第五章：vim 编辑器

尚硅谷云计算 Linux 课程

版本：V1.0

讲师：沈超

一　vi 编辑器简介

vim 是一个全屏幕纯文本编辑器，是 vi 编辑器的增强版，我们主要讲解的是 vim 编辑器。可以利用别名让输入 vi 命令的时候，实际上执行 vim 编辑器，例如：

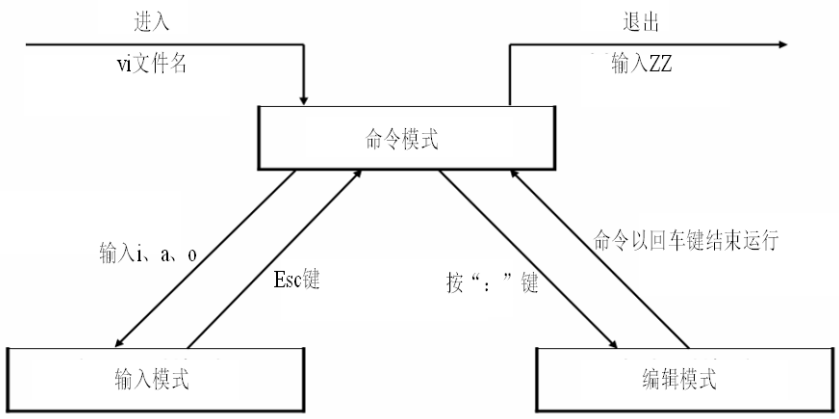
```
[root@localhost ~]# alias vi='vim'
#定义别名
```

这样定义的别名是临时生效，如果需要永久生效，请放入环境变量配置文件（~/.bashrc）

二　vim 基本使用

1　vim 的工作模式

vim 工作在三种模式之下：



命令模式：是主要使用快捷键的模式，是我们后面学习的重点。命令模式想要进入输入模式，可以使用以下的方式：

插入命令

命令	作用
a	在光标所在字符后插入
A	在光标所在行尾插入
i	在光标所在字符前插入
I	在光标所在行行首插入
o	在光标下插入新行
O	在光标上插入新行

输入模式：主要用于文本编辑，和记事本类似，输入数据就好。

末行模式（编辑模式）：

```
:w  保存不退出
    ：w  新文件名       把文件另存为新文件
:q  不保存退出
:wq 保存退出
:!  强制
    :q!       强制不保存退出，用于修改文件之后，不保存数据退出。
   :wq!       强制保存退出，当文件的所有者或者 root 用户，对文件没有写权限的时候，强制写入数据使用
```

2 命令模式操作

2.1 移动光标

1）上下左右移动光标

```
上、下、左、右方向键          移动光标
H、j、k、l                   移动光标
```

2）把光标移动到文件头或尾

```
gg          移动到文件头
G           移动到文件尾（shift+g）
```

3）移动到行首或行尾

```
^           移动到行首
$           移动到行尾
```

4）移动到指定行

```
:n          移动到第几行
```

这里 n 是数字，准备移动到第几行，就用哪个数字。

2.2 删除或剪切

1）删除字母

```
x           删除单个字母
nx          删除 n 个字母
```

n 是数字，如果打算从光标位置删除连续的 10 个字母，可以使用“10x”即可。删除字母并不符

合使用习惯，我们更习惯在编辑模式中，用“Backspace”键删除字母。

2）删除整行或剪切

```
dd          删除单行
ndd         删除多行
:n1,n2d     删除指定范围的行
```

删除整行或多行，这是比较常用的删除方法。这里的 dd 快捷键既是删除，也是剪切。删除内容放入了剪切板，如果不粘贴就是删除，如果粘贴就是剪切。粘贴方法如下：

```
p           粘贴到光标后
P（大）      粘贴到光标前
```

3）从光标所在行删除到文件尾

一直有同学问超哥，是否可以删除整篇文档，vim 没有删除整篇文档的快捷键，但是可以这样：

```
dG          从光标所在行删除到文件尾
```

“d”是删除行，“G”是文件尾，连起来就是从光标所在行删除到文件尾。如果把光标放在文件首，那么“dG”就变成了删除整篇文档了。

2.3 复制

```
yy          复制单行
nyy         复制多行
```

复制之后的粘贴，依然可以使用 p 键或 P（大）键

2.4 撤销

```
u           撤销
ctrl+r      反撤销
```

“u”键能一直撤销到文件打开时的状态，类似 Windows 下“ctrl+z”键的作用。

“ctrl+r”能一直反撤销到最后一次操作状态，类似 Windows 下“ctrl+y”键的作用。

2.5 替换

```
r       替换光标所在处的字符
R       从光标所在处开始替换字符，按 ESC 结束
```

“r”键替换单一字符，不用进入输入模式，实际使用时，比进入输入模式删除后再修改，要方便。

2.6vim 配置文件

这次末行模式参数设置，多数需要在 vim 中才能生效。

设置参数	含　义
:set nu :set nonu	显示与取消行号。
:syntax on :syntax off	是否依据语法显示相关的颜色帮助。在 Vim 中修改相关的配置文件或 Shell 脚本文件时（如前面示例的脚本/etc/init.d/sshd），默认会显示相应的颜色，用来帮助排错。如果觉得颜色产生了干扰，则可以取消此设置
:set hlsearch	设置是否将查找的字符串高亮显示。默认是 set hlsearch 高亮显示

