

# 《C程序设计新思维》

## 图书基本信息

书名：《C程序设计新思维》

13位ISBN编号：9787115386285

出版时间：2015-5

作者：Ben Klemens

页数：266

译者：赵铁成,徐波

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：[www.tushu000.com](http://www.tushu000.com)

# 《C程序设计新思维》

## 内容概要

C语言已经有40年的历史了。经过长时间的发展和普及，C语言应用场景有了很大的变化，的一些旧观念应该被淡化或者不再被推荐。

《C程序设计新思维》展现了传统C教科书所不具有的最新的相关技术。全书分为开发环境和语言两个部分，分别从编译、调试、测试、打包、版本控制等角度，以及指针、语法、文本、结构、面向对象编程、库等主题，对C程序设计的核心知识进行查缺补漏和反思。本书鼓励读者放弃那些对大型机才有意义的旧习惯，拿起新的工具来使用这门与时俱进的简洁语言。

《C程序设计新思维》适合有一定基础的C程序员或C语言学习者阅读，也适合想要较为深入地理解C语言特性的读者参考。

# 《C程序设计新思维》

## 作者简介

Ben Klemens为布鲁金斯学会、世界银行、美国国家精神健康中心和美国政府编写统计分析和计算机模型。他与布鲁金斯学会和自由软件基金会一起合作，确保程序员保留其作品使用权的权利。

## 书籍目录

### 第一部分 开发环境

#### 第1章 准备方便的编译环境

3

##### 1.1 使用包管理器

4

##### 1.2 在Windows下编译C

6

###### 1.2.1 Windows中的POSIX环境

7

###### 1.2.2 在POSIX下编译C

8

###### 1.2.3 不在POSIX环境中编译C

9

##### 1.3 库的路径

10

###### 1.3.1 一些我喜欢的选项

11

###### 1.3.2 路径

13

###### 1.3.3 运行时连接

15

##### 1.4 使用Makefile

16

###### 1.4.1 设定变量

17

###### 1.4.2 规则

19

##### 1.5 以源文件利用库

23

###### 1.6 以源文件利用库（即使你的系统管理员不想叫你这么做）

24

##### 1.7 通过本地文档来编译C程序

26

###### 1.7.1 在命令行里包含头文件

26

###### 1.7.2 统一的头文件

27

###### 1.7.3 嵌入文档

28

###### 1.7.4 从stdin中编译

29

#### 第2章 调试、测试和文档

31

##### 2.1 使用调试器

31

###### 2.1.1 GDB变量

35

2.1.2 打印结构	36
2.2 利用Valgrind检查错误	40
2.3 单元测试	41
2.3.1 把程序用作库	44
2.3.2 测试覆盖	45
2.4 编制文档	46
2.4.1 Doxygen	46
2.4.2 用CWEB解释代码	48
2.5 错误检查	50
2.5.1 在错误中的用户参与是什么	50
2.5.2 用户工作的上下文环境	52
2.5.3 如何返回错误信息	53
第3章 打包项目	55
3.1 Shell	56
3.1.1 用shell命令的输出来替换命令	56
3.1.2 用shell的循环来处理一组文件	58
3.1.3 针对文件的测试	60
3.1.4 fc	62
3.2 makefile还是shell脚本	64
3.3 用Autotools打包代码	67
3.3.1 一个Autotools的示例	68
3.3.2 用makefile.am来描述makefile	71
3.3.3 配置脚本	76
第4章 版本控制	80
4.1 通过diff查看差异	

81	
4.2 Git的对象	
82	
4.3 树和它们的枝	
86	
4.3.1 融合	
88	
4.3.2 迁移	
89	
4.4 远程版本库	
90	
第5章 和谐共处	
93	
5.1 过程	
93	
5.1.1 作为外来语言写程序	
93	
5.1.2 包装函数	
94	
5.1.3 跨越边境的代理数据结构	
94	
5.1.4 连接	
96	
5.2 与Python一起工作	
96	
5.2.1 编译与连接	
98	
5.2.2 Automake的条件子目录	
98	
5.2.3 Autotools支持下的Distutils	
100	
第二部分 语言	
第6章 玩转指针	
106	
6.1 自动、静态和手工内存	
106	
6.2 持久性的状态变量	
109	
6.3 不使用malloc的指针	
110	
6.3.1 结构被复制，数组创建别名	
111	
6.3.2 malloc和内存操纵	
114	
6.3.3 错误来源于星号	
115	
6.3.4 你需要知道的各种指针运算	
116	
第7章 可以忽略的C语法	

