

不要重复发明

快捷开发 信手拈来

学习本书内容需要您已经有了—定的C/C++基础

本书力求将C++ STL所涉及的问题，—列出并结合实例详细阐述，同时融入作者多年实践经验，目的只有一个：让初学者能够少走些弯路。

本书的示例程序皆是作者亲自编写或从MSDN中摘录，精练分析，简单易学，有助于读者掌握STL开发技能，同时希望读者在学习的时候不要漏掉任何一个例题。

轮子

C++ STL

闫常友 王敏 编著

标准程序库开发指南

[第2版]

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

不要重复发明

轮子

C++ STL

闫常友 王敏 编著

标准程序库开发指南

[第2版]

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书献给喜欢 C++ STL 的朋友，当您看见这本书时您应该对 C++ STL 已有一定的了解，希望继续深造；或者您已经暗下决心来学习它。那么本书将是您最佳的选择。

如果您已经有了一些 C/C++ 基础，那么学习起来会更加轻松。本书分为 14 章，按照章节的先后顺序，由浅入深地讲解 C++ STL 应用开发技术。本书力求将 STL 涉及的问题一一列出讲解，使初学者能够少走弯路，并且所有的示例程序都是作者亲自编写或从 MSDN 中摘录的，简单易学，有助于读者掌握 STL 的知识。

图书在版编目 (CIP) 数据

C++ STL 标准程序库开发指南 / 闫常友，王敏编著. —
2 版. —北京：中国铁道出版社，2017.1
ISBN 978-7-113-22377-9

I. ①C… II. ①闫… ②王… III. ①C 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字 (2016) 第 232920 号

书 名：C++ STL 标准程序库开发指南（第 2 版）
作 者：闫常友 王 敏 编著

责任编辑：荆 波
责任印制：赵星辰

读者服务热线：010-63560056
封面设计：**MX** DESIGN
STUDIO

出版发行：中国铁道出版社（北京市西城区右安门西街 8 号 邮政编码：100054）
印 刷：中国铁道出版社印刷厂
版 次：2013 年 5 月第 1 版 2017 年 1 月第 2 版 2017 年 1 月第 1 次印刷
开 本：787mm×1 092mm 1/16 印张：33.25 字数：892 千
书 号：ISBN 978-7-113-22377-9
定 价：79.00 元

版权所有 侵权必究

凡购买铁道版图书，如有印制质量问题，请与本社读者服务部联系调换。电话：(010) 51873174

打击盗版举报电话：(010) 51873659

在开发语言中，C++的应用越来越广泛。而 C++ STL 是标准的 C++模板库，是算法和其他一些标准组件的集合，可以说是世界上众多技术人员多年经验的总结。STL 的目的是标准化组件，这样就不用重复开发，即可使用现成的组件，提高了开发效率。STL 是 C++标准的一部分，开发时不用额外安装插件包。

编写一本全面的、透彻的 C++ STL 的书籍，有助于广大程序开发人员深入掌握 C++ STL 的编程技巧。从 2003 年 10 月 15 日，国际标准 ISO/IEC-14882 second edition 颁布以来，已经过去了十多个年头。2011 年夏天，笔者对目前的 C++类书籍做了调研。除国外的翻译版本之外，国内出版的书籍少之又少，图书的质量良莠不齐。限于国外专家的书写习惯和翻译人员的水平，多数翻译版本表达不够准确。国内出版的多数书籍不能深刻地体现 C++ STL 的博大精深。

国内的程序开发人员对 C++和 STL 的学习和掌握，也停留在较低的水平。当我们打开 C++或 STL 的函数声明或函数定义时，我们应该为 C++语言专家的深邃思想和逻辑思维而折服。

我们在学习使用 C++ STL 时，有时会深刻地认识到自身对计算机语言知识的匮乏，并且深刻地认识现代社会及未来社会中计算机语言的重要性。不管哪一种语言，其算法库的博大精深是毋庸置疑的。程序员在编程过程中只有充分利用这些算法库，才能发挥事半功倍的效果。当然每种算法库也有自己的局限性，遇到这种情况就需要程序员自己编写自定义的算法了。

这里不再赘述，也许您现在已对 C++ STL 产生了浓厚的兴趣。那么尝试读一读本书，相信您会爱上本书。

STL 的发展和本书改版

C++ STL 是学好算法之后的关键学习点，本书配套我社《C/C++常用算法手册》使用，可以在学习算法知识之后，来学习这套积累了程序员大牛智慧结晶的 STL，可以应用在关键的算法和工作应用中。

