Linux shell命令初学解析

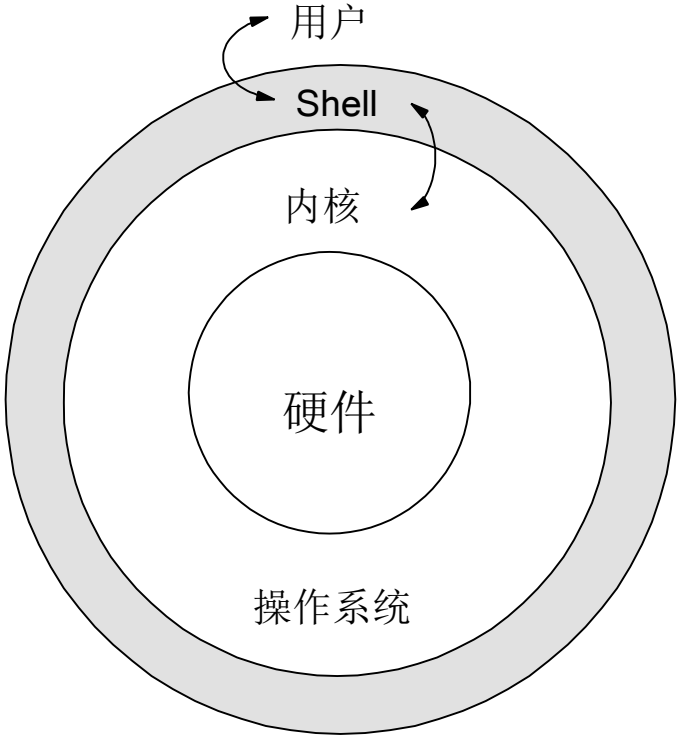
目标

- 了解shell
- 掌握shell的常用命令
- 掌握vi编辑器的用法

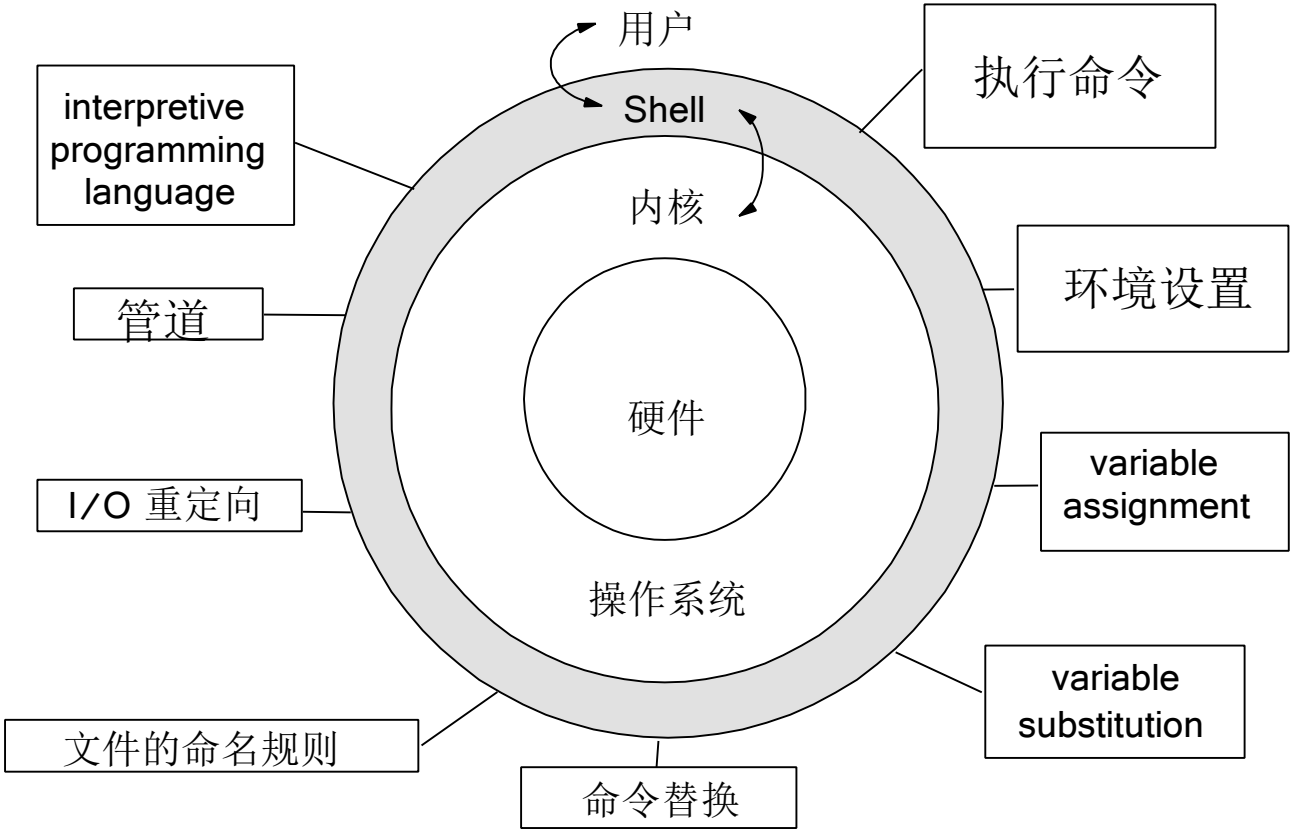
一、了解shell

掌握Shell

- shell是用户和Linux操作系统之间的接口
- 在Linux中几乎所有的操作都可以通过命令行完成
- shell是一个命令语言解释器，它拥有自己内建的shell命令集，可以使用该命令集编写程序，shell也能被系统中其它应用程序所调用。
- 使用shell编写的程序称为shell脚本



什么是Shell


用户
Shell
内核
硬件
操作系统
interpretive programming language
管道
I/O 重定向
文件的命名规则
命令替换
执行命令
环境设置
variable assignment
variable substitution

shell的种类

Linux中的shell有多种类型，最常用的是：

- Bourne shell（sh）
- C shell（csh）
- Korn shell（ksh）
- Bourne Again shell（bash）

- Linux操作系统缺省的shell是Bourne Again shell。它是Bourne shell的扩展，与Bourne shell完全兼容。
- 查询当前用户shell:

```
#echo $SHELL
```

Shell 程序登录

当普通用户成功登录，系统将执行一个shell程序。shell进程提供命令行提示符。作为默认值，对普通用户用“$”作提示符，对超级用户（root）用“#”作提示符。

$	普通用户
#	超级用户

Shell提示符的定制

Bash的提示符是利用环境变量PS1和PS2定制的。

格式：#PS1=新提示符

[root@www /]#PS1=RHEL4： //修改当前提示符为“RHEL4:”

RHEL4：

返回常规的提示符：

RHEL4：PS1=“[\u@\h\w]\$”

或 RHEL4: PS1=“[root@host ~]# ”

bash配置文件

bash 在用户登录时从四个文件中读取环境设定：

