而且无论在指令模式或输入模式下都可以。多容易不是。
当然 PC 键盘也有不足之处。有个很好用的指令 u 可以恢复被删除的文字，而 U 指令则可以恢复光标所
在列的所有改变。这与某些电脑上的 [Undo] 按键功能相同。
三、附件：vi 详细指令表
（一）、基本编辑指令：
新增 (append)

a ：从光标所在位置後面开始新增资料，光标後的资料随新增资料向後移动。

A： 从光标所在列最後面的地方开始新增资料。

插入 (insert)

i： 从光标所在位置前面开始插入资料，光标後的资料随新增资料向後移动。

I ：从光标所在列的第一个非空白字元前面开始插入资料。

开始 (open)

o ：在光标所在列下新增一列并进入输入模式。

O: 在光标所在列上方新增一列并进入输入模式。

x： 删除光标所在字符。

dd ：删除光标所在的列。

r ：修改光标所在字元，r 後接著要修正的字符。

R： 进入取替换状态，新增文字会覆盖原先文字，直到按 [ESC] 回到指令模式下为止。

s： 删除光标所在字元，并进入输入模式。

S： 删除光标所在的列，并进入输入模式。

（二）、光标移动指令：
由於许多编辑工作是藉由光标来定位，所以 vi 提供许多移动光标的方式，这个我们列

几张简表来说明（这些当然是指令模式下的指令）：

指令	说明	功能键
0	移动到光标所在列的最前面	[Home]
$	移动到光标所在列的最後面	[End]
[CTRL][d]	向下半页	
[CTRL][f]	向下一页	[PageDown]
[CTRL]	向上半页	
[CTRL]	向上一页	[PageUp]

指令	说明
H	移动到视窗的第一列
M	移动到视窗的中间列
L	移动到视窗的最後列
b	移动到下个字的第一个字母
w	移动到上个字的第一个字母
e	移动到下个字的最後一个字母
^	移动到光标所在列的第一个非空白字元

指令	说明
n-	减号移动到上一列的第一个非空白字元 前面加上数字可以指定移动到以上 n 列
n+	加号移动到下一列的第一个非空白字元 前面加上数字可以指定移动到以下 n 列
nG	直接用数字 n 加上大写 G 移动到第 n 列

指令	说明
fx	往右移动到 x 字元上
Fx	往左移动到 x 字元上
tx	往右移动到 x 字元前
Tx	往左移动到 x 字元前
;	配合 f&t 使用，重复一次
,	配合 f&t 使用，反方向重复一次
/string	往右移动到有 string 的地方
?string	往左移动到有 string 的地方
n	配合 /&? 使用，重复一次
N	配合 /&? 使用，反方向重复一次

指令	说明	备注
n(	左括号移动到句子的最前面	句子是以
	前面加上数字可以指定往前移动 n 个句子	!.? 三种符号来界定
n)	右括号移动到下个句子的最前面	
	前面加上数字可以指定往後移动 n 个句子	
n{	左括弧移动到段落的最前面	段落是以
	前面加上数字可以指定往前移动 n 个段落	段落间的空白列界定
n}	右括弧移动到下个段落的最前面	
	前面加上数字可以指定往後移动 n 个段落	

（三）、更多的编辑指令

这些编辑指令非常有弹性，基本上可以说是由指令与范围所构成。例如 dw 是由删除指令 d 与范围 w 所

组成，代表删除一个字 d(elete) w(ord) 。

指令列表如下：

d 删除(delete)

y 复制(yank)

p 放置(put)

c 修改(change)

范围可以是下列几个：

e 光标所在位置到该字的最後一个字母

w 光标所在位置到下个字的第一个字母

b 光标所在位置到上个字的第一个字母

$ 光标所在位置到该列的最後一个字母
0 光标所在位置到该列的第一个字母
) 光标所在位置到下个句子的第一个字母
(光标所在位置到该句子的第一个字母
} 光标所在位置到该段落的最後一个字母
{ 光标所在位置到该段落的第一个字母
说实在的，组合这些指令来编辑文件有一点点艺术气息。不管怎麼样，它们提供更多编辑文字的能力。值得
注意的一点是删除与复制都会将指定范围的内容放到暂存区里，然後就可以用指令 p 贴到其它地方去，这
是 vi 用来处理区段拷贝与搬移的办法。
某些 vi 版本，例如 Linux 所用的 elvis 可以大幅简化这一坨指令。如果稍微观察一下这些编辑指令
就会发现问题其实是定范围的方式有点杂，实际上只有四个指令罢了。指令 v 非常好用，只要按下 v 键，
光标所在的位置就会反白，然後就可以移动光标来设定范围，接著再直接下指令进行编辑即可。
对於整列操作， vi 另外提供了更方便的编辑指令。前面曾经提到过删除整列文字的指令 dd 就是其中一个
；cc 可以修改整列文字；而 yy 则是复制整列文字；指令 D 则可以删除光标到该列结束为止所有的文字。
（四）、文件操作指令
文件操作指令多以 : 开头，这跟编辑指令有点区别。
:q 结束编辑(quit)
:q! 不存档而要放弃编辑过的文件。
:w 保存文件(write)其後可加所要存档的档名。
:wq 即存档後离开。
zz 功能与 :wq 相同。
与:wq 相同