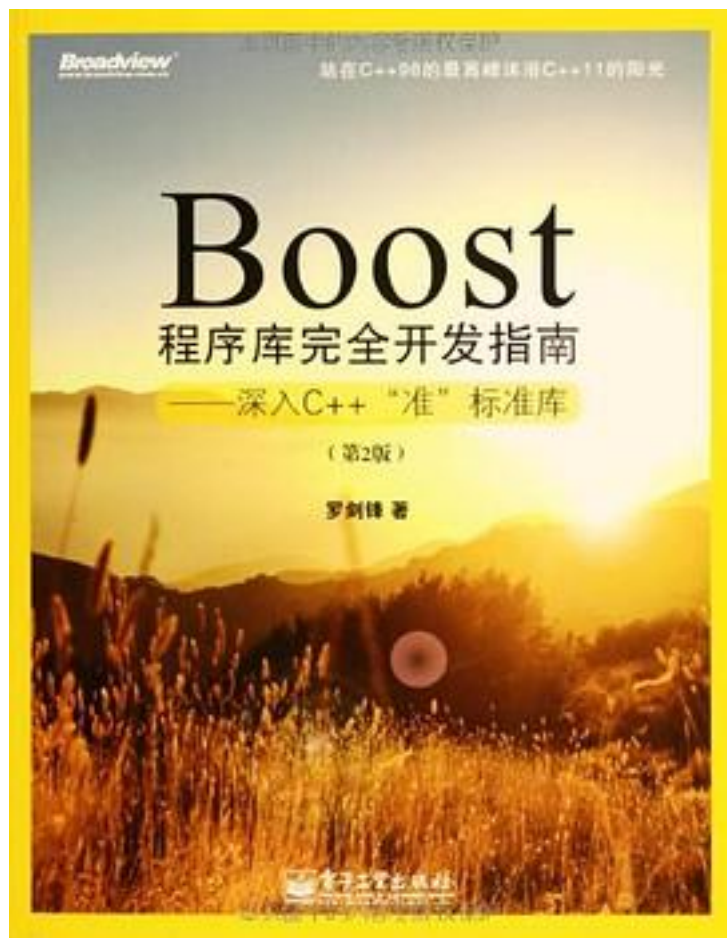


Boost程序库完全开发指南



[Boost程序库完全开发指南_下载链接1](#)

著者:罗剑锋

出版者:电子工业出版社

出版时间:2013-1-1

装帧:平装

isbn:9787121190896

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。

本书基于2012年8月发布的Boost1.51版，介绍了其中的所有117个库，并且结合C++11标准详细、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。

本书内容丰富、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。

作者介绍:

罗剑锋（网名Chrono），1996年就读东北某大学，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。

目录: 第0章 导读	1
0.1 关于本书	1
0.2 读者对象	1
0.3 本书的术语与风格	2
0.4 本书的结构	3
0.5 如何阅读本书	5
第1章 Boost程序库总论	7
1.1 关于Boost	7
1.1.1 什么是Boost	7
1.1.2 安装Boost	8
1.1.3 使用Boost	8
1.2 关于STLport	9
1.2.1 什么是STLport	9
1.2.2 安装STLport	10
1.2.3 编译STLport	10
1.2.4 使用STLport	10
1.3 开发环境简介	11
1.4 开发环境搭建	12
1.4.1 UNIX开发环境	12
1.4.2 Windows开发环境	13
1.4.3 高级议题	14
1.5 总结	16
第2章 时间与日期	17
2.1 timer库概述	17
2.2 timer	18
2.2.1 用法	18
2.2.2 类摘要	19
2.2.3 使用建议	20
2.3 progress_timer	20
2.3.1 用法	20
2.3.2 类摘要	21
2.3.3 扩展计时精度	22
2.4 progress_display	24

