

轻松学  
编程

# 轻松学

# Linux 编程

阎映炳 等编著



CD-ROM

## 720分钟多媒体教学视频

- **结构独特:** 每个知识点按照“概念->例代码->运行结果->代码解析”
- **形式新颖:** 用准确的语言总结概念、用直观的图示演示过程、用详细的注释解释代码、用形象的比喻帮助记忆
- **内容丰富:** 重要知识点覆盖全面, 实例丰富多彩
- **附赠光盘:** 在随书附赠的光盘中配备了完整的视频教学讲座和本书所使用的全部项目文件与代码
- **技术支持:** 读者可以直接登录[www.rzchina.net](http://www.rzchina.net)获取更多的学习资料



化学工业出版社



# 轻松学

# Linux 编程

阎映炳 等编著



化学工业出版社

· 北京 ·

Linux 操作系统是由芬兰 Helsinki 大学的 Linus Torvalds 于 1991 年开发的，经过二十年的发展，目前已经成为全球最受欢迎的操作系统之一。Linux 在服务器领域和桌面应用中都有杰出的表现，大至规模庞大的数据中心，小至可置于掌心的手持设备，都可以找到 Linux 的身影。掌握 Linux 系统下的程序设计技术，是一个优秀程序员的必修课。

要学好 Linux 平台下的编程，关键是选择一本合适的入门书籍。为了使 Linux 平台编程的初学者少走弯路，快速、熟练地掌握 Linux 编程技术，编者总结自身多年来的 Linux 应用开发经验，向广大读者奉献本书。书中结合大量生动翔实的代码实例，使读者在学习理论知识的同时，能够灵活地与实践相结合，以快速地掌握编程的技巧。

## 本书特点

### 1. 内容翔实，重点突出

本书从初学者的角度出发，全面介绍了 Linux 环境下 C 语言编程的基础知识。书中涉及了 Linux 环境下编程的各个方面，在讲解的过程中并不是泛泛列举知识点，而是在对知识点进行介绍的同时，选择当前最热门、应用最广泛的一些技术进行了深入的探讨。

### 2. 概念准确，易于理解

作为一本 Linux 环境下 C 语言编程的入门图书，书中概念描述准确，便于读者理解和掌握。本书对每个概念都使用准确而且精练的语言总结，并配以丰富的插图，使读者更好地理解相关概念。

### 3. 实例丰富，强调实践

为了让读者易于掌握 Linux 环境下 C 语言编程的技巧，书中列举了大量的实例。通过这些实例，读者可以更加深入地理解相关概念，从而达到熟练掌握 Linux 环境下程序设计方法的目的。此外，本书重点强调实践性，书中大多数实例都源自编者的实际开发经历。通过对这些实例的学习，可以增强读者的动手能力。

### 4. 代码规范，注释丰富

本书中程序源代码结构清晰，语句简洁，体现了良好的编码风格，有利于读者养成良好的代码编写习惯。



## 本书内容

本书共分为四篇，从理论介绍到具体实践，循序渐进地介绍了 Linux 系统下 C 语言的编程技术。

第一篇（第 1 章～第 2 章）介绍了 Linux 操作系统和 shell 编程环境。

第二篇（第 3 章～第 8 章）介绍了 C 语言及其编程环境，包括 C 语言简介、vi 编辑器、gcc 编译器、make 的使用、程序调试以及创建与使用库等内容。

第三篇（第 9 章～第 14 章）介绍了 Linux 系统下的输入输出及进程管理，包括文件操作、标准输入输出库，同时还介绍了两种界面程序设计方法——QT 和 GTK+，最后介绍了进程和信号等内容。

第四篇（第 15 章～第 21 章）介绍了 Linux 系统下的进程间通信及网络编程的相关知识，重点讲解了四种比较重要的进程间通信机制——管道、消息队列、共享内存和信号量，最后介绍了 Linux 网络环境、基本套接口编程和综合实例等内容。

本书配光盘一张，内容为本书教学视频、源代码、教学 PPT 及习题参考答案。

## 本书读者

- Linux 系统下 C 语言编程的初学人员。
- Linux 服务器领域的开发人员。
- Linux 桌面应用的开发人员。
- 想了解 Linux 系统下 C 语言编程的其他人员。

## 本书编者

本书主要由阎映炳编写，其他参与编写和资料整理的人员有刘成、马臣云、潘娜、阮履学、陶则熙、王大强、王磊、徐琦、许少峰、颜盟盟、杨娟、杨瑞萍、于海波、俞菲、曾苗苗、赵莹、朱存等。

