目 录

1. 进入 vi

vi 是 Linux 系统下最基本的文本编辑程序，工作在字符模式下，不需要图形界面。它从诞生至今一直得到广大用户的青睐，历经数十年仍然是人们主要使用的文本编辑工具，足以见其生命力之强，而强大的生命力是其强大的功能带来的。可以有多种方式进入 vi，比如在 shell 下输入：

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 -r filename	在上次正用 vi 编辑时发生系统崩溃，恢复文件
vi file1....filen	打开多个文件，依次编辑

2. vi 的工作模式

基本上 vi 可以分为三种状态，分别是命令模式（command mode）、插入模式（Insert mode）和底行模式（last line mode），各模式的功能区分如下：

- 命令行模式 command mode）：控制屏幕光标的移动，字符、字或行的删除，移动复制某区段及进入 Insert mode 下，或者到 last line mode。
- 插入模式（Insert mode）：只有在 Insert mode 下，才可以做文字输入，按「ESC」键可回到命令行模式。
- 底行模式（last line mode）：将文件保存或退出 vi，也可以设置编辑环境，如寻找字符串、列出行号等。

vi 在初始启动后首先进入命令模式，这时用户可以利用一些预先定义的按键来移动光标、删除文字、复制或粘贴文字等。这些按键均是普通的字符，例如 k 是向下移动光标，相当于向下箭头键。在命令模式下，用户还可以利用一些特殊按键选定文字，然后再进行删除、或复制等操作。

在命令模式下键入 i, a, o 等命令之后，可进入插入模式。在插入模式下，用户随后输入的，除 ESC 之外的任何字符均将被看成是插入到编辑缓冲区中的字符。按 ESC 之后，从插入模式切换回到命令模式。

在命令模式下键入:（冒号） 可进入底行模式。在底行模式，vi 将把光标挪到屏幕的最下方，并在第一个字符的位置显示一个: （冒号）符。这时，用户就可以键入一些命令。这些命令可用来保存文件、读取文件内容、执行 Shell 命令、设置 vi 参数、以正则表达式的方式查找字符串或替换字符串等。

3. 命令模式

3.1 移动光标

要对正文内容进行修改，首先必须把光标移动到指定位置。移动光标的最简单的方式是按键盘的上、下、左、右箭头键。除了这种最原始的方法之外，用户还可以利用 vi 提供的众多字符组合键，在正文中移动光标，迅速到达指定的行或列，实现定位。例如：

k、j、h、l	功能分别等同于上、下、左、右箭头键
Ctrl+b	在文件中向上移动一页（相当于 PageUp 键）
Ctrl+f	在文件中向下移动一页（相当于 PageDown 键）
ctrl+u	屏幕往后移动半页
ctrl+d	屏幕往前移动半页
H	将光标移到屏幕的最上行（Highest）
nH	将光标移到屏幕的第 n 行（如 2H： 将光标移到屏幕的第 2 行）
M	将光标移到屏幕的中间（Middle）
L	将光标移到屏幕的最下行（Lowest）
nL	将光标移到屏幕的倒数第 n 行（如 3L：将光标移到屏幕的倒数第 3 行）
w	在指定行内右移光标，到下一个字的开头
e	在指定行内右移光标，到一个字的末尾
b	在指定行内左移光标，到前一个字的开头
0	数字 0，左移光标，到本行的开头
G	光标移动到文章的最后
nG	光标移动到文章的第 n 行（如 8G：移动到文章的第 8 行）
$	右移光标，到本行的末尾
^	移动光标，到本行的第一个非空字符

3.2 替换和删除

将光标定位于文件内指定位置后，可以用其他字符来替换光标所指向的字符，或从当前光标位置删除一个或多个字符。例如：

rc	用 c 替换光标所指向的当前字符

nrc	用 c 替换光标所指向的前 n 个字符（如 5rc： 用 c 替换光标所指向的前 5 个字符）
x	删除光标所在位置后面的一个字符
nx	删除光标所在位置后面的 n 个字符（如 3x：删除光标所在位置后面的 3 个字符）
X	大写的 X，删除光标所在位置前面的一个字符
nX	删除光标所在位置前面的 n 个字符（如 3X ： 删除光标所在位置前面的 3 个字符）
dd	删除光标所在行，并去除空隙
ndd	从光标所在行开始删除 n 行内容，并去除空隙（如 3dd： 删除 3 行内容，并去除空隙）

