

? ---->home

博客园 首页 新随笔 联系 订阅 管理

随笔 - 98 文章 - 1 评论 - 28 阅读 - 67万

公告

昵称: yjig
园龄: 10年11个月
粉丝: 59
关注: 12
+加关注

<	2021年10月						>
日	一	二	三	四	五	六	
26	27	28	29	30	1	2	
3	4	5	6	7	8	9	
10	11	12	13	14	15	16	
17	18	19	20	21	22	23	
24	25	26	27	28	29	30	
31	1	2	3	4	5	6	

搜索

找找看

谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

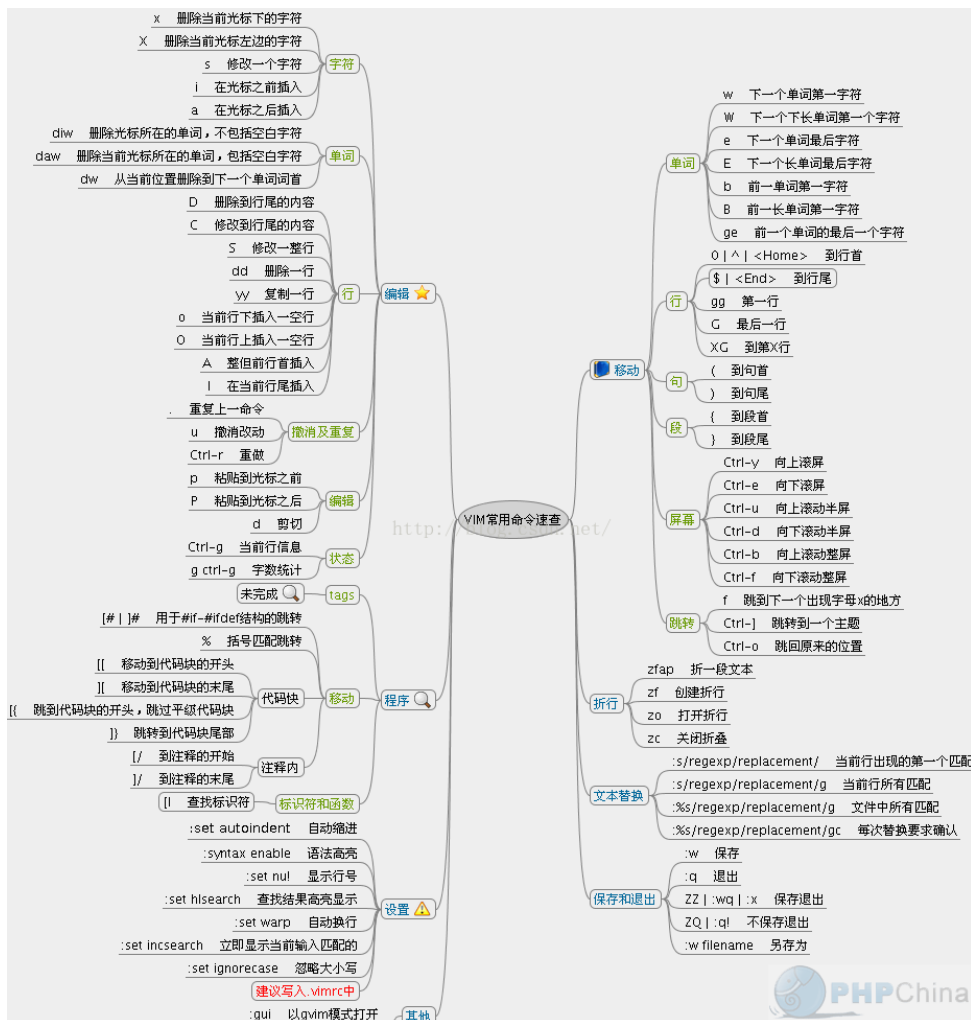
我的标签

linux(28)
c++(14)
ros(7)
vs(6)
vc(5)
vi(5)
proxmox(4)
dsm(4)
api(4)
mfc(4)
更多

随笔档案

2019年12月(1)
2019年11月(1)
2019年8月(1)
2018年12月(8)
2018年11月(25)
2018年3月(11)
2017年5月(2)
2016年12月(5)

vim常用命令总结 (转)