本次改版，我们主要把代码重新进行了调试，修改了上一版本书中因为匆忙而产生的代码小错误，并对全书所有实例的讲解意图进行了更加详细的说明。大规模重写了全书的第 1 章，让读者更加明白 C++模板技术的起源和用途，明白 C++ STL 的重要性。让读者的学习台阶不再陡峭，读者学习了 C++语言，结合 C/C++算法类图书，即可学习本书。

本书适合的读者

- 有一定的 C/C++ 语言基础，想从事 C++ 程序较高级开发的初学者；
- 学习过 C++ 语言，需要用 C++ 做一定规模开发的读者；
- 热爱 C/C++ 程序开发的所有读者。

作者团队

闫常友，高级工程师，电力系统及其自动化专业，九三学社社员，热爱计算机语言和软件开发，尤其对 C/C++ 系列有独到的见解和深厚的感情。

王敏，高级工程师，电力系统及其自动化专业，长期从事 C/C++ 语言的开发和项目实践，经验丰富。

致谢

本书的编写是非常艰难的。首先作者对目前市场上现有的 C/C++ 书籍做了深入调研，利用了无数个夜晚，编写了所有的例题，并完成调试。感谢王敏女士编写了第 1 章的部分内容。感谢本书的修订者袁静静完成了本书的修订，这使得读者对 C++ 的理解更加清晰，对本书的内容，也更加完善。其次，感谢负责本书的第一个审读者兼读者刘霞。她在写作过程中给予了我很多的鼓励和支持，使我得以写完本书。再次，还要感谢一下笔者对程序开发多年来的深深的热爱。笔者从 1998 年喜欢上 C/C++ 语言，目前已近不惑之年，把自己之前的一些想法和经验通过本书表达出来，也是值得庆幸的。

闫常友

2016 年 8 月

第 1 章 类模板简介

1.1 C++为什么需要模板功能	1
1.1.1 计算机实在是太傻了	1
1.1.2 类和函数重载部分解决了这个问题	2
1.1.3 泛型编程完美思路	2
1.1.4 C++的模板	3
1.1.5 C++ STL 的渊源	3
1.2 C++基本概念通览	4
1.2.1 命名空间	4
1.2.2 头文件	5
1.2.3 面向对象的程序设计	6
1.2.4 C++中的声明和定义	8
1.2.5 最简单的 C++程序	10
1.2.6 指针	12
1.2.7 函数	13
1.2.8 文件	17
1.2.9 程序的编译和链接	20
1.2.10 程序的启动和终止	21
1.2.11 异常处理	21
1.3 类模板定义	21
1.3.1 模板库 (STL)	22
1.3.2 STL 之父——Alexander Stepanov	22
1.3.3 类模板的英文原始定义	22
1.3.4 类模板实例化	22
1.3.5 类模板的成员函数	24
1.3.6 类模板的静态数据成员	24
1.4 成员模板	26
1.5 友元模板	27
1.6 函数模板	28
1.7 类模板的参数	31

1.7.1	关键字 <code>typename</code> 的使用	32
1.7.2	关键字 <code>typename</code> 与关键字 <code>class</code>	33
1.8	模板库简介	34
1.8.1	C 语言和 STL 的演变历史	34
1.8.2	STL 的组件	34
1.8.3	STL 的基本结构	35
1.8.4	STL 的编程概述	38
1.8.5	学习重点提示	40
1.9	本章小结	40

第 2 章 C++ 中的字符串

2.1	字符串库简述	41
2.2	字符的特点	42
2.3	字符串类模板 (<code>basic_string</code> 模板类)	43
2.4	字符串操作的通用函数	44
2.4.1	构造器和析构器	45
2.4.2	大小和容量	46
2.4.3	元素存取 (访问)	47
2.4.4	字符串比较	49
2.4.5	字符串内容修改和替换	51
2.4.6	字符串连接	57
2.4.7	字符串 I/O 操作	57
2.4.8	字符串的搜索和查找	58
2.4.9	字符串对迭代器的支持	62
2.4.10	字符串对配置器的支持	63
2.5	本章小结	64

第 3 章 容器

3.1	容器的概念	65
3.1.1	容器成员和函数	65
3.1.2	容器的种类和数据结构	66
3.2	序列式容器概述	67
3.3	序列式容器—— <code>vector</code> 类模板	68
3.3.1	<code>vector</code> 类基础	68
3.3.2	<code>vector</code> 类的成员函数	72
3.3.3	<code>vector</code> 高级编程	76
3.4	序列式容器—— <code>list</code> 类模板	84

