

Apache 权威指南

(影印版)

第二版
含光盘



Apache

The Definitive Guide

O'REILLY®

中国电力出版社

Ben Laurie & Peter Laurie

Desktop Reference

Apache 1.3.4



Apache

Quick Reference

O'REILLY®

Andrew Ford

图书在版编目 (CIP) 数据

实用C语言编程 / (美) 奥莱恩著; 郭大海译. - 北京: 中国电力出版社, 2000. 5
书名原文: Practical C Programming
ISBN 7-5083-0308-3

I . 实 … II . ①奥 … ②郭 … III . C语言程序设计 IV . TP312
中国版本图书馆 CIP 数据核字 (2000) 第 08994 号

北京市版权局著作权合同登记
图字: 01-1999-3746 号

© 1997 by O'Reilly & Associates, Inc.

Simplified Chinese Edition, jointly published by O'Reilly & Associates, Inc. and China Electric Power Press, 2000. Authorized translation of the English edition, 1997 O'Reilly & Associates, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly & Associates, Inc. 出版 1997。

简体中文版由中国电力出版社出版 2000。英文原版的翻译得到 O'Reilly & Associates, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly & Associates, Inc. 的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式复制。

书 名 / 实用C语言编程

书 号 / ISBN 7-5083-0308-3

责任编辑 / 刘君, 关敏, 蒙虎

封面设计 / Ellie Volckhausen, Hanna Dyer, 张健

出版发行 / 中国电力出版社

地 址 / 北京三里河路 6 号 (邮政编码 100044)

经 销 / 全国新华书店

印 刷 / 北京市地矿印刷厂

开 本 / 787 毫米 × 1092 毫米 16 开本 31 印张 310 千字

版 次 / 2000 年 5 月第一版 2000 年 5 月第一次印刷

印 数 / 0001-5000 册

定 价 / 49.00 元 (册)

前言

本书讲述如何使用C语言进行真正的编程。C是目前软件开发者们最主要的编程语言。这也是它受到广泛传播并且成为标准的原因。现在也有新的编程语言出现，如C++，但这些语言仍然在演化中。C仍然是进行健壮的、可移植编程的首选语言。

本书侧重介绍在实际编程过程中需要知道的技巧问题。它不仅告诉你有关C语言的机制，也告诉你用C进行编程的整个过程（包括程序的含义、设计、编码、方法、调试、发布、文档、维护和版本更新等）。

本书也介绍了用C编程的风格和艺术。要写出一个好的程序，需要做许多工作，而不仅仅是敲一些代码。这是一门有关写作和编程的技巧，并将其合二为一的艺术，从而使你写出的程序成为一件杰作。这样创作出来的作品才是真正的艺术。一个非常好的程序不仅仅要功能正确，而且应该简单易读。在程序中允许加进一些评论性的描述性文字。当清晰的评注被加进程序中时，这样的程序就会得到其他人的高度评价。

程序应该尽可能简单。程序员应该避免玩一些聪明的技巧。在本书中我们强调简单、实用的原则。例如，在C中有15条关于操作符的规则，但它们可以被简化为下面两个：

1. 先乘除，后加减。

2. 括号优先于其他一切运算。

考虑两个程序：一个是由聪明的程序员使用了所有技巧编写的程序，程序没有包含任何注解，但是可以运行；另外一个程序有很好的注解和结构，但是它不能运行。两个程序哪一个更有用呢？从长远的角度看，应该是那个不能运行的程序，因为它存在的问题可以被解决掉。尽管那个聪明的程序员的程序现在可以运行，但迟早该程序会被修改的。那么到那时，最坏的事情就会出现——你不得不去修改一个充满技巧、但没有任何注解的程序。

这本书是针对那些没有编程经验，或者已经对C语言有所了解但是还想进一步提高他们的编程风格和程序可靠性的人所作的。在使用这本书之前，你应该知道如何使用计算机并且知道如何使用一些基本的工具，诸如文本编辑器和文件系统。

对于那些想在UNIX操作系统上用generic cc编译器或者自由软件基金会的gcc编译器编写和运行程序的读者，我们给出了一些特别的指导。对于MS-DOS/Windows的用户，我们也给出了包括Borland C++、Turbo C++和Microsoft Visual C++的使用说明(这些编译器可以对C和C++的代码进行编译)。在本书中也给出了一些使用编程工具make进行自动程序编译的例子。