编者

# 目 录

## 第一篇 系统环境

<b>第 1 章 Linux 系统概述</b> .....	2	2.1.2 如何进入 shell	13
1.1 计算机操作系统简介 .....	2	2.1.3 如何使用 shell	15
1.1.1 操作系统的概念	2	<b>2.2 shell 编程基础</b> .....	16
1.1.2 操作系统的基本功能	2	2.2.1 创建和运行 shell 脚本程序	16
1.1.3 主要操作系统简介	3	2.2.2 shell 环境变量	16
1.2 Linux 操作系统介绍 .....	4	2.2.3 常用的 shell 命令	18
1.2.1 Linux 的来源	4	2.2.4 管道与重定向的使用	20
1.2.2 什么是 Linux	5	2.2.5 shell 变量的使用	21
1.2.3 Linux 的特性及优点	7	2.2.6 shell 运算符的应用	22
1.2.4 为什么要选择 Linux	8	2.2.7 在 shell 脚本中进行条件控制	24
1.2.5 内核的组成	8	2.2.8 在 shell 脚本中使用 for 循环	25
1.3 主流 Linux 操作系统及发行版本	9	2.2.9 在 shell 脚本中使用 while 循环	27
1.3.1 Linux 内核的版本	10	2.2.10 在 shell 脚本中使用 until 循环	28
1.3.2 Linux 的发行版本	10	2.2.11 在 shell 脚本中使用函数	28
1.4 小结	10	<b>2.3 综合实例</b> .....	29
1.5 习题	10	2.3.1 实例需求	29
1.6 上机实训	11	2.3.2 系统设计	29
<b>第 2 章 shell 环境</b> .....	12	2.3.3 程序代码	30
2.1 shell 介绍	12	<b>2.4 小结</b> .....	34
2.1.1 shell 的种类	12	<b>2.5 习题</b> .....	34
		<b>2.6 上机实训</b> .....	35

## 第二篇 C 语言及编程环境

<b>第 3 章 C 语言简介</b> .....	38	3.2.2 标识符	40
3.1 C 语言概述 .....	38	3.2.3 关键字	40
3.1.1 C 语言的发展简史	38	3.2.4 常量的类型	41
3.1.2 C 语言的特点	39	3.2.5 变量的类型	41
3.2 C 语言的组成元素 .....	39	3.2.6 变量的存储类型	42
3.2.1 字符集	39	3.2.7 变量的作用域	44
		3.2.8 运算符	44



3.2.9	注释方法	46	4.2.6	文本移动命令	70
3.3	语句与控制结构	46	4.2.7	搜索命令	71
3.3.1	表达式语句	46	4.2.8	ex 转义命令	71
3.3.2	复合语句	46	4.3	vi 编辑器的选项	72
3.3.3	函数调用语句	46	4.3.1	选项的含义	72
3.3.4	控制语句	47	4.3.2	选项的设置方式	73
3.4	函数与程序结构	51	4.4	小结	74
3.4.1	库函数	51	4.5	习题	74
3.4.2	用户自定义函数	51	4.6	上机实训	75
3.5	数组	52	<b>第 5 章</b>	<b>gcc 编译器</b>	<b>76</b>
3.5.1	一维数组的定义和使用	52	5.1	编译过程简述	76
3.5.2	多维数组的定义和使用	53	5.1.1	预编译过程	76
3.6	结构	55	5.1.2	编译的过程	78
3.6.1	结构的定义	55	5.1.3	优化及汇编的过程	79
3.6.2	结构成员的引用	56	5.2	链接过程简述	79
3.7	指针	56	5.2.1	链接的过程	80
3.7.1	指针的概念	57	5.2.2	静态链接与动态链接	80
3.7.2	指针的定义和使用	57	5.3	gcc 编译器简述	81
3.7.3	指针变量的运算	57	5.3.1	程序的编译与链接	81
3.8	综合实例	58	5.3.2	gcc 编译器的工作过程	81
3.8.1	冒泡排序算法原理	58	5.4	gcc 编译器语法	82
3.8.2	冒泡排序算法实现	59	5.4.1	常用语法	82
3.9	小结	61	5.4.2	用 gcc 编译器生成可执行文件	83
3.10	习题	62	5.4.3	用 gcc 编译器生成动态链接库	83
3.11	上机实训	62	5.4.4	如何使用动态链接	85
<b>第 4 章</b>	<b>vi 编辑器</b>	<b>65</b>	5.4.5	gcc 编译器常见错误排除	85
4.1	vi 编辑器概述	65	5.5	小结	86
4.1.1	vi 的启动	65	5.6	习题	86
4.1.2	vi 的操作方式	65	5.7	上机实训	86
4.1.3	vi 编辑器的功能键	67	<b>第 6 章</b>	<b>make 的使用</b>	<b>88</b>
4.1.4	退出 vi 编辑器	68	6.1	makefile 简介	88
4.2	vi 编辑器的命令	68	6.2	makefile 的书写规则	89
4.2.1	光标移动命令	68	6.2.1	基本语法规则	89
4.2.2	滚动屏幕命令	69	6.2.2	定义变量	89
4.2.3	文本编辑命令	69	6.2.3	环境变量	90
4.2.4	文本删除命令	69	6.2.4	通配符的使用	90
4.2.5	文本修改命令	70			

