* **第一部份：一般模式可用的按鈕說明，游標移動、複製貼上、搜尋取代等**

|  |  |
| --- | --- |
| 移動游標的方法 | |
| n<h> 或 向左方向鍵(←) | 游標向左移動一個字元 |
| n<j> 或 向下方向鍵(↓) | 游標向下移動一個字元 |
| n<k> 或 向上方向鍵(↑) | 游標向上移動一個字元 |
| n<l> 或 向右方向鍵(→) | 游標向右移動一個字元 |
| gh, gj, gk, gl | 没有换行符的多行跳转，句子太长造成地多行。 |
| 如果你將右手放在鍵盤上的話，你會發現 hjkl 是排列在一起的，因此可以使用這四個按鈕來移動游標。 如果想要進行多次移動的話，例如向下移動 30 行，可以使用 "30j" 或 "30↓" 的組合按鍵， 亦即加上想要進行的次數(數字)後，按下動作即可！ | |
| [Ctrl] + [f] | 螢幕『向下』移動一頁，相當於 [Page Down]按鍵 (常用) |
| [Ctrl] + [b] | 螢幕『向上』移動一頁，相當於 [Page Up] 按鍵 (常用) |
| [Ctrl] + [d] | 螢幕『向下』移動半頁 |
| [Ctrl] + [u] | 螢幕『向上』移動半頁 |
| [Ctrl] + [y] | 螢幕『向下』移動一行 |
| [Ctrl] + [e] | 螢幕『向上』移动一行 |
| [Ctrl] + [p] | 文本框，命令模式， 向上一行(同k) |
| [Ctrl] + [n]/ [Ctrl] + [m]/ [Ctrl] + [j] | 文本框，命令模式， 向下一行(同j) |
| H | 游標移動到這個螢幕的最上方那一行的第一個字元 |
| M | 游標移動到這個螢幕的中央那一行的第一個字元 |
| L | 游標移動到這個螢幕的最下方那一行的第一個字元 |
| gg | 移動到這個檔案的第一行，相當於 1G 啊！ (常用) |
| G | 移動到這個檔案的最後一行(常用) |
| nG/ ngg/ :n | n 為數字。移動到這個檔案的第 n 行。例如 20G 則會移動到這個檔案的第 20 行(可配合 :set nu， 显示行数在左边) |
| :0 | 第一行位置 |
| :%,:$ | 最后一行位置 |
| n% | 移动到全部的百分之n行位置 |
| n+, n- | n 為數字。向下向上移动多少行。 (同nj,nk) |
| + | 游標移動到非空白字元的下一列首字母 |
| - | 游標移動到非空白字元的上一列首字母 |
| [Ctrl]+[+] | Vim窗口字体变大 |
| [Ctrl]+[-] | Vim窗口字体变小 |
| n<Enter> | n 為數字。游標向下移動 n 行(常用) |
| n<space> | 那個 n 表示『數字』，例如 20 。按下數字後再按空白鍵，游標會向右移動這一行的 n 個字元。例如 20<space> 則游標會向後面移動 20 個字元距離。 |
| n<backspace> | 那個 n 表示『數字』，例如 20 。按下數字後再按空白鍵，游標會向左移動這一行的 n 個字元。例如 20<space> 則游標會向後面移動 20 個字元距離。 |
| 0 或功能鍵[Home] | 這是數字『 0 』：移動到這一行的最前面字元處 (常用) |
| g0 | 没有换行符的多行跳转，句子太长造成地多行。屏幕本行头 |
| ^ | 移动到本行首字母，空白字符不算 |
| g^ | 没有换行符的多行跳转，句子太长造成地多行。屏幕本行非空白字符 |
| $ 或功能鍵[End] | 移動到這一行的最後面字元處(常用) |
| g$ | 没有换行符的多行跳转，句子太长造成地多行。屏幕本行尾 |
| g\_ | 移动到本行最后一个非空白字符 |
| E/e | 移动到本单词最后一个字符(E大跳，log-log-log视为一个单词)  下划线不是单词分隔符 |
| B/b | 移动到本单词第一个字符 (B大跳，log-log-log视为一个单词) |
| W/w | 移动到下一个单词开始 (W大跳，log-log-log视为一个单词) |
| gE/ge | 移动到上一个单词结束 (gE大跳，log-log-log视为一个单词) |
| f/Fx | 向后/前查找某字符，光标停在x字符上 eg. ctx, dfx, d2fx, y2fx |
| t/Tx | 向后/前查找某字符，光标停在x字符前 eg. cfx, dtx, d2tx, y2tx |
| y2/foo | 复制到第二个foo之前 |
| ;/, | 重复f/F t/T命令 |
| n| | 移到第n列 |
| 跳转快捷键(跨文件的) | |
| [Ctrl]+o | 跳转到下一个位置 |
| [Ctrl]+i/ tab | 跳转到上一个位置 |
| :jumps | 显示跳转点 |
| 操作当前单词 | |
| <动作>i<motion> | 不包含空白和符号 eg:vi”,vi},vi) ,v2i), yiw, diw, ciw, viw |
| <动作>a<motion> | 包含空白和符号 eg: va”,va},va) ,v2a), yaw, daw, caw, vaw |
| 搜尋與取代 | |
| /word | 向游標之下尋找一個名稱為 word 的字串。例如要在檔案內搜尋 vbird 這個字串，就輸入 /vbird 即可！ (常用) |
| ?word | 向游標之上尋找一個字串名稱為 word 的字串。 |
| [Ctrl]+p | 查找模式输入上一个匹配的查找输入字符串 eg /,?, 命令模式补齐输入过的命令 eg : |
| [Ctrl]+n | 查找模式输入下一个匹配的查找输入字符串 eg /,?, 命令模式补齐输入过的命令 eg : |
| /pattern/+n | 将光标放在包含pattern的后第n行上 |
| /pattern/-n | 将光标放在包含pattern的前第n行上 |
| n | 這個 n 是英文按鍵。代表『重複前一個搜尋的動作』。舉例來說， 如果剛剛我們執行 /vbird 去向下搜尋 vbird 這個字串，則按下 n 後，會向下繼續搜尋下一個名稱為 vbird 的字串。如果是執行 ?vbird 的話，那麼按下 n 則會向上繼續搜尋名稱為 vbird 的字串！ |
| N | 這個 N 是英文按鍵。與 n 剛好相反，為『反向』進行前一個搜尋動作。 例如 /vbird 後，按下 N 則表示『向上』搜尋 vbird 。 |
| 使用 /word 配合 n 及 N 是非常有幫助的！可以讓你重複的找到一些你搜尋的關鍵字！ | |
| :n1,n2s/word1/word2/g | n1 與 n2 為數字。在第 n1 與 n2 行之間尋找 word1 這個字串，並將該字串取代為 word2 ！舉例來說，在 100 到 200 行之間搜尋 vbird 並取代為 VBIRD 則： 『:100,200s/vbird/VBIRD/g』。(常用) |
| :1,$s/word1/word2/g | 從第一行到最後一行尋找 word1 字串，並將該字串取代為 word2 ！(常用) |
| :%s/word1/word2/gc | 從第一行到最後一行尋找 word1 字串，並將該字串取代為 word2 ！且在取代前顯示提示字元給使用者確認 (confirm) 是否需要取代！(常用) |
| :%s/word1/word2/gic | i忽略大小写 |
| :,$s/word1/word2/gc | 從当前行到最後一行尋找 word1 字串，並將該字串取代為 word2 ！且在取代前顯示提示字元給使用者確認 (confirm) 是否需要取代！(常用) |
| .,.+[n] | 当前行，到当前行下面的n行范围。eg： .,.+33 |
| ‘< | Start of the select |
| ‘> | End of the select |
| #, \* | #向上查找光标处的单词，\*向下查找光标处的单词，完全匹配 |
| g#, g\* | #向上查找光标处的单词，\*向下查找光标处的单词，部分匹配 |
| % | 匹配(),[],{},#if, #else, #endif |
| :set matchpairs=(:),{:},[:],<:>,?:\: | 设置%匹配的元素 |
| **模式查找 pattern** | |
| : [range] g/pattern/command | 匹配某个模式后，执行某个命令 |
| :%g/^ xyz/normal dd | 查找^ xyz开头的行 然后执行nornal 模式下的dd |
| :g/abc/p | 显示有abc的行 |
| :v/abc/p | 显示没有abc的行 |
| :g/abc/d | 删除有abc的行 |
| :v/abc/d | 删除没有abc的行 |
| **正则表达式** | |
| \< , \> | 搜索栏的内容实际格式 |
| /[0-9]\*\.[0-9][0-9] | xxx.xx格式匹配 |
| ：% s/\s\+$//gc | 删除尾部空白符号 \s：空白符号 \+：-个或多个 |
| ：% s/\(\s\*\n\)\+/\r/gc | 删除空白行，\r不能少, 换成/\n/,//都会变成连接邻近两行的效果(J) |
| ：% s!\s\*//.\*!!gc | 删除//开头的注释行， ！代替/ 所以斜杠不用转义了， .: 除换行以为的任意字符 \*：表示0个或者任意多个（最大匹配） |