3.4.1	list 的定义和容量	85
3.4.2	list 容器基础成员函数	90
3.4.3	运算符函数	94
3.4.4	其他重要成员函数	96
3.5	序列式容器——deque (双端队列) 类模板	102
3.5.1	容器 deque 和容器 vector 的对比	102
3.5.2	容器 deque 的定义和容量	103
3.5.3	deque 容器基础成员函数	104
3.5.4	deque 容器的高级编程	107
3.5.5	deque 的模板函数	109
3.6	关联式容器概述	110
3.7	关联式容器——set/multiset 类模板	110
3.7.1	集合 set 的定义	110
3.7.2	set 和 multiset 的容量、搜寻和统计	113
3.7.3	set 和 multiset 的迭代器相关函数和赋值函数	116
3.7.4	set 和 multiset 的插入和移除	118
3.7.5	set 和 multiset 的比较运算符	120
3.8	关联式容器——map/multimap 类模板	121
3.8.1	map 和 multimap 基础	122
3.8.2	map 和 multimap 成员函数	126
3.8.3	map 和 multimap 的高级编程	128
3.9	特殊容器用法	136
3.9.1	bitset 类模板	136
3.9.2	stack 类模板	139
3.9.3	队列 queue 类模板	141
3.9.4	Priority Queues 类模板	144
3.10	本章小结	146

第 4 章 C++ 中的算法

4.1	算法库简介	147
4.2	非修改性序列算法	148
4.2.1	for each 算法	148
4.2.2	元素计数算法	152
4.2.3	最小值和最大值算法	153
4.2.4	搜寻算法	155
4.2.5	区间比较算法	163
4.3	变动性算法	167

4.3.1	复制.....	167
4.3.2	转换.....	169
4.3.3	互换.....	173
4.3.4	赋值.....	174
4.3.5	替换.....	175
4.3.6	逆转.....	176
4.3.7	旋转.....	178
4.3.8	排列.....	179
4.4	排序及相关操作.....	183
4.4.1	全部元素排序.....	183
4.4.2	局部排序.....	185
4.4.3	根据某个元素排序.....	187
4.4.4	堆 (Heap) 操作.....	189
4.4.5	合并排序.....	191
4.4.6	搜索.....	194
4.5	删除算法.....	196
4.6	本章小结.....	199

第 5 章 迭代器 (Iterator)

5.1	迭代器及其特性.....	200
5.2	头文件<iterator>简述.....	201
5.3	迭代器类型详述.....	201
5.3.1	输入型迭代器.....	201
5.3.2	输出型迭代器.....	202
5.3.3	前向迭代器.....	202
5.3.4	双向迭代器.....	202
5.3.5	随机存取迭代器.....	202
5.3.6	vector 迭代器的递增和递减.....	203
5.4	迭代器配接器.....	203
5.4.1	逆向迭代器.....	204
5.4.2	插入型迭代器.....	205
5.4.3	流迭代器.....	207
5.5	迭代器辅助函数.....	210
5.5.1	advance()迭代器前进函数.....	210
5.5.2	distance()迭代器距离.....	211
5.5.3	iter_swap()交换两个迭代器所指内容.....	212
5.6	本章小结.....	213

第 6 章 STL 的数值计算

6.1	复数运算	214
6.1.1	最简单的复数运算例题	214
6.1.2	复数成员函数	215
6.1.3	复数运算符	216
6.1.4	复数运算	216
6.1.5	复数的超越函数	218
6.2	数组（向量）运算	221
6.2.1	类 valarray	221
6.2.2	数组子集类——slice 类和类模板 slice_array	228
6.2.3	类 gslice 和类模板 gslice_array	230
6.2.4	类 mask_array	232
6.2.5	类 indirect_array	233
6.3	通用数值计算	235
6.3.1	求和（accumulate）	235
6.3.2	内积（inner_product）	236
6.3.3	部分和（partial_sum）	238
6.3.4	序列相邻差（adjacent_difference）	239
6.4	全局性数学函数	241
6.5	本章小结	243

第 7 章 输入/输出流

7.1	IOStream 简介	244
7.1.1	Stream 对象	244
7.1.2	Stream 类别	245
7.1.3	Stream 操作符	246
7.1.4	操控器（Manipulators）	246
7.2	IOStream 基本类和标准 IOStream 对象	247
7.2.1	和 IOStream 类相关的头文件	247
7.2.2	标准 Stream 操作符	247
7.2.3	Stream 状态	251
7.2.4	标准输入和输出函数	254
7.3	格式化	259
7.3.1	格式标志	259
7.3.2	bool 类型数据的格式控制	260
7.3.3	详解“字段宽度、充填字符和位置调整”	261

