

<<高质量程序设计指南>>

图书基本信息

书名：<<高质量程序设计指南>>

13位ISBN编号：9787121186172

10位ISBN编号：7121186179

出版时间：2012-10

出版时间：电子工业出版社

作者：林锐,韩永泉

页数：396

字数：576000

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：<http://www.tushu007.com>

<<高质量程序设计指南>>

前言

本书出版后一直受到不少软件公司和C++程序员的关注，但不知不觉间也绝迹很久。

不断有读者问我从何处可以买到本书、什么时候再版。

现在这一版本的推出，也可视作对这一询问的一种回答。

说来惭愧，我从2002年写完本书第1版后，再也没有接触过C++编程，现在对C++已经很陌生了。

2004年1月我离开上海贝尔，创办了上海漫索计算机科技有限公司，专注于IT企业的研发管理整体解决方案（包括软件产品和咨询服务）。

我自己已经从技术专家转型为企业管理者，关注商务多于软件技术。

对于出版本书第3版，我的确心有余而力不足。

幸好第2版的作者韩永泉仍然从事应用软件开发，宝刀未老，让我对第3版的质量充满信心。

在撰写这一版的时候，为了更进一步突出本书一贯强调的“高质量程序设计”理念，对原书前版的内容做了一些调整：首先是进行了全面的修订，改正了所有已经发现的错误，并对原有部分章节的内容进行了补充；其次，删除了第2版的第2章和第17章（名字空间和模板）。

根据我们的观察，除非是开发类库等通用程序，第17章的内容在现阶段对应用软件开发人员一般不具有实际指导价值；最后，增加了大约10个小节的内容，分散在各章中。

这些增加的内容是实际应用软件开发过程中经常会用到的技术，可以显著地提高编程效率，增强软件的健壮性和可移植性。

不论本书第1版和第2版是好是差，它都被过度地使用了，产生了令作者始料不及的影响。

本书的试题被国内软件公司大面积地用于C++程序员招聘考试，结果事先看过答案的应试者考了高分而被录取，还真有人向我致谢；也有不少人未看过答案而考了低分未被录取，在网上把作者骂一通。本书的试题和答案早在2002年就公开了，不知有多少人看过，我很奇怪怎么到现在还被煞有介事地用于考试。

希望读者正确地使用本书：请您学习和应用您（或公司）认为好的东西，不要把本书当作标准来看待，不要全部照搬，也不必花费很多时间去争议本书是好还是坏。

如果你发现书中的错误或不妥之处，请及时告知作者韩永泉，或发邮件至邮箱，或直接上他的Blog与他交流。

林锐上海漫索计算机科技有限公司 总经理睿泰科技集团董事、首席研发管理专家

<<高质量程序设计指南>>

内容概要

高质量程序设计是软件行业的薄弱环节，大部分企业为此付出了高昂的代价，只能通过大量的测试和改错来提高软件产品的质量。

因此，如何让程序员熟练地掌握编程技术和编程规范，在开发过程中内建高质量代码，是IT企业面临的主要挑战之一。

本书以轻松幽默的笔调向读者论述了高质量软件开发方法与 C++ / C 编程规范，而这也是作者多年从事软件开发工作的经验总结。

全书共17章，第1章到第4章重点介绍软件质量和基本的程序设计方法；第5章到第16章重点阐述C++ / C 编程风格、面向对象程序设计方法和一些技术专题；第17章阐述STL的原理和使用方法。

本书第1

版和第2版部分章节曾经在网上广泛流传，被国内IT企业的不少软件开发人员采用。

本书的附录C《大学十年》是作者在网上发表的一个短篇传记，文中所描述的充满激情的学习和生活态度，感染了大批莘莘学子。

<<高质量程序设计指南>>

作者简介

林锐, 1973年生。

1994年和1996年获西安电子科技大学应用物理学学士学位和微电子硕士学位, 2000年获浙江大学计算机应用博士学位。

大学期间两度被评为中国百名跨世纪优秀大学生, 1996年获电子工业部科技进步二等奖, 1997年获首届中国大学生电脑大赛软件一等奖。

2000年7月加入上海贝尔有限公司, 从事软件工程、项目管理和CMM的研究推广工作, 2003年7月当选为

Alcatel 集团技术专家 (Alcatel

集团授予为保持全球技术领先地位做出突出贡献的技术专家荣誉和资格)。

2004年初创建上海漫索计算机科技有限公司 (<http://www.mansuo.com>

) , 致力于创作适合国内软件企业需求的管理方法论和软件产品。

从2000年至今, 林锐博士在程序设计、用户界面设计、软件工程、项目管理、CMMI与软件过程改进、IT企业研发管理、软件企业管理等领域累计出版著作十部, 拥有数十万名读者, 成为国内软件企业管理方法论的领先者。

