教学录像/源代码/PPT

Linux C 编程
从入门到精通

刘学勇◎编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Linux C 编程从入门到精通

刘学勇　编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书以 Ubuntu 11.04 为平台，系统地介绍了 Linux 下用 C 语言进行程序设计的方法，并通过列举大量的程序实例，使读者快速掌握在 Linux 下进行 C 语言程序开发的方法和技巧，并具备开发大型应用程序的能力。

本书内容丰富，主要包括 Linux 基础知识介绍，Linux 下的 C 语言编译器、调试器、程序维护工具及集成开发环境的使用方法，Linux 下通过 C 语言进行文件操作和目录操作的方法，标准 I/O 库函数，进程概念、进程操作及进程间通信的方法，线程操作，用 C 语言进行网络编程、数据库编程及 GUI 编程的方法等。

本书结构合理、概念清晰、深入浅出、易于理解，具有很强的实用性，适合想学习如何在 Linux 系统下进行 C 语言编程的读者使用。

图书在版编目(CIP)数据

Linux C 编程从入门到精通 / 刘学勇编著. —北京：电子工业出版社，2012.7

ISBN 978-7-121-17415-5

Ⅰ. ①L… Ⅱ. ①刘… Ⅲ. ①Linux 操作系统—程序设计②C 语言—程序设计 Ⅳ.①TP316.89②TP312

中国版本图书馆 CIP 数据核字（2012）第 135846 号

策划编辑：陈韦凯
责任编辑：陈韦凯　　特约编辑：刘丽丽
印　　刷：三河市鑫金马印装有限公司
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1 092　1/16　印张：30.5　字数：780 千字
印　　次：2012 年 7 月第 1 次印刷
印　　数：4 000 册　定价：65.00 元（含 DVD 光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

Linux 是当前最流行的操作系统之一。它是由芬兰大学生 Linus Torvalds 开发的类 UNIX 操作系统，它具有系统内核小、稳定性高、可扩展性好、对硬件要求低、网络功能强等特点，现在已经成为一种成熟的操作系统，并以其良好的稳定性、优异的性能给用户带来全新的感受，赢得了人们的普遍青睐。

C 语言原是 AT&T 属下的 Bell Labs 的 Dennis Ritchie 为开发 UNIX 操作系统而独立设计并实现的。随着 UNIX 操作系统的广泛流行及微型计算机的普及推广，C 语言作为 UNIX 操作系统的孪生兄弟，也广泛地应用于软件开发领域里。它具有简洁、高效、可移植性等众多优点，得到软件开发人员的喜爱，成为最受欢迎的编程语言。

Linux 操作系统同 C 语言这种具有多平台、移植性好的编程语言进行完美结合，为用户提供了一个功能强大的编程环境。掌握 Linux 下的 C 语言编程是学习 Linux 下编程必不可少的一环，本书正是以这个为出发点，介绍 Linux 系统下使用 C 语言编程的有关知识。

本书主要针对那些对 Linux 和 C 语言有一定了解，想学习如何在 Linux 系统中使用 C 语言编程的读者。

全书共分为 12 章，内容如下所述。

第 1 章是 Linux 基础知识，介绍了 Linux 的发展、版本、安装，以及 Linux 系统的一些常用命令等。

第 2 章是 Linux 下的 C 语言编程环境，主要讨论了 Linux 下 C 语言编程所使用的编辑器、编译器、调试器、程序管理工具，以及集成开发环境的使用等。

