LINUX

- 由浅入深，按理论分析→实际操作→案例分析的顺序组织内容
- 作者多年教学及工程实践的总结，整合了Linux程序设计的绝大多数知识点，涵盖了Linux操作系统下C语言应用程序设计的所有关键内容

# 高级程序设计（第2版）

杨宗德 邓玉春 编著

Linux

人民邮电出版社
POSTS & TELECOM PRESS

# Linux高级程序设计（第2版）

杨宗德 邓玉春 编著



人民邮电出版社
北京

**图书在版编目（CIP）数据**

Linux高级程序设计 / 杨宗德，邓玉春编著. —2版.
北京：人民邮电出版社，2009.10
ISBN 978-7-115-21390-7

Ⅰ. L… Ⅱ. ①杨…②邓… 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（2009）第157806号

## 内 容 提 要

本书以 Linux 操作系统（内核为 2.6 版本）为开发平台、GCC 4.0/GDB 6.3 为开发调试环境，详细介绍了 Linux 系统下编程环境及编程工具、文件管理（文件类型、ANSI 以及 POSIX 标准下文件读写操作）、进程管理（创建、退出、执行、等待、属性控制）、进程间通信（管道、消息队列、共享内存）、进程间同步机制（信号量）、进程间异步机制（信号）、线程管理（创建、退出、取消等以及属性控制）、线程间同步（互斥锁、读写锁、条件变量）以及网络基本编程、高级应用等内容。

本书内容丰富、紧扣应用，适合从事 Linux 下 C 应用编程的人员阅读，也适合从事嵌入式 Linux 开发的人员阅读。

**Linux 高级程序设计（第 2 版）**

◆ 编　　著　杨宗德　邓玉春
责任编辑　刘　浩

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市潮河印业有限公司印刷

◆ 开本：787×1092　1/16
印张：27.75
字数：660 千字　　2009 年 10 月第 2 版
印数：5 001－9 000 册　　2009 年 10 月河北第 1 次印刷

ISBN 978-7-115-21390-7

定价：49.00 元

**读者服务热线：(010)67132692　印装质量热线：(010)67129223**
**反盗版热线：(010)67171154**

# 第 2 版前言

《Linux 高级程序设计》一书主要介绍 Linux 应用层程序开发中所涉及的系统调用，主要包括文件管理、进程管理、进程间通信与同步、线程管理、线程间同步以及网络编程等内容。要求读者有较好的 C 语言及数据结构基础、网络技术基础。

Linux 应用开发是目前最为广泛的软件开发内容之一，同时也是从事 Linux 内核及驱动开发的基础。《Linux 高级程序设计》一书出版以来，收到了大量的读者来信，对本书提出了各种意见和建议，综合各方面的考虑，笔者做了大量的改进，推出了它的第 2 版。这次修订的原则是：

（1）保持和加强原书优点，如理论观点鲜明，注重实际与应用，并添加新的案例；

（2）在内容上去旧更新，不仅修正了第 1 版中的错误和疏漏之处，更对原版的章节、内容进行了更新和补充，力求删繁就简。

本书修订内容如下。

第 1 章：精简文字，修订部分笔误，更新编程错误处理办法。

第 2 章：精简文字，修订部分笔误，删除部分冗余内容，包括对 GCC、GDB、Makefile 介绍的内容。增加部分工具说明。

第 3 章：精简文字，修订部分笔误，并对命令行参数匹配、进程基本环境等进行介绍。

第 4 章：修订格式化输入输出节内容，增加 sscanf/sprintf 函数应用内容，并增加部分应用案例，主要包括流类型的判断、简单 shell 命令（例如 cp）实现等。

第 5 章：对各函数说明及应用案例进行更新，并增加目录流编程内容。

第 6 章：对链接文件操作函数进行分类归纳整理，并以 ls –l 应用案例总结第 4、5、6 章内容。

第 7 章：对进程创建、进程属性等内容进行修订，添加守候进程以及日志管理的内容，并对孤儿进程、僵死进程进行比较。

第 8 章：精简文字，修订部分笔误，添加了重定向案例，并对信号处理一节进行了总结，更新了各函数的应用案例。

第 9 章：增加双向队列传递应用案例、信号量生产消费问题案例以及信号量的非阻塞应用案例。

第 10 章：精简文字，修订部分笔误，增加线程私有数据介绍以及各函数应用代码。

第 11 章：精简文字，修订部分笔误、增加线程在信号处理方面的内容。

第 12 章：精简文字，修订部分笔误、增加 UDP、TCP 同步及异步通信案例。

新增第 13 章：对网络编程部分专用函数，包括大小端与字节顺序、socket 属性、地址解析、网络调试工具等相关内容进行介绍。

新增第 14 章：增加对 TCP 及 UDP 高级编程的介绍，包括 TCP 多路选择、TCP 非阻塞、

信号驱动、UDP 广播、UDP 组播等内容。

新增第 15 章：增加网络服务器构建案例，并以构建 HTTP 服务器为例，介绍了包括文件及目录管理，进行线程管理及通信、网络编程内容，从而使读者学以致用。

在第 2 版中还增加了大量应用案例，特别是增加了大量网络编程内容，以求为读者进行 Linux 应用程序开发提供一条扎实的进阶之路。

本书还提供了完整的代码和教案，方便广大师生使用。

编　者

2009 年 10 月

# 前　言

在当今流行的通用操作系统阵营中，基本上是 Windows 和类 UNIX 一统江山；在嵌入式领域，Linux 操作系统做为一种专用嵌入式操作系统在嵌入式设备上得到了广泛应用。这主要归功于 Linux 源代码开放、网络功能成熟、极高的安全性、较强的可移植性等特点，另外，在 2.6 内核后，Linux 在实时性上也取得了突破性的发展。这些独特优势得到了人们的青睐，Linux 成为越来越普及的操作系统。