全局设置文件：

① /etc/profile

② /etc/bashrc

用户设置文件：

③ ~/.bash_profile

④ ~/.bashrc\

其中在每个用户的宿主目录下：

.bash_history:记录用户以前输入的命令。

.bash_logout:用户退出shell时要执行的命令。

.bash_profile:用户登入shell时要执行的命令。

.bashrc:每次打开新的shell时要执行的命令。

$PATH环境变量

#set　　//查看全部环境变量

#echo $ PATH　　//查看某一环境变量

#export PATH=$PATH:/some/directory //改变某一环境变量值

#echo $PATH　　//查看更改后环境变量值

虚拟控制台

文本界面下：

Alt+F1~F6切换到字符虚拟控制台， Alt+F7切换到图形界面。

图形界面下：

Alt+Ctrl+F1~F6切换到字符虚拟控制台， Alt+F7切换到图形界面。

系统运行级别

Linux系统运行级别由列在/etc/rc.d/rc<x>.d目录中的服务来定义，其中<x>是运行级别的数字：

0：终止所有进程，关机。
1：单用户模式，用于维护系统，只有少数进程运行。
2：多用户模式，和运行级别3一样（除没有启动NFS服务）。
3：完整的多用户模式，进入Linux系统的文本字符界面。
4：没有使用（可由用户定义）。
5：完整的多用户模式，进入Linux系统的基于X的图形界面。
6：重新启动。

系统运行级别

修改系统运行级别：

#vi /etc/inittab

id:5:initdefault:　//把5修改为想要的运行级别。

或者

#telinit n　//n为0~6，只有root用户才能使用此命令。

系统运行级别

查看系统运行级别：

#runlevel

特殊字符及引号

符号	含义	符号	含义
~	用户宿主目录	|	管道
`	命令取代	?	通配符，代表任何单一字符
#	解释	<	输入重定向
$	变量取值	>	输出重定向
&	后台进程工作	‘	单引号(没有命令置换功能)
*	通配符，代表任何多个字符	“	双引号(具有命令置换功能)
(	子shell开始	/	路径分割符
)	子shell结束	;	命令分割符
\	使命令持续到下一行		

