2 vim编辑器

- vim(vimsual)是Linux/UNIX系列OS中通用的全屏编辑器。

2.1 vim简介

2.1.1 vim的模式

- vim分为三种模式：普通（命令）模式，编辑状态和视觉模式。在命令模式下，所键入的字符系统均作命令来处理，如:q代表退出，而编辑状态则是用来编辑文本资料的。当你进入vim时，会首先进入命令模式。
- 左下角有INSERT字样就是编辑模式，有VISUAL的字样就是视觉模式，否则就是命令模式
- 从命令模式切换到编辑模式，有如下一些常用命令：

```
a       从光标后面开始添加文本（表示新增 append）
A       从光标所在行的末尾开始添加文本
i       从光标前面开始插入文本（表示插入 insert）
I       从光标所在行的开始处插入文本
```

- 从编辑模式切换到命令模式，按下ESC按键

2.2 vim内常用的命令操作

2.2.1 删除与修改

```
x       删除光标处的字符
dd      删除光标所在的整行
3dd     删除光标所在行以及下面的两行（删除3行）
D或d$    删除光标到行尾的文本，常用语删除注释语句  (等价于d$)
d^      删除光标到行首的文本
dw      删除一个字
d3w     删除三个字
yy      复制光标所在的整行
[n]yy   从光标开始往下复制n行，[n]表示一个整数
p       将复制后的文本粘贴到光标处
u       撤销上次操作
```

2.2.2 光标移动

```
^       光标移动到行首
$       光标移动到行尾
ctrl+d          向下翻半页(down)
ctrl+f          向下翻一页(forward)
ctrl+u          向上翻半页(up)
ctrl+b          向上翻一页(backward)
gg      光标定位到文档头
G       光标定位到文档尾
H       光标定位到当前页首（head）
L       光标定位到当前页的最后一行的行首（lastline）
w       光标往后移一个字 （word）
```

```
b       光标往前移一个字 （block）
[n]+        光标向后移动n行,[n]表示一个整数  10+
[n]-        光标向前移动n行,[n]表示一个整数   10-
[n]G        光标定位到第n行行首, [n]表示一个整数 20G
```

2.2.3 查找与替换

```
    /[str]     查找字符串str，[str]表示要查找的字符串
    回车后会加亮显示所有找到的字符串，接着输入n移动到下一个找到的字符串，输入N移动到上一个找到的字符串

    :s/[src]/[dst]/[i忽略大小写][g处理本行中所有的匹配项] 用dst替换字符串src
    :s/hello/world/ig 替换一行

    :x,y s/[src]/[dst]/ig（x-y行中找）
    :3,6 s/hello/world  将3-6行里面，找到第一个hello替换为world

    :%s/[src]/[dst]/g 将文档中所有src的字符串替换为dst字符串
    :%s/^ //g  将文档每一行的行首的空格去掉
    :%s/^\t//g  将文档每一行的行首的制表符去掉
    提问：如何使用替换命令注释所有块内代码？
    提问：如何使用替换命令转换实现制表符和4个空格之间的转换？
```

2.2.4 可视模式

```
    v       进入可视模式
    ctrl+v  竖向选择模式
```

- 进入可视模式以后，可以使用光标移动指令来移动光标，并且会选中区域
- 批量注释代码，输入步骤如下：

```
    首先按ctrl+v，竖选选中要注释的行
    输入I（注意是大写的I，表示在选中区域的前方），然后输入//
    再按下ESC，就会看到选中的行被注释了
```

2.2.5 文档保存及退出

```
    :q      在未修改文档的情况下退出
    :q!     放弃文档的修改，强行退出
    :w      文档存盘
    :wq     文档存盘退出
    :x      和:wq一样，不过不推荐使用，容易和:X混淆
```

2.2.6 其他操作

```
    :help    查看该命令的帮助提示（不常用，当不小心按F1时，通过:q进行退出）
    :%!xxd   十六进制模式
    :%!xxd -r 返回文本模式，注意中间有一个空格
```

