C++ Cookbook

How to write great code with the latest C++ releases

Wayne Murphy

www.bpbonline.com

First Edition 2024

ISBN: 978-93-55515-377

LIMITS OF LIABILITY AND DISCLAIMER OF WARRANTY

The information contained in this book is true to correct and the best of author's and publisher's knowledge. The author has made every effort to ensure the accuracy of these publications, but publisher cannot be held responsible for any loss or damage arising from any information in this book.

All trademarks referred to in the book are acknowledged as properties of their respective owners but BPB Publications cannot guarantee the accuracy of this information.

Dedicated to

My family and friends
who have supported me,
during this book, and for everything else.

and

My managers at work that believed in me
ever since I first started working in software,
and those along the way.
Thanks ***Murray****,* ***Ed****,* ***Shane****, et al.*

About the Author

Wayne Murphy grew up in Brampton, Ontario, Canada. Graduating from Sheridan College in town. He started work at his first programming job in 1987, and over the years been at many companies, in several different roles, with various technologies. For most of the past 20 years, Wayne has been consulting in his company, Great Leap Forwards Inc. When Wayne is not at work, he is working on some of his own code, and spending time with his family.

About the Reviewers

- **Maxim Chetruşca** is a Software Engineer with almost a decade of professional experience in C++. Throughout his career, he focused on real-time, latency sensitive systems. During his work, he used C++ for the backend of the busiest websites, distributed databases as well as squeezing microseconds from high frequency trading systems. A firm believer of C++ philosophy, he had the chance to use the language versions ranging from 03 to 20 in production, running the code on 4 different platforms. Outside of working hours he spends time learning about new car models with his 2-year-old son.

- **Kris Jusiak** is a passionate Software Engineer with experience across various industries, including telecommunications, gaming, and most recently, finance. He specializes in modern C++ development, with a keen focus on performance and quality. Kris is also an active conference speaker and open-source enthusiast, having created multiple open-source libraries.

- **Chetan Sachdeva** is a Tech Enthusiast, with over 13 years of expertise in C++ and Python in software development. With a dynamic career spanning Automotive, AR/VR, IoT, Android native development, Typography, and Printing RIP. His true passion lies in solving complex problems with data structures and algorithms. His entrepreneurial spirit has led to the successful design of software and products for startups. Beyond his professional endeavors, Chetan is an avid reader of non-fiction and IT-related books, staying at the forefront of industry trends.

Acknowledgement

I remember that warm summer day in the mid-1970s, when my father's father told me *You should get into computers*, and me not knowing what one was. I remember my dad taking me into his work, because they just got their first computer there. I later recall when my parents got us our first family computer, I thought to myself, *I will never be bored again*. I have to thank them for that, because I have not been bored yet.

I am very grateful to all those publishing material (books, web-pages, videos) in the goal of wanting people to learn what they know. I hope I have done your efforts justice.

My gratitude also goes to the team at BPB Publications, for giving me a chance to do this.

Preface

C++ Cookbook walks you through all the recent new features. In a cookbook-type style that talks about each new class or function, it shows you in simple terms how to use it. Authored by a software professional, with over 3 decades of experience, their passion for coding shows in the book. This book takes the reader on a tour of what they will need to know to be up to date with the latest C++ abilities.

The book outlines new features, with lots of code examples. They can use it as a reference guide, or progress through each how-to recipe to maximize their knowledge. Sometimes it will give suggestions on the best approach, but mainly wants to inform you of options, and lets you take the right path for your individual situation. The book should help those that have not kept up to date with recent C++ releases. Whether your company was sticking to an older standard, or you are starting with a new product. There are a lot of great new features available, and this book will help you working with the latest and greatest functionality. You can walk through all the chapters with their code examples, or use it as a quick-reference guide for something specific.

After reading it, you will be up to date and will make you and your project work better.

Chapter 1: Working with Concepts – It defines what a concept is, how to use one, and how to create your own. It discusses the different styles how concepts are defined; whether creating a single constraint on a variable or a function, making a requires clause, or combining concepts to make a conjunction or compound requirement.

Chapter 2: Using the New Core Language Concepts – The chapter initiates a long journey to learn what existing concepts are included in C++. There are concepts for comparison, assignment, checking on the type, and hierarchy.

Chapter 3: Using the New Comparison Concepts – The chapter continues delving into predefined concepts. We look at those relating to equality, and comparison.

Chapter 4: Using the New Iterator Concepts – Our next step into concepts looks at all of the concepts defined for iterators, and also talks about the progression of functionality with types of iterators. We also touch on some other concepts relating to moving or copying values.

