

更多嵌入式 Linux 学习资料, 请关注: 一口 Linux 回复关键字:1024



很多粉丝给一口君留言, 想要学习 Linux 资料, 其实关注一口君的公众号, 后台回复 1024 , 就有很多非常不错的电子书, 但是有一个问题, 很多粉丝是初学者, 而这一大堆电子书, 估计随便一本, 还没看完就基本上开始劝退了。

### 为什么呢?

因为 Linux 的知识体系非常的庞大, IT 行业很多领域都需要使用到 Linux, 有运维的、有应用程序开发的、有驱动开发的、有系统优化的、有搞单片机的、有做系统移植的、有做网络产品的等等。

总结一句话, IT 的很大部分从业者都需要掌握 Linux 的部分知识, 但是由于每个人的从事领域不一样, 对 Linux 的要求也不一样, 这就直接导致, 网上搜索学习路线有很多种, 还有就是推荐的书籍很多都不利于初学者。

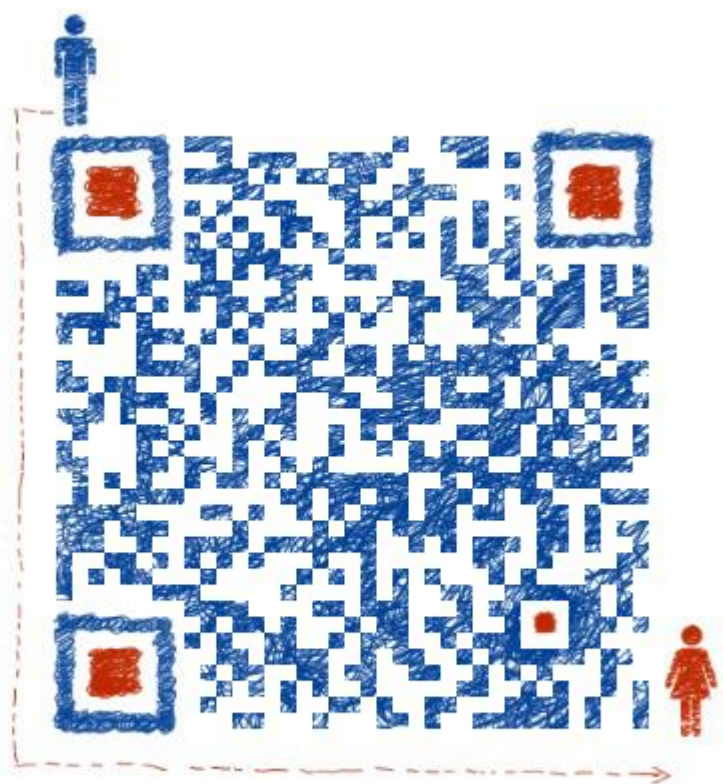
比如 Linux 命令的学习, 很多人都力推《鸟哥的私房菜》这本书, 一口君在直播间多次强调, 初学者不要看这本书, 不是说这本书不好, 而是, 这本书其实你要当做一本工具书来看, 如果你的目标是想快速的基于 Linux 做一些开发工作, 最迫切的就是快速掌握一些基本的命令, 一些和开发相关的最基础的知识, 而不是去学习那些可能这辈子都用不到的命令。

为了让初学者更好的学习 Linux, 入门 Linux, 一口君特地整理了 Linux 入门必须掌握的一些基础知识点, 掌握这些知识点之后, 就可以学习 C 编程的知识了, 后续遇到一些不熟悉的命令和配置, 只需要网上搜索下就可以很快上手了。

后续一口君会陆续录制 Linux 入门的视频, 手把手教大家 Linux 入门。

欢迎大家访问我的 B 站空间, 《[从 0 学 Linux 驱动](#)》正在更新中。

下面是一口君的公众号, 也欢迎大家加我好友[yikoupeng], 拉你进高效技术讨论群, 一起讨论 Linux 相关的技术。



## 一、Linux 操作系统概述

### 1、发展

1991 Linus Linux 0.0.1 版,代码为 8K 行。现在最新版本为 5.12.4。Linux 加入了 GNU, 整体基于 GPL 协议, 允许开源、分享传播、修改。

内核下载的地址【初学者知道即可】:

<https://www.kernel.org/>

### 2、组成

#### 1. kernel 内核:

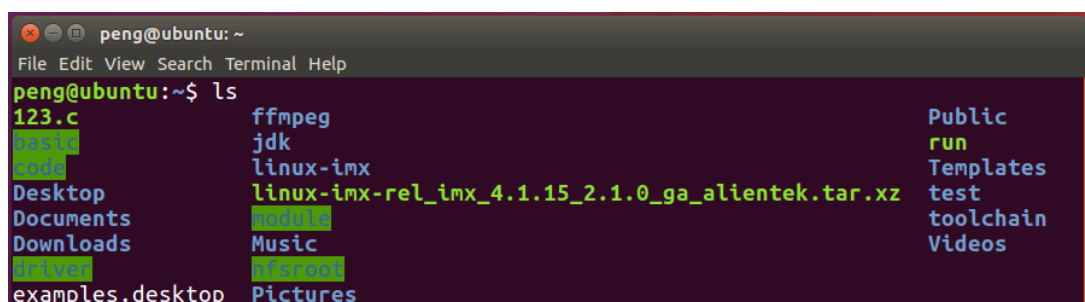
核心程序, 用于管理硬件设备、系统的线程进程、内存、交换空间、文件系统、精灵进程(守护进程)等。主要实现系统程序与硬件之间的控制管理功能。

## 2. Shell:

包裹在内核之外的人机交互界面, 用于用户和内核之间打交道的功能, 类似于 windows 的 cmd。  
通过 Shell 将输入的命令与内核通讯, 好让内核可以控制硬件开正确无误的操作工作。

Shell 有着不同的分类, 比如 Bourne shell (sh), Korn shell (ksh)、C shell (csh)、Bourne-again shell (bash)、tcsh。其中最常用的有 csh 和 bash。

ubuntu 16.04 的 terminal 如下:



```
peng@ubuntu: ~  
File Edit View Search Terminal Help  
peng@ubuntu:~$ ls  
123.c          ffmpeg          Public  
basic          jdk             run  
code          linux-ixmx      Templates  
Desktop       linux-ixmx-rel_ixmx_4.1.15_2.1.0_ga_alientek.tar.xz test  
Documents     module          toolchain  
Downloads     Music           Videos  
driver        nfsroot  
examples.desktop Pictures
```

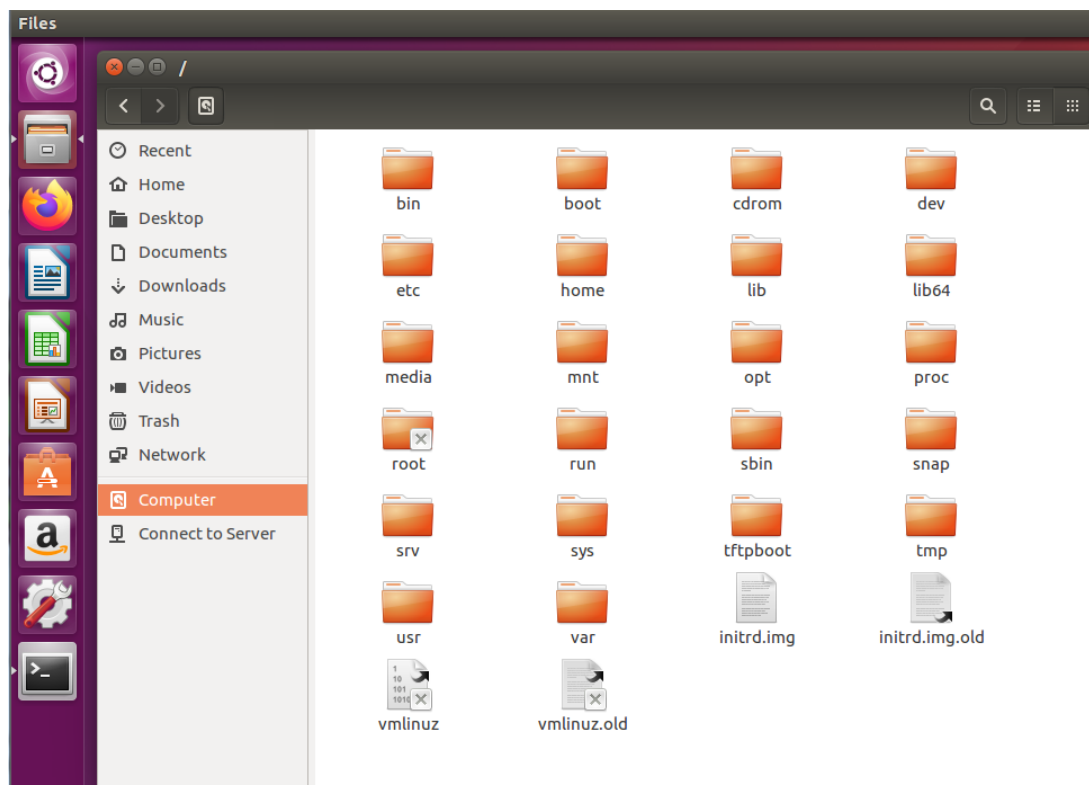
## 3. user application

Linux 根据程序的运行, 分为用户空间和内核空间, 简单的理解就是, 普通的应用程序运行在用户空间, 一些涉及到系统核心资源的操作的程序运行在内核空间, 比如 TCP/IP 协议栈、驱动、进程调度、内存管理、文件系统等都运行于 Linux 内核空间,

其实一些应用程序当需要访问系统资源的时候, 必须通过系统调用, 通过一些内核函数将系统资源由内核空间拷贝到用户空间。

## 4. Files System:

文件系统, 管理文件和目录。



### 3、Linux 的特性:

多用户、多任务（进程、线程处理），多平台，图形化界面（x-windows）、硬件低配置、通信与联网、应用程序的支持（编辑器、编辑工具、数据库、办公软件、图形处理、Internet 应用、游戏）。

## 二、Linux 操作系统安装

### 1、Linux 的选择

**red hat:**

企业级的，已经开始收费（商用）。

**CentOS**

是 RHEL 的克隆版本。RedHat 一直都提供源代码的发行方式，CentOS 就是将 RedHat 发行的源代码重新编译一次，形成一个可使用的二进制版本。由于 Linux 的源代码是 GNU，所以从获得 RedHat 的源代码到编译成新的二进制，都是合法。只是 RedHat 是商标，所以必须在新的发行版里将 RedHat 的商标去掉。

通常搭建服务器选用 CentOS。

**ubuntu:**

桌面开源的，比较纯正的 Linux，android 官方指定的编译操作系统，发展快、已支持 ARM 架构。

ubuntu 在开发者中，非常受欢迎，一口君所有的文章和视频都以 ubuntu 为主。

## 2、安装 Ubuntu Desktop

Linux 环境安装篇幅较长，安装详细步骤参考这篇文章：

《[linux 环境搭建-ubuntu16.04 安装](#)》

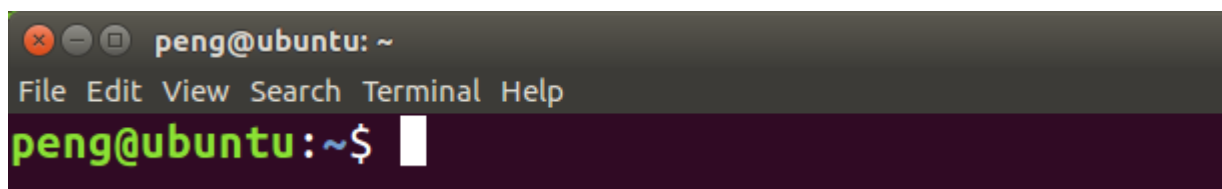
### 3、基本操作

`ctrl+alt F2` 可以进入终端界面

`ctrl+alt F7` 进入桌面界面

`ctrl+alt+t` 文本编辑

我们通常用快捷键 **ctrl+alt+t** 打开一个终端，这个一定要记住。



```
peng@ubuntu: ~$
```

其中:

peng 当前用户

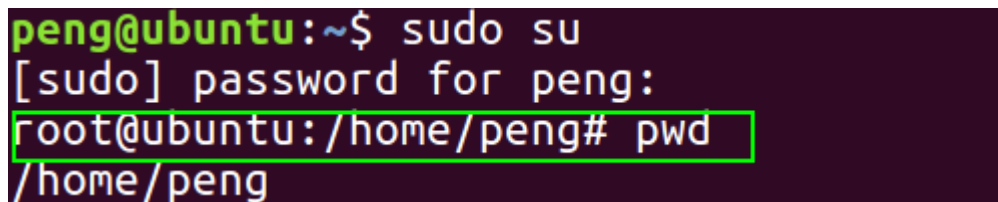
ubuntu 主机名

~ 当前用户主目录, /home/peng, 每一个用户都会在/home 下创建一个与用户名同名的目

录

\$ 普通用户

如果切换到管理员用户



```
peng@ubuntu:~$ sudo su
[sudo] password for peng:
root@ubuntu:/home/peng# pwd
/home/peng
```

root 管理员

/home/peng 当前工作目录

# 当前是管理员

## 三、Linux 文件系统及文件基础

### 1、文件系统概述

Linux 中一切皆为文件，文件系统用来组织计算机的文件和资料的系统，是操作系统封装的一个系统服务程序，实际是一个软件程序，用来存储和管理计算机文件和资料。  
文件系统分类如下：