第 3 章是 Linux 下的文件编程，介绍了 Linux 下 C 语言的文件基本 I/O 操作和一些高级操作等。

第 4 章是标准 I/O 库，介绍了 Linux 下基于流的标准 I/O 操作。

第 5 章是进程操作，介绍了 Linux 进程的基本概念及 Linux 下进程控制的 C 语言编程。

第 6 章是进程间通信（IPC），讲述了 Linux 下进程间通信机制及用 C 语言实现 Linux 下进程间通信的方法等。

第 7 章是线程操作，介绍了 Linux 线程的基本概念及线程管理的 C 语言编程等。

第 8 章是网络编程，介绍了 Linux 下网络套接字编程的基本方法。

第 9 章是数据库编程，介绍了数据库的基本概念及用 C 语言访问 Linux 下 MySQL 数据库的方法。

第 10 章是 Linux 下的 GTK+/GNOME 编程，介绍了 Linux 下 X Window 系统的基本概念及 GTK+/GNOME 编程方法等。

第 11 章是综合案例，以一个网络订票模拟为例，使用了前面几章的知识，说明在 Linux 下开发一个完整软件系统的基本方法。

第 12 章是综合案例，以俄罗斯方块游戏为例介绍了开发 Linux GUI 程序的方法。

本书语言简练、阐述清晰、实例生动，能很好地帮助读者掌握 Linux 平台下使用 C 语言编程的基本方法和技巧。本书每章都提供了一些完整的应用实例或程序段，这些应用实例可直接在机

器上编译。每个应用实例程序都有较强的针对性，并充分说明了在程序设计中的方法与技巧。

本书一些重要章节后还附有习题，方便读者学习。

本书主要由刘学勇编著。此外，李龙、魏勇、王华、李辉、刘峰、徐浩、李建国、马建军、唐爱华、苏小平、朱丽云、马淑娟、周毅、张玉兰等也参与了编写工作。

本书适合那些以前没有接触过 Linux，但又想学习 Linux 下 C 语言编程的读者；而对于有 Linux 操作经验和 C 语言基础的读者，学习起来则更容易一些。

由于作者水平有限，加之时间匆忙，书中难免有错误和不妥之处，恳请读者批评指正。

编著者

目　录

第 1 章　Linux 基础知识 …… (1)

1.1　Linux 简介 …… (1)

1.1.1　Linux 的起源 …… (1)

1.1.2　Linux 的特点 …… (2)

1.1.3　Linux 的版本 …… (3)

1.1.4　Linux 的发展前景 …… (4)

1.2　Linux 的安装 …… (4)

1.2.1　发行版本的选择 …… (4)

1.2.2　安装虚拟机软件 …… (4)

1.2.3　安装 Ubuntu …… (5)

1.3　Linux 系统的常用命令 …… (5)

1.3.1　了解 Shell …… (5)

1.3.2　进入 Shell 命令行界面 …… (5)

1.3.3　文件操作命令 …… (7)

1.3.4　目录及其操作命令 …… (17)

1.3.5　文件压缩打包命令 …… (26)

1.3.6　联机帮助命令 …… (28)

1.3.7　用户操作命令 …… (29)

1.3.8　关机和重启计算机命令 …… (30)

1.4　小结 …… (32)

习题 …… (32)

第 2 章　Linux 下的 C 语言编程环境 …… (34)

2.1　Linux 编程简介 …… (34)

2.2　Linux 下的 C 语言开发环境 …… (34)

2.3　Linux C 语言程序的开发过程 …… (35)

2.4　编译器 gcc 的使用 …… (36)

2.4.1　Ubuntu 下 gcc 的安装与设置 …… (36)

2.4.2　gcc 的使用 …… (37)

2.5　make 工具及使用 …… (40)

2.5.1　make 命令和 Makefile …… (40)

2.5.2　Makefile 的规则 …… (43)

2.5.3　Makefile 中的变量 …… (44)

2.5.4　伪目标 …… (45)

2.5.5　条件语句 …… (46)

2.5.6　调试 make …… (46)

2.6　使用 autoconf …… (47)

2.6.1 创建 configure 脚本 …… (47)
2.6.2 编写 configure.in 文件 …… (48)
2.6.3 使用 autoscan 创建 configure.in 文件 …… (49)
2.6.4 用 autoconf 创建 configure …… (50)
2.6.5 更新 configure 脚本 …… (50)
2.7 使用 automake …… (50)
2.7.1 automake 的工作流程 …… (51)
2.7.2 使用 automake 生成 Makefile.in …… (51)
2.8 使用 gdb 调试程序 …… (52)
2.8.1 初次使用 gdb …… (53)
2.8.2 gdb 的基本命令 …… (57)
2.8.3 gdb 的调用 …… (57)
2.8.4 gdb 运行模式的选择 …… (59)
2.9 IDE 工具 CodeBlocks …… (59)
2.9.1 CodeBlocks 的安装 …… (59)
2.9.2 CodeBlocks 的使用 …… (59)
2.10 小结 …… (64)
习题 …… (64)
第 3 章 Linux 下的文件编程 …… (66)
3.1 概述 …… (66)
3.1.1 超级块 …… (67)
3.1.2 索引节点（inode） …… (68)
3.1.3 文件类型 …… (69)
3.2 文件描述符 …… (70)
3.3 基本文件 I/O 操作 …… (71)
3.3.1 open 函数 …… (71)
3.3.2 close 函数 …… (73)
3.3.3 read 函数 …… (73)
3.3.4 write 函数 …… (74)
3.3.5 creat 函数 …… (74)
3.3.6 lseek 函数 …… (77)
3.4 文件高级操作 …… (79)
3.4.1 文件模式 …… (79)
3.4.2 确定和改变文件模式 …… (80)
3.4.3 查询文件信息 …… (85)
3.4.4 文件其他操作 …… (89)
3.4.5 目录文件操作 …… (93)
3.4.6 特殊文件操作 …… (98)
3.5 小结 …… (102)
习题 …… (103)

