|  |  |
| --- | --- |
| 移动光标的方法 | |
| h 或 向左箭头键(←) | 光标向左移动一个字符 |
| j 或 向下箭头键(↓) | 光标向下移动一个字符 |
| k 或 向上箭头键(↑) | 光标向上移动一个字符 |
| l 或 向右箭头键(→) | 光标向右移动一个字符 |
| 如果你将右手放在键盘上的话，你会发现 hjkl 是排列在一起的，因此可以使用这四个按钮来移动光标。 如果想要进行多次移动的话，例如向下移动 30 行，可以使用 "30j" 或 "30↓" 的组合按键， 亦即加上想要进行的次数(数字)后，按下动作即可！ | |
| [Ctrl] + [f] | 屏幕『向下』移动一页，相当于 [Page Down]按键 (常用) |
| [Ctrl] + [b] | 屏幕『向上』移动一页，相当于 [Page Up] 按键 (常用) |
| [Ctrl] + [d] | 屏幕『向下』移动半页 |
| [Ctrl] + [u] | 屏幕『向上』移动半页 |
| + | 光标移动到非空格符的下一列 |
| - | 光标移动到非空格符的上一列 |
| n<space> | 那个 n 表示『数字』，例如 20 。按下数字后再按空格键，光标会向右移动这一行的 n 个字符。例如 20<space> 则光标会向后面移动 20 个字符距离。 |
| 0 或功能键[Home] | 这是数字『 0 』：移动到这一行的最前面字符处 (常用) |
| $ 或功能键[End] | 移动到这一行的最后面字符处(常用) |
| H | 光标移动到这个屏幕的最上方那一行的第一个字符 |
| M | 光标移动到这个屏幕的中央那一行的第一个字符 |
| L | 光标移动到这个屏幕的最下方那一行的第一个字符 |
| G | 移动到这个档案的最后一行(常用) |
| nG | n 为数字。移动到这个档案的第 n 行。例如 20G 则会移动到这个档案的第 20 行(可配合 :set nu) |
| gg | 移动到这个档案的第一行，相当于 1G 啊！ (常用) |
| n<Enter> | n 为数字。光标向下移动 n 行(常用) |
| 搜寻与取代 | |
| /word | 向光标之下寻找一个名称为 word 的字符串。例如要在档案内搜寻 vbird 这个字符串，就输入 /vbird 即可！ (常用) |
| ?word | 向光标之上寻找一个字符串名称为 word 的字符串。 |
| n | 这个 n 是英文按键。代表『重复前一个搜寻的动作』。举例来说， 如果刚刚我们执行 /vbird 去向下搜寻 vbird 这个字符串，则按下 n 后，会向下继续搜寻下一个名称为 vbird 的字符串。如果是执行 ?vbird 的话，那么按下 n 则会向上继续搜寻名称为 vbird 的字符串！ |
| N | 这个 N 是英文按键。与 n 刚好相反，为『反向』进行前一个搜寻动作。 例如 /vbird 后，按下 N 则表示『向上』搜寻 vbird 。 |
| 使用 /word 配合 n 及 N 是非常有帮助的！可以让你重复的找到一些你搜寻的关键词！ | |
| :n1,n2s/word1/word2/g | n1 与 n2 为数字。在第 n1 与 n2 行之间寻找 word1 这个字符串，并将该字符串取代为 word2 ！举例来说，在 100 到 200 行之间搜寻 vbird 并取代为 VBIRD 则： 『:100,200s/vbird/VBIRD/g』。(常用) |
| :1,$s/word1/word2/g | 从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！(常用) |
| :1,$s/word1/word2/gc | 从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！且在取代前显示提示字符给用户确认 (confirm) 是否需要取代！(常用) |
| 删除、复制与贴上 | |
| x, X | 在一行字当中，x 为向后删除一个字符 (相当于 [del] 按键)， X 为向前删除一个字符(相当于 [backspace] 亦即是退格键) (常用) |
| nx | n 为数字，连续向后删除 n 个字符。举例来说，我要连续删除 10 个字符， 『10x』。 |
| dd | 删除游标所在的那一整列(常用) |
| ndd | n 为数字。删除光标所在的向下 n 列，例如 20dd 则是删除 20 列 (常用) |
| d1G | 删除光标所在到第一行的所有数据 |
| dG | 删除光标所在到最后一行的所有数据 |
| d$ | 删除游标所在处，到该行的最后一个字符 |
| d0 | 那个是数字的 0 ，删除游标所在处，到该行的最前面一个字符 |
| yy | 复制游标所在的那一行(常用) |
| nyy | n 为数字。复制光标所在的向下 n 列，例如 20yy 则是复制 20 列(常用) |
| y1G | 复制游标所在列到第一列的所有数据 |
| yG | 复制游标所在列到最后一列的所有数据 |
| y0 | 复制光标所在的那个字符到该行行首的所有数据 |
| y$ | 复制光标所在的那个字符到该行行尾的所有数据 |
| p, P | p 为将已复制的数据在光标下一行贴上，P 则为贴在游标上一行！ 举例来说，我目前光标在第 20 行，且已经复制了 10 行数据。则按下 p 后， 那 10 行数据会贴在原本的 20 行之后，亦即由 21 行开始贴。但如果是按下 P 呢？ 那么原本的第 20 行会被推到变成 30 行。 (常用) |
| J | 将光标所在列与下一列的数据结合成同一列 |
| c | 重复删除多个数据，例如向下删除 10 行，[ 10cj ] |
| u | 复原前一个动作。(常用) |
| [Ctrl]+r | 重做上一个动作。(常用) |
| 这个 u 与 [Ctrl]+r 是很常用的指令！一个是复原，另一个则是重做一次～ 利用这两个功能按键，你的编辑，嘿嘿！很快乐的啦！ | |
| . | 不要怀疑！这就是小数点！意思是重复前一个动作的意思。 如果你想要重复删除、重复贴上等等动作，按下小数点『.』就好了！ (常用) |

