# linux 基础

## Linux 基本原则

1. 由目的单一的小程序组成，组合小程序完成复杂的任务
2. 一起皆是文件
3. 尽量避免捕获用户接口
4. 配置文件保存为纯文本格式

## 人机交互接口

GUI接口： 点击icon

CLI接口： 命令提示符 prompt bash(shell)

管理员（ROOT）展现为：#

普通用户展现为：$

### 命令格式

命令（路径+命令） 选项 参数

**短选项** - 一个字符 ls -l

**长选项** – 一个词 ls --

Ls –a –l == ls –al

参数： 命令的作用对象

如果有多个参数 彼此用空格隔开。

### 使用凭证

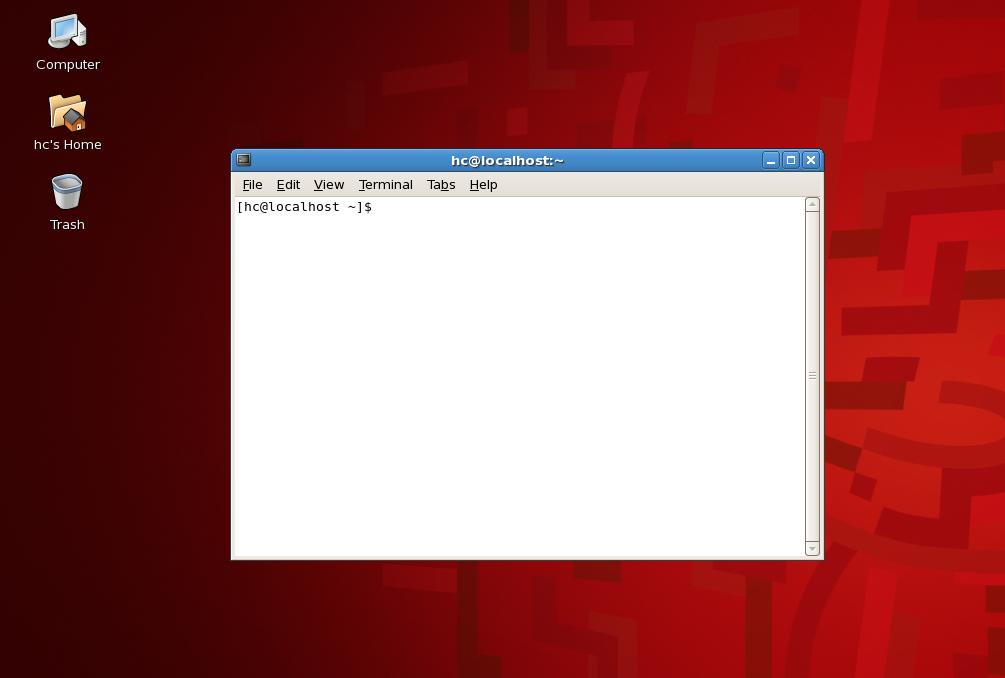
计算机是稀缺资源，又没法限定资源的使用者。如何验证用户身份。

### 虚拟终端

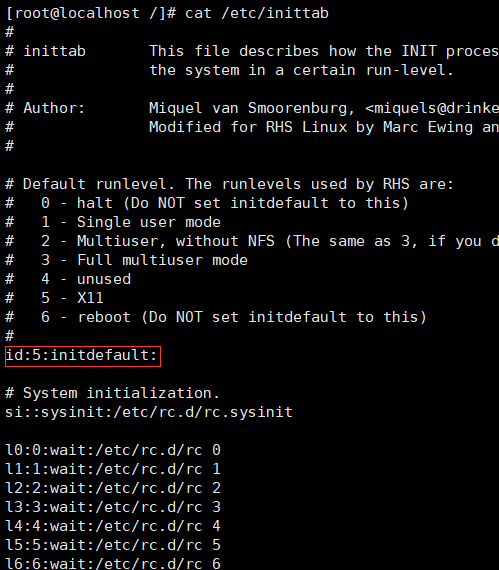
**ctr+alt+(F1~F6)** ： 多个虚拟终端的切换

**startx &**  : 进入图形界面终端（进入系统前没有启动图形终端）

**ctrl+alt+F7** : 进入图形界面终端（进入系统前启动了图形终端）



/etc/inittab



0 系统停机  
1 单用户模式  
2 多用户模式  
3 网络多用户模式   
4 保留  
5 X11模式（即进入图形界面模式）  
6 重起

如果不启动图形界面，改为3

上图的终端是虚拟终端。

Shell： 人机交互接口

Linux 图形界面的类型：**GUI**

x-window(CS架构)

