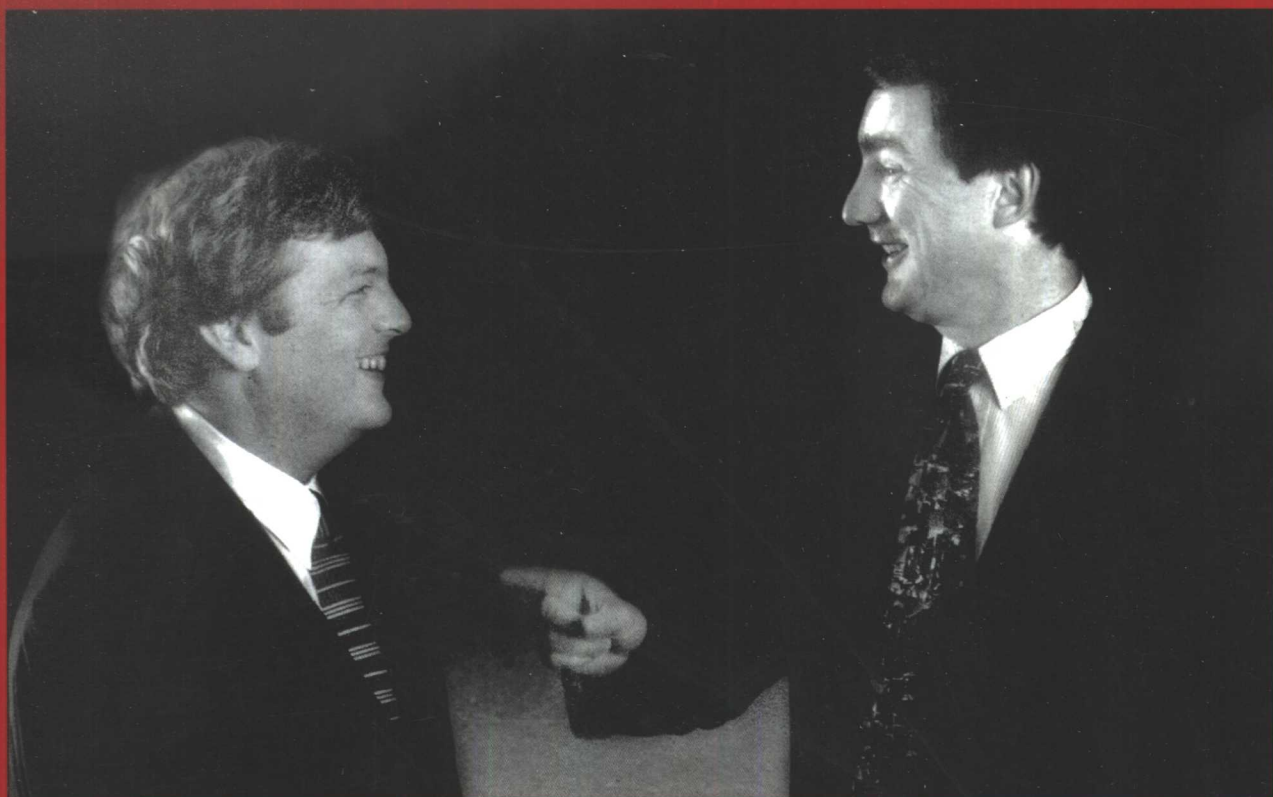




PROGRAMMER TO PROGRAMMER™
Beginning Linux Programming,
2nd Edition

Wrox 程序员参考系列



(原书第2版)

Linux

程序设计

(英) Neil Matthew Richard Stones 著

杨晓云 王建桥 杨涛 高文雅 等译

p2p.wrox.com 提供在线支持



机械工业出版社
China Machine Press



乐思

Wrox 程序员参考系列

Linux 程序设计

(原书第2版)

(英) Neil Matthew 著
Richard Stones

杨晓云 王建桥 杨 涛 高文雅 等译



机械工业出版社
China Machine Press

本书介绍了Linux操作系统的基本知识, 以及如何在Linux和其他UNIX风格的操作系统上进行程序开发。本书的主要内容包括: Linux的入门知识, 使用Shell进行程序设计, Linux的管理及开发工具, Perl程序设计语言, Web程序设计等等。本书内容丰富、深入浅出、易于理解, 还包含大量编程实例。适合Linux的初学者及希望利用Linux进行开发的程序设计人员阅读。

Neil Matthew and Richard Stones: *Beginning Linux Programming*, 2nd Edition.

Authorized translation from the English language edition published by Wrox Press.

Original copyright © 2000 by Wrox Press. All rights reserved.

Chinese simplified language edition published by China Machine Press.

