PROFESSIONAL C++

Continues

PROFESSIONAL

C++

PROFESSIONAL

C++

Fourth Edition

Marc Gregoire

Professional C++, Fourth Edition

Published by
John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Published simultaneously in Canada

ISBN: 978-1-119-42130-6
ISBN: 978-1-119-42126-9 (ebk)
ISBN: 978-1-119-42122-1 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Library of Congress Control Number: 2017963243

Dedicated to my parents and my brother, who are always there for me. Their support and patience helped me in finishing this book.

ABOUT THE AUTHOR

MARC GREGOIRE is a software architect from Belgium. He graduated from the University of Leuven, Belgium, with a degree in "Burgerlijk ingenieur in de computer wetenschappen" (equivalent to master of science in engineering: computer science). The year after, he received an advanced master's degree in artificial intelligence, cum laude, at the same university. After his studies, Marc started working for a software consultancy company called Ordina Belgium. As a consultant, he worked for Siemens and Nokia Siemens Networks on critical 2G and 3G software running on Solaris for telecom operators. This required working with international teams stretching from South America and the United States to Europe, the Middle East, Africa, and Asia. Now, Marc is a software architect at Nikon Metrology (`www.nikonmetrology.com`), a division of Nikon and a leading provider of precision optical instruments and metrology solutions for 3D geometric inspection.

His main expertise is in C/C++, and specifically Microsoft VC++ and the MFC framework. He has experience in developing C++ programs running 24/7 on Windows and Linux platforms: for example, KNX/EIB home automation software. In addition to C/C++, Marc also likes C# and uses PHP for creating web pages.

Since April 2007, he has received the annual Microsoft MVP (Most Valuable Professional) award for his Visual C++ expertise.

Marc is the founder of the Belgian C++ Users Group (`www.becpp.org`), co-author of *C++ Standard Library Quick Reference* (Apress), technical editor for numerous books for several publishers, and a member on the CodeGuru forum (as Marc G). He maintains a blog at `www.nuonsoft.com/blog/`, and is passionate about traveling and gastronomic restaurants.

ABOUT THE TECHNICAL EDITOR

PETER VAN WEERT is a Belgian software engineer, whose main interests and expertise are in C++, programming languages, algorithms, and data structures.

He received his master of science in computer science from the University of Leuven, Belgium, summa cum laude, with congratulations of the Board of Examiners. In 2010, the same university awarded him a PhD for his research on the efficient compilation of rule-based programming languages (mainly Java). During his doctoral studies, he was a teaching assistant for courses on object-oriented analysis and design, Java programming, and declarative programming languages.

After his studies, Peter worked for Nikon Metrology on large-scale, industrial-application software in the area of 3D laser scanning and point cloud inspection. In 2017, he joined the software R&D unit of Nobel Biocare, which specializes in digital dentistry software. Throughout his professional career, Peter has mastered C++ software development, as well as the management, refactoring, and debugging of very large code bases. He also gained further proficiency in all aspects of the software development process, including the analysis of functional and technical requirements, and Agile- and Scrum-based project and team management.

Peter is a regular speaker at, and board member of, the Belgian C++ Users Group. He also co-authored two books: *C++ Standard Library Quick Reference* and *Beginning C++ (5th edition)*, both published by Apress.

CREDITS

PROJECT EDITOR
Adaobi Obi Tulton

TECHNICAL EDITOR
Peter Van Weert

PRODUCTION EDITOR
Athiyappan Lalith Kumar

COPY EDITOR
Marylouise Wiack

MANAGER OF CONTENT DEVELOPMENT AND ASSEMBLY
Mary Beth Wakefield

PRODUCTION MANAGER
Kathleen Wisor

MARKETING MANAGER
Christie Hilbrich

EXECUTIVE EDITOR
Jim Minatel

PROJECT COORDINATOR, COVER
Brent Savage

PROOFREADER
Nancy Bell

INDEXER
Johnna VanHoose Dinse

COVER DESIGNER
Wiley

COVER IMAGE
© ittipon/Shutterstock

ACKNOWLEDGMENTS

I THANK THE JOHN WILEY & SONS AND WROX Press editorial and production teams for their support. Especially, thank you to Jim Minatel, executive editor at Wiley, for giving me a chance to write this new edition, Adaobi Obi Tulton, project editor, for managing this project, and Marylouise Wiack, copy editor, for improving readability and consistency and making sure the text is grammatically correct.

