第3章 全屏幕编辑程序vi

- 早期的UNIX提供的编辑器是行编辑ed。UNIX的全屏幕编辑器vi，现在所有的UNIX版本都支持。目前，行编辑ed还经常用于Shell脚本程序中，在脚本程序中，根据用户的输入信息修改一个文本文件的内容。行编辑程序ed的显示是面向行的，对终端的类型和特性没有任何的特殊要求和限制，ed程序本身也比较简单。
- vi可以交互式编辑文本文件，编辑是面向屏幕的，终端的类型设置必须正确，否则无法正常工作。由于vi在所有UNIX之间通用，也可以应用于各种各样的终端，占用系统资源很少，所以vi仍然被广泛使用。

3.1 vi的启动方法

用法：**vi *filename***

例如：**vi abc.c，启动vi编辑文件abc.c。**

vi有搜索命令和卷动功能，可以用来浏览文本文件，比more，less等更方便，但是应当避免在浏览时对文件无意中做出错误的修改。可以使用view代替vi进入编辑程序，就不允许修改文件内容。

3.2 vi的选项

vi有40多个选项控制vi的运行。

1. 用.exrc文件控制

vi一启动后就自动读取用户自己主目录（Home Directory）下的文件.exrc，获取用户自设定的vi选项，未指定的选项按默认值处理。

如：建立文件.exrc，其中含有如下两行：

```
set number
set showmode
```

其中第一条命令使得vi在列出每一行时，在左边列出行号。第二条命令使得vi在屏幕右下角标志出当前是否处于输入状态。

2．在vi中使用set命令

例：

:set number 在列出每一行时，左边列出行号

:set showmode 在屏幕右下角标志出当前是否处于输入状态

:set nonumber 在列出每一行时，在左边不列出行号

:set noshowmode 在屏幕右下角不标志出当前是否处于输入状态

:set all 列出所有开关的当前状态

一般的，设置选项和取消选项的命令格式分别为（其中***option***为选项名）

:set *option*

:set nooption

3.3 vi的工作方式

vi的工作方式分命令方式和输入方式。vi启动后就进入命令方式，参见图3-1。

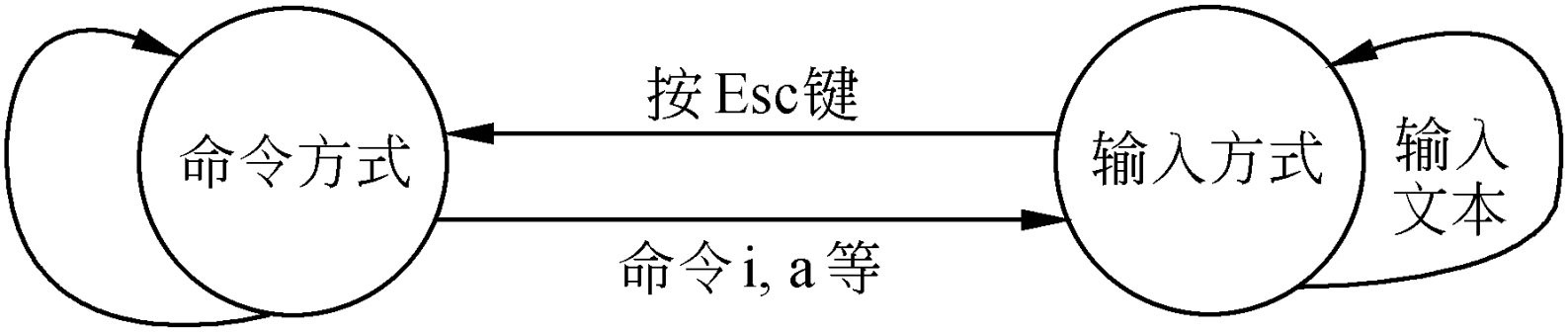

图3-1 vi的工作方式转换

- 处于命令方式时，用户键入的内容被当作vi的命令来解释，一般处于命令方式下按键无回显（以冒号打头的命令和查找命令除外）。编辑命令i，a等，可以从命令方式转到输入方式。
- 处于输入方式时，用户键入的所有内容全部作为输入的正文内容，用户可以输入多行，每输入完一行后按回车键转入下一行，正文输入时有回显。输入完毕，按键盘左上角的Esc键，返回到命令方式。