| ：% s!\s\*/\\*\\_.\{-}\\*/\s\*!!gc | 删除/\* \*/注释， \\_.：包含换行的任意字符， \{-}任意多个（最小匹配） |
| \d | 表示十进制数字(猜测) |
| \s | 空白字符 |
| \S | 非空白字符 |
| \a | 英文字母 |
| \| | 表示或 |
| \. | 表示. |
| {m,n} | 表示m个到n个字符，要和\s和\a等连用。 |
| {m,} | 表示m个到无穷个字符 |
| \*\* | 标示当前目录下的所有子目录 |
| 块跳转 | |
| {, } | 上下空白行跳转，文字块 |
| [{, }} | 当前块开始处，当前块结尾处 |
| [[, ]], [], ][ | 前一个函数{, 后一个函数{, 前一个函数}, 后一个函数{ |
| [/,]/ | 当前注释块的开始处，当前注释块的结束处 |
| (, ) | 函数语句跳转 |
| changelist | |
| :changes | 显示changelist |
| g; | 上个修改过的地方，上一个changelist |
| g, | 下个修改过的地方，下一个changelist |
| 格式化 | |
| n== | 光标所在行开始的n行自动对齐 |
| n<< | 光标所在行开始的n行向左缩进一个tab |
| n>> | 光标所在行开始的n行向右缩进一个tab |
| gg=G | 第一行开始自动对齐全文 |
| =G | 光标所在行开始到文件尾部，自动对齐 |
| =% | 光标先放到语句块起始位置，语句块对齐 |
| n< | 块文字左移n个tab |
| n> | 块文字右移n个tab |
| = | 块文字格式化 |
| :ce(nter) | 居中 |
| :le(ft) | 左对齐 |
| :ri(ght) | 右对齐 |
| Q | 等于gq |
| gq[n]q | 重新排序n行 |
| gq[n]ap | 重新排序n段 |
| gq[n]j | 重新排序当前行和下面n行 |
| gqQ | 重新排序当前行到文章尾 |
| :set lines=[n] | 设置vim的总行数 |
| :set columns=[n] | 设置vim的总列数 |
| :winpos [n] [n] | 设置gvim 的左上角坐标位置，可以在桌面上移动gvim |
| [Ctrl]+ l/redraw | 重回屏幕 |
| 重新排版 | |
| zz | 重绘，当前行在中间 |
| zt | 重绘，当前行在顶部 |
| zb | 重绘，当前行在下面 |
| zl | Scroll left |
| zh | Scroll right |
| 刪除、複製與貼上 | |
| x, X | 在一行字當中，x 為向後刪除一個字元 (相當於 [del] 按鍵)， X 為向前刪除一個字元(相當於 [backspace] 亦即是倒退鍵) (常用) |
| [n]x | n 為數字，連續向後刪除 n 個字元。舉例來說，我要連續刪除 10 個字元， 『10x』。 |
| s | 删除一个字符，进入插入模式 (如： cl一样) |
| S | 删除光标所在行的所有文字，并且进入插入模式 |
| dd | 刪除游標所在的那一整列(常用) |
| [n]dd | n 為數字。刪除游標所在的向下 n 列，例如 20dd 則是刪除 20 列 (常用) |
| :n1,n2 d | 删除n1行到n2行的位置 |
| Dgg | 删除到文件头部 |
| dG | 刪除游標所在到最後一行的所有資料 |
| d[n]G | 刪除游標所在到第一行的所有資料 |
| d$ | 刪除游標所在處，到該行的最後一個字元 |
| d0 | 那個是數字的 0 ，刪除游標所在處，到該行的最前面一個字元 |
| dge, db, de, dw | 词删除 |
| d[n]l | 向右删除(n)一个字符 (同x) |
| d[n]h | 向左删除(n)一个字符 (同X) |
| d[n]j | 向下删除(n)一行， (同ndd) |
| d[n]k | 向上删除(n)一行 |
| D | 删除光标位置到行尾所有字符 |
| daw (aw) | 删除一个词 |
| das （as） | 删除一句句子 |
| dap （ap） | 删除一段(空格隔开) |
| dab (ab) | 删除一块(小圆括号隔开) |
| d2/foo | 删除当前位置到第二个foo之前的数据 |
| C | 重複刪除多個資料，例如向下刪除 10 行，[ 10cj ] |
| C | 删除光标位置到行尾所有字符，进入插入模式 |
| c2/foo | 删除当前位置到第二个foo之前的数据，进入插入模式 |
| Yy | 複製游標所在的那一行(常用) |
| [n]yy | n 為數字。複製游標所在的向下 n 列，例如 20yy 則是複製 20 列(常用) |
| :n1,n2 y | 复制n1行到n2行的位置 |
| Ygg | 复制到文件头部 |
| yG | 複製游標所在列到第一列的所有資料 |
| y[n]G | 複製游標所在列到最後一列的所有資料 |
| y0 | 複製游標所在的那個字元到該行行首的所有資料 |
| y$ | 複製游標所在的那個字元到該行行尾的所有資料 |
| yge, yb, ye, yw | 词复制 |
| y[n]l | 向右复制(n)一个字符 (同x) |
| y[n]h | 向左复制(n)一个字符 (同X) |
| y[n]j | 向下复制(n)一行， (同ndd) |
| y[n]k | 向上复制(n)一行 |
| Y | 复制光标位置到行尾所有字符 |
| y2/foo | 复制到第二个foo之前 |
| yaw (aw) | 复制一个词 |
| yas （as） | 复制一句句子 |
| yap （ap） | 复制一段(空格隔开) |
| yab (ab) | 复制一块(小圆括号隔开) |
| Aw | 一个词 |
| As | 一句句子 |
| Ap | 一段(空格隔开) |
| Ab | 一块(小圆括号隔开) |
| 黏贴 | |
| p, P | p 為將已複製的資料在游標下一行貼上，P 則為貼在游標上一行！ 舉例來說，我目前游標在第 20 行，且已經複製了 10 行資料。則按下 p 後， 那 10 行資料會貼在原本的 20 行之後，亦即由 21 行開始貼。但如果是按下 P 呢？ 那麼原本的第 20 行會被推到變成 30 行。 (常用) |
| ]p | 格式化黏贴，带对齐功能 |
| xp | 交换字母位置 |
| ddp | 交换行的位置 |
| 块移动 | |
| ：n1,n2 co n3 | n1到n2 copy到n3位置后面 |
| ：n1,n2 mo n3 | n1到n2 move到n3位置后面 |
| 合并行 | |
| J | 將游標所在列與下一列的資料結合成同一列 |
| gJ | 将反白的地方全部连成一行,保留空白符号 |
| [n]K | 查字典，n 例如3K ,man 3 XXX |
| 撤销， 反撤销 | |
| [n]u/ g- | 復原前一個動作。(常用) |
| U | 当前行取消所有改动 |
| [Ctrl]+r/redo/ g+ | 重做上一個動作。(常用) |
| :[n]undo | 退回到某个分支，（分支就是修改后恢复，然后再做其他修改，前面恢复后的版本） |
| :undolist | 显示分支列表，有上面这个命令的number [n] undo |
| :earlier [n]s/[n]m/[n]h | 往前多少时间的版本 |
| :later [n]s/[n]m/[n]h | 往后多少时间的版本 |
| 补全操作 | |
| [Ctrl]+p | 文本框，插入模式，变量向前代码补全 |
| [Ctrl]+n | 文本框，插入模式，变量向后代码补全 |
| [Ctrl]+x和[Ctrl]+P | 函数补齐 |
| [Ctrl]+x和[Ctrl]+N | 函数补齐 |
| [Ctrl]+x和[Ctrl]+o | 函数名和关键字，自动补齐 |
| [Ctrl]+x和[Ctrl]+D | 宏定义补全 （注意大写） |
| [Ctrl]+x和[Ctrl]+] | Tag补全 |
| [Ctrl]+x和[Ctrl]+f | 文件名补全 |
| [Ctrl]+x和[Ctrl]+i | 关键词补全, 后面跟关键词的文件位置 |
| [Ctrl]+x和[Ctrl]+v | 命令行补全 |
| [Ctrl]+x和[Ctrl]+l | 整行补全 |
| [Ctrl]+x和[Ctrl]+s | 拼写建议 |
| [Ctrl]+x和[Ctrl]+k | dictionary |
| [Ctrl]+x和[Ctrl]+t | thesaurus |
| 补全列表操作 | |
| [Ctrl]+n | 前一个选项 |
| [Ctrl]+p | 后一个选项 |
| [Ctrl]+e | 退出下拉菜单，退回原来的文字 |
| [Ctrl]+y | 退出下拉菜单，并接受当前选项 |
| 数字加减 | |
| [Ctrl]+a | 光标所在位置的数字+1 |
| [Ctrl]+x | 光标所在位置的数字-1 |
| 重复命令 | |
| [n]. | 不要懷疑！這就是小數點！意思是重複前一個動作的意思。 如果你想要重複刪除、重複貼上等等動作，按下小數點『.』就好了！加上n表示重复n次 (常用) |
| & | 重复: s/aa/bb/gc 这样的替换动作 |
| @: | 重复上一个冒号命令 |