6.2.5 使用条件判断.....	91	7.1.6 其他可能的内存错误.....	104
6.2.6 在 makefile 中使用函数.....	92	7.2 gdb 介绍.....	105
6.2.7 使用 make 与直接使用 gcc 脚本 的区别.....	93	7.2.1 利用 gdb 调试的前提.....	105
6.3 make 工具.....	93	7.2.2 启动 gdb 的方法.....	105
6.3.1 运行 make 命令.....	93	7.2.3 gdb 的基本功能.....	105
6.3.2 make 命令的工作过程.....	94	7.3 使用 gdb 进行调试实例.....	108
6.3.3 在 makefile 中使用伪目标.....	94	7.4 小结.....	109
6.3.4 make 命令的返回值.....	95	7.5 习题.....	109
6.4 综合实例.....	95	7.6 上机实训.....	109
6.4.1 makefile 应用的环境.....	95	<b>第 8 章 创建与使用库.....</b>	<b>111</b>
6.4.2 makefile 的实现及解释.....	95	8.1 函数库介绍.....	111
6.5 小结.....	96	8.1.1 系统函数库的使用.....	111
6.6 习题.....	96	8.1.2 用户自定义函数库的创建和使用.....	112
6.7 上机实训.....	97	8.2 库函数与系统调用.....	114
<b>第 7 章 程序调试.....</b>	<b>98</b>	8.2.1 系统调用介绍.....	115
7.1 错误处理.....	98	8.2.2 库函数介绍.....	116
7.1.1 使用标准错误输出.....	98	8.3 动态库的创建与使用.....	117
7.1.2 使用 errno 全局变量.....	99	8.4 综合实例.....	120
7.1.3 使用错误信号处理.....	100	8.5 小结.....	121
7.1.4 使用 assert 断言.....	101	8.6 习题.....	122
7.1.5 内存泄露的检查.....	102	8.7 上机实训.....	122
<b>第三篇 输入输出及进程管理</b>			
<b>第 9 章 文件操作.....</b>	<b>126</b>	9.2.5 文件的随机存取.....	151
9.1 Linux 文件系统简述.....	126	9.3 文件安全编程.....	153
9.1.1 逻辑磁盘分区管理.....	126	9.3.1 文件的属主和用户组编程.....	153
9.1.2 文件系统的建立与挂载.....	127	9.3.2 设置文件权限 (UGO 模式).....	155
9.1.3 虚拟文件系统.....	131	9.3.3 设置文件权限 (ACL 模式).....	158
9.1.4 ext2 文件系统.....	131	9.4 文件属性编程.....	162
9.1.5 文件类型.....	133	9.5 目录编程.....	166
9.1.6 文件权限管理.....	134	9.6 综合实例.....	170
9.2 文件基本操作.....	139	9.6.1 ELF 文件格式.....	171
9.2.1 文件编程的基本概念.....	140	9.6.2 程序实现.....	171
9.2.2 文件的创建与打开.....	141	9.7 小结.....	175
9.2.3 文件的读写.....	146	9.8 习题.....	176
9.2.4 文件的关闭与删除.....	148	9.9 上机实训.....	176