本书中文简体字版由英国乐思出版公司授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

版权所有, 侵权必究。

本书版权登记号: 图字: 01-2000-4096

图书在版编目(CIP)数据

Linux程序设计(原书第2版)/(英)马太(Matthew, N.)等著; 杨晓云等译. -北京: 机械工业出版社, 2002.1

(Wrox程序员参考系列)

书名原文: *Beginning Linux Programming*, 2nd Edition

ISBN 7-111-09322-4

I. L ... II. ①马... ②杨... III. Linux操作系统-程序设计 IV.TP316-89

中国版本图书馆CIP数据核字(2001)第067258号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑: 宋宏 张鸿斌

北京第二外国语学院印刷厂印刷·新华书店北京发行所发行

2002年1月第1版第1次印刷

787mm × 1092mm 1/16 · 50印张

印数: 0 001-4 000册

定价: 78.00元

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

序 言

每一个计算机程序设计人员都有他自己的一套拿手绝活。每个人都有他自己的程序代码宝库，所有这些来源于他们自己对程序设计语言手册的学习理解，或者来源于通过Usenet学习到的方法。然而，对于那些对程序设计一无所知的人来说，不敢照搬照抄Usenet。但采用本书这样一种风格（指从小程序段开始介绍学习程序设计的风格）的著作可以说少之又少，这不能不说是一种奇怪的现象。在网络世界里存在着大量短小精悍而又切中问题核心的文档，它们直指程序设计或系统管理某些特定的领域。就拿Linux操作系统的文档项目来说吧，它的总篇幅大概只有3~10页，但它所涉及的内容却是包罗万象，从在同一台计算机上同时安装Linux和NT到把你的咖啡机接驳到Linux，可以说是应有尽有。真的，我一点也没有夸大其词。到<http://sunsite.unc.edu/LDP>网址上看看那里的mini-how-to（快速入门）索引就知道了。

从另一方面看，我们目前能够见到的书刊大多数或者可以说是一些学术墓园——它们既详尽又过于全面，读者很难有时间把它们一读到底，只有很少的书籍尝试着对大量实际应用领域的概念和基本做法进行介绍和讨论。本书就是这类书籍之一，它是对程序员个人编程宝库的补充，是通过对程序代码的“破译”（从程序员的手稿里读出正确的内容）而有机地组织起来的。

这本书目前已经是经过修改的第二版了，又扩展了许多内容，因为Linux本身也是在不断扩展中的。增加的主要内容有如何编写线程化程序（有时候我们把多线程程序设计形象地称为“一箭双雕”）和对GTK工具包的介绍，后者是GNOME版GUI程序设计的基础，并且还可能是用C语言编写X窗口界面应用程序最简单的办法。

书中还涉及到大量的Perl语言知识。有许多人认为Perl语言的春天已经到来，但也有些像我这样的人认为Perl的春天准确地说曾经是来临过，只是很久以前就过去了。当然，我个人的观点并不重要，虽说有点过时，可Perl确实是一种功能极为强大的脚本程序设计语言。所有Linux程序员（特别是那些为网络设计CGI脚本程序的人们）迟早会遇到用Perl语言编写的东西，所以这本书里出现大量的Perl程序也就不足为奇了。

本书的最后一章是你加入内核设计人员行列的一个机会。正如你将会看到的，内核程序的设计工作与为大型应用程序而进行的程序模块设计工作并没有太大的区别。戴上你的旧帽子，留出个大胡子，再喝着可乐，让我们加入到快乐的行列里来吧！

Alan Cox

参加本书翻译的人员除封面署名外还有：杨楨和、王玉敏、张玉亭、韩兰、李京山、张乔、李松、张雁东、韩东生、林红、郭明、宋建生、刘厢、郝宏志、郑志生、孙晓迪、苗昔、王大佑、董明才、许椿、赵祥生、陈广学、黄济令、姜义春、沈萍、林超、吴铭、胡平建、曹安、周友远

前言

这是一本易于阅读使用的指南性读物，其主要内容是在Linux和其他UNIX风格的操作系统上进行程序开发。

在这本书里，我们的目标是把对读者（即那些使用UNIX的开发人员）而言非常重要的内容广泛的论题介绍给大家。我们在本书的内容组织方面确实动了不少的脑筋，不管读者现有的经验如何，我们都希望它能够最大限度地帮助你尽可能多地学习到由UNIX提供出来的东西。UNIX系统程序设计的覆盖面是非常广的，我们的目标是对广泛领域内的大量论题都进行足够深度的讨论，让读者在每一个论题方面都学习到足够的“入门”知识。

这本书是为哪些人编写的？

如果你是一个程序设计员，希望利用UNIX（和Linux）为软件开发提供的各种手段加快程序设计进度，充分利用你在程序设计方面花费的时间，并且希望自己设计出来的程序能够最大限度地利用UNIX系统本身的潜力，挑上这本书算是选对了。书中明确清晰的论述和一个步骤一个步骤的试探及检验将帮助你迅速掌握所有关键性的技巧。

我们假设读者对UNIX操作系统所涉及的基本概念都有一定的了解，如果你还有一些非UNIX环境（比如MS-DOS或者微软公司的Windows环境）中的C或C++程序设计经验就更理想了。书中出现直接比较的地方都会在内容里标出来。

如果你刚开始学习UNIX，请注意：这不是一本介绍Linux安装和配置过程的书。如果你只是想多学习一些UNIX系统管理方面的知识以及一般意义上的UNIX概念和UNIX命令，那最好还是去阅读《Instant UNIX》（UNIX初学者）一书，那本书由本书两位作者和Andrew Evans合著，也是Wrox出版社出版的（国际书号是ISBN 1-874416-65-6）。

