

<<C++模板元编程>>

图书基本信息

书名：<<C++模板元编程>>

13位ISBN编号：9787111267423

10位ISBN编号：7111267427

出版时间：2010-1

出版时间：机械工业出版社

作者：David Abrahams, Aleksey Gurtovoy

页数：277

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>

<<C++模板元编程>>

前言

1998年，Dave获权参加在德国Dagstuhl Castle举行的泛型编程研讨会。在研讨会临近尾声时，热情的Kristof Czarnecki和Ulrich Eisenecker（在产生式编程领域颇有声望）散发了一些C++源代码，那是采用C++模板编写的完整的Lisp实现清单。那时候，对于Dave而言那不过是个新奇的玩具而已，是对模板系统迷人但不切实际的“劫持”，以证实我们可以编写执行于编译期的程序。他从未想象有朝一日会在自己的大多数日常编程工作中发挥元编程（metaprogramming）的作用。在许多方面，那个模板代码集合是Boost元编程库（Metaprogramming Library，MPL）的先驱：它可能是第一个设计用于将编译期c++从一个特别的“模板技巧”集合转变为正规的、易理解的软件工程的范例的程序库。随着用于编写和理解元程序（metaprogram）的高阶工具的出现，我们发现使用这些技术不但切实可行，而且简单、有趣，并常常带来令人惊讶的威力。撇开存在许多采用模板元编程和MPL的真实系统不谈，很多人仍然将元编程视作神秘的魔法，并且在日常产品代码中避免使用它。如果你从未进行过任何元编程，你甚至都看不出它和你所做的工作有什么明显的关系。在这本书中，我们希望能够揭开它的神秘面纱，使你不但对如何进行元编程有所理解，对为何（以及何时）进行元编程也会有很好的认知。

<<C++模板元编程>>

内容概要

《C++模板元编程》是关于C++模板元编程的著作。

《C++模板元编程》主要介绍Traits和类型操纵、深入探索元函数、整型外覆器和操作、序列与迭代器、算法、视图与迭代器适配器、诊断、跨越编译期和运行期边界、领域特定的嵌入式语言、DSEL设计演练，另外附录部分还介绍了预处理元编程概述、typename和template关键字。