* **第二部份：一般模式切換到命令模式的可用的按鈕說明**

|  |  |
| --- | --- |
| 模式切换 | |
| Esc | 返回命令模式 |
| [Ctrl]+ [ | 返回命令模式 |
| [Ctrl]+ c | 返回命令模式, Do not check for [abbreviations](http://vimdoc.sourceforge.net/htmldoc/map.html#abbreviations). Does not trigger the |[InsertLeave](http://vimdoc.sourceforge.net/htmldoc/autocmd.html" \l "InsertLeave)| [autocommand](http://vimdoc.sourceforge.net/htmldoc/autocmd.html#autocommand) event. |
| :set insertmode | go into the insert mode |
| [Ctrl] + l | go back to normal mode, when the insertmode is set |
| : | 命令 |
| / , ? | 查找 |
| [n]! , ![n] | 当前行加上以下行的过滤操作  eg: 33! 然后自动进入命令行，输入<.,.+33> !sort –n |
| :gui | 进入gui模式，即. Gvim |

* **\第三部份：一般模式切換到編輯模式的可用的按鈕說明**

|  |  |
| --- | --- |
| 進入插入或取代的編輯模式 | |
| i, I | 進入插入模式(Insert mode)： I 為『從目前游標所在處插入』， I 為『在目前所在行的第一個非空白字元處開始插入』。 (常用) |
| gI | 在当前视频行的第一列插入 |
| [n]idus | 插入n个dus, eg:100idus， 100i=（100个=） |
| a, A | 進入插入模式(Insert mode)： a 為『從目前游標所在的下一個字元處開始插入』， A 為『從游標所在行的最後一個字元處開始插入』。(常用) |
| o, O | 進入插入模式(Insert mode)： 這是英文字母 o 的大小寫。O 為『在目前游標所在的下一行處插入新的一行』； O 為在目前游標所在處的上一行插入新的一行！(常用) |
| r, R | 進入取代模式(Replace mode)： r 只會取代游標所在的那一個字元一次；R會一直取代游標所在的文字，直到按下 ESC 為止；(常用) |
| gR | 不影响格式的替换『--VREPLACE--』 |
| 上面這些按鍵中，在 vi 畫面的左下角處會出現『--INSERT--』或『--REPLACE--』的字樣。 由名稱就知道該動作了吧！！特別注意的是，我們上面也提過了，你想要在檔案裡面輸入字元時， 一定要在左下角處看到 INSERT 或 REPLACE 才能輸入喔！ | |
| 切换到正常模式 | |
| [Esc] | 退出編輯模式，回到一般模式中(常用) |
| [Ctrl]+ c | 退出編輯模式，回到一般模式中(常用) |
| [Ctrl]+ [ | 退出編輯模式，回到一般模式中(常用) |
| 大小写转换 | |
| :gU, gu ,g~ | 设置大小写 |
| :gUU ,guu, g~~ | 全行大写，小写，大小写互换 |
| gUw, guw, g~w | 单词大写，小写，大小写互换 |
| gUW, guW, g~W | 长单词的大写，小写，大小写互换 |
| gUiw, guiw, g~iw | 当前单词大写，小写，大小写互换 |
| V模式下(U,u,~) | 虚拟模式下的大小写转换 |