Gnome: C 编写 redhat 默认安装使用Gnome

KDE（IDE）：C++编写 QT程序

Xface: 简介的 嵌入式桌面

命令行接口：**CLI**

sh （默认sh）

bash（功能最强，最常用，基于sh）

csh

zsh

ksh

tcsh

**su** : **switch user 切换用户**

**su 用户名 半切换**

su -l 用户名 或者 su - 用户名 完全切换

**exit ：返回切换**

[hc@localhost ~]$ su root

Password:

[root@localhost hc]# su -l root

[root@localhost ~]# exit

[root@localhost hc]# exit

exit

[hc@localhost ~]$

### 终端管理

#### 关机

**shutdown -h now**

#### passwd

修改密码

符合密码的复杂度规则：

1. 足够长【7位以上】，
2. 足够复杂【4类字符用3种】，
3. 使用随机字符串，
4. 定期更换，
5. 循环周期足够大

否则

BAD PASSWORD: it is too short 这个密码不咋地

Retype new UNIX password:

Sorry, passwords do not match. 不匹配

Lib:

Windows : .dll dynamic link library 动态链接库

Linux : .so share Object 共享对象

认证机制： Authentication

授 权： Authorization

审 计： Audition（日志）

Prompt: 命令提示符

命令： 开头几个字符被称之为magic number 标志程序入口

#!/bin/bash #!被称为魔数 也称为 shebang

#### command

command option… arguments…

选项： 修改命令执行的特性 ，有些选项是可以带参数

短选项

长选项

参数： 指定命令的作用对象 多个参数用空格隔开

#### ls

ls : list 列出，列表，列出指定目录下的子目录或者指定路径下的目录和文件，默认当前目录下的文件和目录

目录：也是文件，是路径映射

路径 : 从指定的起始点到目的地所经过的位置。层次化管理文件的机制。最佳实践。而层次化管理文件的机制就叫做文件系统 File system

树形目录：



这是一种绝对路径

相对目录：

当前目录： 也称工作目录，登入系统一定会处在某个目录下 FHS

Pwd: printing working directory 打印显示正在工作的目录叫

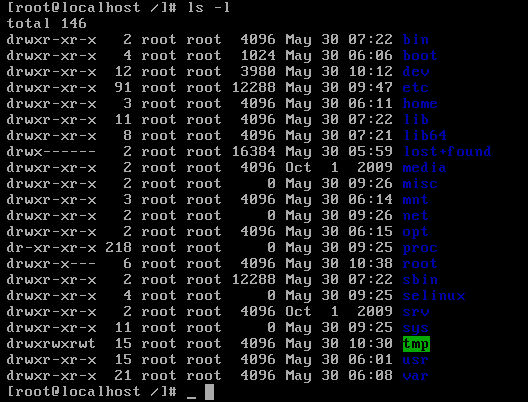
[root@localhost /]# pwd

/

[root@localhost usr]# pwd

/usr

最顶层目录： /



[root@localhost /]# ls -l

total 146

drwxr-xr-x 2 root root 4096 May 30 07:22 bin

drwxr-xr-x 4 root root 1024 May 30 06:06 boot

drwxr-xr-x 12 root root 3980 May 30 10:12 dev

drwxr-xr-x 91 root root 12288 May 30 09:47 etc

drwxr-xr-x 3 root root 4096 May 30 06:11 home

drwxr-xr-x 11 root root 4096 May 30 07:22 lib

drwxr-xr-x 8 root root 4096 May 30 07:21 lib64

-l : 长格式 显示文件的详细信息

1.文件类型：看开头的字符

drwxr-xr-x

-rwxr-xr-x

-：普通文件

**d** : 目录文件 （directory）

**b** : 块设备文件 （block）

**c** : 字符设备文件 (character)

**i** : 符号链接文件（symbolic link file） 软链接文件

**p** : 管道文件 （pipe）

**s** : 套接字文件 (socket)

2.后面的9个位表示文件的权限： 每3位一个组，每一个组: rwx (读 写 执行)

