Linux vim 使用

目录

目录

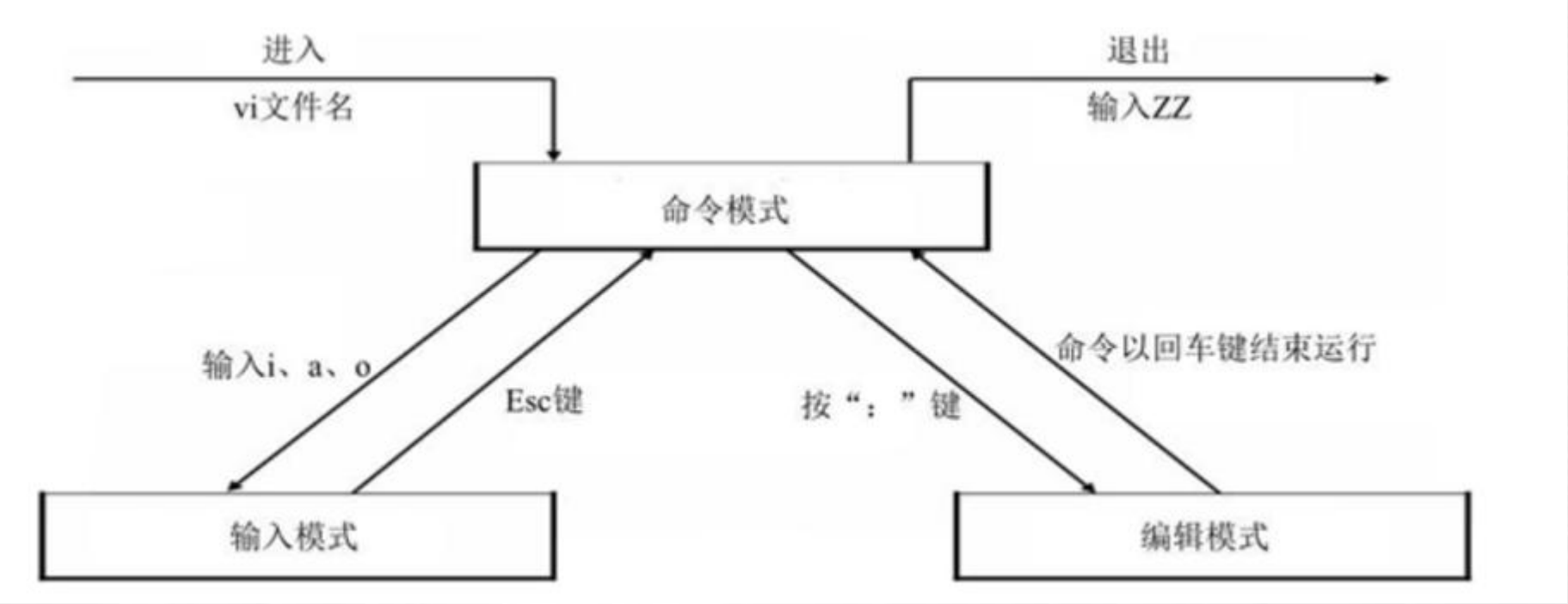
vim介绍

- linux 系统中“一切皆文件”，因此当我们在命令行下更改文件内容时，不可避免地要用到文本编辑器
- 可供选择的编辑器不止一种，例如 vim、emacs、pico、nano 等，很多人都找到了自己所喜爱的编辑器
- vim文本编辑器，是由 vi 发展演变过来的文本编辑器，因其具有使用简单、功能强大、是 linux 众多发行版的默认文本编辑器

