Contents

Preface to the Second Edition

I never thought that the first edition of this book would sell so long. But now, after twelve years, it's time for a new edition that covers C++11, the new C++ standard.

Note that this means more than simply adding new libraries. C++ has changed. Almost all typical applications of parts of the library look a bit different now. This is not the result of a huge language change. It's the result of many minor changes, such as using rvalue references and move semantics, range-based `for` loops, `auto`, and new template features. Thus, besides presenting new libraries and supplementary features of existing libraries, almost all of the examples in this book were rewritten at least partially. Nevertheless, to support programmers who still use "old" C++ environments, this book will describe differences between C++ versions whenever they appear.

I learned C++11 the hard way. Because I didn't follow the standardization as it was happening I started to look at C++11 about two years ago. I really had trouble understanding it. But the people on the standardization committee helped me to describe and present the new features as they are intended to be used now.

Note, finally, that this book now has a problem: Although the book's size grew from about 800 to more than 1,100 pages, I still can't present the C++ standard library as a whole. The library part of the new C++11 standard alone now has about 750 pages, written in very condensed form without much explanation. For this reason, I had to decide which features to describe and in how much detail. Again, many people in the C++ community helped me to make this decision. The intent was to concentrate on what the average application programmer needs. For some missing parts, I provide a supplementary chapter on the Web site of this book, `http://www.cppstdlib.com`, but you still will find details not mentioned here in the standard.

The art of teaching is not the art of presenting everything. It's the art of separating the wheat from the chaff so that you get the most out of it. May the exercise succeed.

Acknowledgments for the Second Edition

This book presents ideas, concepts, solutions, and examples from many sources. Over the past several years, the C++ community introduced many ideas, concepts, proposals, and enhancements to C++ that became part of C++11. Thus, again I'd like to thank all the people who helped and supported me while preparing this new edition.

First, I'd like to thank everyone in the C++ community and on the C++ standardization committee. Besides all the work to add new language and library features, they had a hard time explaining everything to me, but they did so with patience and enthusiasm.

Scott Meyers and Anthony Williams allowed me to use their teaching material and book manuscripts so that I could find many useful examples not yet publicly available.

I'd also like to thank everyone who reviewed this book and gave valuable feedback and clarifications: Dave Abrahams, Alberto Ganesh Barbati, Pete Becker, Thomas Becker, Hans Boehm, Walter E. Brown, Paolo Carlini, Lawrence Crowl, Beman Dawes, Doug Gregor, David Grigsby, Pablo Halpern, Howard Hinnant, John Lakos, Bronek Kozicki, Dietmar Kühl, Daniel Krügler, Mat Marcus, Jens Maurer, Alisdair Meredith, Bartosz Milewski, P. J. Plauger, Tobias Schüle, Peter Sommerlad, Jonathan Wakely, and Anthony Williams.

There is one person who did an especially outstanding job. Whenever I had a question, Daniel Krügler answered almost immediately with incredible accurateness and knowledge. Everyone in the standardization process know that he treats everybody this way. Without him, both the C++ standard and this book would not have the quality they have now.

Many thanks to my editor Peter Gordon, Kim Boedigheimer, John Fuller, and Anna Popick from Addison-Wesley. Besides their support, they found the right balance between patience and pressure. The copy editor Evelyn Pyle and the proofreader Diane Freed did an incredible job translating my German English into American English. In addition, thanks to Frank Mittelbach for solving my LaTeX issues.

Last but not least, all my thanks go to Jutta Eckstein. Jutta has the wonderful ability to force and support people in their ideals, ideas, and goals. While most people experience this only when working with her, I have the honor to benefit in my day-to-day life.

Preface to the First Edition

In the beginning, I only planned to write a small German book (400 pages or so) about the C++ standard library. That was in 1993. Now, in 1999 you see the result — a book in English with more than 800 pages of facts, figures, and examples. My goal is to describe the C++ standard library so that all (or almost all) your programming questions are answered before you think of the question. Note, however, that this is not a complete description of all aspects of the C++ standard library. Instead, I present the most important topics necessary for learning and programming in C++ by using its standard library.

Each topic is described based on the general concepts; this discussion then leads to the specific details needed to support everyday programming tasks. Specific code examples are provided to help you understand the concepts and the details.