2.4.1 类摘要	24
2.4.2 用法	25
2.4.3 注意事项	26
2.5 date_time库概述	27
2.5.1 编译date_time库	28
2.5.2 date_time库的基本概念	29
2.6 处理日期	29
2.6.1 日期	30
2.6.2 创建日期对象	30
2.6.3 访问日期	32
2.6.4 日期的输出	33
2.6.5 与tm结构的转换	34
2.6.6 日期长度	34
2.6.7 日期运算	35
2.6.8 日期区间	37
2.6.9 日期区间运算	38
2.6.10 日期迭代器	40
2.6.11 其他功能	41
2.6.12 综合运用	41
2.7 处理时间	44
2.7.1 时间长度	44
2.7.2 操作时间长度	45
2.7.3 时间长度的精确度	47
2.7.4 时间点	48
2.7.5 创建时间点对象	49
2.7.6 操作时间点对象	50
2.7.7 与tm、time_t等结构的转换	51
2.7.8 时间区间	51
2.7.9 时间迭代器	52
2.7.10 综合运用	52
2.8 date_time库的高级议题	56
2.8.1 编译配置宏	56
2.8.2 格式化时间	56
2.8.3 本地时间	57
2.8.4 序列化	59
2.9 总结	59
第3章 内存管理	61
3.1 smart_ptr库概述	61
3.1.1 RAII机制	61
3.1.2 智能指针	62
3.2 scoped_ptr	63
3.2.1 类摘要	63
3.2.2 操作函数	64
3.2.3 用法	65
3.2.4 与auto_ptr的区别	66
3.2.5 与unique_ptr的区别	67
3.3 scoped_array	69
3.3.1 类摘要	69
3.3.2 用法	69
3.3.3 与unique_ptr的区别	70
3.3.4 使用建议	71
3.4 shared_ptr	72
3.4.1 类摘要	72

- 3.4.2 操作函数 73
- 3.4.3 用法 75
- 3.4.4 工厂函数 76
- 3.4.5 应用于标准容器 77
- 3.4.6 应用于桥接模式 79
- 3.4.7 应用于工厂模式 80
- 3.4.8 定制删除器 81
- 3.4.9 高级议题 83
- 3.5 shared_array 84
- 3.5.1 类摘要 84
- 3.5.2 用法 84
- 3.6 weak_ptr 85
- 3.6.1 类摘要 85
- 3.6.2 用法 86
- 3.6.3 获得this的shared_ptr 87
- 3.6.4 打破循环引用 88
- 3.7 intrusive_ptr 89
- 3.8 pool库概述 89
- 3.9 pool 90
- 3.9.1 类摘要 90
- 3.9.2 操作函数 91
- 3.9.3 用法 91
- 3.10 object_pool 92
- 3.10.1 类摘要 92
- 3.10.2 操作函数 93
- 3.10.3 用法 93
- 3.10.4 使用更多的构造参数 94
- 3.11 singleton_pool 95
- 3.11.1 类摘要 96
- 3.11.2 用法 96
- 3.12 pool_alloc 97
- 3.13 总结 98
- 第4章 实用工具 101
- 4.1 noncopyable 101
- 4.1.1 原理 102
- 4.1.2 用法 102
- 4.1.3 原理 103
- 4.2 typeof 104
- 4.2.1 动机 104
- 4.2.2 用法 106
- 4.2.3 向typeof库注册自定义类 107
- 4.2.4 使用建议 108
- 4.3 optional 108
- 4.3.1 “无意义”的值 108
- 4.3.2 类摘要 109
- 4.3.3 操作函数 109
- 4.3.4 用法 110
- 4.3.5 工厂函数 111
- 4.3.6 高级议题 112
- 4.4 assign 113
- 4.4.1 使用操作符+=向容器增加元素 113
- 4.4.2 使用操作符()向容器增加元素 114

