

Marius Bancila

Modern C++ Programming Cookbook

Over 100 recipes to help you overcome your difficulties with C++ programming and gain a deeper understanding of the working of modern C++



Packt>

Modern C++ Programming Cookbook

C++ is one of the oldest and most widely used programming languages. Fast, efficient, and flexible, it is used to solve many problems. The latest versions of C++ have seen programmers change the way they code, giving up on the old-fashioned C-style programming and adopting modern C++ instead.

Beginning with the modern language features, each recipe addresses a specific problem, with a discussion that explains the solution and offers insight into how it works. You will learn about concepts such as concurrency, variadic templates, lambda expressions, regular expressions, streams and filesystem utilities, algorithms and iterators, move semantics and performance, exception handling, testing, and more in the form of recipes. These recipes will ensure you can make your applications secure and fast.

By the end of the book, you will understand the newer aspects of C++ 11/14/17 and will be able to overcome tasks that are time-consuming or would break your stride while developing.

Things you will learn:

- Understand the standard support for threading and concurrency and learn how to put them to work on daily basic tasks
- Look in depth at the C++17 filesystem library
- Work with various types of string and look at the various aspects of compilation
- Explore functions, lambda expressions and callable objects with a focus on modern features
- Leverage the standard library and work with containers, algorithms, and iterators I/O, time and utilities
- Solve text searching and replacement problems using regular expressions
- Use the new utility additions to the standard library to solve common problems developers encounter, including `string_view`, `any`, `optional`, and `variant` types
- Explore the widely-used testing frameworks for C++ and implement various useful patterns and idioms

Packt
www.packtpub.com

\$ 49.99 US
£ 41.99 UK

Prices do not include local sales
Tax or VAT where applicable



Modern C++ Programming Cookbook

Marius Bancila



Modern C++ Programming Cookbook

Over 100 recipes to help you overcome your difficulties with C++ programming and gain a deeper understanding of the working of modern C++

Marius Bancila



BIRMINGHAM - MUMBAI

Modern C++ Programming 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 author, 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: May 2017

Production reference: 1090517

Published by Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham

B3 2PB, UK.

ISBN 978-1-78646-518-4

www.packtpub.com

Credits

Author

Marius Bancila

Copy Editor

Dhanya Baburaj

Reviewer

David V. Corbin

Project Coordinator

Vaidehi Sawant

Commissioning Editor

Aaron Lazar

Proofreader

Safis Editing

Acquisition Editor

Nitin Dasan

Indexer

Aishwarya Gangawane

Content Development Editor

Anurag Ghogre

Cover Work

Arvindkumar Gupta

Technical Editor

Subhalaxmi Nadar

Production Coordinator

Arvindkumar Gupta

About the Author

Marius Bancila is a software engineer with 14 years of experience in developing solutions for the industrial and financial sectors. He focuses on Microsoft technologies and mainly develops desktop applications with C++ and C#. Over the years, he has worked with other languages and technologies including Java, HTML/CSS, PHP, and JavaScript.

Marius is passionate about sharing his technical expertise with others, and for that reason, he has been recognized as a Microsoft MVP for more than a decade. He has been an active contributor to forums and other developer communities where he has published many articles, for which he has won multiple awards. He also created and contributed to several open source libraries. He is a cofounder of Codexpert, a Romanian community for C++ developers. He is based in Timisoara, Romania, and works as a system architect, building accounting and logistic solutions for a major European software vendor. He can be followed on Twitter at <https://twitter.com/mariusbancila>.

I would like to thank Packt Publishing for getting me on board with this wonderful project that I greatly enjoyed. Many thanks to Anurag Ghogre, Subhalaxmi Nadar and Nitin Dasan for the constant support shown throughout the project, as well as the other members of the team involved. A special thanks to David Corbin who provided valuable feedback to make this book better. Last, but not least, a big thank you to my wife who has been very patient and supportive through the many days and nights I spent writing this book.

About the Reviewer

David V. Corbin began programming during the heyday of the mini-computer era, starting with the DEC PDP-8. His early career was in the defense industry, progressing from the company's first software technician to being the technical lead of the engineering software department, with much of the work being done in C. He cofounded Dynamic Concepts in 1984 to facilitate the introduction of the PC into business environments (this is the same year the original IBM AT was introduced).

By the early 1990s, much of his application development had started migrating to C++. Even after 25 years, C++ remains a valued tool in his development arsenal. In 2005, he began to focus on improving the software development and delivery process via the application of ALM principles. Today, he continues as the President and Chief Architect of Dynamic Concepts and works directly with clients, providing guidance in the rapidly changing ecosystem.

I would like to thank Packt Publishing for the opportunity to be a technical reviewer of this book. I have known Marius Bancila for a decade and he is one of the brightest developers I have known. His work is sure to have a positive impact on the entire C++ developer community in their quest to become familiar with the modern elements of 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/1786465183>.