本书的目标有两个，一是作为指南性教材，向读者介绍大多数UNIX系统上都有的各种程序工具和函数/函数库系列，二是作为一本方便的速查资料。这本书的特点是方法实用，论述清晰，示例众多。

本书都讨论了哪些内容？

这本书希望能够达到以下几个目的：

- 介绍脱胎于IEEE POSIX和X/Open（SPEC 1170）等技术规范的UNIX98标准中所规定的UNIX操作系统的标准化C语言库及其他命令工具。
- 介绍如何制作编写大多数高级开发工具。
- 对流行的快速开发语言如shell、Tcl和Perl等进行简明的介绍。

- 介绍如何为X窗口系统编写图形化的用户操作界面。我们将对普通X系统上使用的Tk工具包和GNOME窗口系统中使用的GTK+工具包进行介绍。
- 在读者打下坚实的基础之后，我们将进一步介绍现实世界中的应用程序，这部分内容可能是广大读者最感兴趣的东西。

在讨论以上这些问题的时候，我们的做法将是先介绍基本的理论知识，然后用一个适当的例子和清晰的程序注释对具体问题加以阐述。在第一遍学习的时候你就能够迅速地学习到有关的知识。如有必要，你还可以回过头来仔细琢磨，进一步体会其中各种基本成分的奥妙。

书中每一个小程序示例的目的主要是演示一组函数的基本用法，或者是为了演示某些新理论概念的具体实现方法，但它们组合在一起将构成一个大型的示范性项目成果：即一个记录音乐CD唱盘详细资料的简单的数据库应用程序。随着知识面的扩展，你可以随意对该项目成果做进一步的开发、重新实现、或者扩充。尽管如此，书中每一章的内容又是相对独立的，所以读者可以根据自己的情况做跳跃性学习，但我们认为各章内容还有着一定的连贯性，前面的内容为我们将要讨论的东西提供了一些有用的示范和预备知识。在各章内容的编排方面我们尽量做到循序渐进，后面的论题肯定要比其前面的内容更高深一些。

在本书里，第一次出现的应用程序示例在shell程序设计这一章节的末尾，它向我们展示了一个比较大型的shell脚本程序是如何组织在一起的，我们将学习到如何通过shell对用户输入进行处理、如何通过shell构造菜单，以及shell是如何保存和检索数据的。

在简单回顾了编译程序、库函数链接以及在线手册等几个方面之后，我们先对shell程序设计做了一个简单的介绍。随后我们将一直使用C语言进行程序设计，讨论内容将包括利用文件进行工作、从UNIX环境里获取信息、对终端的输入输出进行处理以及curses函数库（它使交互式的输入和输出更具可扩展性）等。最终我们得以尝试使用C语言再一次实现CD唱盘管理软件。这个软件的设计思路没有什么变化，但新编写的代码使用了curses函数库向用户提供了一个基于屏幕的操作界面。

接下来，我们开始讨论数据管理方面的问题。为了学习dbm数据库函数库的使用方法，我们将再次实现这个应用软件，而这次实现中所采用的设计思路将贯穿本书其余的章节。这个应用软件的用户操作界面被单独保存在一个文件里，而CD唱盘的数据库则是另外一个程序。经过我们的改进，如今的数据库资料已经是关系化的了。

这些新应用软件的规模已经比较大了，所以我们接下来要对付一些软件维护方面的问题，比如代码调试、源代码控制、软件的发行传播以及制作文件等。

第10章是这本书的一个分水岭。在这一章里，我们将对程序在运行期间的行为特点以及如何让它们按照我们的意愿去执行等问题进行学习。程序进程可以分割和变形，同时它们也开始在彼此之间发送信号。我们还讨论了POSIX线程，并将看到如何在一个进程里创建多个执行线程。

在掌握了多进程概念之后，CD唱盘管理软件就可以被实现为一个客户端和一个服务器端，两者通过几种可靠的手段实现了相互通信。客户/服务器格局的应用软件被实现了两次，但数据库本身和用户操作界面还是相同的，只是在添加间接通信层的时候分别使用了两种办法：管道和System V的IPC。为了使这一部分内容更加完整，我们接下来又研究了套接字，并使用一个TCP/IP网络完成了进程间的通信。

再往下开始了Tcl/Tk方面的学习，我们在那里介绍了Tcl shell并通过Tk创建了各种各样的X用户操作界面。然后我们介绍了如何通过GIMP工具包（GTK+）为GNOME窗口环境开发应用软件，我们在这部分内容里开发了一个桌面时钟作为示例。

接下来，我们开始研究因特网，首先是HTML语言，然后是通用网关接口（Common Gateway Interface）；后者允许我们间断性地访问某个应用程序。这一次，我们通过远程Web浏览器来访问CD唱盘软件的用户操作界面，然后通过Web浏览器去访问由CGI程序生成的网络页面，客户和CGI程序之间隔着Web服务器软件Apache。