3.4 vi的编辑命令

当vi处于命令状态时，用户的按键不回显，被解释成编辑命令，vi大约有100多个编辑命令。下面介绍的vi命令子集，足可以完成一般的编辑任务。

3.4.1 正文插入命令

- **命令i，在当前光标处插入正文段，直至按Esc键。**

- 在命令方式下，按下i键后，进入输入方式。从此以后，输入的文本在屏幕上回显，输入完一整行后，按下Enter键，继续输入下一行。输入结束后，按Esc键，退出输入方式，回到命令方式。
- 回到命令方式之后，按键信息不再回显，所有的按键被解释为命令。
- 许多传统的UNIX中，vi必须首先按下Esc键退出输入模式之后再移动光标到其他行，修改后，移动光标回来，再按下i命令（或者a，o，O）重新进入输入方式。在当前输入行的错误修正，使用Backspace键，不需要退出输入方式。

- 在当前光标后追加（append）正文段的命令a，也可以进入输入模式，直至按Esc键。
- 命令o，在当前行之下插入新行（open），进入输入模式，直至按Esc键。
- 大写字母O命令，在当前行之上插入新行（open），进入输入模式，直至按Esc键。

3.4.2 光标移动命令

1. 单字符移动

h 光标左移一列

j 光标下移一行

k 光标上移一行

l 光标右移一列

在vi中有许多命令可以在命令前加上一个整数，标志这个命令连续执行多少遍，例：

5h 光标左移5列

6j 光标下移6行

23k 光标上移23行

10l 光标右移10列

有的终端可以直接使用键盘上的箭头按键代替这四个字母，更便于用户使用。vi被设计成对终端特性的依赖性最小，所以，不依赖于终端的功能键，仅使用键盘的字母、数字和符号键vi就能完成编辑工作。

2．翻页

Ctrl+B键：向后翻页（Backward）

Ctrl+F键：向前翻页（Forward）

Ctrl+U键：向上翻半页（Up）

Ctrl+D键：向下翻半页（Down）

在vi中，把向文件尾方向定义为“向前”，向文件头方向定义为“向后”，这与许多人的习惯不同。

在PC上的UNIX允许用PgDn键代替Ctrl+F键，用PgUp键代替Ctrl+B。

也可以使用下面的键：

6Ctrl+F键：向前翻6页

15Ctrl+B键：向后翻15页

3．将光标移至当前行首^或0（Linux）

4．将光标移至当前行尾$

5．移到右一个单词 w W，移到左一个单词 b B

w，b与W，B的区别是它们对“单词”的定义不同。小写命令的命令w和b，以非字母、数字、下划线之外的所有字符作为“单词”分界符。大写命令的命令W和B，仅以空白符（空格或者制表符）作为“单词”分界符。同前述其他命令类似，也可以使用类似6w，3W，5b，10B命令。

6. 移到指定的行

使用这种方法可以立即将光标定位到需要修改的行。例：

:476　　将光标定位于第476行

:1　　将光标定位于第1行（文件首）

:$　　将光标定位于文件尾

:$-10　　将光标定位于文件倒数第10行

在描述行号时，可以使用句点（.）代表当前行号，使用$代表最后一行的行号，而且可以使用整数加减法，如最后一例。

7．括号配对命令 %

先把光标移到一个大括号（或括号，或方括号）上，按%键，则光标自动定位到与它配对的那一个括号，对编写和检查C语言的源程序非常有用。

3.4.3 设置书签

vi允许设置以单个英文字母命名的最多26个标记（mark），许多编辑器把这种功能叫“书签（bookmark）”。vi的书签记忆了一个行号。

设置书签的命令是m。例如：顺序按下两个键m和a，尽管终端上没什么特别的显示信息，但是，vi已经将当前行号记为名字为a的书签。设置的所有书签，在vi退出后，不再保存。

