

# Day 1 Linux入门

---

## 1. Unix, Minix ->Linux

---

内核语言由汇编语言编写转为高级语言编写（C language），增强unix多平台可移植性

DOS：单用户

MacOS：只有Mac能用

Linux：开源、简单的操作系统，便于操作系统的学习和开发，用户可自由定制，有很多发行版

## 2. Linux

---

1. 一切皆文件

2. 多用户、多任务

3. 交互性

4. 网络、嵌入式、多平台...

## 3. Shell

---

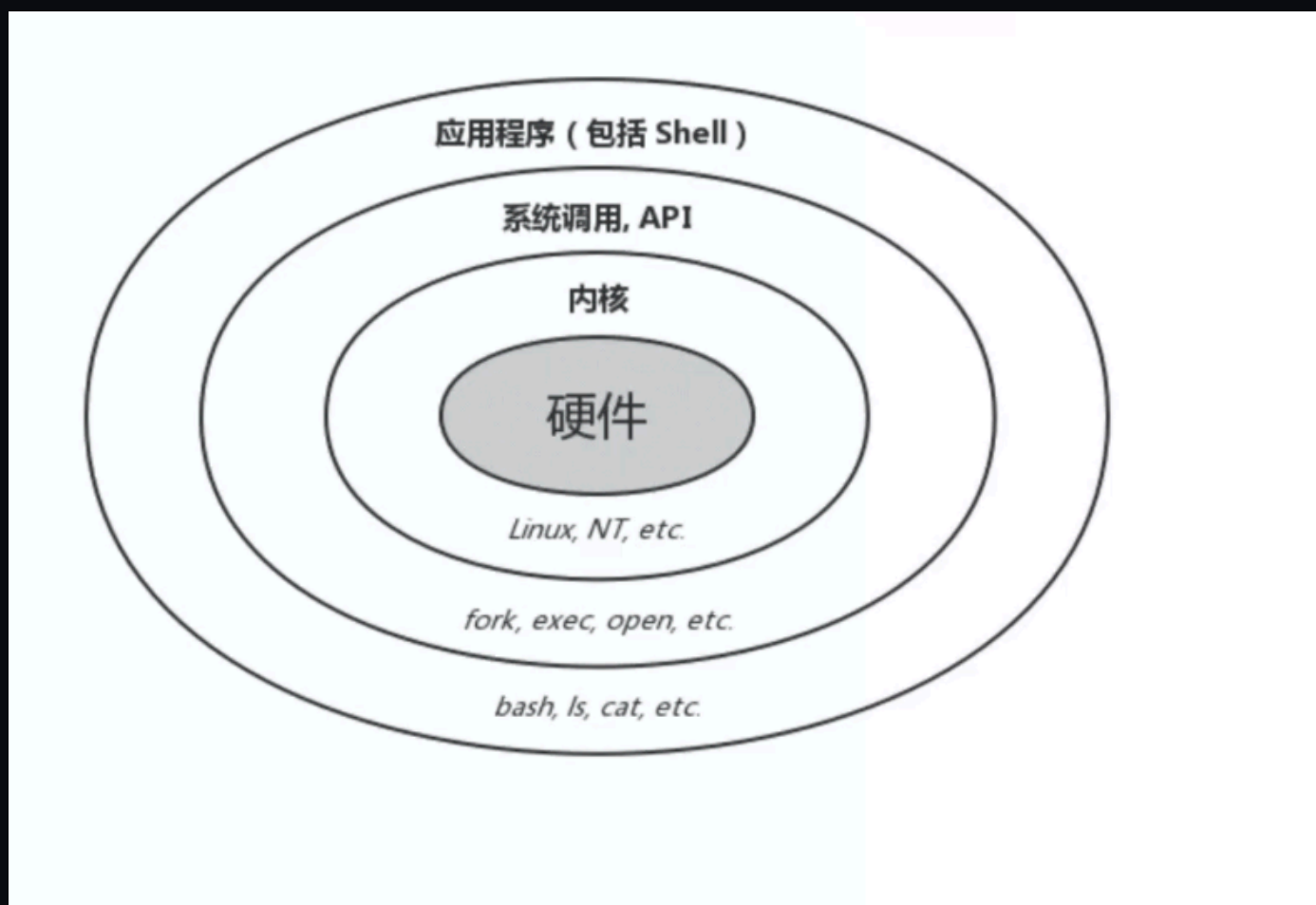
1. 命令行（CLI）：文本交互界面

2. 终端（terminal）：人与计算机交互的接口

3. 控制台（console）：系统管理员的终端

4. TDY

5. Shell：用户操作系统的入口



分为CLI和GUI形式

## 4. Linux用户管理

---

## 1.新建用户：

```
1 sudo useradd [options] [username]
2 #-d: 可指定用户主目录，若不存在则使用-m -d, 可创建主目录
3 #-g: 指定用户所属的用户组
4 #-G: 指定用户所属的用户组
5 #-s: 指定用户的登录shell
6 #这一命令需要超级用户权限。sudo以超级用户权限执行，su以root（超级用户）执行
```