本书所有应用程序采用 2.6 内核为开发平台，GCC 4.0 为开发环境，主要介绍 Linux 操作系统下 C 语言应用程序开发相关内容。

## 本书主要特点

本书有以下几个特点。

（1）内容丰富。本书是作者多年计算机教学及工程经验总结，整合了 Linux 应用编程的绝大多数知识点，几乎涵盖了 Linux 操作系统下 C 应用编程的所有内容，包括工具使用及环境设置、文件及文件管理、进程及进程管理、进程间通信、线程及线程管理、线程通信、网络及网络应用编程等知识点。

（2）循序渐进。本书在写作思路上避开了大量理论的介绍，按知识体系介绍→应用函数分析→应用案例开发的写作顺序，让读者在掌握具体知识点的同时可以掌握实例的具体实现。

（3）案例指导。本书中所有调用函数及引用都标出具体的出处（在 Linux 操作系统中的文件位置），读者可以一目了然地知道对应函数及类型的定义过程。另外，本书遵循案例教学思想，每一个知识点都讲解一个应用程序，且所有代码都在教学实践过程中调试通过，读者可以直接使用。

（4）紧扣应用。本书所采用的开发平台为 2.6 内核，开发工作为 GCC 4.0，所列代码和实例都来源于具体的应用程序。

## 本书主要内容

本书共 12 章，详细讲解了 Linux 操作系统下 C 应用程序开发的方方面面。

第 1 章主要介绍 Linux 下 C 语言开发的基本环境。包括 Linux 操作系统编程基本术语介绍、Linux 编程基本概念、Linux 内核及库管理方式以及 Linux 下的编码风格。

第 2 章主要介绍 Linux 下 C 语言开发工具。包括 Linux 开发基本工具的使用、GCC/G++ 编译器、Make 工具及 Makefile 文件、GDB 调试工具以及自动编译调试工具的介绍。

第 3 章主要介绍 Linux 内存管理原理及内存管理工具。包括内存管理的基本概念、Linux 下常用内存管理函数以及常用 Linux 内存管理及调试工具的使用。

第 4 章主要介绍 ANSI C 标准的文件 I/O 管理。包括文件管理基本概念及文件指针、

ANSI C 文件 I/O 管理 API 函数。

第 5 章主要介绍 POSIX 标准的文件 I/O 管理。包括 Linux 系统下文件类型及属性、POSIX 文件 I/O 管理 API 函数。

第 6 章主要介绍 Linux 文件管理及目录操作。包括 Linux 文件系统管理方式、Linux 文件及目录管理操作 API 函数。

第 7 章主要介绍 Linux 进程管理与程序开发。包括进程环境及进程属性、Linux 进程控制、Linux 进程调度等内容。

第 8 章主要介绍 UNIX 进程间通信机制。包括无名管道 PIPE 通信机制、命名管道 FIFO 通信机制和信号中断处理方式。

第 9 章主要介绍 System V 进程间通信。包括 IPC 基础，消息队列、信号量、共享内存等 3 种通信机制的原理及应用。

第 10 章主要介绍 Linux 多线程编程。包括 Linux 线程概述、Linux 线程基本操作、线程属性控制和线程调度等内容。

第 11 章主要介绍线程间通信机制。包括互斥锁、条件变量、读写锁、线程信号等通信机制的原理及应用。

第 12 章主要介绍 Linux Socket 网络编程。包括网络通信基础、Socket 通信基本概念、Socket 通信编程过程，并详细介绍面向链接的 TCP 方式、面向非链接的 UDP 方式、Socket 通信多路选择套接字编程内容。

另外，本书以附录方式介绍了 GCC 参数、GDB 参数、Vim 编辑器参数、Emacs 编辑器参数和 CVS 服务器配置过程等内容。

本书全部代码均在附赠光盘中。

## 对读者的假定

本书要求读者有较好的 C 语言基础，熟悉 Linux 系统的基本命令。如果读者对操作系统或 Linux 内核有一定了解，则更容易学习本书。

## 本书编写工作

本书所有内容由杨宗德主编完成，邓玉春、曾庆华参与本书相关代码的编写及审稿工作。同时感谢何伟、张兵、刘兆宏、季建华、刘福刚、赵文革、黄弦等老师对本书的提出了大量宝贵指导意见，另外感谢石昀、朱元斌、钱文杰、陈功杰、汪洪、刘超、钟晓媛、刘梨平、石霞等同学试读了本书初稿，为本书相关内容提出了宝贵意见。由于时间仓促，本书难免有疏忽和不足之处，恳请读者批评赐教（book_better@sina.com）。