第 4 章　标准 I/O 库 …… (104)
4.1　概述 …… (104)
4.2　流和 FILE 对象 …… (104)
4.3　打开和关闭流 …… (105)
4.4　读和写流 …… (108)
4.4.1　字符 I/O …… (109)
4.4.2　行 I/O …… (111)
4.4.3　块 I/O …… (112)
4.5　流文件定位 …… (115)
4.6　文件结束和错误 …… (119)
4.7　流缓冲 …… (121)
4.8　格式化 I/O …… (127)
4.8.1　格式输出 …… (128)
4.8.2　格式输入 …… (132)
4.9　临时文件 …… (136)
4.10　小结 …… (138)
习题 …… (139)
第 5 章　进程操作 …… (140)
5.1　进程概述 …… (140)
5.1.1　进程的基本概念 …… (140)
5.1.2　Linux 进程 …… (140)
5.1.3　进程的识别号（ID） …… (141)
5.1.4　进程调度 …… (141)
5.2　进程控制 …… (142)
5.2.1　进程的创建 …… (142)
5.2.2　exec 函数 …… (148)
5.2.3　结束进程 …… (154)
5.2.4　进程等待 …… (155)
5.2.5　system 函数 …… (159)
5.2.6　进程的用户标识号管理 …… (162)
5.2.7　进程标识号管理 …… (164)
5.3　综合应用实例 …… (166)
5.4　小结 …… (173)
习题 …… (174)
第 6 章　进程间通信（IPC） …… (175)
6.1　进程间通信机制概述 …… (175)
6.1.1　信号 …… (175)
6.1.2　管道 …… (177)
6.1.3　System V IPC 机制简介 …… (179)
6.2　信号处理 …… (182)

6.2.1 信号类型 …… (182)
6.2.2 处理信号的系统函数 …… (184)
6.2.3 信号集 …… (190)
6.2.4 发送信号 …… (197)
6.3 管道 …… (201)
6.3.1 基本概念 …… (201)
6.3.2 管道的创建 …… (202)
6.3.3 创建管道的简单方法 …… (206)
6.3.4 命名管道 …… (208)
6.4 System V IPC 机制 …… (212)
6.4.1 基本概念 …… (212)
6.4.2 消息队列 …… (215)
6.4.3 信号量 …… (223)
6.4.4 共享内存 …… (231)
6.4.5 综合应用实例 …… (238)
6.5 小结 …… (242)
习题 …… (242)
第 7 章 线程操作 …… (244)
7.1 线程概述 …… (244)
7.1.1 线程的基本概念 …… (245)
7.1.2 用户态线程与内核态线程 …… (245)
7.2 线程管理 …… (245)
7.2.1 创建线程和结束线程 …… (246)
7.2.2 挂起线程 …… (249)
7.2.3 线程同步 …… (250)
7.2.4 取消线程和取消处理程序 …… (261)
7.2.5 线程特定数据的处理函数 …… (265)
7.2.6 线程属性 …… (269)
7.3 小结 …… (274)
习题 …… (275)
第 8 章 网络编程 …… (276)
8.1 概述 …… (276)
8.2 TCP/IP 基础 …… (277)
8.2.1 参考模型 …… (277)
8.2.2 Linux 中 TCP/IP 网络的层结构 …… (279)
8.3 BSD 套接字接口 …… (280)
8.4 客户机/服务器模式 …… (281)
8.5 套接字网络编程 …… (282)
8.5.1 套接字编程的基本流程 …… (282)
8.5.2 套接字地址 …… (284)