## 2. 切换用户

```
1 su [username]
2 #直接切换时账户锁定
```

### 修改密码：

```
1 passwd [options]
2 #-l: 锁定密码
3 #-u: 解锁密码
4 #-d: 删除密码，无法登陆
5 #-f: 强迫用户下次登录时修改密码
```

### 修改用户：

```
1 sudo usermod [options] [username]
```

### 查看全部用户全部属性：

```
1 cat /etc/passwd
```

### 赋予用户超级用户权限：

```
1 sudo visudo
2 #####
3 root (用户名) ALL (允许登录的主机) =(ALL:ALL) (可使用的身份 用户: 用户组) ALL (授权命令, 绝对路径)
4 %wheel (组名) ALL (被管理的主机地址) =(ALL:ALL) ALL
```

或者把用户添加的有超级用户权限的组: root, sudo, ...

删除用户:

```
1 sudo userdel [options] [username]
2 #-r: 同时删除用户主目录
```

修改用户组:

```
1 sudo groupmod -n [new groupname] [old groupname]
```

切换用户组:

```
1 newgrp [groupname]
```

(\*) 查看目录下所有文件的权限:

```
1 ll
2 # =ls -al
3 #显示: 文件类型 (-普通文件 d文件夹 r可读 w可写 x可执行, 3个一组, 依次为用户权限、群组权限和其他人的权限) — 硬连接数量 — 用户 (拥有者) — 组名 — 大小 — 修改日期
```

修改文件权限:

```
1 chmod [-R] [u/g/o/a] [+/-/=] [r/w/x] [file]
2 #可以直接改成777
```

## 修改文件属主和属组

```
1 sudo chown [-R] [owner] [file]
2 sudo chown [-R] [owner]:[group] [file]
3 #必须超级用户权限执行
```

单独修改属组：

```
1 sudo chgrp [-R] [group] [file]
```

## 5.文件管理

### 1. 绝对路径&相对路径

### 2. tree

```
1 tree -L 2 /
```

### 3. 列出文件系统的整体磁盘用量

```
1 df [options] [file or directory path]
2 du [options] [file or directory path]
3 #options -h 以人们易阅读的GB,MB,KB等格式自行显示
4 #          -a 列出所有文件系统
5 #          -k 以KB为单位显示
6 #          -m 以MB为单位显示
7 #          -s 列出总量, 不列占用量(du)
8 #          -S 不包括子目录下的总计(du)
```

## 操作Linux磁盘分区表

```
1 sudo fdisk [option] [device name]
2 option -l 可以把整个系统内能够搜到的分区都列出来
```

## 磁盘格式化、挂载与卸载

```
1 mkfs [option] [device name]
2 option -t [filesystem]
3 #将指定的磁盘格式化未指定文件系统格式
```

```
1 mount [device name] [directory]
2 unmount [device name or directory]
```

## 4.常用shell命令

---

### 查看帮助

```
1 man [command]
```

### 显示指定目录下的内容

```
1 ls [option] [file]
2
```

### 以树状图显示:

```
1 tree [options] [directory name]
2 options -a 显示所有文件和目录
3          -d 显示目录名称而非内容
4          -D 列出更改时间
5          -f
6
```

切换当前目录：

```
1 cd [directory name]
```

在目录下查找文件：

```
1 find [path] [options]
2 options -path [p] -ipath 路径名称符合, ipath忽略大小写
3         -name [name],-iname 文件名称符合
4         -type [type] 文件类型是 type=d\f\l ... 的文件
5         -amin [n] 过去n分钟内读取过的文件
6         -exec [command] 将find处理结果交给其他命令
```

```
1 whereis [options] [filename]
2 options -b 只查找二进制文件
3         -B [directory] 只在设置目录下查找bin文件
4         -m 只查找man文件
5         -M [directory] 只在设置目录下....
6         -s 只查找src文件
7         -S [directory] 只在设置目录下 ...
```

创建目录

```
1 mkdir [option] [directory name]
2 option -p 不存在就新建
```

创建文件

```
1 touch [filename]
2 > [filename]
```

重定向内容到文件中：

```
1 echo [string] > [filename]
```

## 创建链接

```
1 ln [options] [source path] [destination path]
2 options -b 覆盖之前链接
3         -d 允许超级用户制作目录的硬链接
4         -f 覆盖既有文件之前先询问用户
5         -n 把软连接实为一般目录
6         -s 建立软连接
7         -v 显示详细处理过程
8 #不加-s 默认是硬链接
```