121	
7.1 不需要明确地从main函数返回	
122	
7.2 让声明流动	
122	
在运行时设置数组的长度	
124	
7.3 减少类型转换	
125	
7.4 枚举和字符串	
126	
7.5 标签、goto、switch和break	
128	
7.5.1 考虑goto	
129	
7.5.2 switch	
130	
7.6 被摒弃的float	
132	
第8章 障碍和机遇	
136	
8.1 营造健壮和繁盛的宏	
136	
预处理器技巧	
140	
8.2 static和extern链接	
143	
只在头文件中声明外部链接的元素	
145	
8.3 const关键字	
147	
8.3.1 名词-形容词形式	
148	
8.3.2 压力	
149	
8.3.3 深度	
150	
8.3.4 char const **问题	
150	
第9章 文本	
154	
9.1 使用asprintf，使字符串的处理不再痛苦	
154	
9.1.1 安全	
156	
9.1.2 常量字符串	
156	
9.1.3 用asprintf扩展字符串	
158	

9.1.4 strtok的赞歌	159
9.2 Unicode	163
9.2.1 C代码的编码	165
9.2.2 Unicode函数库	167
9.2.3 示例代码	168
第10章 更好的结构	171
10.1 复合文字	172
通过复合文字进行初始化	173
10.2 可变参数宏	173
10.3 安全终止的列表	175
10.4 Foreach	176
10.5 函数的向量化	176
10.6 指定的初始化值	178
10.7 用零初始化数组和结构	180
10.8 typedef可以化繁为简	181
10.9 从函数返回多个数据项	183
10.10 灵活的函数输入	187
10.10.1 把函数声明为printf风格	187
10.10.2 可选参数和命名参数	189
10.10.3 使无聊的函数焕发光彩	191
10.11 void指针以及它所指向的结构	197
10.11.1 具有通用输入的函数	197
10.11.2 通用结构	201
第11章 C语言的面向对象编程	206
11.1 你所不明白的（以及为什么你不能不明白）	



207
11.1.1 作用域
207
11.1.2 用操作符重载进行重载
210
11.2 扩展结构和字典
214
11.2.1 扩展一个结构
215
11.2.2 实现一个字典
219
11.2.3 基于指向对象的指针编码
223
11.3 你结构中的函数
224
11.4 引用计数
228
11.4.1 示例：一个子字符串对象
229
11.4.2 一个基于代理的组构造模型
233
第12章 库
240
12.1 GLib
240
12.2 POSIX
241
12.2.1 为巨大的数据集合使用mmap
241
12.2.2 用Pthreads轻松实现线程
243
12.3 GNU科学计算库
251
12.4 SQLite
254
12.5 libxml和cURL
256
后记
261
术语表
262

# 《C程序设计新思维》

## 精彩短评

- 1、翻译实在太差了。新手还在入门如我，还是有点东西可以学的。
- 2、目录好过内容。因为oreilly的名头和目录才买了这本书。其实挺好的一个主题，但每个内容作者都蜻蜓点水的说了一两句，没有深入介绍。
- 3、翻译挺差。
- 4、入门。还有东西可学。
- 5、翻译让我放弃此书
- 6、简单的读了一下，无非讲了一些C语言的基础知识，还有其需要的工具链。
- 7、我觉得写得比较杂乱，事实上这个不是一本完整的语法细节书而是工程实践指南和总结。需要你有比较好的基础然后才能理解作者在讲述什么和目的。因为覆盖的很多，所以有不少地方有点太过简练。但是还是很不错的书！