- 4.4.3 初始化容器元素 115
- 4.4.4 减少重复输入 117
- 4.4.5 搭配非标准容器工作 118
- 4.4.6 高级用法 120
- 4.5 swap 121
 - 4.5.1 原理 121
 - 4.5.2 交换数组 122
 - 4.5.3 特化std::swap 122
 - 4.5.4 特化ADL可找到的swap 123
 - 4.5.5 使用建议 124
- 4.6 singleton 124
 - 4.6.1 boost.pool的单件实现 125
 - 4.6.2 boost.serialization的单件实现 127
- 4.7 tribool 129
 - 4.7.1 类摘要 129
 - 4.7.2 用法 130
 - 4.7.3 为第三态更名 131
 - 4.7.4 输入/输出 132
 - 4.7.5 与optional<bool>的区别 132
- 4.8 operators 133
 - 4.8.1 基本运算概念 134
 - 4.8.2 算术操作符的用法 135
 - 4.8.3 基类链 137
 - 4.8.4 复合运算概念 138
 - 4.8.5 相等与等价 140
 - 4.8.6 解引用操作符 141
 - 4.8.7 下标操作符 142
 - 4.8.8 高级议题 143
- 4.9 exception 144
 - 4.9.1 标准库中的异常 145
 - 4.9.2 类摘要 146
 - 4.9.3 向异常传递信息 147
 - 4.9.4 更进一步的用法 148
 - 4.9.5 包装标准异常 150
 - 4.9.6 使用函数抛出异常 151
 - 4.9.7 获得更多的调试信息 152
 - 4.9.8 高级议题 153
- 4.10 uuid 155
 - 4.10.1 类摘要 155
 - 4.10.2 用法 156
 - 4.10.3 生成器 158
 - 4.10.4 增强的uuid类 160
 - 4.10.5 与字符串的转换 161
 - 4.10.6 SHA1摘要算法 162
- 4.11 config 163
 - 4.11.1 BOOST_STRINGIZE 163
 - 4.11.2 BOOST_STATIC_CONSTANT 164
 - 4.11.3 其他工具 165
- 4.12 utility 165
 - 4.12.1 BOOST_BINARY 165
 - 4.12.2 BOOST_CURRENT_FUNCTION 166

- 4.13 总结 167
- 第5章 字符串与文本处理 171
 - 5.1 lexical_cast 171
 - 5.1.1 用法 172
 - 5.1.2 异常bad_lexical_cast 173
 - 5.1.3 对转换对象的要求 174
 - 5.1.4 应用于自己的类 174
 - 5.2 format 175
 - 5.2.1 简单的例子 176
 - 5.2.2 输入操作符% 177
 - 5.2.3 类摘要 179
 - 5.2.4 格式化语法 180
 - 5.2.5 format的性能 181
 - 5.2.6 高级用法 181
 - 5.3 string_algo 182
 - 5.3.1 简单的例子 183
 - 5.3.2 string_algo概述 184
 - 5.3.3 大小写转换 185
 - 5.3.4 判断式（算法） 185
 - 5.3.5 判断式（函数对象） 187
 - 5.3.6 分类 188
 - 5.3.7 修剪 189
 - 5.3.8 查找 190
 - 5.3.9 替换与删除 191
 - 5.3.10 分割 193
 - 5.3.11 合并 195
 - 5.3.12 查找（分割）迭代器 196
 - 5.4 tokenizer 197
 - 5.4.1 类摘要 197
 - 5.4.2 用法 198
 - 5.4.3 分词函数对象 199
 - 5.4.4 char_separator 199
 - 5.4.5 escaped_list_separator 201
 - 5.4.6 offset_separator 201
 - 5.4.7 tokenizer库的缺陷 202
 - 5.5 xpressive 204
 - 5.5.1 两种使用方式 204
 - 5.5.2 正则表达式语法简介 205
 - 5.5.3 类摘要 206
 - 5.5.4 匹配 208
 - 5.5.5 查找 211
 - 5.5.6 替换 212
 - 5.5.7 迭代 213
 - 5.5.8 分词 215
 - 5.5.9 与regex的区别 216
 - 5.5.10 高级议题 217
 - 5.6 总结 219
- 第6章 正确性与测试 221
 - 6.1 assert 221
 - 6.1.1 基本用法 221
 - 6.1.2 禁用断言 222
 - 6.1.3 扩展用法 223
 - 6.1.4 BOOST_ASSERT_MSG 224
 - 6.1.5 BOOST_VERIFY 225