Chapter 5: Using the New Object Concepts – We pick up where we left off, talking about concepts relating to the constraints of moving or copying values. We also discuss the similarities and differences between the concepts semiregular and regular.

Chapter 6: Using the New Callable Concepts – We will talk about concepts relating functions, or other mechanisms that we can call to execute some code.

Chapter 7: Const Related Specifiers – With this chapter, we end our lengthy talk about concepts and discuss specifiers relating to defining what is constant. We go over previously existing specifiers, and talk about newer ones that will help your code be more efficient.

Chapter 8: Concurrent Processing – The chapter discusses threads, callbacks, and different ways for your code to be executed safely by different threads at the same time.

Chapter 9: Coroutines – The chapter presents a new mechanism for executing code. We get into the details of the parts of a coroutine, how to create them, and how to use them.

Chapter 10: Organizing Your Code with Modules – The chapter covers a great new addition to the language. Modules will help you organize your code easier. We get into the details of its components and go through how to using other modules, and learn how to write our own code.

Chapter 11: Introduction to Ranges and Views – The chapter starts discussion about what ranges, views, and spans are. The differences between them, and how to use them are discussed. Since ranges relate to concepts, we will talk about some of the existing range concepts.

Chapter 12: Range Access and Non-Modifying Sequence Functions for Ranges – The chapter continues to discuss ranges, and some of the functionality relating to subranges, sizes, iterating, and comparing.

Chapter 13: Range Algorithms: Sort, Search and More – The next leg of our journey talks about ranges deals with searching and sorting. We can perform functions on a range to get a value, or even a new range by doing a permutation.

Chapter 14: Range Algorithms: Memory and Modification Functions – We focus on memory; moving values around, and doing transformations.

Chapter 15: Views and Range Adaptors – The chapter talks about the many functions that exist so we can get values in a view by calling a range with an adaptor.

Chapter 16: Range Factories and Utilities – We conclude our talk about ranges by showing some exiting new functionality. We can create an infinite view of values, whether by using a simple function or a generator. There is code showing how to even format your range for output.

Chapter 17: New Features for Containers – The chapter covers a lot of ground, dealing with new container types, and simple new functions.

Chapter 18: Making it Easier to Code – There are a number of new features discussed that you will be happy to see such as easier output, new enhancements for strings and ranges, plus an easier way to use enums and bits. There is also new types for dates, times and time zones.

Chapter 19: Making Your Code Cleaner – The chapter covers many recent features for lambdas, new suffixes, optional and expected arguments, plus new preprocessor directives for cleaner code.

Chapter 20: Making Your Code Safer – The chapter touches on some important new features to make code less fragile.

Chapter 21: Making Your Code Faster and Easier to Debug – We conclude by talking about ways to make your code run faster; including some new functions, and new attributes. We also talk about new types that could help you debug your code.

Code Bundle and Coloured Images

Please follow the link to download the
Code Bundle and the ***Coloured Images*** of the book:

https://rebrand.ly/de3g86d

The code bundle for the book is also hosted on GitHub at
https://github.com/bpbpublications/C-Plus-Plus-Cookbook.
In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **https://github.com/bpbpublications**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Welcome to C++

C++ is a wonderful language. You can often be working with high-level abstractions, but still be able to dig down deep and work with raw memory. C++ has been around for decades, and with the efforts of many good people out there, is still evolving. We can assume that this means that it will probably be around for decades more. It is on lists of most used languages (sometimes in the top 10, and if not, then certainly the top 20), as well C++ is on lists of programmers' most liked languages. C++ is not owned by a big company, has multiple companies making compilers for it, and many companies making tools and libraries for it. It runs on different operating systems, for purposes big and small. The governing body meets regularly to discuss, debate, and decide what will go into the next release. Releases were decided to be every three years, which they have done in the age of modern C++ (11, 14, 17, 20, 23, and they are already talking about C++26). To be precise, the releases are defined every three years. At the time of writing this book, at the end of December, C++23 was still being voted on whether it is officially released.

We could not talk about everything to do with C++ just in one book, and so we are here to talk about the latest couple of releases; C++20 and C++23.

Recipe 1: Who is this book for?

Problem: We want to know who should read this book.

Solution: It can appeal to different types of audiences.

Of course, we would like to say anybody and everybody can read this book. However, this is a cookbook focusing on the newest releases. If you are just starting out knowing little about C++, then this probably is not the best first book for you. Having said that though, today's world lets us learn in different media in different ways. If you have a handful of websites and other learning video channels that teach you the basics, then yes, you can buy the book as a secondary source of learning.

