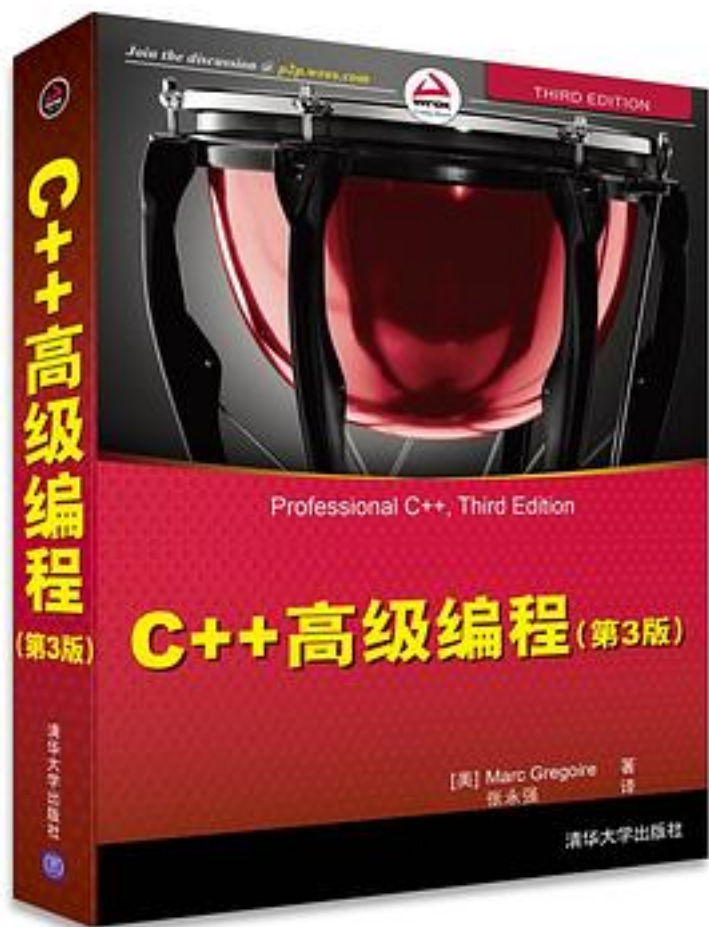


C++高级编程(第3版)



[C++高级编程\(第3版\)_下载链接1](#)

著者:(美) 葛瑞格尔 (Gregoire, M.) 著

出版者:清华大学出版社

出版时间:2015-5-1

装帧:平装

isbn:9787302396970

熟练驾驭C++语言的最新版本C++14

《C++高级编程(第3版)》

是设计和构建C++应用程序的实用指南，代码丰富，并根据C++14版本做了全面更新。本书强调良好编程风格的重要性，阐述如何设计可充分利用C++语言功能的高效解决方案；深入探讨C++语言功能集的更复杂元素，并披露避开常见陷阱的技巧。本书醒目显示了新的C++14信息，使你快速了解影响应用程序构建的显著变化。

主要内容

- ◆ 提供富有挑战的、紧贴实用的、可供下载的示例代码段供读者学习
- ◆ 研究详尽案例，案例中包含已在Windows和Linux上测试的丰富工作代码
- ◆ 列出保持良好编程风格的提示、技巧和方案，包括最佳调试实践
- ◆ 使用C++14的最新语言功能，包括函数返回类型推断、二进制字面量、泛型lambda和lambda捕捉表达式
- ◆ 使用最新标准库功能，例如make_unique、透明函数对象，通过类型寻址的元组、共享mutex和锁类

作者介绍:

Marc

Gregoire是一名在Windows和Linux平台上开发C/C++应用程序的经验丰富的软件工程师和开发人员。Marc是一位微软Visual C++ MVP，是比利时C++用户组的创始人，现供职于Nikon Metrology，负责开发3D激光扫描软件。Marc曾在Siemens和Nokia Siemens Networks开发关键2G和3G电信软件，他还在www.nuonsoft.com/blog/上维护了一个博客。