## 复制文件

```
1 cp [options] [source path..] [destination file or directory]
2 options -a 通常在复制目录时使用，保留链接、文件属性、复制目录下所有内容
3         -d 保留软连接，约等于win的快捷方式
4         -f 覆盖已存在目标文件不提示
5         -i 覆盖前给提示
6         -p 除复制文件内容外还复制修改时间和权限等
7         -r 若为目录文件，复制所有子目录和文件
8         -l 创建文件的硬链接（指向同一节点的新边）
9         -s 创建软连接
10 #source path 可以不止一个，也可使用通配符（*），表明目录下的全部文件
```

## 移动文件

```
1 mv [options] [source path] [destination file or directory]
2 options -b 存在时创建备份
3         -i 同名先询问
4         -f 不询问、
5         -n 不覆盖任何文件或目录
6         -u 不存在时才执行
```



## 删除空目录

```
1 rmdir [option] [directory path]
2 option -p 子目录删除后该目录空，一并删除
```

## 删除文件或目录

```
1 rm [options] [file or directory path]
2 options -i 逐一询问
3          -f 忽略不存在文件，只读时也不提示
4          -r 递归的删除目录及所有子文件
5          -v 显示详细处理过程
```

## 查看CPU占用率和内存占用：

```
1 top
```

## 查看进程：

```
1 ps [options]
2 ps = process status, 相当于任务管理器
3 options -a 列出所有
4          -u 详细信息
5          -x 也显示没有控制终端的进程
```

## 终止进程：

```
1 kill [options] [PID]
2 options -l 列出所有进程信号名称和编号
3          -s [要发送的名称和编号]
4          -u [用户名] 向指定用户的所有进程发送信号
5 kill -9/KILL 强制终止
```

## 5. 软件管理

---

```
1 sudo apt [command]
```

## 6. 文档显示与编辑

---

### 连接并显示文件

```
1 cat [options] [file name]
2 options -n 由1开始对所有输出行数编号
3          -b 对空白行不编号
4
```

### 根据文本内容过滤文件

grep命令可以查找拥有与给定正则表达式相匹配的内容的文件，如果发现匹配成功的文件，grep命令默认会把含有匹配字符串的那一行显示出来。如果没有指定文件，则grep命令会从标准输入设备（键盘）读取数据。

```
grep [options] [想要匹配的正则表达式] [文件或目录...]
options -a 将二进制文档也以文本形式处理
        -A [n] 除了显示匹配的那一行之外，还显示该行之后的n行（after）
        -B [n] 除了显示匹配的那一行之外，还显示该行之前的n行（before）
        -C [n] 除了显示匹配的那一行之外，还显示该行之前和之后的n行（context）
        -c 计算总匹配的行数
        -r 查找目录而非文件时，必须加上这一参数，否则会报错
        -e 使用正则表达式进行匹配，默认也是这样
        -E 使用拓展的正则表达式进行匹配
        -i 忽略字符大小写的差别
        -l 列出含有匹配成功内容的文件名
        -n 在显示匹配的行之前，标示出该行的行号
        -v 显示不包含匹配表达式的所有行，相当于反向选择
```

正则表达式：

补充：正则表达式的基础写法

表达式	含义
^word	搜寻以word开头的行
word\$	搜寻以word结束的行
.	匹配任意一个字符
\c	转义\后面的特殊字符c，在正则表达式中有特殊含义的字符必须要先转义才能使用
c*	表示*前面的字符c可以重复0次到多次
[list]	匹配一系列字符中的一个
[n1-n2]	匹配一个字符范围，如[0-9]、[a-z]中的一个字符
[^list]	匹配一系列字符以外的字符
\<word	匹配以word开头的单词
word\>	匹配以word结尾的单词

补充：拓展正则表达式的几个例子

表达式	含义
c?	匹配0个或1个字符c，即c既可以出现也可以不出现
c+	表示+前面的字符c可以重复1次到多次，注意跟*有区别
	表示“或”，匹配一组可选的字符或字符串

文件编辑

编辑文档——vi/vim

version 1.1  
April 1st, 06  
翻译: 2006-3-21

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

% 按号  
匹配

^ "按"  
行首

& 重复  
:s

\* next  
ident

( 句首

) 下  
句首

"soft" bol  
down

+ 后一行

~ 跳转到  
标注

1

2

3

4

5

6

7

8

9

0 "能"  
行首

= 自动  
格式化

Q 切换到  
ex模式

q 取消  
ex模式

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  
tab

t 下一  
tab

Y 拷贝  
行

y 拷贝  
行

U 取消  
行内命令

u 取消  
命令

I 到行首  
插入

O 空行  
(前)

o 空行  
(后)

P 粘贴  
(前)

p 粘贴  
(后)

{ 段首

}

[ 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  
字符

F 到内字符  
前查找

f 到内字符  
后查找

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/F

" 寄存器  
标识

' 跳转到标  
记的行首

| 行首/  
列

\ 未用

Z 退出  
命令

Z 附加  
命令

X 删除  
(字符)

x 删除  
(字符)

C 修改  
至行末

c 修改  
(字符)

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  
中行

m 设置  
标注

< 后退

> 前进

? 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/sr/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)

100%