第7讲：vim编辑器

**文本编辑神器**



系统管理员的重要工作就是得要修改与设定某些重要软件的设定档，因此至少得要学会一种以上的文字界面的文书编辑器。 在所有的 Linux distributions 上头都会有的一套文书编辑器就是 vi ，而且很多软件预设也是使用 vi 做为他们编辑的界面， 因此鸟哥建议您务必要学会使用 vi 这个正规的文书编辑器。此外，vim 是进阶版的 vi ， vim 不但可以用不同颜色显示文字内容，还能够进行诸如 shell script, C program 等程序编辑功能， 你可以将 vim 视为一种程序编辑器！

https://blog.csdn.net/breakyang/article/details/104247088

1为何要学 vim

由前面一路走来，我们一直建议使用文字模式来处理 Linux 系统的设定问题，因为不但可以让你比较容易了解到 Linux 的运作状况，也比较容易了解整个设定的基本精神，更能『保证』你的修改可以顺利的被运作。 所以，在 Linux 的系统中使用文字编辑器来编辑你的 Linux 参数设定档，可是一件很重要的事情呦！也因此呢，系统管理员至少应该要熟悉一种文本编辑器才行的！

**在 Linux 的世界中，绝大部分的设定档都是以 ASCII 的纯文字形态存在**，因此利用简单的文字编辑软件就能够修改设定了！ 与微软的 Windows 系统不同的是，如果你用惯了 Microsoft Word 或 Corel Wordperfect 的话，那么除了 X window 里面的图形界面编辑程序(如 xemacs )用起来尚可应付外，在 Linux 的文字模式下，会觉得文书编辑程序都没有视窗界面来的直观与方便。

那么 Linux 在文字界面下的文书编辑器有哪些呢？其实有非常多喔！常常听到的就有： [emacs](http://www.gnu.org/software/emacs/" \t "https://linux.vbird.org/linux_basic/centos7/_blank), [pico](https://en.wikipedia.org/wiki/Pico_(text_editor)" \t "https://linux.vbird.org/linux_basic/centos7/_blank), [nano](http://sourceforge.net/projects/nano/" \t "https://linux.vbird.org/linux_basic/centos7/_blank), [joe](http://sourceforge.net/projects/joe-editor/" \t "https://linux.vbird.org/linux_basic/centos7/_blank), 与 [vim](http://www.vim.org/" \t "https://linux.vbird.org/linux_basic/centos7/_blank) 等等([注1](https://linux.vbird.org/linux_basic/centos7/0310vi.php" \l "ps1))Nano编辑器

**gedit是图形模式下才能使用的一个图形界面文本编辑器**

nano编辑器

保存文件ctrl+o(字母欧) 再按回车

退出是 ctrl+x

Ctrl + o 保存文件，需要按回enter键确认

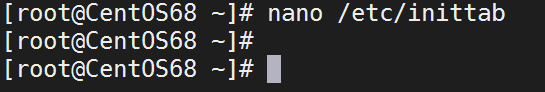
Ctrl+x 在文件已经保存好的情况下，直接退出

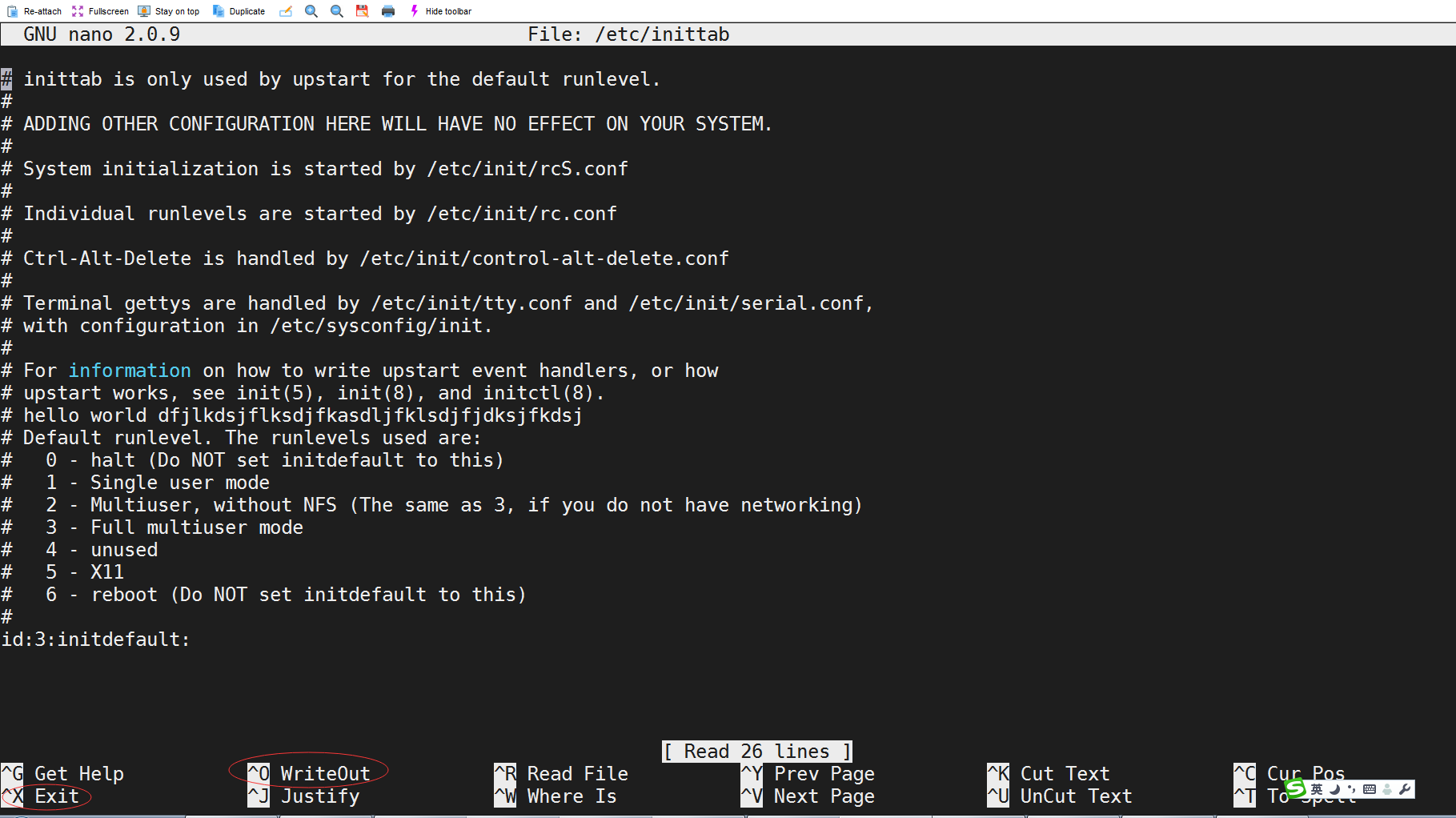
如果文件被修改后并没有保存，这个时候会提示你是否保存

定位行号：ctrl + shift + -(键)