A very special thank you to my technical editor, Peter Van Weert, for his outstanding technical review. His many constructive comments and ideas have certainly made this book better.

Of course, the support and patience of my parents and my brother were very important in finishing this book. I would also like to express my sincere gratitude toward my employer, Nikon Metrology, for supporting me during this project.

Finally, I thank you, the reader, for trying this approach to professional C++ software development.

CONTENTS

INTRODUCTION

For many years, C++ has served as the de facto language for writing fast, powerful, and enterprise-class object-oriented programs. As popular as C++ has become, the language is surprisingly difficult to grasp in full. There are simple, but powerful, techniques that professional C++ programmers use that don't show up in traditional texts, and there are useful parts of C++ that remain a mystery even to experienced C++ programmers.

Too often, programming books focus on the syntax of the language instead of its real-world use. The typical C++ text introduces a major part of the language in each chapter, explaining the syntax and providing an example. *Professional C++* does not follow this pattern. Instead of giving you just the nuts and bolts of the language with little practical context, this book will teach you how to use C++ in the real world. It will show you the little-known features that will make your life easier, and the programming techniques that separate novices from professional programmers.

WHO THIS BOOK IS FOR

Even if you have used the language for years, you might still be unfamiliar with the more-advanced features of C++, or you might not be using the full capabilities of the language. Perhaps you write competent C++ code, but would like to learn more about design and good programming style in C++. Or maybe you're relatively new to C++, but want to learn the "right" way to program from the start. This book will meet those needs and bring your C++ skills to the professional level.

Because this book focuses on advancing from basic or intermediate knowledge of C++ to becoming a professional C++ programmer, it assumes that you have some knowledge of the language. Chapter 1 covers the basics of C++ as a refresher, but it is not a substitute for actual training and use of the language. If you are just starting with C++, but you have significant experience in another programming language such as C, Java, or C#, you should be able to pick up most of what you need from Chapter 1.

In any case, you should have a solid foundation in programming fundamentals. You should know about loops, functions, and variables. You should know how to structure a program, and you should be familiar with fundamental techniques such as recursion. You should have some knowledge of common data structures such as queues, and useful algorithms such as sorting and searching. You don't need to know about object-oriented programming just yet—that is covered in Chapter 5.

You will also need to be familiar with the compiler you will be using to develop your code. Two compilers, Microsoft Visual C++ and GCC, are introduced later in this introduction. For other compilers, refer to the documentation that came with your compiler.

WHAT THIS BOOK COVERS

Professional C++ uses an approach to C++ programming that will both increase the quality of your code and improve your programming efficiency. You will find discussions on new C++17 features throughout this fourth edition. These features are not just isolated to a few chapters or sections; instead, examples have been updated to use new features when appropriate.

Professional C++ teaches you more than just the syntax and language features of C++. It also emphasizes programming methodologies, reusable design patterns, and good programming style. The *Professional* C++ methodology incorporates the entire software development process, from designing and writing code, to debugging, and working in groups. This approach will enable you to master the C++ language and its idiosyncrasies, as well as take advantage of its powerful capabilities for large-scale software development.

Imagine users who have learned all of the syntax of C++ without seeing a single example of its use. They know just enough to be dangerous! Without examples, they might assume that all code should go in the `main()` function of the program, or that all variables should be global—practices that are generally not considered hallmarks of good programming.

Professional C++ programmers understand the correct way to use the language, in addition to the syntax. They recognize the importance of good design, the theories of object-oriented programming, and the best ways to use existing libraries. They have also developed an arsenal of useful code and reusable ideas.

By reading and understanding this book, you will become a professional C++ programmer. You will expand your knowledge of C++ to cover lesser-known and often misunderstood language features. You will gain an appreciation for object-oriented design, and acquire top-notch debugging skills. Perhaps most important, you will finish this book armed with a wealth of reusable ideas that you can actually apply to your daily work.

There are many good reasons to make the effort to be a professional C++ programmer, as opposed to a programmer who knows C++. Understanding the true workings of the language will improve the quality of your code. Learning about different programming methodologies and processes will help you to work better with your team. Discovering reusable libraries and common design patterns will improve your daily efficiency and help you stop reinventing the wheel. All of these lessons will make you a better programmer and a more valuable employee. While this book can't guarantee you a promotion, it certainly won't hurt.