在这本书的最后部分，我们介绍了编写设备驱动程序的方法——这是通向理解Linux内核本身道路上的重要一步。

由于作者水平有限，在这本书里难免有所疏漏，但我们希望在我们讨论到的内容方面能够给读者一个比较清晰的概念。

学习这本书需要准备哪些东西？

在这本书里，我们向大家介绍的主要是UNIX操作系统下的程序设计。为了帮助大家更好地理解各章的内容，我们真心希望大家在阅读本书的时候要使用我们给出的程序设计示例对照学习。这些程序设计示例体现了良好的程序设计经验，指导读者在今后编写出自己的程序。

Linux发行版本是在UNIX环境下进行程序设计的理想办法，它带有一个比较完整的开发环境，其中包括GNU的C/C++编译器、开发辅助工具以及其他有用的小东西。它是自由传播的、符合POSIX规范的、健壮的、不断开发完善的，而且它的功能也十分强大。

Linux可以用在许多不同的系统上。Linux的适应性可以用这样一句话来形容：在人们的努力下，只要是其中有一个处理器芯片的东西，Linux就能以这样或那样的形式来运行！目前能够运行Linux的系统包括基于Alpha、SPARC、ARM、PowerPC和68000系列CPU的计算机，当然也包括了采用英特尔公司x86和Pentium X系列芯片（及其兼容芯片）的现代个人电脑。

我们在编写这本书的时候使用的是基于英特尔公司CPU芯片的系统，但我们讨论的内容很少是只适用于英特尔芯片的。虽然在一台只有2MB内存且没有硬盘的386机器上运行Linux也是可能的（确实可以运行），但要想成功地运行Linux并试验书中的程序示例，我们推荐至少要有以下的配置：

- 奔腾级处理器。
- 32MB内存。
- 600MB可用硬盘空间，并且最好是在同一个分区里。
- 如果想运行X窗口系统，还需要有一块它支持的显卡。

X窗口系统所支持的显卡的资料可以在<http://www.xfree86.org/>网址处查到。

运行本书大部分章节中的代码所需要的硬件配置其实是相当小的，只有涉及X窗口系统的有关章节才需要比较强大的配置（或更多的耐心）！

我们使用了两台不同配置的Linux系统来编写本书的内容和开发那些程序示例，因此我们确信只要Linux能够运行在你的机器上，你就可以顺利地完成本书的学习。另外，在本书的技术校对期间，我们在另外一个Linux版本上完成了全部代码的测试工作。

在软件要求方面需要提醒读者注意的是，程序示例中有一小部分需要比较新的Linux内核版本，即2.2或更高版本才能顺利运行。Java开发工具包（Java Development Kit）要求使用最新版本的GCC和C语言库（glibc 2或更高）。至于其他工具软件，最好的办法是设法获得它们最新的版本。比如说，Tcl和Tk章节中的程序示例分别需要在7.5和8.0版本以上才能顺利运行。我们会在必要时提醒大家注意最低配置要求，如果读者在代码运行方面遇到了麻烦，使用比较新一些的工具可能会有所帮助。好在所有这些软件工具都可以很方便地下载到，而且，我们在附录C里已经准备了一份因特网资源指南以帮助读者找到它们。如果读者使用的是最近推出的Linux发行版本，就应该不会遇到什么问题。

因为Linux、GNU工具包以及其他软件都是在GPL版权规定下发行的，所以它们是有一些特点的，其中之一就是所谓的“自由性”。它们的源代码永远是公开的，任何人都不能抹杀这种自由性。因此，它们也都是些“源代码开放”软件，“源代码开放软件”这个术语的含义要比“自由软件”的含义宽一些，因为有些专利软件在某种规定的条件下也会提供其源代码。在GNU/Linux世界里，你永远能够得到技术支持——你可以自己动手修改源代码，也可以雇佣其他人。目前，为Linux及其相关软件工具提供收费技术支持的公司正在不断增加。

实例代码

我们已经尽了最大的努力向读者提供能够最好地说明书中有关概念的示范性程序和代码段。

特别需要指出的是，我们并没有对我们调用过的每一个函数的返回值是否就是我们所预期的进行检查。在真正的应用软件成品代码里我们当然会对函数的返回值进行检查，而读者也必须这样做，因为只有这样才能为程序的错误处理提供一个有效的手段。我们在书中的第3章里介绍了一些捕获和处理程序错误的办法。

本书所有程序代码都可以从下面这个网址下载到：

<http://www.wrox.com>

书中的所有代码都遵守GNU的公共许可证（GNU Public License）。我们建议读者下载一份全部程序代码的拷贝，这样可以节省你不少的打字时间。

体例

为了让读者从这本书里学习到更多的知识和跟上书中论述的内容，我们在这本书里使用以下一些编写体例。

书中的文字框里是一些重要的，不应该被忘记的内容。它们与其周边的内容息息相关，就像电影《谍中谍》中特工们准备窃取的关键资料。

我们在程序示例开始之前会加上一个“动手试试”标记，目的是为了帮助大家把示范性的代码部分与我们准备介绍学习的代码区分开，并且让它们在书中的内容里更加明显，让读者看清楚应用程序的编写进度。在重要的场合，我们还会在代码部分的后面加上一个“操作注释”来解释与前面理论有关的代码中容易被混淆的关键之处。我们发现这两个标记能够把比较难于理解的代码清单分解为相对简单的部分。