编　者

2007 年 11 月

# 目　录


第 1 章　Linux 下 C 语言开发环境 ……1

1.1　Linux 操作系统简介 ……2
1.1.1　Linux 操作系统简介 ……2
1.1.2　GNU/Linux 简介 ……3
1.1.3　相关术语介绍 ……3
1.2　Linux 开发初步 ……5
1.2.1　Linux 下 C 程序标准 ……5
1.2.2　库函数和系统调用 ……7
1.2.3　在线文档介绍 ……9
1.2.4　获取错误信息 ……10
1.3　部分常用工具简介 ……11
1.3.1　tar 打包器 ……11
1.3.2　Linux 常用命令及工具 ……12
1.4　Linux 下编码风格 ……16
1.4.1　GNU 编码规范 ……16
1.4.2　Linux 内核编码规范 ……17

第 2 章　Linux 下 C 语言开发工具 ……20

2.1　常用编辑工具 ……22
2.1.1　VIM 编辑器 ……22
2.1.2　Emacs 编辑器 ……24
2.1.3　Source Insight 工具 ……25
2.2　GCC 编译工具 ……29
2.2.1　GCC/G++简介 ……30
2.2.2　头文件及预处理结果分析 ……34
2.3　Make 工具与 Makefile 文件 ……36
2.3.1　Make 工具简介 ……36
2.3.2　Makefile 常用规则 ……39
2.4　常用调试工具 ……41
2.4.1　GDB 调试工具简介 ……41
2.4.2　GDB 演示示例 ……42
2.4.3　GDB 调试器常用语法 ……43
2.4.4　strace ……45
2.4.5　GCC 程序开发过程实例 ……46
2.5　Linux 库文件使用与创建 ……49
2.5.1　Linux 系统库文件管理的基本策略 ……49
2.5.2　Linux 下静态库的创建与使用 ……50
2.5.3　Linux 下共享库创建及使用 ……52
2.5.4　静态库与共享库的区别 ……54
2.6　Autoconf/Automake 自动化工具 ……54
2.6.1　Autoconf/Automake 工具介绍 ……54
2.6.2　Autoconf/Automake 工具使用示例 ……55

第 3 章　Linux 进程存储管理 ……61

3.1　Linux 程序存储结构与进程结构 ……62
3.1.1　Linux 可执行文件结构 ……62
3.1.2　Linux 进程结构 ……63
3.1.3　C 变量及函数存储类型 ……64
3.1.4　栈和堆的区别 ……69
3.1.5　示例：查看代码中各数据存储位置 ……70
3.1.6　常见内存错误示例分析 ……73
3.2　ANSI C 内存管理 API 函数 ……74
3.2.1　内存分配的基本方式 ……74
3.2.2　示例：为程序申请动态内存空间 ……74
3.2.3　内存数据管理函数 ……78

3.3 常用 Linux 内存管理及调试工具……80
3.3.1 mcheck 函数……80
3.3.2 Valgrind 内存检测工具……82
3.4 Linux 进程环境及系统限制……85
3.4.1 进程与命令参数……85
3.4.2 进程与环境变量……90
3.4.3 Linux 系统限制……91
3.4.4 Linux 时间管理……93

**第 4 章 ANSI C 文件管理**……96

4.1 文件及文件流……98
4.1.1 文件与流的基本概念……98
4.1.2 标准流及流主要功能……99
4.1.3 文件流指针……100
4.1.4 缓冲区类型……101
4.1.5 指定流缓冲区……103
4.2 ANSI C 标准文件 I/O 操作……105
4.2.1 打开关闭文件……105
4.2.2 读写文件流……107
4.2.3 文件流定位……111
4.2.4 实现文件拷贝操作示例……113
4.3 格式化输入/输出函数操作……114
4.3.1 printf/scanf 函数分析……114
4.3.2 fprintf/fscanf 函数分析……115
4.3.3 sprintf 函数分析……116
4.3.4 sscanf 函数分析……118

**第 5 章 POSIX 文件及目录管理**……121

5.1 文件描述符与内核文件表项……122
5.1.1 文件流与文件描述符的区别……122
5.1.2 文件表结构图……123
5.1.3 文件描述符与文件流的转换操作……123
5.2 POSIX 标准下文件 IO 管理……125
5.2.1 创建/打开/关闭文件……126
5.2.2 文件控制 fcntl……129
5.2.3 读/写文件内容……131
5.2.4 使用 POSIX IO 实现文件拷贝……133
5.2.5 文件定位……134
5.2.6 同步内核缓冲区……135
5.2.7 映射文件到内存……136
5.3 目录文件基本操作……137
5.3.1 打开/关闭目录文件……138
5.3.2 读写目录内容……138
5.3.3 定位目录位置……140
5.3.4 添加删除目录……140
5.3.5 当前工作路径操作……141