HOW THIS BOOK IS STRUCTURED

This book is made up of five parts.

Part I, "Introduction to Professional C++," begins with a crash course in C++ basics to ensure a foundation of C++ knowledge. Following the crash course, Part I goes deeper into working with strings and string views because strings are used extensively in most examples throughout the book. The last chapter of Part I explores how to write *readable* C++ code.

Part II, "Professional C++ Software Design," discusses C++ design methodologies. You will read about the importance of design, the object-oriented methodology, and the importance of code reuse.

Part III, "C++ Coding the Professional Way," provides a technical tour of C++ from the professional point of view. You will read about the best ways to manage memory in C++, how to create reusable classes, and how to leverage important language features such as inheritance. You will also learn about the unusual and quirky parts of the language, techniques for input and output, error handling, string localization, and how to work with regular expressions. You will read about how to implement operator overloading, and how to write templates. This part also explains the C++ Standard Library, including containers, iterators, and algorithms. You will also read about some additional libraries that are available in the standard, such as the libraries to work with time, random numbers, and the filesystem.

Part IV, "Mastering Advanced Features of C++," demonstrates how you can get the most out of C++. This part of the book exposes the mysteries of C++ and describes how to use some of its more-advanced features. You will read about how to customize and extend the C++ Standard Library to your needs, advanced details on template programming, including template metaprogramming, and how to use multithreading to take advantage of multiprocessor and multicore systems.

Part V, "C++ Software Engineering," focuses on writing enterprise-quality software. You'll read about the engineering practices being used by programming organizations today; how to write efficient C++ code; software testing concepts, such as unit testing and regression testing; techniques used to debug C++ programs; how to incorporate design techniques, frameworks, and conceptual object-oriented design patterns into your own code; and solutions for cross-language and cross-platform code.

The book concludes with a useful chapter-by-chapter guide to succeeding in a C++ technical interview, an annotated bibliography, a summary of the C++ header files available in the standard, and a brief introduction to the Unified Modeling Language (UML).

This book is not a reference of every single class, method, and function available in C++. The book *C++ Standard Library Quick Reference* by Peter Van Weert and Marc Gregoire[1] is a condensed reference to all essential data structures, algorithms, and functions provided by the C++ Standard Library. Appendix B lists a couple more references. Two excellent online references are:

- `www.cppreference.com`

  You can use this reference online, or download an offline version for use when you are not connected to the Internet.

- `www.cplusplus.com/reference/`

When I refer to a "Standard Library Reference" in this book, I am referring to one of these detailed C++ references.

[1]Apress, 2016. ISBN: 978-1-4842-1875-4.

WHAT YOU NEED TO USE THIS BOOK

All you need to use this book is a computer with a C++ compiler. This book focuses only on parts of C++ that have been standardized, and not on vendor-specific compiler extensions.

Note that this book includes new features introduced with the C++17 standard. At the time of this writing, some compilers are not yet fully C++17 compliant.

You can use whichever C++ compiler you like. If you don't have a C++ compiler yet, you can download one for free. There are a lot of choices. For example, for Windows, you can download Microsoft Visual Studio 2017 Community Edition, which is free and includes Visual C++. For Linux, you can use GCC or Clang, which are also free.

The following two sections briefly explain how to use Visual C++ and GCC. Refer to the documentation that came with your compiler for more details.

Microsoft Visual C++

First, you need to create a project. Start Visual C++ and click File ➪ New ➪ Project. In the project template tree on the left, select Visual C++ ➪ Win32 (or Windows Desktop). Then select the Win32 Console Application (or Windows Console Application) template in the list in the middle of the window. At the bottom, specify a name for the project and a location where to save it, and click OK. A wizard opens[2]. In this wizard, click Next, select Console Application, Empty Project, and click Finish.

Once your new project is loaded, you can see a list of project files in the Solution Explorer. If this docking window is not visible, go to View ➪ Solution Explorer. You can add new files or existing files to a project by right-clicking the project name in the Solution Explorer and then clicking Add ➪ New Item or Add ➪ Existing Item.

Use Build ➪ Build Solution to compile your code. When it compiles without errors, you can run it with Debug ➪ Start Debugging.

If your program exits before you have a chance to view the output, use Debug ➪ Start without Debugging. This adds a pause to the end of the program so you can view the output.