告诉我们你的想法

我们已经努力使这本书正确而详尽，因此我们希望读者能够告诉我们本书就是你最想要和最需要的，让我们能够得到一丝满足；也希望读者对我们这本书的编排方面多提宝贵意见。

我们欢迎对我们这份努力的反馈意见，不管是批评还是赞扬，我们都将在今后的编辑工作中采纳。如果你有话要说，请按以下地址和我们联系：

Feedback@wrox.com

或者

<http://www.wrox.com>

现在就把这两个地址加到你的书签里去吧！

本书的英文书名为：**Beginning Linux Programming, 2nd Edition**

英文书号为：**ISBN 1-861002-97-1**

目 录

序言

前言

第1章 入门知识1

1.1 什么是UNIX操作系统1

1.2 什么是Linux操作系统1

1.3 发行版本2

1.4 GNU项目和自由软件基金会2

1.5 为Linux系统设计程序3

1.6 UNIX程序4

1.7 获得帮助6

1.8 程序开发系统的预备知识8

1.8.1 程序8

1.8.2 头文件8

1.8.3 库文件9

1.8.4 静态库10

1.8.5 共享库12

1.9 UNIX系统中程序设计的特点和原则13

1.9.1 简单性13

1.9.2 重点性13

1.9.3 可反复使用的程序组件13

1.9.4 过滤器14

1.9.5 开放的文件格式14

1.9.6 灵活适应性14

1.10 本章总结14

第2章 shell程序设计15

2.1 什么是shell16

2.2 管道和重定向17

2.2.1 对输出数据进行重定向17

2.2.2 对输入数据进行重定向18

2.2.3 管道19

2.3 可以被视为程序设计语言的shell19

2.3.1 交互式程序19

2.3.2 编写脚本程序20

2.3.3 把脚本设置为可执行程序21

2.4 shell程序设计的语法23

2.4.1 变量23

2.4.2 条件测试26

2.4.3 控制结构29

2.4.4 函数39

2.4.5 命令42

2.4.6 命令的执行52

2.4.7 即时文档55

2.4.8 调试脚本程序57

2.5 shell程序设计示例58

2.5.1 工作需求58

2.5.2 设计58

2.6 本章总结66

第3章 如何使用和处理文件67

3.1 UNIX的文件结构67

3.1.1 目录结构68

3.1.2 文件和设备68

3.2 系统调用和设备驱动程序70

3.3 库函数70

3.4 文件的底层访问71

3.4.1 write系统调用72

3.4.2 read系统调用72

3.4.3 open系统调用73

3.4.4 访问权限的初始化值74

3.4.5 umask变量75

3.4.6 close系统调用76

3.4.7 ioctl系统调用76

3.4.8 其他与文件管理有关的系统调用78

3.5 标准I/O库80

3.5.1 fopen函数81

3.5.2	fread函数	82	4.6	主机资料	119
3.5.3	fwrite函数	82	4.7	日志记录功能	121
3.5.4	fclose函数	82	4.8	资源和限制	124
3.5.5	fflush函数	83	4.9	本章总结	129
3.5.6	fseek函数	83	第5章	终端	130
3.5.7	fgetc、getc、getchar函数	83	5.1	对终端进行读写	130
3.5.8	fputc、putc、putchar函数	83	5.1.1	对重定向输出进行处理	133
3.5.9	fgets、gets函数	84	5.1.2	与终端进行“对话”	134
3.5.10	格式化输入和输出	84	5.2	终端驱动程序和通用终端接口	136
3.5.11	对数据流进行处理的其他函数	88	5.2.1	概述	136
3.5.12	文件流错误处理	89	5.2.2	硬件模型	137
3.5.13	文件流和文件描述符的关系	90	5.3	termios结构	138
3.6	文件和子目录的维护	90	5.3.1	输入模式	139
3.6.1	chmod系统调用	90	5.3.2	输出模式	140
3.6.2	chown系统调用	90	5.3.3	控制模式	140
3.6.3	unlink、link、symlink系统调用	91	5.3.4	本地模式	141
3.6.4	mkdir和rmdir系统调用	91	5.3.5	特殊的控制字符	141
3.6.5	chdir系统调用和getcwd函数	92	5.3.6	终端的速度	144
3.7	扫描子目录	92	5.3.7	其他功能函数	145
3.7.1	opendir函数	93	5.4	终端的输出	148
3.7.2	readdir函数	93	5.4.1	终端的类型	148
3.7.3	telldir函数	93	5.4.2	确定终端类型的方法	149
3.7.4	seekdir函数	93	5.4.3	terminfo的使用方法	151
3.7.5	closedir函数	94	5.5	检测键盘输入	155
3.8	错误处理	96	5.6	本章总结	158
3.9	高级论题	97	第6章	curses函数库	159
3.9.1	fcntl系统调用	97	6.1	使用curses函数库进行编译	159
3.9.2	mmap函数	98	6.2	基本概念	160
3.10	本章总结	100	6.3	操作的初始化和结束	163
第4章	UNIX环境	101	6.4	向屏幕输出数据	163
4.1	程序参数	101	6.5	从屏幕读取输入数据	164
4.2	环境变量	105	6.6	清除屏幕	164
4.2.1	环境变量的用途	107	6.7	移动光标	165
4.2.2	environ变量	107	6.8	字符的属性	165
4.3	时间与日期	108	6.9	键盘	167
4.4	临时文件	114	6.9.1	键盘的工作模式	167
4.5	用户的个人资料	116	6.9.2	键盘输入	168

