

Jacek Galowicz

C++17 STL Cookbook

Over 90 recipes that leverage the powerful features of
the standard library in C++17



Packt>

C++17 STL Cookbook

C++ has come a long way and is in use in every area of the industry. Fast, efficient, and flexible, it is used to solve many problems. The upcoming version of C++ will see programmers change the way they code. If you want to grasp the practical usefulness of the C++17 standard template library (STL) in order to write smarter, fully portable code, then this book is for you.

Beginning with new language features, this book will help you understand the language's mechanics and library features, and offers insight into how they work. Unlike other books, ours takes an implementation-specific, problem-solution approach that will help you quickly overcome hurdles. You will learn the core STL concepts, such as containers, algorithms, utility classes, lambda expressions, iterators, and more, while working on practical real-world recipes. These recipes will help you get the most from the STL and show you how to program in a better way.

By the end of the book, you will be up to date with the latest C++17 features and save time and effort while solving tasks elegantly using the STL.

Things you will learn:

- Learn about the new core language features and the problems they were intended to solve
- Understand the inner workings and requirements of iterators by implementing them
- Explore algorithms, functional programming style, and lambda expressions
- Leverage the rich, portable, fast, and well-designed and thoroughly tested algorithms provided in the STL
- Work with strings the STL way instead of handcrafting C-style code
- Understand standard support classes for concurrency and synchronization, and how to put them to work
- Use the filesystem library addition available with the C++17 STL

Packt
www.packtpub.com

\$ 49.99 US
£ 41.99 UK

Prices do not include local sales
Tax or VAT where applicable



C++17 STILL Cookbook

Jacek Galowicz



C++17 STL Cookbook

Over 90 recipes that leverage the powerful features of the standard library in C++17

Jacek Galowicz

Packt>

BIRMINGHAM - MUMBAI

C++17 STL Cookbook

Copyright © 2017 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the authors, nor Packt Publishing, and its dealers and distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: June 2017

Production reference: 1230617

Published by Packt Publishing Ltd.

Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-78712-049-5

www.packtpub.com

Credits

Author

Jacek Galowicz

Copy Editor

Muktikant Garimella

Reviewer

Arne Mertz

Project Coordinator

Ulhas Kambali

Commissioning Editor

Aaron Lazar

Proofreader

Safis Editing

Acquisition Editor

Nitin Dasan

Indexer

Francy Puthiry

Content Development Editor

Vikas Tiwari

Graphics

Abhinash Sahu

Technical Editor

Hussain Kanchwala

Production Coordinator

Shantanu Zagade

About the Author

Jacek Galowicz obtained his master of science in electrical engineering/computer engineering at RWTH Aachen University, Germany. While at university, he enjoyed working as a student assistant in teaching and research, and he participated in several scientific publications. During and after his studies, he worked as a freelancer and implemented applications as well as kernel drivers in C and C++, touching various areas, including 3D graphics programming, databases, network communication, and physics simulation. In recent years, he has been programming performance- and security-sensitive microkernel operating systems for Intel x86 virtualization at Intel and FireEye in Braunschweig, Germany. He has a strong passion for modern C++ implementations of low-level software, and he tries hard to combine high performance with an elegant coding style. Learning purely functional programming and Haskell in recent years triggered his drive to implement generic code with the aid of meta programming.

Writing a book and founding a company at the same time was a great and interesting experience in my life and a lot of fun. The fun aspects, however, were only possible because of the support and patience of my wonderful girlfriend Viktoria, my fellow co-founders, and all my friends. Special thanks go to Arne Mertz for his invaluable and meticulous review suggestions, as well as Torsten Robitzki and Oliver Bruns from the C++ user group Hannover for their great feedback.

About the Reviewer

Arne Mertz is a C++ expert with over a decade of experience. He studied physics at the university of Hamburg, and he switched careers to become a software developer. His main background is in financial enterprise applications written in C++. Arne works at Zühlke Engineering, Germany and is known for his blog, *Simplify C++!* (<https://arne-mertz.de>) on clean and maintainable C++.

www.PacktPub.com

For support files and downloads related to your book, please visit www.PacktPub.com.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.PacktPub.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at service@packtpub.com for more details.

At www.PacktPub.com, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.



<https://www.packtpub.com/mapt>

Get the most in-demand software skills with Mapt. Mapt gives you full access to all Packt books and video courses, as well as industry-leading tools to help you plan your personal development and advance your career.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

Customer Feedback

Thanks for purchasing this Packt book. At Packt, quality is at the heart of our editorial process. To help us improve, please leave us an honest review on this book's Amazon page at <https://www.amazon.com/dp/178712049X>.

If you'd like to join our team of regular reviewers, you can e-mail us at customerreviews@packtpub.com. We award our regular reviewers with free eBooks and videos in exchange for their valuable feedback. Help us be relentless in improving our products!

