

Microsoft®

CODE COMPLETE

2

Second Edition



A practical handbook of software construction

Steve McConnell

Two-time winner of the *Software Development Magazine* Jolt Award

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Copyright © 2004 by Steven C. McConnell

All rights reserved. No part of the contents of this book may be reproduced or transmitted in any form or by any means without the written permission of the publisher.

Library of Congress Cataloging-in-Publication Data
McConnell, Steve

Code Complete / Steve McConnell.--2nd ed.

p. cm.

Includes index.

ISBN 0-7356-1967-0

1. Computer Software--Development--Handbooks, manuals, etc. I. Title.

QA76.76.D47M39 2004

005.1--dc22

2004049981

Printed and bound in the United States of America.

ISBN: 978-0-7356-1967-8

Twenty-fourth Printing: February 2015

Distributed in Canada by H.B. Fenn and Company Ltd. A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office or contact Microsoft Press International directly at fax (425) 936-7329. Visit our Web site at www.microsoft.com/mspress. Send comments to mspinput@microsoft.com.

Microsoft, Microsoft Press, PowerPoint, Visual Basic, Windows, and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other product and company names mentioned herein may be the trademarks of their respective owners.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

This book expresses the author's views and opinions. The information contained in this book is provided without any express, statutory, or implied warranties. Neither the authors, Microsoft Corporation, nor its resellers, or distributors will be held liable for any damages caused or alleged to be caused either directly or indirectly by this book.

Acquisitions Editors: Linda Engelman and Robin Van Steenburgh

Project Editor: Devon Musgrave

Indexer: Bill Myers

Principal Desktop Publisher: Carl Diltz

Body Part No. X10-53130

Code Complete

Table of Contents

Cover

Copyright

Further Praise for Code Complete

Contents at a Glance

Table of Contents

Preface

- Who Should Read This Book?

- Where Else Can You Find This Information?

- Key Benefits of This Handbook

- Why This Handbook Was Written

- Author Note

Acknowledgments

Checklists

Tables

Figures

Part I: Laying the Foundation

- Chapter 1: Welcome to Software Construction

 - 1.1 What Is Software Construction?

 - 1.2 Why Is Software Construction Important?

 - 1.3 How to Read This Book

 - Key Points

- Chapter 2: Metaphors for a Richer Understanding of Software

Table of Contents

Development

- 2.1 The Importance of Metaphors
- 2.2 How to Use Software Metaphors
- 2.3 Common Software Metaphors

Additional Resources

Key Points

Chapter 3: Measure Twice, Cut Once: Upstream Prerequisites

- 3.1 Importance of Prerequisites
- 3.2 Determine the Kind of Software You're Working On
- 3.3 Problem-Definition Prerequisite
- 3.4 Requirements Prerequisite
- 3.5 Architecture Prerequisite
- 3.6 Amount of Time to Spend on Upstream Prerequisites

Additional Resources

Key Points

Chapter 4: Key Construction Decisions

- 4.1 Choice of Programming Language
- 4.2 Programming Conventions
- 4.3 Your Location on the Technology Wave
- 4.4 Selection of Major Construction Practices

Key Points

Part II: Creating High-Quality Code

Chapter 5: Design in Construction

- 5.1 Design Challenges
- 5.2 Key Design Concepts
- 5.3 Design Building Blocks: Heuristics
- 5.4 Design Practices
- 5.5 Comments on Popular Methodologies