没有权限用 – 表示

如： rw- 能读 能写 不能执行

r-- 能都 不能写 不能执行

-wx 不能读 能写 能执行

,,,,,,,,

3. drwxr-xr-x 21 root root 4096 May 30 06:08 var

21： 文件被硬链接的次数

root : 文件的属主(owner) 文件的属于哪个用户

root : 文件的属组(group) 文件的属于哪个用户组

4096 : 文件的大小，默认单位字节

May 30 06:08 var ： 时间戳 timestamp 最近一次被修改的时间

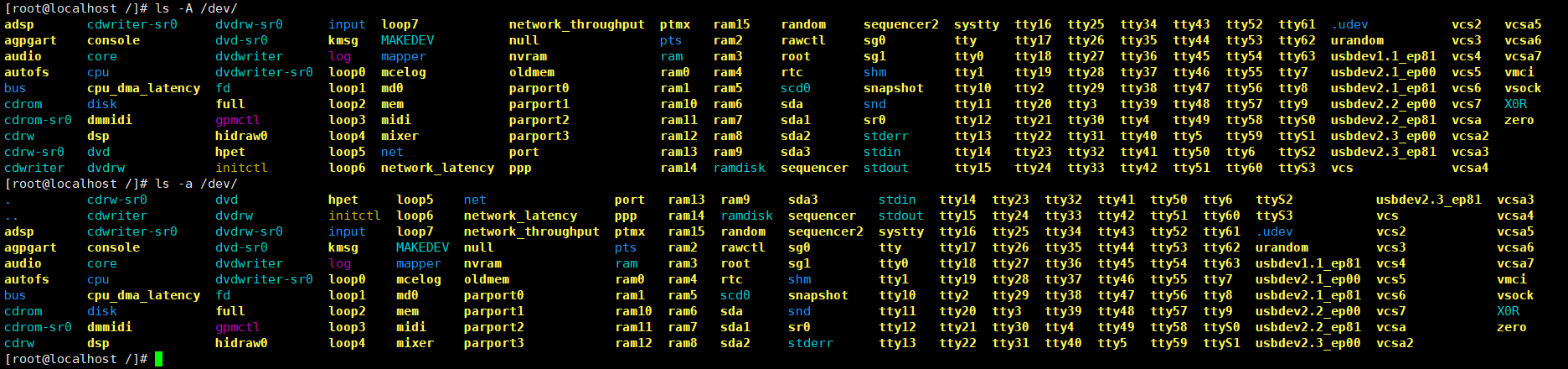
**–h** : 单位换算文件大小 4096 2k

**-a:** all 显示 .开头的文件 显示所有文件

. : 表示当前目录

.. : 表示父级目录

**-A**： 表示显示所有隐藏文件，不包括 . 和 ..



**-d**: 显示目录自身属性

[root@localhost /]# ls -ld /root

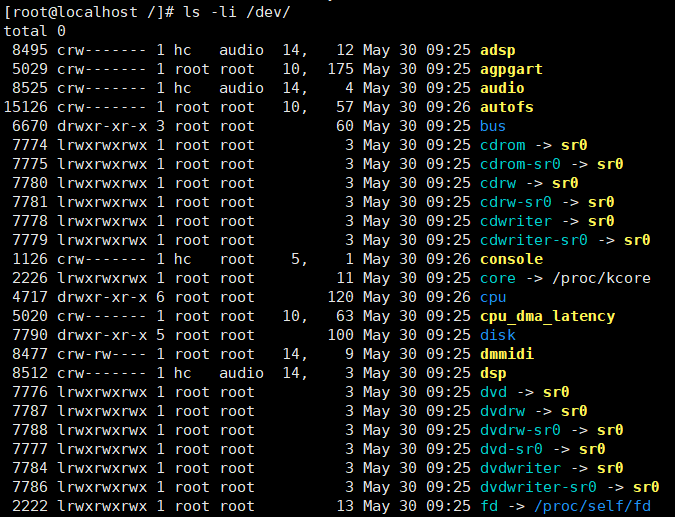
drwxr-x--- 6 root root 4096 May 30 14:49 /root