<b>第 10 章 标准输入输出库</b> .....	178	12.1.1 Linux 图形界面原理	219
10.1 标准 I/O 的基本概念	178	12.1.2 X 协议	220
10.1.1 流	178	12.1.3 GNOME 与 KDE 的启动	220
10.1.2 缓存	179	12.1.4 GNOME 与 KDE 的区别	221
10.1.3 标准输入、标准输出和标准 错误输出	181	<b>12.2 使用 GTK+进行开发</b> .....	221
10.2 使用标准 I/O 进行文件操作	183	12.2.1 GTK+的安装	221
10.2.1 打开关闭流文件	183	12.2.2 GTK+程序的初始化与退出	222
10.2.2 单字符方式读写	184	12.2.3 GTK+的事件处理	222
10.2.3 行方式读写	185	12.2.4 使用 GTK+实现 HelloWorld	223
10.2.4 二进制方式读写	187	12.2.5 编译 GTK+程序	225
10.2.5 格式化输入/输出	189	12.2.6 在 GTK+中使用控件	225
10.2.6 在流文件中定位	192	<b>12.3 综合实例</b> .....	226
10.3 综合实例	193	12.3.1 实例需求	227
10.4 小结	197	12.3.2 实例代码及解释	227
10.5 习题	197	<b>12.4 小结</b> .....	228
10.6 上机实训	197	<b>12.5 习题</b> .....	229
<b>第 11 章 界面程序设计——Qt</b> .....	204	<b>12.6 上机实训</b> .....	229
11.1 Qt 简述	204	<b>第 13 章 进程</b> .....	233
11.1.1 Qt 的组成	204	13.1 进程的基本概念	233
11.1.2 Qt 的优点	204	13.1.1 进程的属性	233
11.2 Qt 开发包的安装	205	13.1.2 进程的内存映像	236
11.3 Qt 集成开发环境介绍	206	13.1.3 进程组	236
11.3.1 启动设计器	206	13.1.4 进程的会话	238
11.3.2 设计器界面元素介绍	206	13.1.5 进程的控制终端	240
11.4 Qt 程序开发	207	13.1.6 进程的状态	241
11.4.1 建立新项目	208	13.1.7 进程的优先级	242
11.4.2 设计窗口	208	<b>13.2 进程的运行环境</b> .....	245
11.4.3 添加事件处理	209	13.2.1 进程的入口函数	245
11.4.4 添加主程序	210	13.2.2 进程的环境变量	250
11.5 Qt 程序的生成	212	13.2.3 进程的内存分配	252
11.6 小结	213	<b>13.3 进程的创建</b> .....	254
11.7 习题	213	13.3.1 调用 fork 创建进程	254
11.8 上机实训	213	13.3.2 调用 exec 系列函数执行程序	256
<b>第 12 章 界面程序设计——GTK+</b> .....	219	13.3.3 调用 system 创建进程	259
12.1 GNOME 与 KDE	219	<b>13.4 进程的终止</b> .....	260
		13.4.1 调用 exit 退出进程	260
		13.4.2 调用 wait 等待进程退出	260



13.5	小结 .....	263	14.2.7	等待信号 .....	279
13.6	习题 .....	263	14.2.8	信号处理函数的实现 .....	281
13.7	上机实训 .....	263	14.3	信号的发送 .....	283
<b>第 14 章</b>	<b>信号 .....</b>	<b>266</b>	14.3.1	使用 kill 发送信号 .....	283
14.1	信号的基本概念 .....	266	14.3.2	使用 sigqueue 发送信号 .....	284
14.1.1	信号的定义 .....	266	14.4	SIGALRM 信号 .....	285
14.1.2	信号的来源 .....	267	14.4.1	安装 SIGALRM 信号 .....	285
14.1.3	信号的分类 .....	267	14.4.2	设置定时器 .....	286
14.2	信号的安装及处理 .....	269	14.5	SIGCLD 信号 .....	287
14.2.1	信号的处理方式 .....	269	14.5.1	子进程的退出过程 .....	287
14.2.2	用 signal 安装信号 .....	270	14.5.2	SIGCLD 信号的处理 .....	288
14.2.3	用 sigaction 安装信号 .....	271	14.6	小结 .....	290
14.2.4	信号的阻塞处理 .....	274	14.7	习题 .....	290
14.2.5	信号集的操作 .....	277	14.8	上机实训 .....	290
14.2.6	未决信号的处理 .....	277			
<b>第四篇 进程间通信(IPC)及网络编程</b>					
<b>第 15 章</b>	<b>进程间通信——管道 .....</b>	<b>294</b>	16.1.1	Shell 环境控制 IPC .....	309
15.1	进程间通信概念 .....	294	16.1.2	进程间通信关键字 .....	311
15.2	管道的概念及分类 .....	295	16.1.3	进程间通信标识符 .....	311
15.2.1	管道的概念及特点 .....	295	16.1.4	IPC 权限许可结构 .....	312
15.2.2	管道的分类 .....	296	16.2	消息队列基本概念 .....	313
15.3	管道编程 .....	296	16.2.1	队列 .....	313
15.3.1	创建管道 .....	296	16.2.2	消息 .....	313
15.3.2	读写管道 .....	299	16.2.3	消息队列 .....	314
15.3.3	关闭管道 .....	301	16.3	消息队列编程 .....	315
15.3.4	管道 I/O .....	301	16.3.1	键值生成函数 .....	315
15.4	命名管道编程 .....	303	16.3.2	创建消息队列 .....	316
15.4.1	创建管道 .....	303	16.3.3	消息发送 .....	319
15.4.2	打开管道及读写 .....	305	16.3.4	消息接收 .....	322
15.4.3	管道的删除 .....	307	16.3.5	控制消息队列 .....	323
15.5	小结 .....	307	16.4	小结 .....	326
15.6	习题 .....	307	16.5	习题 .....	326
15.7	上机实训 .....	307	16.6	上机实训 .....	326
<b>第 16 章</b>	<b>进程间通信——消息队列 .....</b>	<b>309</b>	<b>第 17 章</b>	<b>进程间通信——共享内存 .....</b>	<b>330</b>
16.1	System V 进程间通信概述 .....	309	17.1	共享内存基本概念 .....	330
			17.1.1	共享内存编程模型 .....	330