That's it — in a nutshell. I hope you get as much pleasure from reading this book as I did from writing it. Enjoy!

Acknowledgments for the First Edition

This book presents ideas, concepts, solutions, and examples from many sources. In a way it does not seem fair that my name is the only name on the cover. Thus, I'd like to thank all the people and companies who helped and supported me during the past few years.

First, I'd like to thank Dietmar Kühl. Dietmar is an expert on C++, especially on input/output streams and internationalization (he implemented an I/O stream library just for fun). He not only translated major parts of this book from German to English, he also wrote sections of this book using his expertise. In addition, he provided me with invaluable feedback over the years.

Second, I'd like to thank all the reviewers and everyone else who gave me their opinion. These people endow the book with a quality it would never have had without their input. (Because the list is extensive, please forgive me for any oversight.) The reviewers for the English version of this book included Chuck Allison, Greg Comeau, James A. Crotinger, Gabriel Dos Reis, Alan Ezust, Nathan Myers, Werner Mossner, Todd Veldhuizen, Chichiang Wan, Judy Ward, and Thomas Wikehult. The German reviewers included Ralf Boecker, Dirk Herrmann, Dietmar Kühl, Edda Lörke, Herbert Scheubner, Dominik Strasser, and Martin Weitzel. Additional input was provided by Matt Austern, Valentin Bonnard, Greg Colvin, Beman Dawes, Bill Gibbons, Lois Goldthwaite, Andrew Koenig, Steve Rumsby, Bjarne Stroustrup, and David Vandevoorde.

Special thanks to Dave Abrahams, Janet Cocker, Catherine Ohala, and Maureen Willard who reviewed and edited the whole book very carefully. Their feedback was an incredible contribution to the quality of this book.

A special thanks goes to my "personal living dictionary" — Herb Sutter — the author of the famous "Guru of the Week" (a regular series of C++ programming problems that is published on the `comp.lang.c++.moderated` Internet newsgroup).

I'd also like to thank all the people and companies who gave me the opportunity to test my examples on different platforms with different compilers. Many thanks to Steve Adamczyk, Mike Anderson, and John Spicer from EDG for their great compiler and their support. It was a big help during the standardization process and the writing of this book. Many thanks to P. J. Plauger and Dinkumware, Ltd, for their early standard-conforming implementation of the C++ standard library. Many thanks to Andreas Hommel and Metrowerks for an evaluative version of their CodeWarrior Programming Environment. Many thanks to all the developers of the free GNU and egcs compilers. Many thanks to Microsoft for an evaluative version of Visual C++. Many thanks to Roland Hartinger

from Siemens Nixdorf Informations Systems AG for a test version of their C++ compiler. Many thanks to Topjects GmbH for an evaluative version of the ObjectSpace library implementation.

Many thanks to everyone from Addison Wesley Longman who worked with me. Among others this includes Janet Cocker, Mike Hendrickson, Debbie Lafferty, Marina Lang, Chanda Leary, Catherine Ohala, Marty Rabinowitz, Susanne Spitzer, and Maureen Willard. It was fun.

In addition, I'd like to thank the people at BREDEX GmbH and all the people in the C++ community, particularly those involved with the standardization process, for their support and patience (sometimes I ask really silly questions).

Last but not least, many thanks and kisses for my family: Ulli, Lucas, Anica, and Frederic. I definitely did not have enough time for them due to the writing of this book.

Have fun and be human!

Chapter 1

About This Book

1.1 Why This Book

Soon after its introduction, C++ became a de facto standard in object-oriented programming. This led to the goal of standardization. Only by having a standard could programs be written that would run on different platforms — from PCs to mainframes. Furthermore, a standard *library* would enable programmers to use general components and a higher level of abstraction without losing portability rather than having to develop all code from scratch.

Now, with the second standard, called C++11 (see Section 2.1, page 7, for the detailed history of C++ standards), we have a huge C++ standard library whose specification requires more than double the size of the core language features. The library enables the use of

- Input/output (I/O) classes
- String types and regular expressions
- Various data structures, such as dynamic arrays, linked lists, binary trees, and hash tables
- Various algorithms, such as a variety of sorting algorithms
- Classes for multithreading and concurrency
- Classes for internationalization support
- Numeric classes
- Plenty of utilities

However, the library is not self-explanatory. To use these components and to benefit from their power, you need an introduction that explains the concepts and the important details instead of simply listing the classes and their functions. This book is written exactly for that purpose. First, it introduces the library and all its components from a conceptual point of view. Next, the book describes the details needed for practical programming. Examples are included to demonstrate the exact use of the components. Thus, this book is a detailed introduction to the C++ library for both the beginner and the practicing programmer. Armed with the data provided herein, you should be able to take full advantage of the C++ standard library.