- 磁盘文件系统：NTFS,EXT3
- 闪存文件系统：JFFS2,YAFFS
- 数据库文件系统：BFFS,WINFS
- 网络文件系统：NFS
- 虚拟文件系统：VFS (Proc)

文件系统的功能：能定义文件的组织方式，文件的结构；提供建立和存取文件的环境（目录和文件）

## 2、Linux 文件系统及文件

1. EXT3：是一个日志方式的文件系统，系统中的每个文件都有索引，用户对文件的每个操作都会记录在日志中，形成一个任务队列。
2. SWAP：是交换分区的文件系统，类似于 windows 的虚拟内存，其实现的方式有以下两种：  
第一种是进行内存排列像内存池一样，进行优化；  
第二种是把硬盘上的空间模拟成内存。  
Swap 是 Linux 的虚拟内存，在安装时要设置好大小，一般设置为物理内存的两倍。
3. 目录结构  
linux 的结构是由很多文件块区组成，与 Windows 分区不同。

目录	应放置档案内容
/bin	系统有很多放置执行档的目录，但/bin 比较特殊。因为/bin 放置的是在单人维护模式下还能够被操作的指令。在/bin 底下的指令可以被 root 与一般帐号所使用，主要有：cat,chmod(修改权限),chown, date, mv, mkdir, cp, bash 等等常用的指令。
/boot	主要放置开机会使用到的档案，包括 Linux 核心档案以及开机选单与开机所需设定档等等。Linux kernel 常用的档名为：vmlinuz，如果使用的是 grub 这个开机管理程式，则还会存在/boot/grub/这个目录。
/dev	在 Linux 系统上，任何装置与周边设备都是以档案的型态存在于这个目录当中。 只要通过存取这个目录下的某个档案，就等于存取某个

目录	应放置档案内容
	装置。比要重要的档案有 /dev/null, /dev/zero, /dev/tty, /dev/lp, /dev/hd, /dev/sd* 等等
/etc	系统主要的设定档几乎都放置在这个目录内, 例如人员的帐号密码档、各种服务的启始档等等。一般来说, 这个目录下的各档案属性是可以让一般使用者查阅的, 但是只有 root 有权力修改。FHS 建议不要放置可执行档(binary)在这个目录中。比较重要的档案有: /etc/inittab, /etc/init.d/, /etc/modprobe.conf, /etc/X11/, /etc/fstab, /etc/sysconfig/ 等等。另外, 其下重要的目录有: /etc/init.d/: 所有服务的预设启动 script 都是放在这里的, 例如要启动或者关闭 iptables 的话: /etc/init.d/iptables start、/etc/init.d/iptables stop
/etc/xinetd.d/	这就是所谓的 super daemon 管理的各项服务的设定档目录。
/etc/X11/	与 X Window 有关的各种设定档都在这里, 尤其是 xorg.conf 或 XF86Config 这两个 X Server 的设定档。
/home	这是系统预设的使用者家目录(home directory)。在你新增一个一般使用者帐号时, 预设的使用者家目录都会规范到这里来。比较重要的是, 家目录有两种代号: ~: 代表当前使用者的家目录, 而 ~guest: 则代表用户名为 guest 的家目录。
/lib	系统的函式库非常的多, 而/lib 放置的则是在开机时会用到的函式库, 以及在/bin 或/sbin 底下的指令会呼叫的函式库而已。什么是函式库呢? 妳可以将他想成是外挂, 某些指令必须要有这些外挂才能够顺利完成程式的执行之意。尤其重要的是/lib/modules/这个目录, 因为该目录会放置核心相关的模组(驱动程序)。
/media	media 是媒体的英文, 顾名思义, 这个/media 底下放置的就是可移除的装置。包括软碟、光碟、DVD 等等装置都暂时挂载于此。常见的档名有: /media/floppy, /media/cdrom 等等。
/mnt	如果妳想要暂时挂载某些额外的装置, 一般建议妳可以放置到这个目录中。在古早时候, 这个目录的用途与/media 相同啦。只是有了/media 之后, 这个目录就用来暂时挂载用了。
/opt	这个是给第三方协力软体放置的目录。什么是第三方协力软体啊? 举例来说, KDE 这个桌面管理系统是一个独立的计画, 不过他可以安装到 Linux 系统中, 因此 KDE 的软体就建议放置到此目录下了。另外, 如果妳想要自行安装额外的软体(非原本的 distribution 提供的), 那么也能够将你的软体安装到这里来。不过, 以前的 Linux 系统中, 我们还是习惯放置在/usr/local 目录下。

目录	应放置档案内容
/root	系统管理员(root)的家目录。之所以放在这里，是因为如果进入单人维护模式而仅挂载根目录时，该目录就能够拥有 root 的家目录，所以我们会希望 root 的家目录与根目录放置在同一个分区中。
/sbin	Linux 有非常多指令是用来设定系统环境的，这些指令只有 root 才能够利用来设定系统，其他使用者最多只能用来查询而已。放在/sbin 底下的为开机过程中所需要的，里面包括了开机、修复、还原系统所需要的指令。至于某些伺服器软体程式，一般则放置到/usr/sbin/当中。至于本机自行安装的软体所产生的系统执行档(system binary)，则放置到/usr/local/sbin/当中了。常见的指令包括：fdisk, fsck, ifconfig, init, mkfs 等等。
/srv	srv 可以视为 service 的缩写，是一些网路服务启动之后，这些服务所需要取用的资料目录。常见的服务例如 WWW, FTP 等等。举例来说，WWW 伺服器需要的网页资料就可以放置在/srv/www/里面。呵呵，看来平时我们编写的代码应该放到这里了。
/tmp	这是让一般使用者或者是正在执行的程序暂时放置档案的地方。这个目录是任何人都能够存取的，所以你需要定期的清理一下。当然，重要资料不可放置在此目录啊。因为 FHS 甚至建议在开机时，应该要将/tmp 下的资料都删除。

### 3. Linux 文件属性：

Linux 文件属性一共 7 种：

类型	字母	说明
普通文件类型	-	Linux 中最多的一种文件类型，包括 纯文本文件(ASCII)；二进制文件(binary)；数据格式的文件(data)；各种压缩文件.第一个属性为 [-]
目录文件	d	就是目录，能用 # cd 命令进入的。第一个属性为 [d]，例如 [drwxrwxrwx]
块设备文件	b	块设备文件：就是存储数据以供系统存取的接口设备，简单而言就是硬盘。例如一号硬盘的代码是 /dev/hda1 等文件。第一个属性为 [b]
字符设备	c	字符设备文件：即串行端口的接口设备，例如键盘、鼠标等等。第一个属性为 [c]



类型	字母	说明
套接字文件	s	这类文件通常用在网络数据连接。可以启动一个程序来监听客户端的要求, 客户端就可以通过套接字来进行数据通信。第一个属性为 [s], 最常在 /var/run 目录中看到这种文件类型
管道文件	p	FIFO 也是一种特殊的文件类型, 它主要的目的是, 解决多个程序同时存取一个文件所造成的错误。FIFO 是 first-in-first-out(先进先出)的缩写。第一个属性为 [p]
链接文件	l	类似 Windows 下面的快捷方式。第一个属性为 [l], 例如 [lrwxrwxrwx]

## 1、普通文件

使用 ls -l 命令后,

```
-rwxrwx-rw- 1 peng peng 729 Mar 26 06:34 123.c
```

第一列第一个字符为 "-" 的文件为普通文件。

## 2、目录文件

Linux 中的目录也是文件, 目录文件中保存着该目录下其他文件的 inode 号 和文件名等信息, 目录文件中的每个数据项都是指向某个文件 inode 号的链接, 删除文件名就等于删除与之对应的链接。目录文件的字体颜色是蓝色, 使用 ls -l 命令查看, 第一个字符为"d" (directory)。

目录文件的权限:

- 1) r 表明该目录文件具有可读权限, 即可以使用 ls 命令查看该目录的存储情况;
- 2) w 表明该目录文件具有写权限, 即可以往该目录下添加、修改、删除文件;
- 3) x 表明该目录文件具有可执行文件, 即可以使用 cd 命令进入到该目录下。

可以使用 chmod 指令来改变文件的权限。

## 3、链接文件

链接文件一般指的是一个文件的软连接 (或符号链接), 使用 ls -l 命令查看, 第一个符号为 "l", 文件名为浅蓝色, 如下:

```
peng@ubuntu:~/test$ ls -l
total 12
-rw-rw-r-- 2 peng peng 66 May 14 20:26 test_hardlink
lrwxrwxrwx 1 peng peng 8 May 14 20:58 test_softlink -> test.txt
```

这里, test\_softlink 就是一个链接文件, 从结果上还可以看到它是文件 test.txt 的软链接, 删除原文件

test.txt 的话, 对应的软链接文件 test\_softlink 也会消失。可以使用 ln 命令来创建一个文件的链接文件:

#### 1) 软链接

软链接 (又称符号链接), 使用 ln -s file file\_softlink 命令可以创建一个文件的软链接文件:

```
ln -s test.txt test_softlink
```

软链接相当于给原文件创建了一个快捷方式, 如果删除原文件, 则对应的软链接文件也会消失。

#### 2) 硬链接

硬链接, 相当于给原文件取了个别名, 其实两者是同一个文件, 删除二者中任何一个, 另一个不会消失; 对其中任何一个进行更改, 另一个的内容也会随之改变, 因为这两个本质上是同一个文件, 只是名字不同。使用 ls -li 命令查看, 可以发现硬链接的两个文件的 inode 号是一样的:

```
peng@ubuntu:~/test$ ls -li
total 12
14812879 -rw-rw-r-- 2 peng peng 66 May 14 20:26 test_hardlink
14812882 lrwxrwxrwx 1 peng peng 8 May 14 20:58 test_softlink -> test.txt
14812879 -rw-rw-r-- 2 peng peng 66 May 14 20:26 test.txt
```

同样的, 使用 ln 命令可以创建一个文件的硬链接:

```
ln test.txt test_hardlink
```

## 4、设备文件

Linux 中的硬件设备如硬盘、鼠标等也都被表示为文件, 即为设备文件。

设备文件一般存放在 /dev/ 目录下, 文件名为黄色, 如下:

```
peng@ubuntu:/dev$ ls
agpgart      loop2      snapshot   tty32      tty62      ttyS5
autofs       loop3      snd        tty33      tty63      ttyS6
block        loop4      sr0        tty34      tty7        ttyS7
bsg          loop5      stderr     tty35      tty8        ttyS8
btrfs-control loop6      stdin      tty36      tty9        ttyS9
bus          loop7      stdout     tty37      ttyprintk  uhid
cdrom        loop-control tty         tty38      ttyS0       uinput
cdrw         mapper     tty0       tty39      ttyS1       urandom
char         mcelog    tty1       tty4       ttyS10      userio
console      mem       tty10      tty40      ttyS11      vcs
```

设备文件分两种:

- 1) 块设备文件:

块设备文件支持以块 (block) 为单位的访问方式。在 EXT4 文件系统中, 一个 block 通常为 4KB 的大小, 也就是说每次可以存取 4096 (或其整数倍) 个字节的数据。应用程序可以随机访问块设备文件的数据, 程序可以自行确定数据的位置, 硬盘、软盘等都是块设备。使用 ls -li 命令查看, 块设备文件的第一个字符是 "b" (block)。

- 2) 字符设备文件:

字符设备文件以字节流的方式进行访问, 由字符设备驱动程序来实现这种特性, 这通常要用到 `open`、`close`、`read`、`write` 等系统调用。字符终端、串口和键盘等就是字符设备。另外, 由于字符设备文件是以文件流的方式进行访问的, 因此可以顺序读取, 但通常不支持随机存取。使用 `ls -l` 命令查看, 字符设备文件的第一个字符是 "c" (char)。

## 5、管道文件 (FIFO 文件)

管道文件主要用于进程间通信, 使用 `ls -l` 命令查看, 第一个字符为 "p" (pipe)。可以使用 `mkfifo` 命令来创建一个管道文件:

```
mkfifo fifo_file
```

```
peng@ubuntu:~/test$ mkfifo fifo_file
peng@ubuntu:~/test$ ls
fifo_file  test_hardlink  test_softlink  test.txt  wait.c
peng@ubuntu:~/test$ ls -l
total 12
prw-rw-r-- 1 peng peng    0 May 14 21:05 fifo_file
```

在 FIFO 中可以很好地解决在无关进程间数据交换的要求, FIFO 的通信方式类似于在进程中使用文件来传输数据, 只不过 FIFO 类型的文件同时具有管道的特性, 在读取数据时, FIFO 管道中同时清除数据。

## 6、套接字文件

套接字文件, 通常指域套接字文件, 使用 `ls -l` 命令查看, 第一个字符为 "s"。

域套接字是进程间通信(IPC)的一种方法, 是可靠的一种 IPC 通信, 是 POSIX 标准的一个组件, 只能用于同一主机间的通信。

后续学习进程间通信需要学习该知识。

## 4. 文件颜色

常见的文件颜色如下:

- 蓝色: 目录文件
- 绿色: 可执行文件
- 浅蓝色: 链接文件
- 红色: 压缩文件
- 黄色: 字符设备

- 灰色: 其他文件

```
crw----- 1 root root 10, 231 May 6 05:31 snapshot 字符设备
drwxr-xr-x 3 root root 200 May 6 05:31 snd 目录
brw-rw----+ 1 root cdrom 11, 0 May 6 05:31 sr0
lrwxrwxrwx 1 root root 15 May 6 05:31 stderr -> /proc/self/fd/2 链接文件
lrwxrwxrwx 1 root root 15 May 6 05:31 stdin -> /proc/self/fd/0
lrwxrwxrwx 1 root root 15 May 6 05:31 stdout -> /proc/self/fd/1
-rw-rw-r-- 1 peng peng 10240 May 14 20:51 peng.tar 压缩文件
-rwxr-xr-x 1 root root 10408 Mar 18 05:01 run 可执行文件
```

# 四、Linux 操作系统命令使用基础

## 1、命令格式

```
$command [option(s)] [argument(s)]
```

```
命令名 空格 选项 空格 参数
```

- command : 命令名
- [option(s)] : 选项
- [argument(s)] : 参数

注意:

- 在命令行中, 每两个部分之间有空格分隔
- 每个命令行可使用的最多的命令字符是 256 个
- 命令区分大小写
- 不同的命令提示符使用分隔符号 "/"
- 命令中的参数/选项可以是多个, 并且参数其实就是要传入命令程序主函数 main 的参数。
- [ ] 表示这个内容可以不包含, 比如 [argument(s)], 输入命令时可以不加参数

## 2、联机帮助、清屏与历史记录命令

### 1) 联机帮助

遇到一些函数和命令, 不知道含义时可以用命令 man 来查看帮助信息。

```
man ls
```

```
man -k keyword
```

```
LS(1)                                User Commands                                LS(1)

NAME
  ls - list directory contents

SYNOPSIS
  ls [OPTION]... [FILE]...

DESCRIPTION
  List information about the FILES (the current directory by default).
  Sort entries alphabetically if none of -cftuvSUX nor --sort is speci-
  fied.

  Mandatory arguments to long options are mandatory for short options
  too.
```

man 手册根据内容分为 9 个手册, 可以使用下面命令查看

```
man man
```

```
DESCRIPTION
  man is the system's manual pager. Each page argument given to man is
  normally the name of a program, utility or function. The manual page
  associated with each of these arguments is then found and displayed. A
  section, if provided, will direct man to look only in that section of
  the manual. The default action is to search in all of the available
  sections following a pre-defined order ("1 n l 8 3 2 3posix 3pm 3perl 5
  4 9 6 7" by default, unless overridden by the SECTION directive in
  /etc/manpath.config), and to show only the first page found, even if
  page exists in several sections.

  The table below shows the section numbers of the manual followed by the
  types of pages they contain.

  1  Executable programs or shell commands
  2  System calls (functions provided by the kernel)
  3  Library calls (functions within program libraries)
  4  Special files (usually found in /dev)
  5  File formats and conventions eg /etc/passwd
  6  Games
  7  Miscellaneous (including macro packages and conventions), e.g.
    man(7), groff(7)
  8  System administration commands (usually only for root)
  9  Kernel routines [Non standard]
```

man 文档的分类编号

```
1 - commands (普通的命令)
```

```
2 - system calls (系统调用)
```

3 - library calls (库函数)

4 - special files (特殊文件: /dev 下设备文件)

5 - file formats and conversions (文件格式)

6 - games [for](#) linux (游戏和娱乐)

7 - macro packages and conventions (杂项)

8 - system management commands (管理员命令)

9 - 其他 (Linux 特定, 用来存放内核例行程序的文档)

使用 -k 参数, man 可以在所有的 man 文档和简介中查找符合条件的命令。

```
peng@ubuntu:~/test$ man -k bash
```

```
bash (1) - GNU Bourne-Again SHell
```

```
bash-builtins (7) - bash built-in commands, see bash(1)
```

```
bashbug (1) - report a bug in bash
```

```
builtins (7) - bash built-in commands, see bash(1)
```

```
dh_bash-completion (1) - install bash completions for package
```

```
rbash (1) - restricted bash, see bash(1)
```

有的关键词在系统中对应多个手册,  
使用-f 参数,用于查找同名的手册:

```
peng@ubuntu:~/test$ man -f time
```

```
time (7) - overview of time and timers
```

```
time (1) - run programs and summarize system resource usage
```

```
time (2) - get time in seconds
```

## 2) 清屏 clear

清屏使用命令 clear 或者使用快捷键: ctrl + l

## 3) 历史纪录 history (history -n)

这是一个非常有用的命令, 想知道之前输入过的所有命令, 就可以用他。

```
history : 查看最近使用的命令, 最多 500 条
```

```
history n : 曾经使用的最近 n 条命令
```

```
!n : 执行最近第 n 条命令
```

```
!! : 执行最近使用的第一条命令
```

```
方向上键: 执行上一条命令
```

```
方向下键: 执行下一条命令
```

# 五、文件管理

## 1) 查看文件目录

下面这几个命令是使用最频繁的命令：

命令	说明
pwd	显示当前工作目录
cd [要改变的目录]	改变目录
ls	列出当前目录的文件和子目录
file	辨识文件类型

- cd

```
cd //回到当前用户主目录
```

```
cd ~//回到当前用户主目录
```

```
cd ~[用户名] //进入指定用户主目录
```

- file

功能说明：辨识文件类型。

```
语法：file [-beLvz] [-f <名称文件>] [-m <魔法数字文件>...] [文件或目录...]
```

参数：

```
-b 列出辨识结果时，不显示文件名称。
```

```
-c 详细显示指令执行过程，便于排错或分析程序执行的情形。
```



`-f<名称文件>` 指定名称文件, 其内容有一个或多个文件名称呢感, 让 `file` 依序辨识

这些文件, 格式为每列一个文件名称。

`-L` 直接显示符号连接所指向的文件的类别。

`-m<魔法数字文件>` 指定魔法数字文件。

`-v` 显示版本信息。

`-z` 尝试去解读压缩文件的内容。

补充说明: 通过 `file` 指令, 我们得以辨识该文件的类型。

```
peng@ubuntu:~/test$ file wait.c
```

```
wait.c: ASCII text
```

```
peng@ubuntu:~/test$ file fifo_file
```

```
fifo_file: fifo (named pipe)
```

## 2) 文件路径

什么是文件的路径?

就是文件存放的地方, 可以联想为 文件的“家”。

在 Linux 中, 存在着绝对路径和相对路径:

### 绝对路径:

路径的写法一定是由根目录 `/` 写起的, 例如 `/usr/local/mysql`

### 相对路径:

路径的写法不是由根目录 / 写起的。

例如 首先用户进入到 /home, 然后再进入到 peng, 执行的命令为

```
#cd /home
```

```
#cd peng
```

此时用户所在的路径为 /home/peng。

第一个 cd 命令后紧跟/home, 前面有斜杠, 是绝对路径;

而第二个 cd 命令后紧跟 peng, 前面没有斜杠, 表示从当前目录下找 peng 这个目录, 这个 peng 是相对于/home 目录来讲的, 所以称为相对路径。

## 3) 创建和删除文件目录

### 1、创建文件 touch

功能说明:

文件名不存在, 则创建一个新的空文件

如果文件名存在, 更新该文件或者目录的修改访问时间, 内容不变。

语法:

```
touch [-acfm] [-d <日期时间>] [-r <参考文件或目录>] [-t <日期时间>] [-help] [-
```

```
version] [文件或目录...]
```

```
或 touch [-acfm] [-help] [-version] [日期时间] [文件或目录...]
```

补充说明:

使用 touch 指令可更改文件或目录的日期时间, 包括存取时间和更改时间。

参数:

`-a` 或 `-time=atime` 或 `-time=access` 或 `-time=use` 只更改存取时间。

`-c` 或 `-no-create` 不建立任何文件。

`-d<时间日期>` 使用指定的日期时间, 而非现在的时间。

`-f` 此参数将忽略不予处理, 仅负责解决 BSD 版本 `touch` 指令的兼容性问题。

`-m` 或 `-time=mtime` 或 `-time=modify` 只更改变动时间。

`-r<参考文件或目录>` 把指定文件或目录的日期时间, 统统设成和参考文件或目录的日

期时间相同。

`-t<日期时间>` 使用指定的日期时间, 而非现在的时间。

`-help` 在线帮助。

`-version` 显示版本信息。

## 2、创建目录 `mkdir`

功能说明:

建立目录

语法:

```
mkdir [-p] [-help] [-version] [-m <目录属性>] [目录名称]
```

补充说明:

mkdir 可建立目录并同时设置目录的权限。

参数:

`-m<目录属性>或-mode<目录属性>` 建立目录时同时设置目录的权限。

`-p` 或 `-parents` 若所要建立目录的上层目录目前尚未建立, 则会一并建立上层目录。

`-help` 显示帮助。

`-verbose` 执行时显示详细的信息。

`-version` 显示版本信息。

### 3、删除文件 rm

功能说明:

删除文件或目录。

语法:

```
rm [-dfirv] [-help] [-version] [文件或目录...]
```

补充说明: 执行 rm 指令可删除文件或目录, 如欲删除目录必须加上参数“`-r`”, 否则预设仅会删除文件。

参数:

`-d` 或 `-directory` 直接把欲删除的目录的硬连接数据删成 0, 删除该目录。

`-f` 或 `-force` 强制删除文件或目录。

`-i` 或 `-interactive` 删除既有文件或目录之前先询问用户。

`-r` 或 `-R` 或 `-recursive` 递归处理, 将指定目录下的所有文件及子目录一并处理。

`-v` 或 `-verbose` 显示指令执行过程。

`-help` 在线帮助。

`-version` 显示版本信息。

#### 4、删除目录

`rmdir(remove directory)`

功能说明: 删除目录。

语 法: `rmdir [-p][-help][-ignore-fail-on-non-empty][-verbose][-version][目录...]`

补充说明: 当有空目录要删除时, 可使用 `rmdir` 指令。

参 数:

`-p` 或 `-parents` 删除指定目录后, 若该目录的上层目录已变成空目录, 则将其一并删除。

`-help` 在线帮助。

`-ignore-fail-on-non-empty` 忽略非空目录的错误信息。

`-verbose` 显示指令执行过程。

`-version` 显示版本信息。

### 3) 显示文件内容

#### 1. cat

功能:

把档案串连接后传到基本输出到屏幕或加 `> fileName` 到另一个档案

使用权限:

所有使用者

语法:

```
cat [-AbeEnstTuv] [-help] [-version] fileName
```

参数:

`-n` 或 `-number` 由 1 开始对所有输出的行数编号

`-b` 或 `-number-nonblank` 和 `-n` 相似, 只不过对于空白行不编号

`-s` 或 `-squeeze-blank` 当遇到有连续两行以上的空白行, 就代换为一行的空白行

`-v` 或 `-show-nonprinting`

范例:

```
cat -n textfile1 > textfile2 把 textfile1 的档案内容加上行号后输入 textfile2
```

这个档案里

```
cat -b textfile1 textfile2 >> textfile3 把 textfile1 和 textfile2 的档案内
```

容加上行号 (空白行不加) 之后将内容附加到 textfile3 里。

```
cat < /dev/stdin > 1.txt //利用输入重定向 CTRL+D 结束输入
```

```
cat /dev/null > 1.txt //输出重定向进行清空
```

## 2. head

功能:

查找文件的前多少行

语法:

```
head [-n] filename
```

### 3. tail

功能:

查找文件的末尾多少行

语法:

```
tail [-n] filename
```

### 4. more

功能:

分屏显示

用法

```
more filename
```

## 4) 拷贝和移动文件目录

### 1. cp

功能说明:

```
cp 源文件 目标文件
```

将源文件复制为目标文件或目录。

语法:

```
cp [-abdfilpPrRsuvx] [-S <备份字尾字符串>] [-V <备份方式>] [-help] [-
```

```
spares=<使用时机>] [-version] [源文件或目录] [目标文件或目录] [目的目录]
```

补充说明: cp 指令用在复制文件或目录, 如同时指定两个以上的文件或目录, 且最后的目的地是一个已经存在的目录, 则它会把前面指定的所有文件或目录复制到该目录中。若同时指定多个文件或目录, 而最后的目的地并非是一个已存在的目录, 则会出现错误信息。

## 2. mv

功能说明:

```
mv [源文件、目录] [目的文件、目录]
```

移动或更名现有的文件或目录。

语法:

```
mv [-bfiuv] [-help] [-version] [-S <附加字尾>] [-V <方法>] [源文件或目录] [目标文件
```

```
或目录]
```

补充说明:

mv 可移动文件或目录, 或是更改文件或目录的名称。

参数:

```
-b 或 -backup 若需覆盖文件, 则覆盖前先行备份。
```



`-f` 或 `-force` 若目标文件或目录与现有的文件或目录重复, 则直接覆盖现有的文件

或目录。

`-i` 或 `-interactive` 覆盖前先行询问用户。

`-s`<附加字尾>或

`-suffix=<附加字尾>` 与 `-b` 参数一并使用, 可指定备份文件的所要附加的字尾。

`-u` 或 `-update` 在移动或更改文件名时, 若目标文件已存在, 且其文件日期比源文件

新, 则不覆盖目标文件。

`-v` 或 `-verbose` 执行时显示详细的信息。

`-V=<方法>`或

`-version-control=<方法>` 与 `-b` 参数一并使用, 可指定备份的方法。

`-help` 显示帮助。

`-version` 显示版本信息

## 5) 文件目录权限

### 1. 权限

文件的权限:

以普通文件为例, 使用 `ls -l` 命令, 可以看到结果的第一列是 `-rwxrwxrwx` 的形式, 其中第一个字符 `-` 表示这个文件为普通文件, 它也可以是其他的字符, 不同的字符代表不同类型的文件。其后的一串字符表明了该文件的权限, 其中:

- 1) `r` 表明该文件具有可读权限, 若该位置为 `-`, 则表明文件不可读;
- 2) `w` 表明该文件具有写权限, 若该位置为 `-`, 则表明文件不可写;
- 3) `x` 表明该文件具有可执行权限, 若该位置为 `-`, 则表明文件不具有可执行权限;
- 4) 第一个 `rwx` 表示该文件的所有者对该文件的权限; 第二个 `rwx` 表示该文件所属组对该文件的权限; 第三个 `rwx` 表示其他用户对该文件的权限。

## 2. 权限所属对象

文件所有者: 生成文件或目录的当前人, 权限最高, 用 `u` 表示。

文件所属用户组: 系统管理员分配的的同组一个或几个人, 用 `g` 表示。

其他人对此文件的权限: 除拥有者、用户组以外的人, 用 `o` 表示。

所有人: 包括拥有者, 所属用户组、其他用户, 用 `a` 表示

```
-rwxr(所有者) -xr(所在组) -x(其他人)
```

## 3. 修改属性 chmod

`chmod` 命令用来修改文件目录的访问权限, 修改权限的前提条件是在修改权限时具有可操作权限。

(a) 用字母表示权限

Who (`u`、`g`、`o`、`a`) + `cp` (`=` 设置权限 `+` 添加权限 `-` 删除权限) + permission (`r` 读权限 `w` 写权限 `x` 操作权限)

例如:

```
chmod g=wr 1.c;
```

```
chmod u+w,g-w,o=wr 1.c
```

(b) 用八进制数字表示权限

r	w	x	
0	0	0	无权限
1	1	1	有权限

```
R: 4, w; 2, x: 1
```

例如：

```
Chmod 777 build 将 build 的权限成所有人 rwx
```

```
peng@ubuntu:~/test$ touch 123.c
peng@ubuntu:~/test$ ls -l
total 0
-rw-rw-r-- 1 peng peng 0 May 15 05:35 123.c
peng@ubuntu:~/test$ chmod o+w 123.c
peng@ubuntu:~/test$ ls -l
total 0
-rw-rw-rw- 1 peng peng 0 May 15 05:35 123.c
```

other其他用户只有r权限  
other用户组增加w权限

注意：在 Ubuntu 中建立的文件默认权限是 664

## 4. Chown 改变文件的所属者和所属组

chown (英文全拼: change owner) 命令用于设置文件所有者和文件关联组的命令。

语法

```
chown [-cfhvR] [--help] [--version] user[:group] file...
```

参数：

```
user : 新的文件拥有者的使用者 ID
```

```
group : 新的文件拥有者的使用者组 (group)
```

```
-c : 显示更改的部分的信息
```

```
-f : 忽略错误信息
```

```
-h : 修复符号链接
```

`-v` : 显示详细的处理信息

`-R` : 处理指定目录以及其子目录下的所有文件

`--help` : 显示辅助说明

`--version` : 显示版本

实例:

把 `/var/run/httpd.pid` 的所有者设置 `root`:

```
chown root /var/run/httpd.pid
```

## 5. chgrp 改变文件或目录所属组

Linux `chgrp` (英文全拼: change group) 命令用于变更文件或目录的所属群组。

与 `chown` 命令不同, `chgrp` 允许普通用户改变文件所属的组, 只要该用户是该组的一员。

语法

```
chgrp [-cfhRv] [--help] [--version] [所属群组] [文件或目录...] 或 chgrp [-
```

```
cfhRv] [--help] [--reference=<参考文件或目录>] [--version] [文件或目录...]
```

参数说明:

`-c` 或 `--changes` 效果类似 "`-v`" 参数, 但仅回报更改的部分。

`-f` 或 `--quiet` 或 `--silent` 不显示错误信息。

`-h` 或 `--no-dereference` 只对符号连接的文件作修改, 而不更动其他任何相关文

件。

`-R` 或 `--recursive` 递归处理, 将指定目录下的所有文件及子目录一并处理。

`-v` 或 `--verbose` 显示指令执行过程。

`--help` 在线帮助。

`--reference=<参考文件或目录>` 把指定文件或目录的所属群组全部设成和参考文

件或目录的所属群组相同。

`--version` 显示版本信息。

Chown、chgroup 使用频率较低, 了解即可。

## 6) find 查找文件

Linux `find` 命令用来在指定目录下查找文件。任何位于参数之前的字符串都将被视为欲查找的目录名。

如果使用该命令时, 不设置任何参数, 则 `find` 命令将在当前目录下查找子目录与文件。并且将查找到的子目录和文件全部进行显示。

我们有时候还要模糊查找某个文件, 比如根据文件的后缀名, 文件创建时间, 文件大小等等。

语法:

```
find path -option [ -print ] [ -exec -ok command ] {} \;
```

`find` 根据下列规则判断 `path` 和 `expression`, 在命令列上第一个 `-()`, `!` 之前的部份为 `path`, 之后的是 `expression`。

如果 `path` 是空字符串则使用目前路径, 如果 `expression` 是空字符串则使用 `-print` 为预设 `expression`。

`expression` 中可使用的选项有二三十个之多, 在此只介绍最常用的部份。

参数说明:

`-name name`, `-iname name` : 文件名称符合 `name` 的文件。`iname` 会忽略大小写

`-user` 按照用户 (文件的属主)

`-size n` : 文件大小 是 `n` 单位, `b` 代表 512 位元组的区块, `c` 表示字元数, `k` 表示 kilo

bytes, `w` 是二个位元组。

`-mtime` 按照最后一次修改时间

`-atime` 按照最后一次访问时间

`-perm` : 按照文件的权限

`-type typen` : 查找文件类型为 `typen` 的文件

`c`: 字符设备

`d`: 目录

`c`: 字型装置文件

`b`: 区块装置文件

`p`: 具名贮列

`f`: 一般文件

`l`: 符号连结

`s`: socket

## 注意

find 的使用条件所查找的路径必须具有读权限。

查找选项通过文件属性来查找。

例如:

## 实例

- 1. 将当前目录及其子目录下所有文件后缀为 .c 的文件列出来:

```
# find . -name "*.c"
```

- 2. 将目前目录其下子目录中所有一般文件列出

```
# find . -type f
```

- 3. 将当前目录及其子目录下所有最近 20 天内更新过的文件列出:

```
# find . -ctime -20
```

- 4. 查找 /var/log 目录中更改时间在 7 日以前的普通文件, 并在删除之前询问它们:

```
# find /var/log -type f -mtime +7 -ok rm {} \;
```

- 5. 查找当前目录中文件属主具有读、写权限, 并且文件所属组的用户和其他用户具有读权限的文件:

```
# find . -type f -perm 644 -exec ls -l {} \;
```

- 6. 查找系统中所有文件长度为 0 的普通文件, 并列出现它们的完整路径:

```
# find / -type f -size 0 -exec ls -l {} \;
```

## 2. whereis

whereis 命令用于查找文件。

该指令会在特定目录中查找符合条件的文件。这些文件应属于原始代码、二进制文件, 或是帮助文件。

该指令只能用于查找二进制文件、源代码文件和 man 手册页, 一般文件的定位需使用 locate 命令。

## 语法

```
whereis [-bfmsu] [-B <目录>...] [-M <目录>...] [-S <目录>...] [文件...]
```

参数:

`-b` 只查找二进制文件。

`-B<目录>` 只在设置的目录下查找二进制文件。

`-f` 不显示文件名前的路径名称。

`-m` 只查找说明文件。

`-M<目录>` 只在设置的目录下查找说明文件。

`-s` 只查找原始代码文件。

`-s<目录>` 只在设置的目录下查找原始代码文件。

`-u` 查找不包含指定类型的文件。

实例:

- 1.使用指令"whereis"查看指令"bash"的位置, 输入如下命令:

```
peng@ubuntu:~/test$ whereis bash
```

```
bash: /bin/bash /etc/bash.bashrc /usr/share/man/man1/bash.1.gz
```

- 2.查找标准库头文件 stdio.h 位置

```
peng@ubuntu:~/test$ whereis stdio.h
```

```
stdio: /usr/include/stdio.h /usr/share/man/man3/stdio.3.gz
```



## 7) grep 过滤和统计

功能:

查出包含某些字符串的结果, 对文件或输出结果进行过滤, 对于大小写有一定 要求。

语法:

```
grep [option] string filename
```

补充说明:

grep 指令用于查找内容包含指定的范本样式的文件, 如果发现某文件的内容符合所指定的范本样式, 预设 grep 指令会把含有范本样式的那一行显示出来。若不指定任何文件名称, 或是所给予的文件名为 “-”, 则 grep 指令会从标准输入设备读取数据。

-A<显示行数>: 除了显示匹配 pattern 的那一行外, 显示该行之后的内容

-B<显示行数>: 除了显示匹配 pattern 的那一行外, 显示该行之前的内容

-C<显示行数>: 除了显示匹配 pattern 的那一行外, 显示该行前、后的内容

-c: 统计匹配的行数

-e: 同时匹配多个 pattern

-i: 忽略字符的大小写

-n: 显示匹配的行号

-o: 只显示匹配的字符串

-v: 显示没有匹配 pattern 的那一行, 相当于反向匹配

-w: 匹配整个单词

举例:

1、在当前目录中, 查找后缀有 file 字样的文件中包含 test 字符串的文件, 并打印出该字符串的行。此时, 可以使用如下命令:

```
grep test *file
```

2、以递归的方式查找符合条件的文件。

例如, 查找指定目录/etc/acpi 及其子目录 (如果存在子目录的话) 下所有文件中包含字符串"update"的文件, 并打印出该字符串所在行的内容, 使用的命令为:

```
grep -r update /etc/acpi
```

3、反向查找。前面各个例子是查找并打印出符合条件的行, 通过"-v"参数可以打印出不符合条件行的内容。

查找文件名中包含 test 的文件中不包含 test 的行, 此时, 使用的命令为:

```
grep -v test *test*
```

grep 应用非常频繁, 经常还会和正则表达式一起使用, 常用的正则表达式:

.: 任意单个字符

\*: 任意字符多次

[ ]: 指定范围, 如 [0-9]、[a-z]、[A-Z]、[0-9a-zA-Z]

^: 行首

\$: 行尾

^\$: 空行

举例

假定如下文件:

```
peng@ubuntu:~/test$ cat 123.c
hello world
world hello
```

```
peng@ubuntu:~/test$ grep -n "hello" 123.c
1:hello world
2:world hello
```

```
peng@ubuntu:~/test$ grep -n "^hello" 123.c
1:hello world
peng@ubuntu:~/test$ grep -n "hello$" 123.c
2:world hello
```

## 9) wc 计数

功能:

Linux wc 命令用于计算字数。

利用 wc 指令我们可以计算文件的 Byte 数、字数、或是列数, 若不指定文件名称、或是所给予的文件名为 "-", 则 wc 指令会从标准输入设备读取数据。

wc [选项] 文件名

- l 统计多少行
- w 统计多少单词
- c 统计多少个字符

语法

```
wc [-clw] [--help] [--version] [文件...]
```

参数:

-c 或 --bytes 或 --chars 只显示 Bytes 数。

-l 或 --lines 显示行数。

-w 或 --words 只显示字数。

```
--help 在线帮助。
```

```
--version 显示版本信息。
```

### 实例

在默认的情况下, `wc` 将计算指定文件的行数、字数, 以及字节数。使用的命令为:

```
wc testfile
```

## 8) tar 文件压缩解压

`tar` 命令可以为 linux 的文件和目录创建档案。利用 `tar` 命令, 可以把一大堆的文件和目录全部打包成一个文件, 这对于备份文件或将几个文件组合成为一个文件以便于网络传输是非常有用的。

### 语法

```
tar [必要参数] [选择参数] [文件]
```

### 常用命令参数:

```
-A 新增压缩文件到已存在的压缩
```

```
-B 设置区块大小
```

```
-c 建立新的压缩文件
```

```
-d 记录文件的差别
```

```
-r 添加文件到已经压缩的文件
```

```
-u 添加改变了和现有的文件到已经存在的压缩文件
```

`-x` 从压缩的文件中提取文件

`-t` 显示压缩文件的内容

`-z` 支持 gzip 解压文件

`-j` 支持 bzip2 解压文件

`-Z` 支持 compress 解压文件

`-v` 显示操作过程

`-l` 文件系统边界设置

`-k` 保留原有文件不覆盖

`-m` 保留文件不被覆盖

`-W` 确认压缩文件的正确性

## 常见解压/压缩命令

### tar

解包: `tar xvf FileName.tar`

打包: `tar cvf FileName.tar DirName`

(注: tar 是打包, 不是压缩!)

### .gz

```
解压 1: gunzip FileName.gz
```

```
解压 2: gzip -d FileName.gz
```

```
压缩: gzip FileName
```

## .tar.gz 和 .tgz

```
解压: tar zxvf FileName.tar.gz
```

```
压缩: tar zcvf FileName.tar.gz DirName
```

## .bz2

```
解压 1: bzip2 -d FileName.bz2
```

```
解压 2: bunzip2 FileName.bz2
```

```
压缩: bzip2 -z FileName
```

## .tar.bz2

```
解压: tar jxvf FileName.tar.bz2
```

```
压缩: tar jcvf FileName.tar.bz2 DirName
```

## .bz

```
解压 1: bzip2 -d FileName.bz
```

```
解压 2: bunzip2 FileName.bz
```

压缩: 未知

## .tar.bz

解压: `tar jxvf FileName.tar.bz`

压缩: 未知

## .Z

解压: `uncompress FileName.Z`

压缩: `compress FileName`

## .tar.Z

解压: `tar Zxvf FileName.tar.Z`

压缩: `tar Zcvf FileName.tar.Z DirName`

## .zip

解压: `unzip FileName.zip`

压缩: `zip FileName.zip DirName`

## .rar

解压: `rar x FileName.rar`

压缩: `rar a FileName.rar DirName`

## 举例:

实例 1: 将文件 log2021 全部打包成 tar 包

命令:

```
tar -cvf log.tar log2021.log
```

 仅打包, 不压缩!

```
tar -zcvf log.tar.gz log2021.log
```

 打包后, 以 gzip 压缩

```
tar -jcvf log.tar.bz2 log2021.log
```

 打包后, 以 bzip2 压缩

实例 2: 查阅上述 tar 包内有哪些文件

```
peng@ubuntu:~/test$ tar -ztvf log.tar.gz
-rw-rw-r-- peng/peng      14 2021-05-15 06:22 log2021.log
```

 说

明:

由于我们使用 gzip 压缩的 log.tar.gz, 所以要查阅 log.tar.gz 包内的文件时, 就得要加上 z 这个参数了。

实例 3: 将 tar 包解压缩

命令:

```
tar -zxvf /opt/soft/test/log.tar.gz
```

## 六、shell 的特殊字符

### 1) 通配符\* ?

\* :通配 0 个或多个字符

\* ? : 通配任意单个字符

\* [s] :通配某个范围内的任意一个字符



举例:

```
peng@ubuntu:~/test$ ls
123.c  menu.c  run  wait.c
peng@ubuntu:~/test$
peng@ubuntu:~/test$ ls *.c
123.c  menu.c  wait.c
peng@ubuntu:~/test$ ls ????.c
123.c

peng@ubuntu:~$ ls
123.c          ffmpeg          Public
basic          jdk             run
code          linux-imx       Templates
Desktop       linux-imx-rel_imx_4.1.15_2.1.0_ga_alientek.tar.xz test
Documents     module          toolchain
Downloads     Music           Videos
driver        nfsroot
examples.desktop Pictures
peng@ubuntu:~$
peng@ubuntu:~$
peng@ubuntu:~$ cd [r-w]est
peng@ubuntu:~/test$ pwd
/home/peng/test
```

## 2) 一行执行多条命令

一行执行多条命令: 在命令与命令之间用 “;” 隔开

```
cd ; ls
```

## 3) 输出重定向: >, >>

> : 将一个命令的输出放入文件中

>> : 输出重定向但不会把源文件覆盖, 在原文件末尾追加

举例:

将 ls 的输出结果输出给 test.txt

1. 当前目录没有操作权限  
cd .. 退回上一级目录

```
peng@ubuntu:~/test$ ls > test.txt
bash: test.txt: Permission denied
peng@ubuntu:~/test$
peng@ubuntu:~/test$ cd ..
```

## 2. 修改文件夹权限

为方便起见, 我们将 test 的权限全部打开

```
peng@ubuntu:~$ sudo chmod 777 test  
[sudo] password for peng:
```

用 ">" 的输出结果如下:

```
peng@ubuntu:~$ cd test/  
peng@ubuntu:~/test$ ls > test.txt  
peng@ubuntu:~/test$ cat test.txt  
123.c  
menu.c  
run  
test.txt  
wait.c
```

然后再用 ">>" 做测试其结果如下

```
peng@ubuntu:~/test$ ls >> test.txt  
peng@ubuntu:~/test$ cat test.txt  
123.c  
menu.c  
run  
test.txt  
wait.c  
123.c  
menu.c  
run  
test.txt  
wait.c
```

第一次重定向的信息

追加的信息

## 4) 输入重定向: <

下面再实现以下输入重定向:

```
peng@ubuntu:~/test$ cat < test.txt > newtest.txt
peng@ubuntu:~/test$ cat newtest.txt
123.c
menu.c
run
test.txt
wait.c
123.c      newtest2.txt里的内容
menu.c
run
test.txt
wait.c
```

## 5) 管道符: |

管道符 “|” : 将一个进程的输出作为另一个进程的输入

输入命令 :

```
ls -l /etc | cat
```

显示的结果如上图所示。

## 6) 其他: %, \$, ~

% : 作业控制, 提示符等

\$ : 取某一系列的值, 取变量值等

# 七、用户及进程

## 1、日期时间进程查看

a) date: 显示日期时间

```
peng@ubuntu:~/test$ date
Sat May 15 06:24:42 PDT 2021
```

b) cal : 显示日历

```
peng@ubuntu:~/test$ cal
      May 2021
Su Mo Tu We Th Fr Sa
                1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

## 2、ps 进程查询

功能:

ps (英文全拼: process status) 命令用于显示当前进程的状态, 类似于 windows 的任务管理器。

语法

```
ps [options] [--help]
```

ps 的参数非常多, 在此仅列出几个常用的参数并大略介绍含义:

-A 列出所有的进程

-w 显示加宽可以显示较多的资讯

-au 显示较详细的资讯

-aux 显示所有包含其他使用者的行程

au(x) 输出格式 :

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
```

USER: 行程拥有者

PID: pid

%CPU: 占用的 CPU 使用率

%MEM: 占用的记忆体使用率

VSZ: 占用的虚拟记忆体大小

RSS: 占用的记忆体大小

TTY: 终端的次要装置号码 (minor device number of `tty`)

STAT: 该行程的状态:

D: 无法中断的休眠状态 (通常 IO 的进程)

R: 正在执行中

S: 静止状态

T: 暂停执行

Z: 不存在但暂时无法消除

W: 没有足够的记忆体分页可分配

<: 高优先序的行程

N: 低优先序的行程

L: 有记忆体分页分配并锁在记忆体内 (实时系统或握 A I/O)

START: 行程开始时间

TIME: 执行的时间

COMMAND: 所执行的指令

## 实例

1. 显示所有进程信息, 连同命令行

```
peng@ubuntu:~/test$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1        0  0 May14 ?        00:00:04 /sbin/init auto noprompt
root           2        0  0 May14 ?        00:00:00 [kthreadd]
root           4        2  0 May14 ?        00:00:00 [kworker/0:0H]
root           6        2  0 May14 ?        00:00:00 [mm_percpu_wq]
root           7        2  0 May14 ?        00:00:01 [ksoftirqd/0]
root           8        2  0 May14 ?        00:00:01 [rcu_sched]
root           9        2  0 May14 ?        00:00:00 [rcu_bh]
root          10        2  0 May14 ?        00:00:00 [migration/0]
root          11        2  0 May14 ?        00:00:00 [watchdog/0]
root          12        2  0 May14 ?        00:00:00 [cpuhp/0]
root          13        2  0 May14 ?        00:00:00 [kdevtmpfs]
```

2. 查找指定进程 init:

```
peng@ubuntu:~/test$ ps -ef | grep init
root           1        0  0 May14 ?        00:00:04 /sbin/init auto noprompt
peng          11548     5757  0 06:28 pts/1    00:00:00 grep --color=auto init
```