**适合短和小的文件编辑一下是可以滴,但是长的配置文件用起来不方便了**

**范例：修改/etc/inittab这个文件，将最后一行数字修改，将3改成5**

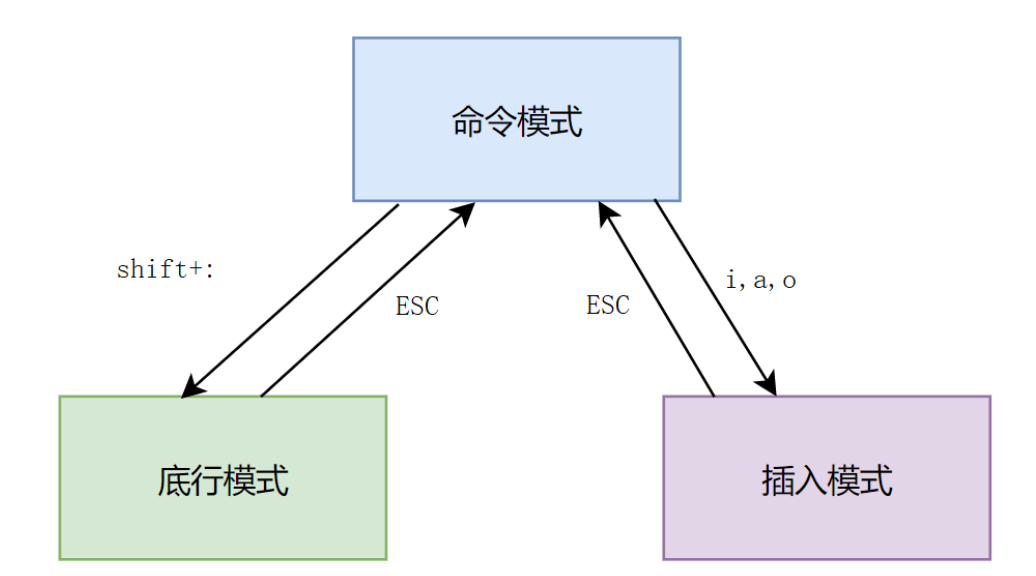




2 Vim模式介绍

**vim 是 一个模式编辑器，击键行为是依赖于 vim的“模式**

1 三种主要模式和转换



**命令模式（Command mode）**，**输入模式（Insert mode）**和**底线命令模式（Last line mode）**

1 命令或普通模式[command mode]：

默认模式，可以实现移动光标，剪切/粘贴文本

以 vim 打开一个档案就直接进入一般指令模式了(这是预设的模式，也简称为一般模式)。在这个模式中， 你可以使用『上下左右』按键来移动游标，你可以使用『删除字符』或『删除整列』来处理档案内容， 也可以使用『复制、贴上』来处理你的文件资料。