1、v2贴子：<http://v2ex.com/t/239274>书是好书，读书笔记另外开贴，这里只谈翻译，真像是机器翻译注：以下页码为中文版页码，[]内是我发的牢骚P28:1.7.3嵌入文档[看看原文，所谓的“嵌入文档”原来是“here document”。另外，P26页章节标题“通过本地文档来编译C程序”中的“本地文档”其实还是“here document”。]P35中间：@可以用来显示一些列的元素阵列原文：the @ shows you a sequence of elements in an array[你妹，array翻译成阵列，就算是学过一点点编程的人都知道这是数组好吗？？]P37最后：例2-1一组用于在GDB中方便地显示链表的宏——关于你可能会用到的最善于阐述的调试宏原文：Example 2-1. A set of macros to easily display a linked list in GDB—about the most elaborate debugging macro you'll ever need[请告诉我，什么叫“最善于阐述”？elaborate这里明显是“复杂”的意思]P39倒数第四行：只有可执行文件被分析，连接的库。[是的，我没打错，句号前面就只有4个字。]原文：Only the executable is profiled, not libraries that are linked to it.[明显少字了。校对在哪里？我想啪啪啪你（打屁股，别想歪。。。）]P41中间这一整段：对所有的C用法提供建议是很难的。它可能是在同一个语言里，但是写一个全天候运行的企业系统的关键应用和写一个你和同事们研究用的仿真程序可是两回事。原文：It's hard to give advice for all uses of C. It may all be in the same language, but writing for a mission-critical enterprise system that is expected to have 100% runtime is not the same game as writing a research simulation to be run by you and a few colleagues.[这句话勉强还能反映原文的意思。作者用的“100% runtime”这词翻译成全天候总觉得怪怪的。全天候，百度百科解释：谓不受天气限制，能适应各种复杂的气候条件。如：全天候战斗机。][我斗胆翻译一下第二句：即使用的是同一种编程语言，但是编写用于需保证100%运行时间的企业级系统中的关键任务代码，与编写仅供你和同事少数几个人使用的、出于研究目的仿真代码，这可是两码事。]就是说，我的建议是：除非我期望这些代码某一天在一个循环中返回，我会忽略任何小于100KB的内存泄漏。原文：That said, here's my advice: unless I expect that the code could someday be rerun in the center of a loop, I take any memory leak under maybe 100 KB as ignorable.[译者同志，请睁大眼睛，不要把“rerun”看作“return”了。看错一个字，整句话的意思都不对了啊！]我们在拥有数以G计的内存的计算机上工作，因此我们花在追踪细微的内存泄漏的时间看起来不太有可能对提高程序的效率有任何影响。原文：We're working on computers with gigabytes of memory, so the time spent tracking down those little leaks is unlikely to have any appreciable effect on the efficacy of the program.[近视眼译者把“efficacy功效”错看成“efficiency效率”][刚读到这段话（使用valgrind检测内存泄漏一节最后一段），作者建议读者可以放过小小的内存泄漏时，我的心情是震惊的。但是如果读原文，可以了解作者的原意——作者认为如果为了实现某种功能写点简单的小程序，而不是编写企业级应用，那么应以实现功能为主，没必要把大把时间花在追查一点点内存泄漏上。是的，想写一个没有内存泄漏的程序需要注意很多细节，如果初学者被这些细节绊住而没有信心往下学习，可真是罪过。][再一次谴责译者，整段话翻成这样，真是毁人不倦]P51第二行：。。。我们先考虑一个现代的问题。。。原文：consider the complementary question[译者把“complementary辅助的”看成了“contemporary现代的”]P52第一段：被放在火上烤。。烤在火上原文：on fire[着火]P52第三段结尾：本节的余下部分考察一个善意的错误。原文：The rest of this section considers the bona fide errors.[本节的其余部分关注那些真正的错误]P52中间：#inckude &lt;stdlib.h&gt;原文：#include &lt;stdlib.h&gt;P55第二段：Autotools是一个为系统自动产生完美的makefile的系统，它现在也开始聚焦在如何发布代码上。原文：In the present day, Autotools, a system for autogenerating the perfect makefile for a given system, is central to how code is distributed.[当今，Autotools，一套为给定系统生成完美的makefile的系统，处于发布代码工作的核心位置]P55第三段：不过一言以蔽之，makefile其实就是被精心组织的shell元素，用以自动化你的工作。原文：But to a first approximation, makefiles are organized sets of shell commands, so you'll need to get to know the various facilities the shell offers for automating your work.[错译][makefile其实大致可以看作一组shell命令，所以你得对shell为了使你的工作自动化而提供的各种工具有点了解][另外，这段话最后一个标点在原文里是个引号，因为下面列出了好几点，译文莫名其妙的变成了句号，像是一句话无疾而终了][ps：刚发现这本书去年已经出了第二版，而2015年5月出版的本书仍然是第一版]

2、因为oreilly的名头和目录才买了这本书。其实挺好的一个主题，但每个内容作者都蜻蜓点水的说了一两句，没有深入介绍，也没提供更多的线索。就像作者在的Q&A里说的，这本书只给了读者必须知道的方向，剩下的工作就是读者去自己习惯的搜索引擎中查找相关的主题。但是说实话，这所谓

## 《C程序设计新思维》

方向也给的有些过于简单了。感觉就像按照目录中的条目，google一下，然后把每个条目的tutorial或者文档提炼一下汇编成书.....

# 《C程序设计新思维》

## 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:[www.tushu000.com](http://www.tushu000.com)