3.3 粘贴和复制

从正文中删除的内容（如字符、字或行）并没有真正丢失，而是被剪切并复制到了一个内存缓冲区中。用户可将其粘贴到正文中的指定位置。完成这一操作的命令是：

p	小写字母 p，将缓冲区的内容粘贴到光标的后面
P	大写字母 P，将缓冲区的内容粘贴到光标的前面

如果缓冲区的内容是字符或字，直接粘贴在光标的前面或后面；如果缓冲区的内容为整行正文，则粘贴在当前光标所在行的上一行或下一行。

注意上述两个命令中字母的大小写。vi 编辑器经常以一对大、小写字母（如 p 和 P）来提供一对相似的功能。通常，小写命令在光标的后面进行操作，大写命令在光标的前面进行操作。

有时需要复制一段正文到新位置，同时保留原有位置的内容。这种情况下，首先应当把指定内容复制（而不是剪切）到内存缓冲区。完成这一操作的命令是：

yy	复制当前行到内存缓冲区
nyy	复制 n 行内容到内存缓冲区（如 5yy： 复制 5 行内容到内存缓冲区）

3.4 搜索字符串

和许多先进的编辑器一样，vi 提供了强大的字符串搜索功能。要查找文件中指定字或短语出现的位置，可以用 vi 直接进行搜索，而不必以手工方式进行。搜索方法是：键入字符 / ，后面跟以要搜索的字符串，然后按回车键。编辑程序执行正向搜索（即朝文件末尾方向），并在找到指定字符串后，将光标停到该字符串的开头；键入 n 命令可以继续执行搜索，找出这一字符串下次出现的位置。用字符 ? 取代 / ，可以实现反向搜索（朝文件开头方向）。例如：

/str1	正向搜索字符串 str1
n	继续搜索，找出 str1 字符串下次出现的位置
?str2	反向搜索字符串 str2

无论搜索方向如何，当到达文件末尾或开头时，搜索工作会循环到文件的另一端并继续执行。

3.5 撤销和重复

在编辑文档的过程中，为消除某个错误的编辑命令造成的后果，可以用撤消命令。另外，如果用户希望在新的光标位置重复前面执行过的编辑命令，可用重复命令。

u	撤消前一条命令的结果
.	重复最后一条修改正文的命令

3.6 文本选中

vi 可进入到一种成为 visual 的模式，在该模式下，用户可以用光标移动命令可视地选择文本，然后再执行其他编辑操作，例如删除、复制等。

v	字符选中命令
V	行选中命令

4. 插入模式

4.1 进入插入模式

在命令模式下正确定位光标之后，可用以下命令切换到插入模式：

i	在光标左侧输入正文
a	在光标右侧输入正文
o	在光标所在行的下一行增添新行
O	在光标所在行的上一行增添新行
I	在光标所在行的开头输入正文
A	在光标所在行的末尾输入正文

4.2 退出插入模式

退出插入模式的方法是，按 ESC 键或组合键 Ctrl+[。

5. 底行模式

在 vi 的底行模式下，可以使用复杂的命令。在命令模式下键入“:”，光标就跳到屏幕最后一行，并在那里显示冒号，此时已进入底行模式。底行模式又称“末行模式”，用户输入的内容均显示在屏幕的最后一行，按回车键，vi 执行命令。

5.1 退出命令

在命令模式下可以用 ZZ 命令退出 vi 编辑程序，该命令保存对正文所作的修改，覆盖原始文件。如果只需要退出编辑程序，而不打算保存编辑的内容，可用下面的命令：

: q　　在未作修改的情况下退出

: q!　　放弃所有修改，退出编辑程序

5.2 行号与文件保存

编辑中的每一行正文都有自己的行号，用下列命令可以移动光标到指定行：

: n　　将光标移到第 n 行

:set nu　　显示行号

:set nonu　　取消行号显示

底行模式下，可以规定命令操作的行号范围。数值用来指定绝对行号；字符“.”表示光标所在行的行号；字符“$”表示正文最后一行的行号；简单的表达式，例如“.+5”表示当前行往下的第 5 行。例如：