目录: 目录

第I部分 专业的C++简介

第1章 C++和STL速成 3

1.1 C++基础知识 3

1.1.1 小程序“hello world” 4

1.1.2 名称空间 6

1.1.3 变量 8

1.1.4 字面量 9

1.1.5 运算符 9

1.1.6 类型 11

1.1.7 条件 13

1.1.8 数组 16

1.1.9 循环 18

1.1.10 函数 19

1.1.11 类型推断(上) 21

1.1.12 这些都是基础 21

1.2 深入研究C++ 21

1.2.1 指针和动态内存 22

1.2.2 引用 26

1.2.3 C++中的字符串 26

1.2.4 异常 27

- 1.2.5 const的多种用法 28
- 1.2.6 类型推断(下) 29
- 1.3 作为面向对象语言的C++ 30
- 1.4 标准库 32
- 1.5 第一个有用的C++程序 33
 - 1.5.1 雇员记录系统 33
 - 1.5.2 Employee类 33
 - 1.5.3 Database类 36
 - 1.5.4 用户界面 39
 - 1.5.5 评估程序 41
- 1.6 本章小结 41
- 第2章 使用字符串 43
 - 2.1 动态字符串 43
 - 2.1.1 C风格的字符串 43
 - 2.1.2 字符串字面量 45
 - 2.1.3 C++ string类 46
 - 2.1.4 原始字符串字面量 49
 - 2.1.5 非标准字符串 50
 - 2.2 本章小结 50
- 第3章 编码风格 51
 - 3.1 良好外观的重要性 51
 - 3.1.1 事先考虑 51
 - 3.1.2 良好风格的元素 52
 - 3.2 为代码编写文档 52
 - 3.2.1 使用注释的原因 52
 - 3.2.2 注释的风格 55
 - 3.2.3 本书的注释 59
 - 3.3 分解 59
 - 3.3.1 通过重构分解 59
 - 3.3.2 通过设计分解 60
 - 3.3.3 本书中的分解 60
 - 3.4 命名 60
 - 3.4.1 选择恰当的名称 60
 - 3.4.2 命名约定 61
 - 3.5 使用具有风格的语言特性 63
 - 3.5.1 使用常量 63
 - 3.5.2 使用引用代替指针 63
 - 3.5.3 使用自定义异常 64
 - 3.6 格式 64
 - 3.6.1 关于大括号对齐的争论 64
 - 3.6.2 关于空格和圆括号的争论 65
 - 3.6.3 空格和制表符 66
 - 3.7 风格的挑战 66
 - 3.8 本章小结 66
- 第II部分 专业的C++软件设计
- 第4章 设计专业的C++程序 69
 - 4.1 程序设计概述 69
 - 4.2 程序设计的重要性 70
 - 4.3 C++设计的特点 72
 - 4.4 C++设计的两个原则 73
 - 4.4.1 抽象 73
 - 4.4.2 重用 74
 - 4.5 重用代码 75
 - 4.5.1 关于术语的说明 76

- 4.5.2 决定是否重用代码 76
- 4.5.3 重用代码的策略 78
- 4.5.4 绑定第三方应用程序 82
- 4.5.5 开放源代码库 82
- 4.5.6 C++标准库 83
- 4.6 设计模式和技巧 84
- 4.7 设计一个国际象棋程序 84
 - 4.7.1 需求 84
 - 4.7.2 设计步骤 85
- 4.8 本章小结 88
- 第5章 面向对象设计 91
 - 5.1 过程化的思考方式 91
 - 5.2 面向对象思想 92
 - 5.2.1 类 92
 - 5.2.2 组件 92
 - 5.2.3 属性 93
 - 5.2.4 行为 93
 - 5.2.5 综合考虑 93
 - 5.3 生活在对象世界里 94
 - 5.5.1 过度使用对象 94
 - 5.5.2 过于通用的对象 95
 - 5.4 对象之间的关系 96
 - 5.4.1 “有一个”关系 96
 - 5.4.2 “是一个”关系(继承) 97
 - 5.4.3 “有一个”与“是一个”的区别 98
 - 5.4.4 Not-a关系 101
 - 5.4.5 层次结构 101
 - 5.4.6 多重继承 102
 - 5.4.7 混入类 103
 - 5.5 抽象 104
 - 5.5.1 接口与实现 104
 - 5.5.2 决定公开的接口 104
 - 5.5.3 设计成功的抽象 106
 - 5.6 本章小结 106
- 第6章 设计可重用代码 107
 - 6.1 重用哲学 107
 - 6.2 如何设计可重用的代码 108
 - 6.2.1 使用抽象 108
 - 6.2.2 构建理想的重用代码 109
 - 6.2.3 设计有用的接口 113
 - 6.2.4 协调通用性和使用性 116
 - 6.3 本章小结 117
- 第III部分 专业的C++编码方法
- 第7章 熟悉类和对象 121
 - 7.1 电子表格示例介绍 121
 - 7.2 编写类 122
 - 7.2.1 类定义 122
 - 7.2.2 定义方法 124
 - 7.2.3 使用对象 127
 - 7.3 对象的生命周期 129
 - 7.3.1 创建对象 129
 - 7.3.2 销毁对象 143
 - 7.3.3 对象赋值 144
 - 7.3.4 复制和赋值的区别 147