6.10 窗口	169	8.2.2 make命令的选项和参数	243
6.10.1 WINDOW结构	170	8.2.3 制作文件中的注释	246
6.10.2 通用化函数	170	8.2.4 制作文件中的宏	246
6.10.3 移动和刷新窗口	171	8.2.5 多个制作目标	248
6.10.4 优化窗口的刷新操作	174	8.2.6 内建规则	250
6.11 子窗口	175	8.2.7 后缀规则	251
6.12 键盘上的数字小键盘	177	8.2.8 用make命令管理函数库	252
6.13 彩色显示功能	178	8.2.9 高级论题:制作文件和下级子目录	254
6.14 逻辑屏幕和显示平面	181	8.2.10 GNU的make和gcc命令	255
6.15 CD唱盘管理软件	182	8.3 源代码控制系统	256
6.16 本章总结	194	8.3.1 RCS系统	256
第7章 数据管理	196	8.3.2 SCCS系统	261
7.1 内存管理	196	8.3.3 CVS系统	262
7.1.1 简单的内存分配机制	196	8.4 编写使用手册	266
7.1.2 分配大量的内存	197	8.5 软件的发行传播	269
7.1.3 内存的滥用	200	8.5.1 patch程序	269
7.1.4 空指针	201	8.5.2 软件发行方面的其他工具	271
7.1.5 内存的释放	202	8.6 本章总结	273
7.1.6 其他内存分配函数	203	第9章 调试与纠错	274
7.2 文件封锁	204	9.1 错误的分类	274
7.2.1 创建锁文件	204	9.1.1 功能定义错误	274
7.2.2 文件中的封锁区	207	9.1.2 设计规划错误	274
7.2.3 封锁状态下的读写操作	209	9.1.3 代码编写错误	274
7.2.4 文件封锁的竞争现象	214	9.2 常用调试技巧	275
7.2.5 其他封锁命令	217	9.2.1 一个有漏洞的程序	275
7.2.6 死锁现象	217	9.2.2 代码审查	277
7.3 数据库	218	9.2.3 取样法	278
7.3.1 dbm数据库	218	9.2.4 程序的受控执行	280
7.3.2 dbm例程	219	9.3 用gdb进行调试纠错	281
7.3.3 dbm数据库的访问函数	220	9.3.1 启动gdb	281
7.3.4 其他dbm函数	224	9.3.2 运行一个程序	282
7.4 CD唱盘管理软件	225	9.3.3 堆栈跟踪	282
7.5 本章总结	241	9.3.4 对变量进行检查	283
第8章 开发工具	242	9.3.5 列出程序清单	284
8.1 多个源文件带来的问题	242	9.3.6 设置断点	284
8.2 make命令和制作文件	243	9.3.7 用调试器打补丁	287
8.2.1 制作文件的语法	243	9.3.8 深入学习gdb	288

9.4 其他调试工具	288	11.6.3 schedparam属性	342
9.4.1 lint: 清理程序中的“垃圾”	289	11.6.4 inheritsched属性	342
9.4.2 函数调用工具	290	11.6.5 scope属性	342
9.4.3 执行记录	291	11.6.6 stacksize属性	343
9.5 假设验证	292	11.6.7 线程属性——调度	344
9.6 内存调试	294	11.7 取消一个线程	345
9.6.1 ElectricFence	294	11.8 多线程	348
9.6.2 Checker	295	11.9 本章总结	350
9.7 资源	297	第12章 进程间通信: 管道	351
9.8 本章总结	297	12.1 什么是管道	351
第10章 进程与信号	298	12.2 进程管道	352
10.1 什么是进程	298	12.2.1 popen函数	352
10.2 进程的结构	298	12.2.2 pclose函数	352
10.2.1 进程表	300	12.2.3 把输出送往popen	353
10.2.2 查看进程	300	12.3 pipe函数	356
10.2.3 系统进程	301	12.4 父进程和子进程	359
10.2.4 进程的调度	302	12.4.1 管道关闭后的读操作	361
10.3 启动新的进程	303	12.4.2 把管道用做标准输入和标准输出	361
10.3.1 等待进程	308	12.5 命名管道: FIFO文件	364
10.3.2 僵进程	310	12.6 高级论题: 以FIFO文件为基础的客户 /服务器架构	372
10.3.3 输入和输出重定向	312	12.7 CD唱盘管理软件	375
10.3.4 线程	313	12.7.1 目标	376
10.4 信号	313	12.7.2 实现	376
10.4.1 发送信号	317	12.7.3 客户接口函数	380
10.4.2 信号集	321	12.7.4 服务器接口	385
10.5 本章总结	325	12.7.5 管道	389
第11章 POSIX线程	326	12.7.6 对CD唱盘管理软件的总结	394
11.1 什么是线程	326	12.8 本章总结	394
11.2 检查有无线程支持	327	第13章 信号量、消息队列和共享内存	395
11.3 第一个线程程序	329	13.1 信号量	395
11.4 同时执行	332	13.1.1 信号量的定义	396
11.5 同步	333	13.1.2 一个理论性的例子	396
11.5.1 用信号量进行同步	333	13.1.3 UNIX中的信号量功能	397
11.5.2 用互斥量进行同步	337	13.1.4 使用信号量	400
11.6 线程的属性	341	13.1.5 信号量总结	403
11.6.1 detachedstate属性	342	13.2 共享内存	403
11.6.2 schedpolicy属性	342		