[root@localhost /]# ls -ldh /root

drwxr-x--- 6 root root 4.0K May 30 14:49 /root

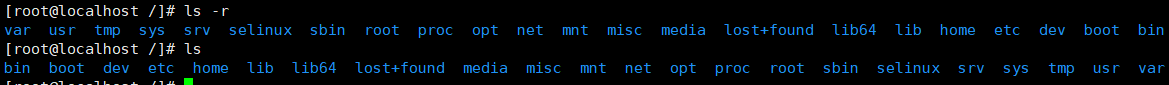
我们记住文件名称而系统则是对应的数字，所以

**-i** ： index node 索引节点



root文件夹 系统编号 6670

**-r** ： 按照名称字母倒序排列文件



**-R**：(recursive) 递归显示 也要显示子目录或子目录中的文件

#### cd

**cd**: 切换目录

cd : 回到家目录（~） home directory 用户有所有操作权限

[root@localhost etc]# cd

[root@localhost ~]#

[root@localhost ~]# cd ~hc 到hc的家目录中去，当然只有管理员有权限去别人家。

[root@localhost hc]#

cd - : 在当前一个目录和前一个目录来回切换。

[root@localhost ~]# cd -

/home/hc

[root@localhost hc]# cd -

/root

[root@localhost ~]# cd -

/home/hc

[root@localhost hc]# cd -

/root

[root@localhost ~]# cd -

/home/hc

[root@localhost hc]#

#### type

**查看命令类型**，Linux命令类型有两种，一种内置命令(shell内置),一种外置命令，

type可以显示属于哪种命令

[root@localhost hc]# type cd

cd is a shell builtin 表示cd 是shell内置命令或者叫做内建命令

[root@localhost hc]# type su

su is /bin/su su 是外部命令，在bin/su 对应了一个与命令名称相对应的可执行文件

[root@localhost hc]# type ls

ls is aliased to `ls --color=tty'

[root@localhost hc]# type ifconfig

ifconfig is /sbin/ifconfig

外部命令和内部命令的区别：

外部命令在文件系统的某个目录下存在一个与之名称对应的可执行文件。内部命令是shell自带的为了完成自我管理而存在。

[hc@localhost /]$ **which ls** 在哪个路径下 /bin/ls

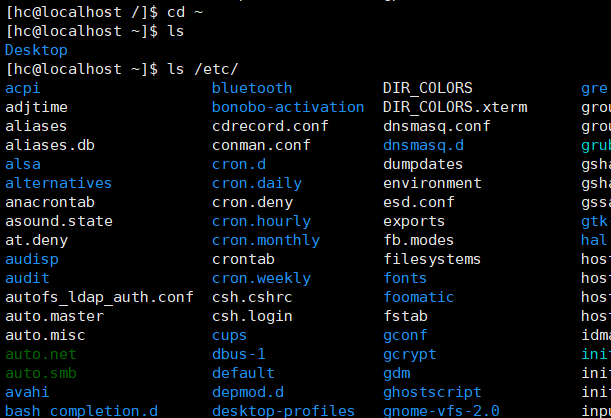
alias ls='ls --color=tty'

/bin/ls

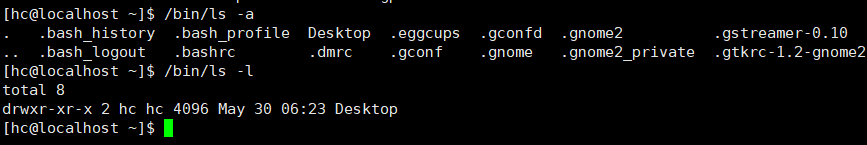
[hc@localhost /]$ type ls

ls is aliased to `ls --color=tty'

那么问题来了：外部命令 ls 在 /bin/ls ，为什么用户在home目录下可以使用ls ?