- 7.4 本章小结 148
- 第8章 掌握类与对象 149
 - 8.1 对象的动态内存分配 149
 - 8.1.1 Spreadsheet类 149
 - 8.1.2 使用析构函数释放内存 151
 - 8.1.3 处理复制和赋值 152
 - 8.2 不同的数据成员类型 158
 - 8.2.1 静态数据成员 158
 - 8.2.2 常量数据成员 159
 - 8.2.3 引用数据成员 160
 - 8.2.4 常量引用数据成员 161
 - 8.3 与方法有关的更多内容 162
 - 8.3.1 静态方法 162
 - 8.3.2 const方法 162
 - 8.3.3 方法重载 164
 - 8.3.4 默认参数 165
 - 8.3.5 内联方法 166
 - 8.4 嵌套类 167
 - 8.5 类内的枚举类型 168
 - 8.6 友元 169
 - 8.7 运算符重载 170
 - 8.7.1 示例：为SpreadsheetCell实现加法 170
 - 8.7.2 重载算术运算符 174
 - 8.7.3 重载比较运算符 176
 - 8.7.4 创建具有运算符重载的类型 178
 - 8.8 创建稳定的接口 178
 - 8.9 本章小结 181
- 第9章 揭秘继承技术 183
 - 9.1 使用继承构建类 183
 - 9.1.1 扩展类 184
 - 9.1.2 重写方法 187
 - 9.2 使用继承重用代码 190
 - 9.2.1 WeatherPrediction类 190
 - 9.2.2 在派生类中添加功能 191
 - 9.2.3 在派生类中替换功能 192
 - 9.3 利用父类 193
 - 9.3.1 父类构造函数 193
 - 9.3.2 父类的析构函数 194
 - 9.3.3 使用父类方法 196
 - 9.3.4 向上转型和向下转型 198
 - 9.4 继承与多态性 199
 - 9.4.1 回到电子表格 199
 - 9.4.2 设计多态性的电子表格单元格 200
 - 9.4.3 电子表格单元格的基类 200
 - 9.4.4 独立的派生类 202
 - 9.4.5 利用多态性 204
 - 9.4.6 考虑将来 205
 - 9.5 多重继承 206
 - 9.5.1 从多个类继承 206
 - 9.5.2 名称冲突和歧义基类 207
 - 9.6 有趣而晦涩的继承问题 210
 - 9.6.1 修改重写方法的特征 210
 - 9.6.2 继承的构造函数 214
 - 9.6.3 重写方法时的特殊情况 217