**第 6 章 普通文件、连接文件及目录文件属性管理**……143

6.1 Linux 文件系统管理……144
6.1.1 Linux 下 VFS 虚拟文件系统……144
6.1.2 ext2 文件系统结构……145
6.1.3 目录文件及常规文件存储方法……147
6.2 Linux 系统下文件类型及属性……147
6.2.1 Linux 文件类型及权限……147
6.2.2 Linux 文件类型……148
6.2.3 文件权限修饰位……151
6.2.4 文件访问权限位……152
6.3 Linux 文件属性管理……153
6.3.1 读取文件属性……153
6.3.2 修改文件权限操作……156
6.3.3 修改系统 umask 值……157
6.3.4 修改文件的拥有者及组……158
6.3.5 用户名/组名与 UID/GID 的转换……159
6.3.6 创建/删除硬连接……160
6.3.7 符号连接文件特殊操作……161
6.3.8 文件时间属性修改与时间处理……162
6.4 应用示例：实现 ls-l 基本操作……164

6.4.1 需求及知识点涵盖 ……164
6.4.2 流程及源代码实现 ……164

**第 7 章 Linux 进程管理与程序开发** ……169

7.1 进程环境及进程属性 ……170
7.1.1 进程资源 ……170
7.1.2 进程状态 ……170
7.1.3 进程基本属性 ……172
7.1.4 进程用户属性 ……176
7.2 进程管理及控制 ……179
7.2.1 创建进程 ……179
7.2.2 在进程中运行新代码 ……185
7.2.3 等待进程结束 ……189
7.2.4 退出进程 ……191
7.2.5 修改进程用户相关信息 …194
7.2.6 进程调度管理函数 ……197
7.3 Linux 特殊进程 ……202
7.3.1 守候进程及其创建过程 …202
7.3.2 日志信息及其管理 ……203
7.3.3 守候进程应用示例 ……205
7.3.4 孤儿进程与僵死进程 ……207

**第 8 章 进程间通信—管道和信号** ……210

8.1 进程间通信—PIPE ……212
8.1.1 无名管道概念 ……212
8.1.2 无名管道管理及应用 ……212
8.1.3 文件描述符重定向 ……215
8.1.4 实现 who|sort ……218
8.1.5 流重定向 ……220
8.2 进程间通信—FIFO ……221
8.2.1 有名管道概念 ……221
8.2.2 有名管道管理及应用 ……222
8.2.3 管道基本特点总结 ……225
8.3 信号中断处理 ……226
8.3.1 Linux 常见信号与处理 ……226
8.3.2 产生信号 ……229
8.3.3 信号处理与 signal 安装信号 ……233
8.3.4 sigaction 安装信号 ……234
8.3.5 信号集与屏蔽信号 ……238
8.3.6 等待信号 ……244
8.3.7 信号应用示例 ……246

**第 9 章 System V 进程间通信** ……250

9.1 System VIPC 基础 ……252
9.1.1 key 值和 ID 值 ……252
9.1.2 拥有者及权限 ……254
9.2 消息队列 ……254
9.2.1 消息队列 IPC 原理 ……254
9.2.2 Linux 消息队列管理 ……257
9.2.3 消息队列应用实例 ……259
9.3 信号量通信机制 ……264
9.3.1 信号量 IPC 原理 ……264
9.3.2 Linux 信号量管理操作 …265
9.3.3 SEM_UNDO 参数的应用 ……270
9.3.4 使用信号量实现生产消费问题 ……272
9.4 共享内存 ……275
9.4.1 共享内存 IPC 原理 ……275
9.4.2 Linux 共享内存管理 ……276
9.4.3 共享内存的权限管理示例 ……278
9.4.4 共享内存处理应用示例 …279
9.4.5 共享内存处理应用示例 …281

**第 10 章 Linux 多线程编程** ……285

10.1 线程基本概念与线程操作 ……286
10.1.1 线程与进程的对比 ……286
10.1.2 创建线程 ……287
10.1.3 线程退出与等待 ……288
10.1.4 取消线程 ……291
10.1.5 线程与私有数据 ……294
10.2 线程属性控制 ……297
10.2.1 线程 ID ……298
10.2.2 初始化线程属性对象 …298

10.2.3 获取/设置线程 detachstate 属性……299
10.2.4 获取/设置线程栈相关属性……300
10.2.5 线程属性控制实例……301
10.3 线程调度策略……303
10.3.1 获取/设置线程属性调度属性……303
10.3.2 获取/设置指定线程调度属性……305

**第 11 章 线程间同步机制……309**

11.1 互斥锁通信机制……310
11.1.1 互斥锁基本原理……310
11.1.2 互斥锁基本操作……310
11.1.3 互斥锁应用实例……311
11.2 条件变量通信机制……313
11.2.1 条件变量基本原理……313
11.2.2 条件变量基本操作……315
11.2.3 条件变量应用实例……316
11.3 读写锁通信机制……320
11.3.1 读写锁基本原理……320
11.3.2 读写锁基本操作……320
11.3.3 读写锁应用实例……322
11.4 线程与信号……325
11.4.1 线程信号管理……325
11.4.2 线程信号应用实例……326