如果像这样



执行一个命令都要带上一个路径，那是很繁琐的一件事。

所有就有环境变量这个说法。将ls配置在环境变量中，不用带路径就可以使用了。

环境变量： 命名的内存空间。定义当前用户的工作空间的属性。

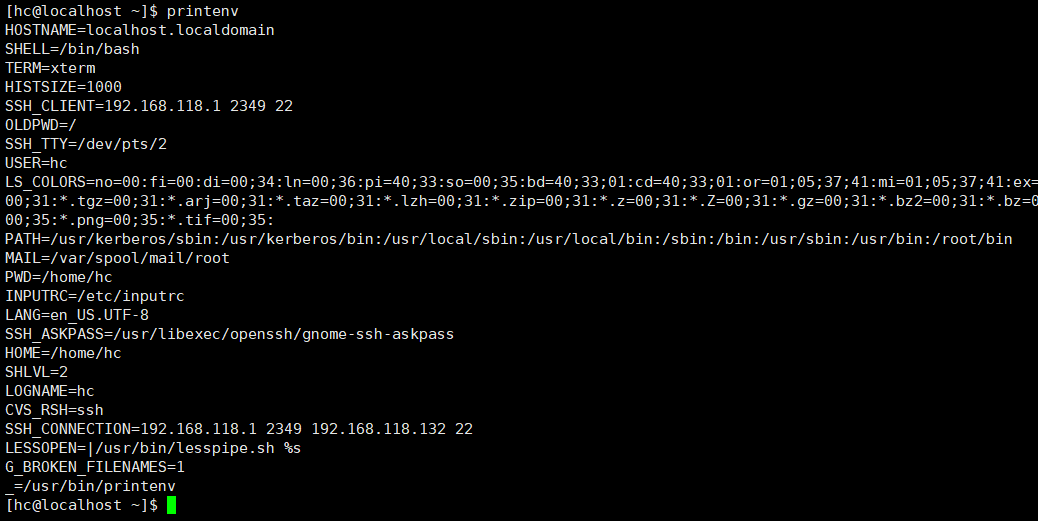
#### clear

clear 清屏

#### printenv

打印显示当前用户工作空间的环境变量

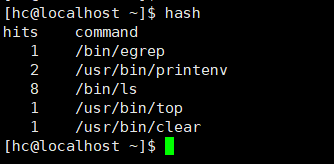
**env** : environment（环境） printenv就是打印环境变量



PATH=/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin

ls 就是逐个从path当中去找。直到找到为止，实际上bash使用了缓存。用过一次的命令缓存下来就不用去找。如命令 hash 就可以看到哪些命令被缓存下来了。

Hits 命中，即使用了缓存中的命令命中了多少次了



计算机中发展史上,缓存被称为 缓存为王 cache is king

hash查找速度：O(1)

#### date

显示系统的时间:

date： 不但可以显示时间，还可以修改时间。 用于时间管理

[hc@localhost ~]$ date

Wed May 31 05:21:27 PDT 2017

[hc@localhost mydictory]$ date +%F

2017-06-04

[hc@localhost mydictory]$ date +%F-%H

2017-06-04-02

[hc@localhost mydictory]$ date +%F-%H-%M

2017-06-04-02-45

[hc@localhost mydictory]$ date +%F-%H-%M-%s

2017-06-04-02-46-1496569561

[hc@localhost mydictory]$ date +%F-%H-%M-%S

2017-06-04-02-46-06

[hc@localhost mydictory]$ date +%F" "%T

2017-06-04 03:23:44

[hc@localhost ~]$

Linux 有两个时间： **硬件时钟 ，系统时钟**

系统时间是对硬件时间的模拟。

[hc@localhost ~]$ date 系统时间

Wed May 31 05:59:57 PDT 2017

[hc@localhost ~]$ clock 硬件时间

Wed 31 May 2017 06:00:57 AM PDT -0.334384 seconds

[hc@localhost ~]$ hwclock 硬件时间 PM后半天 AM前半天

Wed 31 May 2017 06:01:07 AM PDT -0.398613 seconds

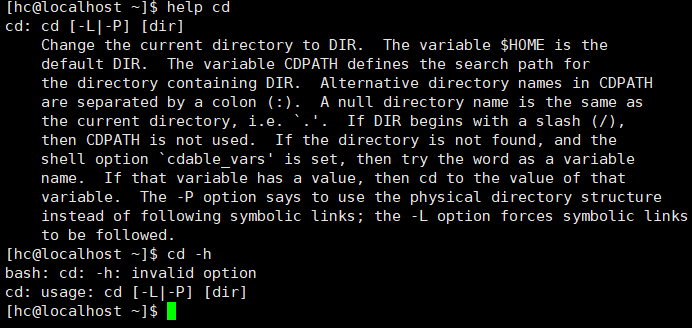
clock or hwclock (hard ware clock)