7.3.4	正记号与大写字符	263
7.3.5	数值进制	264
7.3.6	浮点数输出	266
7.3.7	一般性格式定义	267
7.4	StreamBuffer 类介绍	268
7.4.1	Stream 缓冲区	268
7.4.2	缓冲区迭代器	269
7.4.3	自定义缓冲区	271
7.5	基于字符串的流	277
7.5.1	streambuf 类	277
7.5.2	类模板 basic_istream	279
7.5.3	类模板 basic_ostringstream	279
7.5.4	类模板 basic_stringstream	280
7.6	基于文件的流	280
7.6.1	文件标志及其使用	281
7.6.2	随机存取	290
7.6.3	4 个类模板简介	294
7.6.4	C 库中的文件存取功能概述	296
7.7	本章小结	298

第 8 章 异常处理

8.1	异常概念和基本思想	299
8.1.1	异常的概念	299
8.1.2	异常的分类	300
8.1.3	异常的捕捉和处理	302
8.1.4	资源管理	304
8.1.5	异常和效率	306
8.1.6	异常的描述	307
8.1.7	未捕捉的异常	309
8.2	异常类及几个重要问题	311
8.2.1	类 exception	311
8.2.2	调用 abort()	316
8.2.3	堆栈解退	318
8.2.4	错误代码	319
8.2.5	异常的迷失	319
8.2.6	异常处理的局限性	323
8.3	处理异常详述	324

8.3.1	异常处理的实现机制	325
8.3.2	异常处理语句的语法	326
8.3.3	异常处理不唤醒	326
8.3.4	函数声明	326
8.3.5	使用异常	326
8.4	异常的特殊处理函数	327
8.5	本章小结	328

第9章 通用工具

9.1	通用工具库简介	329
9.1.1	相等比较	329
9.1.2	小于比较	329
9.1.3	复制构造	332
9.1.4	默认构造	332
9.1.5	配置器要求	332
9.1.6	运算符	333
9.1.7	对组(pairs)	334
9.2	动态内存管理	339
9.2.1	默认配置器	339
9.2.2	raw storage iterator	341
9.2.3	temporary buffers (临时缓冲区)	341
9.2.4	特定算法	341
9.2.5	C 函数库中的内存管理函数	342
9.3	堆的内存分配	343
9.3.1	new 和 delete 运算符	343
9.3.2	分配固定维数的数组	343
9.3.3	分配动态内存数组	344
9.3.4	处理堆耗尽	345
9.4	辅助功能	345
9.4.1	数值极限	345
9.4.2	最大最小值 (较大较小值)	348
9.4.3	两值交换	349
9.4.4	辅助性比较	351
9.4.5	头文件 cstdlib 和 cstdint 简介	352
9.5	日期和时间	352
9.5.1	3 个类型	352
9.5.2	结构体 tm	353

9.5.3	相关时间函数	353
9.5.4	时间示例	356
9.6	模板类 auto_ptr	358
9.6.1	auto_ptr 类构造函数	359
9.6.2	类 auto_ptr 的成员及转换	359
9.6.3	使用 auto_ptr 类	360
9.7	本章小结	363

第 10 章 语言支持

10.1	类型	364
10.2	执行属性	364
10.2.1	类模板 numeric_limits 及其成员	365
10.2.2	float_round_style 和 float_denorm_style	367
10.2.3	数值极限的特化	368
10.2.4	库函数	368
10.2.5	应用举例	369
10.3	程序的启动和终止	376
10.4	动态内存管理	376
10.4.1	内存的分配和释放	377
10.4.2	内存分配错误	379
10.4.3	应用举例	380
10.5	类型标识符	382
10.5.1	类 type_info	382
10.5.2	类 bad_cast	383
10.5.3	类 bad_typeid	384
10.5.4	操作符 typeid	384
10.5.5	操作符 dynamic_cast	385
10.5.6	应用举例	385
10.6	异常处理	387
10.6.1	异常类 (class exception)	387
10.6.2	violating exception-specifications	388
10.6.3	abnormal termination	389
10.6.4	未捕获异常 (uncaught_exception)	390
10.6.5	应用举例	390
10.7	其他运行支持	392
10.7.1	概述	392
10.7.2	应用举例	393