### 3. 显示指定用户信息

```
peng@ubuntu:~/test$ ps -u peng
  PID TTY          TIME CMD
 1648 ?            00:00:00 systemd
 1649 ?            00:00:00 (sd-pam)
 1655 ?            00:00:00 gnome-keyring-d
 1709 ?            00:00:00 upstart
 1795 ?            00:00:00 upstart-udev-br
 1799 ?            00:00:00 dbus-daemon
 1811 ?            00:00:00 window-stack-br
 1839 ?            00:00:00 upstart-dbus-br
 1840 ?            00:00:00 upstart-dbus-br
 1853 ?            00:00:00 upstart-file-br
```

## 3、sudo 用户管理

终端的命令行最右边的字符

\$ 普通用户

# 管理员用户

有很多命令需要管理员权限才能使用, 可以输入命令前加 sudo, 也可以直接切换到管理员再执行。

切换到管理员 root

```
peng@ubuntu:~/test$ sudo su
[sudo] password for peng:
root@ubuntu:/home/peng/test#
```

切换用户

su 用户名 :切换账户

```
root@ubuntu:/home/peng/test# su peng
peng@ubuntu:~/test$
```

## 4、电源管理

## a) shutdown

安全关闭或重启 Linux 系统, 它在系统关闭之前给系统上的所有登陆用户提示一条警告信息。该命令还允许用户指定一个时间参数、可以是一个精确的时间、也可以是从现在开始的一段时间。

精确时间的格式: hh:mm 表示小时和分钟, 时间段由 + 和分钟数表示。系统执行该命令后会自动进行数据同步的工作。

功能说明:

系统关机指令。

```
语 法: shutdown [-efFhknr] [-t 秒数] [时间] [警告信息]
```

补充说明:

shutdown 指令可以关闭所有程序, 并依用户的需要, 进行重新开机或关机的动作。

参数:

-c 当执行“shutdown -h 11:50”指令时, 只要按+键就可以中断关机的指令。

-f 重新启动时不执行 fsck。

-F 重新启动时执行 fsck。

-h 将系统关机。

-k 只是送出信息给所有用户, 但不会实际关机。

-n 不调用 init 程序进行关机, 而由 shutdown 自己进行。

-r shutdown 之后重新启动。

-t<秒数> 送出警告信息和删除信息之间要延迟多少秒。

[时间] 设置多久时间后执行 shutdown 指令。



[警告信息] 要传送给所有登入用户的信息。

## b) reboot

功能说明:

重新开机。

语 法:

```
dreboot [-dfinw]
```

补充说明:

执行 reboot 指令可让系统停止运作, 并重新开机。

参数:

-d 重新开机时不把数据写入记录文件/var/tmp/wtmp。本参数具有“-n”参数的效果。

-f 强制重新开机, 不调用 shutdown 指令的功能。

-i 在重新开机之前, 先关闭所有网络界面。

-n 重新开机之前不检查是否有未结束的程序。

-w 仅做测试, 并不真的将系统重新开机, 只会把重新开机的数据写入/var/log 目录下的

wtmp 记录文件。

## c) halt

功能说明:

关闭系统。

语法:

```
halt [-dfinpw]
```

补充说明:

halt 会先检测系统的 runlevel。若 runlevel 为 0 或 6, 则关闭系统, 否则即调用 shutdown 来关闭系统。

参数:

-d 不要在 wtmp 中记录。

-f 不论目前的 runlevel 为何, 不调用 shutdown 即强制关闭系统。

-i 在 halt 之前, 关闭全部的网络界面。

-n halt 前, 不用先执行 sync。

-p halt 之后, 执行 poweroff。

-w 仅在 wtmp 中记录, 而不实际结束系统。

## 4、用户管理补充

### (1) 用户密码要求

用户的密码要求有 6~8 个字符, 其中至少要包含 2 个字母、1 个数字或特殊字符, 而且不能与用户名相同, 还要不同于以前的密码, 至少要有三个字符不同与以前的密码。

### (2) passwd 修改密码命令

输入命令 passwd

输入原密码 \*\*\*\*

输入新密码

确认新密码

注意: 在输入密码过程中机器是没有任何动作的

### (3) 查找用户

id

查看用户 ID (用户名) 及其所属组 ID (组名)

user

查看已经登陆到当前系统中的用户, 只显示出用户名。

who

查看用户的详细信息

who am i

查看当前用户自己的信息

whoami

查看当前用户自己的用户名

## 八、相关信息查询

查看磁盘信息

### 1. du

显示磁盘使用摘要信息

`du` 以 Block 为单位方式显示

`-k` 以 k 字节方式显示

`-m` 以 m 字节方式显示

`-s` 显示当前目录下的内容总的占用磁盘的大小, 以 Block 为单位

以 Block 单位显示的数字是以 k 字节方式显示的数字的 2 倍, 1k 字节=2 个 Block

## 2. df

显示整个文件系统的空间使用磁盘情况

`-k` 以 k 字节方式显示

# 九、网络配置

## 1、ping

查看当前机器与另一台机器的联通情况

ping 主机名称或者主机的 IP: 向 ping 后面的主机发送数据包, 若被 ping 的主机有回复则表示连通的。

语法:

```
ping [-dfnqrRv] [-c<完成次数>] [-i<间隔秒数>] [-I<网络界面>] [-l<前置载入>] [-p<范
```

```
本样式>] [-s<数据包大小>] [-t<存活数值>] [主机名称或 IP 地址]
```

补充说明:

执行 ping 指令会使用 ICMP 传输协议, 发出要求回应的信息, 若远端主机的网络功能没有问题, 就会回应该信息, 因而得知该主机运作正常。

参数:

-d 使用 Socket 的 SO\_DEBUG 功能。

-c<完成次数> 设置完成要求回应的次数。

-f 极限检测。

-i<间隔秒数> 指定收发信息的间隔时间。

-I<网络界面> 使用指定的网络界面送出数据包。

-l<前置载入> 设置在送出要求信息之前, 先行发出的数据包。

-n 只输出数值。

-p<范本样式> 设置填满数据包的范本样式。

-q 不显示指令执行过程, 开头和结尾的相关信息除外。

-r 忽略普通的 Routing Table, 直接将数据包送到远端主机上。

-R 记录路由过程。

-s<数据包大小> 设置数据包的大小。

-t<存活数值> 设置存活数值 TTL 的大小。

-v 详细显示指令的执行过程。

```
peng@ubuntu:~/test$ ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=2.44 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=64 time=5.01 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=64 time=2.23 ms
64 bytes from 192.168.0.1: icmp_seq=4 ttl=64 time=2.16 ms

peng@ubuntu:~/test$ ping baidu.com
PING baidu.com (39.156.69.79) 56(84) bytes of data.
64 bytes from 39.156.69.79: icmp_seq=1 ttl=50 time=41.4 ms
64 bytes from 39.156.69.79: icmp_seq=2 ttl=50 time=24.6 ms
64 bytes from 39.156.69.79: icmp_seq=3 ttl=50 time=29.9 ms
64 bytes from 39.156.69.79: icmp_seq=4 ttl=50 time=26.1 ms
```

## 2、ifconfig

查看和配置当前机器的网络参数信息

语 法:

```
ifconfig [网络设备] [down up -allmulti -arp -promisc] [add<地
```

```
址>] [del<地址>] [<hw<网络设备类型><硬件地址>] [io_addr<I/O 地址>] [irq<IRQ 地
```

```
址>] [media<网络媒介类型>] [mem_start<内存地址>] [metric<数目>] [mtu<字
```

```
节>] [netmask<子网掩码>] [tunnel<地址>] [-broadcast<地址>] [-pointopoint<
```

```
地址>] [IP 地址]
```

补充说明:

ifconfig 可设置网络设备的状态, 或是显示目前的设置。

参数:

add<地址> 设置网络设备 IPv6 的 IP 地址。

del<地址> 删除网络设备 IPv6 的 IP 地址。

down 关闭指定的网络设备。

<hw<网络设备类型><硬件地址> 设置网络设备的类型与硬件地址。

io\_addr<I/O 地址> 设置网络设备的 I/O 地址。

irq<IRQ 地址> 设置网络设备的 IRQ。

media<网络媒介类型> 设置网络设备的媒介类型。

mem\_start<内存地址> 设置网络设备在主内存所占用的起始地址。

metric<数目> 指定在计算数据包的转送次数时, 所要加上的数目。

mtu<字节> 设置网络设备的 MTU。

netmask<子网掩码> 设置网络设备的子网掩码。

tunnel<地址> 建立 IPv4 与 IPv6 之间的隧道通信地址。

up 启动指定的网络设备。

-broadcast<地址> 将要送往指定地址的数据包当成广播数据包来处理。

-pointopoint<地址> 与指定地址的网络设备建立直接连线, 此模式具有保密功能。

-promisc 关闭或启动指定网络设备的 promiscuous 模式。

[IP 地址] 指定网络设备的 IP 地址。

[网络设备] 指定网络设备的名称。

## 举例

```
ifconfig -a // 显示查看当前机器的 IP、Netmask、Gateway 等网络信息
```

```
ifconfig eth0 up(down) //激活与关闭某个网络适配器
```

```
ifconfig eth0 [ip address] netmask [address] //设置 IP 和子网掩码
```

```
peng@ubuntu:~/test$ sudo ifconfig ens33 down down掉以太网口ens33
peng@ubuntu:~/test$ ifconfig
lo                Link encap:Local Loopback
                  inet addr:127.0.0.1  Mask:255.0.0.0
                  inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING  MTU:65536  Metric:1
                  RX packets:494 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:494 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:36753 (36.7 KB)  TX bytes:36753 (36.7 KB)

peng@ubuntu:~/test$
peng@ubuntu:~/test$ sudo ifconfig ens33 up 启用以太网口ens33
peng@ubuntu:~/test$
peng@ubuntu:~/test$ ifconfig
ens33             Link encap:Ethernet  HWaddr 00:0c:29:bb:bd:40
                  inet addr:192.168.0.104 Bcast:192.168.0.255 Mask:255.255.255.0
                  inet6 addr: fe80::6abf:1256:56f4:c740/64 Scope:Link
                  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
                  RX packets:74576 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:13871 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:89261805 (89.2 MB)  TX bytes:1076805 (1.0 MB)

lo                Link encap:Local Loopback
                  inet addr:127.0.0.1  Mask:255.0.0.0
                  inet6 addr: ::1/128 Scope:Host
                  UP LOOPBACK RUNNING  MTU:65536  Metric:1
                  RX packets:530 errors:0 dropped:0 overruns:0 frame:0
                  TX packets:530 errors:0 dropped:0 overruns:0 carrier:0
                  collisions:0 txqueuelen:1000
                  RX bytes:38945 (38.9 KB)  TX bytes:38945 (38.9 KB)
```

# 十、Linux 应用程序的安装与卸载基础

## 1、Linux 安装包

Linux 安装包的通常是 tar 的格式, 同时也支持自己定义的格式。

在 redhat 中软件安装包的格式通常是 rpm

在 Ubuntu 中软件安装包的格式通常是 deb



## 2、安装包命名通用规则

在 Linux 中常用的命名格式是:

```
软件名称版本号-修订版本号体系架构.扩展名
```

## 3、安装包的离线安装及卸载

```
dpkg
```

```
dpkg -i 安装
```

```
dpkg -p 卸载
```

源文件安装的过程:

```
配置 configure >> 编译 make >> 安装 make install
```

## 4、在线安装及卸载

安装 :

```
apt-get install
```

卸载 :

```
apt-get remove -purge
```

# 十一、VIM 编译工具

Vim 是从 vi 发展出来的一个文本编辑器。代码补完、编译及错误跳转等方便编程的功能特别丰富, 在程序员中被广泛使用。

简单的来说, vi 是老式的字处理器, 不过功能已经很齐全了, 但是还是有可以进步的地方。vim 则可以说是程序开发者的一项很好用的工具。

连 vim 的官方网站 (<http://www.vim.org>) 自己也说 vim 是一个程序开发工具而不是文字处理软件。

## 1、vim 优势:

- a)所有 Unix Like 系统都会内置 vi 文本编辑器, 其他的文本编辑器则不一定会存在;
- b)很多软件的编辑接口都会主动调用 vi
- c)vi 具有程序编辑能力, 可以主动以字体颜色辨别语法的正确性, 方便程序设计;
- d)程序简单编辑速度快。