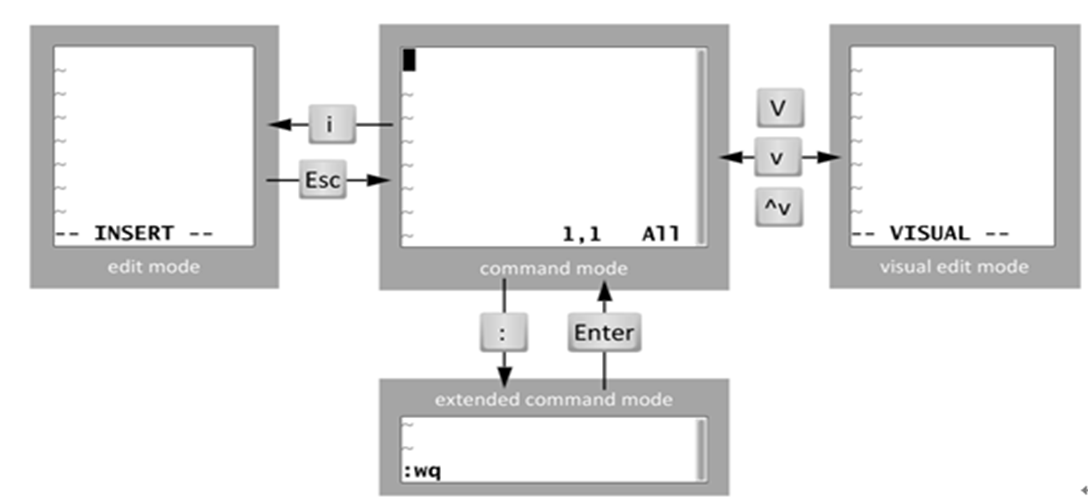
2 插入(insert mode)或编辑模式：

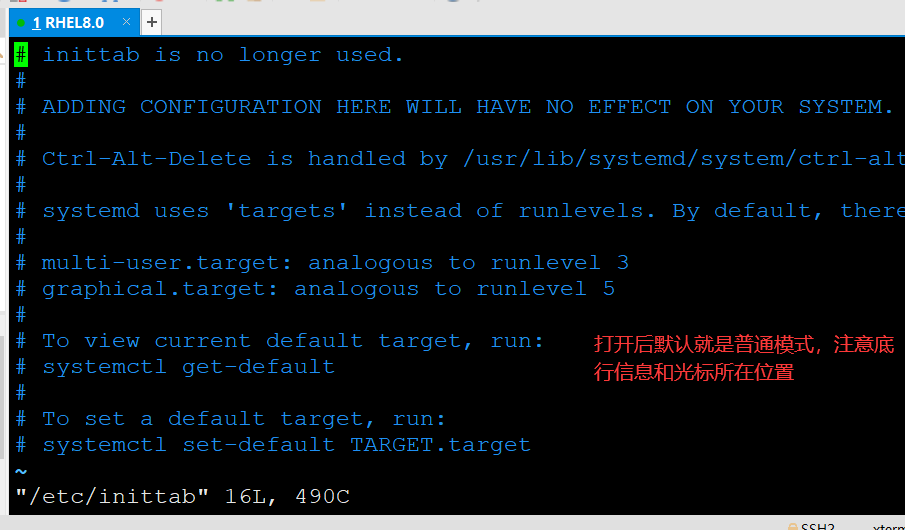
用于修改文本

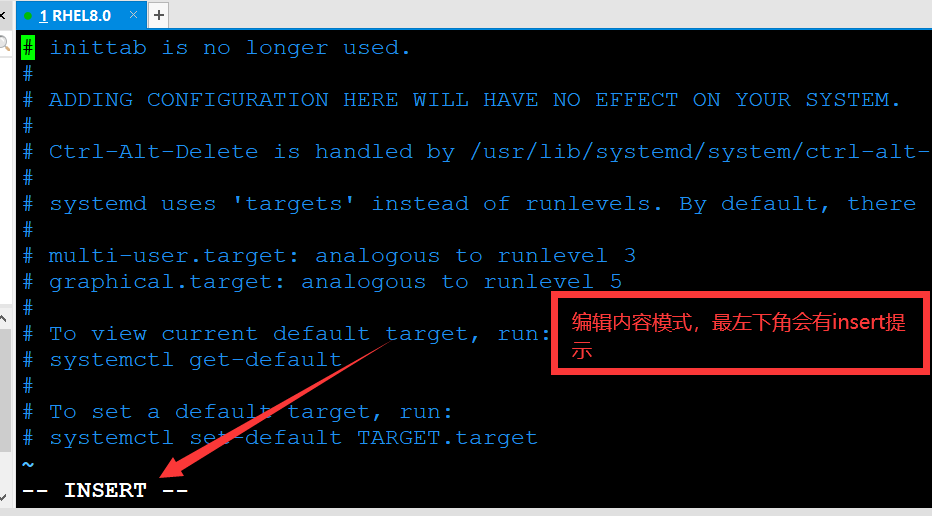
在一般指令模式中可以进行删除、复制、贴上等等的动作，但是却无法编辑文件内容的！ 要等到你按下『i, I, o, O, a, A, 』等任何一个字母之后才会进入编辑模式。注意了！通常在 Linux 中，按下这些按键时，在画面的左下方会出现『 INSERT 或 REPLACE 』的字样，此时才可以进行编辑。而如果要回到一般指令模式时， 则必须要按下『Esc』这个按键即可退出编辑模式。

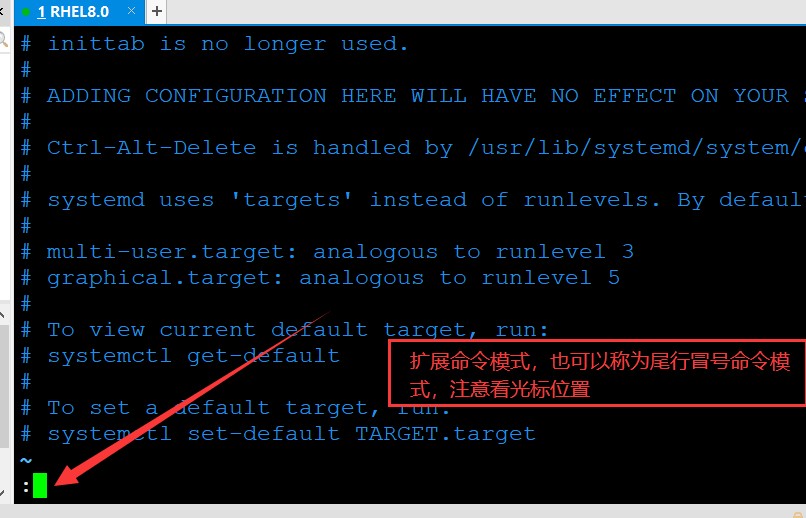
**3 扩展命令(extended command [command-line mode])或命令(末)行模式：**

在一般模式当中，输入『 : / ? 』三个中的任何一个按钮，就可以将游标移动到最底下那一列。在这个模式当中， 可以提供你『搜寻资料』的动作，而读取、存档、大量取代字符、离开 vi 、显示行号等等的动作则是在此模式中达成的！









1. command mode
2. edit mode
3. ~~visual edit mode （可视模式，可视行模式，可视块模式）~~
4. extended command mode

**Vim 是以命令模式为中心，通过不同的指令进入其他模式**

**命令模式 --> 插入模式**

i insert, 在光标所在处输入

I 在当前光标所在行的行首输入

a append, 在光标所在处后面输入

A 在当前光标所在行的行尾输入

o 在当前光标所在行的下方打开一个新行

O 在当前光标所在行的上方打开一个新行

**插入模式 --- ESC-----> 命令模式**

**命令模式 ---- : ----> 扩展命令模式**

**扩展命令模式 ----ESC,enter----> 命令模式**

**PS: PS**

一定搞清楚vim处于什么模式状态下，打开vim,之后默认就是普通模式，如果要想进入编辑内容模式，按字母i进入

如何从编辑内容模式回到普通模式： ESC

从普通命令模式进入到尾行命令： 输入冒号或者 / 都可以

尾行命令输入完成后，回车键执行，如果你不想执行，连续按两次ESC可以回到普通模式

**注意：从编辑内容模式不可以直接切换尾行命令模式**

3 快速上手范例

演示如何创建编辑及保存退出

范例1 创建编辑保存

#vim f1.txt

文件内容：

hello,this is the vim editor file

Welcome to the vim world

编辑文件内容

保存文件，

另存文件，

保存退出，

扩展命令模式之冒号后的命令

:w

:w 文件名

放弃保存退出 按:q！

**如果编辑了文件的内容，又想放弃保存退出，则要使用强行退出的方式**

**：q!**

范例2 行号设置及跳转行

设置行号及定位

Web服务器的配置文件

/etc/httpd/conf/httpd.conf

大家操作时把上面文件复制到自己的主目录下

# cp /etc/httpd/conf/httpd.conf ./

#vim httpd.conf

**设置文件是否显示行号（这个设置是临时的）**

**尾行冒号 :set nu 取消行号 :set nonu**

4 普通命令模式下操作

（普通）命令模式下常用功能键

|  |  |
| --- | --- |
| 命令 | 作用 |
| gg | 把光标定位到文件首行 |
| G | 把光标定位到文件最后一行 |
| nG | 定位到第n行 |
| dd | 删除（剪切）**光标所在整行** |
| ndd | 删除（剪切）**从光标处开始的连续n行** 另外一种dnd |
| x | 删除光标所在字符 |
| r字符 | 把当前光标所在字符替换成按下去的字符，只能替换一个字符 |
| **u** | **撤销上一步的操作** |
| yy | 复制光标所在整行 |
| nyy | 复制从光标处开始的n行 |
|  |  |
| p | 将之前删除（剪切）（dd）或复制（yy）过的数据粘贴到光标所在行的下面 |
| ZZ | 保存退出（大写字母按两次的意思）shift+小写字母 |
| ZQ | 不保存退出（大写字母按两次的意思） |
| 0 | 数字0表示光标移动行首 |
| $ | 表示光标移动行尾 |