2016年11月(15)
2016年10月(2)
2016年3月(4)
2016年2月(3)
2014年8月(12)
2011年12月(2)
2011年11月(1)
更多

阅读排行榜

- 1. vim常用命令总结 (转)(481093)
- 2. VC MFC 屏蔽ESC和ENTER键关闭对话框(23634)
- 3. C++输出中文字符(转)(14622)
- 4. centos yum 完全卸载依赖(10477)
- 5. 解决WIN7下 Administrator 账户不能使用指纹的问题(8353)

评论排行榜

- 1. vim常用命令总结 (转)(20)
- 2. centos yum 完全卸载依赖(2)
- 3. vs vsvim viemu vax 备忘(2)
- 4. Windows窗口消息大全(转)(1)
- 5. C++输出中文字符(转)(1)

推荐排行榜

- 1. vim常用命令总结 (转)(61)
- 2. C++输出中文字符(转)(3)
- 3. centos yum 完全卸载依赖(1)
- 4. vim 查找替换(1)
- 5. vsvim _vsvimrc 设置 (转) (1)

最新评论

- 1. Re:vim常用命令总结 (转)
注册个账号来点赞
--Super_Jandroid
- 2. Re:centos yum 完全卸载依赖
漂亮
--sqrtcat
- 3. Re:vim常用命令总结 (转)
赞一个
--heshengquan
- 4. Re:vim常用命令总结 (转)
感谢, 好文点赞+转走 ;)
--Penicillin
- 5. Re:vim常用命令总结 (转)
w 向后一个单词
b 向前一个单词
--渔阳nice

Esc
命令
模式

排序:(例: 1-5行排序:光标放在1行上,然后!5G,然后sort)
字体设定:(set guifont=Raize\ Bold\ 13)
键绑定:(如: map h <Insert>,map <F2> hf) 按下F2后进入Insert模式并打印()

~ 转换大小写	! 外部过滤器	@ 执行寄存器	# 反向查找	\$ 行末	% 括号匹配	^ 行首	& :s//~	*	(句首) 下一句首	_ 前一行首	+ 次行首
1	2	3	4	5	6	7	8	9	0	- 前一行首	= 自动格式化	
Q 切换至ex模式	W 下一单词	E 单词尾	R 替换模式	T 替换字符	Y 复制行	U 撤销	I 到行首插入	O 分段(前)	P 粘贴(前)	{ 段首	}	段尾
q	w	e	r	t	y	u	i	o	p			
A 在行末附加	S 删除行并插入	D 删除至行末	F 行内字符反向查找	G 文件尾/行号	H 画面顶	J 合并两行	K 画面底	:	"			行首/列
a	s	d	f	g	h	j	k	:	"			
Z 退出	X 退格	C 修改至行末	V visual行模式	B 查找上一除	N 查找下一除	M 设定标记	< 反缩进	> 缩进	?	向前查找		
z	x	c	v	b	n	m						

动作 移动光标或欲定义操作的范围

指令 直接执行的指令

操作 后接用以表示操作范围的指令

extra 特殊功能需额外输入

主要ex指令: :w (保存)
:q (退出)
:e (打开文件) <Tab>
:%s/x/y/g (以y全文替换x)
:h (帮助)

其它重要指令:
Ctrl-R: 重做
Ctrl-F/B: 向前(下)翻页/向后(上)翻页
Ctrl-E/Y: 向前(下)一列/向后(上)一列
Ctrl-V: 切换visual模式

(1) 在复制/粘贴/删除 指令前使用 "x (x=a..z) 使用指令的寄存器(如: "ay\$ 复制该行目前位置到行尾的内容到寄存器'a')
(2) 命令前添加数字 重复指定次数的操作 (如: :2p, d2w, 5i, d4j)
(3) 重复光标所在字符处指定的查找 (dd = 删除本行, >>= 行首缩进)
(4) ZZ 保存并退出
(5) zb: 移动游标所在行至画面顶端, zb: 底端, zz: 中央
(6) gg: 文件首;ddp/P: 交换上下两行
daw: 删除单词;全选: ggVG

数字) n + >> :行共同缩进(如: >>>表示此行到下一行共同缩进)