常用控制组合键

组合键	含义
Ctrl+C	中止对话进程
Ctrl+\	中止对话进程
Ctrl+D	结束输入，即EOF，或注销Linux。
Ctrl+Z	暂停/挂起对话进程
Ctrl+M	相当于按“Enter”键
Ctrl+S	暂停/冻结屏幕输出
Ctrl+Q	恢复屏幕输出
Ctrl+U	将命令整列删除

二、掌握shell的常用命令

Linux的常用命令

命令行

用户登录时，实际进入了shell，它遵循一定的语法将输入的命令加以解释并传给系统。命令行中输入的第一个字必须是一个命令的名字，第二个字是命令的选项或参数，命令行中的每个字必须由空格或TAB隔开，格式如下：

$ Command Option Arguments

选项和参数（续）

- 选项可用于改变命令执行的动作的类型，包括一个或多个字母的代码，多个选项可用一个“-” 连起来，如 ls -l -a 与 ls -la 相同。
- 参数是在命令行中的选项之后键入的一个或多个单词

```
$ ls -l /test
-rw-r--r-- 2 wzh book 22 Apr 20 20:37 motd
-rw-r--r-- 2 wzh book 796 Apr 20 20:37 passwd
$
```

命令行特征

- 命令行实际上是可以编辑的一个文本缓冲区
- bash保存着命令历史表
- 在一个命令行中还可以置入多个命令，用分号将各个命令隔开，如：

```
$ ls -l;cp -i mydata newdata
```

- 也可以在几个命令行中输入一个命令，用反斜杠将一个命令行持续到下一行，如：

```
$ cp -i \
>mydata  \
>newdata
```

命令行特征

- 只有当command1正确运行完毕后,才执行command2:

```
# command1 && command2
```

- Command1和command2只要执行其中一个:

```
# command1 || command2
```

shell中的特殊字符

- 通配符
  - ?
  - *
  - []

模式串	意 义
*	当前目录下所有文件的名称。
Text	当前目录下所有文件名中包含有Text的文件的名称。
[ab-dm]*	当前目录下所有以a、b、c、d、m开头的文件的名称。
[!ab-dm]?	当前目录下不以a、b、c、d、m开头且后面只跟有一个字符的文件的名称。
/usr/bin/??	目录/usr/bin下所有名称为两个字符的文件的名称。

shell中的特殊字符(续)

- 引号

  1）单引号 '

  由单引号括起来的字符都作为普通字符出现。特殊字符用单引号括起来以后，也会失去原有意义，而只作为普通字符解释。例如：

```
# pwd
/home/xyz
# string=‘current directory is `pwd`’
# echo $string
current directouy is `pwd`
#
```

shell中的特殊字符(续)

2）双引号 "

由双引号括起来的字符，除$、＆这几个字符仍是特殊字符并保留其特殊功能外，其余字符仍作为普通字符对待。对于$来说，就是用其后指定的变量的值来代替这个变量和$。

```
#pwd
/home/xyz
# string="current directory is `pwd`"
# echo $string
current directory is /home/xyz
#
```

shell中的特殊字符(续)

3）反引号 `

反引号括起来的字符串被shell解释为命令行，在执行时，shell首先执行该命令行，并以它的标准输出结果取代整个反引号（包括两个反引号）部分。例如：

```
# pwd
/home/xyz
# string="current directory is `pwd`"
#echo $string
current directour is /home/xyz
#
```

shell执行**echo**命令时，首先执行**`pwd`**中的命令**pwd**，并将输出结果**/home/xyz**取代**`pwd`**这部分，最后输出替换后的整个结果

信息交流

1）echo 命令

功能：

- 在显示器上显示一段文字，一般起到一个提示的作用。

语法：

echo ［ -n ］ 字符串

选项：

- n表示输出文字后不换行；字符串可以加引号，也可以不加引号。用echo命令输出加引号的字符串时，将字符串原样输出；用echo命令输出不加引号的字符串时，将字符串中的各个单词作为字符串输出，各字符串之间用一个空格分割。
- echo命令也可以用来显示变量指定的值，如：

```
#echo  hello linux!!!!!!
# echo $PATH
```