* **第三部份：一般模式切換到指令列模式的可用的按鈕說明**

|  |  |
| --- | --- |
| 指令列的儲存、離開等指令 | |
| :tohtml | 生成html格式文件， source $VIMRUNTIME/syntax/2html.vim |
| :w | 將編輯的資料寫入硬碟檔案中(常用) |
| :w! | 若檔案屬性為『唯讀』時，強制寫入該檔案。不過，到底能不能寫入， 還是跟你對該檔案的檔案權限有關啊！ |
| :q | 離開 vi (常用) |
| :q! | 若曾修改過檔案，又不想儲存，使用 ! 為強制離開不儲存檔案。 |
| 注意一下啊，那個驚嘆號 (!) 在 vi 當中，常常具有『強制』的意思～ | |
| :wq | 儲存後離開，若為 :wq! 則為強制儲存後離開 (常用) |
| ZZ | 這是大寫的 Z 喔！若檔案沒有更動，則不儲存離開，若檔案已經被更動過，則儲存後離開！(同x) |
| ZQ | 不保存直接退出 |
| [Ctrl]+ z/:suspend | 挂起vim |
| :f [filename] | 重新命名窗口，w可以完成复制功能 |
| :w [filename] | 將編輯的資料儲存成另一個檔案（類似另存新檔） |
| :n1,n2 w [filename] | 將 n1 到 n2 的內容儲存成 filename 這個檔案。 |
| :w >> filename | 追加到文件后面 eg. 10,20 w >> filename |
| : n1,n2 w >> filename | 將 n1 到 n2 的內容,追加到文件后面 eg. 10,20 w >> filename |
| :saveas path | 另存为 |
| :r [filename] | 在編輯的資料中，讀入另一個檔案的資料。亦即將 『filename』 這個檔案內容加到游標所在行後面 |
| :e [filename] | 切换编辑filename,如果本文被改动了会发出警告 |
| :e! [filename] | 不保存，切换编辑filename |
| :e+ [filename] | 新开一个文件，从尾开始编辑 |
| :e+n [filename] | 新开一个文件，从第n行开始编辑 |
| :! Cmd | 暫時離開 vi 到指令列模式下執行 command 的顯示結果！例如 『:! Ls /home』即可在 vi 當中察看 /home 底下以 ls 輸出的檔案資訊！ |
| :sh(shell)  :exit | 打开一个shell，进入shell编辑，exit退出。 |
| :!! | 重复上传命令!!date –u |
| :nr !cmd | 读取命令输出到第n行位置 eg: 3r|date –u |
| :n1,n2 w !cmd | 读取n1行到n2行之间的文本作为命令的输入 |
| **直接命令** | |
| :cd, pwd,  make, grep, abbr | 直接运行的命令，无需！ |
| **Grep** | |
| :vim[grep]/pattern/[g] [j] files | grep查找 eg：  ：vim /^d{1,}./%  [g]表示一行中出现多次，这一行也在结果中出现多次  [j]表示跳转到第j项 |
| **Make** | |
| :set makeprg=javac\ abc.java | 如果目录下面没有makefile，需要设置好makeprg才能：make编译 |
| :setl efm=%A%f:%l:\ %m,%-Z%p^,%-C%.%# | %f:文件  %l:行号  %m:错误信息 |
| **Source** | |
| :source % | 执行命令，在当前文件中 |
| :exec ‘…’ | 执行命令 eg: exec ‘!ls -l’ |
| **vim加密** | |
| vim –x file | 加密，注意需要w保存 |
| ：X | 加密，注意需要w保存 |
| ：set key= | 去除文件的密码 |
| **二进制打开** | |
| vim –b file | 二进制打开文件 |
| :%!xxd | 查看unicode字母编码 eg: 1,5 !xxd |
| :%!xxd –r | 查看unicode字母编码 |
| :set display=uhex | 显示方式 |
| **打印** | |
| :hardcopy | 打印，可以选选择打印的块 eg: %hardcopy/1,15 hardcopy |
| **排序** | |
| ：%!sort –n | 全文排序（按照字母） |
| **设置缩写** | |
| :abbr ms microsoft | 缩写功能，在insert模式下， |
| ms<space> | 自动扩展，后面多个<space> |
| ms, | 自动扩展，后面多个<，> |
| ms然后输入[Ctrl]+ ] | 自动扩展，后面不会多字符 |
| **历史记录** | |
| :history | 查看vim历史命令 |
| q: | 查看 命令历史 |
| q/,q? | 查看 查找历史 |
| [Ctrl]+ p | 向上查找 |
| [Ctrl]+ n | 向下查找 |
| **目录窗口控制** | |
| :browse [e]/[w] | e:edit 打开浏览窗口  w:saveas 另存为打开浏览窗口  会打开一个文件浏览器让你选择要编辑的文件。如果是终端中，则会打开netrw的文件浏览窗口； 如果是gvim,则会打开一个图形界面的浏览窗口。 实际上：browse后可以更任何编辑文档的命令，如sp等。用browse打开的起始目录可以由browsedir来设置：  :set browsedir=last --上次访问过的目录  :set browsedir=buffer --当前文件所在的目录  :set browsedir=current --当前工作目录  eg: set browsedir=/home/xumin/cmc/mk3Fake |
| Se(x), Se(x)! | 横上边打开目录， 竖着左边打开目录 |
| E(x), e . | 全屏打开目录 |
| Ve(x), Ve(x)! | 竖着左，右边打开目录 |
| He(x), He(x)! | 横着下，上边打开目录 |
| :Te | 打开有一个tab页面 |
| **Browse 目录操作** | |
| - | 进入上级目录 |
| C | 设为当前目录 |
| D | Delete 删除文件 |
| R | rename重命名文件 |
| S | Sort 排序 |
| X | Exec运行，xwindow下可以打开目录 |
| **操作Buffer** | |
| :bn, bp, bl, bf | 下一个，上一个，最后一个，第一个 |
| :[n]bd,[n]bw | 关闭当前(第n个)buffer,真正关闭这个文件 |
| :b[n]/ ：[n]b | 打开第n个文件，n是数字 |
| :ba[ll] | 打开所有缓冲文件，分割 |
| :e#/ [Ctrl]+ ^ | 前一个，后一个之间切换 |
| :buffers, ls, files | 查看缓冲区文件列表 |
| :bufdo cmd | 所有buff执行cmd. Eg:bufdo tab split 所有buff展开成tab |
| buffer n/filename | 打开文件 n编号， 也可以文件名打开 |
| set scb/set ncscb | 设置/取消鼠标滚轴同步滚动屏幕 |
| **Tab** | |
| vim –p file1 file2 … | tab方式打开多个页面 vim –p \*.c |
| [Ctrl]+ w + T | 分页变成一个tab页面 |
| :Te | 打开有一个tab页面 (同：tabe filename) |
| [n]gt | 下一个(n个)tab页面 |
| [n]gT | 上一个(n个)tab页面 |
| :tabr/tabfirst/tabrewind | 第一个tab页面 |
| :tabl[ast] | 最后一个tab页面 |
| :tabn[ext] [n] | 下一个(n个)tab页面 |
| :tabN[ext] [n] | 上一个(n个)tab页面 |
| :tabp[revious] | 上一个tab页面 |
| :tabm[ove] [n] | 切换tab页面的位置，n>=0. Move |
| :[n]tabn | 指定打开第一个tab |
| :tabs | 显示所有的tab页面 |
| :tabe[dit] filename | 新建打开一个tab页面，可以打开某个文件 filename |
| :tabnew filename | 新建tab |