在命令状态下对当前行用== (连接=两次), 或对多行用n== (n是自然数) 表示自动缩进从当前行起的下面n行。你可以试试把代码缩进任意打乱再用n==排版, 相当于一般IDE里的code format。使用gg=G可对整篇代码进行排版。

vim 选择文本, 删除, 复制, 粘贴

文本的选择, 对于编辑器来说, 是很基本的东西, 也经常用到, 总结如下:

v 从光标当前位置开始, 光标所经过的地方会被选中, 再按一下v结束。

V 从光标当前行开始, 光标经过的行都会被选中, 再按一下V结束。

Ctrl + v 从光标当前位置开始, 选中光标起点和终点所构成的矩形区域, 再按一下Ctrl + v结束。

ggVG 选中全部的文本, 其中gg为跳到行首, V选中整行, G末尾

选中后就可以用编辑命令对其进行编辑, 如

d 删除

y 复制 (默认是复制到"寄存器)

p 粘贴 (默认从"寄存器取出内容粘贴)

" + y 复制到系统剪贴板(也就是vim的+寄存器)

" + p 从系统剪贴板粘贴

=====
==

vim命令总结

1.删除字符

要删除一个字符，只需要将光标移到该字符上按下"x"。

2.删除一行

删除一整行内容使用"dd"命令。删除后下面的行会移上来填补空缺。

3.删除换行符

在Vim中你可以把两行合并为一行，也就是说两行之间的换行符被删除了：命令是"J"。

4.撤销

如果你误删了过多的内容。显然你可以再输入一遍，但是命令"u"更简便，它可以撤销上一次的操作。

5.重做

如果你撤销了多次，你还可以用CTRL-R(重做)来反转撤销的动作。换句话说，它是对撤销的撤销。撤销命令还有另一种形式，"U"命令，它一次撤销对一行的全部操作。第二次使用该命令则会撤销前一个"U"的操作。用"u"和CTRL-R你可以找回任何一个操作状态。

6.追加

"i"命令可以在当前光标之前插入文本。

"a"命令可以在当前光标之后插入文本。

"o"命令可以在当前行的下面另起一行，并使当前模式转为Insert模式。

"O"命令(注意是大写的字母O)将在当前行的上面另起一行。

7.使用命令计数

假设你要向上移动9行。这可以用"kkkkkkkkk"或"9k"来完成。事实上，很多命令都可以接受一个数字作为重复执行同一命令的次数。比如刚才的例子，要在行尾追加三个感叹号，当时用的命令是"a!!!". 另一个办法是用"3a!"命令。3说明该命令将被重复执行3次。同样，删除3个字符可以用"3x"。指定的数字要紧挨在它所要修饰的命令前面。

8.退出

要退出Vim，用命令"ZZ"。该命令保存当前文件并退出Vim。

9.放弃编辑

丢弃所有的修改并退出，用命令":q!". 用":e!"命令放弃所有修改并重新载入该文件的原始内容。

10.以Word为单位的移动

使用"w"命令可以将光标向前移动一个word的首字符上；比如"3w"将光标向前移动3个words。"b"命令则将光标向后移动到前一个word的首字符上。

"e"命令会将光标移动到下一个word的最后一个字符。命令"ge"，它将光标移动到前一个word的最后一个字符上。、

11.移动到行首或行尾

"\$"命令将光标移动到当前行行尾。如果你的键盘上有一个键，它的作用也一样。"^"命令将光标移动到当前行的第一个非空白字符上。"0"命令则总是把光标移动到当前行的第一个字符上。键也是如此。"\$"命令还可接受一个计数，如"1\$"会将光标移动到当前行行尾，"2\$"则会移动到下一行的行尾，如此类推。"0"命令却不能接受类似这样的计数，命令"^"前加上一个计数也没有任何效果。

12.移动到指定字符上

命令"fx"在当前行上查找下一个字符x（向右方向），可以带一个命令计数"F"命令向左方向搜索。"tx"命令形同"fx"命令，只不过它不是把光标停留在被搜索字符上，而是在它之前的一个字符上。提示："t"意为"To"。该命令的反方向版是"Tx"。这4个命令都可以用";"来重复。以","也是重复同

样的命令, 但是方向与原命令的方向相反。

13.以匹配一个括号为目的移动

用命令"`%`"跳转到与当前光标下的括号相匹配的那一个括号上去。如果当前光标在"`(`"上, 它就向前跳转到与它匹配的"`)`"上, 如果当前在"`)`"上, 它就向后自动跳转到匹配的"`(`"上去。