13.2.1 概述	403	15.1.4 引用和替换	452
13.2.2 共享内存函数	404	15.1.5 计算	455
13.2.3 共享内存总结	409	15.1.6 控制结构	456
13.3 消息队列	409	15.1.7 错误处理	458
13.3.1 概述	409	15.1.8 字符串操作	459
13.3.2 消息队列函数	410	15.1.9 数组	464
13.3.3 消息队列总结	414	15.1.10 列表	465
13.4 应用示例	414	15.1.11 过程	470
13.5 查看IPC功能状态的命令	418	15.1.12 输入和输出	471
13.5.1 信号量	418	15.2 一个Tcl程序	474
13.5.2 共享内存	418	15.3 创建一个新Tcl语言	477
13.5.3 消息队列	419	15.4 Tcl语言的扩展	478
13.6 本章总结	419	15.4.1 expect	478
第14章 套接字	420	15.4.2 [incr Tcl]	478
14.1 什么是套接字	420	15.4.3 TclX	478
14.2 套接字连接	420	15.4.4 图形	478
14.2.1 套接字属性	424	15.5 本章总结	479
14.2.2 创建一个套接字	426	第16章 X窗口系统的程序设计	480
14.2.3 套接字地址	427	16.1 什么是X	480
14.2.4 给套接字起名字	427	16.1.1 X服务器	480
14.2.5 创建套接字队列	428	16.1.2 X协议	481
14.2.6 接受连接	428	16.1.3 Xlib库	481
14.2.7 请求连接	429	16.1.4 X客户	481
14.2.8 关闭一个套接字	430	16.1.5 X工具包	481
14.2.9 套接字通信	430	16.2 X窗口管理器	482
14.2.10 主机字节顺序和网络字节顺序	432	16.3 X程序设计模型	483
14.3 网络信息	434	16.3.1 启动	483
14.3.1 因特网守护进程	438	16.3.2 主循环	484
14.3.2 套接字选项	439	16.3.3 退出整理	485
14.4 多客户	439	16.4 X程序设计概述	485
14.5 select系统调用	442	16.5 Tk工具包	485
14.6 本章总结	447	16.5.1 窗口程序设计概述	487
第15章 工具命令语言Tcl	449	16.5.2 配置文件	488
15.1 Tcl语言概述	449	16.5.3 其他命令	489
15.1.1 第一个Tcl程序	449	16.5.4 Tk素材	489
15.1.2 Tcl命令	450	16.5.5 Tk内建的对话框	515
15.1.3 变量和值	451	16.5.6 颜色方案	518

16.5.7 字体	519	18.4.1 CPAN	607
16.5.8 绑定	520	18.4.2 安装一个模块	607
16.5.9 bindtags命令	521	18.4.3 perldoc命令	607
16.5.10 几何尺寸管理	523	18.4.4 网络功能	608
16.5.11 焦点及其切换	525	18.4.5 数据库	609
16.5.12 选项数据库	527	18.5 改进版CD唱盘数据库	609
16.5.13 应用程序间的通信	528	18.6 本章总结	613
16.5.14 selection命令	528	第19章 因特网程序设计: HTML	614
16.5.15 Clipboard命令	529	19.1 什么是World Wide Web	614
16.5.16 窗口管理器	530	19.2 术语	615
16.5.17 动态/静态加载	531	19.2.1 超文本传输协议	615
16.5.18 Safe Tk	532	19.2.2 因特网邮件多媒体扩展	615
16.6 一个复合素材	533	19.2.3 标准通用置标语言	615
16.7 使用复合型树素材的应用程序	543	19.2.4 文档类型定义	615
16.8 Tk进程记录查看器	544	19.2.5 超文本置标语言	615
16.8.1 国际化	554	19.2.6 可扩展置标语言	616
16.8.2 业界动态	554	19.2.7 层叠样式表	616
16.9 本章总结	555	19.2.8 可扩展超文本置标语言	616
第17章 使用GTK+进行GNOME程序 设计	557	19.2.9 统一资源定位器	617
17.1 GNOME简介	557	19.2.10 统一资源标识符	617
17.1.1 GNOME的体系结构	558	19.3 一个HTML文档	617
17.1.2 GNOME桌面	560	19.4 深入学习HTML	618
17.1.3 在GNOME里利用Gtk+设计程序	560	19.4.1 HTML标签	620
17.1.4 GNOME应用程序	575	19.4.2 图像	625
17.2 本章总结	583	19.4.3 表格	628
第18章 Perl程序设计语言	584	19.4.4 锚点或超链接	631
18.1 Perl语言简介	584	19.4.5 给图像加上锚点	633
18.1.1 “Hello” Perl程序	585	19.4.6 非HTML的URL地址	635
18.1.2 Perl语言中的变量	585	19.4.7 链接到其他站点	635
18.1.3 操作符和函数	588	19.5 编写HTML文件	637
18.1.4 规则表达式	593	19.6 HTML页面服务	638
18.1.5 控制结构和子例程	596	19.6.1 网络中的HTML文档	638
18.1.6 文件的输入和输出	599	19.6.2 设置一个服务器	639
18.2 一个完整的例子	600	19.7 可点击图片	640
18.3 命令行上的Perl	605	19.7.1 服务器端可点击图片	640
18.4 模块	607	19.7.2 客户端可点击图片	641
		19.8 服务器端的预处理功能	641

