Contents

Preface

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 — an English book 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 every-day 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

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 fogive 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 Meyers, 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 Rumbsby, 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.std.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.

The standardization process was started in 1989 by an international ANSI/ISO committee. It developed the standard based on Bjarne Stroustrup's books *The C++ Programming Language* and *The Annotated C++ Reference Manual*. After the standard was completed in 1997, several formal motions by different countries made it an international ISO and ANSI standard in 1998. The standardization process included the development of a C++ standard library. The library extends the core language to provide some general components. By using C++'s ability to program new abstract and generic types, the library provides a set of common classes and interfaces. This gives programmers a higher level of abstraction. The library provides the ability to use

- String types
- Different data structures (such as dynamic arrays, linked lists, and binary trees)
- Different algorithms (such as different sorting algorithms)
- Numeric classes
- Input/output (I/O) classes
- Classes for internationalization support

All of these are supported by a fairly simple programming interface. These components are very important for many programs. These days, data processing often means inputting, computing, processing, and outputting large amounts of data, which are often strings.

The library is not self-explanatory. To use these components and to benefit from their power, you need a good 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 of its components from a conceptional point of view. Next, it describes the details needed for practical programming. Examples are included to demonstrate the exact usage of the components. Thus, this book is a detailed introduction to the C++ library for both the beginner and the practical 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. Beware that the library has traps and pitfalls, which I point out when we encounter them and suggest ways of avoiding them.

1.2 What You Should Know 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, and exception handling. However, you don't have to know all of 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 use it). Note that the language has changed during the standardization process, so your knowledge might not be up to date. Section 2.2, page 9, provides a brief overview and introduction of the latest language features that are important for using the library. You should read this section if you are not sure whether you know all the new features of C++ (such as the keyword `typename` and the concept of namespaces).

1.3 Style and Structure of the Book

The C++ standard library provides different components that are somewhat but not totally independent of each other, so there is no easy way to describe each part without mentioning others. I considered several different 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 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 are used by the library. 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 numeric classes. The next component discussed is one you probably know and use already: the IOStream library. It is followed by a discussion of internationalization, which had some influence on the IOStream library.

Each component description begins with the component's purpose, design, and some examples. Next, a detailed description follows that begins with different 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.

The following is a description of the book's contents. The first four chapters introduce this book and the C++ standard library in general:

- **Chapter 1: About this Book**
  This chapter (which you are reading right now) introduces the book's subject and describes its contents.
- **Chapter 2: Introduction to C++ and the Standard Library**
  This chapter provides a brief overview of the history of the C++ standard library and the context of its standardization. It also contains some general hints regarding the technical background for this book and the library, such as new language features and the concept of complexity.
- **Chapter 3: General Concepts**
  This chapter describes the fundamental concepts of the library that you need to understand to work with all the components. In particular, it introduces the namespace `std`, the format of header files, and the general support of error and exception handling.
- **Chapter 4: Utilities**
  This chapter describes several small utilities provided for the user of the library and for the library itself. In particular, it describes auxiliary functions such as `max()`, `min()`, and `swap()`, types `pair` and `auto_ptr`, as well as `numeric_limits`, which provide more information about implementation-specific details of numeric data types.

Chapters 5 through 9 describe all aspects of the STL:

- **Chapter 5: The Standard Template Library**
  This chapter presents a detailed introduction to the concept of the STL, which provides container classes and algorithms that are used to process collections of data. It 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 6: STL Containers**
  This chapter explains the concepts and describes the abilities of the STL's container classes. First it describes the differences between vectors, deques, lists, sets, and maps, then their common abilities, and all with typical examples. Lastly it lists and describes all container functions in form of a handy reference.
- **Chapter 7: STL Iterators**
  This chapter deals in detail with the STL's iterator classes. In particular, it explains the different iterator categories, the auxiliary functions for iterators, and the iterator adapters, such as stream iterators, reverse iterators, and insert iterators.
- **Chapter 8: STL Function Objects**
  This chapter details the STL's function object classes.
- **Chapter 9: STL Algorithms**
  This chapter 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 10 through 12 describe "simple" individual standard classes:

- **Chapter 10: Special Containers**
  This chapter describes the different special container classes of the C++ standard library. It covers 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 11: Strings**
  This chapter describes the string types of the C++ standard library (yes, there are more than one). The standard provides strings as kind of "self-explanatory" fundamental data types with the ability to use different types of characters.
- **Chapter 12: Numerics**
  This chapter describes the numeric components of the C++ standard library. In particular, it covers types for complex numbers and classes for the processing of arrays of numeric values (the latter may be used for matrices, vectors, and equations).

Chapters 13 and 14 deal with I/O and internationalization (two closely related subjects):

- **Chapter 13: Input/Output Using Stream Classes**
  This chapter covers the I/O component of C++. This component is the standardized form of the commonly known IOStream library. The chapter also describes details that may be important to programmers but are typically not so well known. For example, it describes the correct way to define and integrate special I/O channels, which are often implemented incorrectly in practice.
- **Chapter 14: Internationalization**
  This chapter covers the concepts and classes for the internationalization of programs. In particular, it describes the handling of different character sets, as well as the use of different formats for such values as floating-point numbers and dates.

The rest of the book contains:

- **Chapter 15: Allocators**
  This chapter describes the concept of different memory models in the C++ standard library.
- An **appendix** with
  - **Internet Resources**
  - **Bibliography**
  - **Index**

1.4 How to Read this Book

This book is a mix of introductory user's guide and structured reference manual regarding the C++ standard library. The individual components of the C++ standard library are independent of each other, to some extent, so after reading Chapters 2 through 4 you could read the chapters that discuss the individual components in any order. Bear in mind, that Chapter 5 through Chapter 9 all describe the same component. To understand the other STL chapters, you should start with the introduction to the STL in Chapter 5.

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 the beginning to the 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. I have tried to make the index very comprehensive 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 my Web site at `http://www.josuttis.com/libbook/`.

1.5 State of the Art

While I was writing this book, the C++ standard was completed. Please bear in mind that some compilers might not yet confirm to it. 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. I can compile almost all example programs with version 2.8 or higher of the EGCS compiler, which is free for almost all platforms and available on the Internet (see `http://egcs.cygnus.com/`) and on several software CDs.

1.6 Example Code and Additional Information

You can access all example programs and acquire more informations about this book and the C++ standard library from my Web site at `http://www.josuttis.com/libbook/`. Also, you can find a lot of additional information about this topic on the Internet. See Internet Resources on page 743 for details.

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 time 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:

`libbook@josuttis.com`

You can also reach me by phone, fax, or “snail” mail:

Nicolai M. Josuttis
Berggarten 9
D–38108 Braunschweig
Germany

Phone: +49 5309 5747
Fax: +49 5309 5774

Many thanks.