这本书是如何组织的

在你学会跑步之前，你必须先学会走路。

在第一部分——基础中，你将会学习如何走路。本章只讲述如何编写一个非常简单的程序。读者从编程技巧和编程风格入手。下一步，你将学习如何使用变量和非常简单的判断和控制语句。在第七章程序设计过程中，我们才将编写真正程序的完整过程展现给读者。

在第二部分——简单程序设计中，我们介绍编程时所有需要的简单语句和运算符。读者还将学习到如何把这些语句组织成为简单函数的方法。

在第三部分——高级编程概念中，我们将介绍一些基本的说明和语句来构造高

目录

前言	1
第一部分 基础	11
第一章 什么是 C?	13
编程原理	14
C 语言简史	17
C 如何工作	17
如何学习 C	19
第二章 编程基础	21
程序从概念到运行	21
编写一个真正的程序	22
使用命令行编译器编程	23
使用集成开发环境 (IDE) 编程	27
获取 UNIX 帮助	45
获取集成开发环境帮助	45

集成开发环境菜单	45
编程练习	48
第三章 风格	49
基础编码练习	54
编码盲从	56
缩进与编码格式	56
清晰	57
简明	58
小结	59
第四章 基本定义与表达式	60
程序要素	60
程序的基本结构	61
简单表达式	62
变量和存储	64
变量定义	65
整型	66
赋值语句	66
printf 函数	68
浮点型	70
浮点数与整数的除法运算	70
字符	73
答案	74
编程练习	75
第五章 数组、修饰符与读取数字	76
数组	76
串	78
读取串	81

多维数组	84
读取数字	86
变量初始化	88
整型	90
浮点型	92
常量说明	93
十六进制与八进制常量	93
快捷运算符	94
副作用	95
++x 或 x++	96
更多的副作用问题	97
答案	98
编程练习	99
第六章 条件和控制语句	101
if 语句	101
else 语句	102
怎样避免误用 strcmp 函数	104
循环语句	104
while 语句	105
break 语句	107
continue 语句	108
随处赋值的副作用	109
答案	111
编程练习	111
第七章 程序设计过程	113
设置	115
程序规范	116
代码设计	116

原型	118
Makefile	119
测试	123
调试	124
维护	126
修改	126
代码分析	127
注释程序	128
使用调试器	128
用文本编辑器浏览	128
增加注释	128
编程练习	131

第二部分 简单程序设计 133

第八章 更多的控制语句 135

for 语句	135
switch 语句	139
switch, break 和 continue	145
答案	145
编程练习	147

第九章 变量作用域和函数 149

作用域和类	149
函数	153
无参数的函数	157
结构化程序设计	158
递归	160
答案	161
编程练习	162

第十章 C 预处理器	163
#define 语句	163
条件编译	170
包含文件	173
带参数的宏	174
高级特征	176
小结	176
答案	177
编程练习	180
第十一章 位运算	181
位运算符	183
与运算符 (&)	183
按位或 ()	186
按位异或 (^)	187
非运算符 (~)	187
左移与右移运算符 (<<, >>)	188
设置、清除和检测位	190
位图图形	194
答案	200
编程练习	201
第十二章 高级类型	202
结构	202
联合	205
typedef	207
枚举类型	209
强制类型转换	210
位字段或紧缩结构	210
结构数组	212

小结	213
编程练习	213
第十三章 简单指针	215
函数自变量指针	220
常量指针	222
指针和数组	224
如何不使用指针	229
用指针分隔字符串	231
指针和结构	235
命令行参数	236
编程练习	242
答案	242
第十四章 文件输入 / 输出	245
转换程序	249
二进制和 ASCII 码文件	252
行尾难题	253
二进制 I/O	255
缓冲问题	257
非缓冲 I/O	258
设计文件格式	264
答案	266
编程练习	267
第十五章 调试和优化	268
调试	268
交互调试器	280
调试一个二分查找程序	285
实时运行错误	297

公开声明调试方法	299
优化	300
答案	309
编程练习	309
第十六章 浮点数	310
浮点数格式	310
浮点数加法 / 减法	312
乘法	313
除法	313
上溢和下溢	314
舍入误差	314
精度	315
舍入误差最小化	316
判定精度	317
精度和速度	318
幂级数	319
编程练习	321
第三部分 高级编程观念	323
第十七章 高级指针	325
指针和结构	325
free 函数	329
链表	330
结构指针运算符	333
顺序链表	334
双向链表	337
树	340