17.1.2 共享内存的映射.....	331	19.2 TCP/IP 协议概述.....	367
17.1.3 共享内存数据结构.....	331	19.2.1 TCP/IP 分层模型.....	367
17.2 共享内存编程.....	332	19.2.2 TCP/IP 协议族.....	368
17.2.1 创建共享内存.....	332	19.2.3 网络地址.....	369
17.2.2 映射共享内存.....	334	19.2.4 端口.....	371
17.2.3 删除共享内存映射.....	336	19.3 客户机/服务器模型.....	372
17.2.4 控制共享内存.....	336	19.4 传输控制协议.....	374
17.3 小结.....	339	19.4.1 连接建立.....	374
17.4 习题.....	339	19.4.2 连接关闭.....	374
17.5 上机实训.....	339	19.4.3 TCP 数据报格式.....	376
<b>第 18 章 进程间通信——信号量.....</b>	<b>342</b>	19.5 用户数据报协议.....	377
18.1 PV 操作原理.....	342	19.6 小结.....	377
18.1.1 PV 操作的来源.....	342	19.7 习题.....	378
18.1.2 PV 操作的定义.....	343	19.8 上机实训.....	378
18.1.3 PV 操作的应用.....	344	<b>第 20 章 基本套接字编程.....</b>	<b>381</b>
18.2 信号量基本概念.....	346	20.1 套接字编程简述.....	381
18.2.1 Linux 信号量简介.....	346	20.1.1 半相关与全相关.....	381
18.2.2 信号量的控制结构.....	347	20.1.2 地址族与协议族.....	382
18.3 信号量编程.....	347	20.1.3 面向连接与面向无连接.....	382
18.3.1 创建信号量.....	347	20.1.4 套接字类型.....	383
18.3.2 信号量操作.....	349	20.1.5 字节序.....	383
18.3.3 信号量控制.....	353	20.1.6 套接字连接方式.....	385
18.4 综合实例——利用信号量实现 生产者-消费者模型.....	356	20.1.7 数据传输方式.....	385
18.4.1 需求.....	356	20.2 套接字数据结构.....	386
18.4.2 需求分析与设计.....	356	20.2.1 套接字地址结构.....	387
18.4.3 实现代码及分析.....	357	20.2.2 通用套接字地址结构.....	387
18.5 小结.....	360	20.2.3 主机名称数据结构.....	388
18.6 习题.....	360	20.2.4 服务名称数据结构.....	390
18.7 上机实训.....	360	20.2.5 通用数据收发结构.....	391
<b>第 19 章 Linux 网络环境.....</b>	<b>362</b>	20.3 基本套接字函数.....	392
19.1 计算机网络基础.....	362	20.3.1 字节操作函数.....	392
19.1.1 计算机网络分类.....	362	20.3.2 字节序操作函数.....	395
19.1.2 网络拓扑结构.....	363	20.3.3 地址转换函数.....	395
19.1.3 网络通信协议.....	364	20.3.4 套接字函数.....	398
19.1.4 OSI 参考模型.....	365	20.4 套接字选项.....	415
		20.4.1 套接字选项函数.....	415
		20.4.2 SO_KEEPALIVE 选项.....	416