- 9.6.4 派生类中的复制构造函数和赋值运算符 223
- 9.6.5 virtual的真相 224
- 9.6.6 运行时类型工具 227
- 9.6.7 非public继承 228
- 9.6.8 虚基类 228
- 9.7 本章小结 229
- 第10章 理解灵活而奇特的C++ 231
 - 10.1 引用 231
 - 10.1.1 引用变量 232
 - 10.1.2 引用数据成员 233
 - 10.1.3 引用参数 234
 - 10.1.4 引用作为返回值 235
 - 10.1.5 使用引用还是指针 235
 - 10.1.6 右值引用 238
 - 10.2 关键字的疑问 242
 - 10.2.1 const关键字 243
 - 10.2.2 static关键字 246
 - 10.2.3 非局部变量的初始化顺序 249
 - 10.2.4 非局部变量的销毁顺序 249
 - 10.3 类型和类型转换 250
 - 10.3.1 typedef 250
 - 10.3.2 函数指针typedef 251
 - 10.3.3 类型别名 251
 - 10.3.4 类型转换 252
 - 10.4 作用域解析 256
 - 10.5 C++11/C++14 257
 - 10.5.1 统一初始化 257
 - 10.5.2 初始化列表 258
 - 10.5.3 显式转换运算符 259
 - 10.5.4 特性 260
 - 10.5.5 用户定义的字面量 260
 - 10.6 头文件 262
 - 10.7 C的实用工具 263
 - 10.7.1 变长参数列表 263
 - 10.7.2 预处理器宏 265
 - 10.8 本章小结 266
- 第11章 利用模板编写泛型代码 267
 - 11.1 模板概述 268
 - 11.2 类模板 268
 - 11.2.1 编写类模板 268
 - 11.2.2 尖括号 275
 - 11.2.3 编译器处理模板的原理 275
 - 11.2.4 将模板代码分布在多个文件中 276
 - 11.2.5 模板参数 278
 - 11.2.6 方法模板 280
 - 11.2.7 模板类特例化 284
 - 11.2.8 从类模板派生 286
 - 11.2.9 继承还是特例化 287
 - 11.2.10 模板别名 287
 - 11.2.11 替换函数语法 288
 - 11.3 函数模板 289
 - 11.3.1 函数模板特例化 290
 - 11.3.2 函数模板重载 291
 - 11.3.3 类模板的friend函数模板 292

- 11.4 可变模板 293
- 11.5 本章小结 293
- 第12章 C++ I/O揭秘 295
 - 12.1 使用流 295
 - 12.1.1 流的含义 296
 - 12.1.2 流的来源和目标 296
 - 12.1.3 流式输出 297
 - 12.1.4 流式输入 301
 - 12.1.5 对象的输入输出 306
 - 12.2 字符串流 308
 - 12.3 文件流 309
 - 12.3.1 通过seek()和tell()在文件中转移 310
 - 12.3.2 将流连接在一起 312
 - 12.4 双向I/O 312
 - 12.5 本章小结 314
- 第13章 错误处理 315
 - 13.1 错误与异常 315
 - 13.1.1 异常的含义 316
 - 13.1.2 C++中异常的优点 316
 - 13.1.3 C++中异常的缺点 317
 - 13.1.4 我们的建议 317
 - 13.2 异常机制 317
 - 13.2.1 抛出并捕获异常 318
 - 13.2.2 异常类型 321
 - 13.2.3 抛出并捕获多个异常 322
 - 13.2.4 未捕获的异常 325
 - 13.2.5 抛出列表 326
 - 13.3 异常与多态性 330
 - 13.3.1 标准异常体系 330
 - 13.3.2 在类层次结构中捕获异常 332
 - 13.3.3 编写自己的异常类 333
 - 13.3.4 嵌套异常 335
 - 13.4 堆栈的释放与清理 337
 - 13.4.1 使用智能指针 338
 - 13.4.2 捕获、清理并重新抛出 339
 - 13.5 常见的错误处理问题 339
 - 13.5.1 内存分配错误 339
 - 13.5.2 构造函数中的错误 342
 - 13.5.3 构造函数的function-try-blocks 343
 - 13.5.4 析构函数中的错误 345
 - 13.6 综合应用 346
 - 13.7 本章小结 350
- 第14章 C++运算符重载 351
 - 14.1 运算符重载概述 351
 - 14.1.1 重载运算符的原因 352
 - 14.1.2 运算符重载的限制 352
 - 14.1.3 运算符重载的选择 352
 - 14.1.4 不要重载的运算符 354
 - 14.1.5 可重载运算符小结 354
 - 14.1.6 右值引用 357
 - 14.1.7 关系运算符 358
 - 14.2 重载算术运算符 358
 - 14.2.1 重载一元负号和一元正号 358