信息交流

2）write命令

功能：

- 向系统中某一个用户发送信息。

语法：

write 用户帐号 ［终端名称］

- 例如：

```
# write user1  tty2
```

- 此时系统进入发送信息状态，用户可以输入要发送的信息，输入完毕，希望退出发送状态时，按组合键< Ctrl+c>即可。

信息交流

3） mesg命令

功能：

- 设定是否允许其他用户用write命令给自己发送信息。

语法：

mesg 选项

说明：

- 如果允许别人给自己发送信息，输入命令：

```
# mesg  y
```

- 否则，输入：

```
# mesg n
```

- 对于超级用户，系统的默认值为 n；而对于一般用户系统的默认值为y。如果mesg后不带任何参数，则显示当前的状态是y还是n。

信息交流

4）wall命令

功能：

- 对全部已登录的用户发送信息。

语法：

wall [消息内容]

```
# wall ‘Please shutdown your computer after worktime!!!'
Broadcast message from root （tty1） Fri Nov 26 14：15：07 2006…
‘Please shutdown your computer after worktime!!!
```

- 执行以上命令后，用户的屏幕上显示出“‘Please shutdown your computer after worktime!!!' ”信息后，并不出现系统提示符$（#），再次按回车键后，屏幕出现系统提示符。

信息交流

4）talk命令

功能：

- 发送信息给其他使用者。

语法：

talk [用户] [终端名称]

```
# talk user2    //等待user2输入：talk user1 开始对话。

#talk   user2@abc.com  pts/2
                    //等待user2输入：talk user1@xyz.com 开始对话，按
    Ctrl+c结束对话。
```

系统信息显示

1）uname命令

功能：

- 显示计算机以及操作系统的相关信息。

语法：

uname ［-amnrsv］

选项：

- - a：显示全部信息。
- -m：显示计算机类型。
- -n：显示在网络上的主机名称。
- -r：显示操作系统的发行编号。
- -s：显示操作系统的名称。
- -v：显示操作系统的版本。

```
#uname –r
```

系统信息显示

2）hostname命令

功能：

显示或修改计算机名称。

语法：

hostname ［新计算机名］

```
#hostname
#hostname   myhost
```

系统信息显示

3）dmesg命令

功能：

显示开机信息。

语法：

dmesg［-cn］ ［-s <缓冲区大小>］

选项：

- c：显示信息后，清除缓冲区中内容。

-s：预设为8139，刚好等于缓冲区大小。

-n：设置记录信息的等级。

```
#dmesg
```

系统信息显示

4）**cal**命令

功能：

显示某年某月的日历。

语法：

cal ［选项］ ［月 ［年］］

选项：

- j：显示出给定月中的每一天是一年中的第几天（从1月1日算起）。
- y：显示出整年的日历。

-m：以星期一为每周的第一天的方式显示。

```
#cal
#cal 11 2007
#cal 2007
#cal –j 11 2007
```

系统信息显示

5）date命令

功能：

显示和设置系统日期和时间。

语法：

date ［选项］ 显示时间格式（以+开头，后面接格式）

说明：

只有超级用户才有权限使用date命令设置时间，一般用户只能使用date命令显示时间。

选项：

-d datestr, --date datestr ：显示由datestr描述的日期 。

-s datestr, --set datestr ：设置datestr 描述的日期。

-u, --universal ：显示或设置通用时间。

系统信息显示

5）date命令(续)

时间域

% H小时（00..23）

% I小时（01..12）

% k小时（0..23）

% l小时（1..12）

% M 分（00..59）

% p 显示出AM或PM

% r 时间（hh：mm：ss AM或PM），12小时

% s 从1970年1月1日00：00：00到目前经历的秒数

% S 秒（00..59）

% T 时间（24小时制）（hh:mm:ss）

% X 显示时间的格式（％H:％M:％S）

% Z 时区 日期域

系统信息显示

5）date命令(续)

% a 星期几的简称（ Sun..Sat）
% A 星期几的全称（ Sunday..Saturday）
% b 月的简称（Jan..Dec）
% B 月的全称（January..December）
% c 日期和时间（ Mon Nov 8 14：12：46 CST 1999）
% d 一个月的第几天（01..31）
% D 日期（mm / dd / yy）
% h 和%b选项相同
% j 一年的第几天（001..366）
% m 月（01..12）
% w 一个星期的第几天（0代表星期天）
% W 一年的第几个星期（00..53，星期一为第一天）
% x 显示日期的格式（mm/dd/yy）
% y 年的最后两个数字（ 1999则是99）
% Y 年（例如：1970，1996等）

系统信息显示

5）date命令(续)

```
#date
#date   0217142598
#date  +"%r %a %d %h %y"
#date  -s 09:16:00
#date  -s 040414
#date  'The date of today is :%x,it is %X'
```