目录

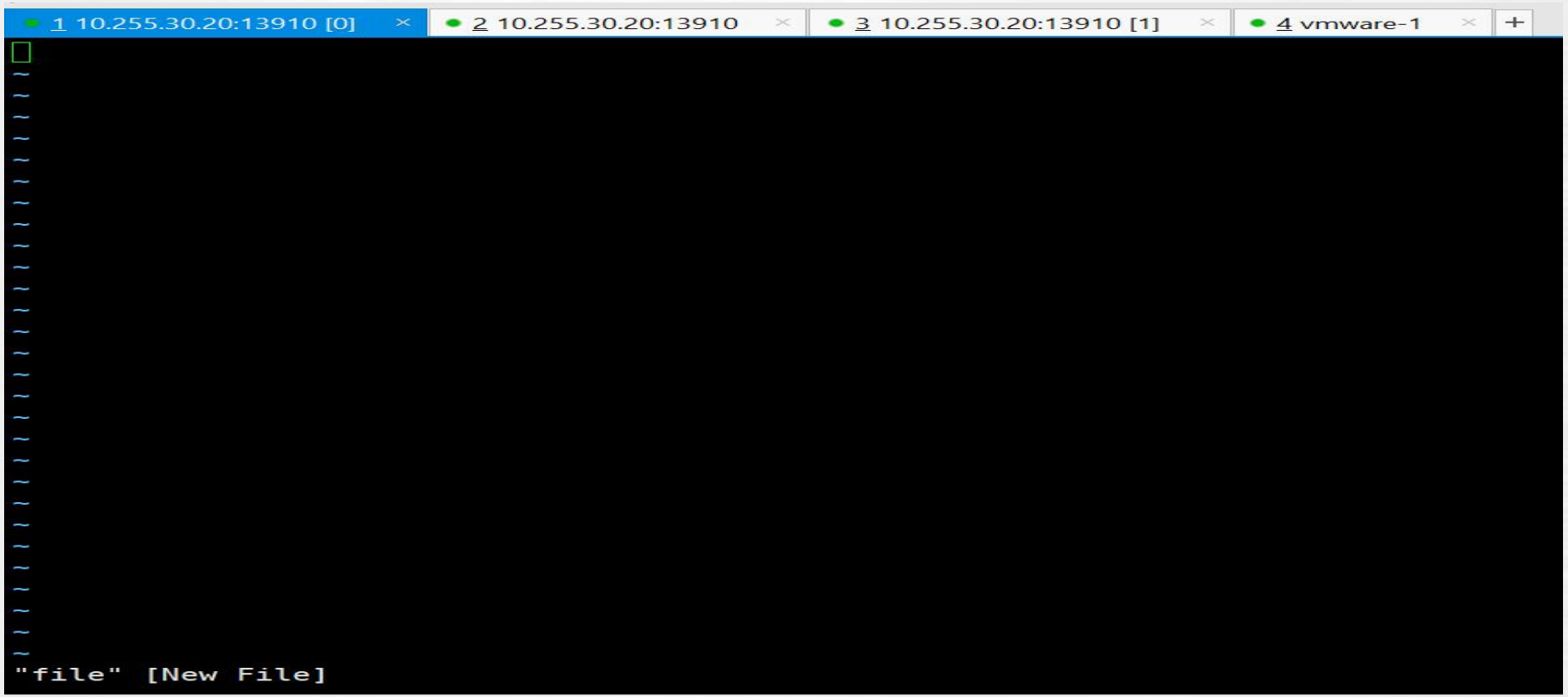
vim模式

- vim 编辑文件时，存在 3 种工作模式，分别是**命令模式、输入模式和编辑模式**，这 3 种工作模式可随意切换



vim模式-命令模式

- 使用 vim 编辑文件时，默认处于**命令模式**。此模式下，可使用方向键（上、下、左、右键）或 k、j、h、i 移动光标的位置，还可以对文件内容进行复制、粘贴、替换、删除等操作



vim模式-输入模式

- 在**输入模式**下，vim 可以对文件执行写操作，类似于在 Windows 系统的文档中输入内容
- 使 vim 进行输入模式的方式是在下输入 i、I、a、A、o、O 等插入命令，当编辑文件完成后按 Esc 键即可返回命命令模式状态令模式

快捷键	功能描述
i	在当前光标所在位置插入随后输入的文本，光标后的文本相应向右移动
I	在光标所在行的行首插入随后输入的文本，行首是该行的第一个非空白字符，相当于光标移动到行首执行 i 命令
o	在光标所在行的下面插入新的一行。光标停在空行首，等待输入文本
O	在光标所在行的上面插入新的一行。光标停在空行的行首，等待输入文本
a	在当前光标所在位置之后插入随后输入的文本
A	在光标所在行的行尾插入随后输入的文本

vim模式-输入模式

键盘输入字母i

```
411 function main(){
412     if [[ -z "$@" ]] || [[ "$@" = "--help" ]];then
413         print_help
414         exit 1
415     fi
416     get_clusops_home
417     source $CLUSOPS_HOME/etc/clusops.env
418     make_sub_dir
419     #define logger globle params
420     export LOG_OUTPUT_FILE=${CLUSOPS_HOME}/log/clusops.log
421     export LOG_OUTPUT_CONSOLE=true
422     main_init_config
423     main_parse_cmdline "$@"
424     run_module
425     if [ -f $TMP_FILE ];then
426         rm -f $TMP_FILE
427     fi
428 }
429
430 main $@
-- INSERT --
```