## 2、vi 的模式:

基本上 vi/vim 共分为三种模式, 分别是命令模式 (Command mode), 输入模式 (Insert mode) 和底线命令模式 (Last line mode)。这三种模式的作用分别是:

### 1) 命令模式:

用户刚刚启动 vi/vim, 便进入了命令模式。

此状态下敲击键盘动作会被 Vim 识别为命令, 而非输入字符。比如我们此时按下 i, 并不会输入一个字符, i 被当作了一个命令。

以下是常用的几个命令:

i 切换到输入模式, 以输入字符。

x 删除当前光标所在处的字符。

: 切换到底线命令模式, 以在最底一行输入命令。

若想要编辑文本: 启动 Vim, 进入了命令模式, 按下 i, 切换到输入模式。

命令模式只有一些最基本的命令, 因此仍要依靠底线命令模式输入更多命令。

### 2) 输入模式

在命令模式下按下 i 就进入了输入模式。

在输入模式中, 可以使用以下按键:

字符按键以及 Shift 组合, 输入字符

ENTER, 回车键, 换行

BACK SPACE, 退格键, 删除光标前一个字符

DEL, 删除键, 删除光标后一个字符

方向键, 在文本中移动光标

HOME/END, 移动光标到行首/行尾

Page Up/Page Down, 上/下翻页

Insert, 切换光标为输入/替换模式, 光标将变成竖线/下划线

ESC, 退出输入模式, 切换到命令模式

### 3) 底行命令模式

在命令模式下按下: (英文冒号) 就进入了底线命令模式。

底线命令模式可以输入单个或多个字符的命令, 可用的命令非常多。

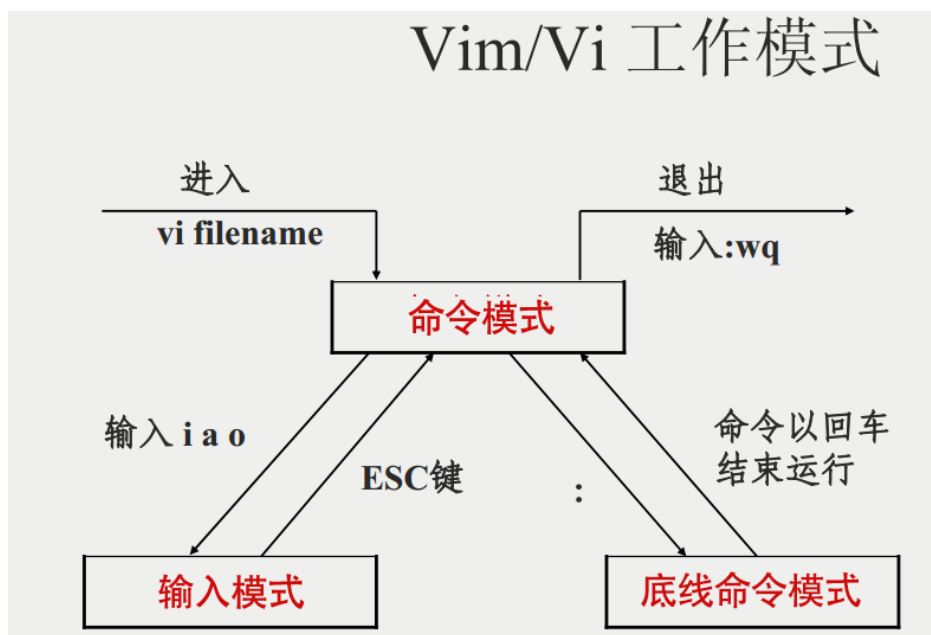
在底线命令模式中, 基本的命令有 (已经省略了冒号):

q 退出程序

w 保存文件

按 ESC 键可随时退出底线命令模式。

简单的说, 我们可以将这三个模式想成底下的图标来表示:



### 3. vim cheat sheet

下图就是赫赫有名的 vim cheat sheet, 一图包含了大部分常用的 vim 命令, 建议保存

version 1.1  
April 1st, 06  
翻译: 2006-5-21

vi / vim 键盘图

Esc  
命令模式

~ 转换大小写  
! 外部过滤器  
@ 运行宏  
# prev ident  
\$ 行尾  
% 括号匹配  
^ "软"行首  
& 重复:s  
\* next ident  
( 句首  
) 下一句首  
"soft" bol down  
+ 后一行行首  
\_ "硬"行首  
- 前一行行首  
= 自动格式化

Q 切换至 ex 模式  
q 录制宏

W 下一单词  
w 下一单词

E 词尾  
e 词尾

R 替换模式  
r 替换字符

T back 'till  
t 'till

Y 拷贝行  
y 拷贝

U 撤销行内命令  
u 撤销命令

I 到行首插入  
i 插入模式

O 分段(前)  
o 分段(后)

P 粘贴(前)  
p 粘贴(后)

{ 段首  
} 段尾

[ 杂项  
] 杂项

A 在行尾附加  
a 附加

S 删除行并插入  
s 删除字符并插入

D 删除至行尾  
d 删除

F 行内字符反向查找  
f 行内字符查找

G 文尾/行号  
g 附加命令

H 屏幕顶行  
h 左

J 合并两行  
j 下

K 帮助  
k 上

L 屏幕底行  
l 右

: ex 命令  
: 重复  
: u/T/t/E

" 寄存器标识  
' 跳转到标注的行  
\ 未用!

Z 退出  
Z 附加命令

X 退格  
X 删除(字符)

C 修改至行末  
c 修改

V 可视行模式  
v 可视模式

B 前一单词  
b 前一单词

N 查找上一处  
n 查找下一处

M 屏幕中间行  
m 设置标注

< 反缩进  
> 缩进

? 向前搜索  
/ 向后搜索

动作 移动光标, 或者定义操作的范围

命令 直接执行的命令, 红色命令进入编辑模式

操作 后面跟随表示操作范围的指令

extra 特殊功能, 需要额外的输入

q 后跟字符参数

w, e, b 命令  
小写(b): quux([foo], bar[, baz])  
大写(B): quux(foo, bar, baz)

主要 ex 命令:  
:w (保存), :q (退出), :q! (不保存退出)  
:e f (打开文件 f),  
:%s/x/y/g ('y' 全局替换 'x'),  
:h (帮助 in vim), :new (新建文件 in vim),

其它重要命令:  
CTRL-R: 重复 (vim),  
CTRL-F/-B: 上翻/下翻,  
CTRL-E/-Y: 上滚/下滚,  
CTRL-V: 块可视模式 (vim only)

可视模式:  
漫游后对选中的区域执行操作 (vim only)

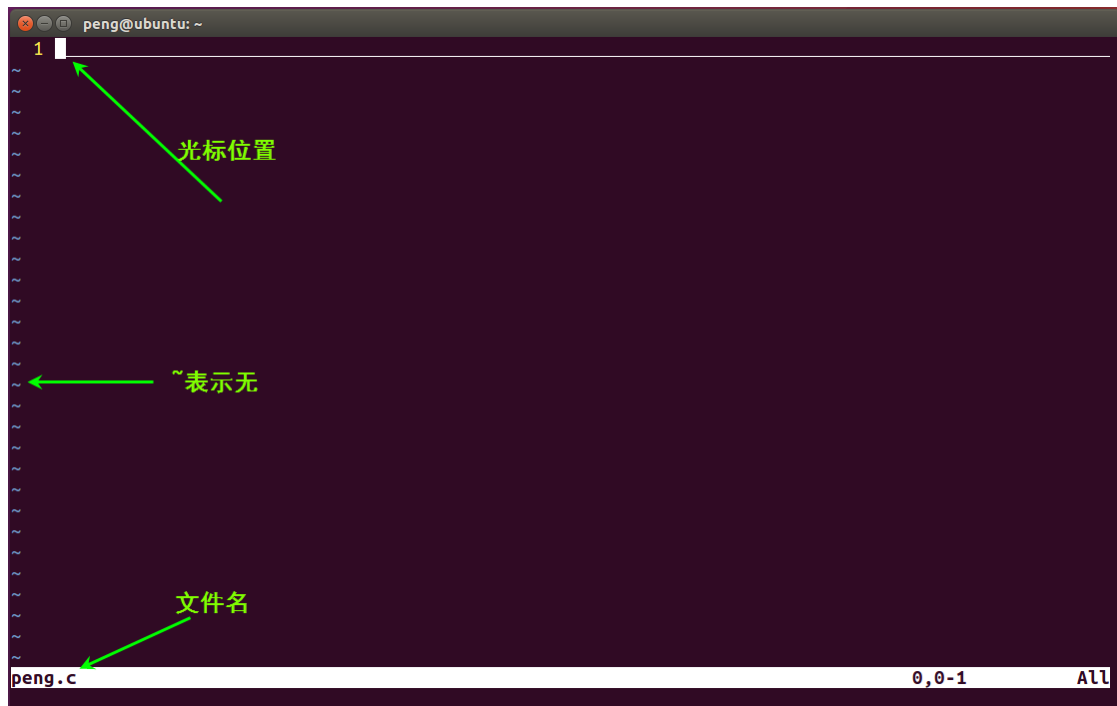
备注:  
(1) 在 拷贝/粘贴/删除 命令前使用 "x (x=a..z,\*) 使用命令的寄存器('剪贴板') (如: "ay\$ 拷贝剩余的行内容至寄存器 'a')  
(2) 命令前添加数字 多遍重复操作 (e.g.: 2p, d2w, 5i, d4j)  
(3) 重复本字符在光标所在行执行操作 (dd = 删除本行, >> = 行首缩进)  
(4) ZZ 保存退出, ZQ 不保存退出  
(5) zt: 移动光标所在行至屏幕顶端, zb: 底端, zz: 中间  
(6) gg: 文首 (vim only), gf: 打开光标处的文件名 (vim only)

原图: www.viemu.com 翻译: fdl (linuxsir)

### 4. 举例

如果你想要使用 vim 来建立一个名为 peng.c 的文件时, 输入下面命令:

```
peng@ubuntu:~$ vim peng.c
```

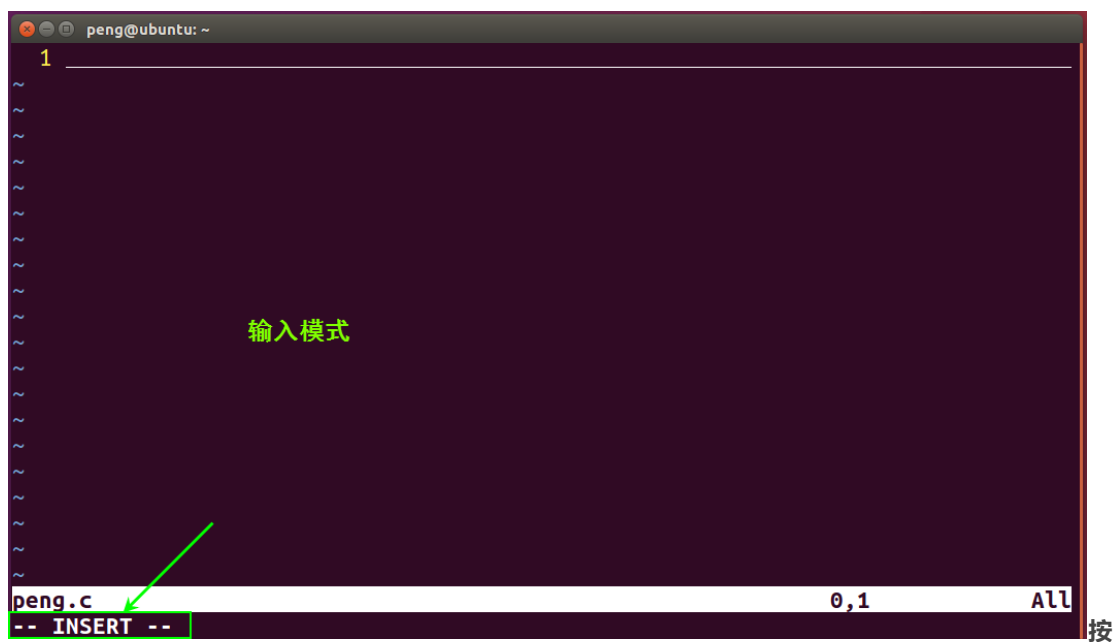


**按下 i 进入输入模式(也称为编辑模式), 开始编辑文字**

在一般模式之中, 只要按下 i, o, a 等字符就可以进入输入模式了!