14.移动到指定行

用"`G`"命令指定一个命令计数, 这个命令就会把光标定位到由命令计数指定的行上。比如"`33G`"就会把光标置于第33行上。没有指定命令计数作为参数的话, "`G`"会把光标定位到最后一行上。"`gg`"命令是跳转到第一行的快捷的方法。

另一个移动到某行的方法是在命令"`%`"之前指定一个命令计数比如"`50%`"将会把光标定位在文件的中间。"`90%`"跳到接近文件尾的地方。

命令"`H`", "`M`", "`L`", 分别将光标跳转到第一行, 中间行, 结尾行部分。

15.告诉你当前的位置

使用CTRL-G命令。"`set number`"在每行的前面显示一个行号。相反关闭行号用命令"`:set nonumber`"。"`:set ruler`"在Vim窗口的右下角显示当前光标位置。

16.滚屏

CTRL-U显示文本的窗口向上滚动了半屏。CTRL-D命令将窗口向下移动半屏。一次滚动一行可以使用CTRL-E(向上滚动)和CTRL-Y(向下滚动)。要向前滚动一整屏使用命令CTRL-F。另外CTRL-B是它的反向版。"`zz`"命令会把当前行置为屏幕正中央, "`zt`"命令会把当前行置于屏幕顶端, "`zb`"则把当前行置于屏幕底端。

17.简单搜索

"`/string`"命令可用于搜索一个字符串。要查找上次查找的字符串的下一个位置,使用"`n`"命令。如果你知道你要找的确切位置是目标字符串的第几次出现, 还可以在"`n`"之前放置一个命令计数。"`3n`"会去查找目标字符串的第3次出现。

"`?`"命令与"`/`"的工作相同, 只是搜索方向相反。"`N`"命令会重复前一次查找, 但是与最初用"`/`"或"`?`"指定的搜索方向相反。

如果查找内容忽略大小写, 则用命令"`set ignorecase`", 返回精确匹配用命令"`set noignorecase`"。

18.在文本中查找下一个word

把光标定位于这个word上然后按下"`W`"键。Vim将会取当前光标所在的word并将它作用目标字符串进行搜索。"`#`"命令是"`W`"的反向版。还可以在这两个命令前加一个命令计数: "`3W`"查找当前光标下的word的第三次出现。

19.查找整个word

如果你用"`/the`"来查找Vim也会匹配到"`there`"。要查找作为独立单词的"`the`"使用如下命令: "`/the\>`"。"`\>`"是一个特殊的记法, 它只匹配一个word的结束处。近似地, "`\<`"匹配到一个word的开始处。这样查找作为一个word的"`the`"就可以用: "`/\`"。

20.高亮显示搜索结果

开启这一功能用"`:set hlsearch`", 关闭这一功能: "`:set nohlsearch`"。如果只是想去掉当前的高亮显示, 可以使用下面的命令: "`:nohlsearch`"(可以简写为`noh`)。

21.匹配一行的开头与结尾

`^` 字符匹配一行的开头。`$`字符匹配一行的末尾。

所以"`/was$`"只匹配位于一行末尾的单词was, 所以"`/^was`"只匹配位于一行开始的单词was。

22.匹配任何的单字符

这个字符可以匹配到任何字符。比如"`c.m`"可以匹配任何前一个字符是c, 后一个字符是m的情况, 不管中间的字符是什么。

23.匹配特殊字符

放一个反斜杠在特殊字符前面。如果你查找"ter.", 用命令"/ter\."

24.使用标记

当你用"G"命令从一个地方跳转到另一个地方时, Vim会记得你起跳的位置。这个位置在Vim中是一个标记。使用命令"``"可以使你跳回到刚才的出发点。

``命令可以在两点之间来回跳转。CTRL-O命令是跳转到你更早些时间停置光标的位置(提示:O意为older)。CTRL-I则是跳回到后来停置光标的更新的位置(提示: I在键盘上位于O前面)。

注:使用CTRL-I 与按下键一样。

25.具名标记

命令"ma"将当前光标下的位置名之为标记"a"。从a到z一共可以使用26个自定义的标记。要跳转到一个你定义过的标记, 使用命令" `marks "marks就是定义的标记的名字。命令" 'a "使你跳转到a所在行的行首, " `a "会精确定位a所在的位置。命令: ":marks"用来查看标记的列表。