8.5.3 字节顺序 …… (285)
8.5.4 字节处理函数 …… (287)
8.5.5 面向连接的基本套接字函数 …… (288)
8.5.6 其他套接字操作函数 …… (297)
8.5.7 数据报套接字操作 …… (305)
8.6 小结 …… (309)
习题 …… (309)
第 9 章 数据库编程 …… (311)
9.1 数据库基本概念 …… (311)
9.1.1 数据与数据库 …… (311)
9.1.2 数据库管理系统 …… (312)
9.1.3 数据库语言 …… (312)
9.1.4 数据库系统 …… (313)
9.1.5 主要数据模型 …… (313)
9.2 SQL 简介 …… (313)
9.2.1 数据库表格 …… (314)
9.2.2 数据查询 …… (314)
9.2.3 创建表格 …… (315)
9.2.4 向表格中插入数据 …… (316)
9.2.5 更新记录 …… (316)
9.2.6 删除记录 …… (316)
9.2.7 删除数据库表格 …… (317)
9.3 MySQL 数据库 …… (317)
9.3.1 MySQL 的安装 …… (317)
9.3.2 MySQL 管理 …… (319)
9.4 用 C 语言访问 MySQL 数据库 …… (329)
9.4.1 连接数据库 …… (329)
9.4.2 错误处理 …… (332)
9.4.3 执行 SQL 语句 …… (333)
9.5 小结 …… (347)
习题 …… (347)
第 10 章 Linux 下的 GTK+/GNOME 编程 …… (349)
10.1 X Window 简介 …… (349)
10.1.1 X 服务器 …… (349)
10.1.2 X 协议 …… (350)
10.1.3 Xlib 库 …… (350)
10.1.4 X 客户 …… (350)
10.2 GTK+/GNOME 简介 …… (350)
10.3 安装 GTK+/GNOME 库 …… (352)
10.4 GTK+编程 …… (352)

10.4.1 第一个 GTK+程序 …… (353)
10.4.2 数据类型 …… (355)
10.4.3 信号和事件 …… (355)
10.4.4 布局管理 …… (359)
10.4.5 菜单栏和工具栏 …… (365)
10.4.6 对话框构件 …… (373)
10.4.7 文本构件 …… (377)
10.4.8 使用 GTK+编写 GNOME 程序 …… (381)
10.5 小结 …… (387)
习题 …… (387)
第 11 章 综合案例 …… (389)
11.1 系统框架 …… (389)
11.1.1 数据格式 …… (390)
11.1.2 服务器端程序框架 …… (390)
11.1.3 客户端程序框架 …… (392)
11.2 程序源代码和说明 …… (393)
11.2.1 服务器端源代码 …… (393)
11.2.2 客户端源代码 …… (418)
11.3 小结 …… (435)
第 12 章 综合案例：绘图与俄罗斯方块游戏 …… (436)
12.1 GdkWindow …… (436)
12.2 颜色与颜色表 …… (436)
12.3 绘图区构件和 pixmap …… (438)
12.4 图形上下文 …… (439)
12.5 绘图 …… (442)
12.6 事件 …… (442)
12.7 基于 GDK 的绘图程序 …… (444)
12.8 俄罗斯方块游戏 …… (447)
12.8.1 global.h …… (448)
12.8.2 contorl.h 和 control.c …… (450)
12.8.3 display.h 和 display.c …… (461)
12.8.4 menu.h 和 menu.c …… (464)
12.8.5 main.c …… (467)
12.8.6 程序运行结果 …… (471)
12.9 小结 …… (474)
附录 习题答案 …… (475)

第 1 章　Linux 基础知识

Linux 从 1991 年问世到现在，短短的 20 年时间已经发展成为功能强大、设计完善的操作系统之一。作为最能体现互联网自由和开放精神的代表，Linux 自诞生以来就以软件源代码开放、可自主开发和高效灵活的特点而迅速得到众多软件开发者的推崇，越来越多的人开始在 Linux 下进行软件开发。本章主要介绍一些 Linux 的基础知识。