vim模式-编辑模式

- **编辑模式**用于对文件中的指定内容执行保存、查找或替换等操作
- 使 vim 切换到编辑模式的方法是在命令模式状态下按“：”键，此时 vim 窗口的左下方出现一个“：”符号，这是就可以输入相关指令进行操作了
- 指令执行后 vim 会自动返回命令模式。如想直接返回命令模式，按 Esc 即可

目录

vim操作-打开文本

vim <路径+filename>

例如：打开vim.test文件

```
vim /home/vim.test
```

```
11 function resume_node(){
12   $SLURM_SCONTROL $SLURM_SC_ONLINE_ARGS NodeName=$HOST
13   for i in `seq 1 3`;do
14     $SLURM_SINFO -h -n $HOST vim /home/vim.test   *' >/dev/null
15     if [ $? -eq 0 ];then
16       $SLURM_SCONTROL $SLURM_SC_ONLINE_ARGS NodeName=$HOST
17       echo "$(date +'%F %T') LOG : $PROGNAME : $FUNCNAME : Marking $HOST online ret
18       sleep 10
19     else
20       echo "$(date +'%F %T') LOG : $PROGNAME : $FUNCNAME : Marking $HOST online and
21       return 0
22     fi
23   done
24   return 1
25 }
26 resume_node
~
~
~
"test_online.sh" 26L, 768C
```

vim操作-打开文本

- 若出现下图的情况，可能是由于以下两种原因导致

① 其他程序同时打开同一个文件

② 正在编辑文件的异常退出

```
E325: ATTENTION
Found a swap file by the name "/home/.vim.test.swp"
          owned by: root   dated: Thu Feb  9 23:16:19 2023
         file name: /home/vim.test
          modified: YES
         user name: root   host name: admin01
        process ID: 19634 (still running)
While opening file "/home/vim.test"
             dated: Thu Feb  9 23:15:56 2023

(1) Another program may be editing the same file.  If this is the case,
    be careful not to end up with two different instances of the same
    file when making changes.  Quit, or continue with caution.
(2) An edit session for this file crashed.
    If this is the case, use ":recover" or "vim -r /home/vim.test"
    to recover the changes (see ":help recovery").
    If you did this already, delete the swap file "/home/.vim.test.swp"
    to avoid this message.

Swap file "/home/.vim.test.swp" already exists!
[O]pen Read-Only, (E)dit anyway, (R)ecover, (Q)uit, (A)bort:
```

- 处理后需要删除 .filename.swp 文件才会消除这个提示

vim操作-打开文件

- 除此之外，我们还可以利用下表中打开文件的命令格式，针对特定情形使用适当的打开方式，可以大大提高我们的效率

vim 使用的选项	说 明
vim filename	打开或新建一个文件，并将光标置于第一行的首部
vim -r filename	恢复上次 vim 打开时崩溃的文件
vim -R filename	把指定的文件以只读方式放入 vim 编辑器中
vim + filename	打开文件，并将光标置于最后一行的首部
vim +n filename	打开文件，并将光标置于第 n 行的首部
vim +/pattern filename	打开文件，并将光标置于第一个与 pattern 匹配的位置
vim -c command filename	在对文件进行编辑前，先执行指定的命令

vim操作-vim 查找文本

快捷键	功能描述
/abc	从光标所在位置向前查找字符串 abc
/^abc	查找以 abc 为行首的行
/abc$	查找以 abc 为行尾的行
?abc	从光标所在为主向后查找字符串 abc
n	向同一方向重复上次的查找指令
N	向相反方向重复上次的查找指定