Additional Resources

Table of Contents

Key Points

Chapter 6: Working Classes

6.1 Class Foundations: Abstract Data Types (ADTs)

6.2 Good Class Interfaces

6.3 Design and Implementation Issues

6.4 Reasons to Create a Class

6.5 Language-Specific Issues

6.6 Beyond Classes: Packages

Additional Resources

Key Points

Chapter 7: High-Quality Routines

7.1 Valid Reasons to Create a Routine

7.2 Design at the Routine Level

7.3 Good Routine Names

7.4 How Long Can a Routine Be?

7.5 How to Use Routine Parameters

7.6 Special Considerations in the Use of Functions

7.7 Macro Routines and Inline Routines

Key Points

Chapter 8: Defensive Programming

8.1 Protecting Your Program from Invalid Inputs

8.2 Assertions

8.3 Error-Handling Techniques

8.4 Exceptions

8.5 Barricade Your Program to Contain the Damage Caused by Errors

8.6 Debugging Aids

8.7 Determining How Much Defensive Programming to Leave in Production Code

8.8 Being Defensive About Defensive Programming

Additional Resources

Table of Contents

Key Points

Chapter 9: The Pseudocode Programming Process

9.1 Summary of Steps in Building Classes and Routines

9.2 Pseudocode for Pros

9.3 Constructing Routines by Using the PPP

9.4 Alternatives to the PPP

Key Points

Part III: Variables

Chapter 10: General Issues in Using Variables

10.1 Data Literacy

10.2 Making Variable Declarations Easy

10.3 Guidelines for Initializing Variables

10.4 Scope

10.5 Persistence

10.6 Binding Time

10.7 Relationship Between Data Types and Control Structures

10.8 Using Each Variable for Exactly One Purpose

Key Points

Chapter 11: The Power of Variable Names

11.1 Considerations in Choosing Good Names

11.2 Naming Specific Types of Data

11.3 The Power of Naming Conventions

11.4 Informal Naming Conventions

11.5 Standardized Prefixes

11.6 Creating Short Names That Are Readable

11.7 Kinds of Names to Avoid

Key Points

Chapter 12: Fundamental Data Types

Table of Contents

- 12.1 Numbers in General
- 12.2 Integers
- 12.3 Floating-Point Numbers
- 12.4 Characters and Strings
- 12.5 Boolean Variables
- 12.6 Enumerated Types
- 12.7 Named Constants
- 12.8 Arrays
- 12.9 Creating Your Own Types (Type Aliasing)
- Key Points

Chapter 13: Unusual Data Types

- 13.1 Structures
- 13.2 Pointers
- 13.3 Global Data
- Additional Resources
- Key Points

Part IV: Statements

Chapter 14: Organizing Straight-Line Code

- 14.1 Statements That Must Be in a Specific Order
- 14.2 Statements Whose Order Doesn't Matter
- Key Points

Chapter 15: Using Conditionals

- 15.1 if Statements
- 15.2 case Statements
- Key Points

Chapter 16: Controlling Loops

- 16.1 Selecting the Kind of Loop
- 16.2 Controlling the Loop
- 16.3 Creating Loops Easily From the Inside Out