1.1　Linux 简介

本节将简单介绍一下 Linux 的起源、特点、版本，以及它的发展前景等，使读者对 Linux 系统有一个宏观上的了解。

1.1.1　Linux 的起源

Linux 操作系统是一种类 UNIX 操作系统，它最早是由芬兰人 Linus Torvalds 设计的。

在 Linux 诞生之前，为了教学和研究的需要，阿姆斯特丹 Vrije 大学的计算机科学家 Andrew S. Tanwnbaum 以 UNIX 为蓝本开发了 Minix 作为一个教育工具。1991 年初，Linus 开始在一台 386sx 兼容计算机上学习 Minix 操作系统。通过学习，他逐渐不能满足于 Minix 系统的现有性能，并开始酝酿开发一个新的免费操作系统，该想法很快就在 Minix 新闻组得到了响应。

1991 年 10 月 5 日，Linus 在 comp.os.minix 新闻组上发布消息，正式向外宣布 Linux 内核系统的诞生（Free minix-like kernel sources for 386-AT）：0.02 版。1991 年 11 月，Linux 0.10 版本推出；0.11 版本随后在 1991 年 12 月推出。当 Linux 非常接近于一种稳定可靠的系统时，Linus 决定将 0.13 版本改称为 0.95 版本。后来在 1994 年 3 月，终于出现了带有独立宣言意味的 Linux 1.0 版本。Linux 1.0 已经是一个功能完备的操作系统了，其内核写得紧凑高效，可以充分发挥硬件的性能，在 4MB 内存的 80386 计算机上也表现得非常好。

事实上，Linux 系统是全世界各地成千上万志愿者设计和实现的。其目的是建立不受任何商品化软件版权制约的、全世界都能自由使用的类 UNIX 操作系统。在 Linux 操作系统的设计过程中，借鉴了很多 UNIX 的思想，但源代码是全部重写的。目前，Linux 操作系统可以运行在 x86，Aplpa，MIPS，Power Mac，ARM 等类型的计算机上。从功能来看，它既可以作为普通的桌面操作系统，也可以作为中小型的网络操作系统，甚至作为大型网络的操作系统。

1.1.2 Linux 的特点

为什么 Linux 如此备受青睐？先来看一下 Linux 有什么特点吧。

1. 自由软件

Linux 可以说是作为开放源码的自由软件的代表，正是由于这一点，来自全世界的无数程序员参与了 Linux 的修改、编写工作，程序员可以根据自己的兴趣和灵感对其进行改变。这让 Linux 吸收了无数程序员智慧的精华，不断壮大。

2. 完全兼容 POSIX 1.0 标准

POSIX 是基于 UNIX 的第 1 个操作系统国际标准，这使得用户可以在 Linux 下通过相应的模拟器运行常见的 DOS、Windows 的程序。

3. 多用户、多任务

Linux 支持多用户，各个用户对于自己的文件设备有自己特殊的权利，保证了各用户之间互不影响。多任务则是现在计算机最主要的一个特点，Linux 可以使多个程序同时并独立地运行。

4. 良好的用户界面

Linux 向用户提供了两种界面：文本界面和图形用户界面。Linux 的传统用户界面是基于文本的命令行界面，即 shell，它既可以联机使用，又可存在文件上脱机使用。

Linux 还为用户提供了图形用户界面。它利用鼠标、菜单、窗口、滚动条等设施，给用户呈现一个直观、易操作、交互性强的友好的图形化界面。Linux 的图形用户界面在最近几年又有很大的改进。在图形用户界面下，几乎可以做全部的工作。

5. 支持多种文件系统

Linux 能支持多种文件系统。目前，支持的文件系统有 EXT、EXT2、EXT3、XIAFS、ISOFS、HPFS、MSDOS、UMSDOS、PROC、NFS、XFS、SYSV、MINIX、SMB、UFS、NCP、VFAT、NTFS、AFFS 等。

6. 丰富的网络功能

完善的内置网络是 Linux 的一大特点。Linux 在通信和网络功能方面优于其他操作系统。其他操作系统不包含如此紧密地和内核结合在一起的连接网络的能力，也没有内置这些联网特性的灵活性。而 Linux 为用户提供了完善的强大的网络功能。