- 6.2 static_assert 225
 - 6.2.1 定义 226
 - 6.2.2 用法 226
 - 6.2.3 使用建议 228
- 6.3 test 228
 - 6.3.1 编译test库 228
 - 6.3.2 最小化的测试套件 229
 - 6.3.3 单元测试框架简介 231
 - 6.3.4 测试断言 231
 - 6.3.5 测试用例与套件 232
 - 6.3.6 测试实例 234
 - 6.3.7 测试夹具 235
 - 6.3.8 测试日志 237
 - 6.3.9 运行参数 238
 - 6.3.10 函数执行监视器 239
 - 6.3.11 程序执行监视器 242
 - 6.3.12 高级议题 242
- 6.4 总结 245
- 第7章 容器与数据结构 247
 - 7.1 array 247
 - 7.1.1 类摘要 248
 - 7.1.2 操作函数 248
 - 7.1.3 用法 249
 - 7.1.4 能力限制 250
 - 7.1.5 初始化 251
 - 7.1.6 零长度的数组 251
 - 7.1.7 与C++11标准的区别 252
 - 7.1.8 实现ref_array 252
 - 7.1.9 ref_array的用法 254
 - 7.2 dynamic_bitset 254
 - 7.2.1 类摘要 255
 - 7.2.2 创建与赋值 256
 - 7.2.3 容器操作 257
 - 7.2.4 位运算与比较运算 258
 - 7.2.5 访问元素 259
 - 7.2.6 类型转换 260
 - 7.2.7 集合操作 261
 - 7.2.8 综合运用 261
 - 7.3 unordered 263
 - 7.3.1 散列集合简介 263
 - 7.3.2 散列集合的用法 265
 - 7.3.3 散列映射简介 267
 - 7.3.4 散列映射的用法 269
 - 7.3.5 高级议题 271
 - 7.4 bimap 272
 - 7.4.1 类摘要 273
 - 7.4.2 基本用法 273
 - 7.4.3 值的集合类型 275
 - 7.4.4 集合类型的用法 276
 - 7.4.5 使用标签类型 277
 - 7.4.6 使用assign库 279
 - 7.4.7 查找与替换 279
 - 7.4.8 投射 281
 - 7.4.9 高级议题 282

- 7.5 circular_buffer 283
 - 7.5.1 类摘要 283
 - 7.5.2 用法 284
 - 7.5.3 环形缓冲区 285
 - 7.5.4 空间优化型缓冲区 286
- 7.6 tuple 287
 - 7.6.1 最简单的tuple:pair 287
 - 7.6.2 类摘要 288
 - 7.6.3 创建与赋值 288
 - 7.6.4 访问元素 290
 - 7.6.5 比较操作 291
 - 7.6.6 输入输出 292
 - 7.6.7 连结变量 293
 - 7.6.8 应用于assign库 293
 - 7.6.9 应用于exception库 294
 - 7.6.10 内部结构 294
 - 7.6.11 使用访问者模式 295
 - 7.6.12 高级议题 297
- 7.7 any 299
 - 7.7.1 类摘要 299
 - 7.7.2 访问元素 300
 - 7.7.3 用法 301
 - 7.7.4 简化的操作函数 302
 - 7.7.5 保存指针 303
 - 7.7.6 输出 304
 - 7.7.7 应用于容器 306
- 7.8 variant 306
 - 7.8.1 类摘要 307
 - 7.8.2 访问元素 308
 - 7.8.3 用法 308
 - 7.8.4 访问器 309
 - 7.8.5 与any的区别 312
 - 7.8.6 高级议题 312
- 7.9 multi_array 314
 - 7.9.1 类摘要 314
 - 7.9.2 用法 316
 - 7.9.3 多维数组生成器 318
 - 7.9.4 改变形状和大小 319
 - 7.9.5 创建子视图 320
 - 7.9.6 适配普通数组 322
 - 7.9.7 高级议题 323
- 7.10 property_tree 326
 - 7.10.1 类摘要 327
 - 7.10.2 读取配置信息 328
 - 7.10.3 写入配置信息 330
 - 7.10.4 更多用法 331
 - 7.10.5 XML数据格式 332
 - 7.10.6 其他数据格式 333
 - 7.10.7 高级议题 335
 - 7.11 总结 336
- 第8章 算法 339
 - 8.1 foreach 339
 - 8.1.1 用法 340
 - 8.1.2 详细解说 341