---

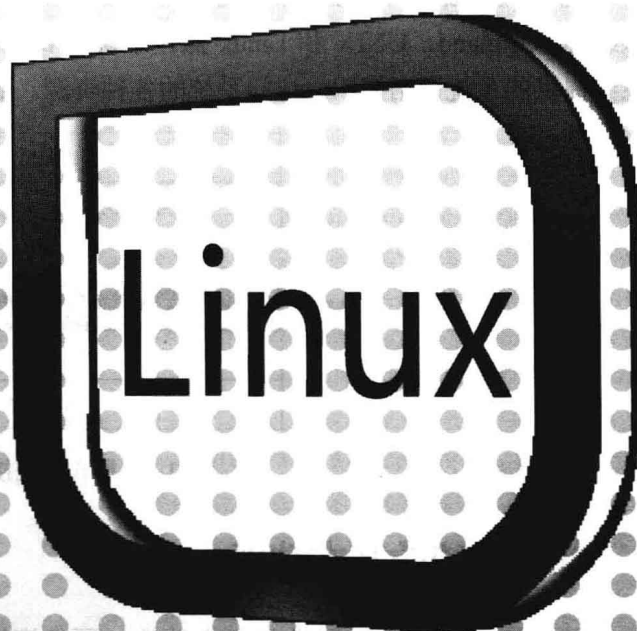
20.4.3	SO_LINGER 选项.....	417	20.7	小结.....	431
20.4.4	SO_RCVBUF 和 SO_SNDBUF 选项.....	419	20.8	习题.....	432
20.4.5	SO_RCVTIMEO 和 SO_SNDTIMEO 选项.....	420	20.9	上机实训.....	432
20.4.6	SO_REUSEADDR 和 SO_REUSEPORT 选项.....	421	<b>第 21 章</b>	<b>综合实例——银行代理收费 服务器.....</b>	<b>436</b>
20.5	TCP 套接字编程.....	421	21.1	程序需求.....	436
20.5.1	重复服务器编程.....	421	21.2	程序实现.....	437
20.5.2	并发服务器编程.....	422	21.3	小结.....	451
20.6	UDP 套接字编程.....	427	<b>附录 1</b>	<b>常见面试题.....</b>	<b>452</b>
20.6.1	UDP 编程模型.....	428	<b>附录 2</b>	<b>Linux 下常见 C 函数字母索引.....</b>	<b>470</b>
20.6.2	UDP 客户/服务器编程.....	428			

# 第一篇 系统环境

---

第 1 章 Linux 系统概述

第 2 章 shell 环境



# 第 1 章 Linux 系统概述

自上世纪 90 年代初，Linux 操作系统以其强大的功能和开源的特性快速占领了服务器市场和桌面应用领域。Linux 被业界认为是最有前途的操作系统之一。本章将简要介绍 Linux 操作系统的功能和特点。

## 1.1 计算机操作系统简介

计算机系统由硬件和软件两个部分组成。操作系统是计算机中的底层软件，其他软件如数据库系统、编译系统以及各种应用软件等都依赖于操作系统的支持。

### 1.1.1 操作系统的概念

操作系统是用户与计算机硬件之间的接口，是用于控制和管理系统资源，方便用户使用计算机程序的集合。通过操作系统，用户可以安全、快捷地操纵计算机硬件系统并运行自己的程序。同时，使用操作系统也便于对计算机中的各项资源进行管理。

用户使用计算机操作系统主要通过两种方式进行：一种是通过执行 shell 命令进行，典型的如 DOS 操作系统的 command，UNIX 和 Linux 操作系统的 bash、csh 等；另一种则是编写程序，通过调用操作系统提供的系统调用接口，访问系统的各种资源。用户使用操作系统的两种方式如图 1-1 所示。

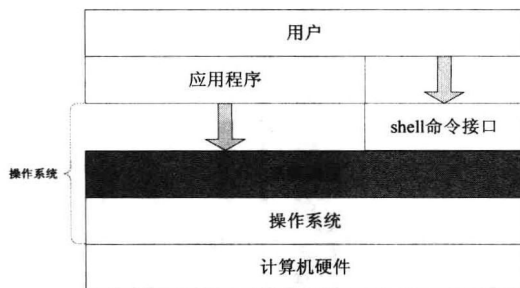


图 1-1 操作系统接口

### 1.1.2 操作系统的基本功能

典型的计算机系统中，通常包含多种软、硬件资源，如处理器（CPU）、存储器（内存、

磁盘存储器)、输入/输出设备、数据或程序等,其结构如图 1-2 所示。

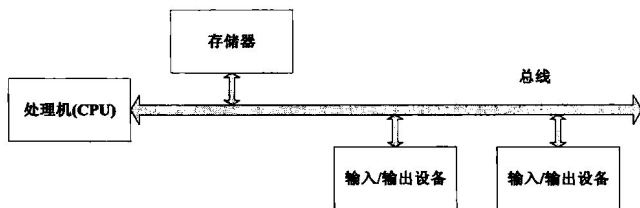


图 1-2 计算机系统的组织结构

操作系统的主要功能就是对上述这些资源进行有效的管理。简要地说,操作系统的基本功能包括以下几个方面。

- 处理器管理:对处理器进行分配,并对其运行进行有效的控制和管理。通常情况下,处理器的分配是以进程为单位的,所以对处理器的管理也可以归结为对进程的管理。
- 存储器管理:负责内存的分配与回收。为程序的运行分配内存空间,提供其运行环境,方便进程合理使用存储器。
- 输入/输出设备管理:负责输入/输出设备的分配与控制,如打印机、键盘等。
- 文件管理:负责文件的存取。通常情况下,程序和数据总是以文件的形式存储在磁盘等介质中供用户使用。文件管理的主要任务就是对文件进行有效的管理,以方便用户使用。