vi许多编辑命令可以使用命名的书签，将光标移动到指定书签处的命令是’(单引号)。

例：’a 连续按下单引号和字母a，光标会移动到书签a处

’e 光标移动到书签e处

3.4.4 删除

1. 删除当前字符的命令：x

类似的，命令5x删除从当前光标开始的5个字符。

2. 删除当前行的命令：dd

类似的，命令3dd删除从当前行开始的3行。

3. 与光标移动命令相关的删除命令

d'e 从当前光标处删除到书签e处

d$ 从当前光标处删除到行尾

d^ 从当前光标处删除到行首

dw 删除一个单词

d% 将光标移动到一个括号字符上，删除和它配对的括号括起的段落

3.4.5 字符替换

1. 替换光标处字符的命令r

例：ra命令将当前光标处字符替换为a。

如果希望将当前光标处开始的三个字符依次替换为abc，则需要按命令rarbrc。

2. 替换多个字符的命令R

例：命令Rabcdef，然后按Esc键。

该命令把从当前光标开始的字符依次替换为abcdef，用Esc键来结束多字符替换命令。这类似于以“覆盖”（overwrite）方式进入编辑状态。

3.4.6 取消和重复

1. 命令u

取消上一次的编辑操作。例如：误删除了一段正文，用u命令可以恢复到删除前的状态（undo）。

vi的取消操作，只能回退一次。许多新版本的vi对这个问题作了改进。

2. 命令.

重复上一次的编辑操作。按句点键，可以重复上一次的编辑操作。例如：按3dd命令删除了3行，然后按句点键就再删除3行，如果接着连续按句点（.）键，则每按一次删除3行。

3.4.7 文件命令

（1）存盘退出ZZ，存盘退出:wq

（2）存盘不退出:w

（3）不存盘退出:q!

（4）读入一个文件插入到当前行之下:r a.c

（5）写文件

把第50行至文件尾的内容写到文件junk中

:50,$w junk

如果文件junk事先已经存在，使用下述命令强制把它覆盖掉

:50,$w! junk

如果编辑了文件之后，无法存盘（例如：文件没有写权限），那么可以用

:w file1

将当前编辑好的文件内容存到另一个文件中。

3.4.8 段落的删除、复制和移动

1. 删除命令:d

用法为：:l1,l2d，删除第*l1*～*l2*行。例如：

:10,50d　　删除第10～50行

:1,.d　　删除文件首至当前行的部分

2. 复制命令:co

用法为：:l1,l2col3，将第*l1*～*l2*行复制到第*l3*行之下。例如：

:5,10co56　复制第5～10行到第56行之下

3．移动命令:m

用法为：**:l1,l2ml3**，将第*l1*～*l2*行移动到第*l3*行之下。例如：

:8,34m78　移动第8～34行到第78行之下

行号描述时除了可以使用句点代表当前行，$代表最后一行，还可以使用“书签”，例如:下面的命令中'e代表书签e的行号。

:'e,.d

3.4.9 剪贴板

vi有一个通用的缓冲区和用单个的英文字母命名的26个有名缓冲区，用于保留一些文本。前面介绍的删除命令，vi会在删除了这些信息之后，自动把这些信息保留在通用缓冲区中。下面的命令都会执行两个操作：删除信息和将删除的信息放置到通用缓冲区中。