- 8.1.3 更优雅的名字 342
- 8.1.4 支持的序列类型 343
- 8.1.5 一个小问题 344
- 8.2 minmax 345
 - 8.2.1 用法 345
 - 8.2.2 使用tuples::tie 346
- 8.3 minmax_element 347
 - 8.3.1 用法 347
 - 8.3.2 其他函数的用法 348
- 8.4 总结 349
- 第9章 数学与数字 351
 - 9.1 integer 351
 - 9.1.1 integer_traits 351
 - 9.1.2 标准整数类型 353
 - 9.1.3 整数类型模板类 355
 - 9.2 rational 358
 - 9.2.1 类摘要 358
 - 9.2.2 创建与赋值 359
 - 9.2.3 算术运算与比较运算 360
 - 9.2.4 类型转换 360
 - 9.2.5 输入输出 361
 - 9.2.6 分子与分母 361
 - 9.2.7 与数学函数配合工作 361
 - 9.2.8 异常 361
 - 9.2.9 rational的精度 362
 - 9.2.10 实现无限精度的整数类型 362
 - 9.2.11 最大公约数和最小公倍数 367
 - 9.3 crc 367
 - 9.3.1 类摘要 368
 - 9.3.2 预定义的实现类 368
 - 9.3.3 计算CRC 369
 - 9.3.4 CRC函数 370
 - 9.3.5 自定义CRC函数 371
 - 9.4 random 371
 - 9.4.1 伪随机数发生器 372
 - 9.4.2 伪随机数发生器的构造 373
 - 9.4.3 伪随机数发生器的拷贝 374
 - 9.4.4 随机数分布器 375
 - 9.4.5 随机数分布器类摘要 376
 - 9.4.6 随机数分布器用法 379
 - 9.4.7 变量发生器 379
 - 9.4.8 产生随机数据块 381
 - 9.4.9 真随机数发生器 382
 - 9.4.10 实现真随机数发生器 383
 - 9.5 总结 384
- 第10章 操作系统相关 387
 - 10.1 io_state_savers 387
 - 10.1.1 类摘要 388
 - 10.1.2 用法 388
 - 10.1.3 简化new_progress_timer 390
 - 10.2 system 390
 - 10.2.1 编译system库 391