系统时间和硬件时间 都可以修改。

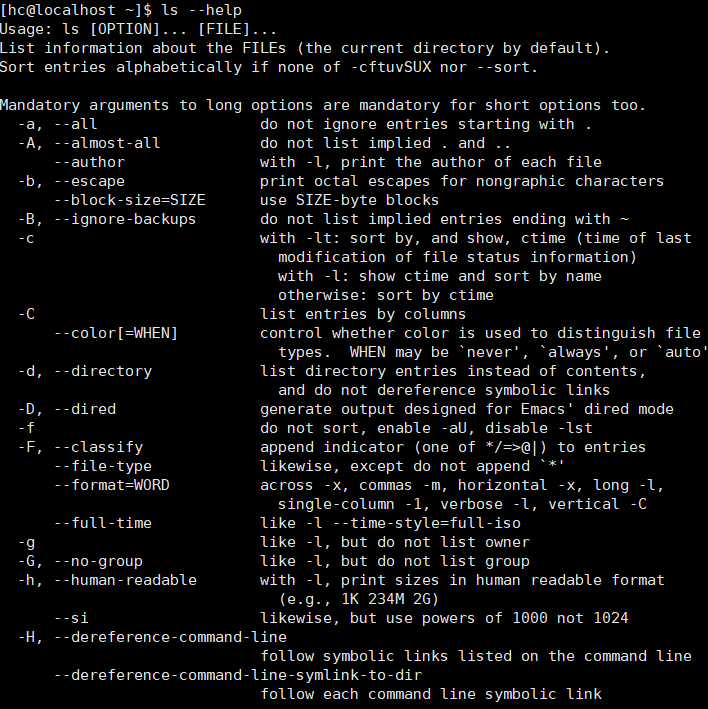
#### man

获得命令的使用帮助：

1. 内部命令： **help command**  or **command -h**



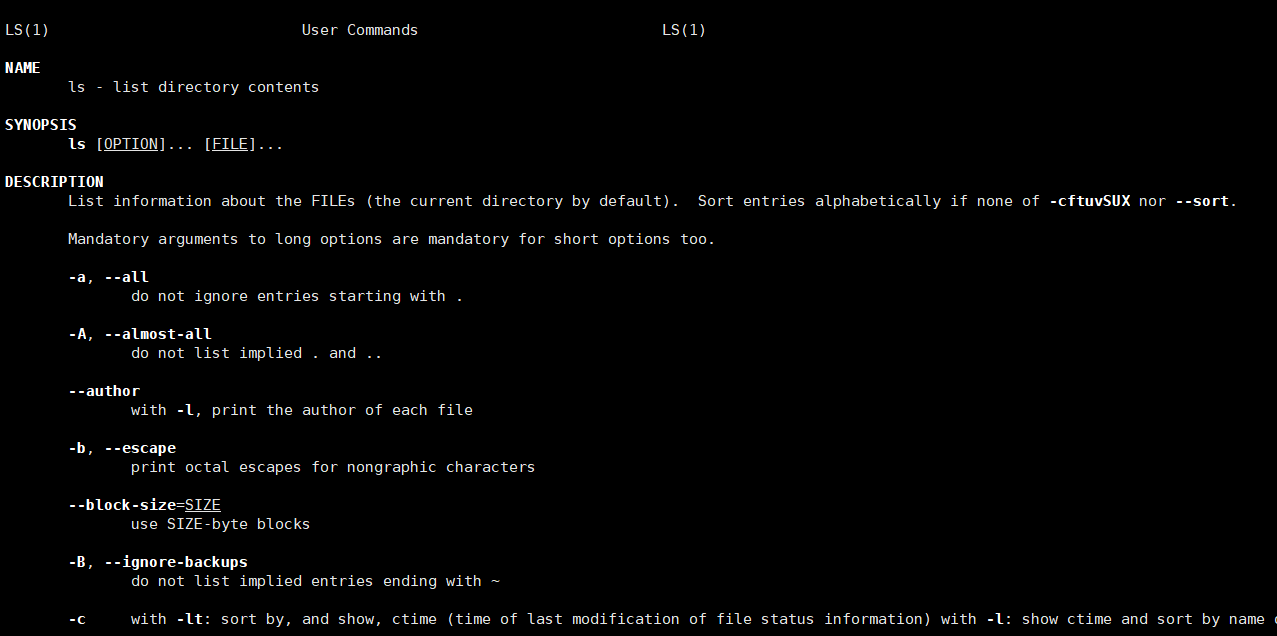
1. 外部命令： **command --help**



外部还内部命令都有一个公告的查询手册：

manual ： 手册(在Linux中是压缩文档)