7. 可靠的系统安全

Linux 采取了许多安全技术措施，包括对读/写进行权限控制、带保护的子系统、审计跟踪、核心授权等，这为网络多用户环境中的用户提供了必要的安全保障。

8. 良好的可移植性

Linux 是一种可移植的操作系统，能够在从微型计算机到大型计算机的任何环境中和任何平台上运行。可移植性为运行 Linux 的不同计算机平台与其他任何机器进行准确而有效的通信提供了手段，不需要另外增加特殊的和昂贵的通信接口。

正是由于以上特点，使 Linux 在短时间内获得了飞速的发展，成为计算机发展史上的一个奇迹。

1.1.3 Linux 的版本

任何一个软件都有版本号，例如，微软的 Windows7，Office 2007 等，Linux 也不例外。Linux 的版本号又分为两部分：内核（Kernel）与发行套件（Distribution）版本。

Linux 的内核是系统的心脏，内核包括了几百万行代码，是运行程序和管理硬件设备的核心程序。没有内核，就不能运行程序，但内核不是操作系统的全部。Linux 初学者常会把内核版本与发行套件版本弄混了，实际上，内核版本指的是在 Linus 领导下的开发小组开发出的系统内核的版本号。Linux 的每个内核版本使用形式为 x.y.zz-www 的一组数字来表示。其中，x.y 为 Linux 的主版本号，zz 为次版本号，www 代表发行号（注意，它与发行版本号无关）。当内核功能有一个飞跃时，主版本号升级，如 Kernel2.2、2.4、2.6 等。内核增加了少量补丁时，常常会升级次版本号，如 Kernle2.6.15、2.6.20 等。当然还有更复杂的版本号系统，如 2.6.20-32 等。通常 y 若为奇数，表示此版本为测试版，系统会有较多 bug，主要用途是提供给用户测试。随着每一次系统的小的 bug 修正，zz 会增加。编写本书时，Linux 的内核最新稳定版本号是 3.0.3（主版本号 3.0 表明它是可以使用的稳定版本）。

一般而言，一个基本的 Linux 只是包含了 Linux 核心（Kernel）和 GNU 软件的一些基本的系统软件和实用工具（Utilities），这样一个操作系统仅仅能够让那些 Linux 专家完成一些很基本的系统管理任务，若要满足普通用户的办公或基于视窗的应用开发等需要，则还需要在系统中加入 GNOME、KDE 等桌面环境，以及相应的办公应用软件（如 Office）等。因此，一些组织或厂家将 Linux 系统内核与 GNU 软件（系统软件和工具）整合起来，并提供一些安装界面和系统设定与管理工具，这样就构成了一个发行套件，例如，最常见的 Ubuntu，Fedora 等。实际上发行套件就是 Linux 的一个大软件包而已，通常包括 C 语言及 C++的编译器、Perl 脚本解释程序、Shell 命令解释器、图形用户界面，以及众多的应用程序等。相对于内核版本，发行套件的版本号随发布者的不同而不同，与系统内核的版本号是相对独立的。因此，把 Ubuntu，Fedora 等直接说成是 Linux 是不确切的，它们是 Linux 的发行版本，更确切地说，应该称为“以 Linux 为核心的操作系统软件包”。根据 GPL 准则，这些发行版本虽然都源自一个内核，并都有自己各自的贡献，但都没有自己的版权。Linux 的各个发行版本，都是使用 Linux 主导开发并发布的同一个 Linux 内核，因此，在内核层不存在兼容性问题。至于每个版本都有不一样的感觉，只是在发行版本的最外层才有所体现，而绝不是本身，也不是内核不统一或不兼容。

目前，Linux 的发行版很多，其中，比较流行的国外版本有 Ubuntu、Fedora、Slackware、Debian、OpenSUSE 和 Mandriva 等；国内的有红旗 Linux 和 TurboLinux 等。

1.1.4 Linux 的发展前景

Linux 以其独立、开放、安全、免费、强大的网络功能等特点，已在各个行业得到了广泛的应用，同时，Linux 的嵌入式和中间件方式具有优秀的移植性，利用 Linux 系统来进行软件开发已经成为一种趋势。可以想象，Linux 的发展前景非常可观。