**范例：打开文件**

**[root@CentOS68 ~]# vim /etc/httpd/conf/httpd.conf**

**行号的跳转**

**回首行 gg (两次小写字母gg)**

**到文件最后一行 大写字母G (shift+g)**

**定位行号**

**定位行号，比如定位到httpd.conf的136行**

**普通模式下 nG n代表行号就是数字 这个时候按 136G**

**尽量不要去按小数字键盘**

在命令模式下按

136G

按好就直接定位到136行了

问题：大写字母G应该怎么输入，在键盘上应该怎么按

Shift + g

？？？？怎么确认是命令模式

5 底行命令模式下的操作

底行命令模式[**之冒号命令**]

**号外，号外**

**每一次输入指令后回车后自动回命令模式，如果要再进这个底行冒号命令模式就需要重新输入shift+:才能继续输入指令。**

命令模式切换到末行模式

1. ： //对文本的设置或保存工作

2. / //对文本进行全文向下搜索字符串 string

3. ？ //对文本进行全文向上搜索字符串 string

**在命令模式下**

**按“:”进入Ex模式 ，创建一个命令提示符: 处于底部的屏幕左侧**

输入命令后，按回车执行，执行完成后，自动回到普通模式

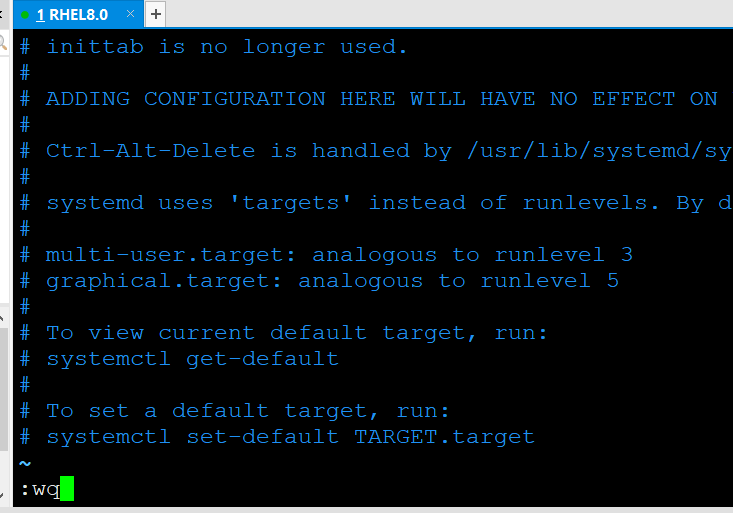
**下面命令均是尾行冒号命令哈**

|  |  |
| --- | --- |
| 命令 | 作用 |
| w | 保存 |
| w filename | 将编辑的数据储存成另一个文件（类似另存新档），  但是：这个和我们的word不同的地方是，另存文件后，依然是编辑的原来的文件 |
| q | 退出，是已经对文件保存过了，或者压根没有修改过 |
| q! | 强制退出（放弃对文档的修改内容） |
| wq! | 强制保存退出 (root才可以)，文件所有者 |
| *!命令* | *执行该命令* |
| *r!command* | *读入命令的输出到当前文件光标的地方* |
| *r filename* | *读文件内容到当前文件中* |
| **整数** | **跳转到该行** |
| **$** | **回车 表示到最后一行** |
| **nd** | **表示删除第n行，n代表了行号** |
| **n,md** | 表示删除从第n到m行  **如果 1，$d表示把文件全部内容删除了** |
| **n,m co k** | 表示把从第n到m行连续行拷贝到第k行的后面  :1005,1012 co 996  表示把1005到1012行的内容copy到996行的下面 |
| 替换功能 |  |
| s/one/two | 将当前光标所在行的第一个one替换成two |
| s /one/two/ g | 将当前光标所在行的所有one替换成two |
| %s/one/two/g | 将全文中的所有one替换成two |
| n,m d | 表示将从n行到m行内容删除 |

zhangsan.com

malin.com

范例：



输入：wq就是保存退出

**扩展模式(底行)下搜索，**

**在命令模式下，按 / 或者 ？进扩展命令模式**

|  |  |
| --- | --- |
| 命令 | 作用 |
| ?字符串 | 在文本中从下至上搜索该字符串 |
| /字符串 | 在文件中从光标行开始向下搜索该字符串，如果搜索到文件尾，又继续从文件开头搜索，搜索时是区分大小写的  查找时忽略大小写字母 /关键字\c |

