**Vim入门基础知识集锦**

|  |  |  |
| --- | --- | --- |
| [日期：2017-02-22] | 来源：Linux社区  作者：Linux | [字体：[大](javascript:ContentSize(16)) [中](javascript:ContentSize(0)) [小](javascript:ContentSize(12))] |

想学习Vim，但是Vim官网的手册又太大而全，而网上各方资料要么不全面，要么不够基础。在网上搜集各方资料，按照自己的框架整理一份Vim入门基础教程，分享出来。特点是偏向基础，但对入门者来说足够全面，而且结构框架清晰。

**1. 简介**

Vim（Vi[Improved]）编辑器是功能强大的跨平台文本文件编辑工具，继承自Unix系统的Vi编辑器，支持Linux/Mac OS X/Windows系统，利用它可以建立、修改文本文件。进入Vim编辑程序，可以在终端输入下面的命令：

$vim [filename]

其中filename是要编辑器的文件的路径名。如果文件不存在，它将为你建立一个新文件。Vim编辑程序有三种操作模式，分别称为 **编辑模式**、**插入模式** 和 **命令模式**，当运行Vim时，首先进入编辑模式。

将Vim改造为强大的IDE—Vim集成Ctags/Taglist/Cscope/Winmanager/NERDTree/OmniCppComplete  [http://www.linuxidc.com/Linux/2017-02/140282.htm](https://www.linuxidc.com/Linux/2017-02/140282.htm)

**2. 编辑模式**

Vim编辑方式的主要用途是在被编辑的文件中移动光标的位置。一旦光标移到到所要的位置，就可以进行剪切和粘贴正文块，删除正文和插入新的正文。当完成所有的编辑工作后，需要保存编辑器结果，退出编辑程序回到终端，可以发出ZZ命令，连续按两次大写的Z键。

**2.1 跳转**

如果键盘上有上、下、左、右箭头的导航键，就由这些键来完成光标的移动。另外，可以用下面的键完成同样的 **按字符移动** 功能：

k 上移；

j 下移；

h 左移；

l 右移。

上面这４个键将光标位置每次移动一行或一个 **字符** 。Vim还提供稍大范围移动光标的命令：

ctrl+f 在文件中前移一页（相当于 page down）；

ctrl+b 在文件中后移一页（相当于 page up）；

更大范围的移动：

\* 当光标停留在一个单词上，\* 键会在文件内搜索该单词，并跳转到下一处；

# 当光标停留在一个单词上，# 在文件内搜索该单词，并跳转到上一处；

(/) 移动到 前/后 句 的开始；

{/} 跳转到 当前/下一个 段落 的开始。

g\_ 到本行最后一个不是 blank 字符的位置。

fa 到下一个为 a 的字符处，你也可以fs到下一个为s的字符。

t, 到逗号前的第一个字符。逗号可以变成其它字符。

3fa 在当前行查找第三个出现的 a。

F/T 和 f 和 t 一样，只不过是相反方向;

gg 将光标定位到文件第一行起始位置；

G 将光标定位到文件最后一行起始位置；

NG或Ngg 将光标定位到第 N 行的起始位置。

在屏幕中找到需要的 **一页** 时，可以用下面的命令快速移动光标：

H 将光标移到屏幕上的起始行（或最上行）；

M 将光标移到屏幕中间；

L 将光标移到屏幕最后一行。

同样需要注意字母的大小写。H 和 L 命令还可以加数字。如 2H 表示将光标移到屏幕的第２行，3L 表示将光标移到屏幕的倒数第3行。  
当将光标移到所要的行是，**行内移动** 光标可以用下面的命令来实现：

w 右移光标到下一个字的开头；

e 右移光标到一个字的末尾；

b 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始；

$ 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。

**2.2 搜索匹配**

和许多先进的编辑器一样，Vim 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Vim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；

N 继续搜索，找出 str1 字符串上一次出现的位置；

?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。  
Vim中执行搜索匹配最强大的地方是结合 **正则表达式** 来搜索，后续将会介绍。

**2.3 替换和删除**

Vim常规的删除命令是 d、 x (前者删除 行 ，后者删除 字符 ),结合Vim的其他特性可以实现基础的删除功能。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或一行、多行。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；

5rA 用 A 替换光标所指向的前 5 个字符；

x 删除光标所指向的当前字符；

nx 删除光标所指向的前 n 个字符；

3x 删除光标所指向的前 3 个字符；

dw 删除光标右侧的字；

ndw 删除光标右侧的 n 个字；

3dw 删除光标右侧的 3 个字；

db 删除光标左侧的字；

ndb 删除光标左侧的 n 个字；

5db 删除光标左侧的 5 个字；

dd 删除光标所在行，并去除空隙；

ndd 删除（剪切） n 行内容，并去除空隙；

3dd 删除（剪切） 3 行内容，并去除空隙；

其他常用的删除命令有：

d$ 从当前光标起删除字符直到行的结束；

d0 从当前光标起删除字符直到行的开始；

J 删除本行的回车符（CR），并和下一行合并。

Vim常规的替换命令有 c 和 s ，结合Vim的其他特性可以实现基础的替换功能，不过替换命令执行以后，通常会由 **编辑模式** 进入 **插入模式** ：

s 用输入的正文替换光标所指向的字符；

S 删除当前行，并进入编辑模式；

ns 用输入的正文替换光标右侧 n 个字符；

nS 删除当前行在内的 n 行，并进入编辑模式；

cw 用输入的正文替换光标右侧的字；

cW 用输入的正文替换从光标到行尾的所有字符（同 c$ )；

ncw 用输入的正文替换光标右侧的 n 个字；

cb 用输入的正文替换光标左侧的字；

ncb 用输入的正文替换光标左侧的 n 个字；

cd 用输入的正文替换光标的所在行；

ncd 用输入的正文替换光标下面的 n 行；

c$ 用输入的正文替换从光标开始到本行末尾的所有字符；

c0 用输入的正文替换从本行开头到光标的所有字符。

**2.4 复制粘贴**

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，执行上述粘贴命令将会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。Vim 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；