韩永泉, 1975年生。

1994年至2001年就读于西安电子科技大学计算机系, 获硕士学位。

2001年4月加入上海大唐移动通信技术有限公司, 担任高级软件工程师, 从事电信设备网管软件的研发工作。

2004年加入北京新岸线软件科技有限公司, 从事数字电视和手机电视相关软件产品的设计、开发和研发管理工作, 曾负责所在公司与上海漫索计算机科技有限公司合作开展的软件过程改进和研发管理解决方案的实施项目。

2010年加入诺基亚西门子 (NSN) 。

面向对象和面向组件软件开发技术及编程技术的爱好者。

<<高质量程序设计指南>>

书籍目录

第1章 高质量软件开发之道

1.1 软件质量基本概念

1.1.1 如何理解软件的质量

1.1.2 提高软件质量的基本方法

1.1.3 “零缺陷”理念

1.2 细说软件质量属性

1.2.1 正确性

1.2.2 健壮性

1.2.3 可靠性

1.2.4 性能

1.2.5 易用性

1.2.6 清晰性

1.2.7 安全性

1.2.8 可扩展性

1.2.9 兼容性

1.2.10 可移植性

1.3 人们关注的不仅仅是质量

1.3.1 质量、生产率和成本之间的关系

1.3.2 软件过程改进的基本概念

1.4 高质量软件开发的基本方法

1.4.1 建立软件过程规范

1.4.2 复用

1.4.3 分而治之

1.4.4 优化与折中

1.4.5 技术评审

1.4.6 测试

1.4.7 质量保证

1.4.8 改错

1.5 关于软件开发的一些常识和思考

1.5.1 有最好的编程语言吗

1.5.2 编程是一门艺术吗

1.5.3 编程时应该多使用技巧吗

1.5.4 换更快的计算机还是换更快的算法

1.5.5 错误是否应该分等级

1.5.6 一些错误的观念

1.6 小结

第2章 编程语言发展简史

2.1 编程语言大事记

2.2 Ada的故事

2.3 C/C++发展简史

2.4 Borland与Microsoft之争

2.5 Java阵营与Microsoft的较量

2.6 小结

第3章 程序的基本概念

3.1 程序设计语言

<<高质量程序设计指南>>

3.2 语言实现

3.3 程序库

3.4 开发环境

3.5 程序的工作原理

3.6 良好的编程习惯

第4章 C++/C程序设计入门

4.1 C++/C程序的基本概念

4.1.1 启动函数main()

4.1.2 命令行参数

4.1.3 内部名称

4.1.4 连接规范

4.1.5 变量及其初始化

4.1.6 C Runtime Library

4.1.7 编译时和运行时的不同

4.1.8 编译单元和独立编译技术

4.2 基本数据类型和内存映像

4.3 类型转换

4.3.1 隐式转换

4.3.2 强制转换

4.4 标识符

4.5 转义序列

4.6 运算符

4.7 表达式

4.8 基本控制结构

4.9 选择（判断）结构

4.9.1 布尔变量与零值比较

4.9.2 整型变量与零值比较

4.9.3 浮点变量与零值比较

4.9.4 指针变量与零值比较

4.9.5 对if语句的补充说明

4.9.6 switch结构

4.10 循环（重复）结构

4.10.1 for语句的循环控制变量

4.10.2 循环语句的效率

4.11 结构化程序设计原理

4.12 goto/continue/break语句

4.13 示例

第5章 C++/C常量

5.1 认识常量

5.1.1 字面常量

5.1.2 符号常量

5.1.3 契约性常量

5.1.4 枚举常量

5.2 正确定义符号常量

5.3 const与#define的比较

5.4 类中的常量

5.5 实际应用中如何定义常量

<<高质量程序设计指南>>

第6章 C++/C函数设计基础

6.1 认识函数

6.2 函数原型和定义

6.3 函数调用方式

6.4 认识函数堆栈

6.5 函数调用规范

6.6 函数连接规范

6.7 参数传递规则

6.8 返回值的规则

6.9 函数内部实现的规则

6.10 存储类型及作用域规则

6.10.1 存储类型

6.10.2 作用域规则

6.10.3 连接类型

6.11 递归函数

6.12 使用断言

6.13 使用const提高函数的健壮性

6.13.1 用const修饰函数的参数

6.13.2 用const修饰函数的返回值

第7章 C++/C指针、数组和字符串

7.1 指针

7.1.1 指针的本质

7.1.2 指针的类型及其支持的运算

7.1.3 指针传递

7.2 数组

7.2.1 数组的本质

7.2.2 二维数组

7.2.3 数组传递

7.2.4 动态创建、初始化和删除数组的方法

7.3 字符数组、字符指针和字符串

7.3.1 字符数组、字符串和'\0'的关系

7.3.2 字符指针的误区

7.3.3 字符串拷贝和比较

7.4 函数指针

7.5 引用和指针的比较

第8章 C++/C高级数据类型

8.1 结构 (struct)