This book does not only discuss the basics but focuses on new features. In that regard, it can be used by any who knows the basics but is not kept up to date with all the latest features. If you look at the following figure, we would say that we are trying to focus on what you will want to know to get caught up on the latest features.

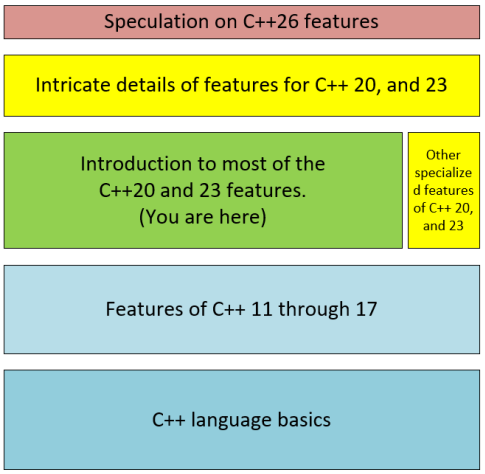

Figure 1: *What this book is about*

Recipe 2: What is the format for each chapter?

Problem: How is each chapter planned out?

Solution: It is a cookbook-type design, with many problem and solution topics for various Recipes.

Each chapter has a particular focus:

- We have an **Introduction**, or overview of what the chapter is about.
- The **Structure** is given, which lists the features or topics that we will talk about in detail.
- We have a brief statement on the **Objectives** of the chapter.

- Then, we iterate over our **Recipes**, which are individual topics in the chapter. Sometimes, a recipe can be about an idea, but often it is about a particular class or function. There can be between 5-15 recipes in a chapter. We have tried to provide lots of code samples to help illustrate what the recipe is about.
- Next, we give a **Conclusion**, which summarizes what you have learned in the chapter.
- We give reminders of some important facets in the **Points to remember** section.
- Lastly, we provide **References**, aside from needing to say where the information came from, which is also a huge and sincere thank-you to those who provided information and helped inform others. There are probably more resources that we could have mentioned. For brevity's sake, if we saw similar information on a handful of different websites and videos, then we jotted down what we thought best presented the information.

Recipe 3: Is C++23 worth reading about?

Problem: If there have been so many releases, can we just learn it as we go?

Solution: You probably can learn a bit as you go, however the good and bad part about C++, is that it has evolved over the decades. So yes, you could continue using nothing but new and delete, and maybe you could get by and use a few new classes as you trip over them. The result though, is that the code, and coder, will not be as good as they could be. Some things get deprecated, but a lot of code is still around due to historical use. Everybody says you should not use function `A()`, but now use function `B()`. And in truth, we are probably on function `D()` in some cases. You should want to at least be aware of the latest features. You may not want to, or need to, rewrite your entire code-base because new features exist, but for new functionality, you should understand what is in the latest release(s), to make an informed decision on when to use them. Compilers are smarter, computers are faster, learning tools are better, and so using newer features will not make your job more difficult. Releases are created based on input like yours, to make coding with new features safer, smarter, faster, and easier.

To get into slightly more detail, the C++20 release was huge. It was arguably the biggest C++ update ever in terms of features. What is more, it helped to define a direction on where the language should head. C++20's *Big Four* features will influence other features for years to come. C++23 is not as big as C++20, partially, because of the global pandemic, and admitted partially because there were multiple changes from C++20. Having said that, C++23 is an impressive release on its own merit, which will change how coders create software.

Refer to the following figure:

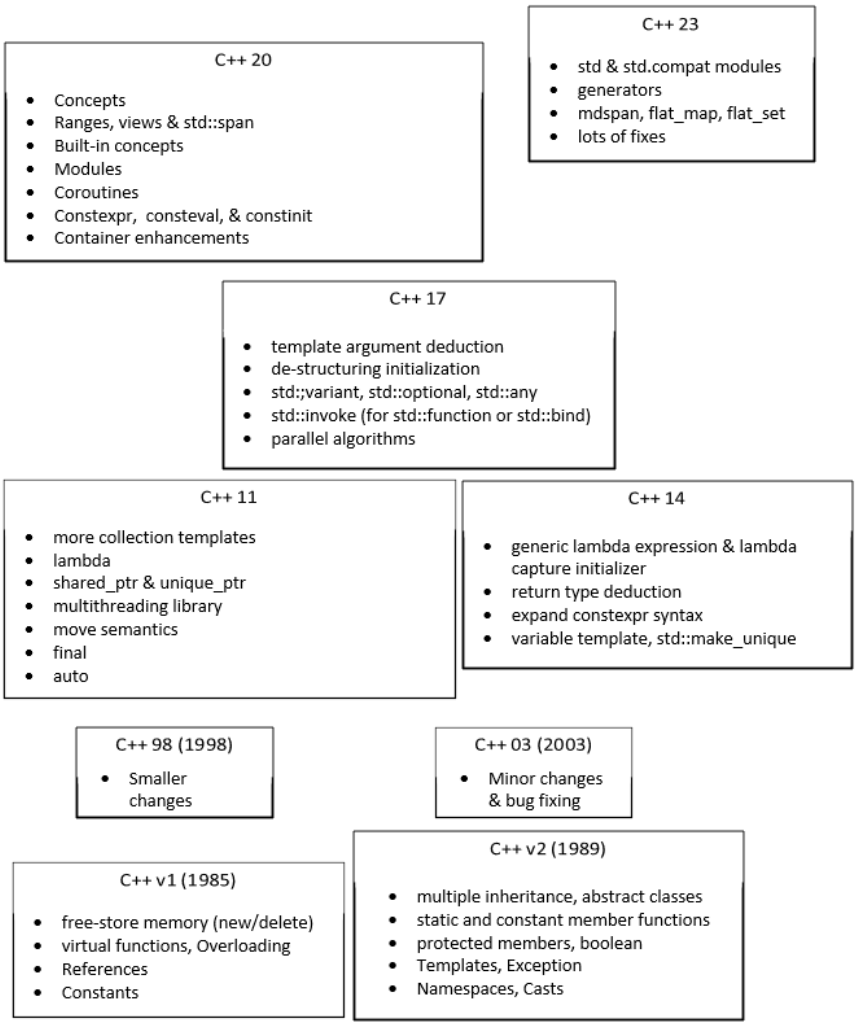

Figure 2: *Overview of C++ releases*

Recipe 4: Is the book just about C++23?

Problem: We want to learn more about the latest version, but have not focused too much on other recent releases.

Solution: This book talks wants you to learn the latest, but understands if you might have missed something along the way.

The book has tried to include everything that has been defined in the C++23 release. A couple of features might be vague or short, partially because some features are more important than others, but also a couple of C++23 features are still a bit vague. They are included in the release, but no one has implemented the feature, and there is not much information about them outside of the spec. As we have said, C++20 was such an important release, that some parts of C++23 are tough to understand if you do not know about the related C++20 features. We do talk quite a bit about the C++20 release, and even some parts of the C++17, but that is not to distract you from learning about C++23, but here to help you understand how we got to C++23.

Recipe 5: How to work with the code

Problem: How do you use the code samples provided?

Solution: We can walk you through some of that.

Most of the code samples were built and run in Visual Studio 2022. We have used Visual Studio Code as well. There are two main steps to use C++23 in Visual Studio. The first is to run the Visual Studio Installer program and enable using the *Latest* C++ tools. Refer to the following figure:

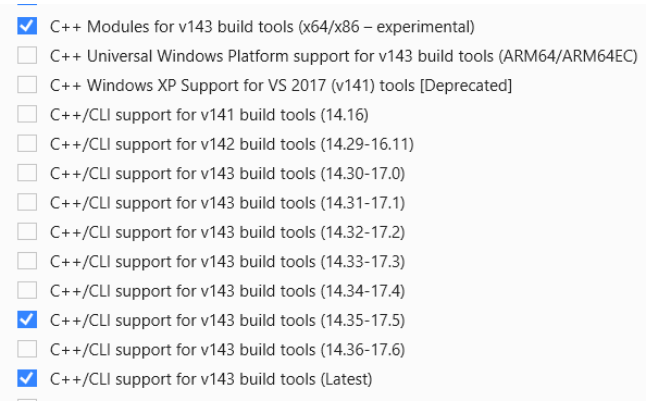

Figure 3: *Showing we are using the "Latest" build tools, in the Visual Studio Installer*

The next is to go into your project settings and select if you want to use the *Preview* release of the C++ Standard. This would equate to having /std:c++latest in your makefile. Refer to the following figure:

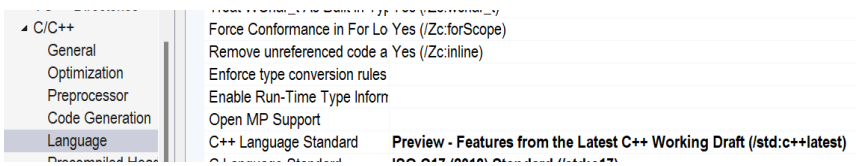

Figure 4: *Visual Studio project setting to use the latest C++ standard*