Caveat: I don't promise that everything described is easy and self-explanatory. The library provides a lot of flexibility, but flexibility for nontrivial purposes has a price. The library has traps and pitfalls, which I point out when we encounter them and suggest ways of avoiding them.

1.2 Before Reading This Book

To get the most from this book, you should already know C++. (The book describes the standard components of C++ but not the language itself.) You should be familiar with the concepts of classes, inheritance, templates, exception handling, and namespaces. However, you don't have to know all the minor details about the language. The important details are described in the book; the minor details about the language are more important for people who want to implement the library rather than to use it.

Note that the language has changed during the standardization of C++11, just as it changed during the standardization of C++98, so your knowledge might not be up-to-date. Chapter 3 provides a brief overview of and introduction to the latest language features that are important for using the C++11 library. Many of the new library features use these new language features, so you should read Chapter 3 to review all the new features of C++. But I will also refer to that chapter when libraries use new language features.

1.3 Style and Structure of the Book

The C++ standard library provides components that are somewhat, but not totally, independent of one another, so there is no easy way to describe each part without mentioning others. I considered various approaches for presenting the contents of this book. One was on the order of the C++ standard. However, this is not the best way to explain the components of the C++ standard library from scratch. Another approach was to start with an overview of all components, followed by chapters that provided more details. Alternatively, I could have sorted the components, trying to find an order that had a minimum of cross-references to other sections. My solution was to use a mixture of all three approaches. I start with a brief introduction of the general concepts and the utilities that the library uses. Then, I describe all the components, each in one or more chapters. The first component is the standard template library (STL). There is no doubt that the STL is the most powerful, most complex, and most exciting part of the library. Its design influences other components heavily. Then, I describe the more self-explanatory components, such as special containers, strings, and regular expressions. The next component discussed is one you probably know and use already: the IOStream library. That component is followed by a discussion of internationalization, which had some influence on the IOStream library. Finally, I describe the library parts dealing with numerics, concurrency, and allocators.

Each component description begins with the component's purpose, design, and some examples. Next, a detailed description begins with various ways to use the component, as well as any traps and pitfalls associated with it. The description usually ends with a reference section, in which you can find the exact signature and definition of a component's classes and its functions.

List of Contents

The first five chapters introduce this book and the C++ standard library in general:

- **Chapter 1: About This Book** introduces the book's subject and describes its contents.
- **Chapter 2: Introduction to C++ and the Standard Library** provides a brief overview of the history of the C++ standard library and the context of its standardization and introduces the concept of complexity.
- **Chapter 3: New Language Features** provides an overview of the new language features you should know to read this book and to use the C++ standard library.
- **Chapter 4: General Concepts** describes the fundamental library concepts that you need to understand to work with all the components. In particular, the chapter introduces the namespace `std`, the format of header files, and the general support of error and exception handling.
- **Chapter 5: Utilities** describes several small utilities provided for the user of the library and for the library itself. In particular, the chapter describes classes `pair<>` and `tuple<>`, smart pointers, numeric limits, type traits and type utilities, auxiliary functions, class `ratio<>`, clocks and timers, and available C functions.

Chapters 6 through 11 describe all aspects of the STL:

- **Chapter 6: The Standard Template Library** presents a detailed introduction to the concept of the STL, which provides container classes and algorithms that are used to process collections of data. The chapter explains step-by-step the concept, the problems, and the special programming techniques of the STL, as well as the roles of its parts.
- **Chapter 7: STL Containers** explains the concepts and describes the abilities of the STL's container classes. The chapter describes arrays, vectors, deques, lists, forward lists, sets, maps, and unordered containers with their common abilities, differences, specific benefits, and drawbacks and provides typical examples.
- **Chapter 8: STL Container Members in Detail** lists and describes all container members (types and operations) in the form of a handy reference.
- **Chapter 9: STL Iterators** explains the various iterator categories, the auxiliary functions for iterators, and the iterator adapters, such as stream iterators, reverse iterators, insert iterators, and move iterators.
- **Chapter 10: STL Function Objects and Using Lambdas** details the STL's function object classes, including lambdas, and how to use them to define the behavior of containers and algorithms.
- **Chapter 11: STL Algorithms** lists and describes the STL's algorithms. After a brief introduction and comparison of the algorithms, each algorithm is described in detail, followed by one or more example programs.