Table of Contents

16.4 Correspondence Between Loops and Arrays

Key Points

Chapter 17: Unusual Control Structures

17.1 Multiple Returns from a Routine

17.2 Recursion

17.3 goto

17.4 Perspective on Unusual Control Structures

Additional Resources

Key Points

Chapter 18: Table-Driven Methods

18.1 General Considerations in Using Table-Driven Methods

18.2 Direct Access Tables

18.3 Indexed Access Tables

18.4 Stair-Step Access Tables

18.5 Other Examples of Table Lookups

Key Points

Chapter 19: General Control Issues

19.1 Boolean Expressions

19.2 Compound Statements (Blocks)

19.3 Null Statements

19.4 Taming Dangerously Deep Nesting

19.5 A Programming Foundation: Structured Programming

19.6 Control Structures and Complexity

Key Points

Part V: Code Improvements

Chapter 20: The Software-Quality Landscape

20.1 Characteristics of Software Quality

20.2 Techniques for Improving Software Quality

Table of Contents

20.3 Relative Effectiveness of Quality Techniques

20.4 When to Do Quality Assurance

20.5 The General Principle of Software Quality

Additional Resources

Key Points

Chapter 21: Collaborative Construction

21.1 Overview of Collaborative Development Practices

21.2 Pair Programming

21.3 Formal Inspections

21.4 Other Kinds of Collaborative Development Practices

21.5 Comparison of Collaborative Construction Techniques

Additional Resources

Key Points

Chapter 22: Developer Testing

22.1 Role of Developer Testing in Software Quality

22.2 Recommended Approach to Developer Testing

22.3 Bag of Testing Tricks

22.4 Typical Errors

22.5 Test-Support Tools

22.6 Improving Your Testing

22.7 Keeping Test Records

Additional Resources

Key Points

Chapter 23: Debugging

23.1 Overview of Debugging Issues

23.2 Finding a Defect

23.3 Fixing a Defect

23.4 Psychological Considerations in Debugging

23.5 Debugging Tools Obvious and Not-So-Obvious

Table of Contents

Additional Resources

Key Points

Chapter 24: Refactoring

24.1 Kinds of Software Evolution

24.2 Introduction to Refactoring

23.3 Specific Refactorings

23.4 Refactoring Safely

23.5 Refactoring Strategies

Additional Resources

Key Points

Chapter 25: Code-Tuning Strategies

25.1 Performance Overview

25.2 Introduction to Code Tuning

25.3 Kinds of Fat and Molasses

25.4 Measurement

25.5 Iteration

25.6 Summary of the Approach to Code Tuning

Additional Resources

Key Points

Chapter 26: Code-Tuning Techniques

26.1 Logic

26.2 Loops

26.3 Data Transformations

26.4 Expressions

26.5 Routines

26.6 Recoding in a Low-Level Language

26.7 The More Things Change, the More They Stay the Same

Additional Resources

Key Points

Table of Contents

Part VI: System Considerations

Chapter 27: How Program Size Affects Construction

- 27.1 Communication and Size
- 27.2 Range of Project Sizes
- 27.3 Effect of Project Size on Errors
- 27.4 Effect of Project Size on Productivity
- 27.5 Effect of Project Size on Development Activities
- Additional Resources
- Key Points

Chapter 28: Managing Construction

- 28.1 Encouraging Good Coding
- 28.2 Configuration Management
- 28.3 Estimating a Construction Schedule
- 28.4 Measurement
- 28.5 Treating Programmers as People
- 28.6 Managing Your Manager
- Additional Resources on Managing Construction
- Key Points

Chapter 29: Integration

- 29.1 Importance of the Integration Approach
- 29.2 Integration Frequency Phased or Incremental?
- 29.3 Incremental Integration Strategies
- 29.4 Daily Build and Smoke Test
- Additional Resources
- Key Points

Chapter 30: Programming Tools

- 30.1 Design Tools
- 30.2 Source-Code Tools

Table of Contents

- 30.3 Executable-Code Tools
- 30.4 Tool-Oriented Environments
- 30.5 Building Your Own Programming Tools
- 30.6 Tool Fantasyland
- Additional Resources
- Key Points

Part VII: Software Craftsmanship

Chapter 31: Layout and Style

- 31.1 Layout Fundamentals
- 31.2 Layout Techniques
- 31.3 Layout Styles
- 31.4 Laying Out Control Structures
- 31.5 Laying Out Individual Statements
- 31.6 Laying Out Comments
- 31.7 Laying Out Routines
- 31.8 Laying Out Classes
- Additional Resources
- Key Points

Chapter 32: Self-Documenting Code

- 32.1 External Documentation
- 32.2 Programming Style as Documentation
- 32.3 To Comment or Not to Comment
- 32.4 Keys to Effective Comments
- 32.5 Commenting Techniques
- 32.6 IEEE Standards
- Additional Resources
- Key Points

Chapter 33: Personal Character

Table of Contents

33.1 Isn't Personal Character Off the Topic?

33.2 Intelligence and Humility

33.3 Curiosity

33.4 Intellectual Honesty

33.5 Communication and Cooperation

33.6 Creativity and Discipline

33.7 Laziness

33.8 Characteristics That Don't Matter As Much As You Might Think

3.9 Habits

Additional Resources

Key Points

Chapter 34: Themes in Software Craftsmanship

34.1 Conquer Complexity

34.2 Pick Your Process

34.3 Write Programs for People First, Computers Second

34.4 Program into Your Language, Not in It

34.5 Focus Your Attention with the Help of Conventions

34.6 Program in Terms of the Problem Domain

34.7 Watch for Falling Rocks

34.8 Iterate, Repeatedly, Again and Again

34.9 Thou Shalt Rend Software and Religion Asunder

Key Points

Chapter 35: Where to Find More Information

35.1 Information About Software Construction

35.2 Topics Beyond Construction

35.3 Periodicals

35.4 A Software Developers Reading Plan

35.5 Joining a Professional Organization

Bibliography

Table of Contents

Index

About the Author