5yy 复制 5 行内容到内存缓冲区；

“+y 复制 1 行到操作系统的粘贴板；

“+nyy 复制 n 行到操作系统的粘贴板。

**2.5 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. 插入模式**

**3.1 进入插入模式**

在编辑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 在光标左侧插入正文

a 在光标右侧插入正文

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插入

A 在光标所在行的末尾插入

**3.2 退出插入模式**

退出插入模式的方法是，按 ESC 键或组合键 Ctrl+[ ，退出插入模式之后，将会进入编辑模式 。

**4. 命令模式**

在Vim的命令模式下，可以使用复杂的命令。在编辑模式下键入 : ，光标就跳到屏幕最后一行，并在那里显示冒号，此时已进入命令模式。命令模式又称 **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，Vim 执行命令。

**4.1 打开、保存、退出**

在已经启动的Vim中打开一个文件需要用 **:e** 命令：

:e path\_to\_file/filename

保存当前编辑的文件需要用 **:w** 命令（单词 write 的缩写）：

:w

将当前文件另存为 file\_temp 则：

:w file\_temp

在编辑模式下可以用 ZZ 命令退出Vim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。

保存并退出则可以讲两条命令结合起来使用（注意命令顺序，先保存，后退出）：

:wq

**4.2 行号与文件**

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（效果与 **编辑模式** 下的 ngg 或 nG 相同）：

: n 将光标移到第 n 行

命令模式下，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

:345w file 将第 345 行写入 file 文件

:3,5w file 将第 3 行至第 5 行写入 file 文件

:1,.w file 将第 1 行至当前行写入 file 文件

:.,$w file 将当前行至最后一行写入 file 文件

:.,.+5w file 从当前行开始将 6 行内容写入 file 文件

:1,$w file 将所有内容写入 file 文件，相当于 :w file 命令

在命令模式下，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

:r file 读取 file 文件的内容，插入当前光标所在行的后面

:e file 编辑新文件 file 代替原有内容

:f file 将当前文件重命名为 file

:f 打印当前文件名称和状态，如文件的行数、光标所在的行号等

**4.3 字符串搜索**

在 **编辑模式** 讲过字符串的搜索，此处的 **命令模式** 也可以进行字符串搜索，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

:/str/w file 正向搜索，并将第一个包含字符串 str 的行写入 file 文件

:/str1/,/str2/w file 正向搜索，并将包含字符串 str1 的行至包含字符串 str2 的行写

**4.4 Vim中的正则表达式**

当给Vim指定搜索字符串时，可以包含具有特殊含义的字符。包含这些特殊字符的搜索字符串称为正则表达式（Regular Expressions）。例如，要搜索一行正文，这行正文的开头包含 struct 字。下面的命令做不到这一点：

:/struct/

因为它只找出在行中任意位置包含 struct的第一行，并不一定在行的开始包含 struct 。解决问题的办法是在搜索字符串前面加上特殊字符^：

:/^struct/

^ 字符比较每行开头的字符串。所以上面的命令表示：找出以字符串 struct 开头的行。  
也可以用类似办法在搜索字符串后面加上表示行的末尾的特殊字符 $ 来找出位于行末尾的字：

:/^struct/

下表给出大多数特殊字符和它们的含义：

^ 放在字符串前面，匹配行首的字；

$ 放在字符串后面，匹配行尾的字；

\< 匹配一个字的字头；

\> 匹配一个字的字尾；

. 匹配任何单个正文字符；

[str] 匹配 str 中的任何单个字符；

[^str] 匹配任何不在 str 中的单个字符；

[a-b] 匹配 a 到 b 之间的任一字符；

\* 匹配前一个字符的 0 次或多次出现；

\ 转义后面的字符。

简单介绍这么多，正则表达式知识可以参考

正则表达式基础语法摘要 [http://www.linuxidc.com/Linux/2016-11/136907.htm](https://www.linuxidc.com/Linux/2016-11/136907.htm)

正则表达式语法速查表  [http://www.linuxidc.com/Linux/2017-01/139561.htm](https://www.linuxidc.com/Linux/2017-01/139561.htm)

另外，进阶的Vim正则表达式还有对Magic 模式的介绍，可以参考  
《Vim正则表达式详解》:[http://www.linuxidc.com/Linux/2012-11/75113.htm](https://www.linuxidc.com/Linux/2012-11/75113.htm)

**4.5 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

:m,ns/str1/str2/g 将从m行到n行的str1替换成str2

从上述替换命令可以看到：  
1. g 放在命令末尾，表示对搜索字符串的每次出现进行替换,不止匹配每行中的第一次出现；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;  
2. s 表示后面跟着一串替换的命令；  
3. % 表示替换范围是所有行，即全文。

另外一个实用的命令，在Vim中统计当前文件中字符串 str1 出现的次数，可用替换命令的变形：

:%s/str1/&/gn

**更多详情见请继续阅读下一页的精彩内容**： [http://www.linuxidc.com/Linux/2017-02/140903p2.htm](https://www.linuxidc.com/Linux/2017-02/140903p2.htm)