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
Chapter 1: Learning Modern Core Language Features	7
Introduction	7
Using auto whenever possible	8
How to do it...	8
How it works...	9
See also	13
Creating type aliases and alias templates	13
How to do it...	13
How it works...	14
Understanding uniform initialization	15
Getting ready	16
How to do it...	16
How it works...	17
There's more	20
See also	21
Understanding the various forms of non-static member initialization	22
How to do it...	22
How it works...	23
Controlling and querying object alignment	27
Getting ready	27
How to do it...	27
How it works...	28
Using scoped enumerations	32
How to do it...	32
How it works...	33
Using override and final for virtual methods	34
Getting ready	34
How to do it...	35
How it works...	36
Using range-based for loops to iterate on a range	38
Getting ready	38
How to do it...	39
How it works...	39

See also	40
Enabling range-based for loops for custom types	41
Getting ready	41
How to do it...	42
How it works...	44
See also	45
Using explicit constructors and conversion operators to avoid implicit conversion	45
Getting ready	45
How to do it...	45
How it works...	46
See also	50
Using unnamed namespaces instead of static globals	51
Getting ready	51
How to do it...	51
How it works...	52
See also	53
Using inline namespaces for symbol versioning	54
Getting ready	54
How to do it...	54
How it works...	55
See also	57
Using structured bindings to handle multi-return values	58
Getting ready	58
How to do it...	58
How it works...	59
Chapter 2: Working with Numbers and Strings	61
Introduction	61
Converting between numeric and string types	62
Getting ready	62
How to do it...	62
How it works...	63
See also	67
Limits and other properties of numeric types	68
Getting ready	68
How to do it...	68
How it works...	69
Generating pseudo-random numbers	72
Getting ready	72

How to do it...	72
How it works...	73
See also	79
Initializing all bits of internal state of a pseudo-random number generator	79
Getting ready	79
How to do it...	80
How it works...	80
Creating cooked user-defined literals	81
Getting ready	81
How to do it...	82
How it works...	82
There's more...	86
See also	86
Creating raw user-defined literals	87
Getting ready	87
How to do it...	87
How it works...	89
See also	92
Using raw string literals to avoid escaping characters	92
Getting ready	93
How to do it...	93
How it works...	93
See also	94
Creating a library of string helpers	94
Getting ready	95
How to do it...	96
How it works...	98
See also	101
Verifying the format of a string using regular expressions	101
Getting ready	101
How to do it...	102
How it works...	102
There's more...	108
See also	110
Parsing the content of a string using regular expressions	110
Getting ready	110
How to do it...	111
How it works...	112

See also	115
Replacing the content of a string using regular expressions	115
Getting ready	115
How to do it...	116
How it works...	116
See also	119
Using string_view instead of constant string references	119
Getting ready	119
How to do it...	120
How it works...	120
See also	123
Chapter 3: Exploring Functions	125
Introduction	125
Defaulted and deleted functions	126
Getting started	126
How to do it...	126
How it works...	128
Using lambdas with standard algorithms	130
Getting ready	130
How to do it...	130
How it works...	131
There's more...	134
See also	134
Using generic lambdas	134
Getting started	135
How to do it...	135
How it works...	136
See also	137
Writing a recursive lambda	137
Getting ready	138
How to do it...	138
How it works...	139
Writing a function template with a variable number of arguments	141
Getting ready	141
How to do it...	141
How it works...	142
See also	146
Using fold expressions to simplify variadic function templates	146
Getting ready	146

How to do it...	147
How it works...	148
There's more...	149
See also	150
Implementing higher-order functions map and fold	150
Getting ready	150
How to do it...	151
How it works...	153
There's more...	156
See also	158
Composing functions into a higher-order function	158
Getting ready	159
How to do it...	159
How it works...	160
There's more...	160
See also	161
Uniformly invoking anything callable	162
Getting ready	162
How to do it...	163
How it works...	164
See also	165
Chapter 4: Preprocessor and Compilation	167
Introduction	167
Conditionally compiling your source code	168
Getting ready	168
How to do it...	168
How it works...	171
See also	172
Using the indirection pattern for preprocessor stringification and concatenation	172
Getting ready	173
How to do it...	173
How it works...	173
See also	175
Performing compile-time assertion checks with static_assert	176
Getting ready	176
How to do it...	176
How it works...	177
See also	178

Conditionally compiling classes and functions with <code>enable_if</code>	178
Getting ready	178
How to do it...	179
How it works...	180
There's more...	182
See also	184
Selecting branches at compile time with <code>constexpr if</code>	185
Getting ready	185
How to do it...	186
How it works...	187
Providing metadata to the compiler with attributes	188
How to do it...	188
How it works...	190
Chapter 5: Standard Library Containers, Algorithms, and Iterators	193
Introduction	193
Using <code>vector</code> as a default container	194
Getting ready	194
How to do it...	194
How it works...	197
There's more...	199
See also	199
Using <code>bitset</code> for fixed-size sequences of bits	200
Getting ready	200
How to do it...	200
How it works...	203
There's more...	205
See also	206
Using <code>vector<bool></code> for variable-size sequences of bits	206
Getting ready...	206
How to do it...	207
How it works...	208
There's more...	208
See also	211
Finding elements in a range	211
Getting ready	211
How to do it...	211
How it works...	215
There's more...	216
See also	216