命令delm! 删除所有标记。

26.操作符命令和位移

"dw"命令可以删除一个word, "d4w"命令是删除4个word, 依此类推。类似有"d2e"、"d\$"。此类命令有一个固定的模式: 操作符命令+位移命令。首先键入一个操作符命令。比如"d"是一个删除操作符。接下来是一个位移命。比如"w"。这样任何移动光标命令所及之处, 都是命令的作用范围。

27.改变文本

操作符命令是"c", 改变命令。它的行为与"d"命令类似, 不过在命令执行后会进入Insert模式。比如"cw"改变一个word。或者, 更准确地说, 它删除一个word并让你置身于Insert模式。

"cc"命令可以改变整行。不过仍保持原来的缩进。

"c\$"改变当前光标到行尾的内容。

快捷命令: x 代表dl(删除当前光标下的字符)

X 代表dh(删除当前光标左边的字符)

D 代表d\$(删除到行尾的内容)

C 代表c\$(修改到行尾的内容)

s 代表cl(修改一个字符)

S 代表cc(修改一整行)

命令"3dw"和"d3w"都是删除3个word。第一个命令"3dw"可以看作是删除一个word的操作执行3次; 第二个命令"d3w"是一次删除3个word。这是其中不明显的差异。事实上你可以在两处都放上命令记数, 比如, "3d2w"是删除两个word, 重复执行3次, 总共是6个word。

28.替换单个字符

"r"命令不是一个操作符命令。它等待你键入下一个字符用以替换当前光标下的那个字符。"r"命令前缀以一个命令记数是将多个字符都替换为即将输入的那个字符。要把一个字符替换为一个换行符使用"r"。它会删除一个字符并插入一个换行符。在此处使用命令记数只会删除指定个数的字符: "4r"将把4个字符替换为一个换行符。

29.重复改动

"."命令会重复上一次做出的改动。"."命令会重复你做出的所有修改, 除了"u"命令CTRL-R和以冒号开头的命令。"."需要在Normal模式下执行, 它重复的是命令, 而不是被改动的内容,

30.Visual模式

按"v"可以进入Visual模式。移动光标以覆盖你想操纵的文本范围。同时被选中的文本会以高亮显示。最后键入操作符命令。

31.移动文本

以"d"或"x"这样的命令删除文本时, 被删除的内容还是被保存了起来。你还可以用p命令把它取回来。"P"命令是把被去回的内容放在光标之前, "p"则是放在光标之后。对于以"dd"删除的整行内容, "P"会把它置于当前行的上一行。"p"则是至于当前行的后一行。也可以对命令"p"和"P"命令使用命令记数。它的效果是同样的内容被取回指定的次数。这样一来"dd"之后的"3p"就可以把被删除

行的3 份副本放到当前位置。

命令"xp"将光标所在的字符与后一个字符交换。

32.复制文本 (VIM编辑器内复制)

"y"操作符命令会把文本复制到一个寄存器中。然后可以用"p"命令把它取回。因为"y"是一个操作符命令, 所以你可以用"yw"来复制一个word. 同样可以使用命令记数。如下例中用"y2w"命令复制两个word, "yy"命令复制一整行, "Y"也是复制整行的内容, 复制当前光标至行尾的命令是"y\$".

33.文本对象

"diw" 删除当前光标所在的word(不包括空白字符) "daw" 删除当前光标所在的word(包括空白字符)

34.快捷命令

x 删除当前光标下的字符("dl"的快捷命令)
X 删除当前光标之前的字符("dh"的快捷命令)
D 删除自当前光标至行尾的内容("d\$"的快捷命令)
dw 删除自当前光标至下一个word的开头
db 删除自当前光标至前一个word的开始
diw 删除当前光标所在的word(不包括空白字符)
daw 删除当前光标所在的word(包括空白字符)
dG 删除当前行至文件尾的内容
dgg 删除当前行至文件头的内容

如果你用"c"命令代替"d"这些命令就都变成更改命令。使用"y"就是yank命令, 如此类推。

35.编辑另一个文件

用命令":edit foo.txt", 也可简写为":e foo.txt".