其他命令

1）clear命令

功能：

清除屏幕上的信息，清屏后，提示符移动到屏幕左上角。

语法：

```
# clear
```

其他命令

2）sync命令

功能：

更新i-node表，并将缓冲文件写到硬盘中。

语法：

sync

```
#sync
```

其他命令

3）uptime命令

功能：

显示系统已经运行了多长时间，显示内容：现在时间、系统已运行了多长时间、目前已登录用户，以及系统在过去的1min、5min和10min内的平均负载。

语法：

uptime ［-V］

选项：

-V：显示版本信息。

```
#uptime
```

其他命令

4）last命令

功能：

显示近期用户或者终端的登录情况。

语法：

last ［选项］

选项：

-n：指定输出记录的条数。

-t：只显示指定虚拟控制台上的登录情况。

-h：只显示指定节点上的登录情况。

-I：只显示指定IP上的登录情况。

-I：用IP来显示远端地址。

-y：显示记录的年、月、日。

-ID：要查询的用户名。

-x：显示系统关闭、用户登录和退出历史。

```
#last root console
#last reboot
#last –n  10
```

其他命令（续）

7） 标准输入与输出

执行一个shell命令行时通常会自动打开三个标准文件:

标准输入文件（stdin），通常对应终端的键盘;

标准输出文件（stdout），对应终端的屏幕

标准错误输出文件（stderr），对应终端的屏幕。

进程将从标准输入文件中得到输入数据，将正常输出数据输出到标准输出文件，而将错误信息送到标准错误文件中。

其他命令（续）

7） 标准输入与输出(续)

以cat命令为例，cat命令的功能是从命令行给出的文件中读取数据，并将这些数据直接送到标准输出。若使用如下命令：

```
$ cat config
```

将会把文件config的内容依次显示到屏幕上。但是，如果cat的命令行中没有参数，它就会从标准输入中读取数据，并将其送到标准输出。例如：

```
$ cat
Hello world
Hello world
Bye
Bye
<ctrl+d>
$
```

其他命令（续）

8）输入重定向

输入重定向是指把命令（或可执行程序）的标准输入重定向到指定的文件中。也就是说，输入可以不来自键盘，而来自一个指定的文件。所以说，输入重定向主要用于改变一个命令的输入源，特别是改变那些需要大量输入的输入源。

```
$ cat < ./list.txt
```

其他命令（续）

9）输出重定向

指把标准输出或标准错误输出重新定向到指定文件中， 而不显示在屏幕上。

- 〉：覆盖输出重定向，文件不存在则建立，存在就覆盖
- 〉〉：追加输出重定向，文件不存在则建立，文件存在则追加到末尾

```
$ls > directory.out
$cat directory.out
bin
boot
dev
$
```

其他命令（续）

9）错误重定向

把标准错误输出重新定向到指定文件中， 而不显示在屏幕上。

```
$cat  /root/kk 2>/root/b
$cat  /root/b

#cat /nn  &>/root/oo    //同时使用输出和错误重定向。
```

其他命令（续）

10）管道

管道可以把一系列命令连接起来，这意味着第一个命令的输出会作为第二个命令的输入通过管道传给第二个命令，第二个命令的输出又会作为第三个命令的输入，以此类推。显示在屏幕上的是管道中最后一个命令的输出（如果命令行中未使用输出重定向）。通过使用管道符“|”来建立一个管道行。

```
#cat note.txt  |grep "High"|wc –l
```

其他命令（续）

10）管道

管道将cat命令（列出一个文件的内容）的输出送给grep命令。grep命令在输入里查找单词High，grep命令的输出则是所有包含单词High的行，这个输出又被送给wc命令，wc命令统计出输入中的行数。假设sample.txt文件的内容如下：

```
Things to do today:
Low:Go grocery shopping
High:Return movie
High:Clear level 3 in Alien vs. Predator
Medium:Pick up clothes from dry cleaner
```

那么该管道行的结果是2。

其他命令（续）

11）命令替换

和重定向相似，区别在于命令替换是将一个命令的输出作为另外一个命令的参数。常用命令格式为：

command1 `command2`

- 其中，command2的输出将作为command1的参数。需要注意的是这里的`符号，被它括起来的内容将作为命令执行，执行后的结果作为command1的参数。例如：

```
$ cd `pwd`
```

- 该命令将pwd命令列出的目录作为cd命令的参数，结果仍然是停留在当前目录下。

其他命令（续）

11）命令替换

格式：命令1　$（命令2）
例如：

```
#less  $(locate READE.mouse)
```

其中，　命令“locate READE.mouse”的输出“=/usr/X11R6/lib/X11/doc/READE.mouse”作为less的参数。

其他命令（续）

12）命令别名

格式：#alias 别名=“命令”