- 14.2.2 重载递增和递减运算符 359
- 14.3 重载按位运算符和二元逻辑运算符 360
- 14.4 重载插入运算符和提取运算符 360
- 14.5 重载下标运算符 362
 - 14.5.1 通过operator[]提供只读访问 364
 - 14.5.2 非整数数组索引 365
- 14.6 重载函数调用运算符 366
- 14.7 重载解除引用运算符 367
 - 14.7.1 实现operator* 368
 - 14.7.2 实现operator-> 369
 - 14.7.3 operator->*的含义 369
- 14.8 编写转换运算符 370
 - 14.8.1 转换运算符的多义性问题 371
 - 14.8.2 用于布尔表达式的转换 372
- 14.9 重载内存分配和释放运算符 373
 - 14.9.1 new和delete的工作原理 374
 - 14.9.2 重载operator new和operator delete 375
 - 14.9.3 显式地删除/默认化operator new和operator delete 377
 - 14.9.4 重载带有额外参数的operator new和operator delete 377
- 14.10 本章小结 379
- 第15章 C++标准库概述 381
 - 15.1 编码原则 382
 - 15.1.1 使用模板 382
 - 15.1.2 使用运算符重载 382
 - 15.2 C++标准库概述 382
 - 15.2.1 字符串 382
 - 15.2.2 正则表达式 382
 - 15.2.3 I/O流 383
 - 15.2.4 智能指针 383
 - 15.2.5 异常 383
 - 15.2.6 数学工具 383
 - 15.2.7 时间工具 384
 - 15.2.8 随机数 384
 - 15.2.9 初始化列表 384
 - 15.2.10 Pair 和Tuple 384
 - 15.2.11 函数对象 384
 - 15.2.12 多线程 384
 - 15.2.13 类型特质 385
 - 15.2.14 标准模板库 385
 - 15.3 本章小结 397
- 第16章 理解容器与迭代器 399
 - 16.1 容器概述 399
 - 16.1.1 对元素的要求 400
 - 16.1.2 异常和错误检查 401
 - 16.1.3 迭代器 401
 - 16.2 顺序容器 404
 - 16.2.1 vector 404
 - 16.2.2 vector<bool>特化 420
 - 16.2.3 deque 420
 - 16.2.4 list 421
 - 16.2.5 forward_list 424
 - 16.2.6 array 426
 - 16.3 容器适配器 427
 - 16.3.1 queue 427

- 16.3.2 priority_queue 429
- 16.3.3 stack 432
- 16.4 关联容器 432
 - 16.4.1 pair工具类 432
 - 16.4.2 map 433
 - 16.4.3 multimap 439
 - 16.4.4 set 442
 - 16.4.5 multiset 444
- 16.5 无序关联容器/哈希表 444
 - 16.5.1 哈希函数 444
 - 16.5.2 unordered_map 446
 - 16.5.3 unordered_multimap 449
 - 16.5.4 unordered_set/unordered_multiset 449
- 16.6 其他容器 449
 - 16.6.1 标准C风格数组 449
 - 16.6.2 string 450
 - 16.6.3 流 451
 - 16.6.4 bitset 451
- 16.7 本章小结 455
- 第17章 掌握STL算法 457
 - 17.1 算法概述 457
 - 17.1.1 find和find_if算法 458
 - 17.1.2 accumulate算法 460
 - 17.1.3 在算法中使用移动语义 461
 - 17.2 lambda表达式 461
 - 17.2.1 语法 462
 - 17.2.2 泛型Lambda表达式 464
 - 17.2.3 Lambda捕捉表达式 464
 - 17.2.4 将Lambda表达式用作返回值 465
 - 17.2.5 将Lambda表达式用作参数 466
 - 17.2.6 STL算法示例 466
 - 17.3 函数对象 467
 - 17.3.1 算术函数对象 468
 - 17.3.2 透明运算符仿函数 468
 - 17.3.3 比较函数对象 469
 - 17.3.4 逻辑函数对象 470
 - 17.3.5 按位函数对象 470
 - 17.3.6 函数对象适配器 470
 - 17.3.7 编写自己的函数对象 474
 - 17.4 算法详解 475
 - 17.4.1 迭代器 475
 - 17.4.2 非修改序列算法 476
 - 17.4.3 修改序列算法 480
 - 17.4.4 操作算法 486
 - 17.4.5 分区算法 487
 - 17.4.6 排序算法 488
 - 17.4.7 二叉树搜索算法 489
 - 17.4.8 集合算法 489
 - 17.4.9 最大/最小算法 491
 - 17.4.10 数值处理算法 492
 - 17.5 算法示例：审核选民登记 493
 - 17.5.1 选民登记审核问题描述 493
 - 17.5.2 auditVoterRolls函数 493
 - 17.5.3 getDuplicates函数 494