8.1.1 关键字struct与class的困惑

8.1.2 使用struct

8.1.3 位域

8.1.4 成员对齐

8.2 联合 (Union)

8.3 枚举 (Enum)

8.4 文件

第9章 C++/C编译预处理

9.1 文件包含

9.1.1 内部包含卫哨和外部包含卫哨

<<高质量程序设计指南>>

- 9.1.2 头文件包含的合理顺序
- 9.2 宏定义
- 9.3 条件编译
 - 9.3.1 #if、#elif和#else
 - 9.3.2 #ifdef 和 #ifndef
- 9.4 #error
- 9.5 #pragma
- 9.6 #和##运算符
- 9.7 预定义符号常量
- 第10章 C++/C文件结构和程序版式
 - 10.1 程序文件的目录结构
 - 10.2 文件的结构
 - 10.2.1 头文件的用途和结构
 - 10.2.2 版权和版本信息
 - 10.2.3 源文件结构
 - 10.3 代码的版式
 - 10.3.1 适当的空行
 - 10.3.2 代码行及行内空格
 - 10.3.3 长行拆分
 - 10.3.4 对齐与缩进
 - 10.3.5 修饰符的位置
 - 10.3.6 注释风格
 - 10.3.7 ADT/UDT版式
- 第11章 C++/C应用程序命名规则
 - 11.1 共性规则
 - 11.2 简单的Windows应用程序命名
- 第12章 C++面向对象程序设计方法概述
 - 12.1 漫谈面向对象
 - 12.2 对象的概念
 - 12.3 信息隐藏与类的封装
 - 12.4 类的继承特性
 - 12.5 类的组合特性
 - 12.6 动态特性
 - 12.6.1 虚函数
 - 12.6.2 抽象基类
 - 12.6.3 动态绑定
 - 12.6.4 运行时多态
 - 12.6.5 多态数组
 - 12.7 C++对象模型
 - 12.7.1 对象的内存映像
 - 12.7.2 隐含成员
 - 12.7.3 C++编译器如何处理成员函数
 - 12.7.4 C++编译器如何处理静态成员
 - 12.8 小结
- 第13章 对象的初始化、拷贝和析构
 - 13.1 构造函数与析构函数的起源
 - 13.2 为什么需要构造函数和析构函数

<<高质量程序设计指南>>

- 13.3 构造函数的成员初始化列表
- 13.4 对象的构造和析构次序
- 13.5 构造函数和析构函数的调用时机
- 13.6 构造函数和赋值函数的重载
- 13.7 示例：类String的构造函数和析构函数
- 13.8 何时应该定义拷贝构造函数和拷贝赋值函数
- 13.9 示例：类String的拷贝构造函数和拷贝赋值函数
- 13.10 用偷懒的办法处理拷贝构造函数和拷贝赋值函数
- 13.11 如何实现派生类的基本函数
- 第14章 C++函数的高级特性
 - 14.1 函数重载的概念
 - 14.1.1 重载的起源
 - 14.1.2 重载是如何实现的
 - 14.1.3 小心隐式类型转换导致重载函数产生二义性
 - 14.2 成员函数的重载、覆盖与隐藏
 - 14.2.1 重载与覆盖
 - 14.2.2 令人迷惑的隐藏规则
 - 14.2.3 摆脱隐藏
 - 14.3 参数的默认值
 - 14.4 运算符重载
 - 14.4.1 基本概念
 - 14.4.2 运算符重载的特殊性
 - 14.4.3 不能重载的运算符
 - 14.4.4 重载++和--
 - 14.5 函数内联
 - 14.5.1 用函数内联取代宏
 - 14.5.2 内联函数的编程风格
 - 14.5.3 慎用内联
 - 14.6 类型转换函数
 - 14.7 const成员函数
- 第15章 C++异常处理和RTTI
 - 15.1 为什么要使用异常处理
 - 15.2 C++异常处理
 - 15.2.1 异常处理的原理
 - 15.2.2 异常类型和异常对象
 - 15.2.3 异常处理的语法结构
 - 15.2.4 异常的类型匹配规则
 - 15.2.5 异常说明及其冲突
 - 15.2.6 当异常抛出时局部对象如何释放
 - 15.2.7 对象构造和析构期间的异常
 - 15.2.8 如何使用好异常处理技术
 - 15.2.9 C++的标准异常
 - 15.3 虚函数面临的难题
 - 15.4 RTTI及其构成
 - 15.4.1 起源
 - 15.4.2 typeid运算符
 - 15.4.3 dynamic_cast运算符