**范例**

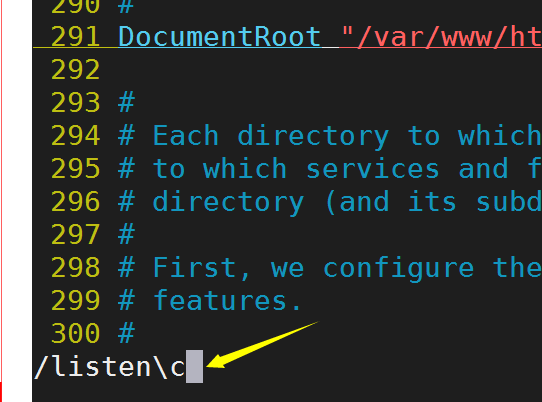
**利用关键字进行查找**

**查找文件中的内容**

**在命令模式下输入 /**

**这个时候光标会跳到尾行（其实就是我们状态栏那一行）**

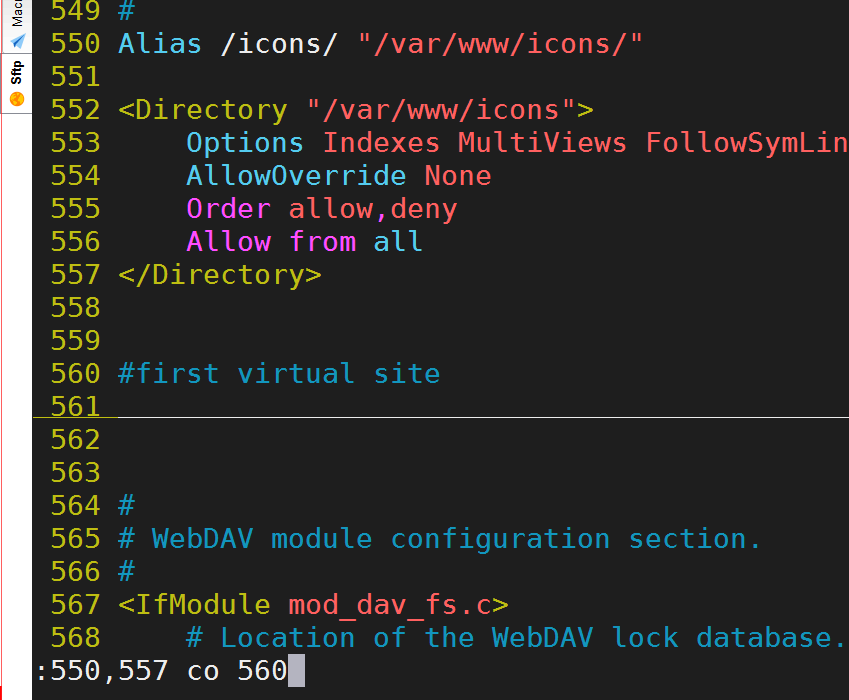
**查找时忽略大小写字母 /关键字\c**



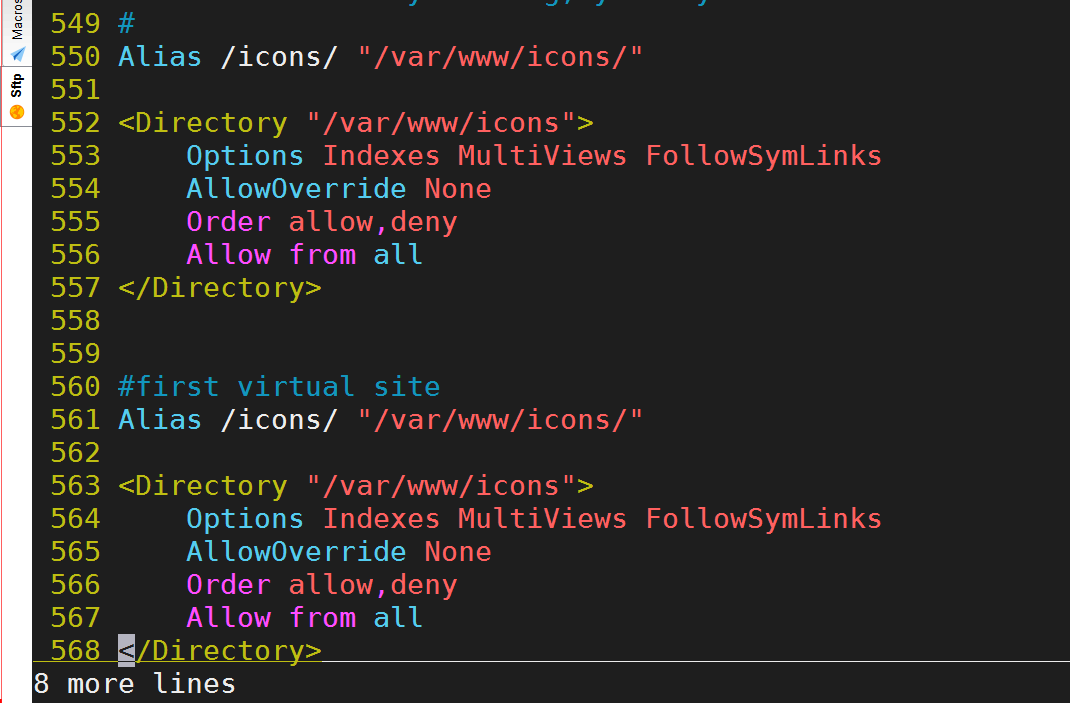
复制一段内容到指定位置

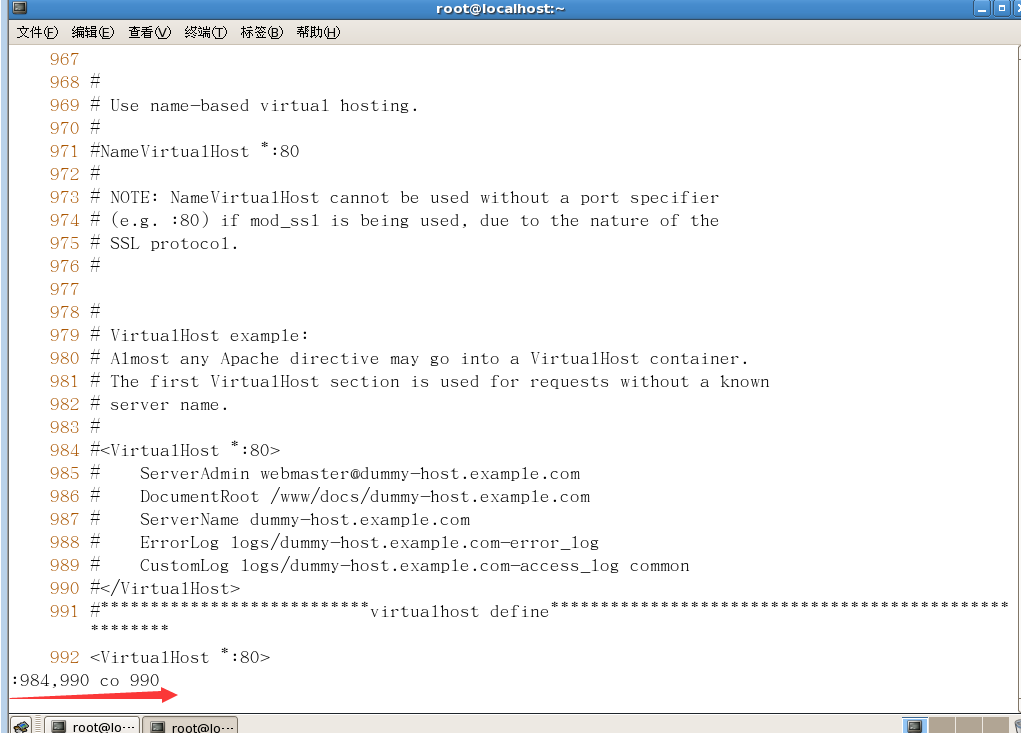
：开始行，结束行 co 哪一行的下面

栗子把550行到557行的内容复制到560行的后面



回车后执行





6 vim配置文件

1 Vim配置文件的作用

　　Vim启动时，会根据配置文件（.vimrc）来设置 Vim，因此我们可以通过此文件来定制适合自己的 Vim

2 Vim配置文件分类

1 系统Vim配置文件/etc/vimrc

　　所有系统用户在启动Vim时，都会加载这个配置文件。默认目录位于/etc/vimrc

2 用户Vim配置文件~/.vimrc

　　当前用户在启动Vim时，会加载家目录下的配置文件。默认目录位于~/.vimrc

3 配置文件加载优先级~/.vimrc > /etc/vimrc

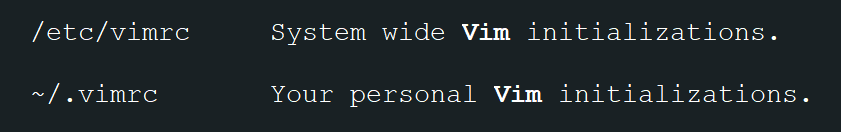
　　vim启动时，优先读取~/.vimrc配置文件，再去读取/etc/vimrc的配置文件

4 配置文件的环境参数

　　用:set all可以查看配置文件的所有环境参数，以下列出几个常用的环境参数

/etc/vimrc System wide Vim initializations.

~/.vimrc Your personal Vim initializations.



**作用：要每次打开都能显示行号，以及搜索时忽略大小写**

**vim配置文件的名字和位置,需要按要求创建**

第1个位置：~/.vimrc 只能当前登录用户

/root/.vimrc /home/tom/.vimrc

第2个位置： /etc/vimrc

范例：

#vim ~/.vimrc

"打开行号

set nu

"打开光标所在行的下划线

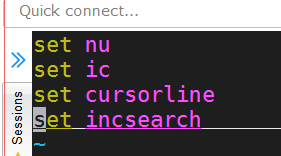
set cul

"搜索时忽略大小写。

set ic

" 输入搜索内容时就自动匹配搜索到结果

set incsearch



备注：这个配置文件中的注释只能使用 “ （左双引号），不能使用#号

|  |  |
| --- | --- |
| 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。预设值是 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~ 档名的档案，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列说明吗？ 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列。 |
| :set backspace=(012) | 一般来说， 如果我们按下 i 进入编辑模式后，可以利用倒退键 (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 来设定囉～ 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 | 显示与系统预设值不同的设定参数， 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，在编辑一个纯文字档时，如果开头是以 # 开始，那么该列就会变成蓝色。 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字档案，要避免颜色对你的荧幕产生的干扰，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 』。如果你常常发现注解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！ |