dd 删除当前行

3dd删除当前行开始的3行

d'e 删除从当前光标处到书签e处（书签e需要事先用命令me设置好）

d$ 删除从当前光标处到行尾

d^ 删除从当前光标处到行首

dw 删除一个单词

d% 首先将光标定位到一个括号（或方括号、大括号）字符上，删除从此开始到和它配对的括号处

:1,.d　　删除文件首到当前行的段落

除了上述的d命令之外，还有“抽取（yank）”命令y，它仅仅把指定的信息复制到通用缓冲区，但不删除它们。用法和d命令类似。

yy 当前行

3yy 当前行开始的3行

y'e 从当前光标处到书签e处（书签e需要事先用命令me设置好）

y$ 从当前光标处到行尾

y^ 从当前光标处到行首

yw 一个单词

y% 首先将光标定位到一个括号字符上，从此开始到和它配对的括号处

:1,.y 文件首到当前行的段落

将缓冲区中保留的信息粘贴到光标处，使用p命令（put或paste），即在信息保留到缓冲区之后，就可以使用p命令，将它们粘贴到文件合适的位置。

vi除了使用这个默认的缓冲区之外，还有26个用单个英文字母命名的有名字的缓冲区。与缓冲区有关的3个命令是d，y，p，在使用有名字缓冲区时，在这些命令前加两个字符的前缀，一个字符是双引号，一个字符是英文字母代表的缓冲区名字。例如：

"a3dd　　删除当前行开始的3行，并把信息保留到a缓冲区中

"by'e　　复制当前光标到书签e处内容到b缓冲区中

"ky%　　光标定位在一个括号字符上，复制从此开始到和它配对的括号处的段落到缓冲区k中

"kp　　粘贴出缓冲区k中的内容

3.4.10 其他命令

1．两行合并J（大写字母，Join）

把当前行下面的行合并到当前行。

2．刷新屏幕显示Ctrl+L键

在阅读完 “突然”出现的信息后，按Ctrl+L键，恢复vi原先的屏幕显示。在more命令中也介绍过类似的功能。

3．状态显示Ctrl+G键

在屏幕最下面一行列出正在编辑的文件的名字、总行数、当前行号、文件是否被修改过等信息。

3.4.11 模式查找

在vi的模式查找命令中，使用“正则表达式”来描述一个字符串模式。命令格式为：

/模式

例：/[0-9][0-9]*

继续查找命令：n和N

- 小写字母n键，向下继续查找下一个，查到文件尾后，自动折到文件首继续向下查找
- 大写字母N，向上继续查找下一个，查到文件头后自动折到文件尾继续向上查找

3.4.12 模式替换

1. 基本用法

在vi的模式替换命令中，也使用“正则表达式”来描述一个字符串模式。命令格式为：

:行号，行号s/模式/替换字符串/g

【例】 将abc字符串替换为xyz。

:1,50s/abc/xyz/

将第1～50行的字符串abc换为xyz，如果同一行内有多个abc字符串，则只替换第一个。

:1,50s/abc/xyz/g

将第1～50行的字符串abc换为xyz，如果同一行内有多个abc字符串，则替换所有的abc字符串。

:s/abc/xyz/g

仅将当前行的字符串abc换为xyz，如果当前行中有多个abc字符串，则替换所有的abc字符串。

2．模式描述使用正则表达式

使用替换命令时应特别注意在描述模式时使用的是正则表达式

【例】 注意vi替换命令中用于模式描述的正则表达式中的特殊字符。

（1）将end.改为middle.

```
:1,$s/end\./middle./g
```

（2）删除每行尾部的一个字符

```
:1,$s/.$//g
```

（3）将a[i]*b[j]替换为x[k]*y[n]

:1,$s/a\[i]*b\[j]/x[k]*y[n]/g

将buf.length/1000替换为 buffer.size/1024。

:1,$s/buf\.length\/1000/buffer.size\/1024/g

（4）在编辑C语言程序的时候，将f[n]替换为f[i]

使用下面的命令：:1,$s/f\[n]/f[i]/g

类似的，将当前行开始到文件尾的段落中所有的x*y替换为x+y，使用下面的命令：

:.,$s/x*y/x+y/g（源程序中的x*y被替换为x*x+y，而且，其他地方的y都被替换成了x+y。）

正确的用法是：:.,$s/x*y/x+y/g

3. 替换字符串中的符号&

在替换字符串中特殊字符 & 代表被模式所匹配的那部分。

【例3】 替换字符串中&符的作用。

设文件当前只含有四行，每行为一个整数，内容为：

5

2

10

18

执行下边的命令

```
:1,$s/[0-9][0-9]*/192.168.24.&   host&/
```

然后，文件内容变为：

192.168.24.5　host5

192.168.24.2　host2

192.168.24.10　host10

192.168.24.18　host18

&是C语言中常用的符号，所以要特别注意。将*pointer替换为&record的命令是：

:1,$s/*pointer/\&record/g

4. 使用正则表达式的更灵活替换\(和\)