**第 12 章 Linux socket 网络编程……329**

12.1 网络通信基础……330
12.1.1 TCP/IP 协议簇基础……330
12.1.2 IPV4 协议基础……331
12.1.3 网络数据包封包与拆包过程……334
12.2 BSD Socket 网络通信编程……339
12.2.1 BSD TCP 通信编程流程……339
12.2.2 BSD UDP 通信编程流程……341
12.2.3 BSD Socket 网络编程 API……342
12.2.4 使用 AF_UNIX 实现本机数据流通信示例……349
12.2.5 使用 AF_INET 实现 UDP 点对点通信示例……352
12.3 使用 TCP 实现简单聊天程序…354
12.3.1 服务器端代码分析……355
12.3.2 客户器端代码分析……357

**第 13 章 网络编程工具介绍……359**

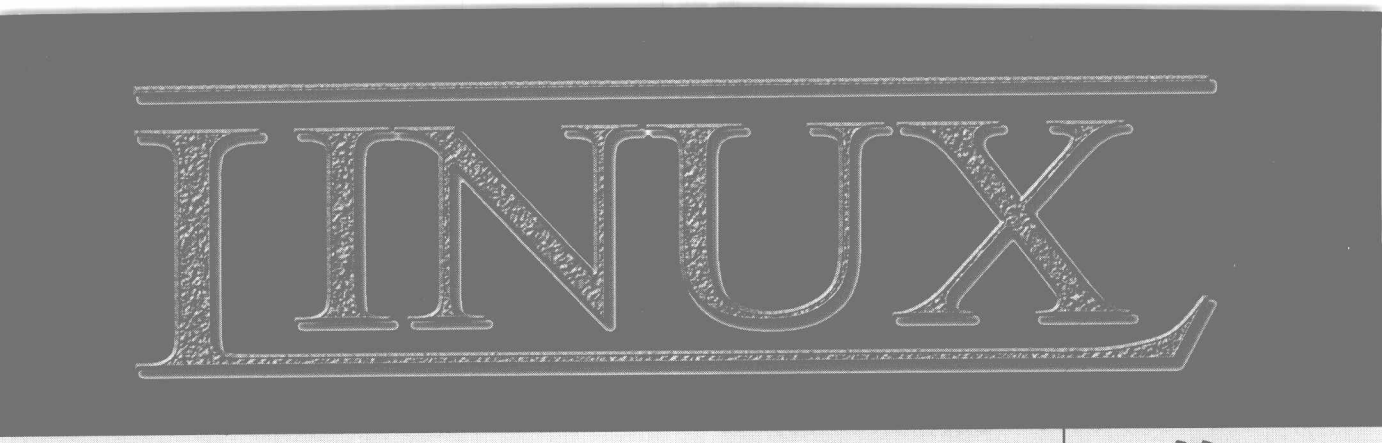
13.1 地址处理函数说明……360
13.1.1 大小端问题与网络字节顺序……360
13.1.2 字节顺序处理函数……361
13.1.3 点分十进制 IP 地址与二进制 IP 地址转换……363
13.1.4 通过 IP 地址获取网络 ID 和主机 ID……365
13.2 域名与 IP 信息解析……365
13.2.1 Linux 下域名解析过程…365
13.2.2 通过域名返回主机信息……366
13.2.3 通过域名和 IP 返回主机信息……367
13.2.4 getaddrinfo 获取主机信息……368
13.3 控制 socket 文件描述符属性…371
13.3.1 set/getsockopt()修改 socket 属性……371
13.3.2 fcntl 控制 socket……374
13.3.3 ioctl 控制文件描述符…374
13.4 网络调试工具……377
13.4.1 tcpdump 的使用……378
13.4.2 netstat 工具使用……380
13.4.3 lsof 工具使用……381

**第 14 章　网络编程高级应用** ……383

14.1　I/O 阻塞与非阻塞操作应用…384
14.1.1　非阻塞处理方法 ……384
14.1.2　非阻塞应用示例 ……384
14.2　socket 多路复用技术……388
14.2.1　select()函数介绍 ……388
14.2.2　pselect()函数 ……390
14.2.3　多路选择应用示例 ……390
14.3　socket 信号驱动 ……396
14.3.1　各类 I/O 操作比较……396
14.3.2　SIGIO 信号处理机制…397
14.3.3　UDP 信号驱动实现应用示例 ……398
14.4　UDP 广播与组播通信 ……401
14.4.1　广播地址与广播通信 …401
14.4.2　组播地址与组播通信 …403
14.4.3　UDP 广播应用示例 ……404
14.4.4　UDP 组播应用示例…… 407
14.5　原始套接口基本应用…… 412
14.5.1　原始套接口基本原理… 412
14.5.2　原始套接口应用示例… 412

**第 15 章　构建网络服务器** …… 415

15.1　多客户端实现…… 416
15.1.1　多进程实现多客户端… 416
15.1.2　多线程实现多客户端… 420
15.2　基于 xinetd 的网络服务应用 … 420
15.2.1　xinetd 服务介绍…… 421
15.2.2　应用示例…… 422
15.3　构建简单的 HTTP 网络服务器…… 423
15.3.1　服务器运行及测试结果…… 424
15.3.2　主函数运行流程…… 425
15.3.3　支撑函数…… 427