:345　　将光标移到第 345 行

: .+5　　将光标移到当前行之后的第 5 行

: $　　将光标移到正文最后一行

在底行模式下，允许从文件中读取正文，或将正文写入文件。例如：

:w　　将编辑的内容写入原始文件，用来保存编辑的中间结果

:wq　　将编辑的内容写入原始文件并退出编辑程序（相当于 ZZ 命令）

:w file　　将编辑的内容写入 file 文件，保持原有文件的内容不变

:a,bw file　　将第 a 行至第 b 行的内容写入 file 文件 （如:1,.w file　将第 1 行至当前行写入 file 文件）

:r file　　读取 file 文件的内容，插入当前光标所在行的后面

:f file　　将当前文件重命名为 file

5.3 字符串搜索

给出一个字符串，可以通过搜索该字符串到达指定行。如果希望进行正向搜索，将待搜索的字符串置于两个“/”之间；如果希望反向搜索，则将字符串放在两个“?”之间。例如：

:/str/　　正向搜索，将光标移到下一个包含字符串 str 的行

:?str?　　反向搜索，将光标移到上一个包含字符串 str 的行

5.4 正文替换

利用 :s 命令可以实现字符串的替换。具体的用法包括：

:s/str1/str2/　　用字符串 str2 替换行中首次出现的字符串 str1

:s/str1/str2/g　　用字符串 str2 替换行中所有出现的字符串 str1

:.,$ s/str1/str2/g　用字符串 str2 替换正文当前行到末尾所有出现的字符串 str1

:1,$ s/str1/str2/g　用字符串 str2 替换正文中所有出现的字符串 str1

:g/str1/s//str2/g　功能同上

从上述替换命令可以看到：g 放在命令末尾，表示对搜索字符串的每次出现进行替换；不加 g，表示只对搜索字符串的首次出现进行替换；g 放在命令开头，表示对正文中所有包含搜索字符串的行进行替换操作。

5.5 删除正文

在底行模式下，同样可以删除正文中的内容。例如：

:d　　删除光标所在行

:3d　　删除第 3 行

:.,$d　　删除当前行至正文的末尾

:/str1/,/str2/d　删除从字符串 str1 到 str2 的所有行

5.6 恢复文件

vi 在编辑某个文件时，会另外生成一个临时文件，这个文件的名称通常以 . 开头，并以 .swp 结尾。vi 在正常退出时，该文件被删除，若意外退出，而没有保存文件的最新修改内容，则可以使用恢复命令，也可以在启动 vi 时利用 -r 选项。

:recover　　恢复文件

5.7 选项设置

为控制不同的编辑功能，vi 提供了很多内部选项。利用 :set 命令可以设置选项。基本语法为：

:set option　　设置选项 option

常见的功能选项包括：

autoindent	设置该选项，则正文自动缩进
ignorecase	设置该选项，则忽略规则表达式中大小写字母的区别
number	设置该选项，则显示正文行号
ruler	设置该选项，则在屏幕底部显示光标所在行、列的位置
tabstop	设置按 Tab 键跳过的空格数。例如 :set tabstop=n，n 默认值为 8
mk	将选项保存在当前目录的 .exrc 文件中

5.8 shell 切换

在编辑正文时，利用 vi 底行模式下提供的 shell 切换命令，无须退出 vi 即可执行 Linux 命令，十分方便。语法格式为：

:! command　　执行完 shell 命令 command 后回到 vi

另外，在命令模式下，键入 K ，可命令 vi 查找光标所在单词的手册页，相当于运行 man 命令。

6. vim 的高级特色

vim 代表 vi IMproved，如同其名称所暗示的那样，vim 作为标准 linux 系统 vi 编辑器的提高版而存在。 vim 除提供和 vi 编辑器一样强大的功能外，还提供有多级恢复、命令行历史以及命令及文件名补全等功能。关于 vim 的高级用法，请在熟练使用 vi 后自行研究。