学习 Vim https://github.com/iggredible/Learn-Vim

Chapter 2：窗口操作：

:split fileA(可选) 上下分窗口，可将fileA放置在上方

:vsplit fileB(可选) 左右分窗口，可将fileB放左边

切分后移动当前窗口至：

Ctrl-W h Moves the cursor to the left window

Ctrl-W j Moves the cursor to the window below

Ctrl-W k Moves the cursor to the window upper

Ctrl-W l Moves the cursor to the right window

操作当前窗口和其他窗口：

Ctrl-W v Opens a new vertical split

Ctrl-W s Opens a new horizontal split

Ctrl-W c Closes a window

Ctrl-W o Makes the current window the only one on screen and closes other windows

:tabnew fileA open fileA in a new tab

:tabclose Close the current tab

:tabnext Go to next tab

:tabprevious Go to previous tab

:tablast Go to last tab

:tabfirst Go to first tab

Buffer/Window/Tab之间的关系

**Buffer**是文件在内存中的缓存数据，一个buffer总是也只能对应一个文件，当使用edit(e)时，即将文件内容载入到Buffer里面。

**Window**只负责展示Buffer的数据，而不会影响Buffer。当执行split或vsplit时，实际上是打开了一个新的Window并载入当前的Buffer。一个 Window 在同一时间内只能展示一个 Buffer，一个 Buffer 可以同时被多个 Window 展示，使用上一节中的命令可以切换 Window 中展示的 Buffer。

**Tab**是一系列Window的集合，是一种“布局”。一个Tab上可以有多个Window，不同Tab之间的Window互不影响。

Chapter 3：文件操作：

find命令，find从path中搜索文件，因此需要配置path。

假设有以下目录级：

procise/

include/

src/

device/

:set path=.,/procise/src,,

就可以直接查找include下面的所有文件。

另外，还可以加路径：

:set path+=procise/include/

:set path+=$PWD/\*\* add the entire project directories. \*\* 表示递归搜索当前目录下的所有子目录

vimgrep 自觉不实用。

Chapter 4：基本操作：

vim中的名词：

移动：

h：左

j：下

k：上

l：右

w：移动到下个单词的开头

}：跳到下一段

$：移动到本行末尾

动词：

y 复制文本 yank

d 删除文本，并保存到寄存器 delete

c 删除文本，保存到寄存器，并开启「插入」模式 change

组合：

复制到行末 y$ 删除当前位置到下一单词末dw 更改当前位置到当前段末 c}

当然可以加数字加方向，例如

y2h 复制左边两个字符 d2w 删下面两个单词 c2j 改变下面两行

重复动词可以对当前整行进行操作：

yy dd cc 复制/删除/更改当前行

文本对象：text object

i + object Inner text object

a + object Outer text object

总是以a或i开头，一般说来，以i(inside)开头的文本对象会选择分隔符内部的文本，而以a(around or all)开头的文本对象则会选择包括分隔符在内的整个文本。

非块文本对象：

|  | **说明** |
| --- | --- |
| aw | 当前单词及单词前面或者后面的空格 |
| iw | 当前单词 |
| aW | 当前字串及字串前面或者后面的空格 |
| iW | 当前字串 |
| as | 当前句子及句子前面或后面的空格 |
| is | 当前句子 |
| ap | 当前段落及段落前面或后面的空行 |
| ip | 当前段落 |

w(word), p(paragraph), s(sentence)。( or ) 一对括号（其他括号同理），'一对引号（其他引号同理），t(xml tags)。所以可以用diw删除一个文本对象，可以用daw删除文本对象（包含后面空格）。在可视模式中，可以用iwd删除。

使用h text-object查看具体说明。

块文本对象一般常见于xml tags中。

Chapter 5：移动操作：详细见:h motion.txt

vimdiff FILE\_LEFT FILE\_RIGHT 比较文件不同

ctrl+O 返回 上一次指针出现的位置

———————————————————————————————————————

vim 替换指令

:[range] s/pattern/string/[c,e,g,i]

[range]表被替换范围： 1,7（line 1 to line 7） 1,$（line 1 to end）

pattern 被替换的字串

string 替换的字串

c:每次询问 e：不提示error g：globe 不询问直接替换（每次基本都要加） i：不区分大小写

———————————————————————————————————————

vim recording入门：

recording 是 vim 的一个强大功能。他可以录制一个宏（ Macro)，在开始记录后，会记录你所有的键盘输入，包括在 insert 模式下的输入、正常模式下使用的各种命令等。

具体使用：

第一步：在正常模式下（非 insert 模式、非 visual 模式）按下 q 键盘

第二步：选择 a-z 或 0-9 中任意一个作为缓冲器的名字，准备开始录制宏

第三步：正常的操作，此次所有的操作都会被记录在上一步中定义的缓冲器中

第四步：在非 insert 模式下输入 q 停止宏的录制

第五步：使用 @ + 第二步中定义的缓冲器的名字即可。

例如想把下面文字：

line1

line-2

line3-1

l4

改成：

System.out.println(line1)

System.out.println(line-2)

System.out.println(line3)

System.out.println(l4)

操作如下：

1. 输入gg，光标移动至文章首

2. 输入qt，（缓冲器名字为t），录制动作为：

3. shift+^ 回到行首，i 插入， 输入“System.out.println(”，按esc，shift +$ 至行尾，i插入，输入“)”，按esc，按q停止录制

4. 光标移动到下面一行的任意位置输入 @+t

还可以 输入 100@t，表示执行100次。

· Ctrl + W：关闭浏览器中当前浏览页面

· Alt + F4：快速关闭整个浏览器窗口

· Ctrl + Shift + T：恢复浏览器中之前关闭的窗口，老板走了马上恢复。

· Win/Alt+Tab：切换多个窗口/程序

· Win + D/M：一键返回桌面