功能:可以定制命令。

例：

```
#alias  psa=“ps  -aux|more”
```

其他命令（续）

13 ）命令补全

```
# ifco  TAB
# if  TAB TAB
```

在送入命令的任何时刻，可以按<Tab>键，当这样做时，系统将试图补全此时已输入的命令。如果已经输入的字符串不足以唯一地确定它应该使用的命令，系统将发出警告声。再次按<Tab>键，系统则会给出可用来用来补全的字符串清单。

其他命令（续）

14）查看历史命令
格式:history [n]

```
$ history  5
```

- 列出用户在命令模式下执行的最后（1000）个命令
- 执行历史命令:!命令编号

其他命令（续）

15）linux文本模式下，乱码问题

#locale

#export LC_ALL=en_US

#export LC_CTYPE=zh_CN.GBK

SSH、Telnet远程登录汉字出现乱码问题：

#vi /etc/sysconfig/i18n

将“LANG=zh_CN.UTF-8”该为“LANG=zh_CN.GB18030”即可。

其他命令（续）

16）用户环境变量查看

```
$ set
```

- 输出环境变量的内容：

```
$ echo  $PATH
$ echo  $HOME
```

其他命令（续）

17）查看系统内核版本命令

```
# uname -a
# more /proc/version
```

其他命令（续）

18）查看系统发行版本命令

#more /etc/issue

其他命令（续）

19）查看CPU信息命令

```
# more /proc/cpuinfo
```

其他命令（续）

20）查看内存信息命令

```
# more /proc/meminfo
```

其他命令（续）

21）测定执行特定的命令所用的时间等信息

格式：#time 〈command〉

实例：

#time ls

Linux中的帮助命令

1） help命令

- 功能：获取相关命令的帮助信息
- 用法： 命令 --help
- 例： mkdir --help

Linux中的帮助命令

2） man命令

- 功能：获取相关命令的帮助信息
- 用法： man 命令
- 例：man umask
- -s:显示被查询命令的简短语法描述。
- -f:只显示命令的功能而不显示其中详细的说明文件。
- 显示结果中，输入：/单词，可以找到需要的内容。

Linux中的帮助命令

3）info命令

功能：获取相关命令的详细使用方法

用法：info　命令

注：退出 **man** 或 **info** 按 **q** 即可

Linux中的帮助命令

4）whatis命令

功能：解释命令的功能

格式： whatis 命令

```
# whatis mkdir
```

Linux中的帮助命令

5）whereis命令

功能：查询命令的位置

格式： whereis 命令

```
# whereis mkdir
```

Linux中的帮助命令

6）which命令

功能：显示可执行命令的路径和别名。

格式： which 命令

```
# which  ls
```

三、掌握vi编辑器的用法

vi 编辑器的使用

vi 简介

- vi（发音为“vee-eye”，可视显示编辑程序的缩写）是标准的Linux文本编辑程序。由于vi不是基于窗口的，所以这个多用途编辑程序可以用在任何类型的终端上编辑各式各样的文件。
- 可以通过vi来输入和编辑文本，但它却不是一个字处理程序。与商业化字处理程序不同，它不是用来处理格式化文本的。
- vi包含一大批命令，其中许多功能互相重叠。

创建一个文件

启动vi并编辑文件hello.txt，如所示：

```
# vi    hello.txt
```

1）如果 文件已经存在，vi会打开现存文件；如果是一个新文件，vi会创建它。

2）光标出现在屏幕的左上角。空白行用垂直的一列否定号（~）来表示。最下面为状态行。注意，也可以通过只输入vi，不指定一个文件名来启动vi。可以在退出vi时命名这个文件。屏幕的最后一行被称为状态行，用于显示文件名及文件中行和字符的个数。