| :tabfind filename | 查找并打开filename tab |
| :tab split filename | 打开filename tab, 不分割 |
| :tabc[lose] [n] | 关闭tab页面，如果不加n就是关闭当前页面 |
| :tabo[nly] | 关闭其他的tab |
| :tabdo | 在所有的tab中执行某个命令 eg： tabdo %s/foo/bar/g |
| [[Ctrl] –](http://vimdoc.sourceforge.net/htmldoc/index.html#CTRL-W)w [gf](http://vimdoc.sourceforge.net/htmldoc/editing.html#gf) | Open a new [tab](http://vimdoc.sourceforge.net/htmldoc/intro.html#tab) page and edit the file name under the cursor. |
| [[Ctrl] –](http://vimdoc.sourceforge.net/htmldoc/index.html#CTRL-W)w [gF](http://vimdoc.sourceforge.net/htmldoc/editing.html#gf) | Open a new [tab](http://vimdoc.sourceforge.net/htmldoc/intro.html#tab) page and edit the file name under the cursor  and jump to the line number following the file name. #include “aa.h 100” |
| **vim 環境的變更** | |
| :set nu | 顯示行號，設定之後，會在每一行的字首顯示該行的行號 |
| :set nonu | 與 set nu 相反，為取消行號！ |
| :set ic | 忽略大小写 |
| :set noic | 取消忽略大小写 |
| :set scb/set ncscb | 设置/取消鼠标滚轴同步滚动屏幕 |
| :set backup | 打开自动备份功能 |
| :set backupext=.bak | 设置备份文件的后缀名设置 |
| :set patchmode=.org | 设置备份原始未更改文件的后缀名 |
| :set fileformat=unix,dos  :set ff=unix,dos | 设置文件为unix或dos格式 |
| :set fileformat?  :set ff=? | 查看文件格式 |
| :set so=n | 鼠标下行的时候，底部和上部剩余可见的行数 eg: set so=8 |
| :set autowrite | 设置 自动保存 |
| :set noautowrite | 设置取消自动保存 |
| :set autochdir | 根据编辑的文件自动切换目录 |
| **\區塊選擇的按鍵意義** | |
| v | 字元選擇，會將游標經過的地方反白選擇！ |
| V | 行選擇，會將游標經過的行反白選擇！ |
| [Ctrl]+ v | 區塊選擇，可以用長方形的方式選擇資料 |
| [Ctrl]+ q | 區塊選擇，可以用長方形的方式選擇資料 gui模式 |
| h, j, k, l, $, g\_, … | Motion控制块的选择(方向键) |
| [Ctrl]+v; 99j，$ | 块选中当前行开始的100行 |
| I | 插入文字，需要C-v选择块，然后I插入文字，ESC |
| A | 追加文字，需要C-v选择块，然后$移动到行尾，再A追加文字，ESC |
| J | 将反白的地方全部连成一行 |
| y | 將反白的地方複製起來 |
| d | 將反白的地方刪除掉 |
| gv | 选中上次选中的块 |
| **多檔案編輯的按鍵** | |
| vim file1 file2 file3 … | 打开多个文件，只显示一个在前台 |
| :n | 編輯下一個檔案 |
| :n filename | 编辑filename 这个文件 |
| :N | 編輯上一個檔案 |
| [n]+ [Ctrl]+ ^ | 打开第n个buffer，不加n为打开上次编辑的文件 |
| :e#+ [n] | 打开第n个buffer，不加n为打开上次编辑的文件 |
| :files/ls | 列出目前這個 vim 的開啟的所有檔案 |
| :args | 显示所有 (同ls,files) |
| :[n]new | 横着新建 分割,控制n行 |
| [Ctrl]+ w +n | 全屏新建 分割 ，快捷键 |
| : [n]vnew | 竖着新建 分割，控制n列 |
| :enew | 全屏新建 分割 |
| :[n] split | 横着分割控制行数 |
| :[n] vertical split | 竖着分割控制列数 |
| :all | 全部横着显示，分割 |
| :vertical all | 全部竖着显示，分割 |
| :qall | 退出所有 |
| :qall! | 强制退出所有 |
| :wall | 保存所有 |
| :wall! | 强制保存所有 |
| :wqall | 保存退出所有 |
| :next | 不保存后一个 |
| :prev | 不保存前一个 |
| :next! | 强制不保存切换到后一个 |
| :prev! | 强制不保存切换到前一个 |
| :wnext | 保存切换到后一个 |
| :wprev | 保存切换到前一个 |
| :first | 第一个 |
| :last | 最后一个 |
| :only | 关闭其他 |
| :close | 关闭 |
| :sf[ind] [filename] | 全屏新建，打开 filename |
| :sv[iew] [filename] | 只读属性打开 filename |
| :hide e filename | 全屏打开编辑一个文件,隐藏模式 |
| **比较区别 diff** | |
| vim –d file1 file2 … | 比较文件的区别 |
| vimdiff file1 file2 … | 文件差异比较 |
| gvimdiff file1 file2 … | gui模式比较文件差异 |
| :set guifont=ProggyTinyTT\ 12 | 设置gui模式的字体 和大小 |
| ：[n] split | 控制打开文件的行数 分割 |
| ：[n] vsplit | 控制打开文件的列数 分割 |
| ：diffthis | 本文件加入比较 |
| ：[vertical] diffsplit | 分割打开文件进行比较，后面加文件名 |
| ：[vertical] diffpatch | 打补丁 |
| ：[vertical] diffupdate | 更近差异比较，修改后手动更新 |
| ：[range]diffget [bufspec]  /do | 合并差异 range 哪几行， bufspec 那个缓存。 Do get 做一次 |
| ：[range]diffput [bufspec]  /dp | 合并差异 dp put一次 |
| [c | 下一个差异 |
| ]c | 上一个差异 |
| :hi DiffAdd guifg=#rrggbb | ctermfg=# |
| :hi DiffChange guibg=#rrggbb | ctermbg=# |
| :hi DiffText gui=non |  |
| :hi DiffDelete | Vim着色板 |
| **语法设置** | |
| :syntax enable | 使能语法检查 |
| :set filetype= | Choose only one type: c cpp make per python |
| **Quickfix** | |
| :cw | 打开(make,grep) |
| :copen | 打开(make,grep) |
| :cclose | 关闭 |
| :cp | 跳转到上一项错误 |
| :cn | 跳转到下一项错误 |
| :cr | 跳转到第一项错误 |
| :cl | 显示错误列表 |
| :cc [n] | 跳转到第n个错误 |
| :col | 下一页 |
| :cnew | 上一页 |
| **拼写错误** | |
| :set spell | 打开拼写检查 |
| :set nospell | 关闭拼写检查 |
| ]s | 下一个拼写错误 |
| [s | 上一个拼写错误 |
| z= | 显示一个拼写错误单词的列表 |
| Zg | 告诉拼写检查器这个词是正确的 |