- 17.5.4 测试auditVoterRolls函数 495
- 17.6 本章小结 496
- 第18章 字符串本地化与正则表达式 497
 - 18.1 本地化 497
 - 18.1.1 本地化字符串字面量 497
 - 18.1.2 宽字符 498
 - 18.1.3 非西方字符集 498
 - 18.1.4 locale和facet 500
 - 18.2 正则表达式 502
 - 18.2.1 ECMAScript语法 503
 - 18.2.2 regex库 507
 - 18.2.3 regex_match() 508
 - 18.2.4 regex_search() 510
 - 18.2.5 regex_iterator 512
 - 18.2.6 regex_token_iterator 513
 - 18.2.7 regex_replace() 515
 - 18.3 本章小结 517
- 第19章 其他库工具 519
 - 19.1 std::function 519
 - 19.2 有理数 521
 - 19.3 Chrono库 523
 - 19.3.1 持续时间 523
 - 19.3.2 时钟 526
 - 19.3.3 时点 528
 - 19.4 生成随机数 529
 - 19.4.1 随机数引擎 530
 - 19.4.2 随机数引擎适配器 531
 - 19.4.3 预定义的引擎和引擎适配器 532
 - 19.4.4 生成随机数 532
 - 19.4.5 随机数分布 533
 - 19.5 元组 536
 - 19.6 本章小结 539
- 第IV部分 掌握C++的高级特性
- 第20章 自定义和扩展STL 543
 - 20.1 分配器 543
 - 20.2 迭代器适配器 544
 - 20.2.1 反向迭代器 544
 - 20.2.2 流迭代器 545
 - 20.2.3 插入迭代器 546
 - 20.2.4 移动迭代器 547
 - 20.3 扩展STL 548
 - 20.3.1 扩展STL的原因 549
 - 20.3.2 编写STL算法 549
 - 20.3.3 编写STL容器 551
 - 20.4 本章小结 582
- 第21章 模板的高级特性 583
 - 21.1 深入了解模板参数 583
 - 21.1.1 深入了解模板类型参数 583
 - 21.1.2 模板参数模板介绍 586
 - 21.1.3 深入了解非类型模板参数 587
 - 21.2 模板类部分特例化 589
 - 21.3 通过重载模拟函数部分特例化 592
 - 21.4 模板递归 593
 - 21.4.1 N维网格：初次尝试 593

- 21.4.2 真正的N维网格 595
- 21.5 类型推导 597
- 21.6 可变参数模板 600
 - 21.6.1 类型安全的可变长度参数列表 600
 - 21.6.2 可变数目的混入类 602
- 21.7 元编程 603
 - 21.7.1 编译时阶乘 603
 - 21.7.2 循环展开 604
 - 21.7.3 打印元组 605
 - 21.7.4 类型trait 607
 - 21.7.5 结论 612
- 21.8 本章小结 612
- 第22章 内存管理 613
 - 22.1 使用动态内存 614
 - 22.1.1 如何描绘内存 614
 - 22.1.2 分配和释放 615
 - 22.1.3 数组 616
 - 22.1.4 使用指针 622
 - 22.2 数组-指针的对偶性 624
 - 22.2.1 数组就是指针 624
 - 22.2.2 并非所有的指针都是数组 626
 - 22.3 低级内存操作 626
 - 22.3.1 指针运算 626
 - 22.3.2 自定义内存管理 627
 - 22.3.3 垃圾回收 627
 - 22.3.4 对象池 628
 - 22.3.5 函数指针 628
 - 22.3.6 方法和数据成员的指针 630
 - 22.4 智能指针 630
 - 22.4.1 旧的过时的auto_ptr 631
 - 22.4.2 shared_ptr和unique_ptr智能指针 631
 - 22.5 内存常见的陷阱 636
 - 22.5.1 分配不足的字符串 637
 - 22.5.2 访问内存越界 637
 - 22.5.3 内存泄漏 638
 - 22.5.4 双重删除和无效指针 640
 - 22.6 本章小结 641
- 第23章 C++多线程编程 643
 - 23.1 简介 643
 - 23.1.1 竞争条件 645
 - 23.1.2 死锁 646
 - 23.1.3 撕裂 647
 - 23.1.4 缓存的一致性 647
 - 23.2 线程 647
 - 23.2.1 通过函数指针创建线程 647
 - 23.2.2 通过函数对象创建线程 649
 - 23.2.3 通过lambda创建线程 650
 - 23.2.4 通过成员函数创建线程 651
 - 23.2.5 线程本地存储 651
 - 23.2.6 取消线程 652
 - 23.2.7 从线程获得结果 652
 - 23.2.8 复制和重新抛出异常 652
 - 23.3 原子操作库 654
 - 23.3.1 原子类型示例 655