* **第二部份：一般模式切换到编辑模式的可用的按钮说明**

|  |  |
| --- | --- |
| 进入插入或取代的编辑模式 | |
| i, I | 进入插入模式(Insert mode)： i 为『从目前光标所在处插入』， I 为『在目前所在行的第一个非空格符处开始插入』。 (常用) |
| a, A | 进入插入模式(Insert mode)： a 为『从目前光标所在的下一个字符处开始插入』， A 为『从光标所在行的最后一个字符处开始插入』。(常用) |
| o, O | 进入插入模式(Insert mode)： 这是英文字母 o 的大小写。o 为『在目前光标所在的下一行处插入新的一行』； O 为在目前光标所在处的上一行插入新的一行！(常用) |
| r, R | 进入取代模式(Replace mode)： r 只会取代光标所在的那一个字符一次；R会一直取代光标所在的文字，直到按下 ESC 为止；(常用) |
| 上面这些按键中，在 vi 画面的左下角处会出现『--INSERT--』或『--REPLACE--』的字样。 由名称就知道该动作了吧！！特别注意的是，我们上面也提过了，你想要在档案里面输入字符时， 一定要在左下角处看到 INSERT 或 REPLACE 才能输入喔！ | |
| [Esc] | 退出编辑模式，回到一般模式中(常用) |