# 第1章 Linux下C语言开发环境

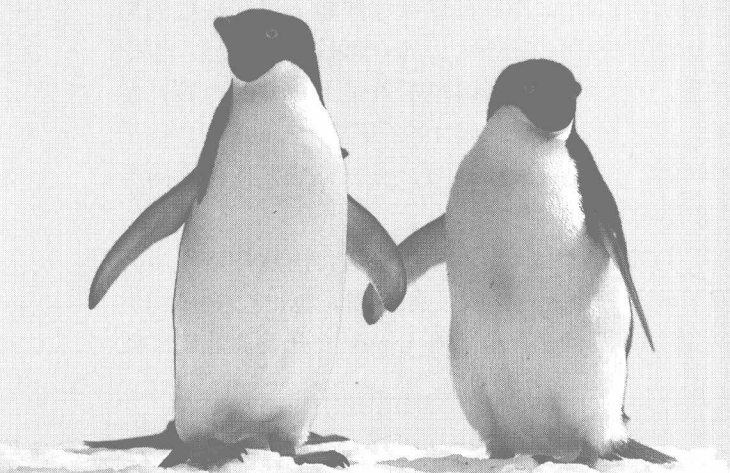
Linux 应用程序开发平台有别于 Windows 应用程序开发平台，因此在介绍具体编程内容之前，本书第 1、2 章主要介绍 Linux 操作系统下 C 语言程序的开发环境和开发工具。

本章主要介绍 Linux 下 C 语言开发环境，包括一些基本概念和基本编程环境。本章第 1 节主要对 Linux 操作系统及其相关术语进行了简要介绍。

本章第 2 节主要介绍 Linux 操作系统下编程基本概念以及如何获得 Linux 下的帮助文件，包括 Linux 操作系统下 C 语言库文件标准以及系统调用的基本概念。

本章第 3 节主要介绍 Linux 部分常用工具简介，包括文件打包工具、查找搜索工具，熟练使用这些命令或工具在编程时能够很好地提高效率。

本章第 4 节为读者展示了 GNU 编码规范和 Linux 内核编码规范。读者在学习 Linux 编程之前应养成良好的编码规范，这样不仅能增强代码的可读性，还能减少代码维护的工作量，提高代码的可扩展性。



# 1.1 Linux 操作系统简介

## 1.1.1 Linux 操作系统简介

UNIX 操作系统于 1969 年由 Ken Thompson 在 AT& T 贝尔实验室的一台 DEC PDP-7 计算机上实现。后来 Ken Thompson 和 Dennis Ritchie 使用 C 语言对整个系统进行了再加工和编写，使得 UNIX 能够很容易地移植到其他硬件的计算机上。由于此时 AT&T 还没有把 UNIX 作为它的正式商品，因此研究人员只是在实验室内部使用并完善它。正是由于 UNIX 是被作为研究项目，其他科研机构和大学的计算机研究人员也希望能得到这个系统，以便进行自己的研究。AT&T 采用分发许可证的方法，大学和研究机构仅仅需要很少的费用就能获得 UNIX 的源代码以进行研究。UNIX 的源代码被散发到各个大学，一方面使得科研人员能够根据需要改进系统，或者将其移植到其他的硬件环境中去，另一方面培养了大量懂得 UNIX 使用和编程的学生，这使 UNIX 的使用更为普及。

到了 20 世纪 70 年代末，在 UNIX 发展到版本 6 之后，AT&T 认识到了 UNIX 的价值，并成立了 UNIX 系统实验室（UNIX System Lab，USL）来继续发展 UNIX。因此一方面 AT&T 继续发展内部使用的 UNIX 版本 7，一方面由 USL 开发对外正式发行的 UNIX 版本，同时 AT&T 也宣布对 UNIX 产品拥有所有权。几乎在同时，加州大学伯克利分校计算机系统研究小组（CSRG）借助 UNIX 对操作系统进行了研究，他们对 UNIX 进行的改进相当多，增加了很多当时非常先进的特性，包括更好的内存管理、快速且健壮的文件系统等，大部分原有的源代码都被重写，很多其他的 UNIX 使用者，包括其他大学和商业机构，都希望能得到经 CSRG 改进的 UNIX 系统。因此 CSRG 的研究人员把他们的 UNIX 组成一个完整的 UNIX 系统——BSD UNIX（Berkeley Software Distribution）向外发行。

与此同时，AT&T 的 UNIX 系统实验室也在不断改进他们的商用 UNIX 版本，直到他们吸收了 BSD UNIX 中已有的各种先进特性，并结合其本身的特点，推出了 UNIX System V 版本。从此以后，BSD UNIX 和 UNIX System V 形成了当今 UNIX 的两大主流，现代的 UNIX 版本大部分都是这两个版本的衍生产品：IBM 的 AIX4.0、HP/UX11 和 SCO 的 UNIXWare 等属于 System V，而 Minix、freeBSD、NetBSD、OpenBSD 等属于 BSD UNIX。