:set nohlsearch	
:set ruler :set noruler	设置是否显示右下角的状态栏。默认是 set ruler 显示
:set showmode :set noshowmode	设置是否在左下角显示如“—INSERT--”之类的状态栏。默认是 set showmode 显示
:set list :set nolist	设置是否显示隐藏字符（Tab 键用“^I”表示，回车符用“$”表示）。默认是 nolist 显示。如果使用 set list 显示隐藏字符，类似“cat -A 文件名”。

vim 支持更多的设置参数，可以通过“:set all”进行查看。

大家会发现，这些设置参数都只是临时生效，一旦关闭文件再打开，又需要重新输入。如果想要永久生效，需要手工建立 vim 的配置文件“~/.vimrc”,把你需要的参数写入配置文件就永久生效了。

补充：Windows 下回车符在 Linux 中是用“^M$”符号显示，而不是“$”符。这样会导致 Windows 下编辑的程序脚本，无法在 Linux 中执行。这时可以通过命令“dos2unix”，把 Windows 格式转为 Linux 格式，当然反过来“unix2dos”命令就是把 Linux 格式转为 Windows 格式。这两个命令默认没有安装，需要手工安装才能使用。

2.7 查找

```
/查找内容        从光标所在行向下查找
?查找内容        从光标所在行向上搜索
n                下一个
N                上一个
```

2.8 替换

```
:1,10s/old/new/g        替换 1 到 10 行的所有 old 为 new
:%s/old/new/g           替换整个文件的 old 为 new
```

替换字符串，我举几个例子：在 shell 中“#”开头是注释，那我是否可以注释文件的前 10 行呢？手工一个一个注释很麻烦，那么批量替换吧：

```
:1,10s/^/#/g            注释 1 到 10 行
:1,10s/^#//g            取消注释
```

而在 C 语言，PHP 语言等大多数语言中，是使用“//”开头作为注释的，我们当然可以用 vim 来写这些程序语言脚本，那么批量加入“//”注释吧：

```
:1,10s/^/\/\//g         1 到 10 行，行首加入//
:1,10s/^\/\///g         取消 1 到 10 行行首的//
```

三. vim 使用技巧

1. 在 vim 中导入其他文件内容或命令结果

1.1 导入其他文件内容

```
:r 文件名               把文件内容导入光标位置
```

可以把其他文件的内容导入到光标所在位置

1.2 在 vim 中执行系统命令

```
:!命令                  在 vim 中执行系统命令
```

这里只是在 vim 中执行系统命令，但并不把系统命令的结果写入到文件中。主要用于在文件编辑中，查看系统信息，如时间。

1.3 导入命令结果

```
:r  !命令               在 vim 中执行系统命令，并把命令结果导入光标所在行
```

在 vim 中执行系统命令，并把命令结果导入光标所在行。

2. 设定快捷键

```
:map 快捷键 快捷键执行的命令                自定义快捷键
```

vim 允许自定义快捷键，常用的自定义快捷键如下：

```
:map ^P I#<ESC>      按“ctrl+p”时，在行首加入注释
:map  ^B ^x          按“ctrl+b”时，删除行首第一个字母（删除注释）
```

注意：^P 快捷键不能手工输入，需要执行 ctrl+V+P 来定义，或 ctrl+V ，然后 ctrl+P。^B 快捷键也是一样

3. 字符替换

```
:ab 源字符 替换为字符              字符替换
```

在 vim 编辑中，有时候需要频繁输入某一个长字符串（比如邮箱），这时使用字符串替换，能增加输入效率，例如：

```
:ab mymail shenchao@163.com      当碰到“mymail”时，转变为邮箱
```

注意：“源字符”不应设置的太短，否则有可能影响输入。

4. 多文件打开

在 vim 中可以同时打开两个文件，只要执行如下命令：

```
[root@localhost ~]# vim  -o  abc  bcd
[root@localhost ~]# vim  -O  abc  bcd
#-o     小写 o 会上下分屏打开两个文件
#-O     大写 O 会左右分屏打开两个文件
```

这样可以同时打开两个文件，方便操作。如果是“-o”上下打开两个文件，可以通过先按“ctrl+w”，再按“上下箭头”的方式在两个文件之间切换。

如果是“-O”左右打开两个文件，可以通过先按“ctrl+w”，再按“左右箭头”的方式在两个文件之间切换。