* **第三部份：一般模式切换到指令列模式的可用的按钮说明**

|  |  |
| --- | --- |
| 指令列的储存、离开等指令 | |
| :w | 将编辑的数据写入硬盘档案中(常用) |
| :w! | 若文件属性为『只读』时，强制写入该档案。不过，到底能不能写入， 还是跟你对该档案的档案权限有关啊！ |
| :q | 离开 vi (常用) |
| :q! | 若曾修改过档案，又不想储存，使用 ! 为强制离开不储存档案。 |
| 注意一下啊，那个惊叹号 (!) 在 vi 当中，常常具有『强制』的意思～ | |
| :wq | 储存后离开，若为 :wq! 则为强制储存后离开 (常用) |
| ZZ | 这是大写的 Z 喔！若档案没有更动，则不储存离开，若档案已经被更动过，则储存后离开！ |
| :w [filename] | 将编辑的数据储存成另一个档案（类似另存新档） |
| :r [filename] | 在编辑的数据中，读入另一个档案的数据。亦即将 『filename』 这个档案内容加到游标所在行后面 |
| :n1,n2 w [filename] | 将 n1 到 n2 的内容储存成 filename 这个档案。 |
| :! command | 暂时离开 vi 到指令列模式下执行 command 的显示结果！例如 『:! ls /home』即可在 vi 当中察看 /home 底下以 ls 输出的档案信息！ |
| vim 环境的变更 | |
| :set nu | 显示行号，设定之后，会在每一行的前缀显示该行的行号 |
| :set nonu | 与 set nu 相反，为取消行号！ |

特别注意，在 vi 中，『数字』是很有意义的！数字通常代表重复做几次的意思！ 也有可能是代表去到第几个什么什么的意思。举例来说，要删除 50 行，则是用 『50dd』 对吧！ 数字加在动作之前～那我要向下移动 20 行呢？那就是『20j』或者是『20↓』即可。

OK！会这些指令就已经很厉害了，因为常用到的指令也只有不到一半！通常 vi 的指令除了上面鸟哥注明的常用的几个外，其他是不用背的，你可以做一张简单的指令表在你的屏幕墙上， 一有疑问可以马上的查询呦！这也是当初鸟哥使用 vim 的方法啦！

**区块选择(Visual Block)**

刚刚我们提到的简单的 vi 操作过程中，几乎提到的都是以行为单位的操作。那么如果我想要搞定的是一个区块范围呢？ 举例来说，像底下这种格式的档案：

|  |
| --- |
| 192.168.1.1 host1.class.net  192.168.1.2 host2.class.net  192.168.1.3 host3.class.net  192.168.1.4 host4.class.net  .....中间省略...... |

这个档案我将他放置到 <http://cn.linux.vbird.org/linux_basic/0310vi/hosts> ，你可以自行下载来看一看这个档案啊！现在我们来玩一玩这个档案吧！假设我想要将 host1, host2... 等等复制起来， 并且加到每一行的后面，亦即每一行的结果要是『 192.168.1.2 host2.class.net host2 』这样的情况时， 在传统或现代的窗口型编辑器似乎不容易达到这个需求，但是咱们的 vim 是办的到的喔！那就使用区块选择 (Visual Block) 吧！当我们按下 v 或者 V 或者 [Ctrl]+v 时， 这个时候光标移动过的地方就会开始反白，这三个按键的意义分别是：

|  |  |
| --- | --- |
| 区块选择的按键意义 | |
| v | 字符选择，会将光标经过的地方反白选择！ |
| V | 行选择，会将光标经过的行反白选择！ |
| [Ctrl]+v | 区块选择，可以用长方形的方式选择资料 |
| y | 将反白的地方复制起来 |
| d | 将反白的地方删除掉 |

假设一个例子，你想要将刚刚我们的 hosts 内的 IP 复制到你的 /etc/hosts 这个档案去， 那么该如何编辑？我们知道在 vi 内可以使用 :r filename 来读入某个档案的内容， 不过，这样毕竟是将整个档案读入啊！如果我只是想要部分内容呢？呵呵！这个时候多档案同时编辑就很有用了。 我们可以使用 vim 后面同时接好几个档案来同时开启喔！相关的按键有：

|  |  |
| --- | --- |
| 多档案编辑的按键 | |
| :n | 编辑下一个档案 |
| :N | 编辑上一个档案 |
| :files | 列出目前这个 vim 的开启的所有档案 |