36.文件列表

可以在启动Vim时就指定要编辑多个文件, 用命令"vim one.c two.c three.c"。Vim将在启动后只显示第一个文件, 完成该文件的编辑后, 可以用令: ":next"或":n"要保存工作成果并继续下一个文件的编辑, 命令: ":wnext"或":wn"可以合并这一过程。

37.显示当前正在编辑的文件

用命令":args".

38.移动到另一个文件

用命令":previous" ":prev"回到上一个文件,合并保存步骤则是":wprevious" ":wprev"。要移到最后一个文件":last",到第一个":first".不过没有":wlast"或者":wfirst"这样的命令。可以在":next"和":previous"命令前面使用一个命令计数。

39.编辑另一个文件列表

不用重新启动Vim, 就可以重新定义一个文件列表。命令":args five.c six.c seven.h"定义了要编辑的三个文件。

39.自动存盘

命令":set autowrite", "set aw"。自动把内容写回文件: 如果文件被修改过, 在每个:next、:rewind、:last、:first、:previous、:stop、:suspend、:tag、:!、:make、CTRL-] 和 CTRL-^命令时进行。

命令":set autowriteall", "set awa"。和 'autowrite' 类似, 但也适用于":edit"、":enew"、":quit"、":qall"、":exit"、":xit"、":recover" 和关闭 Vim 窗口。置位本选项也意味着 Vim 的行为就像打开 'autowrite' 一样。

40.切换到另一文件

要在两个文件间快速切换, 使用CTRL-^。

41.文件标记

以大写字母命名的标记。它们是全局标记，它们可以用在任何文件中。比如，正在编辑"fab1.java",用命令"50%mF"在文件的中间设置一个名为F的标记。然后在"fab2.java"文件中，用命令"GnB"在最后一行设置名为B的标记。在可以用"F"命令跳转到文件"fab1.java"的半中间。或者编辑另一个文件，"B"命令会再把你带回文件"fab2.java"的最后一行。

要知道某个标记所代表的位置是什么，可以将该标记的名字作为"marks"命令的参数":marks M"或者连续跟上几个参数":marks MJK"

可以用CTRL-O和CTRL-I可以跳转到较早的位置和靠后的某位置。

42.查看文件

仅是查看文件，不向文件写入内容，可以用只读形式编辑文件。用命令：

vim -R file。如果是想强制性地避免对文件进行修改，可以用命令：

vim -M file。

43.更改文件名

将现有文件存成新的文件，用命令":sav(eas) move.c"。如果想改变当前正在编辑的文件名，但不想保存该文件，就可以用命令：":f(ile) move.c"。

44.分割一个窗口

打开一个新窗口最简单的办法就是使用命令：":split"。CTRL-W 命令可以切换当前活动窗口。

45.关闭窗口

用命令："close".可以关闭当前窗口。实际上,任何退出文件编辑的命令":quit"和"ZZ"都会关闭窗口，但是用":close"可以阻止你关闭最后一个Vim，以免以意外地整个关闭了Vim。

46.关闭除当前窗口外的所有其他窗口

用命令：":only",关闭除当前窗口外的所有其它窗口。如果这些窗口中有被修改过的，你会得到一个错误信息，同时那个窗口会被留下来。

47.为另一个文件分隔出一个窗口

命令":split two.c"可以打开第二个窗口同时在新打开的窗口中开始编辑作为参数的文件。如果要打开一个新窗口并开始编辑一个空的缓冲区，使用命令":new"。

48.垂直分割

用命令":vsplit或：:vsplit two.c"。同样有一个对应的":vnew"命令，用于垂直分隔窗口并在其中打开一个新的空缓冲区。

49.切换窗口

CTRL-W h 到左边的窗口

CTRL-W j 到下面的窗口

CTRL-W k 到上面的窗口

CTRL-W l 到右边的窗口

CTRL-W t 到顶部窗口

CTRL-W b 到底部窗口

50.针对所有窗口操作的命令

":qall"放弃所有操作并退出，":wall"保存所有，":wqall"保存所有并退出。

51.为每一个文件打开一个窗口

使用"-o"选项可以让Vim为每一个文件打开一个窗口：

"vim -o one.txt two.txt three.txt"。

52.使用vimdiff查看不同

"vimdiff main.c~ main.c",另一种进入diff模式的办法可以在Vim运行中操作。编辑文件"main.c", 然后打开另一个分隔窗口显示其不同:

```
":edit main.c"
":vertical diffpatch main.c.diff"。
```

53.页签

命令":tabe(dit) thatfile"在一个窗口中打开"thatfile", 该窗口占据着整个的Vim显示区域。命令":tab split/new"结果是新建了一个拥有一个窗口的页签。以用"gt"命令在不同的页签间切换。

本文转自: <http://fableking.iteye.com/blog/1141518>

这是我总结的一些基本用法,可能对初用者会有帮助,独乐乐不如众乐乐,是吧!