| Zw | 告诉拼写检查器这个词是错误的 |
| **打开多行** | |
| vim –On file1 file2 … | 纵向分割打开多个窗口，n是打开窗口的数量 |
| vim –on file1 file2 … | 横向分割打开多个窗口，n是打开窗口的数量 |
| :sp/vsp [filename] | 開啟一個新視窗，如果有加 filename， 表示在新視窗開啟一個新檔案，否則表示兩個視窗為同一個檔案內容(同步顯示)。 |
| **多視窗情況下的按鍵功能** | |
| [ctrl]+w+ ， :wincmd |  |
| [ctrl]+w+ h |  |
| [ctrl]+w+ l |  |
| [ctrl]+w+ j | 按鍵的按法是：先按下 [ctrl] 不放， 再按下 w 後放開所有的按鍵，然後再按下 j (或向下方向鍵)，則游標可移動到下方的視窗。 |
| [ctrl]+w+ k | 同上，不過游標移動到上面的視窗。 |
| [ctrl]+w+ t | Top 最上面的一个窗口 |
| [ctrl]+w+ b | Bottom 最下面的一个窗口 |
| [ctrl]+w+ p | 最近连个窗口间切换 |
| [ctrl]+w+ w | 顺序切换下一个窗口 |
| [ctrl]+w+ q | 其實就是 :q 結束離開啦！ 舉例來說，如果我想要結束下方的視窗，那麼利用 [ctrl]+w+↓ 移動到下方視窗後，按下 :q 即可離開， 也可以按下 [ctrl]+w+q 啊！ |
| [ctrl]+w+ c | Close， 关闭窗口，除了最后一个不能关闭 |
| [ctrl]+w+ o | Only， 关闭其他窗口，就剩当前窗口 |
| [ctrl]+w+ f/  [ctrl]+w+ gF/  [ctrl]+w+ gf/  gf/  g[/  g] | 水平分割，查找，并打开光标下面文件 如：#include<stdio.h>, 光标在stdio.h 打开此文件 gf |
| [ctrl]+w+ n | New，新建一个窗口 （同 :[n]new） |
| [ctrl]+w+ s | Split |
| [ctrl]+w+ v | Vertical split |
| [ctrl]+w+ ^ | 水平分割，上次打开的buffer |
| [ctrl]+w+ [ctrl]+^ | 水平分割，上次打开的buffer |
| **窗口本身的移动** | |
| [ctrl]+w+ H | 窗口本身向左移动 |
| [ctrl]+w+ J | 窗口本身向下移动 |
| [ctrl]+w+ K | 窗口本身向上移动 |
| [ctrl]+w+ L | 窗口本身向右移动 |
| [ctrl]+w+ T | 当前窗口最大化(bp,bn) |
| [ctrl]+w+ r | 旋转窗口位置 |
| [ctrl]+w+ x | 旋转窗口位置 |
| **窗口大小变化** | |
| [n] [ctrl]+w+ - | 窗口size纵向变小 |
| [n] [ctrl]+w+ + | 窗口size纵向变大 |
| [n][ctrl]+w+ \_ | 窗口横向最大化，其他窗口挤在下面 |
| [n] [ctrl]+w+ < | 窗口size横向变小 |
| [n] [ctrl]+w+ > | 窗口size横向变小 |
| [n][ctrl]+w+ | | 窗口纵向最大化，其他窗口挤在右边 |
| [ctrl]+w+ = | 窗口屏幕分配大小， 全部窗口恢复 |
| :resize +/-/=num | 纵向调整窗口大小 |
| :vertical resize +/-/= | 横向调整窗口大小 |
| **命令行模式下的快捷键** | |
| [Ctrl] + u | 当前位置移动到命令开头，删除字符 |
| [Ctrl] + b | 当前位置移动到命令开头 |
| [Ctrl] + e | 当前位置移动到命令末尾 |
| **插入模式下的快捷键** | |
| [Ctrl] + p | 向上补全 |
| [Ctrl] + n | 向下补全 |
| [Ctrl] + d | 同<<, undent |
| 0 [Ctrl] + d | 删除所有的tab |
| ^ [Ctrl] + d | 删除所有的tab, The indent is restored in the next line. |
| [Ctrl] + t | 同>>, indent |
| [Ctrl] + i/tab | 插入一个tab, 中间 |
| [Ctrl] + a | 插入上次插入的字符串 |
| [Ctrl] + @ | 插入上次插入的字符串，进入命令模式 |
| [Ctrl] + j / [Ctrl] + m | 插入行, 同O |
| [Ctrl] + u | 向前删除一行 |
| [Ctrl] + w | 向前删除一个单词 |
| [Ctrl] + h | 向前删除一个字符 |
| [Ctrl] +y | 补全上面行的字符 |
| [Ctrl] + e | 补全下面行的字符 |
| [Ctrl]+x和[Ctrl]+y | 向上滚动一行 |
| [Ctrl]+x和[Ctrl]+e | 向下滚动一行 |
| [Ctrl]+g和[Ctrl]+j  [Ctrl]+g +j | 向上移动一行 |
| [Ctrl]+g和[Ctrl]+k  [Ctrl]+g +k | 向下移动一行 |
| [Ctrl]+x和[Ctrl]+f | 文件名补全 |
| [Ctrl]+x和[Ctrl]+s | 拼写建议 |
| [Ctrl]+x和[Ctrl]+k | Dictionary |
| [Ctrl]+x和[Ctrl]+t | thesaurus |
| [Ctrl]+v n/ [Ctrl]+q n | 输入特殊编码，在插入模式eg: C-v-273 |
| [Ctrl]+k {char} {char} | 输入二合字符 |
| [Ctrl]+r +[register name] | Paste register eg: [Ctrl]+r % **results in “ac”.** |
| [Ctrl]+r [Ctrl]+r +[register name] | 本义黏贴，保留特殊字符 **results in “ab^Hc”.** |
| [Ctrl]+r [Ctrl]+p +[register name] | 本义黏贴，不自动缩进 |
| [Ctrl]+r [Ctrl]+o +[register name] | 本义黏贴，自动缩进 |
| :help c\_CTRL-R | Ctrl –r 帮助查询 |
| [a-z] | 寄存器 |
| [0-9] |  |
| : | 最近一次冒号命令 |
| / | 最近一次查找pattern |
| + | Vim系统寄存器 |
| \* | X window 系统寄存器 |
| “ | 最近一次 yy, dd 的备份黏贴， last del/copy |
| % | 当前文件名 |
| # | 切换过来的文件名 |
| - | 小于一行的最短删除 |
| . | 上次insert的内容 |
| = | 扩展宏，用于数字计算，浮点数自动转换为字符串 |
| [Ctrl] +\ 和 [[Ctrl] +](http://vimdoc.sourceforge.net/htmldoc/motion.html#CTRL-O)o | like [CTRL-O](http://vimdoc.sourceforge.net/htmldoc/motion.html#CTRL-O) but don't move the cursor |
| [Ctrl] + o | 进入命令模式，执行一条命令后，返回插入模式 |
| [Ctrl] + o :reg | 查看寄存器内容 |
| [Ctrl] + Left/[Shift] + Left | 左移一个单词 |
| [Ctrl] + Right /[Shift] + Right | 右移一个单词 |
| [Ctrl]+ ] | Trigger abbreviation, without [inserting](http://vimdoc.sourceforge.net/htmldoc/insert.html#inserting) a character. [abbr] |
| [Ctrl] +g +u | 打断undo序列，重新开始新的change ，作用在同一串字符串 |
| [Ctrl] +o +u | 撤销 |
| <insert> | Toggle between [Insert](http://vimdoc.sourceforge.net/htmldoc/insert.html#Insert) and [Replace](http://vimdoc.sourceforge.net/htmldoc/insert.html#Replace) mode |