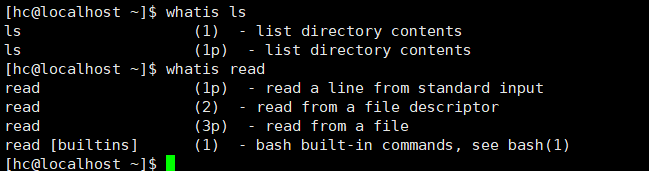
**man command** : man ls



ls(1) 表示分章节 如： man 1 ls

那怎么知道有多少章节呢？

使用命令：whatis command 可以显示命令的摘要信息



那么章节怎么分类：

第一章节： 介绍用户命令(/bin, /usr/bin, /usr/local/bin) 普通用户和管理员都可以用

第二章节： 介绍系统调用

第三章节： 介绍库调用 ，不是所有调用都有库调用，只有库才有库调用

第四章节： 介绍特殊文件（设备文件，块设备，字符文件，）没有存储数据，只是程序访问硬件资源的一个入口。如man 4 tty 介绍用户控制终端的



/dev/tty is a character file /dev/tty 是一个字符文件

第五章节： 介绍文件格式，配置文件的语法

第六章节： 游戏

第七章节： 杂项(miscellaneous)

第八章节： 管理命令（/sbin , /usr/sbin, /usr/local/sbin, ）通常只有管理员可以使用

[ ] 可省略

<> 必选

… 可出现多次

|: 多选一

{}： 分组

man:

Name : 名称及功能说明

SYNOPSIS:用法说明，包括可用选项

Description：命令功能的详尽说明，可能包括每一个选项的意义

Options: 说明每一个选项的意思

Files: 此命令的配置文件

Bugs:

Example： 使用示例

See also: 另外参见

**翻屏**：

向后翻一页： space 空格

向前翻一页： b

向后翻一行： enter 回车

向前翻一行： k

查找功能： 不区分大小写

/关键字 先后

n : 下一个

N： 前一个

?关键字 向前

n : 下一个

N： 前一个

退出: 使用**q**

[hc@localhost ~]$ date +%y

17

[hc@localhost ~]$ date +"This year is %Y"

This year is 2017

[hc@localhost ~]$ date +"this year is %y,this day is %d."

this year is 17,this day is 31.

[hc@localhost ~]$ date +"this year is %y,%nthis day is %d."

this year is 17,

this day is 31.

[hc@localhost ~]$

[hc@localhost ~]$ date +%F" "%T

2017-05-31 10:23:44

[hc@localhost ~]$

%s 格式: UTC Unix元年计时法 格林尼治时间 （1970-01-01 00：00：00）开始

太阳经过本初子午线的那一刻，开始偏移计时。很精确。所有那些跨国的订单要记录数据库，使用时间就必须使用格林尼治时间(时间戳)，在换算成当地时间显示

[hc@localhost ~]$ date +%s

1496253678

[hc@localhost ~]$ date

Wed May 31 11:02:22 PDT 2017

[hc@localhost ~]$ clock

Wed 31 May 2017 11:03:56 AM PDT -0.931504 seconds

#### hwclock

hardware clock

**hwclock –w** 将硬件时间改为系统时间

[root@localhost /]# hwclock -w

[root@localhost /]# date

Wed May 31 11:05:38 PDT 2017

[root@localhost /]# date

Wed May 31 11:05:42 PDT 2017

**hwclock –s**将系统时间改为硬件时间

#### clock

在线文档：(更多的是命令的历史)