Vim小抄



## vim简答题部分（来自鸟哥网站）：

1 我用 vim 开启某个档案后，要在第 34 列向右移动 15 个字符，应该在一般指令模式中下达什么指令？

(1)先按下 34G 到第 34 列；(2)再按下 [ 15 + 向右键 ]，或 [ 15l ] 亦可！

2 在 vim 开启的档案中，如何去到该档案的页首或页尾？

去页首按下 1G 或 gg ；去页尾按下 G 即可

3 在 vim 开启的档案中，如何在游标所在列中，移动到行头及行尾？

移动到行头，按 0 ，移动到行尾按 $ 即可！

4 vim 的一般指令模式情况下，按下『 r 』有什么功能？

取代游标所在的那个字符

5 在 vim 的环境中，如何将目前正在编辑的档案另存新档名为 newfilename？

:w newfilename

6 在 linux 底下最常使用的文书编辑器为 vim ，请问如何进入编辑模式？

在一般指令模式底下输入： i, I, a, A 为在本列当中输入新字符；(出现 –Insert- )  
在一般指令模式当中输入： o, O 为在一个新的一列输入新字符；  
在一般指令模式当中输入： r, R 为取代字符！（左下角出现 –Replace-）

7 在 vim 软件中，如何由编辑模式跳回一般指令模式？

**可以按下[Esc]**

8 在 vim 环境中，若上下左右键无法使用时，请问如何在一般指令模式移动游标？

[h, j, k, l]分别代表[左、下、上、右]

9 在 vim 的一般指令模式中，如何删除一列、n列；如何删除一个字符？

分别为 dd, ndd, x 或 X （dG 及 d1G 分别表示删除到页首及页尾）

10 在 vim 的一般指令模式中，如何复制一列、n列并加以贴上？

分别为 yy, nyy, p 或 P

11 在 vim 的一般指令模式中如何搜寻 string 这个字符串？

?string (往前搜寻)  
/string (往后搜寻)

12 在 vim 的一般指令模式中，如何取代 word1 成为 word2，而若需要使用者确认机制，又该如何？

:1,$s/word1/word2/g 或  
:1,$s/word1/word2/gc （需要使用者确认）

13 在 vim 目前的编辑档案中，在一般指令模式下，如何读取一个档案 filename 进来目前这个档案？

:r filename

14 在 vim 的一般指令模式中，如何存档、离开、存档后离开、强制存档后离开？

:w； :q： :wq； :wq!

15 在 vim 底下作了很多的编辑动作之后，却想还原成原来的档案内容，应该怎么进行？

直接按下 :e! 即可恢复成档案的原始状态！

16 我在 vim 这个程序当中，不想离开 vim ，但是想执行 ls /home 这个指令，vim 有什么额外的功能可以达到这个目的：

事实上，可以使用[ :! ls /home ]不过，如果你学过后面的章节之后，你会发现，执行[ ctrl + z ]亦可暂时退出 vim 让你在指令列模式当中执行指令喔！

## 7 Vim实验操作

### 第一部份:基础练习

**Vim小练习1**

**在当前用户主目录下创建一个文件 vim file1.txt**

1. 在文件中输入两行内容,

This is the vim txt

This is second line

1. 然后保存退出
2. 再次打开这个文件
3. 输入第三行内容

This is the third line

5 将这个文件另存为 /tmp/file1.txt

6 保存退出

**Vim小练习2**

**打开文件[root@CentOS68 ~]# vim /etc/httpd/conf/httpd.conf**

这个文件是apache (web服务器的一个配置文件)

