

<<Boost程序库完全开发指南>>

图书基本信息

书名：<<Boost程序库完全开发指南>>

13位ISBN编号：9787121166297

10位ISBN编号：7121166291

出版时间：2012-5

出版时间：电子工业

作者：罗剑锋

页数：578

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>

## <<Boost程序库完全开发指南>>

### 内容概要

Boost是一个功能强大、构造精巧、跨平台、开源并且完全免费的C++程序库，有着“C++‘准’标准库”的美誉。

Boost由C++标准委员会部分成员所设立的Boost社区开发并维护，使用了许多现代C++编程技术，内容涵盖字符串处理、正则表达式、容器与数据结构、并发编程、函数式编程、泛型编程、设计模式实现等许多领域，极大地丰富了C++的功能和表现力，能够使C++软件开发更加简捷、优雅、灵活和高效。

《Boost程序库完全开发指南：深入C++“准”标准库（修订版）》基于Boost1.42版，介绍了其中的所有99个库，并且详细深入地讲解了其中数十个库，同时实现了若干颇具实用价值的工具类和函数，可帮助读者迅速地理解掌握Boost的用法及其在实际开发工作中的应用。

《Boost程序库完全开发指南：深入C++“准”标准库（修订版）》内容丰富、结构严谨、详略得当、讲解透彻，带领读者领略了C++的最新前沿技术，相信会是每位C++程序员的必备工具书。