3）结束一个会话过程

- 保存

  :w后面接回车键

- 保存并退出

  输入:wq后面接回车键

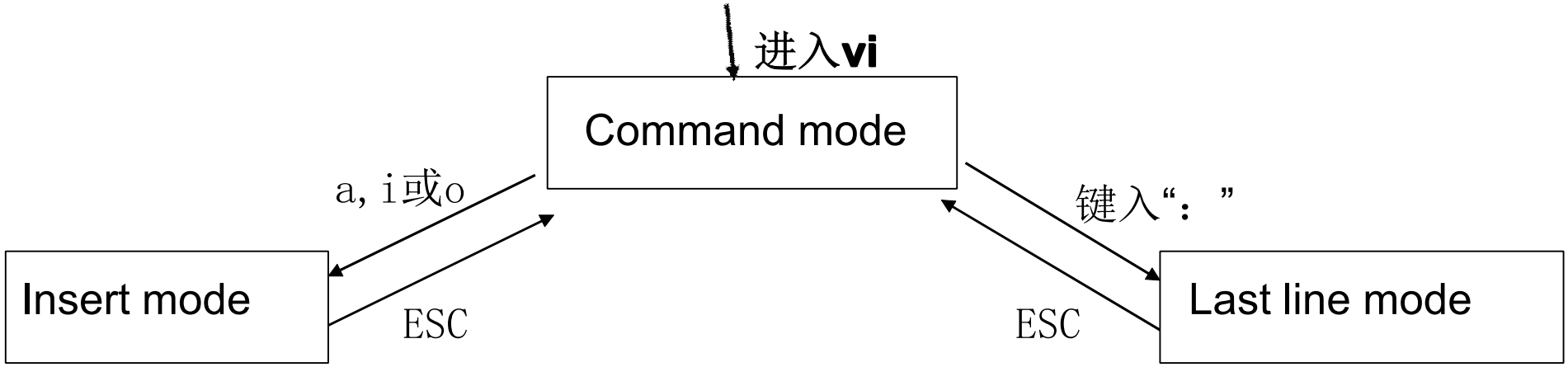
- 退出但不保存

  输入:q后面接回车键

vi 的三种模式

vi有三种操作模式：输入模式、命令模式和末行模式。输入模式用于在一个文件中输入文本，命令模式用于输入那些执行特定vi功能的命令，而末行模式用于执行保存、搜索等操作。

- 命令模式（Command mode）：删除字符、排版
- 插入模式（Insert mode）：插入字符、删除、修改字符
- 末行模式（Last line mode）：通过命令操作vi软件



命令模式

- 在该模式中，可以输入命令来执行多种功能
- 大多数的vi命令都是由一个或两个字母加上一个可选数字组成
- 以冒号开头的命令实际上是ex 命令，即进入末行模式进行保存、退出等操作

命令模式（续）
—在文件中移动光标

- 向左移动一个字符：按h
- 向右移动一个字符：按l
- 向下移动一行：按j
- 向上移动一行：按k
- 移动到当前行的开头处：按 ^ （即Shift－6）
- 移动到当前行的结尾处：按 $ （即Shift－4）
- 移动到顶部：按H（大写）使光标移动到屏幕的顶部
- 移动到中部：按M（大写）使光标移动到屏幕的中间
- 移动到底部：按L（大写）使光标移动到屏幕的底部

命令模式（续）
—在文件中移动光标（续）

- 向前翻页一屏：要向前滚动（向下移动）一整屏，需按下Ctrl-f。光标将移动到新屏的左上角
- 向前滚动半屏：要向前滚动半屏，需按Ctrl-d
- 向后翻页一屏：要向后滚动（即向上移动）一整屏，需按下Ctrl-b
- 向后滚动半屏：要向后滚动半屏，需按下Ctrl-u

命令模式（续）
——删除文本

- 删除一个字符：
  - 为删除一个字符，需将光标放置在要删除的字符上并输入x
  - 为删除光标之前（其左边）的一个字符，需输入X
- 删除一个词或词的部分内容：
  - 为删除一个词，需将光标放置到该词的开头并输入dw
  - 为删除词的部分内容，需将光标放置到该词要保存部分的右边。输入dw来删除该词余下的部分
- 删除一行：
  - 将光标放置到该行的任意处并输入dd
- 删除多行：
  - 从光标所在行开始删除n行输入：ndd

命令模式（续）
—— 删除文本（续）

- 删除行的部分内容：

  将光标放置到该行要保存部分的右边，并输入D。为删除光标左边的所有内容，须将光标放置到该行要删除部分的右边，并输入d0（d-零）。

- 删除到文件的结尾：

  为删除从当前行到文件结尾的所有内容，需输入dG

命令模式（续）
——撤消更改

- 撤消前一个命令

  如果在vi中出错，或者在一个操作完成后改主意了，可以在最后一个命令之后立即输入u来撤消该命令（输入u之后，不必按Esc），按多次u可执行多次撤销操作。

- 撤消对一行的更改

  输入U来撤消对一行所做的所有更改。这个命令只有在没将光标移动到该行以外时才生效。

命令模式（续）
——复制和移动文本

- 粘贴命令：p
- 复制一行命令：yy
- 复制多行命令：复制从光标所在行开始输入：nyy
- 如果要复制第m行到第n行之间的内容，可以在末行模式中输入m，ny，例如：3，5y复制第三行到第五行内容到缓存区。