Table of Contents

Preface	1
<hr/> Chapter 1: The New C++17 Features	<hr/> 11
Introduction	11
Using structured bindings to unpack bundled return values	12
How to do it...	12
How it works...	14
There's more...	14
Limiting variable scopes to if and switch statements	16
How to do it...	17
How it works...	17
There's more...	19
Profiting from the new bracket initializer rules	20
How to do it...	20
How it works...	21
Letting the constructor automatically deduce the resulting template class type	22
How to do it...	22
How it works...	23
There's more...	24
Simplifying compile time decisions with constexpr-if	25
How to do it...	25
How it works...	26
There's more...	27
Enabling header-only libraries with inline variables	29
How it's done...	29
How it works...	30
There's more...	32
Implementing handy helper functions with fold expressions	32
How to do it...	33
How it works...	33
There's more...	34
Match ranges against individual items	36
Check if multiple insertions into a set are successful	37
Check if all the parameters are within a certain range	38
Pushing multiple items into a vector	39

Chapter 2: STL Containers	41
Introduction	42
Contiguous storage	42
List storage	43
Search trees	43
Hash tables	44
Container adapters	44
Using the erase-remove idiom on std::vector	44
How to do it...	45
How it works...	47
There's more...	48
Deleting items from an unsorted std::vector in O(1) time	49
How to do it...	49
How it works...	52
Accessing std::vector instances the fast or the safe way	53
How to do it...	53
How it works...	54
There's more...	55
Keeping std::vector instances sorted	55
How to do it...	55
How it works...	57
There's more...	58
Inserting items efficiently and conditionally into std::map	58
How to do it...	59
How it works...	61
There's more...	62
Knowing the new insertion hint semantics of std::map::insert	62
How to do it...	62
How it works...	63
There's more...	64
Efficiently modifying the keys of std::map items	65
How to do it...	66
How it works...	68
There's more...	68
Using std::unordered_map with custom types	69
How to do it...	69
How it works...	71
Filtering duplicates from user input and printing them in alphabetical order with std::set	72

How to do it...	73
How it works...	74
std::istream_iterator	74
std::inserter	75
Putting it together	76
Implementing a simple RPN calculator with std::stack	76
How to do it...	77
How it works...	80
Stack handling	80
Distinguishing operands from operations from user input	81
Selecting and applying the right mathematical operation	82
There's more...	82
Implementing a word frequency counter with std::map	83
How to do it...	83
How it works...	86
Implement a writing style helper tool for finding very long sentences in text with std::multimap	87
How to do it...	88
How it works...	91
There's more...	92
Implementing a personal to-do list using std::priority_queue	92
How to do it...	93
How it works...	95
Chapter 3: Iterators	97
<hr/>	
Introduction	97
Iterator categories	99
Input iterator	100
Forward iterator	100
Bidirectional iterator	100
Random access iterator	101
Contiguous iterator	101
Output iterator	101
Mutable iterator	101
Building your own iterable range	101
How to do it...	102
How it works...	104
Making your own iterators compatible with STL iterator categories	105
How to do it...	105
How it works...	108
There's more...	108
Using iterator adapters to fill generic data structures	109

How to do it...	109
How it works...	111
std::back_insert_iterator	111
std::front_insert_iterator	111
std::insert_iterator	112
std::istream_iterator	112
std::ostream_iterator	112
Implementing algorithms in terms of iterators	112
How to do it...	113
There's more...	116
Iterating the other way around using reverse iterator adapters	116
How to do it...	117
How it works...	118
Terminating iterations over ranges with iterator sentinels	119
How to do it...	120
Automatically checking iterator code with checked iterators	122
How to do it...	123
How it works...	125
There's more...	126
Building your own zip iterator adapter	127
How to do it...	129
There's more...	132
Ranges library	133
Chapter 4: Lambda Expressions	135
Introduction	135
Defining functions on the run using lambda expressions	137
How to do it...	137
How it works...	140
Capture list	141
mutable (optional)	142
constexpr (optional)	142
exception attr (optional)	142
return type (optional)	142
Adding polymorphy by wrapping lambdas into std::function	142
How to do it...	143
How it works...	145
Composing functions by concatenation	146
How to do it...	147
How it works...	149
Creating complex predicates with logical conjunction	150
How to do it...	150

There's more...	152
Calling multiple functions with the same input	153
How to do it...	153
How it works...	155
Implementing transform_if using std::accumulate and lambdas	157
How to do it...	157
How it works...	160
Generating cartesian product pairs of any input at compile time	163
How to do it...	164
How it works...	166
Chapter 5: STL Algorithm Basics	169
<hr/>	
Introduction	170
Copying items from containers to other containers	172
How to do it...	173
How it works...	175
Sorting containers	177
How to do it...	177
How it works...	181
Removing specific items from containers	181
How to do it...	182
How it works...	185
Transforming the contents of containers	185
How to do it...	186
How it works...	188
Finding items in ordered and unordered vectors	188
How to do it...	189
How it works...	193
Limiting the values of a vector to a specific numeric range with std::clamp	195
How to do it...	196
How it works...	199
Locating patterns in strings with std::search and choosing the optimal implementation	199
How to do it...	200
How it works...	202
Sampling large vectors	204
How to do it...	205
How it works...	208
Generating permutations of input sequences	209

How to do it...	209
How it works...	210
Implementing a dictionary merging tool	211
How to do it...	212
How it works...	214
Chapter 6: Advanced Use of STL Algorithms	215
<hr/>	
Introduction	215
Implementing a trie class using STL algorithms	216
How to do it...	217
How it works...	221
Implementing a search input suggestion generator with tries	222
How to do it...	223
How it works...	227
There's more...	228
Implementing the Fourier transform formula with STL numeric algorithms	228
How to do it...	229
How it works...	235
Calculating the error sum of two vectors	237
How to do it...	237
How it works...	240
Implementing an ASCII Mandelbrot renderer	241
How to do it...	242
How it works...	246
Building our own algorithm - split	247
How to do it...	248
How it works...	250
There's more...	251
Composing useful algorithms from standard algorithms - gather	251
How to do it...	252
How it works...	254
Removing consecutive whitespace between words	256
How to do it...	257
How it works...	258
Compressing and decompressing strings	259
How to do it...	260
How it works...	262
There's more...	263