vim操作-vim 查找文本

- 从行开头查询字符串

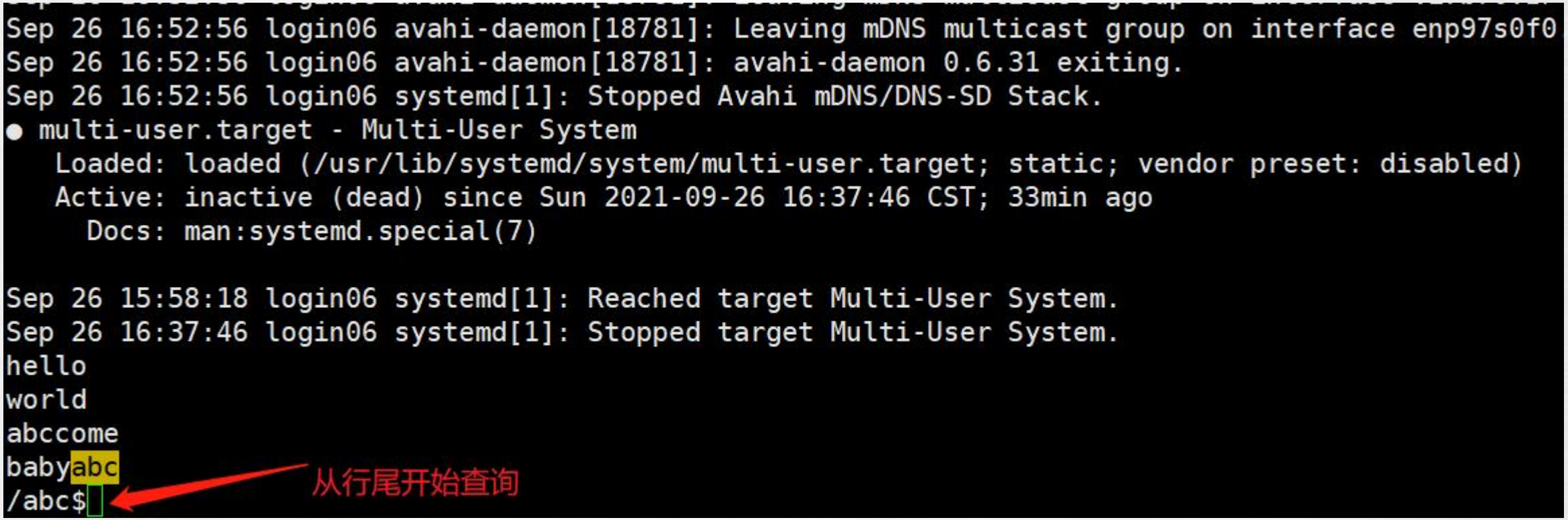
```
/^abc
```

```
Sep 26 16:52:56 login06 avahi-daemon[18781]: Leaving mDNS multicast group on interface virbr0
Sep 26 16:52:56 login06 avahi-daemon[18781]: Leaving mDNS multicast group on interface enp97s
Sep 26 16:52:56 login06 avahi-daemon[18781]: avahi-daemon 0.6.31 exiting.
Sep 26 16:52:56 login06 systemd[1]: Stopped Avahi mDNS/DNS-SD Stack.
● multi-user.target - Multi-User System
   Loaded: loaded (/usr/lib/systemd/system/multi-user.target; static; vendor preset: disabled
   Active: inactive (dead) since Sun 2021-09-26 16:37:46 CST; 33min ago
     Docs: man:systemd.special(7)

Sep 26 15:58:18 login06 systemd[1]: Reached target Multi-User System.
Sep 26 16:37:46 login06 systemd[1]: Stopped target Multi-User System.
hello
world
abccome
babyabc
search hit BOTTOM, continuing at TOP
```

vim操作-vim 查找文本

- 从行开头查询字符串

```
/abc$
```



vim操作-查找文本

- 查找的字符串是严格区分大小写的
- 如果想忽略大小写，则输入命令 ":set ic"；调整回来输入":set noic"