### 1.1.3 主要操作系统简介

操作系统从诞生至今已有近 50 年的历史。上世纪 60 年代至 70 年代是操作系统发展的活跃时期。到上世纪 80 年代,操作系统已发展得比较成熟。经过多年的发展,操作系统的功能日趋完善,性能更加优异。按照通常的划分方法,操作系统可以划分为以下几种。

- MS-DOS 操作系统:提到 MS-DOS,读者可能会立即回忆起那熟悉的“C>:”提示符。MS-DOS 是在 1981 年推出的,随 IBM 公司的 IBM-PC 计算机一起发行。随着 IBM-PC 计算机的畅销,同时也由于 MS-DOS 强大的功能,MS-DOS 成为 16 位微型计算机上操作系统的标准。
- Windows 系列操作系统:1990 年,Microsoft 推出了 Windows 3.0 版本,它以其友好的图形界面征服了广大计算机用户。该操作系统同时具有支持多任务、易学好用的特点,因而在微机领域迅速流行。1993 年推出的 Windows NT 是真正的 32 位多任务的操作系统,在服务器操作系统领域占有了很大市场。后来从 Windows 95、Windows 98、Windows XP、Windows Vista,一直发展至现在的 Windows 7,Windows 系列操作系统在微型机领域仍然是绝对的主流。
- UNIX 操作系统:UNIX 操作系统是由美国 Bell 实验室开发的多用户、多任务的操作系统。本书将介绍的 Linux 操作系统也是在 UNIX 系统的基础上发展而来的。UNIX 操作系统不仅可以在微型计算机上使用,也支持小型计算机乃至大型计算机。同时,UNIX



操作系统具有优良的性能、强大的安全控制机制，到目前为止仍然是服务器领域最重要的操作系统。

- **Linux 操作系统：**Linux 实际上是 UNIX 操作系统的变种。除继承 UNIX 操作系统的全部优点外，Linux 操作系统还将其源代码完全开放。这使得 Linux 操作系统在服务器领域大行其道，占据了部分本属于 UNIX 操作系统的市场。除此之外，Linux 操作系统一改 UNIX 操作系统单一的字符界面，提供了不亚于 Windows 操作系统的华丽的图形界面，这也吸引了大量桌面办公领域的用户由 Windows 操作系统转向 Linux 阵营。本书将介绍的内容，就是在 Linux 操作系统下进行程序设计的方法与技巧。

## 1.2 Linux 操作系统介绍

Linux 操作系统源自于荷兰 Helsinki 大学的学生 Linus Torvalds。他当时想要开发一个 Minix 系统的替代品，可用于 x86 体系结构的个人计算机上，并具有 UNIX 操作系统的全部功能，这就是 Linux 系统的开端。

### 1.2.1 Linux 的来源

Linux 的起源最早要追溯到 UNIX 操作系统。1969 年，Bell 实验室的 Ken Thompson 利用一台闲置的 PDP-7 计算机开发了一个多用户、多任务的操作系统。后来，另一个研究人员 Dennis Richie 加入了该项目，在他们的共同努力下诞生了最早的 UNIX 操作系统。UNIX 操作系统最早是用汇编语言实现的。

由于 UNIX 操作系统强大的功能和优良的性能，很多商业公司和研究机构纷纷加入到该系统的研发之中。如此一来，产生了大量的 UNIX 版本，如 AT&T 的 System V、加州大学的 BSD、IBM 的 AIX 等。这些操作系统版本之间互不兼容，只支持基于各自硬件体系结构的计算机系统，不支持个人计算机。这一情况成为 Linux 操作系统诞生的重要因素。

1991 年，荷兰 Helsinki 大学的学生 Linus Torvalds 需要一款多用户、多任务的操作系统。但财力有限，他能买得起的只有 Minix 操作系统。而 Minix 系统的功能简单，在当时是一个被用于教学的类似于 UNIX 的操作系统。Linus Torvalds 对 Minix 很不满意，决定自己编写一款新的操作系统。

由于 UNIX 操作系统具有优良的性能，Linus Torvalds 决定以 UNIX 操作系统为原型开始进行开发工作，并且进展很快。到 1991 年，这个操作系统的 0.02 版已经编写完成，被命名为 Hobby。受开放源码思想的影响，Linus Torvalds 将他新编写的操作系统放置到互联网上并开放源码与其他开发人员共享。随着大量开发人员的加入，使得这个新的系统发展很快，并最终成为一个真正意义上的操作系统。这个操作系统就是 Linux 的最早版本。

## 提示:

1991 年, Linus Torvalds 在 BBS 中发布了一个公告(如图 1-3 所示),成为 Linux 系统诞生的重要标志。其中文意思大致为:各位,我正在开发一款操作系统(只是个人兴趣,不会做得像 GNU 那么大、那么专业)。从这段公告可以看出, Linus Torvalds 也没有想到, Linux 会发展到今天这个程度。