* **第三部份：vim 打开命令**

|  |  |
| --- | --- |
| **vim 打开命令** | |
| vim –p file1 file2 … | tab方式打开多个页面 vim –p \*.c |
| vim –d file1 file2 … | 差异比较 |
| vim –x file | 加密vim |
| vim –d file | 二进制打开 |
| vim –On file1 file2 … | 纵向分割打开多个窗口，n是打开窗口的数量 |
| vim –on file1 file2 … | 横向分割打开多个窗口，n是打开窗口的数量 |
| vim –c cmd file | 打开文件前先执行指定的命令 |
| vim –r file | 恢复上次异常退出的文件 |
| vim –R file | 以只读方式打开文件，但可以强制保存 |
| vim –M file | 以只读方式打开文件，不可以强制保存 |
| vim –y num file | 将编辑窗口的大小设为num行 |
| vim + file | 从文件尾开始编辑 |
| vim +num file | 从文件num行开始编辑 |
| vim +/string file | 更近差异比较，修改后手动更新 |
| vim –remote file | 用已有的vim进程打开指定的文件。 |
| vim filename +[n] | 打开文件，从n行开始编辑 |
| vim filename … –t Foo | 打开文件，在tag Foo处开始编辑 |
| vim filename … –c “/Foo” | 打开文件，找到字符串‘Foo’开始编辑 |
| vim –g filename, gvim filename | gui模式编辑 |

* **第三部份：gloab命令**

|  |  |
| --- | --- |
| **gloab 命令** | |
| Ga | 查看unicode字母编码 |
| g8 | 查看utf8编码 |
| Gd | 跳到变量定义处，局部 |
| gD | 跳到变量定义处，全局 |
| Gv | 选中上次选中的块 |
| :changes | 显示changelist |
| g; | 上个修改过的地方,changelist |
| g, | 下个修改过的地方,changelist |
| gf/]f/[f | 打开光标处所在的文件，特别是#include，需要先设定set path=../,cmc |
| [ctrl]+ w +f | 打开光标处所在的文件，特别是#include，需要先设定set path=../,cmc |
| Gh | 块选择，不是虚拟模式，不能用=对齐 |
| [ctrl]+ g | 在底部显示文件信息 |
| :f | 在底部显示文件信息 |
| :f [filename] | 更改缓存的名字 |
| g+[ctrl]+ g | 字数，行数统计 |
| gR | 不影响格式的替换, r/R |
| gI | 在当前视频行的第一列插入 |
| gJ | 将反白的地方全部连成一行,保留空白符号 |
| [Ctrl]+v n | 输入特殊编码，在插入模式eg: C-v-273 |

* **第三部份：折叠**

|  |  |  |
| --- | --- | --- |
| **折叠** | | |
| zc/zC | 折叠当前行 | Fold close/ |
| zo/zO | 打开当前折叠 | Fold open/ |
| Za | 打开/关闭当前折叠 |  |
| zm/zM | 关闭折叠 | Fold more/ |
| zr/zR | 打开嵌套的所有 | Fold reduce/fold remove |
| zd/zD | 删除所有折叠 |  |
| [z/]z | 到当前打开的折叠的开始/结尾处 |  |
| zi | 打开/关闭折叠功能 | Invert all |
| zv | 查看此行 |  |
| zf | 创建折叠 10zf , zfG |  |
| zfap | 折叠光标所在段 |  |
| zj/zk | 向下移动。到达下一个折叠的开始/结尾处。关闭的折叠也被计入。 |  |