Linux 由 UNIX 操作系统发展而来，它的内核由 Linus Torvalds 以及网络上组织松散的黑客队伍一起从零开始编写而成。Linux 从一开始就决定自由扩散 Linux（包括源代码），他把源代码发布在网上，随即就引起爱好者的注意，他们通过互联网也加入了 Linux 的内核开发工作。一大批高水平程序员的加入使 Linux 得到了迅猛发展。到 1993 年底，Linux 1.0 终于诞生。Linux 1.0 已经是一个功能完备的操作系统，其内核紧凑高效，可以充分发挥硬件的性能，在 4MB 内存的 80386 机器上也表现得非常好。

Linux 加入 GNU 并遵循通用公共许可证（GPL），由于不排斥商家对自由软件进一步开发，故而使 Linux 开始了又一次飞跃，出现了很多 Linux 发行版，有如 Slackware、Redhat、TurboLinux、OpenLinux 等十多种，而且还在增加；还有一些公司在 Linux 上开发商业软件或

把其他 UNIX 平台的软件移植到 Linux 上来，如今很多 IT 界的大腕如 IBM、Intel、Oracle、Infomix、Sysbase、Netscape、Novell 等都宣布支持 Linux。商家的加盟弥补了纯自由软件的不足，扫清了发展障碍，Linux 得以迅速普及。

Linux 操作系统具有以下特点。

- Linux 具备现代一切功能完整的 UNIX 系统所具备的全部特征，其中包括真正的多任务、虚拟内存、共享库、需求装载、优秀的内存管理以及 TCP/IP 网络支持等。
- Linux 的发行遵守 GNU 的通用公共许可证（GPL）。
- 在原代码级上兼容绝大部分的 UNIX 标准（如 IEEE POSIX，System V，BSD），遵从 POSIX 规范。读者可以在网络上获得关于这一内容的更多信息。

## 1.1.2 GNU/Linux 简介

GNU 工程（GNU 是“GNU's Not UNIX”首字母缩写语）开始于 1984 年，旨在发展一个类 UNIX 且为自由软件的完整操作系统：GNU 系统。更精确地说，各种使用 Linux 作为内核的 GNU 操作系统应该被称为 GNU/Linux 系统。

GNU 工程开发了大量用于 UNIX 的自由软件工具和类 UNIX 操作系统，例如 Linux。虽然有许多组织和个人都对 Linux 的发展作出了帮助，但自由软件基金会依然是最大的单个贡献者。它不仅仅创造了绝大部分在 Linux 中使用的工具，还为 Linux 的存在提供了理论和社会基础。

为保证 GNU 软件可以自由地“使用、复制、修改和发布”，所有 GNU 软件都遵循无条件授权所有权利给任何人的协议条款——GNU 通用公共许可证（GNU General Public License，GPL）。

1985 年 Richard Stallman 创立的自由软件基金会（FSF，Free Software Foundation）为 GNU 计划提供了技术、法律以及财政支持。尽管 GNU 计划大部分时候是由个人自愿无偿贡献，但 FSF 有时还是会聘请程序员帮助编写。当 GNU 计划开始逐渐获得成功时，一些商业公司开始介入开发和技术支持。

到了 1990 年，GNU 计划已经开发出的软件包括了一个功能强大的文字编辑器 Emacs、C 语言编译器 GCC 以及大部分 UNIX 系统的程序库和工具。唯一依然没有完成的重要组件就是操作系统的内核（称为 HURD）。

## 1.1.3 相关术语介绍

### 1. POSIX 及其重要地位

POSIX 表示可移植操作系统接口（Portable Operating System Interface，缩写为 POSIX 是为了读音更像 UNIX）。它由电气和电子工程师协会（Institute of Electrical and Electronics Engineers，简称为 IEEE）开发，可以提高类 UNIX 环境下应用程序的可移植性。然而，POSIX 并不局限于 UNIX，许多其他的操作系统，例如 DEC OpenVMS 和 Microsoft Windows NT，都支持 POSIX 标准，尤其是 IEEE STD.1003.1-1990（1995 年修订）和 POSIX.1。POSIX.1 给操作系统提供了源代码级别的 C 语言应用编程接口（API），例如读写文件 read/write。POSIX.1 已经被国际标准化组织（International Standards Organization，简称为 ISO）所接受，被命名

为 ISO/IEC9945-1:1990 标准。虽然某些部分还处在开发过程中，但是 POSIX 现在已经发展成为一个庞大的标准族。

2．GNU 和 Linux 的关系

GNU 项目已经开发了许多高质量的编程工具，包括 emacs 编辑器、著名的 GNU C 和 C++ 编译器（gcc 和 g++），这些编译器可以在任何计算机系统上运行。所有的 GNU 软件和派生工作均使用 GNU 通用公共许可证（GPL）。GPL 允许软件作者拥有软件版权，但要授予其他任何人以合法复制、发行和修改软件的权利。

Linux 中使用了许多 GNU 工具，用于实现 POSIX.2 标准的工具几乎都是 GNU 项目开发的。Linux 内核、GNU 工具以及其他的一些自由软件组成了人们常说的 Linux，包括 C 语言编译器和其他开发工具及函数库、X Window 窗口系统、各种应用软件（包括字处理软件、图像处理软件等）、其他各种 Internet 软件（包括 FTP 服务器、WWW 服务器）、关系数据库管理系统等。