在过去，鸟哥想要将 A 档案内的十条消息『移动』到 B 档案去，通常要开两个 vim 窗口来复制， 偏偏每个 vim 都是独立的，因此并没有办法在 A 档案下达『 nyy 』再跑到 B 档案去『 p 』啦！ 在这种情况下最常用的方法就是透过鼠标圈选， 复制后贴上。不过这样一来还是有问题，因为鸟哥超级喜欢使用 [Tab] 按键进行编排对齐动作， 透过鼠标却会将 [Tab] 转成空格键，这样内容就不一样了！此时这个多档案编辑就派上用场了！

|  |  |
| --- | --- |
| 多窗口情况下的按键功能 | |
| :sp [filename] | 开启一个新窗口，如果有加 filename， 表示在新窗口开启一个新档案，否则表示两个窗口为同一个档案内容(同步显示)。 |
| [ctrl]+w+ j [ctrl]+w+↓ | 按键的按法是：先按下 [ctrl] 不放， 再按下 w 后放开所有的按键，然后再按下 j (或向下箭头键)，则光标可移动到下方的窗口。 |
| [ctrl]+w+ k [ctrl]+w+↑ | 同上，不过光标移动到上面的窗口。 |
| [ctrl]+w+ q | 其实就是 :q 结束离开啦！ 举例来说，如果我想要结束下方的窗口，那么利用 [ctrl]+w+↓ 移动到下方窗口后，按下 :q 即可离开， 也可以按下 [ctrl]+w+q 啊！ |

**vim 环境设定与记录： ~/.vimrc, ~/.viminfo**

有没有发现，如果我们以 vim 软件来搜寻一个档案内部的某个字符串时，这个字符串会被反白， 而下次我们再次以 vim 编辑这个档案时，该搜寻的字符串反白情况还是存在呢！甚至于在编辑其他档案时， 如果其他档案内也存在这个字符串，哇！竟然还是主动反白耶！真神奇！ 另外，当我们重复编辑同一个档案时，当第二次进入该档案时， 游标竟然就在上次离开的那一行上头呢！真是好方便啊～但是，怎么会这样呢？

这是因为我们的 vim 会主动的将你曾经做过的行为登录下来，好让你下次可以轻松的作业啊！ 那个记录动作的档案就是： ~/.viminfo ！如果你曾经使用过 vim， 那你的家目录应该会存在这个档案才对。这个档案是自动产生的， 你不必自行建立。而你在 vim 里头所做过的动作，就可以在这个档案内部查询到啰～ ^\_^

此外，每个 distributions 对 vim 的预设环境都不太相同，举例来说，某些版本在搜寻到关键词时并不会高亮度反白， 有些版本则会主动的帮你进行缩排的行为。但这些其实都可以自行设定的，那就是 vim 的环境设定啰～ vim 的环境设定参数有很多，如果你想要知道目前的设定值，可以在一般模式时输入『 :set all 』 来查阅，不过.....设定项目实在太多了～所以，鸟哥在这里仅列出一些平时比较常用的一些简单的设定值， 提供给你参考啊。

|  |  |
| --- | --- |
| **Tips:** 所谓的缩排，就是当你按下 Enter 编辑新的一行时，光标不会在行首，而是在与上一行的第一个非空格符处对齐！ | 鸟哥的图示 |