## <<Boost程序库完全开发指南>>

### 作者简介

罗剑锋（网名Chrono），1996年就读于东北财经大学，1997年开始接触C / C++。

1998年参加全国计算机等级考试，获高级程序员资质。

2003年毕业于北京理工大学，获计算机专业硕士学位。

目前供职于某部委下属软件公司，任项目经理，主要研究方向为C / C++、设计模式、密码学、数据库、嵌入式系统开发。

业余爱好是阅读、欣赏音乐和旅游。

## 书籍目录

第0章 导读 0.1 关于本书 0.2 读者对象 0.3 本书的术语与风格 0.4 本书的结构 0.5 如何阅读本书 第1章 Boost程序库总论 1.1 关于Boost 1.1.1 什么是Boost 1.1.2 安装Boost 1.1.3 使用Boost 1.2 关于STLport 1.2.1 什么是STLport 1.2.2 安装STLport 1.2.3 使用STLport 1.3 开发环境 1.3.1 STLport的编译方法 1.3.2 Boost的编译方法 1.3.3 Visual Studio 2005环境设置 第2章 时间与日期 2.1 timer库概述 2.2 timer 2.2.1 用法 2.2.2 类摘要 2.2.3 使用建议 2.3 progress\_timer 2.3.1 用法 2.3.2 类摘要 2.3.3 扩展计时精度 2.4 progress\_display 2.4.1 类摘要 2.4.2 用法 2.4.3 注意事项 2.5 date\_time库概述 2.5.1 编译date\_time库 2.5.2 date\_time库的基本概念 2.6 处理日期 2.6.1 日期 2.6.2 创建日期对象 2.6.3 访问日期 2.6.4 日期的输出 2.6.5 与tm结构的转换 2.6.6 日期长度 2.6.7 日期运算 2.6.8 日期区间 2.6.9 日期区间运算 2.6.10 日期迭代器 2.6.11 其他功能 2.6.12 综合运用 2.7 处理时间 2.7.1 时间长度 2.7.2 操作时间长度 2.7.3 时间长度的精确度 2.7.4 时间点 2.7.5 创建时间点对象 2.7.6 操作时间点对象 2.7.7 与tm、time\_t等结构的转换 2.7.8 时间区间 2.7.9 时间迭代器 2.7.10 综合运用 2.8 date\_time库的高级议题 2.8.1 编译配置宏 2.8.2 格式化时间 2.8.3 本地时间 2.8.4 序列化 2.9 总结 第3章 内存管理 3.1 smart\_ptr库概述 3.1.1 RA 机制 3.1.2 智能指针 3.2 scoped\_ptr 3.2.1 类摘要 3.2.2 操作函数 3.2.3 用法 3.2.4 与auto\_ptr的区别 3.3 scoped\_array 3.3.1 类摘要 3.3.2 用法 3.3.3 使用建议 3.4 shared\_ptr 3.4.1 类摘要 3.4.2 操作函数 3.4.3 用法 3.4.4 工厂函数 3.4.5 应用于标准容器 3.4.6 应用于桥接模式 3.4.7 应用于工厂模式 3.4.8 定制删除器 3.4.9 高级议题 3.5 shared\_array 3.5.1 类摘要 3.5.2 用法 3.6 weak\_ptr 3.6.1 类摘要 3.6.2 用法 3.6.3 获得this的shared\_ptr 3.7 intrusive\_ptr 3.8 pool库概述 3.9 pool 3.9.1 类摘要 3.9.2 操作函数 3.9.3 用法 3.10 object\_pool 3.10.1 类摘要 3.10.2 操作函数 3.10.3 用法 3.10.4 使用更多的构造参数 3.11 singleton\_pool 3.11.1 类摘要 3.11.2 用法 3.12 pool\_alloc 3.13 总结 第4章 实用工具 4.1 noncopyable 4.1.1 原理 4.1.2 用法 4.2 typeid 4.2.1 动机 4.2.2 用法 4.2.3 向typeid库注册自定义类 4.2.4 高级议题 4.3 optional 4.3.1 “无意义”的值 4.3.2 类摘要 4.3.3 操作函数 4.3.4 用法 4.3.5 工厂函数 4.3.6 高级议题 4.4 assign 4.4.1 使用操作符+=向容器增加元素 4.4.2 使用操作符( )向容器增加元素 4.4.3 初始化容器元素 4.4.4 减少重复输入 4.4.5 与非标准容器工作 4.4.6 高级用法 4.5 swap 4.5.1 原理 4.5.2 交换数组 4.5.3 特化std::swap 4.5.4 特化ADL可找到的swap 4.5.5 使用建议 4.6 singleton 4.6.1 boost.pool的单件实现 4.6.2 boost.serialization的单件实现 4.7 tribool 4.7.1 类摘要 4.7.2 用法 4.7.3 为第三态更名 4.7.4 输入输出 4.7.5 与optional的区别 4.8 operators 4.8.1 基本运算概念 4.8.2 算术操作符的用法 4.8.3 基类链 4.8.4 复合运算概念 4.8.5 相等与等价 4.8.6 解引用操作符 4.8.7 下标操作符 4.8.8 高级议题 4.9 exception 4.9.1 标准库中的异常 4.9.2 类摘要 4.9.3 向异常传递信息 4.9.4 更进一步的用法 4.9.5 包装标准异常 4.9.6 使用函数抛出异常 4.9.7 获得更多的调试信息 4.9.8 高级议题 4.10 uuid 4.10.1 类摘要 4.10.2 用法 4.10.3 生成器 4.10.4 增强的uuid类 4.10.5 与字符串的转换 4.10.6 SHA1摘要算法 4.11 config 4.11.1 BOOST\_STRINGIZE 4.11.2 BOOST\_STATIC\_CONSTANT 4.11.3 禁止编译器警告 4.11.4 其他工具 4.12 utility 4.12.1 BOOST\_BINARY 4.12.2 BOOST\_CURRENT\_FUNCTION 4.13 总结 第5章 字符串与文本处理 5.1 lexical\_cast 5.1.1 用法 5.1.2 异常bad\_lexical\_cast 5.1.3 对转换对象的要求 5.1.4 应用于自己的类 5.2 format 5.2.1 简单的例子 5.2.2 输入操作符% 5.2.3 类摘要 5.2.4 格式化语法 5.2.5 format的性能 5.2.6 高级用法 5.3 string\_algo 5.3.1 简单的例子 5.3.2 string\_algo概述 5.3.3 大小写转换 5.3.4 判断式(算法) 5.3.5 判断式(函数对象) 5.3.6 分类 5.3.7 修剪 5.3.8 查找 5.3.9 替换与删除 5.3.10 分割 5.3.11 合并 5.3.12 查找(分割)迭代器 5.4 tokenizer 5.4.1 类摘要 5.4.2 用法 5.4.3 分词函数对象 5.4.4 char\_separator 5.4.5 escaped\_list\_separator 5.4.6 offset\_separator 5.4.7 tokenizer库的缺陷 5.5 xpressive 5.5.1 两种使用方式 5.5.2 正则表达式语法简介 5.5.3 类摘要 5.5.4 匹配 5.5.5 查找 5.5.6 替换 5.5.7 迭代 5.5.8 分词 5.5.9 与regex的区别 5.5.10 高级议题 5.6 总结 第6章 正确性与测试 6.1 assert 6.1.1 基本用法 6.1.2 禁用断言 6.1.3 扩展用法 6.1.4 BOOST\_VERIFY 6.2 static\_assert 6.2.1 用法 6.2.2 使用建议 6.3 test 6.3.1 编译test库 6.3.2 最小化的测试套件 6.3.3 单元测试框架简介 6.3.4 测试断言 6.3.5 测试用例与套件 6.3.6 测试实例 6.3.7 测试夹具 6.3.8 测试日志 6.3.9 运行参数 6.3.10 函数执行监视器 6.3.11 程序执行监视器 6.3.12 高级议题 6.4 总结 ..... 第7章 容器与数据结构 第8章 算法 第9章 数学与数字 第10章 操作系统相关 第11章 函数与回调 第12章 并发编程 第13章 编程语言支持 第14章 其他Boost组件 第15章 Boost与设计模式 第16章 结束语 附录A 推荐书目 附录B 网络资源 附录C C++标准简述 附录D STL简述 附录E ref\_array实现代码



