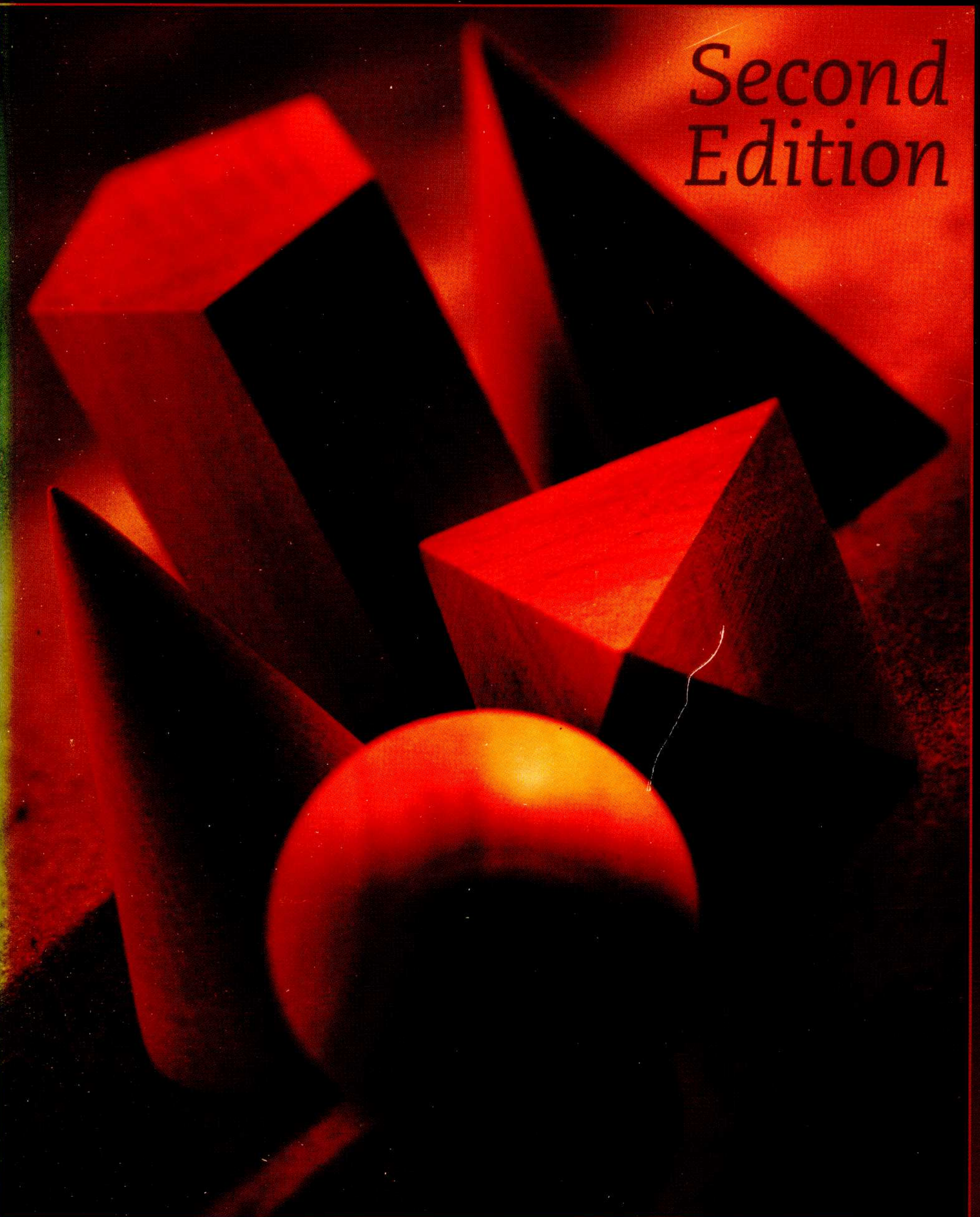
Problem Solving *with* Java™

Second Edition



Elliot B. Koffman ■ Ursula Wolz

1001664

Problem Solving with Java™

Elliot B. Koffman
Ursula Wolz

江苏工业学院图书馆
藏书章

东南大学图书馆藏书章

Addison Wesley

Boston San Francisco New York
London Toronto Sydney Tokyo Singapore Madrid
Mexico City Munich Paris Cape Town Hong Kong Montreal

Executive Editor	Susan Hartman Sullivan
Executive Marketing Manager	Michael Hirsch
Production Supervisor	Marilyn Lloyd
Project Management	Argosy Publishing
Composition and Art	Argosy Publishing
Copyeditor	Nancy Young
Proofreader	Janet Renard
Indexer	Elliot B. Koffman
Text Design	Sandra Rigney
Cover Design	Gina Hagen Kolenda
Cover Photo	Photo Disk ©2002
Design Manager	Gina Hagen Kolenda
Prepress and Manufacturing	Caroline Fell

Access the latest information about Addison-Wesley titles from our World Wide Web site: http://www.aw.com/cs

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 Addison-Wesley was aware of a trademark claim, the designations have been printed in initial caps or all caps.