1.2 Linux 的安装

“工欲善其事，必先利其器”。要进行 Linux 下的编程，必须先安装 Linux 系统。在安装之前，首先需要选择 Linux 的发行版本。

1.2.1 发行版本的选择

Linux 有多个发行版本可供选择。Ubuntu 是一个相对较新的发行版本，它的出现可能改变了许多潜在用户对 Linux 的看法。也许，从前人们会认为 Linux 难以安装、难以使用，但是，Ubuntu 出现后，这些都成为了历史。Ubuntu 的名称来自非洲南部祖鲁语或豪萨语的“ubuntu”一词（译为吾帮托或乌班图），意思是“人性”、“我的存在是因为大家的存在”，是非洲传统的一种价值观，类似华人社会的“仁爱”思想。Ubuntu 以 Debian Linux 为基础，针对桌面应用进行了优化。Ubuntu 的安装非常的人性化，只要按照提示一步一步进行，安装和 Windows 同样简便！并且，Ubuntu 被誉为对硬件支持最好最全面的 Linux 发行版之一，许多在其他发行版上无法使用，或者默认配置时无法使用的硬件，在 Ubuntu 上轻松搞定。并且，Ubuntu 默认不能直接 root 登录，必须从第 1 个创建的用户通过 su 或 sudo 来获取 root 权限（这也许不太方便，但无疑增加了安全性，避免用户由于粗心而损坏系统）。Ubuntu 的版本周期为 6 个月，弥补了 Debian 更新缓慢的不足。因此，在本书中以 Ubuntu 作为 Linux 程序的开发平台。写作本书时，Ubuntu 的最新发行版本为 11.04，读者可以从 Ubuntu 官方网站 http://www.ubuntu.com/免费下载发行版本的安装文件。

1.2.2 安装虚拟机软件

很多读者日常使用的操作系统是 Windows，而要在 Linux 下进行编程，又必须要安装 Linux 系统。解决的方法有两种：一种方法是在计算机上安装双系统，这种方法必须在开机时选择要进入的操作系统，如果要切换到另一种系统，只能重新启动计算机。对于经常需要在两种系统之间切换的人来说，这种方法是很不方便的。另一种方法是在 Windows 或 Linux 系统中安装虚拟机软件，然后在虚拟机软件中安装另外的操作系统。关于虚拟机，本书不做过多介绍，读者可以自行查阅相关资料。比较有名的虚拟机软件有 VMware、VirtualPC 等。笔者就是在 Windows 系统下安装了 VMware，然后在 VMware 上安装了 Ubuntu。如果读者

日常使用的系统就是 Linux，则可以省略这一过程。

1.2.3　安装 Ubuntu

在虚拟机上安装 Ubuntu 的过程在本书所带的光盘中有相应的视频教程，这里就不再介绍。

1.3　Linux 系统的常用命令

在 Linux 图形界面下，可以完成大部分工作，然而许多 Linux 功能在命令行界面下要比在图形界面下完成得更快。本节先介绍 Shell 的概念，然后介绍 Shell 的命令行操作界面，最后介绍常用的 Shell 命令。

1.3.1　了解 Shell

在学习 Linux 常用命令之前，首先应该了解一下这些命令的运行环境。这个环境就是 Shell，它是一种命令解释器，在用户和操作系统之间提供了一个交互接口。用户在命令行输入命令，然后 Shell 对该命令进行解释并将它作为指令代码发送给操作系统。Shell 的这种解释功能提供了许多高级特性。例如，Shell 有一套能产生文件名的通配符，Shell 不仅能够对输入/输出（I/O）重定向，而且能够在后台执行命令，从而使用户同时可以进行其他任务的操作。在 Linux 操系统中有许多可选的 Shell。每种 Shell 提供不同的特性和功能，大多数 Shell 有自己的脚本语言，使用脚本语言可以建立复杂的自动执行程序。由于 Ubuntu 的默认 Shell 是 Bash，因此，本节介绍的命令也是在 Bash Shell 环境下运行。