## &lt;&lt;Boost程序库完全开发指南&gt;&gt;

## 章节摘录

版权页：插图：1.1.2安装Boost从Boost网站(www.boost.org)下载boost\_1\_42\_0.7z，一个约30MB左右大小的压缩包文件，使用7Zip、WinRAR或者其他工具把该文件解压缩到硬盘任意位置即可完成安装，本书使用的路径是：D:\boost。

注意：Boost1.42版解压缩后约有200MB，请确保硬盘有足够的空间。

1.1.3使用Boost Boost库大部分组件(近90%)不需要编译，直接包含头文件即可。

例如，如果要使用boost::tribool，只需要在C++源文件中添加如下include语句即可(当然，接下来的代码可能还需要using namespace boost;)：#include //使用tribool库细心的读者会发现，Boost库的头文件与我们平常所用的头文件(\*.h)或c++标准库头文件(没有后缀名)不同，这正是Boost的独特之处。

它把C++类的声明和实现都放在了一个文件中，而不是分成两个文件，也就是“.h+.cpp”，故文件的后缀是.hpp。

之所以这么做，当然是有理由的。

首先就是与普通的C头文件(\*.h)区分，另一个很重要的原因就是使Boost库不需要预先编译，直接引入程序员的工程即可编译链接，方便了库的使用。

最后一个(无奈的)原因则是c++编译器的限制，许多编译器尚不支持c++标准提出的模板的分离编译模式(export关键字)，而Boost库大量使用了模板，为了保持与各个编译器的兼容，也不得不采用这种.hpp的头文件形式。

剩下的共十五个库(包括date\_time、regex、program\_options、test、thread、python等)必须编译成静态库或者动态库后才能使用。

不过有个好消息，其中有的库不需要编译也可以使用部分功能，而更好的消息是有的库已经有了不需要编译的替代品(xpressive可替代regex、signals2可替代signals)。

在Windows下的VC编译器支持自动链接技术，VC程序员可以不必为链接静态库或动态库、调试库或发行库等问题而费心了。

其他编译器就没有这样幸运，必须在命令行上手工指定链接库。

1.2关于STLport 本节将介绍C++标准库的一个高效实现——STLport，它是本书的默认标准库配置，用于配合Boost程序库工作。

1.2.1 什么是STLport STLport是一个完全符合C++98标准(及2003年修订)的一个免费的c++标准库实现。

它是由俄罗斯人Boris Fomitchev发起的开源项目，目的是基于著名的SGISTL开发一个可移植到各种平台上使用的高效的C++标准库。

STLport具有很多其他STL实现所没有的优点。

首先是高度的可移植性，可以配合市面上几乎所有的操作系统和编译器使用，使开发的程序能够在不同的编译平台上获得一致的标准库实现。

其次是性能表现优秀，其原始版本SGISTL就以高效而闻名，STLport在移植时也特别注重性能与效率，而且100%完全符合C++标准规范。

第三个优点是在标准之外增加了若干有用的扩展，如rope(增强的字符串类)、slist(单链表数据结构)、hash\_map(散列映射容器)，以及支持线程安全。

## <<Boost程序库完全开发指南>>

### 编辑推荐

《Boost程序库完全开发指南:深入C++"准"标准库(修订版)》内容丰富、结构严谨、详略得当、讲解透彻，带领读者领略了C++的最新前沿技术，相信会是每位C++程序员的必备工具书。

<<Boost程序库完全开发指南>>

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>