说明: 以下黑色为vi和vim均有的一般功能, 而红色为Vim (Vi Improved) 所特有功能。Vim一般的Unix和Linux下均有安装。

三种状态

Command: 任何输入都会作为编辑命令, 而不会出现在屏幕上, 任何输入都引起立即反映

Insert: 任何输入的数据都置于编辑寄存器, 按ESC, 可跳回command方式

Escape: 以 ":" 或者 "/" 为前导的指令, 出现在屏幕的最下一行, 任何输入都被当成特别指令。

离开vi

:q! 离开vi, 并放弃刚在缓冲区内编辑的内容。

:wq 将缓冲区内资料写入磁盘中, 并离开vi。

:x 同wq。

(注意—— :X 是文件加密, 一定要与:x存盘退出相区别)

进入输入模式

a (append) 由游标之后加入资料。

A 由该行之末加入资料。

i (insert) 由游标之前加入资料。

I 由该行之首加入资料。

o (open) 新增一行於该行之下供输入资料之用。

O 新增一行於该行之上供输入资料之用。

删除与修改

x 删除游标所在该字元。

X 删除游标所在之前一字元。

r 用接於此指令之后的字元取代(replace)游标所在字元。如: ra将游标所在字元以 a 取代之。

R 进入取代状态, 直到《ESC》为止。

s 删除游标所在之字元, 并进入输入模式直到《ESC》。

S 删除游标所在之该行资料, 并进入输入模式直到《ESC》。

光标的移动

m<a-z> 设置书签<a-z>

'<a-z> 移至书签<a-z>处

O 移至该行之首

\$ 移至该行之末。

e 移动到下个字的最後一个字母

w 移动到下个字的第一个字母。

b 移动到上个字的第一个字母。

^ 移至该行的第一个字元处。

H 移至视窗的第一行。

M 移至视窗的中间那行。

L 移至视窗的最后一行。

G 移至该文件的最后一行。

+ 移至下一列的第一个字元处。

- 移至上一列的第一个字元处。

:n 移至该文件的第 n 列。

n+ 移至游标所在位置之后的第 n 列。

n- 移至游标所在位置之前的第 n 列。

<Ctrl><g> 显示该行之行号、文件名称、文件中最末行之行号、游标所在行号占总行号之百分比。

(Vim) 光标移动基本用法小解:

(这只要组合上边的功能就可以明白了, 不用再一一讲解了吧!)

ge b w e

← ← ---→ --→

This is-a line, with special/separated/words (and some more).

←- ←-- -----→ ----→

GE B W E

视窗的移动

<Ctrl><f> 视窗往下卷一页。

<Ctrl> 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。

剪切、复制、删除

Operator + Scope = command

Operator

d 剪切

y 复制。

p 粘帖, 与 d 和 y 配和使用。可将最后d或y的资料放置於游标所在位置之行列下。

c 修改, 类似delete与insert的组和。删除一个字组、句子等之资料, 并插入新建资料。

Scope

e 由游标所在位置至该字串的最后一个字元。

w 由游标所在位置至下一个字串的第一个字元。

b 由游标所在位置至前一个字串的第一个字元。

\$ 由游标所在位置至该行的最后一个字元。

0 由游标所在位置至该行的第一个字元。

整行动作

dd 删除整行。

D 以行为单位, 删除游标后之所有字元。

cc 修改整行的内容。

yy 使游标所在该行复制到记忆体缓冲区。

取消前一动作(Undo)

u 恢复最后一个指令之前的结果。

U 恢复游标该行之所有改变。

(vim) u 可以多次撤消指令, 一次撤消一个操作, 直至本次操作开始为止。

(vim) Ctrl+r 可以恢复撤消前内容, 按多次可恢复多次。

查找与替换

/字串 往游标之后寻找该字串。

?字串 往游标之前寻找该字串。

n 往下继续寻找下一个相同的字串。

N 往上继续寻找下一个相同的字串。

% 查找 "(" , ")" , "{" , "}" 的配对符。

s 搜寻某行列范围。

g 搜寻整个编辑缓冲区的资料。

:1,\$s/old/new/g 将文件中所有的『old』改成『new』。

:10,20s/^/ / 将第10行至第20行资料的最前面插入5个空白。

(vim)