10.8 本章小结	396
第 11 章 检测库详解	
11.1 异常类 Exception	397
11.1.1 类 logic_error	397
11.1.2 类 domain_error	398
11.1.3 类 invalid_argument	399
11.1.4 类 length_error	400
11.1.5 类 out_of_range	401
11.1.6 类 runtime_error	402
11.1.7 类 range_error	403
11.1.8 类 overflow_error	404
11.1.9 类 underflow_error (下溢出)	405
11.2 assertions (断言)	406
11.3 错误编码	408
11.4 本章小结	409
第 12 章 国际化库详解	
12.1 国际化问题和国际化元素	410
12.2 多种字符编码	411
12.2.1 宽字符和多字节文本	411
12.2.2 字符特性	412
12.2.3 特殊字符国际化	414
12.3 类 locale	414
12.3.1 类 locale 概述	414
12.3.2 类 locale 的 facet	417
12.3.3 区域表示和混合区域表示	420
12.3.4 流和区域	423
12.3.5 刻面的处理	424
12.4 标准 locale 的分类	426
12.4.1 类 ctype	426
12.4.2 数值类的 locale 类	440
12.4.3 刻面 numeric_punctuation	447
12.4.4 类 collate	449
12.4.5 time 类 (category)	451
12.4.6 C 库 locale	457
12.5 本章小结	458

第 13 章 仿函数

13.1	仿函数的概述.....	459
13.1.1	仿函数的概念.....	459
13.1.2	仿函数的作用.....	460
13.2	预定义仿函数.....	467
13.3	辅助用仿函数.....	468
13.3.1	一元组合函数配接器.....	469
13.3.2	二元组合函数配接器.....	471
13.4	关系仿函数.....	472
13.4.1	等于 (equal_to<type>()).....	472
13.4.2	不等于 (not_equal_to<type>()).....	473
13.4.3	小于 (less<type>()).....	474
13.4.4	大于 (greater<type>()).....	475
13.4.5	大于等于 (greater_equal) 和小于等于 (less_equal).....	475
13.5	逻辑仿函数.....	476
13.5.1	谓词.....	476
13.5.2	逻辑仿函数.....	477
13.6	算术仿函数.....	482
13.6.1	加减乘除运算仿函数 (plus).....	482
13.6.2	“求余”仿函数和“求反”仿函数.....	484
13.7	其他类型的仿函数.....	485
13.7.1	证和映射.....	486
13.7.2	仿函数 hash 和 subtractive_rng.....	489
13.8	适配器.....	489
13.8.1	成员函数适配器.....	490
13.8.2	其他适配器.....	495
13.9	本章小结.....	504

第 14 章 配置器

14.1	使用配置器.....	505
14.2	C++标准程序库的默认配置器 (标准配置器).....	507
14.3	自定义配置器.....	508
14.4	配置类的详细讨论.....	509
14.4.1	型别.....	509
14.4.2	配置类的成员函数.....	509
14.4.3	广义配置器.....	510

14.4.4 动态存储	510
14.4.5 C 风格的分配	511
14.5 未初始化的内存	511
14.6 配置器举例	513
14.7 本章小结	514

参考文献

第 1 章 类模板简介

在现今的 C++ 标准模板库中，几乎所有东西都被设计为 `template` 形式，不支持模板，就无法使用标准程序库。模板可以认为是针对一个或多个尚未明确的类型而编写一套函数或类型。模板是 C++ 的一个新特性。通过使用模板，C++ 允许推迟对某些类型的选择，直到想使用模板或者对模板进行专门化处理时。使用模板，可以让程序员面对相似而又略有不同的特性时，更快捷地编写代码，提高开发效率。

本章主要讲述 C/C++ 语言中常用的一些基本概念，如何定义类模板，介绍成员模板、友元模板、函数模板、类模板的参数，最后介绍模板库。

1.1 C++ 为什么需要模板功能

理解了这个话题，程序员才能深度理解 STL，才能用好 STL，所以，本书的开篇，让我们深入浅出地理解 C++ 的引入模板功能的思想。

1.1.1 计算机实在是太傻了

在大多数人的眼里，计算机既神秘又能干。

但在程序员的眼里，计算机实在是又蠢又笨。只不过运算的比人类速度快（电子的速度吗），记忆力好（能存储很多数据）。而且不给指令还什么都干不了。就是给指令，写一个程序，计算机也不灵活。比如，在 C++ 中，同样一个加法，居然要给出不同的数据类型的运算方式。

整数型，我们写这样一个函数：

```
int add(int m,int n)
{
    return m+n;
}
```

浮点型，我们写这样一个函数：

```
float add(float i,float j)
{
    return i+j;
}
.....
```