3．通用公共许可证（General Public License，GPL）

GPL 的文本保存在 Linux 系统的不同目录中的 COPYING 文件里。例如，键入“cd /usr/doc/ghostscript*”，然后再键入“more COPYING”可查看 GPL 的内容。GPL 和软件是否免费无关，它的主要目标是保证软件对所有用户来说是自由的。GPL 通过如下途径实现这一目标：

（1）要求软件以源代码的形式发布，并规定任何用户都能够以源代码的形式将软件复制或发布给其他用户。

（2）提醒每个用户，对于该软件不提供任何形式的担保。

（3）如果用户的软件使用了受 GPL 保护的软件的任何一部分，该软件都会成为 GPL 软件，也就是说必须随应用程序一起发布源代码。

（4）GPL 并不排斥对自由软件进行商业性质的包装和发行，也不限制在自由软件的基础上打包发行其他非自由软件。

遵照 GPL 的软件并不是可以任意传播的，这些软件通常都有正式的版权，GPL 在发布软件或者复制软件时都会声明限制条件。但是，从用户的角度考虑，这些根本不能算是限制条件，相反，用户只会从中受益，因为它可以确保用户获得源代码。

尽管 Linux 内核也属于 GPL 范畴，但 GPL 并不适用于通过系统调用而使用内核服务的应用程序，通常把这种应用程序看作是内核的正常使用。假如准备以二进制的形式发布应用程序（像大多数商业软件那样），则必须确保自己的程序未使用 GPL 保护的任何软件。如果软件通过库函数调用而且使用了其他软件，则不必受此限制。大多数函数库受另一种 GNU 公共许可证，即 LGPL 的保护，下面将会介绍。

4．LGPL（Libraray General Public License）

GNU LGPL（GNU 程序库通用公共许可证）的内容包括在 COPYING.LIB 文件中。如果安装了内核源程序，在任意一个源程序的目录下都可以找到 COPYING.LIB 文件的一个拷贝。

LGPL 允许在自己的应用程序中使用程序库，即使不公开自己的源代码。但是，LGPL 还规定，用户必须能够获得在应用程序中使用的程序库的源代码，并且允许用户对这些程序库进行修改。

大多数 Linux 程序库，包括 C 程序库（libc.a）都属于 LGPL 范畴。因此，如果在 Linux 环境下，使用 GCC 编译器建立自己的应用程序，程序所连接的多数程序库是受 LGPL 保护的。如果想以二进制的形式发布自己的应用程序，则必须注意遵循 LGPL 有关规定。

遵循 LGPL 的一种方法是，随应用程序一起发布目标代码，并发布这些目标程序和受 LGPL 保护的、更新的 Linux 程序库连接起来的 makefile 文件。

遵循 LGPL 的另一种方法是使用动态连接。使用动态连接时，即使程序在运行中调用函数库中的函数，应用程序本身和函数库也是不同的实体。通过动态连接，用户可以直接使用更新后的函数库，而不用对应用程序重新连接。

# 1.2 Linux 开发初步

## 1.2.1 Linux 下 C 程序标准

在 Linux 操作系统下进行 C 程序开发的标准主要有两个：ANSI C 标准和 POSIX 标准。

ANSI C 标准是 ANSI（美国国家标准局）于 1989 年制定的 C 语言标准，后来被 ISO（国际标准化组织）接受为标准，因此也称为 ISO C。

POSIX 标准是最初由 IEEE 开发的标准族，部分已经被 ISO 接受为国际标准。

### 1. ANSI C

ANSI C 的目标是为各种操作系统上的 C 程序提供可移植性保证(例如 Linux 与 Windows 之间)，而不仅仅限于类 UNIX 系统。该标准不仅定义了 C 编程语言的语法和语义，而且还定义了一个标准库。ISO C 标准定义的头文件如表 1-1 所示。

表 1-1　　ISO C 标准定义的头文件

| 头 文 件 | 说　明 | 头 文 件 | 说　明 |
|---|---|---|---|
| <assert.h> | 验证程序断言 | <signal.h> | 信号 |
| <complex.h> | 支持复数算术运算 | <stdarg.h> | 可变参数表 |
| <ctype.h> | 字符类型 | <stdbool.h> | 布尔类型和值 |
| <errno.h> | 出错码 | <stddef.h> | 标准定义 |
| <fenv.h> | 浮点环境 | <stdint.h> | 整型 |
| <float.h> | 浮点常量 | <stdio.h> | 标准 I/O 库 |
| <inttypes.h> | 整型格式转换 | <stdlib.h> | 实用程序库函数 |
| <iso646.h> | 替代关系操作符宏 | <string.h> | 字符串操作 |
| <limits.h> | 实现常量 | <tgmath.h> | 通用类型数学宏 |
| <locale.h> | 局部类别 | <time.h> | 时间和日期 |
| <math.h> | 数学常量 | <wchar.h> | 扩展的多字节和宽字符支持 |
| <setjmp.h> | 非局部 goto | <wctype.h> | 宽字符分类和映射支持 |