《C++模板元编程》通过理论联系实际，深入讲解了C++高级编程技术。

《C++模板元编程》适合中、高阶C++程序员等参考。

附赠光盘内容包含所有Boost C++程序库：Boost Metaprogramming Library及其参考文件，还包含所有随书代码示例以及大量的补充材料。

作者简介

David Abrahams, Boost C++程序库开发组的一名发起成员和主持人。

自从1996年起, Dave就是一名ANSI/ISO C++委员会成员, 他因对C++标准程序库异常处理的理论、规格和实现的贡献而名声鹊起。

他的公司Boost Consulting提供了与Boost有关的支持和开发服务, 以及软件构建艺术的职业培训。

Aleksey Gurtovoy, MetaCommunications的一名技术领导, 并且是Boost C++社区的功勋成员。

他是Boost Metaprogramming Library最初的作者。

自1993年起他就开始研究C++并使用它工作, Aleksey拥有俄罗斯克拉斯诺雅茨克州立技术大学计算机科学理学硕士学位。

<<C++模板元编程>>

书籍目录

译者序序言前言致谢第1章 概述1.1 起步走1.2 元程序的概念1.3 在宿主语言中进行元编程1.4 在C++中进行元编程1.4.1 数值计算1.4.2 类型计算1.5 为何进行元编程1.5.1 替代方案1：运行期计算1.5.2 替代方案2：用户分析1.5.3 为何进行C++元编程1.6 何时进行元编程1.7 为何需要元编程程序库第2章 Traits和类型操纵2.1 类型关联2.1.1 采用一种直接的方式2.1.2 采用一种迂回方式2.1.3 寻找一个捷径2.2 元函数2.3 数值元函数2.3.4 在编译期作出选择2.4.1 进一步讨论iter_swap2.4.2 美中不足2.4.3 另一个美中不足2.4.4 “美中不足”之外覆器2.5 Boost Type Traits程序库概览2.5.1 一般知识2.5.2 主类型归类 (Primary Type Categorization) 2.5.3 次类型归类 (Secondary Type Categorization)2.5.4 类型属性2.5.5 类型之间的关系2.5.6 类型转化2.6 无参元函数2.7 元函数的定义2.8 历史2.9 细节2.9.1 特化2.9.2 实例化2.9.3 多态2.10 练习第3章 深入探索元函数3.1 量纲分析3.1.1 量纲的表示3.1.2 物理量的表示3.1.3 实现加法和减法3.1.4 实现乘法3.1.5 实现除法3.2 高阶元函数3.3 处理占位符3.3.1 lambda元函数3.3.2 apply元函数3.4 lambda的其他能力3.4.1 偏元函数应用3.4.2 元函数复合3.5 Lambda的细节3.5.1 占位符3.5.2 占位符表达式的定义3.5.3 Lambda和非元函数模板3.5.4 “懒惰”的重要性3.6 细节3.7 练习第4章 整型外覆器和操作4.1 布尔外覆器和操作4.1.1 类型选择4.1.2 缓式类型选择4.1.3 逻辑运算符4.2 整数外覆器和运算4.2.1 整型运算符4.2.2 _c整型速记法4.3 练习第5章 序列与迭代器5.1 Concepts5.2 序列和算法5.3 迭代器5.4 迭代器Concepts5.4.1 前向迭代器5.4.2 双向迭代器5.4.3 随机访问迭代器5.5 序列Concepts5.5.1 序列遍历Concepts5.5.2 可扩展性5.5.3 关联式序列5.5.4 可扩展的关联式序列5.6 序列相等性5.7 固有的序列操作5.8 序列类5.8.1 list5.8.2 vector5.8.3 deque5.8.4 range_c5.8.5 map5.8.6 set5.8.7 iterator_range5.9 整型序列外覆器5.10 序列派生5.11 编写你自己的序列5.11.1 构建tiny序列5.11.2 迭代器的表示5.11.3 为tiny实现at.5.11.4 完成tiny_iterator的实现5.11.5 begin和end5.11.6 加入扩充性5.12 细节5.13 练习第6章 算法6.1 算法、惯用法、复用和抽象6.2 MPL的算法6.3 插入器6.4 基础序列算法6.5 查询算法6.6 序列构建算法6.7 编写你自己的算法6.8 细节6.9 练习第7章 视图与迭代器适配器7.1 一些例子7.1.1 对从序列元素计算出来的值进行比较7.1.2 联合多个序列7.1.3 避免不必要的计算7.1.4 选择性的元素处理7.2 视图Concept7.3 迭代器适配器7.4 编写你自己的视图7.5 历史7.6 练习第8章 诊断8.1 调试错误8.1.1 实例化回溯8.1.2 错误消息格式化怪癖8.2 使用工具进行诊断分析8.2.1 听取他者的意见8.2.2 使用导航助手8.2.3 清理场面8.3 有目的的诊断消息生成8.3.1 静态断言8.3.2 MPL静态断言8.3.3 类型打印8.4 历史8.5 细节8.6 练习第9章 跨越编译期和运行期边界9.1 for_each9.1.1 类型打印9.1.2 类型探访9.2 实现选择9.2.1 if语句9.2.2 类模板特化9.2.3 标签分派9.3 对象生成器9.4 结构选择9.5 类复合9.6 (成员)函数指针作为模板实参9.7 类型擦除9.7.1 一个例子9.7.2 一般化9.7.3 “手工”类型擦除9.7.4 自动类型擦除9.7.5 保持接口9.8 奇特的递归模板模式9.8.1 生成函数9.8.2 管理重载决议9.9 显式管理重载集9.10 sizeof技巧9.11 总结9.12 练习第10章 领域特定的嵌入式语言10.1 一个小型语言10.2 路漫漫其修远兮10.2.1 Make工具语言10.2.2 巴科斯-诺尔模式10.2.3 YACC10.2.4 DSL摘要10.3 DSL10.4 C++用作宿主语言10.5 Blitz++和表达式模板10.5.1 问题10.5.2 表达式模板10.5.3 更多的Blitz++魔法10.6 通用DSEL10.6.1 具名参数10.6.2 构建匿名函数10.7 BoostSpirit程序库10.7.1 闭包10.7.2 子规则10.8 总结10.9 练习第11章 DSEL设计演练11.1 有限状态机11.1.1 领域抽象11.1.2 符号11.2 框架设计目标11.3 框架接口基础11.4 选择一个DSL11.4.1 转换表11.4.2 组装成一个整体11.5 实现11.6 分析11.7 语言方向11.8 练习附录A 预处理元编程简介附录B typename和template关键字附录C 编译期性能附录D MPL可移植性摘要参考文献