忽略大小写
:set ic
大小写区分
:set noic

vim操作-查找文本

- 特殊字符，则需要加上转义字符 “\” 。常见的特殊字符有 \、*、?、$ 等

比如查找字符 "$"

```
/\$
```

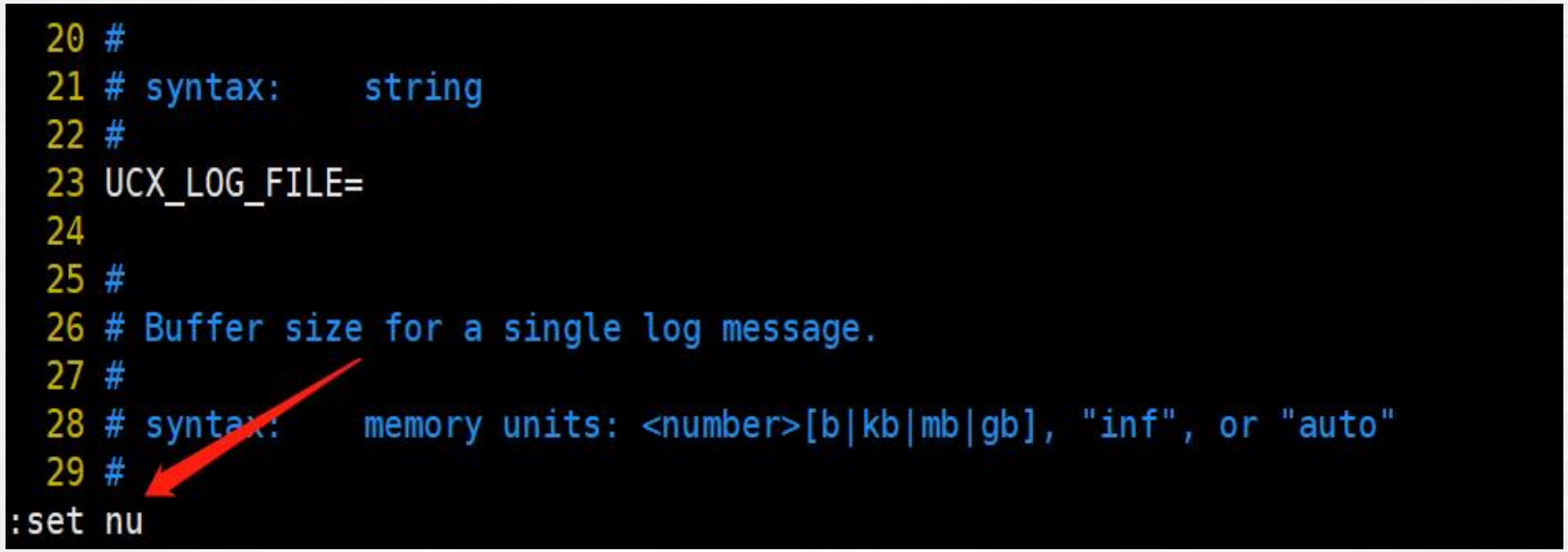
```
Sep 26 16:52:56 login06 avahi-daemon[18781]: Leaving mDNS multicast group on interface enp97s0f0.I
Sep 26 16:52:56 login06 avahi-daemon[18781]: avahi-daemon 0.6.31 exiting.
Sep 26 16:52:56 login06 systemd[1]: Stopped Avahi mDNS/DNS-SD Stack.
● multi-user.target - Multi-User System
   Loaded: loaded (/usr/lib/systemd/system/multi-user.target; static; vendor preset: disabled)
   Active: inactive (dead) since Sun 2021-09-26 16:37:46 CST; 33min ago
     Docs: man:systemd.special(7)

Sep 26 15:58:18 login06 systemd[1]: Reached target Multi-User System.
Sep 26 16:37:46 login06 systemd[1]: Stopped target Multi-User System.
hello
world
abccome
babyabc\$
/\$
```

vim操作-设置行号

- 临时显示行号

退出vim后再次打开vim就不显示行号了

```
:set nu
```



vim操作-设置行号

- 永久显示行号

修改vim配置文件vimrc，如果没有此文件可以创建一个

```
vim ~/.vimrc
```

```
56 if &term=="xterm"
57       set t_Co=8
58       set t_Sb=^[[4%dm
59       set t_Sf=^[[3%dm
60 endif
61
62 " Don't wake up system with blinking cursor:
63 " http://www.linuxpowertop.org/known.php
64 let &guicursor = &guicursor . ",a:blinkon0"
65 set nu
~/.vimrc" 65L, 1989C
```

vim操作-替换文本

快捷键	功能描述
r	替换光标所在位置的字符
R	从光标所在位置开始替换字符，其输入内容会覆盖掉后面等长的文本内容，按“Esc”可以结束
:s/a1/a2/g	将当前光标所在行中的所有 a1 用 a2 替换
:n1,n2s/a1/a2/g	将文件中 n1 到 n2 行中所有 a1 都用 a2 替换
:g/a1/a2/g	将文件中所有的 a1 都用 a2 替换