At the time of this writing, Visual C++ 2017 does not yet automatically enable C++17 features. To enable C++17 features, in the Solution Explorer window, right-click your project and click Properties. In the properties window, go to Configuration Properties ➪ C/C++ ➪ Language, and set the C++ Language Standard option to "ISO C++17 Standard" or "ISO C++ Latest Draft Standard," whichever is available in your version of Visual C++. These options are only accessible if your project contains at least one .cpp file.

Visual C++ supports so-called precompiled headers, a topic outside the scope of this book. In general, I recommend using precompiled headers if your compiler supports them. However, the source

[2]Depending on your version of VC++ 2017, you might not see any wizard. Instead, a new project will be created automatically containing four files: stdafx.h, stdafx.cpp, targetver.h, and <projectname>.cpp. If that is the case, and you want to compile source code files from the downloadable source archive for this book, then you have to select those files in the Solution Explorer (View ➪ Solution Explorer) and delete them.

code files in the downloadable source code archive do not use precompiled headers, so you have to disable that feature for them to compile without errors. In the Solution Explorer window, right-click your project and click Properties. In the properties window, go to Configuration Properties ➪ C/C++ ➪ Precompiled Headers, and set the Precompiled Header option to "Not Using Precompiled Headers."

GCC

Create your source code files with any text editor you prefer and save them to a directory. To compile your code, open a terminal and run the following command, specifying all your `.cpp` files that you want to compile:

```
gcc -lstdc++ -std=c++17 -o <executable_name> <source1.cpp> [source2.cpp ...]
```

The `-std=c++17` option is required to tell GCC to enable C++17 support.

For example, you can compile the `AirlineTicket` example from Chapter 1 by changing to the directory containing the code and running the following command:

```
gcc –lstdc++ -std=c++17 -o AirlineTicket AirlineTicket.cpp AirlineTicketTest.cpp
```

When it compiles without errors, you can run it as follows:

```
./AirlineTicket
```

CONVENTIONS

To help you get the most from the text and keep track of what's happening, a number of conventions are used throughout this book.

> **WARNING** *Boxes like this one hold important, not-to-be-forgotten information that is directly relevant to the surrounding text.*

> **NOTE** *Tips, hints, tricks, and asides to the current discussion are placed in boxes like this one.*

As for styles in the text:

Important words are *highlighted* when they are introduced.

Keyboard strokes are shown like this: Ctrl+A.

Filenames and code within the text are shown like so: `monkey.cpp`.

URLs are shown like this: `www.wrox.com`.

Code is presented in three different ways:

```
// Comments in code are shown like this.
In code examples, new and important code is highlighted like this.
Code that's less important in the present context or that has been shown before is
formatted like this.
```

C++17

Paragraphs or sections that are specific to the C++17 standard have a little C++17 icon on the left, just as this paragraph does. C++11 and C++14 features are not marked with any icon.

SOURCE CODE

As you work through the examples in this book, you may choose either to type in all the code manually, or to use the source code files that accompany the book. However, I suggest you type in all the code manually because it greatly benefits the learning process and your memory. All of the source code used in this book is available for download at `www.wiley.com/go/proc++4e`.

Alternatively, you can go to the main Wrox code download page at `www.wrox.com/dynamic/books/download.aspx` to see the code that is available for this book and all other Wrox books.

> **NOTE** *Because many books have similar titles, you may find it easiest to search by ISBN; for this book, the ISBN is 978-1-119-42130-6.*

Once you've downloaded the code, just decompress it with your favorite decompression tool.

ERRATA

At Wrox, we make every effort to ensure that there are no errors in the text or in the code of our books. However, no one is perfect, and mistakes do occur. If you find an error in one of our books, such as a spelling mistake or faulty piece of code, we would be very grateful for your feedback. By sending in errata, you may save another reader hours of frustration, and at the same time you will be helping us provide even higher-quality information.

To find the errata page for this book, go to `www.wrox.com` and locate the title by using the Search box or one of the title lists. Then, on the book details page, click the Book Errata link. On this page you can view all errata that has been submitted for this book and posted by Wrox editors. A complete book list, including links to each book's errata, is also available at `www.wrox.com/misc-pages/booklist.shtml`.

If you don't spot "your" error on the Book Errata page, go to `www.wrox.com/contact/techsupport.shtml` and complete the form there to send us the error you have found. We'll check the information and, if appropriate, post a message to the book's errata page and fix the problem in subsequent editions of the book.