/字符串 后边输入查询内容可保存至缓冲区中, 可用↑↓进行以往内容选择。

另外：将光标移动在选定单词下方按*，则可以选中此单词作为查询字符，可以避免输入一长串字符的麻烦。

(vim) 大小写替换

首先用按v开启选择功能，然后用↑↓←→键来选定所要替换的字符，若是小写变大写，则按U;反之按u;

如果是选择单词，则可以在按v后，按w，最后按U/u,这样就可以将字符随意的改变大小写了，而不用删除后重新敲入。

资料的连接

J 句子的连接。将光标所在之下一行连接至光标该行的后面。

环境的设定

:set all 可设置的环境变量列表

:set 环境变量的当前值

:set nu 设定资料的行号。

:set nonu 取消行号设定。

:set ai 自动内缩。

:set noai 取消自动内缩。

(vim)

:set ruler 会在屏幕右下角显示当前光标所处位置，并随光移动而改变，占用屏幕空间较小，使用也比较方便，推荐使用。

:set hlsearch 在使用查找功能时，会高亮显示所有匹配的内容。

:set nohlsearch 关闭此功能。

:set incsearch 使Vim在输入字符串的过程中，光标就可定位显示匹配点。

:set nowrapscan 关闭查找自动回环功能，即查找到文件结尾处，结束查找；默认状态是自动回环

ex指令

读写资料

:10,20w test 将第10行至第20行的资料写入test文件。

:10,20w > test 将第10行至第20行的资料加在test文件之后。

:r test 将test文件的资料读入编辑缓冲区的最后。

:e [filename] 编辑新的文件。

:e! [filename] 放弃当前修改的文件，编辑新的文件。

:sh 进入shell环境，使用exit退出，回到编辑器中。

!cmd 运行命令cmd后，返回到编辑器中。

删除、复制及搬移

:10,20d 删除第10行至第20行的资料。

:10d 删除第10行的资料。

:%d 删除整个编辑缓冲区。

:10,20co30 将第10行至第20行的资料复制至第30行之后。

:10,20mo30 将第10行至第20行的资料搬移至第30行之后。

标签: [vi](#)



yjig
关注 - 12
粉丝 - 59

+加关注

61

推荐

0


反对

« 上一篇: [Windows窗口消息大全\(转\)](#)

» 下一篇: [vsvim vsvimrc 设置 \(转\)](#)

posted @ 2016-10-30 22:01 yjig 阅读(481112) 评论(20) 编辑 收藏 举报

[刷新评论](#) [刷新页面](#) [返回顶部](#)

 登录后才能查看或发表评论, 立即 [登录](#) 或者 [逛逛](#) 博客园首页

【推荐】并行超算云面向博客园粉丝推出“免费算力限时申领”特别活动

【推荐】跨平台组态\工控\仿真\CAD 50万行C++源码全开放免费下载!

【推荐】和开发者在一起: 华为开发者社区, 入驻博客园科技品牌专区



编辑推荐:

- 五个维度打造研发管理体系
- 不会SQL也能做数据分析? 浅谈语义解析领域的机会与挑战
- Spring IoC Container 原理解析
- 前端实现的浏览器端扫码功能
- ASP.NET Core Filter 与 IOC 的羁绊

最新新闻:

- Valve 推出 Steam Deck 兼容性项目 (2021-10-20 09:16)
 - 获得硅谷青睐的新信用卡 (2021-10-20 09:10)
 - 信用卡 PIN 码很容易猜测 (2021-10-20 09:02)
 - 神经元簇发能模拟 AI 学习策略 (2021-10-20 08:55)
 - 蜘蛛丝可能根本不具有抗菌性质 (2021-10-20 08:48)
- » 更多新闻...

Copyright © 2021 yjig

Powered by .NET 6 on Kubernetes