

THE

C++

C++11

PROGRAMMING LANGUAGE

FOURTH EDITION

BJARNE STROUSTRUP

THE CREATOR OF C++

**The
C++
Programming
Language**

Fourth Edition

Bjarne Stroustrup

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special orders. These discounts may include electronic versions and/or custom covers and content particular to your business, training, sales, and marketing interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States, please contact:
International Sales
international@pearsoned.com

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data

Stroustrup, Bjarne.

The C++ programming language / Bjarne Stroustrup.—Fourth edition.
pages cm

Includes bibliographical references and index.

ISBN 978-0-321-56384-2 (pbk. : alk. paper)—ISBN 0-321-56384-0 (pbk. : alk. paper)

1. C++ (Computer programming language) I. Title.

QA76.73.C153 S77 2013

005.13'3—dc23

2013002159

Copyright © 2013 by Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and any reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise, without the prior written permission of the publisher is prohibited. To obtain permission for reproduction or translation, contact the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. For more information, contact www.copyright.com.

Contents

Preface

Preface to the Fourth Edition	v
Preface to the Third Edition	ix
Preface to the Second Edition	xi
Preface to the First Edition	xii

Part I: Introductory Material

1. Notes to the Reader	3
2. A Tour of C++: The Basics	37
3. A Tour of C++: Abstraction Mechanisms	59
4. A Tour of C++: Containers and Algorithms	87

iv Contents

11. Select Operations	273
12. Functions	305
13. Exception Handling	343
14. Namespaces	389
15. Source Files and Programs	419

Part III: Abstraction Mechanisms

16. Classes	449
17. Construction, Cleanup, Copy, and Move	481
18. Overloading	527
19. Special Operators	549
20. Derived Classes	577
21. Class Hierarchies	613
22. Run-Time Type Information	641
23. Templates	665
24. Generic Programming	699
25. Specialization	721
26. Instantiation	741
27. Templates and Hierarchies	759
28. Metaprogramming	779
29. A Matrix Design	827

Part IV: The Standard Library

30. Standard Library Summary	859
31. STL Containers	885
32. STL Algorithms	927
33. STL Iterators	953
34. Memory and Resources	973
35. Utilities	1009
36. Strings	1033
37. Regular Expressions	1051
38. I/O Streams	1073

I

*All problems in computer science
can be solved by another level of abstraction,
except for the problem of too many layers of abstraction.*

– Dan Abbot

C++ feels like a new language. That is, I can express my ideas more clearly, more directly in C++11 than I could in C++98. Furthermore, the resulting programs are checked by the compiler and run faster.

In this book, I aim for *completeness*. I describe every language feature and its component that a professional programmer is likely to need. For each, I provide:

- *Rationale*: What kinds of problems is it designed to help solve? What principles guided the design? What are the fundamental limitations?
- *Specification*: What is its definition? The level of detail is chosen for the expert; the aspiring language lawyer can follow the many references to the ISO standard.
- *Examples*: How can it be used well by itself and in combination with other features? What are the key techniques and idioms? What are the implications for maintainability and performance?

The use of C++ has changed dramatically over the years and so has the language itself.

techniques for writing quality code. C++ is a language for someone who takes the task seriously. Our civilization depends critically on software; it had better be quality.

There are billions of lines of C++ deployed. This puts a premium on stability: 1995 C++ code still works and will continue to work for decades. However, for all that you can do better with modern C++; if you stick to older styles, you will be writing slower and worse-performing code. The emphasis on stability also implies that standard C++ code you write today will still work a couple of decades from now. All code in this book conforms to the 2011 ISO C++ standard.

This book is aimed at three audiences:

- C++ programmers who want to know what the latest ISO C++ standard has to offer.
- C programmers who wonder what C++ provides beyond C, and
- People with a background in application languages, such as Java, C#, Python, etc., looking for something “closer to the machine” – something more flexible, something offering better compile-time checking, or something offering better performance.

Naturally, these three groups are not disjoint – a professional software developer may know just one programming language.

This book assumes that its readers are programmers. If you ask, “What’s a compiler?” or “What’s a compiler?” then this book is not (yet) for you; instead, I recommend my book *Principles and Practice Using C++* to get started with programming and C++. I assume that readers have some maturity as software developers. If you ask “Why bother?” or say, “All languages are basically the same; just show me the syntax” or are convinced that there is a single language that is ideal for every task, this is not the book for you.