vim操作-替换文本

- 要将某文件中所有的 “admin" 替换为 “root"，则有两种输入命令

```
:1,$s/admin/root/g
```

vim操作-替换文本

:%s/admin/root/g

```
  1 hello world
  2 sjdofsddsdaby
  3 sugon
  4 hello10$admin
  5 china<admin>
  6 beijing
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:%s/root/admin/g
```

vim操作-删除文本

- 被删除的内容并没有真正删除，都放在了剪贴板中。
- 将光标移动到指定位置处，按下 "p" 键，就可以将刚才删除的内容又粘贴到此处

快捷键	功能描述
x	删除光标所在位置的字符
dd	删除光标所在行
ndd	删除当前行（包括此行）后 n 行文本
dG	删除光标所在行一直到文件末尾的所有内容
D	删除光标位置到行尾的内容
:a1,a2d	函数从 a1 行到 a2 行的文本内容

vim操作-删除文本

- 删除光标所在的字符

按键x

```
● multi-user.target - Multi-User System
   Loaded: loaded (/usr/lib/systemd/system/multi-user.target; static; vendor preset: disabled)
   Active: inactive (dead) since Sun 2021-09-26 16:37:46 CST; 33min ago
     Docs: man:systemd.special(7)

Sep 26 15:58:18 login06 systemd[1]: Reached target Multi-User System.
Sep 26 16:37:46 login06 systemd[1]: Stopped target Multi-User System.
hello
world
abccome
babybc
```

源文本

```
Sep 26 15:58:18 login06 systemd[1]: Reached target Multi-User System.
Sep 26 16:37:46 login06 systemd[1]: Stopped target Multi-User System.
hello
world
abccome
babyc
```

删除字符

vim操作-复制和粘贴文本

快捷键	功能描述
p	将剪贴板中的内容粘贴到光标后
P（大写）	将剪贴板中的内容粘贴到光标前
y	复制已选中的文本到剪贴板
yy	将光标所在行复制到剪贴板，此命令前可以加数字 n，可复制多行
yw	将光标位置的单词复制到剪贴板

vim操作-保存退出文本

- “w!” 和 “wq!” 等类似的指令，通常用于对文件没有写权限的时候（显示 readonly，修改失败），但如果你是文件的所有者或者 root 用户，就可以强制执行

命令	功能描述
:wq	保存并退出 vim 编辑器
:wq!	保存并强制退出 vim 编辑器
:q	不保存就退出 vim 编辑器
:q!	不保存，且强制退出 vim 编辑器
:w	保存但是不退出 vim 编辑器
:w!	强制保存文本
:w filename	另存到 filename 文件
x!	保存文本，并退出 vim 编辑器，更通用的一个 vim 命令
ZZ	直接退出 vim 编辑器