<<高质量程序设计指南>>

15.4.4 RTTI的魅力与代价

第16章 内存管理

16.1 内存分配方式

16.2 常见的内存错误及其对策

16.3 指针参数是如何传递内存的

16.4 free和delete把指针怎么啦

16.5 动态内存会被自动释放吗

16.6 杜绝“野指针”

16.7 有了malloc/free为什么还要new/delete

16.8 malloc/free的使用要点

16.9 new有3种使用方式

16.9.1 plain new/delete

16.9.2 nothrow new/delete

16.9.3 placement new/delete

16.10 new/delete的使用要点

16.11 内存耗尽怎么办

16.12 用对象模拟指针

16.13 泛型指针auto_ptr

16.14 带有引用计数的智能指针

16.15 智能指针作为容器元素

第17章 学习和使用STL

17.1 STL简介

17.2 STL头文件的分布

17.2.1 容器类

17.2.2 泛型算法

17.2.3 迭代器

17.2.4 数学运算库

17.2.5 通用工具

17.2.6 其他头文件

17.3 容器设计原理

17.3.1 内存映像

17.3.2 存储方式和访问方式

17.3.3 顺序容器和关联式容器的比较

17.3.4 如何遍历容器

17.3.5 存储空间重分配问题

17.3.6 什么样的对象才能作为STL容器的元素

17.4 迭代器

17.4.1 迭代器的本质

17.4.2 迭代器失效及其危险性

17.4.3 如何在遍历容器的过程中正确删除元素

17.5 存储分配器

17.6 适配器

17.7 泛型算法

17.8 一些特殊的容器

17.8.1 string类

17.8.2 bitset并非set

17.8.3 节省存储空间的vector

<<高质量程序设计指南>>

17.8.4 空容器

17.9 STL容器特征总结

17.10 STL使用心得

附录A C++/C试题

附录B C++/C试题答案与评分标准

附录C 大学十年

附录D 《大学十年》后记

附录E 术语与缩写解释

<<高质量程序设计指南>>

媒体关注与评论

在中国，我很欣赏以至于崇拜的一个人便是林锐先生了。

我写文章时经常引用他的话--“作一个优秀、正直、诚实的人，为中国软件产业的振兴而努力！”

他的这本书颇有经典外版书的特征，集幽默与严谨于一身，很少有书能如此吸引我。

诚然，它不是一本“巨著”，但却是学习C++过程中可以让你享受，或停下来思考的一本书。

你不必像看《C++编程思想》一样慢而稳地看这本书，你甚至可以一下午就搞定它，但它却是一本你最好能拥有的书。

——@ouyangj0林锐先生的书是在你懂了一些语言之后，教给你一些真正做项目要注意的问题。

对学过C语言和C++语言而没有参与过大项目的学生以及刚参加编程工作的研发人员来说，都有较强参考价值。

即使是有多年编程工作经验的人，也建议你翻一下。

——@macogg这本书读了两年多，当这些东西经潜移默化成为本能时，的确对高质量的产品设计起到了不少作用！

——@gavin.mu绝对是一本好书，让我弄懂了很多以前困惑的问题。

总体感觉，这本书不是一本入门教材，也不是一本像《Inside The C++ Object Model》一样深奥的典籍，它更像两者之间的一个梯子，让已经有一定基础的读者经由它爬上能够理解《Inside The C++ Object Model》的高度。

——@smartm看过第一版和第二版，堪称中国本土版的《Effective C++》和《More Effective C++》。

——@cn_lb一口气读完，如饮甘饴。

作者把个人的社会责任感融注于一本技术性书籍当中，语言风趣，文字简洁，读起来别有一番风味。

老外的书大多不符合中国人的语言习惯，读起来很晦涩，这本书把中文运用得很好。

——@SED

<<高质量程序设计指南>>

编辑推荐

同类书目请见：《Effective C++：改善程序与设计的55个具体做法（第三版）（评注版）》《More Effective C++：35个改善编程与设计的有效方法（中文版）》《Effective C++：改善程序与设计的55个具体做法（第三版）中文版》《Boost程序库完全开发指南：深入C++“准”标准库（修订版）》《大学十年》感染一代IT人，林锐亲述多年一线经验。

《高质量程序设计指南：C++/C语言（第3版）》提供内建高质量代码必须熟练掌握的编程技术与规范。
“高质量”试图挽回的是--投入大量人、材、物力的事后检测和补救。

<<高质量程序设计指南>>

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:<http://www.tushu007.com>