* **第三部份：标记 ，宏，寄存器**

|  |  |
| --- | --- |
| **mark 标记** | |
| m[a-z]/[A-Z] | 作跳转标记，小写是局部标记，大写是全局标记(文件之间跳转) |
| g`mark/g’mark | 跳转，例如ma, g`a/g’a |
| ‘[a-z]/[A-Z] | 跳转到标记行首字母 |
| `[a-z]/[A-Z] | 跳转到标记行的列 |
| [Ctrl]+o | 跳转到上次跳转点，层次嵌套，多个文件跳转 |
| `’/ ’’ | 跳转到上次跳转的位置 |
| `. / ’. | 跳转到上次编辑的位置 |
| `”/ ’” | 跳转到上次文件关闭时的光标位置 |
| `0 / ’0 | 跳转到上次文件关闭时的光标位置 |
| `1 / ’1 | 跳转到上上次文件关闭时的光标位置 |
| `2 / ’2 … | 跳转到上上上次文件关闭时的光标位置 |
| :marks | 显示跳转标记列表 |
| :delmarks [a-z]/[A-Z] | 删除跳转标记, |
| :delmarks a-c | 删除a-c的标记 |
| :delmarks! | 加！删除所有标记 |
| **Vim 宏操作** | |
| q[a-z]/[A-Z] | 记录操作宏，小写开始记录一个宏，大写往小写宏追加操作 |
| Q | 停止记录 |
| [n] @[a-z] | 执行一次宏操作，n为重复n次 |
| [n] @@ | 重复一次执行 ，n为重复n次 |
| **寄存器操作** | |
| “[a-z]/[A-Z] | 执行n次宏操作 eg: “kyy/V“ky,kp, 大写[A-Z]为追加内容到[a-z]小写寄存器 |
| “+ | Xwindow 系统寄存器 |
| “\* | Vim 系统寄存器 |
| “= | “=, [Ctrl]+R = |
| “ | 无名寄存器 |
| “” | 不加寄存器索引时默认使用 |
| :reg[isters] | 查看寄存器 |
| [Crtl]+ [Shift]+ r | 查看寄存器 |

* **第三部份：make session 恢复打开之前的窗口**

|  |  |
| --- | --- |
| **Session保存** | |
| :mksession[!] filename | 保存一个session,! 为强制覆盖保存 |
| vim –S filename | 打开上次保存的session，恢复当时打开的窗口 |

* **第三部份：mkview保存查看的特征**

|  |  |
| --- | --- |
| **保存读取特征** | |
| ：mkview [n] | 设置保存标记（如折叠），一个文件最多十个 |
| ：loadview [n] | 加载的标记方式 |

* **第三部份：修改文件编码**

|  |  |
| --- | --- |
| **编码** | |
| ：e ++enc=utf8 filename | 以某个格式打开filname |
| :w ++enc=gbk | 以某个格式保存这个文件 |
| :set fenc/ :set fileencoding | 查看当前编码 |
| :set fileencoding=ucs-bom,utf8,cp936 | 根据文件格式自动选择编码方式，编码之间不能有空格 |

* **第三部份：修改文件格式**

|  |  |
| --- | --- |
| **文件格式** | |
| ：e ++ff=dos filename | 以某个格式打开文件 |
| :w ++eff=mac fileane | 以某个格式保存文件 |
| :set ff | 查看当前格式 |
| :set fileformats=unix,dos,mac | 让vim自动识别格式 |

* **第三部份：CTAG**

|  |  |
| --- | --- |
| **CTAG** | |
| ctags –R path | 生成tag文件， -R 表示目录递归 |
| :set tags=path/tags | 告诉ctags使用哪个tag文件 |
| :ta[g] xyz/ [Ctrl] + ] | 跳到函数定义处 |
| :ts | List active tags |
| [Ctrl] + t | 返回 |
| [Ctrl]+ p | Complete word |
| :stag xyz/ [Ctrl]+w + [n] ] | 分割 跳到函数定义处 |
| :ptag xyz/ [Ctrl]+w + [n] } | 预览窗口，跳到函数定义处 |
| :pclose/ [Ctrl] + w +z | 关闭预览窗口 |
| :pedit filename | 在预览窗口编辑filename |
| :psearch xyz | 在当前文件和include包含的文件中查找，显示包含xyz的行 |
| :[n]tnext | 下一个匹配项 |
| :[n]tprev | 上一个匹配项 |
| :tfirst | 第一个匹配项 |
| :tlast | 最后一个匹配项 |
| :tselect tagname | 打开选择项列表 |
| :tag xyz<tab> | 补齐以xyz开头的tag名，继续按<tag>会显示其他的 |
| :tag /xyz<tab> | 补齐名字中含有xyz的tag名 |
| [Ctrl] + o | 调回引用处 注意 nG,ngg,:n 的跳转C-O,C-I,jumps |
| [Ctrl] + i | 跳到下一个点 |
| ：ju[mps] | 显示跳转点 |

* **第三部份：GTAG**

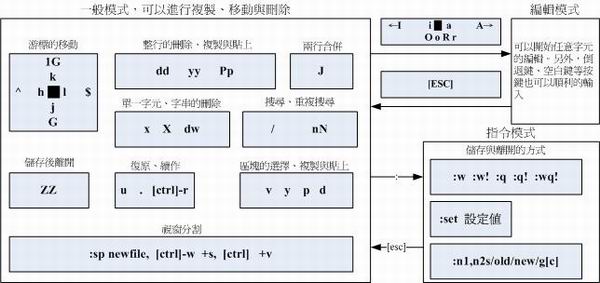
|  |  |
| --- | --- |
| **GTAG** | |
| Gtags funcname | 定位到funcname的定义处 |
| Gtags –r funcname | 查询funcname的引用处 |
| Gtags –s symbol | 定位symbol出现的地方 |
| Gtags –g string | 查询string出现的地方 eg:  Gtags –gi string 忽略大小写 |
| Gtags –f filename | 显示文件中的函数列表 eg:  Gtags –f % 显示当前文件 |
| Gtags –P pattern | 显示路径中包含特定模式的文件。eg:  Gtags –P .h$ 显示所有头文件  Gtags –P /vm/ 显示vm目录下的文件 |

* **第三部份：CSCOPE**

|  |  |
| --- | --- |
| **CSCOPE** | |
| cscope –Rbq | 生成cscope.out文件 |
| :cs add /path/to/cscope.out /your/work/dir | 加入cscope.out文件 |
| :cs find c func | 跳到函数调用处 |
| :cs find d func | 跳刀函数定义处 |

* **第三部份：在线帮助**

|  |  |
| --- | --- |
| **在线帮助** | |
| :h(elp), F1 | 打开总的帮助 |
| :help user-manual | 打开用户手册 |
| :helptags somepath | 为某个路径下的文档生成索引 |
| :helpgrep | 搜索这个歌帮助文档，匹配的列表显示在quickfix窗口中 |
| [Ctrl] + ] | 跳转到tag主题 |
| [Ctrl] + t | 跳回 |
| :ver | 显示版本信息 |



**中文編碼的問題**

|  |
| --- |
| [root@www ~]# **LANG=zh\_TW.big5** |

然後在終端介面工具列的『終端機』-->『設定字元編碼』 -->『中文 (正體) (BIG5)』項目點選一下， 如果一切都沒有問題了，再用 vim 去開啟那個 big5 編碼的檔案，就沒有問題了！以上！報告完畢！

**DOS 與 Linux 的斷行字元**

[root@www ~]# **dos2unix [-kn] file [newfile]**

[root@www ~]# **unix2dos [-kn] file [newfile]**

**語系編碼轉換**

[root@www ~]# **iconv --list**

[root@www ~]# **iconv -f 原本編碼 -t 新編碼 filename [-o newfile]**