- 23.3.2 原子操作 657
- 23.4 互斥 658
 - 23.4.1 互斥体类 658
 - 23.4.2 锁 660
 - 23.4.3 std::call_once 661
 - 23.4.4 互斥体的用法示例 663
- 23.5 条件变量 665
- 23.6 future 667
- 23.7 异常处理 669
- 23.8 示例：多线程日志记录器类 669
- 23.9 线程池 673
- 23.10 线程设计和最佳实践 674
- 23.11 本章小结 675

第V部分 C++软件工程

第24章 充分利用软件工程方法 679

- 24.1 过程的必要性 679
- 24.2 软件生命周期模型 680
 - 24.2.1 分段模型和瀑布模型 680
 - 24.2.2 螺旋模型 683
 - 24.2.3 Rational统一过程 685
- 24.3 软件工程方法学 686
 - 24.3.1 敏捷 686
 - 24.3.2 Scrum 686
 - 24.3.3 极限编程 688
 - 24.3.4 软件分流 691
- 24.4 构建自己的过程和方法 691
 - 24.4.1 对新思想采取开放态度 692
 - 24.4.2 提出新想法 692
 - 24.4.3 知道什么行得通什么行不通 692
 - 24.4.4 不要逃避 692
- 24.5 源代码控制 692
- 24.6 本章小结 694

第25章 编写高效的C++程序 695

- 25.1 性能和效率概述 695
 - 25.1.1 提升效率的两种方式 696
 - 25.1.2 两种程序 696
 - 25.1.3 C++是不是低效的语言 696
- 25.2 语言层次的效率 696
 - 25.2.1 高效地操纵对象 697
 - 25.2.2 使用内联方法和函数 700
- 25.3 设计层次的效率 700
 - 25.3.1 尽可能多地缓存 701
 - 25.3.2 使用对象池 701
- 25.4 剖析 705
 - 25.4.1 使用gprof的剖析范例 705
 - 25.4.2 使用Visual C++ 2013的剖析范例 713
- 25.5 本章小结 716

第26章 熟练掌握调试技术 717

- 26.1 调试的基本定律 717
- 26.2 bug分类学 718
- 26.3 避免bug 718
- 26.4 为bug做好规划 719
 - 26.4.1 错误日志 719
 - 26.4.2 调试跟踪 720

26.4.3 断言 727
26.4.4 静态断言 728
26.4.5 崩溃转储 729
26.5 调试技术 729
26.5.1 重现bug 729
26.5.2 调试可重复的bug 730
26.5.3 调试不可重现的bug 730
26.5.4 调试退化 731
26.5.5 调试内存问题 731
26.5.6 调试多线程程序 735
26.5.7 调试示例：文章引用 736
26.5.8 从ArticleCitations示例中总结的教训 746
26.6 本章小结 747
附录A C++面试 749
附录B 带注解的参考文献 767
附录C 标准库头文件 777
• • • • • ([收起](#))

[C++高级编程\(第3版\) 下载链接1](#)

标签

C++

计算机技术

C++11

编程

C++14

想读

C/C++

评论

正在看第三版，感觉言语比较直白易懂，值得多看几遍！

[C++高级编程\(第3版\) 下载链接1](#)

书评

[C++高级编程\(第3版\) 下载链接1](#)