- 10.2.2 错误值枚举 391
- 10.2.3 错误类别 392
- 10.2.4 错误代码 393
- 10.2.5 错误异常 395
- 10.3 cpu_timer 396
 - 10.3.1 编译cpu_timer库 396
 - 10.3.2 时间类型 397
 - 10.3.3 cpu_timer 398
 - 10.3.4 auto_cpu_timer 400
 - 10.3.5 定制输出格式 401
- 10.4 filesystem 402
 - 10.4.1 编译filesystem库 402
 - 10.4.2 类摘要 403
 - 10.4.3 路径表示 405
 - 10.4.4 可移植的文件名 406
 - 10.4.5 路径处理 407
 - 10.4.6 异常 409
 - 10.4.7 文件状态 410
 - 10.4.8 文件属性 412
 - 10.4.9 文件操作 413
 - 10.4.10 迭代目录 414
 - 10.4.11 实例1：实现查找文件功能 417
 - 10.4.12 实例2：实现模糊查找文件功能 418
 - 10.4.13 实例3：实现拷贝目录功能 420
 - 10.4.14 文件流操作 422
- 10.5 program_options 422
 - 10.5.1 编译program_options库 423
 - 10.5.2 概述 424
 - 10.5.3 选项值 426
 - 10.5.4 选项描述器 427
 - 10.5.5 选项描述器的用法 428
 - 10.5.6 分析器 430
 - 10.5.7 存储器 432
 - 10.5.8 使用位置选项值 432
 - 10.5.9 分析环境变量 434
 - 10.5.10 分组选项信息 435
 - 10.5.11 高级用法 437
- 10.6 总结 440
- 第11章 函数与回调 443
 - 11.1 result_of 443
 - 11.1.1 原理 444
 - 11.1.2 用法 444
 - 11.2 ref 446
 - 11.2.1 类摘要 447
 - 11.2.2 基本用法 447
 - 11.2.3 工厂函数 448
 - 11.2.4 操作包装 449
 - 11.2.5 综合应用 450
 - 11.2.6 为ref增加函数调用功能 451
 - 11.3 bind 453
 - 11.3.1 工作原理 453

- 11.3.2 绑定普通函数 454
- 11.3.3 绑定成员函数 455
- 11.3.4 绑定成员变量 457
- 11.3.5 绑定函数对象 457
- 11.3.6 使用ref库 458
- 11.3.7 高级议题 459
- 11.4 function 461
 - 11.4.1 类摘要 462
 - 11.4.2 function的声明 462
 - 11.4.3 操作函数 463
 - 11.4.4 比较操作 464
 - 11.4.5 用法 464
 - 11.4.6 使用ref库 465
 - 11.4.7 用于回调 467
 - 11.4.8 与typeof的区别 469
- 11.5 signals2 469
 - 11.5.1 类摘要 470
 - 11.5.2 操作函数 471
 - 11.5.3 插槽的连接与调用 472
 - 11.5.4 信号的返回值 474
 - 11.5.5 合并器 474
 - 11.5.6 管理信号的连接 476
 - 11.5.7 更灵活的管理信号连接 477
 - 11.5.8 自动连接管理 480
 - 11.5.9 应用于观察者模式 482
 - 11.5.10 高级议题 485
- 11.6 总结 489
- 第12章 并发编程 491
 - 12.1 thread 491
 - 12.1.1 编译thread库 492
 - 12.1.2 时间功能 493
 - 12.1.3 互斥量 493
 - 12.1.4 线程对象 496
 - 12.1.5 创建线程 497
 - 12.1.6 操作线程 499
 - 12.1.7 中断线程 500
 - 12.1.8 线程组 504
 - 12.1.9 条件变量 505
 - 12.1.10 共享互斥量 508
 - 12.1.11 future 510
 - 12.1.12 高级议题 513
 - 12.2 asio 518
 - 12.2.1 概述 519
 - 12.2.2 定时器 520
 - 12.2.3 定时器用法 521
 - 12.2.4 网络通信简述 524
 - 12.2.5 IP地址和端点 525
 - 12.2.6 同步socket处理 526
 - 12.2.7 异步socket处理 528
 - 12.2.8 查询网络地址 532
 - 12.2.9 高级议题 533
 - 12.3 总结 537
- 第13章 编程语言支持 539
 - 13.1 python库概述 539

