

说明

本文档内容来自 O'REILLY 出版社的《学习 vi 编辑器》，该书的中文版由机械工业出版社出版。切勿用于商业用途。

本文档不是 vi/vim 的教程，只是列出一些快捷键和命令以方便记忆和查询。如果你需要 vim 的教程，我推荐李果正先生的《大家来学 vim》（正体中文），你可以在下面的网站浏览这个教程或下载 pdf 文档：

<http://edt1023.sayya.org/vim/>

基本的 Vi 命令

一、编辑命令

文本目标	修改	删除	复制
1 个单词	cw	dw	yw
2 个单词，不计标点	2cw 或 c2w	2dw 或 c2w	2yd 或 y2w
向后 3 个单词	3cb 或 c3b	3db 或大 b	3yb 或也 b
1 行	cc	dd	yy 或 Y
到行尾	c\$或 C	d\$或 D	y\$
到行首	c0 (数字零)	d0 (数字零)	y0 (数字零)
单个字符	r	x 或 X	y1 或 yh
5 个字符	5s	5x	5y1

二、移动

移动	命令
←, ↓, ↑, →	h, j, k, l
到下一行的首字符	+
到上一行的首字符	-
到单词的尾部	e 或 E
按单词前移	w 或 W
按单词后移	b 或 B
到行尾	\$
到行首	0 (数字零)

三、其他操作

操作	命令
从缓冲区输出文本	p 或 P
启动 i，如果指定了文件，就打开该文件	vi file
保存编辑、退出文件	ZZ (大写)
不保存编辑、退出文件	:q!

四、文本创建和操作命令

编辑行为	命令
在当前位置插入文本	i
在行首插入文本	I
在当前位置追加文本	a
在行尾追加文本	A
在光标所在行的上面新建一行，等待输入新文本	o（小写字母 o）
在光标所在行的下面新建一行，等待输入新文本	O（大写字母 O）
删除行并替换文本	S（大写）
使用新文本覆盖现有文本	R
合并当前行和下一行	J
转换大小写	~
重复上次操作	.（句号）
取消上次修改	u（小写）
恢复行到初始状态	U（大写）

vi 移动命令

移动	命令
向前滚动一屏	<code>^F</code> (<code>^</code> 表示 Ctrl, <code>^F=Ctrl+F</code> , F 对大小写不敏感, 下同)
向后滚动一屏	<code>^B</code>
向前滚动半屏	<code>^D</code>
向后滚动半屏	<code>^U</code>
向前滚动一行	<code>^E</code>
向后滚动一行	<code>^Y</code>
把当前行移动到屏幕顶部并滚动	<code>z</code> 回车 (小写字母 <code>z</code> +回车)
把当前行移动到屏幕中央并滚动	<code>z.</code> (小写字母 <code>z</code> +句号)
把当前行移动到屏幕底部并滚动	<code>z-</code> (小写 <code>z</code>)
刷新当屏幕	<code>^L</code>
移动到起始点——屏幕首行	<code>H</code>
移动到屏幕的中间行	<code>M</code>
移动到屏幕的末行	<code>L</code>
移动到下一行的首字符	回车
移动到下一行的首字符	<code>+</code>
移动到上一行的首字符	<code>-</code>
移动到当前行的第一个非空格字符	<code>^</code> (字符 <code>^</code> , 不是 Ctrl)
移动到当前行的第 <code>n</code> 列	<code>n!</code>
移动到词尾	<code>e</code>
移动到词尾 (忽略标点)	<code>E</code>
移动到当前句子的开始	<code>(</code>
移动到下一句的开始	<code>)</code>
移动到当前段落的开始	<code>{</code>
移动到下一段落的开始	<code>}</code>
移动到当前节的开始	<code>[[</code>
移动到下一节的开始	<code>]]</code>
向前搜索模式	<code>/pattern</code> (<code>pattern</code> 表示搜索内容)
向后搜索模式	<code>?pattern</code>
重复上次搜索	<code>n</code>
反方向重复上次搜索	<code>N</code>
向前重复上次搜索	<code>/</code>

移动	命令
向后重复上次搜索	?
移动到当前行中 x 的下一个实例	f x
移动到当前行中 x 的上一个实例	F x
移动到当前行中 x 的下一个实例的前面	t x
移动到当前行中 x 的上一个实例的后面	T x
同方向重复前面的搜索命令	;
反方向重复前面的搜索命令	, (逗号)
移动到第 n 行	nG
移动到文件的尾部	G
返回到以前的标记或上下文	` (反引号, 在数字键 1 的左边)
返回到包含以前标记的行的开始	' (单引号, 在回车键的左边)
显示当前行 (不是移动命令)	^G

更多的组合命令

修改	删除	复制	从光标到……
cH	dH	yH	到屏幕顶部
cL	dL	yL	到屏幕底部
c+	d+	y+	下一行
c5	d5	y5	当前行的第 5 列
2c)	2y)	2y)	接下来的第二个句子
c{	y{	y{	上一段落
c/pattern	y/pattern	y/pattern	模式
cn	yn	yn	下一个模式
cG	yG	yG	文件尾部
c13G	y13G	y13G	第 13 行

vi 的缓冲区和标记命令

一、命令行选项

选项	含义
+n file	打开文件到第 n 行
+file	打开文件到最后一行
+/pattern file	打开文件到模式首次出现的地方
-c command file	打开文件后运行命令；通常是行号或搜索（POSIX+版）
-R	以只读方式进行（与用 view 代替 vi 一样）
-r	系统崩溃后恢复文件

二、缓冲区名

缓冲区名	缓冲区用途
1-9	前 9 次删除操作，由最近的到最早的
a-z	需要时使用的名字缓冲区，大写字母表示添加到该缓冲区尾

三、缓冲区和标记命令

命令	含义
"b 命令	对缓冲区 b 执行命令
mx	用 x 标记当前位置
'x	移动光标到 x 所记的行的首字符
`x	移动光标到 x 所标记的字符
``	返回到上一标记或上下文的确切位置
''	返回到上一标记或上下文所在行的位置

vi 命令的一般格式

到目前为止，在我们已经提到的所有修改命令中，你可能已经注意到下面的模式：

（命令）（文本目标）

其中命令是修改命令 `c`，文本目标是移动命令（不用输入圆括号）。但是 `c` 并不是唯一需要文本目标的命令，`d` 命令（删除）和 `y` 命令（复制）也遵循这个模式。

请记住移动命令可以带数字参数，因此可以把数字加到 `c`、`d` 和 `y` 命令的文本目标的前面。例如 `d2w` 和 `2dw` 都是删除两个单词的命令。下面是大部分 `vi` 命令都遵循的一般模式：

（命令）（数字）（文本目标）

或者等同于：

（数字）（命令）（文本目标）

这就是 `vi` 命令的使用规则。数字和命令是可选的，如果没有他们，则只有一个移动命令。如果加上数字，就可以移动多次。相反，将命令（`c`、`d` 或 `y`）与文本目标结合就得到编辑命令。

当你认识到可以按照这种方式把多个命令结合在一起时，`vi` 才会真正变成一个功能强大的编辑器。