19.9 编写WWW主页的技巧	644	21.3.7 等待队列	700
19.10 本章总结	645	21.3.8 文件操作write: 向设备写入数据	702
第20章 因特网程序设计II: CGI	646	21.3.9 非阻塞性读操作	702
20.1 表元素	647	21.3.10 查找操作	703
20.1.1 FORM标签	647	21.3.11 文件操作ioctl: I/O控制	704
20.1.2 INPUT标签	647	21.3.12 检查用户权限	706
20.1.3 SELECT标签	649	21.3.13 文件操作poll: 设备对进程的调度	706
20.1.4 TEXTAREA标签	650	21.3.14 模块的参数	709
20.2 一个主页示例	650	21.3.15 proc文件系统接口	710
20.3 向WWW服务器发送信息	653	21.3.16 Schar的执行情况	712
20.3.1 对信息进行编码	653	21.3.17 小结	713
20.3.2 服务器程序	654	21.4 定时和时基: jiffies变量	713
20.3.3 编写服务器端的CGI程序	654	21.4.1 短暂延时	715
20.3.4 使用扩展URL的CGI程序	660	21.4.2 定时器	715
20.3.5 对表单数据进行解码	662	21.4.3 让出处理器	718
20.4 向客户返回HTML	668	21.4.4 任务队列	719
20.5 技巧与窍门	671	21.4.5 预定义任务队列	719
20.5.1 确保CGI程序能够退出	671	21.4.6 小结	721
20.5.2 对客户进行重定向	671	21.5 内存管理	721
20.5.3 动态图形	671	21.5.1 虚拟内存区	722
20.5.4 隐藏上下文信息	672	21.5.2 地址空间	722
20.6 一个应用程序	672	21.5.3 内存地址的类型	723
20.7 应用Perl语言	678	21.5.4 在设备驱动程序里申请内存	723
20.8 本章总结	683	21.5.5 在用户空间和内核空间之间传递 数据	725
第21章 设备驱动程序	685	21.5.6 简单的内存映射	727
21.1 设备	685	21.5.7 I/O内存	729
21.1.1 设备的分类	686	21.5.8 I/Omap里的设备分配	730
21.1.2 用户空间与内核空间	687	21.5.9 对I/O内存实现mmap文件操作	731
21.2 字符设备	693	21.6 I/O端口	733
21.3 字符设备驱动程序示例: Schar	695	21.6.1 可移植性	734
21.3.1 MSG宏命令	696	21.6.2 中断处理	734
21.3.2 字符设备的注册	696	21.6.3 IRQ处理器	737
21.3.3 模块的使用计数	697	21.6.4 中断的后处理	738
21.3.4 open和release: 设备的打开和 关闭	698	21.6.5 可重入性	739
21.3.5 文件操作read: 从设备读出数据	699	21.6.6 单独禁止一个中断	740
21.3.6 current任务	700	21.6.7 原子化操作	740

21.6.8 对关键节进行保护	741	21.8.5 远程调试	754
21.7 块设备	742	21.8.6 调试工作中的注意事项	754
21.7.1 一个简单的RAM盘模块: Radimo	743	21.9 可移植性	755
21.7.2 介质的更换	745	21.9.1 数据类型	755
21.7.3 块设备的ioctl文件操作	746	21.9.2 字节的存储顺序	755
21.7.4 请求函数: request	746	21.9.3 数据的对齐	756
21.7.5 缓冲区缓存	748	21.10 本章总结	756
21.7.6 小结	750	21.11 内核源代码解剖图	757
21.8 调试	750	附录A 可移植性	758
21.8.1 Oops追查法	751	附录B 自由软件基金会和GNU项目	766
21.8.2 对模块进行调试	753	附录C 因特网资源	772
21.8.3 “魔术键”	753	附录D 参考书目	779
21.8.4 内核调试器——KDB	754		