- 在正则表达式中出现的\(和\)不影响匹配操作。
- 使用\(和\)的目的不在于匹配操作，而是在于替换操作。
- 在替换字符串中使用\1，就代表了\(和\)标注出来的那段正则表达式所匹配的部分

【例】使用正则表达式中\(和\)实现更灵活的替换功能。

（1）下边的命令完成了number到num的替换，但是这种替换仅限于在一个C语言指针引用之后的number，对于其他地方出现的number不进行替换。

:1,$s/\([a-zA-Z_][a-zA-Z0-9_]*\)->number/\1->num/g

（2）将文件中的日期格式“月-日-年”改为“年.月.日”，比如：04-26-1997替换为1997.04.26，使用的命令为：

:1,$s/\([0-9][0-9]\)-\([0-9][0-9]\)-\([0-9][0-9]*\)/\3.\1.\2/g

（3）替换字符串中可以使用特殊的\0代表被整个正则表达式匹配的部分，前面例3-4中的命令:

:1,$s/[0-9][0-9]*/192.168.24.&　host&/

也可以等价地写作下列的命令:

:1,$s/[0-9][0-9]*/192.168.24.\0　host\0/

使用正则表达式的替换方法，除了在vi和前面介绍的流编辑命令sed中使用之外，在许多其他的信息系统中，包括非UNIX的系统中，也得到了广泛的使用。这几乎成为了一种事实上的标准，但是在某些细节上，可能会有些差别。如果设计的软件系统也需要类似的功能，最好也遵循这种多数人都熟悉的成熟规则，以便于简化设计、开发和使用的培训。

3.4.13 编辑命令小结

表3-1 vi编辑命令

命　　令	举　　例	说　　明
i a o O		（Insert/Append/Open）正文插入命令，进入输入模式，直到按下 Esc 键
h j k l		光标移动一行或一列
Ctrl+F 键　Ctrl+B 键		（Forward/Backward）翻页
Ctrl+U 键　Ctrl+D 键		（Up/Down）翻半页
^ $		光标移到行首和行尾
w W b B		（Word）光标移动一个单词
:*n*	:25	光标移动到指定行
%		括号配对
m	mq	（Mark）设置一个命名的书签
'	'q	引用书签，光标移动到指定名字的书签处
x		删除光标处字符
d	dd d'q dw d%	（Delete）删除
y	yy　y% y'q	（Yank）抽取命令，将正文段复制到剪贴板
p		（Put）将剪贴板内容粘贴到当前位置
"	"a3dd　"ap "by'e　"bp "cd%　"cp	使用命名剪贴板，用于 d，y，p 命令，作这些命令前缀

命　　令	举　　例	说　　明
r	rarb	（Replace）替换
R		替换，直到按下 Esc 键
u		（Undo）撤销上一次操作
.		重复上次操作
ZZ		存盘退出
:wq		存盘退出
:w		（Write）存盘不退出
:q!		（Quit）不存盘退出
:r file		（Read）读入文件
:w file	:10,35w fa	（Write）写入到文件
:行号,行号 d	:.$d	（Delete）删除
:行号,行号 co 行号	:1,30co42	（Copy）复制
:行号,行号 m 行号	:70,100m43	（Move）移动
J		（Join）两行合并
Ctrl+L 键		刷新屏幕显示
Ctrl+G 键		显示当前状态（文件名，行号等）
/模式	/Test	模式查找
n N		（Next）查找下一个
:行号，行号 s/模式/替换字符串/	:1,$s/abc/xyz/g	（Substitute）模式替换