13.1.1 Python语言简介	540
13.1.2 安装Python环境	541
13.1.3 编译python库	541
13.1.4 使用python库	542
13.2 嵌入Python	543
13.2.1 初始化解释器	543
13.2.2 封装Python对象	544
13.2.3 执行Python语句	546
13.2.4 异常处理	547
13.3 扩展Python	548
13.3.1 最简单的例子	549
13.3.2 导出函数	551
13.3.3 导出重载函数	552
13.3.4 导出类	554
13.3.5 导出类的更多细节	556
13.3.6 高级议题	558
13.4 总结	560
第14章 其他Boost组件	563
14.1 算法	563
14.2 字符串和文本处理	564
14.3 容器与数据结构	565
14.4 迭代器	566
14.5 函数对象与高级编程	566
14.6 泛型编程	568
14.7 模板元编程	569
14.8 预处理元编程	569
14.9 并发编程	570
14.10 数学与数字	570
14.11 TR1实现	571
14.12 输入输出	571
14.13 杂项	572
14.14 总结	574
第15章 Boost与设计模式	575
15.1 创建型模式	575
15.2 结构型模式	577
15.3 行为模式	580
15.4 其他模式	583
15.5 总结	584
第16章 结束语	587
16.1 未臻完美的Boost	587
16.2 让Boost工作得更好	588
16.3 工夫在诗外	590
附录A 推荐书目	593
附录B C++标准简述	595
附录C STL简述	597
• • • • •	(收起)

[Boost程序库完全开发指南_下载链接1_](#)

标签

C++

boost

C/C++

编程语言

计算机

编程

程序设计

STL

评论

真无耻，抄袭boost文档翻译而且自己都没搞懂，很多地方含混着就过去了。真是浪费纸张。而且这么脑残的书居然出到了第三版，出版社的人怎么会接这种烂东西？豆瓣评分7.4分，你们确定看了？

挑了些有兴趣的读了读，工具文档。

很多内容都已经包含在C++11/14里了，boost库中我想了解的mpl、协程、asio这些东西要不就是没提，要不就是讲的很简略，还不如看文档

boost概要和简介

虽然讲有些复杂的库没详细讲，但用来了解boost也差不多了，建议和C++11一起学习

收获不少

Boost不错的入门读物，介绍了Boost的简单应用，但是Boost.Asio这方面讲的内容过少。可以结合这本书和Boost官网进行学习。

比较浅，就是大概介绍一下，可以翻翻了解下boost的内容

先粗略浏览一遍，需要的时候再细看，是比较有效率的阅读这本书的方法。

相比书的质量，价格严重偏高，大概是几年来买的最亏的一本。

上手使用boost快

boost的api介绍吧。可以当一个cookbook来读。但是不要期望还有其他的東西可以学到。

工具书

这本书类似于介绍性质，想学深层的东西，还是翻官方文档吧，作者写的太浅。

书评

外面的4篇书评，清一色的五毛，拜托下次做的专业一点，不要那么明显。。。说明明显是因为这几个用户都满足以下条件： 1. 只读过这一本书 2. 只写过这一篇书评 3. 书评内容短而无意义 4. 没有上传头像 5. 没有好友 PS: 豆瓣写书评就必须得给星星，但这本书我没看过，没资格评价...

赖勇浩 (<http://laiyonghao.com>) 作为一个时不时要用一点 C++ 的程序员，我常常自嘲为斯德哥尔摩综合症患者，用 Python 写着懒散的代码时，会怀念以前编写 C++ 代码的那种被虐感。但当真正要写一些 C++ 代码的时候，又会怀念 Python 带给我的自在，这也许就是所谓的由奢入俭...

入门还行，如果你要做应用级开发,还是看文档吧.
里边的东西用法太简单了，好多也都是大概介绍下，复杂的应用下根本不够用的.
还是得看文档和源码，比如asio这一章....被坑的不浅...
计时器用法介绍太粗略了,谁知到每次计完数还要重新设置超时数值。
io_service介绍太简略了，wo...

这书吧，看看，对boost大概有个了解就行了。。。要真正地在项目中使用boost时不建议参考该书。该书中有些地方说得不明不白很容易让人误解，有的就完全搞错了，比如pool一章；有的该说的没有说看了也无法正确使用。而且，书的章节结构也不太好(我希望能是侯捷翻的那本《C++标准...