这个新生成的操作系统是基于 UNIX 操作系统的,所以被称为 Linus's UNIX,简称为 Linux。其标志是一个活泼可爱的企鹅形象,如图 1-4 所示。

```
Hello everybody out there using minix-
I'm doing a (free) operation system (just a hobby,
won't be big and professional like gnu) for 386(486) AT
clones.
```

图 1-3 Linus 的 BBS 公告



图 1-4 Linux 的标志

## 1.2.2 什么是 Linux

Linux 是一个多用户、多任务的操作系统。它由 UNIX 操作系统发展而来,但又比 UNIX 操作系统功能更为强大,尤为重要是它可以免费使用并自由传播。由于 Linux 的开源特性,自从 Linux 诞生以来,来自世界各地的程序开发人员不断对其功能进行丰富完善。发展到现在, Linux 已经成为服务器领域和桌面应用领域非常重要的操作系统之一。

Linux 具有十分重要的用途,它被广泛应用于服务器联机事务处理、办公自动化桌面应用以及个人工作站等领域。由于 Linux 是一种类似于 UNIX 的操作系统,这就使得其在服务器领域与 UNIX 操作系统具有同等的功能。加上 Linux 的开源特性,在服务器领域的联机事务处理方面, Linux 已经取得了非凡的成绩。在桌面应用方面, Linux 提供了 KDE、GNOME 等大量的图形操作环境供用户使用。其应用领域如图 1-5 所示。

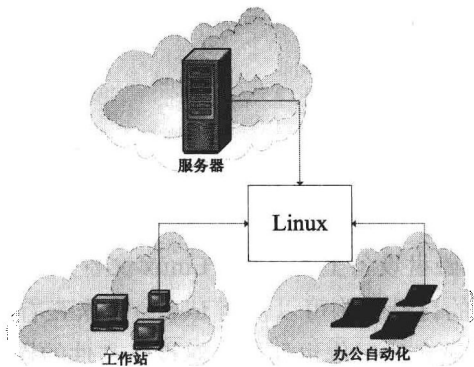


图 1-5 Linux 的应用领域

Linux 主要由四个部分内容组成:系统内核、shell、文件系统和实用工具。内核可以看做是操作系统的“心脏”,它对外提供一种称为“系统调用”的接口,供外部程序调用内核提供的



服务。其余的三个部分（shell、文件系统和实用工具）都是通过访问系统调用实现各自功能的。

系统内核是 Linux 操作系统的核心，它包含了诸如进程管理、存储管理等核心功能的处理。Linux 自诞生以来，产生了多种不同版本的内核，目前最新的内核是 3.3.6 版本。Linux 操作系统内核的版本在系统登录时可以看到，如图 1-6 所示，该系统的内核版本为 2.6.32。该版本号也可以通过 Linux 执行命令 `uname -r` 获得。

```
Red Hat Enterprise Linux Server release 6.0 (Santiago)
Kernel 2.6.32-71.el6.i686 on an i686

localhost login: _
```

图 1-6 Linux 的内核版本

shell 是操作系统对用户提供的交互操作的接口，它的作用类似于 DOS 操作系统的 `command`。shell 接收用户输入的命令并提交给系统内核去执行。传统意义上的 shell 是基于字符界面的，较常见的 shell 如 Bourne shell、C shell、Korn shell 等。Bourne shell 的操作界面如图 1-7 所示。

```
localhost login: root
Password:
Last login: Tue Apr 19 15:33:26 on tty1
root@localhost ~]# _
```

图 1-7 Bourne shell 的操作界面

除这些字符界面的 shell 外，Linux 还提供了像 KDE、GNOME 等图形操作界面。这些图形操作环境提供了一种可视化的操作方式供用户使用，这与 Windows 操作系统颇为相似。如图 1-8 所示是典型的 KDE 图形操作界面。



图 1-8 KDE 图形操作界面

文件系统用于管理存储在磁盘设备上的文件。Linux 文件系统是基于树形结构的，支持文件和目录。Linux 操作系统基本的目录结构可以使用 `tree` 命令来查看，如图 1-9 所示。在 Linux 系统中，一切文件，包括对硬件设备的操作也抽象为对设备文件的操作。为保证安全性，Linux 提供了复杂的安全管理功能。它既支持传统的 UNIX 操作系统下的文件权限方式，也支持其独特的访问控制列表（ACL）方式。

实用工具是 Linux 系统提供给用户使用的各种工具软件。在不同厂商的操作系统中，这些实用工具各不相同，如编辑工具软件、计算器工具软件等。