- 如果在编辑过程中不小心按了ctrl+s,vim会处于僵死状态，按ctrl+q可以恢复
- 光标定位

```
执行 vim +3 main.c     表示定位到main.c的第3行
执行 vim +/printf main.c  表示定位到第一个printf处
```

- 多窗口

```
:new 2.c    再打开一个vim，是横向的
:vnew 2.c   再打开一个vim，是纵向的
也可以通过:split，vsplit，sp，vsp
ctrl+w,w（连续按两次w） 两个窗口之间进行切换的方式
```

- 多标签

```
:tabnew 文件名 再打开一个vim标签
gt 切换到下个标签
gT 切换到上个标签
```

- 代码格式对齐使用

```
使用 = 可以进行代码对齐
gg=G     对齐全文件
gg=10gg 对齐第一行到第十行
在可视模式选中代码块，再按下=也能对齐代码块
```

2.2.7 总结

- 基本上，vim的命令是保持 "操作""参数""对象" 的基本结构。
- 操作包括了g（跳转），d（删除），c（change），y（yank 拷贝）等等
- 参数包括了i（之前），a（后面），t（to），数字等等
- 对象就包括了w（单词），s（句子），b（块），)（句子），}（块）等等

```
例子：
修改括号内内容:ct)
```

2.3 vim外使用到的命令

- 为了批量替换文件内容，需要在shell上（也就是vim编辑器以外）调用sed命令

```
grep printf main.c
sed "s/printf/puts/g" main.c 这里不会修改文件内容，只会显示替换好的结果
sed -i "s/printf/puts/g" main.c 修改文件内容
```

> - 提问：怎么样找到所有的.c后缀文件（不只在本目录下的），将里面的printf修改为puts?

- 很多时候，需要对比两个代码文件的区别，比如新旧版本代码到底执行了哪些修改。使用vimdiff命令就可以轻松实现比较

```
vimdiff [文件1] [文件2]
```

在比较界面里面的操作方式和双窗口的vim的操作方式是一致的

- 在shell里面调用命令的时候可以使用 +[行号] 参数，这样的进入文件的时候光标的默认位置会在该行的行首

```
    vim +[行号] 文件名
```

2.4 修改配置文件

- .vimrc里面存放了vim的配置文件。rc是running command的意思，每次启动vim的时候，都会读取.vimrc里面的命令然后执行（"是单行注释的意思）
- 配置文件有两种，一种全局的配置文件，对所有用户生效，放置在/etc目录下。而本用户的配置文件就放置在用户的家目录下

```
    "将下面的内容复制到.vimrc文件里面，然后重新启动vim即可
    set nu "设置行号 set number
    set hlsearch "高亮所有查找结果
    syntax on "语法高亮
    set showmode "显示当前模式
    set showcmd "显示输入的命令
    set cursorline "光标所在行添加横线显示
    set shiftwidth=4 "设置每次缩进的宽度
    set cindent "C语言缩进规则
    set path +=.,/usr/include "使用gf命令查找的目录
    set fileencodings=utf-8,gb18030,gbk,gb2312
    filetype indent on "文件类型检查
```

- 命令行可以使用PS1来修改配色方案，只需要在家目录.bashrc文件添加依据export语句即可

```
    export PS1="\[\e[37;40m\][\[\e[32;40m\]\u\[\e[37;40m\]@\h \[\e[36;40m\]\w\
[\e[0m\]]\\$ "
```

- 提问：怎么样修改.bashrc 让每次rm的时候等价于mv到trashCan目录?

2.5 查看帮助

- 在命令模式下键入:help或者按下F1，就能进入帮助文件
- 键入:help 信息，就可以查看该信息的具体帮助文件

2.6 vimtutor

在命令行输入vimtutor可以进行vim的训练模式

```
$ vimtutor
```

在vimtutor可以熟悉vim的各项操作，建议不熟练的同学可以每天抽出一定的时间去练习