在编辑模式当中, 你可以发现在左下角状态栏中会出现 -INSERT- 的字样, 那就是可以输入任意字符的提示。

这个时候, 键盘上除了 Esc 这个按键之外, 其他的按键都可以视作为一般的输入按钮了, 所以你可以进行任何的编辑。

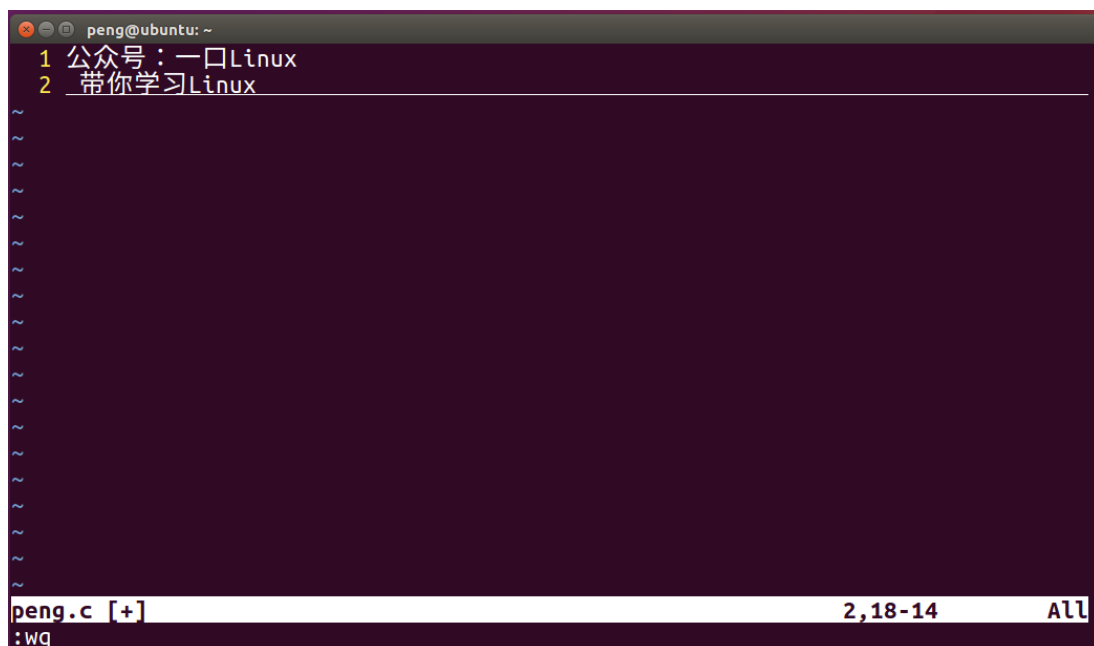


#### 按下 ESC 按钮回到一般模式

好了, 假设我已经按照上面的样式给他编辑完毕了, 那么应该要如何退出呢? 是的! 没错! 就是给他按下 Esc 这个按钮即可! 马上你就会发现画面左下角的 - INSERT - 不见了!

#### 在一般模式中按下 :wq 储存后离开 vi

OK, 我们要存档了, 存盘并离开的指令很简单, 输入 :wq 即可保存离开!



OK! 这样我们就成功创建了一个 peng.c 的文件。

## 5. 快捷键说明

### 移动光标的方法

操作	含义
h 或 向左箭头键(←)	光标向左移动一个字符
j 或 向下箭头键(↓)	光标向下移动一个字符
k 或 向上箭头键(↑)	光标向上移动一个字符
l 或 向右箭头键(→)	光标向右移动一个字符

如果你将右手放在键盘上的话, 你会发现 hjkl 是排列在一起的, 因此可以使用这四个按钮来移动光标。如果想要进行多次移动的话, 例如向下移动 30 行, 可以使用 "30j" 或 "30↓" 的组合按键, 亦即加上想要进行的次数(数字)后, 按下动作即可!

快捷键	含义
[Ctrl] + [f]	屏幕『向下』移动一页, 相当于 [Page Down]按键 (常用)
[Ctrl] + [b]	屏幕『向上』移动一页, 相当于 [Page Up] 按键 (常用)
[Ctrl] + [d]	屏幕『向下』移动半页
[Ctrl] + [u]	屏幕『向上』移动半页
+	光标移动到非空格符的下一行
-	光标移动到非空格符的上一行
n	那个 n 表示『数字』, 例如 20 。按下数字后再按空格键, 光标会向右移动这一行的 n 个字符。例如 20 则光标会向后面移动 20 个字符距离。
0 或 功能键 [Home]	这是数字『0』: 移动到这一行的最前面字符处 (常用)
\$ 或 功能键 [End]	移动到这一行的最后面字符处(常用)
H	光标移动到这个屏幕的最上方那一行的第一个字符
M	光标移动到这个屏幕的中央那一行的第一个字符
L	光标移动到这个屏幕的最下方那一行的第一个字符
G	移动到这个档案的最后一行(常用)
nG	n 为数字。移动到这个档案的第 n 行。例如 20G 则会移动到这个档

快捷键	含义
	案的第 20 行(可配合 :set nu)
gg	移动到这个档案的第一行, 相当于 1G 啊! (常用)
n	n 为数字。光标向下移动 n 行(常用)

## 搜索替换

快捷键	含义
/word	向光标之下寻找一个名称为 word 的字符串。例如要在档案内搜寻 vbird 这个字符串, 就输入 /vbird 即可! (常用)
?word	向光标之上寻找一个字符串名称为 word 的字符串。
n	这个 n 是英文按键。代表重复前一个搜寻的动作。举例来说, 如果刚刚我们执行 /vbird 去向下搜寻 vbird 这个字符串, 则按下 n 后, 会向下继续搜寻下一个名称为 vbird 的字符串。如果是执行 ?vbird 的话, 那么按下 n 则会向上继续搜寻名称为 vbird 的字符串!
N	这个 N 是英文按键。与 n 刚好相反, 为『反向』进行前一个搜寻动作。例如 /vbird 后, 按下 N 则表示『向上』搜寻 vbird。

使用 /word 配合 n 及 N 是非常有帮助的! 可以让你重复的找到一些你搜寻的关键词!

快捷键	含义
:n1,n2s/word1/word2/g	n1 与 n2 为数字。在第 n1 与 n2 行之间寻找 word1 这个字符串, 并将该字符串取代为 word2 ! 举例来说, 在 100 到 200 行之间搜寻 vbird 并取代为 VBIRD 则: 『:100,200s/vbird/VBIRD/g』。(常用)
:1,\$s/word1/word2/g 或 :%s/word1/word2/g	从第一行到最后一行寻找 word1 字符串, 并将该字符串取代为 word2 ! (常用)
:1,\$s/word1/word2/gc 或 :%s/word1/word2/gc	从第一行到最后一行寻找 word1 字符串, 并将该字符串取代为 word2 ! 且在取代前显示提示字符给用户确认 (confirm) 是否需要取代! (常用)

## 删除、复制与贴上

快捷键	含义
x, X	在一行字当中, x 为向后删除一个字符 (相当于 [del] 按键), X 为向前删



快捷键	含义
	除一个字符(相当于 [backspace] 亦即是退格键) (常用)
nx	n 为数字, 连续向后删除 n 个字符。举例来说, 我要连续删除 10 个字符, 『10x』。
dd	删除光标所在的那一整行(常用)
ndd	n 为数字。删除光标所在的向下 n 行, 例如 20dd 则是删除 20 行 (常用)
d1G	删除光标所在到第一行的所有数据
dG	删除光标所在到最后一行的所有数据
d\$	删除光标所在处, 到该行的最后一个字符
d0	那个是数字的 0, 删除光标所在处, 到该行的最前面一个字符
yy	复制光标所在的那一行(常用)
nyy	n 为数字。复制光标所在的向下 n 行, 例如 20yy 则是复制 20 行(常用)
y1G	复制光标所在行到第一行的所有数据
yG	复制光标所在行到最后一行的所有数据
y0	复制光标所在的那个字符到该行行首的所有数据
y\$	复制光标所在的那个字符到该行行尾的所有数据
p, P	p 为将已复制的数据在光标下一行贴上, P 则为贴在光标上一行! 举例来说, 我目前光标在第 20 行, 且已经复制了 10 行数据。则按下 p 后, 那 10 行数据会贴在原本的 20 行之后, 亦即由 21 行开始贴。但如果是按下 P 呢? 那么原本的第 20 行会被推到变成 30 行。 (常用)
J	将光标所在行与下一行的数据结合成同一行
c	重复删除多个数据, 例如向下删除 10 行, [10cj]
u	复原前一个动作。(常用)
.	重复前一个动作的意思。如果你想要重复删除、重复贴上等等动作, 按下小数点『.』就好了! (常用)
[Ctrl]+r	重做上一个动作。(常用)

这个 u 与 [Ctrl]+r 是很常用的指令! 一个是复原, 另一个则是重做一次~ 利用这两个功能按键, 你的编辑, 嘿嘿! 很快乐的啦!

## 一般模式切换到编辑模式的可用的按钮说明

进入输入或取代的编辑模式

快捷键	含义
i, I	进入输入模式(Insert mode): i 为『从目前光标所在处输入』, I 为『在目前所在行的第一个非空格符处开始输入』。(常用)
a, A	进入输入模式(Insert mode): a 为『从目前光标所在的下一个字符处开始输入』, A 为『从光标所在行的最后一个字符处开始输入』。(常用)
o, O	进入输入模式(Insert mode): 这是英文字母 o 的大小写。o 为在目前光标所在的下一行处输入新的一行; O 为在目前光标所在的上一行处输入新的一行! (常用)
r, R	进入取代模式(Replace mode): r 只会取代光标所在的那一个字符一次; R 会一直取代光标所在的文字, 直到按下 ESC 为止; (常用)
[Esc]	退出编辑模式, 回到一般模式中(常用)

上面这些按键中, 在 vi 画面的左下角处会出现『--INSERT--』或『--REPLACE--』的字样。由名称就知道该动作了吧!! 特别注意的是, 我们上面也提过了, 你想要在档案里面输入字符时, 一定要在左下角处看到 INSERT 或 REPLACE 才能输入。

## 一般模式切换到指令行模式的可用的按钮说明

指令行的储存、离开等指令

快捷键	含义
:w	将编辑的数据写入硬盘档案中(常用)
:w!	若文件属性为『只读』时, 强制写入该档案。不过, 到底能不能写入, 还是跟你对该档案的档案权限有关啊!
:q	离开 vi (常用)
:q!	若曾修改过档案, 又不想储存, 使用 ! 为强制离开不储存档案。
:wq	储存后离开, 若为 :wq! 则为强制储存后离开 (常用)

快捷键	含义
ZZ	这是大写的 Z 喔! 如果修改过, 保存当前文件, 然后退出! 效果等同于(保存并退出)
ZQ	不保存, 强制退出。效果等同于 :q!。
:w [filename]	将编辑的数据储存成另一个档案 (类似另存新档)
:r [filename]	在编辑的数据中, 读入另一个档案的数据。亦即将 『filename』 这个档案内容加到游标所在行后面
:n1,n2 w [filename]	将 n1 到 n2 的内容储存成 filename 这个档案。
:! command	暂时离开 vi 到指令行模式下执行 command 的显示结果! 例如: 『:! ls /home』 即可在 vi 当中察看 /home 底下以 ls 输出的档案信息!

#### vim 环境的变更

快捷键	含义
:set nu	显示行号, 设定之后, 会在每一行的前缀显示该行的行号(常用)
:set nonu	与 set nu 相反, 为取消行号!

这些基础知识学会以后, 就可以进入到下一步, 学习 Linux 下的 C 程序开发了!

想学习 Linux, 快关注一口 Linux 公众号吧!

欢迎关注公众号: 一口 Linux