<<C++模板元编程>>

章节摘录

第8章 诊断 因为c++元程序执行于编译期，所以给调试（debugging）工作带来了特别的挑战。没有调试器允许我们单步跟踪元程序的执行，设置断点，检视数据等等。

这类调试工作需要我们对编译器内部状态的互动式探查。

我们能做的全部事情，就是等待编译过程失败，然后破译编译器倾泻到屏幕上的错误信息。

c++模板的诊断（diagnostics）是一种常见的让人感到挫折的源泉，因为它们通常与导致错误的原因没有明显的关系，并且呈现了大量的无用信息。

在本章中，我们将讨论如何理解元编程程序员通常遭遇的错误种类，甚至如何使得这些诊断屈服于我们的“邪恶”的目的。

c++标准将错误报告的具体实现方式完全留给编译器实现者，因此我们将讨论几款不同的编译器的行为，通常是以批评的措辞。

因为你的编译器的错误消息是你能得到的全部帮助，所以对工具的选择会对你调试元程序的能力产生巨大的影响。

如果你在构建程序库，当出现错误时，你的客户对工具的选择将会影响他们对代码的理解，也会影响到你花在回答问题上的时间。

因此，即使当我们在讨论你一般不使用的编译器时，我们也建议你聚精会神，因为你也许会发现你希望将它加入自己的工具箱，或者，希望为可能使用该款编译器的客户提供特别的支持。

同样，如果我们看上去是在抨击你喜欢的工具，希望你不要感觉自己受到了冒犯。

8.1 调试错误 这一节的标题实际上取自另一本书【VJ02】，但用在这本书里也极为切题。

实际上，模板错误报告通常如此像《战争与和平》，很多程序员忽略它们并且求助于随机的代码并改来改去，希望能碰巧改正确。

在这一节中，我们将为你提供工具来剔除那些冗长的诊断信息，并且帮助你找到解决问题的正确途径。

<<C++模板元编程>>

媒体关注与评论

“如果你像我一样，对人们在模板元编程(Template Metaprogramming, TMP)方面所做的工作感到兴奋，但对缺乏这方面的清晰指导和强有力的工具感到沮丧。

那好，这本书正是我们期待已久的。

借助于优秀的Boost Metaprogramming Library，DaVid和Afeksey将TMP从实验室带到了生产一线，以易读的文字和实际的例子向我们展示了‘编译期STL’与其运行期对应物具有同样的能耐。

本书既可以作教程，也可以用作专家手册。

这是一本关于C++的扛鼎之作。

” ——Chuck Allison . The C++Source编辑

<<C++模板元编程>>

编辑推荐

本书清晰地揭示了现代C++最具威力的使用手法，将实际的工具和技术交付普通程序员的手中。

元编程是产生或操纵程序码的程序。

自从泛型编程被引入C++中以来，程序员们已经发现用于当程序被编译时对其进行操纵的无数“tricks template”，这些tricks有效地消除了横亘在程序和元编程之间的藩篱。

尽管C++专家们对这种能力的兴奋已经波及整个C++社区，然而它们的实际应用对于大多数程序员来说仍然是遥不可及。

本书解释了何谓元编程以及如何最佳化地使用它，为你在自己的工作中有效地使用模板元编程提供了必备的基础。

本书瞄准于任何熟悉标准模板库(Standard Template Library, STL)惯用法的程序员。

C++高级用户会获得对既有工作新的洞察以及对元编程领域新的认知。

那些已经学习了一些高级模板技术的中级程序员将会明白这些tricks是从哪儿适合大画面的。

并将获得有秩序地使用它们所需的概念基础。

对于那些已经嗅到元编程味道但对其仍然感到神秘的程序员而言，最终将获得对元编程如何工作、何时工作以及为何工作的清晰的理解。

无论如何，所有读者都将获得一个可以自由支配的、威力空前的新工具：Boost MetaProgramming Library。

<<C++模板元编程>>

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>