命令模式（续）
——查找字符串

- 一个字符串是一行上的一个或几个字符。它可能包括字母、数字、标点符号、特殊字符、空格、跳格或回车。一个串可能是一个合语法的词，或一个词的部分内容。
- 为查找一个字符串，输入/，并在/后面输入要查找的串，然后按下回车。vi将光标定位在该串下一次出现的地方。
- 例如，要查找串meta，则输入/meta，接着按回车。输入 n 跳转到该串的下一个出现处；输入N跳转到该串的上一个出现处。

命令模式（续）
——查找字符串

- 为在文件中回头查找，使用？代替/。在此情况下，n与N的方向是相反的。查找通常是区分大小写的。
- 某些特殊字符（ / & ! . ^ * $ \ ? ）对查找过程有特殊意义，并且在查找中被使用时必须“转意”。
- 为转意一个特殊字符，需在其前面加一个反斜杠（\）。例如，要查找串“anything? ”，则输入“/anything\? ”并且按下回车。这些特殊字符可以被当作有查找功能的命令来使用，因此当要查找一个包含一个或多个这种字符的串时，必须在该字符前加一个反斜杠作为指示。要转意反斜杠本身，则输入\\。

命令模式（续）
—— 替换字符串

- 在一行内替换头一个字符串old为新的字符串new，输入：:s/old/new
- 在一行内替换所有的字符串old为新的字符串new，输入：:s/old/new/g
- 在多行内替换所有的字符串old为新的字符串new，输入：:#,#s/old/new/g（两个#之间的部分为要替换的行数范围）
- 在文件内替换所有的字符串old为新的字符串new，输入：:%s/old/new/g
- 进行全文替换时询问用户确认每个替换需添加c选项，输入：:%s/old/new/gc

输入模式

- 要想在示范文件hello中输入内容，键入vi的“输入”命令i。这使vi退出命令模式，进入输入模式。
- 现在输入几个短的文本行，每一行以回车结尾。输入的字符显示在光标的左边，并且将所有的现有字符往右推。在按下回车之前，可以通过倒退和重新输入一行来改正错误。
- 当结束对hello的文本输入时，按下Esc返回到命令模式。光标回到最后输入的字符上。现在可以输入更多的vi命令了。如果vi看起来反应不正常，请确认不是在大写字母锁定模式下,因为在该模式下，输入的都是大写字母，而vi的命令是区分大小写的。

输入模式（续）

- 添加：
  - 输入a后，在光标的右边插入文本
  - 输入A，在一行的结尾处添加文本
- 插入：
  - 通过在命令模式下输入i，在光标的左边插入文本
  - 通过在命令模式下输入I，在行首插入文本
- 插入新行：
  - 输入o，在当前光标位置下面打开一行
  - 输入O，在当前光标位置上面打开一行

末行模式

- 大多数vi命令不要求按下回车来执行这些命令，但以冒号（：）开头的命令则要求在命令后按下回车。在一些对于vi编辑程序的讨论中，将这种前面带有冒号的命令看作是独立的、第三种vi模式——末行模式。这是因为在命令模式下输入这个冒号时，冒号和所输入的其他内容会出现在屏幕的最下面一行。然而，对这种讨论来说，所有的vi命令都是从命令模式开始的。

- 以冒号开头的命令实际上是ex 命令，vi和ex是同一个文本编辑程序的两种独立的界面。即vi面向屏幕的界面，而ex面向行的界面。ex命令的整个集合在vi中都可以用。当按下冒号时，实际上是切换到以行为主的ex界面。这允许在不离开vi的情况下，执行许多文件操作命令。

- vi用大量多少有些同义的命令来控制将缓冲区内容保存到一个文件和退出vi。这些命令提供了保存、保存并退出、退出但不保存的选择。

末行模式

- 末行模式操作

set nu	显示行号
set nonu	取消行号
set ic	搜索时忽略大小写
set noic	搜索时区分大小写
/关键字	按n向后查找字符
?关键字	按n向前查找字符
n	跳到某一行
!cmd	运行shell命令

末行模式(续)

- 保存

  输入:w后面接回车键。

- 保存并退出

  输入:wq后面接回车键。或者输入ZZ（大写，命令行模式）。

  注意，命令ZZ既不以冒号开头，也不后接回车键。

- 退出但不保存

  当没有更改文件，只想退出时，则输入:q后按回车键。

  如果不想保存这些更改，则输入:q!后面接回车键。

总结

- Linux的常用命令
- 命令行的使用方法
- 帮助的使用方法
- 文本编辑器（vi）的使用方法

问题交流