|  |  |
| --- | --- |
| vim 的环境设定参数 | |
| :set nu :set nonu | 就是设定与取消行号啊！ |
| :set hlsearch :set nohlsearch | hlsearch 就是 high light search(高亮度搜寻)。 这个就是设定是否将搜寻的字符串反白的设定值。默认值是 hlsearch |
| :set autoindent :set noautoindent | 是否自动缩排？autoindent 就是自动缩排。 |
| :set backup | 是否自动储存备份档？一般是 nobackup 的， 如果设定 backup 的话，那么当你更动任何一个档案时，则源文件会被另存成一个档名为 filename~ 的档案。 举例来说，我们编辑 hosts ，设定 :set backup ，那么当更动 hosts 时，在同目录下，就会产生 hosts~ 文件名的档案，记录原始的 hosts 档案内容 |
| :set ruler | 还记得我们提到的右下角的一些状态栏说明吗？ 这个 ruler 就是在显示或不显示该设定值的啦！ |
| :set showmode | 这个则是，是否要显示 --INSERT-- 之类的字眼在左下角的状态栏。 |
| :set backspace=(012) | 一般来说， 如果我们按下 i 进入编辑模式后，可以利用退格键 (backspace) 来删除任意字符的。 但是，某些 distribution 则不许如此。此时，我们就可以透过 backspace 来设定啰～ 当 backspace 为 2 时，就是可以删除任意值；0 或 1 时，仅可删除刚刚输入的字符， 而无法删除原本就已经存在的文字了！ |
| :set all | 显示目前所有的环境参数设定值。 |
| :set | 显示与系统默认值不同的设定参数， 一般来说就是你有自行变动过的设定参数啦！ |
| :syntax on :syntax off | 是否依据程序相关语法显示不同颜色？ 举例来说，在编辑一个纯文本档时，如果开头是以 # 开始，那么该行就会变成蓝色。 如果你懂得写程序，那么这个 :syntax on 还会主动的帮你除错呢！但是， 如果你仅是编写纯文本档案，要避免颜色对你的屏幕产生的干扰，则可以取消这个设定 。 |
| :set bg=dark :set bg=light | 可用以显示不同的颜色色调，预设是『 light 』。如果你常常发现批注的字体深蓝色实在很不容易看， 那么这里可以设定为 dark 喔！试看看，会有不同的样式呢！ |

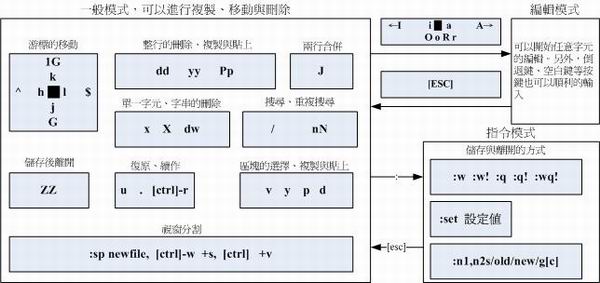
总之，这些设定值很有用处的啦！但是......我是否每次使用 vim 都要重新设定一次各个参数值？ 这不太合理吧？没错啊！所以，我们可以透过配置文件来直接规定我们习惯的 vim 操作环境呢！ 整体 vim 的设定值一般是放置在 /etc/vimrc 这个档案，不过，不建议你修改他！ 你可以修改 ~/.vimrc 这个档案 (预设不存在，请你自行手动建立！)，将你所希望的设定值写入！ 举例来说，可以是这样的一个档案：

|  |
| --- |
| [root@www ~]# **vim ~/.vimrc**  "这个档案的双引号 (") 是批注  set hlsearch "高亮度反白  set backspace=2 "可随时用退格键删除  set autoindent "自动缩排  set ruler "可显示最后一行的状态  set showmode "左下角那一行的状态  set nu "可以在每一行的最前面显示行号啦！  set bg=dark "显示不同的底色色调  syntax on "进行语法检验，颜色显示。 |

在这个档案中，使用『 set hlsearch 』或『 :set hlsearch 』，亦即最前面有没有冒号『 : 』效果都是一样的！ 至于双引号则是批注符号！不要用错批注符号，否则每次使用 vim 时都会发生警告讯息喔！ 建立好这个档案后，当你下次重新以 vim 编辑某个档案时，该档案的预设环境设定就是上头写的啰～ 这样，是否很方便你的操作啊！多多利用 vim 的环境设定功能呢！^\_^

**vim 常用指令示意图**

为了方便大家查询在不同的模式下可以使用的 vim 指令，鸟哥查询了一些 vim 与 Linux 教育训练手册， 发现底下这张图非常值得大家参考！可以更快速有效的查询到需要的功能喔！看看吧！

  
图 3.5.1 、 vim 常用指令示意图