树的打印	344
程序的剩余部分	345
象棋程序中用到的数据结构	349
答案	351
编程练习	353
第十八章 模块化编程	354
模块	354
公用和专用	355
extern 修饰符	356
头文件	358
模块体	361
使用无限数组的程序	361
用于多文件的 Makefile	364
使用无限数组	368
把一项任务分成模块	376
模块划分实例：文本编辑器	376
编译器	378
电子表格	380
模块设计准则	380
编程练习	380
第十九章 旧式编译器	382
K&R 风格的函数	382
库的发展	386
遗漏的特性	386
Free/Malloc 的发展	387
lint	388
答案	388

第二十章 移植问题	391
模块化	391
字大小	392
字节顺序问题	392
对齐问题	393
NULL 指针问题	395
文件名问题	396
文件类型	397
小结	397
答案	398
第二十一章 C 内的“角落”	399
do/while	399
goto	400
?: 指令	401
, 运算符	402
不稳定限定词	402
答案	402
第二十二章 组合到一起	403
需求	403
规范说明	404
代码设计	406
编码	412
功能描述	412
扩展	414
测试	415
修改	416
最后的警告	416

程序文件	416
编程练习	443
第二十三章 程序设计格言	444
概述	444
设计	445
定义	445
switch 语句	445
预处理器	446
风格	446
编译	446
最后的注解	447
答案	447
第四部分 其他语言特性	449
附录一 ASCII 表	451
附录二 范围和参数传递转换	453
附录三 运算符优先规则	455
附录四 使用幂级数计算正弦函数的程序	457
词汇表	463

1

Getting Started

When you connect to the URL of someone's home page—say the notional *http://www.butterthlies.com/* we shall meet later on—you send a message across the Internet to the machine at that address. That machine, you hope, is up and running, its Internet connection is working, and it is ready to receive and act on your message.

URL stands for Universal Resource Locator. A URL such as *http://www.butterthlies.com/* comes in three parts:

`<method>://<host>/<absolute path URL (apURL)>`

So, in our example, `<method>` is `http`, meaning that the browser should use HTTP (Hypertext Transfer Protocol); `<host>` is `www.butterthlies.com`; and `<apURL>` is `/`, meaning the top directory of the host. Using HTTP/1.1, your browser might send the following request:

```
GET / HTTP/1.1
Host: www.butterthlies.com
```

The request arrives at port 80 (the default HTTP port) on the host *www.butterthlies.com*. The message is again in three parts: a method (an HTTP method, not a URL method), that in this case is `GET`, but could equally be `PUT`, `POST`, `DELETE`, or `CONNECT`; the Uniform Resource Identifier (URI) `/`; and the version of the protocol we are using. It is then up to the web server running on that host to make something of this message.

It is worth saying here—and we will say it again—that the whole business of a web server is to translate a URL either into a filename, and then send that file back over the Internet, or into a program name, and then run that program and send its output back. That is the meat of what it does: all the rest is trimming.

The host machine may be a whole cluster of hypercomputers costing an oil sheik's ransom, or a humble PC. In either case, it had better be running a web server, a program that listens to the network and accepts and acts on this sort of message.

What do we want a web server to do? It should:

- Run fast, so it can cope with a lot of inquiries using a minimum of hardware.
- Be multitasking, so it can deal with more than one inquiry at once.
- Be multitasking, so that the person running it can maintain the data it hands out without having to shut the service down. Multitasking is hard to arrange within a program: the only way to do it properly is to run the server on a multitasking operating system. In Apache's case, this is some flavor of Unix (or Unix-like system), Win32, or OS/2.
- Authenticate inquirers: some may be entitled to more services than others. When we come to virtual cash, this feature (see Chapter 13, *Security*) becomes essential.
- Respond to errors in the messages it gets with answers that make sense in the context of what is going on. For instance, if a client requests a page that the server cannot find, the server should respond with a "404" error, which is defined by the HTTP specification to mean "page does not exist."
- Negotiate a style and language of response with the inquirer. For instance, it should—if the people running the server can rise to the challenge—be able to respond in the language of the inquirer's choice. This ability, of course, can open up your site to a lot more action. And there are parts of the world where a response in the wrong language can be a bad thing. If you were operating in Canada, where the English/French divide arouses bitter feelings, or in Belgium, where the French/Flemish split is as bad, this feature could make or break your business.
- Offer different formats. On a more technical level, a user might want JPEG image files rather than GIF, or TIFF rather than either of the former. He or she might want text in vdi format rather than PostScript.
- Run as a proxy server. A proxy server accepts requests for clients, forwards them to the real servers, and then sends the real servers' responses back to the clients. There are two reasons why you might want a proxy server:
 - The proxy might be running on the far side of a firewall (see Chapter 13), giving its users access to the Internet.
 - The proxy might cache popular pages to save reaccessing them.