1.3.2　进入 Shell 命令行界面

Shell 是终端下的用户操作界面。Linux 终端又称虚拟控制台，如果读者使用过 UNIX，那么对此一定不会陌生。显示器和键盘合称为终端，因为它们可以对系统进行控制，所以又称控制台，一台计算机的 I/O 设备就是一个物理控制台。如果在一台计算机上用软件的方法实现多个互不干扰、独立工作的控制台界面，就是实现了多个虚拟控制台。Linux 终端采用字符命令行方式工作，用户通过键盘输入命令，通过了 Linux 终端对系统进行控制。通常情况下，Linux 默认启动 6 个虚拟终端，如果启动方式选择是直接启动 X Window，则 X Window 是在第 7 个虚拟终端上。因此，Ubuntu 的 X Window 就是在第 7 个虚拟终端上。

在 X Window 图形界面中，按下 Alt+Ctrl+Fn（n=1~6）（虚拟机中是 Ctrl+Alt+Shift+Fn）就可以进入控制台字符操作界面。在控制台字符操作界面下，按下 Alt+Fn（n=1~6）可以切换到其他字符操作界面，在控制台字符操作界面下，可以按下 Alt+F7 返回 X Window 图形

界面。此外，在 X Window 图形界面中，Linux 还提供了图形终端。在 Ubuntu 中进入图形终端的方法是选择“应用程序”|“附件”|“终端”命令，打开图形终端窗口，如图 1-1 所示。

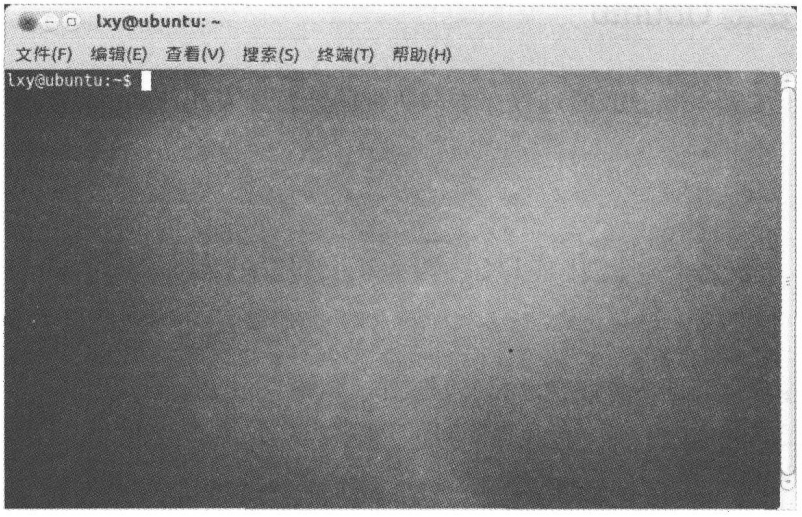

图 1-1 Ubuntu 图形终端窗口

其中，lxy@ubuntu:~$称为 Shell 的命令提示符；“@”前面的字符表示登录的用户名，“@”后面的字符表示登录的计算机名。所以提示符 lxy@ubuntu:~$表示的意思是指登录的用户是 lxy；登录的计算机是 ubuntu。另外，$是普通用户的提示符，而#是超级用户提示符。

出现命令提示符后，就可在光标处输入命令。每条命令输入完毕后，必须按回车键才会执行。如果输入的命令中有某个字符需要删除或修改，可以用左右方向键将光标移到要修改字符的后面或前面，再按 Back Space 或 Del 键删除，然后再输入正确的字符。如果想调用以前输入过的命令，可向上或向下的方向键进行选择。如在命令提示符后输入 date，按回车键，系统就会显示当前的日期和时间，如下：

```
1.  $ date
2.  2011 年 09 月 13 日 星期二 06:27:47 PDT
3.  $
```

第 1 行是输入的 date 命令，第 2 行是系统对命令的响应，这里显示的是当前日期及时间。命令响应完成后，系统又返回到等待输入命令的状态，如第 3 行所示。

又如询问当前有哪些用户挂在系统里。命令及响应如下：

```
1.  $ who
2.  lxy      tty7         2011-09-13 06:22 (:0)
3.  lxy      pts/0        2011-09-13 06:22 (:0.0)
```

从显示结果可以看出，当前系统中只有一个用户 lxy，分别在 tty7 虚拟终端和 pts/0 虚拟终端登陆，其中，tty7 为 X Window 图形界面所在的虚拟终端，而 pts/0 则为 X Window 下的图形终端。

在字符终端下和在图形终端下的操作是没有区别的，读者可以根据自己的兴趣选择相应的终端操作类型。