1. 打开文件行号 尾行命令 :set nu
2. 切到文件尾 大写字母 G
3. 回到文件首 gg (连按两次g字母)

**Vim小练习3**

**利用vim打开/etc/inittab文件**

1. 打开行号 :set nu
2. 定位到文件第一行，定位到文件最后一行（命令模式下的操作）

gg首行 G尾行

1. 行号定位到第17行 17G 数字G
2. 查找关键字 initdefault
3. 在命令模式下，输入 /查找的关键字 /initdefault
4. 找到设置启动级别的所在行，进入编辑内容模式将数字5改成3
5. 保存退出 :wq

### 第二部份：使用配置文件

**1、创建vim配置文件，内容至少包含如下**

配置文件可以参看下面内容

这个文件的双引号 (")是注释

#vim ~/.vimrc

"vim的配置文件，这个是注释用的标记，不是#号，下面的设置是我们常用的几个设置

set ic "搜索时忽略大小写。

set cursorline "突出显示当前行

set incsearch " 输入搜索内容时就显示搜索结果

set nu "可以在每一列的最前面显示行号啦！

“下面内容不要求可试试效果

set hlsearch "搜索高亮度反白

set backspace=2 "可随时用退格键删除

set autoindent "自动缩排

set ruler "可显示最后一列的状态

set showmode "左下角那一列的状态

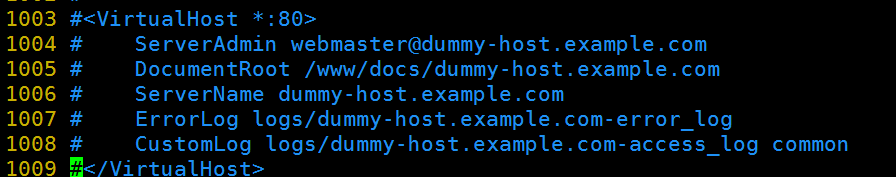
set bg=dark "显示不同的底色色调

syntax on "进行语法检验，颜色显示。

**2 打开apache的配置文件**

[root@localhost ~]# vim /etc/httpd/conf/httpd.conf

1. 打开之后，看看文件有多少行，快速到最后一行
2. 查找listen 并将端口改成8080
3. ServerAdmin 改成你的邮箱
4. 利用尾行命令方式将文件最后虚拟主机头配置的内容（如下图所示的几行内容）复制一份到文件的最后



## RK8新实验：

先执行如下操作,此操作是先vim的配置文件删除掉

[root@rk8 ~]#rm -f .vimrc



### 第一部份：基础操作

1 在当前用户主目录下，创建文件，文件名命名为：学号姓名拼音.txt，在文件中输入如下两行内容

 Hello,Welcome to my linux server

I am 学号姓名拼音

2 内容编辑完成后保存退出

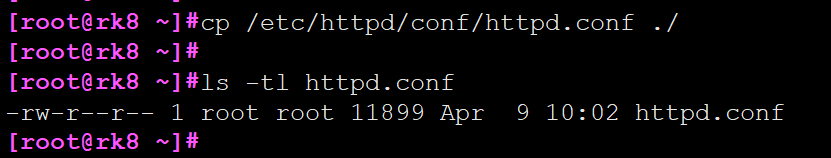
3 利用cat命令显示上面这个文件内容

### 第二部份 进阶操作

将apache的配置文件httpd.conf复制一份到当前用户的主目录下再用vim编辑器打开这个文件

#cp /etc/httpd/conf/httpd.conf ./

#vim httpd.conf



1 打开行号

2 分别利用命令模式下功能键和底行命令两种方式实现下面的操作：定位到文件第1行和文件最后一行，看看这个文件总的有多少行

3 用命令模式和底行命令两种方式定位到文件的45行处，看看是什么内容

4 在命令模式下删除文件的前5行，然后将光标定位到第10行处，在命令模式下按下p字母，看看是什么情况

5 将光标定位到文件第1行，在底行模式下利用正向查找，查找关键字为Listen,找到要求将端口修改为8080后保存文件

6 将光标定位到文件第1行，在底行模式下利用正向查找，查找关键字为DocumentRoot，并找到DocumentRoot "/var/www/html"所在行，将/var/www/html修改为/www/website/htdocs

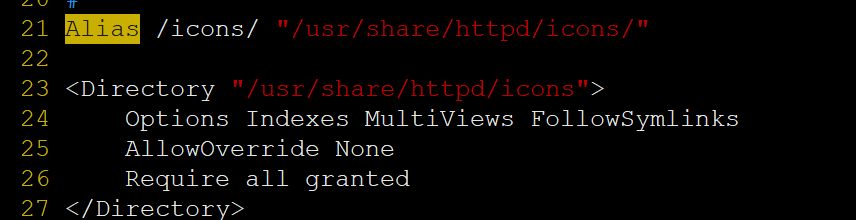
7 保存退出

8将/etc/httpd/conf.d/autoindex.conf 文件复制到当前用户的主目录

[root@rk8 ~]#cp /etc/httpd/conf.d/autoindex.conf ~/

9 打开用户主目录下刚才第8小题复制的文件autoindex.conf

将21行到27行内容复制一份到文件的第29行后面



然后保存退出文件

### 第三部份 使用配置文件

**1、创建vim配置文件，内容至少包含如下**

配置文件可以参看下面内容

这个文件的双引号 (")是注释

#vim ~/.vimrc

"vim的配置文件，这个是注释用的标记，不是#号，下面的设置是我们常用的几个设置

set ic "搜索时忽略大小写。

set cursorline "突出显示当前行

set incsearch " 输入搜索内容时就显示搜索结果

set nu "可以在每一列的最前面显示行号啦！

“下面内容不要求可试试效果

set hlsearch "搜索高亮度反白

set backspace=2 "可随时用退格键删除

set autoindent "自动缩排

set ruler "可显示最后一列的状态

set showmode "左下角那一列的状态

set bg=dark "显示不同的底色色调

syntax on "进行语法检验，颜色显示。

配置文件创建好后保存退出

配置文件效果测试

再利用vim打开前面练习的几个文件观察搜索测试行号，光标所在行下划线，搜索自动提示并且忽略大小功能是否有效

# **附：nano文本编辑器**

一、概述

Nano是一个简单易使用的文本编辑器，适用于终端环境。它具有轻量级、快速启动和基本功能的特点，非常适合新手使用。

1.1、起源

Nano最早是Pico编辑器的克隆版本，后来发展成为一个独立且功能更加强大的编辑器。它具有简洁的用户界面、直观的操作方式和丰富的快捷键，使得编辑文本变得更加高效和愉快