Chapters 12 through 14 describe "simple" individual standard classes of the C++ standard library:

- **Chapter 12: Special Containers** describes the container adapters for queues and stacks, as well as the class `bitset`, which manages a bitfield with an arbitrary number of bits or flags.
- **Chapter 13: Strings** describes the string types of the C++ standard library (yes, there are more than one). The standard provides strings as "kind of" fundamental data types with the ability to use different types of characters.

- **Chapter 14: Regular Expressions** describes the interface to deal with regular expressions, which can be used to search and replace characters and substrings.

Chapters 15 and 16 deal with the two closely related subjects of I/O and internationalization:

- **Chapter 15: Input/Output Using Stream Classes** covers the standardized form of the commonly known IOStream library. The chapter also describes details that are typically not so well known but that may be important to programmers, such as the correct way to define and integrate special I/O channels.
- **Chapter 16: Internationalization** covers the concepts and classes for the internationalization of programs, such as the handling of different character sets and the use of different formats for floating-point numbers and dates.

The remaining chapters cover numerics, concurrency, and allocators:

- **Chapter 17: Numerics** describes the numeric components of the C++ standard library: in particular, classes for random numbers and distributions, types for complex numbers, and some numeric C functions.
- **Chapter 18: Concurrency** describes the features provided by the C++ standard library to enable and support concurrency and multithreading.
- **Chapter 19: Allocators** describes the concept of different memory models in the C++ standard library.

The book concludes with a **bibliography** and an **index**.

Due to the size of this book I had to move material that is not so relevant for the average application programmer but should be covered to a **supplementary chapter** provided on the Web site of this book: `http::/www.cppstdlib.com`. That material includes:

- Details of bitsets (introduced in Section 12.5)
- Class `valarray<>` (very briefly introduced in Section 17.4)
- Details of allocators (introduced in Chapter 19)

1.4 How to Read This Book

This book is both an introductory user's guide and a structured reference manual about the C++ standard library. The individual components of the C++ standard library are somewhat independent of one another, so after reading Chapters 2 through 5 you could read the chapters that discuss the individual components in any order. Bear in mind that Chapters 6 through 11 all describe the same component. To understand the other STL chapters, you should start with the introduction to the STL in Chapter 6.

If you are a C++ programmer who wants to know, in general, the concepts and all parts of the library, you could simply read the book from beginning to end. However, you should skip the reference sections. To program with certain components of the C++ standard library, the best way to find something is to use the index, which I have tried to make comprehensive enough to save you time when you are looking for something.

In my experience, the best way to learn something new is to look at examples. Therefore, you'll find a lot of examples throughout the book. They may be a few lines of code or complete programs.

In the latter case, you'll find the name of the file containing the program as the first comment line. You can find the files on the Internet at the Web site of the book: `http://www.cppstdlib.com`.

1.5 State of the Art

The C++11 standard was completed while I was writing this book. Please bear in mind that some compilers might not yet conform to the standard. This will most likely change in the near future. As a consequence, you might discover that not all things covered in this book work as described on your system, and you may have to change example programs to fit your specific environment.

1.6 Example Code and Additional Information

You can access all example programs and acquire more information about this book and the C++ standard library from my Web site: `http://www.cppstdlib.com`. Also, you can find a lot of additional information about this topic on the Internet. See the bibliography, which is also provided on the Web site, for some of them.

1.7 Feedback

I welcome your feedback (good and bad) on this book. I tried to prepare it carefully; however, I'm human, and at some point I have to stop writing and tweaking. So, you may find some errors, inconsistencies, or subjects that could be described better. Your feedback will give me the chance to improve later editions.

The best way to reach me is by email. However, to avoid spam problems, I haven't included an email address inside this book. (I had to stop using the email address I put in the first edition after I started getting thousands of spam emails per day.) Please refer to the book's Web site, `http://www.cppstdlib.com`, to get an email address for feedback.

Many thanks.