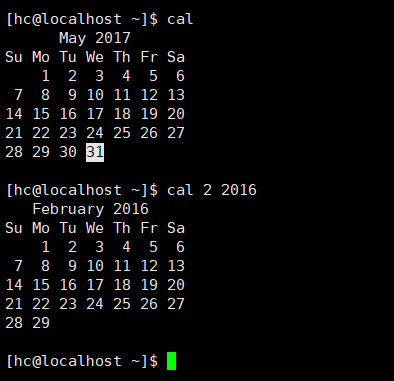
Info command

/usr/share/doc 目录下也有文档

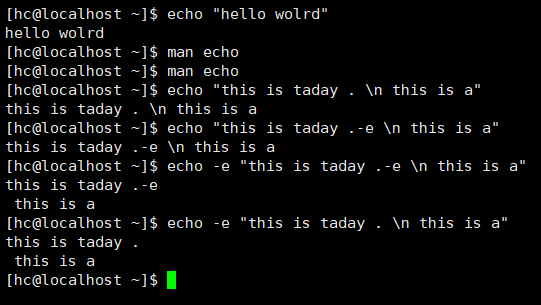
Google 百度

#### cal

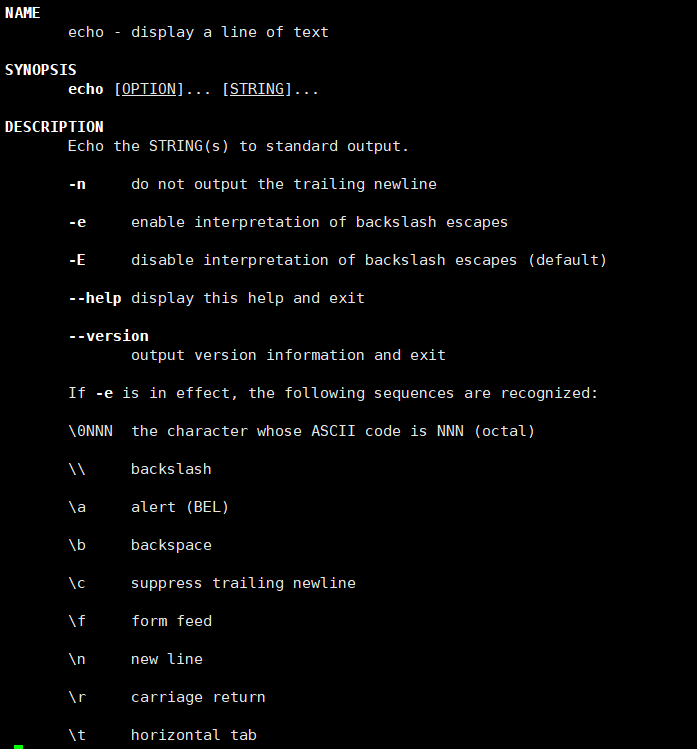
calendar (日历)



#### echo



显示一行文本



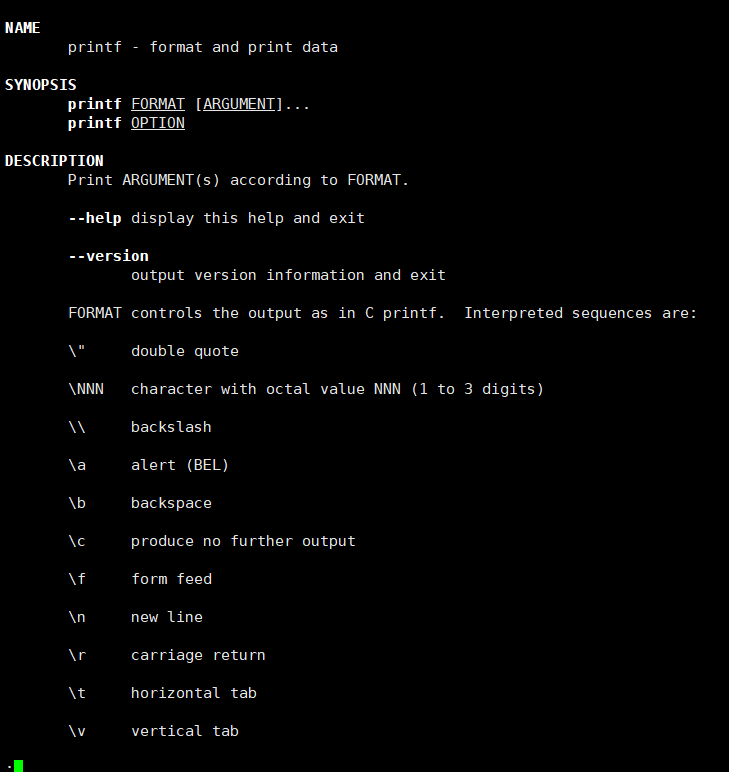
\a 转义

-e（escapes） 逃义

写脚本的时候会使用到这些命令

#### printf

转换和打印数据



[hc@localhost ~]$ printf "the year is 2017.\n"

the year is 2017.

[hc@localhost ~]$ printf "the year is 2017"

the year is 2017[hc@localhost ~]$

**printf** 和 **echo** 的区别： 如果需要格式化 printf直接跟上转义，echo不需要加逃义