在读此书前，我对boost的了解几乎为0，因为想系统的学习一下boost，所以买了此书。看完此书，总的来说对boost的大多数组件有了入门级的了解，本书对一些常用的组件比如date_time, bind/function, smart_ptr等都有介绍，也配有一些简单的实例，适合boost的初学者，但是对于有一...

以下读后感只针对[《Boost 程序库完全开发指南》第一版] (2010.9) 本书目前 (2018.8) 已被作者更新到了第四版。我没有仔细读过后续的版本，只是从简介中了解到作者精简了部分不常用功能的篇幅。对于 Boost User 来说，除了查阅 [Boost Library Documentation]，将本书作为使用 ...

勘误： P71 shared_ptr 应为：获得指向类型 Y 的指针 p 的管理权 P430 下画线
应为：下划线 P440 P496 const asio::error_code 应为： const system::error_code static
address (const char* str); 应为： static address from_string (const char* str); P505
shar...

只看了书上的thread,date_time等有数的几个章节,体验如下: 优点: 1.
目录很全,罗列出来了boost中实现的几乎所有的跨平台的组件; 2.
初碰boost时,可以大大缩短安装,编译,编写示例代码的时间. 缺点:
1.浏览目录的次数比浏览内容的次数还要多,一般是根据目录再翻过头去看boost...

应该说，这个书对于初学者很不错的。
当然了，看这个书只是对boost有个大概的了解。
实际应用时，还是多看看开发文档吧。对于这个库有个宏观的把握。
里面的一些东西用起来还是很方便的。
作者说在windwos下装boost挺麻烦，但我在ubuntu上 一条命令就搞定了...

我是boost初学者，看了这本书觉得帮助很大，至于有同学由于那个编译脚本的错误就
指责这本书觉得有点过，这本书在国人写的技术书里算很严谨了。
内容可以看出来是作者经验的总结，尤其喜欢全书最后一句话：
“生活中不只有C++，代码和编程，还有更多的东西值得我们去体味，朋友...

作为一位c++开发人员，在拿到本书之前听名字我也像大家一样感觉很Nx，可是自从到
手了，花了一个小时翻了一遍，心里有种失落的感觉，就把它丢在了废书堆里了。自己
安装个boost，目录下的帮助文档比它强多了。本人认为，用上boost的程序员应该说c
++应该基础不错，基础不好的不要...

刚刚接触boost，发现这是一本很好的工具书籍，讲的很好很实用，例子也比较典型和
代表性，正在学习中，并且已经体验到利用boost开发带来的益处，很不错哦

我是本书的编辑，正与作者商量写本关于Boost高级应用相关的，希望各位Boost大侠能够多多给出宝贵意见，不胜感谢。

之前完全没搞过c++，在学了段时间的基本语法后就研究这个了，感觉还是蛮强大，用来辅助服务端程序开发真的是很不错，性能也很好，代码比较优雅，推荐一读哦，虽然我现在基本不写c++了，但对于想要了解的人来说此书还是不错的，（我现在主要用python了）

Boost是C++中很强大的一个库，但是有时候在写代码时候比较困惑不知如何利用这样强大的功能，目前我在读这本书，觉得是一本很好的工具书。希望更多的C++爱好者能够像我一样从中吸收到自己想要的知识！

一直以来都被boost的强大所吸引，但是国内的相关书籍真是少之又少，前两天无意中发现了这本书，让我更加认识到boost的强大力量。原来我之前所接触到的才是冰山一角，作者的底子很深，深入详细的介绍了boost库，很高兴能发现这本书，感谢作者！

一直以来，boost的资料都是网上查找，比较麻烦，最近买了一本，感觉不错，可以当成一本随手的手册，比较方便，推荐大家可以读一下。

[Boost程序库完全开发指南_下载链接1](#)