二、安装

一般系统都会内置nano编辑器，如果没有，可以自己安装。

2.1、Centos系统

yum install -y nano

2.2、Debian / Ubuntu 系统

apt-get install -y nano

三、基本操作

语法

nano 文件名或文件绝对路径

3.1、光标移动和选择文本

Nano中光标移动的方式与vim有所不同，可以使用方向键来进行上下左右的移动。在选择文本时，可以按住Shift键并与方向键结合使用，来选择文本。

3.2、插入和删除文本

在Nano中，直接开始输入即可在光标处插入文本。按下Backspace键或Delete键可以删除字符，相比于vim，这里没有i、a、o、A、I等模式，更为简单直接。

3.3、保存和退出文件

Nano的保存和退出命令是通过组合按键而实现的：按下Ctrl键加上字母O来保存文件，然后按下Ctrl键加上字母X来退出Nano。而vim中则是输入：wq来保存文件并退出。

3.4、搜索和替换文本

Nano中搜索文本的快捷键是Ctrl + w，而替换文本的快捷键是Ctrl + r。和vim相比，vim中搜索文本的快捷键是/，而替换文本的快捷键是:%s/old/new/g，相对来说较为复杂。

四、配置Nano

4.1、修改Nano的配置文件

要修改Nano的配置，可以编辑 ~/.nanorc 文件。这个文件中包含了各种配置选项，可以根据自己的需要进行修改。

常用的配置

set tabsize 4 # 设置制表符宽度

set autoindent # 允许自动缩进

set cut # 设置 CTRL-K 可以剪贴到行末

set noconvert # 不要转换 DOS/UNIX 换行符

set nowrap # 不要自动换行

set nohelp # 不显示下面两行帮助

set morespace # 隐藏标题下的空白行，换取更多编辑空间

set smooth # 平滑卷屏

set suspend # 允许 ctrl-z 将 nano 置于后台

set smarthome # 第一次 Home 跳到行首非空字符，第二次到行首

set tabstospaces # 展开制表符为空格（如果需要的话）

set mouse # 允许鼠标

set linenumbers # 显示行号（可以在编辑时 ALT-# 切换）

set backupdir path # 设置备份路径

set backup # 允许保存备份

set casesensitive # 搜索使用大小写敏感

set multibuffer # 使用 CTRL-r 读取文件时，默认读取到新缓存

set nonewlines # 不在文件末尾添加新行

include <filename> # 加载额外配置，通常是 /usr/share/nano 下的各种语法文件

4.2、自定义快捷键

如果你想要添加自定义的快捷键，可以在~/.nanorc文件中进行配置。例如，你可以将Ctrl+S绑定到保存文件的命令。

4.3、更改外观和配色

Nano支持不同的外观和配色方案。你可以在~/.nanorc文件中修改color选项来改变编辑器的外观。

五、高级操作

5.1、使用剪贴板和撤销

Nano中使用剪贴板的快捷键是Ctrl + Shift + 6来复制选中的文本，而撤销操作的快捷键是Ctrl + Shift + U。在vim中同样有撤销操作，快捷键是u，而使用剪贴板则需要利用系统剪贴板。

5.2、块操作和矩形剪切

在Nano中，按下Ctrl + k可以剪切整行文本，而粘贴剪切的文本到光标位置则是通过按下 Ctrl + u 实现的。而在vim中，可以使用d或y等命令进行剪切，并且可以使用p命令进行粘贴。

5.3、自动补全和拼写检查

Nano具有自动补全和拼写检查的功能，但需要进入Nano的配置文件中进行设置。而在vim中，则需要借助插件才能实现这些功能。

六、使用Nano进行编程

6.1、编辑shell脚本

Nano非常适合编辑Shell脚本。你可以使用语法高亮和自动缩进等功能来提高编程效率。和vim一样，都支持Shell脚本的编辑。

6.2、编辑Python代码

Nano也可以用于编辑Python代码。它可以帮助你发现代码中的错误，并提供了一些有用的编辑功能。和vim一样，都支持Python代码的编辑。

6.3、编辑配置文件

对于编辑各种配置文件，如Apache的配置文件等，Nano是一个非常实用的工具。它的简单性使得修改配置文件变得更加容易。和vim同样适合编辑各类配置文件。

七、常见问题

7.1、大文件编辑时的性能问题

Nano对于大型文件的编辑可能会有性能方面的限制。在处理大文件时，建议使用更专业的编辑器。而vim则以其强大的性能和处理大文件的能力著称。

7.2、如何显示行号

要在Nano中显示行号，可以在 ~/.nanorc 文件中添加 set linenumbers 选项。而在vim中，则需要输入:set number。

八、总结

Nano是一个简单易用的文本编辑器，与vim相比，它更加直观和友好。在基本操作方面，Nano使用方向键进行光标移动，按下Shift键进行文本选择，通过Backspace键或Delete键进行文本删除。保存文件和退出编辑的快捷键是Ctrl键加上相应的字母。搜索和替换文本的快捷键分别是Ctrl + W 和Ctrl + R。

在高级功能方面，Nano支持使用剪贴板进行复制和撤销操作，使用快捷键Ctrl + Shift + 6和Ctrl + Shift + u。它还具备块操作和矩形剪切的能力，可以通过Ctrl + k剪切整行文本，通过Ctrl + u粘贴剪切的文本到光标位置。另外，Nano还可以配置自动补全和拼写检查功能。

对于编程任务，Nano适用于编辑Shell脚本和Python代码，具备语法高亮、自动缩进等功能，有助于提高编程效率。此外，Nano也非常适合编辑各种配置文件，如Apache的配置文件等。

然而，Nano在处理大型文件时可能会有性能限制，因此在处理大文件时建议使用更专业的编辑器。同时，如果遇到文件保存权限问题，可以尝试使用sudo命令获取足够的权限进行保存。

综上所述，Nano是一个功能简单但实用的文本编辑器，适合进行日常的文本编辑、编程和配置文件编辑任务。它具有直观的操作方式和一些高级功能，能够满足大部分基本编辑需求。

————————————————

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

原文链接：https://blog.csdn.net/m\_j\_y0\_0/article/details/134642184