What features does C++11 offer over and above C++98? A machine model suitable for multi-processor computers with lots of concurrency. Language and standard-library facilities for doing high-level concurrent programming (e.g., using multicores). Regular expression handling, atomic management pointers, random numbers, improved containers (including, hash tables). General and uniform initialization, a simpler `for`-statement, move semantics, basic Uniform Initialization, lambdas, general constant expressions, control over class defaults, variadic template functions, literals, and more. Please remember that those libraries and language features exist to support modern programming techniques for developing quality software. They are meant to be used in combination, as bricks in a building set – rather than to be used individually in relative isolation to solve a specific problem. A computer is a universal machine, and C++ serves it in that capacity. C++’s design aims to be sufficiently flexible and general to cope with future problems posed by its designers.

Acknowledgments

In addition to the people mentioned in the acknowledgment sections of the previous editions, I would like to thank Pete Becker, Hans-J. Boehm, Marshall Clow, Jonathan Coe, Larry DeRose, Walter Daugherty, J. Daniel Garcia, Robert Harle, Greg Hickman, Howard Hinnant, Kernighan, Daniel Krüger, Nevin Liber, Michel Michaud, Gary Powell, Jan Christiaan Reinders, and Leor Zolman. Without their help this book would have been much poorer.

Thanks to Howard Hinnant for answering many questions about the standard library.

Andrew Sutton is the author of the Origin library, which was the testbed for much of the discussion of emulating concepts in the template chapters, and of the matrix library that is discussed in Chapter 29. The Origin library is open source and can be found by searching the Web for “Andrew Sutton.”

Thanks to my graduate design class for finding more problems with the “tour of the library” than anyone else.

Had I been able to follow every piece of advice of my reviewers, the book would have been much improved, but it would also have been hundreds of pages longer. Some reviewers suggested adding technical details, advanced examples, and many useful conventions; every novice reviewer (or educator) suggested adding examples; and one reviewer observed (correctly) that the book may be too long.

Thanks to Princeton University’s Computer Science Department, and especially Kernighan, for hosting me for part of the sabbatical that gave me time to write this book.

Thanks to Cambridge University’s Computer Lab, and especially Prof. Andy Hertzberg, for hosting me for part of the sabbatical that gave me time to write this book.

Thanks to my editor, Peter Gordon, and his production team at Addison-Wesley for their patience.

College Station, Texas

Bjarne Stroustrup

This page intentionally left blank

Preface to the Third Edition

Programming is fun
— K. R. M. Martin

I find using C++ more enjoyable than ever. C++'s support for design and programming has improved dramatically over the years, and lots of new helpful techniques have been introduced into its use. However, C++ is not *just* fun. Ordinary practical programmers have achieved significant improvements in productivity, maintainability, flexibility, and quality in projects of all kinds and scale. By now, C++ has fulfilled most of the hopes I originally had for it, and has succeeded at tasks I hadn't even dreamt of.

This book introduces standard C++[†] and the key programming and design techniques supported by C++. Standard C++ is a far more powerful and polished language than the version introduced by the first edition of this book. New language features such as namespaces, templates, and run-time type identification allow many techniques to be applied more easily than was possible before, and the standard library allows the programmer to start from a higher level than the bare language.

About a third of the information in the second edition of this book came from the experience of an experienced C++ programmer; at the same time, this book is easier for the novice to use than its predecessors were. The explosion of C++ use and the massive amount of experience accumulated as a result makes this possible.

The definition of an extensive standard library makes a difference to the way C++

be presented. As before, this book presents C++ independently of any particular implementation and as before, the tutorial chapters present language constructs and concepts in an order so that a construct is used only after it has been defined. However, it is much easier to use the well-designed library than it is to understand the details of its implementation. Therefore, the standard library can be used to provide realistic and interesting examples well before one is assumed to understand its inner workings. The standard library itself is also a fertile source of programming examples and design techniques.

This book presents every major C++ language feature and the standard library. It is organized around language and library facilities. However, features are presented in the context of

† ISO/IEC 14882, Standard for the C++ Programming Language.