目录

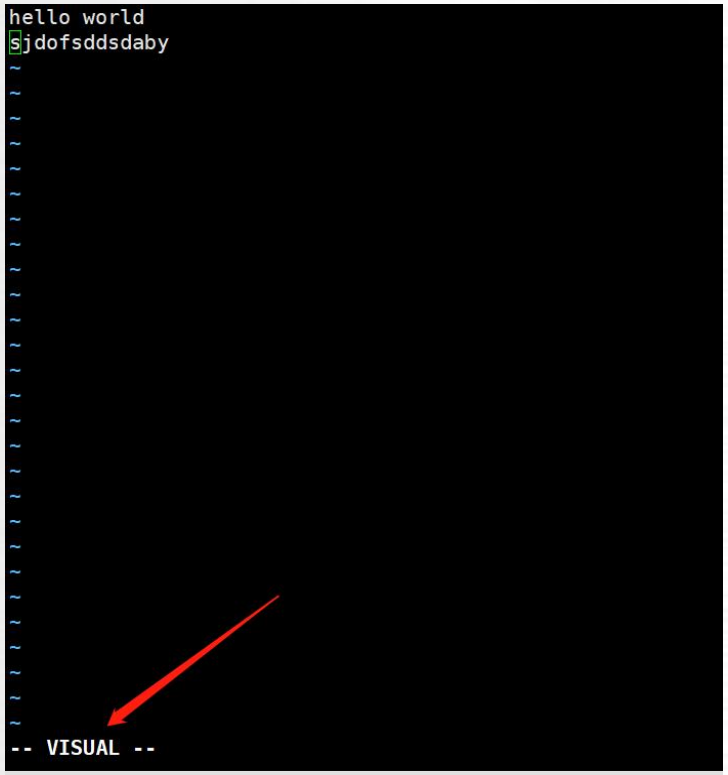
vim可视化模式及其用法

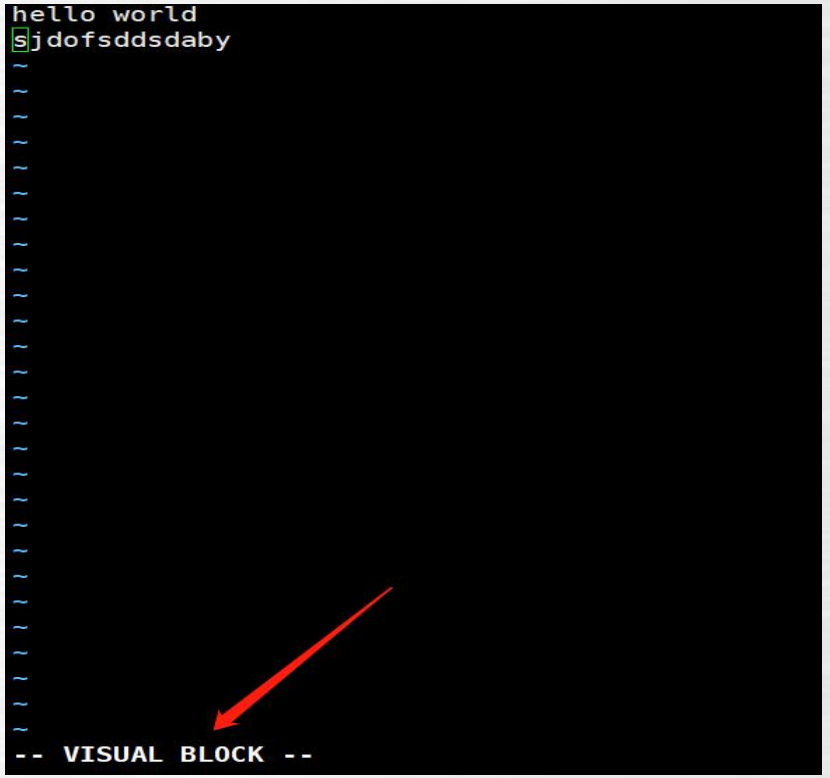
- 在 vim 中，如果想选中目标文本，就需要调整 vim 进入可视化模式，通过在 vim 命令模式下键入不同的键，可以进入不同的可视化模式

命令	功能
v（小写）	又称字符可视化模式，此模式下目标文本的选择是以字符为单位的，也就是说，该模式下要一个字符一个字符的选中要操作的文本。
V（大写）	又称行可视化模式，此模式化目标文本的选择是以行为单位的，也就是说，该模式化可以一行一行的选中要操作的文本。
Ctrl+v（组合键）	又称块可视化模式，该模式下可以选中文本中的一个矩形区域作为目标文本，以按下 Ctrl+v 位置作为矩形的一角，光标移动的终点位置作为它的对角。

vim可视化模式

- 以字符可视化模式为例，vim 成功进入该模式的标志是窗口底部出现 --VISUAL--
- 进入行可视化模式的标志是图 2 标记处显示 --VISUAL LINE--
- 进入块可视化模式的标志是图 2 标记处显示 --VISUAL BLOCK--





vim可视化用法

- 当选中文本并做完相应操作（例如选中文件并按 p 键将其复制到剪贴板中）后，vim 会自动从可视化模式转换为命令模式。当然，也可以再次按 v（或者 V、Ctrl+v）手动退出可视化模式。
- 之前所学的在 vim 命令模式下编辑文本的很多命令，在可视化模式下仍然可以使用。
- 下图列出了常用的几个可以在可视化模式下使用的命令。

命令	功能
d	删除选中的部分文本。
D	删除选中部分所在的行，和 d 不同之处在于，即使选中文本中有些字符所在的行没有都选中，删除时也会一并删除。
y	将选中部分复制到剪贴板中。
p（小写）	将剪贴板中的内容粘贴到光标之后。
P（大写）	将剪贴板中的内容粘贴到光标之前。
u（小写）	将选中部分中的大写字符全部改为小写字符。
U（大写）	将选中部分中的小写字符全部改为大写字符。
>	将选中部分右移（缩进）一个 tab 键规定的长度（CentOS 6.x 中，一个tab键默认相当于 8 个空白字符的长度）。
<	将选中部分左移一个 tab 键规定的长度

谢 谢