The programs and applications presented in this book have been included for their instructional value. They have been tested with care, but are not guaranteed for any particular purpose. The publisher does not offer any warranties or representations, not does it accept any liabilities with respect to the programs or applications.

Library of Congress Cataloging-in-Publication Data

Koffman, Elliot B.
Problem Solving with Java / Elliot B. Koffman and Ursula Wolz. – 2nd ed.
p. cm.
ISBN 0-201-72214-3
1. Java (Computer program language) I. Wolz, Ursula. II. Title.

QA76.73.J38 K64 2002
005.13'3--dc21 2001045827

Copyright © 2002 by Addison-Wesley

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. Printed in the United States of America.

ISBN 0-201-72214-3
2345678910-DOC-040302

DEDICATION

To Elliot's family:
Caryn, Debbie, Robin and Jeff, Richard, Jacquie, and Dustin

and to Ursula's family:
Jim, Chris, Henry, Kathy and their little girl

Preface

This textbook is intended for a first course in problem solving and program design with Java (CS1). It assumes no prior knowledge of computers or programming, and for most of its material, high school algebra is sufficient mathematics background. A limited knowledge of discrete mathematics is desirable for a few sections.

Problem Solving and Program Design

The primary emphasis in this text is on **problem solving** with Java. We accomplish this by selecting features of the language that lend themselves to good program design. We also emphasize abstraction and follow a standard five-step approach to program design (problem specification, analysis, design, implementation, and testing). We have modified this time-tested approach for software development to the object-oriented paradigm. We follow it faithfully in the solution of more than 20 case studies throughout the book. Ten of the case studies are new to this edition.

Classes and Objects Early

Students use predefined classes like `String` and `Math` to write small applications in Chapter 2. They begin to write their own worker classes to model real-world objects in Chapter 3. Examples are a `FoodItem` class, a `CoinChanger` class, a `Circle` class, and a `Washer` class. Methods are introduced in Chapter 2 and thoroughly covered in Chapter 3.

Object-Oriented Programming (OOP)

We continue to use worker classes in applications and discuss OOP concepts in an informal manner. We provide a detailed discussion of OOP in Chapter 6. We introduce class hierarchies, inheritance, interfaces, and abstract classes by studying several case studies that use these features.

此为试读，需要完整PDF请访问：www.ertongbook.com

Standard Input/Output Stressed

We use standard Java for input and output. Starting in Chapter 1, we use class `JOptionPane` (part of `Swing`) for windows-based input and output and we use the console window for more extensive output. In Chapter 3, we provide an optional package with static methods for input (based on `JOptionPane`) that simplifies data entry with dialog windows. The input methods check for number format errors and can check for range violations. Most programs in the book use standard Java I/O methods, but students can use the optional package if they wish.

This is a departure from the first edition, which utilized a nonstandard graphical user interface (GUI) package. Our experience is that many of the benefits of this package can be derived through class `JOptionPane`. Many Java programming instructors preferred to teach standard methods rather than rely on a nonstandard package. We hope that we have met the needs of most users by relying on standard input/output and by also providing an optional nonstandard package that is simpler to use.

Applets and Applications

We focus on applications in Chapters 2 and 3, where we attempt to teach the basics of simple programs that calculate results. We use applications rather than applets because we don't want students to have to deal with the details of providing a GUI. We introduce applets, HyperText Markup Language (HTML), and graphics programming using the AWT Graphics class to draw simple graphical patterns in Chapter 3. When we cover GUIs in Chapter 7, we use `Swing` components (see below).

Control Structures and Indexed Data Structures

In this edition, we cover selection and loop control structures together in Chapter 4. However, the control structures are not intermixed. We complete the selection control structures before we begin loop control structures, so instructors can separate these structures if they wish to.

We study arrays in Chapter 5, along with other Java indexed data collections, the `Vector` class and `ArrayList` class. We also discuss wrapper classes for the primitive types in this chapter.

GUIs

We revisit applets and HTML in Chapter 7 when we describe how to build GUIs using `Swing` components. We also show how to use class `JFrame` to write applications that have GUIs. We show several examples of GUIs in both applications and applets.

Exceptions and Streams

Chapter 8 is a new chapter on exceptions and streams. Knowing how to catch and throw exceptions is critical to stream processing, so we begin the chapter with a discussion of exceptions. The chapter covers streams of characters, binary streams, and streams of objects.

Coverage of Advanced Topics

Chapters 9 and 10 concern themselves with more traditional aspects of programming often found in CS2: recursion and processing linked data structures. We develop classes for linked lists, stacks, queues, and binary search trees. We define the node structure in inner classes. We also show how to use the **`LinkedList`** collection class and the **`ListIterator`** class. Many CS1 courses would not include this advanced material.

Flexibility of Coverage

There is sufficient material in the textbook for one and a half semesters or for two quarters. We consider Chapters 1 through 7 the core of the book, and they should be covered by all students. The first four chapters (through control structures) must be covered in sequence:

1. Introduction to Computers, Problem Solving, and Programming
2. Using Primitive Data Types and Using Classes
3. Object-Oriented Design and Writing Worker Classes
4. Control Structures: Decisions and Loops

The next three chapters deal with arrays, OOP, and GUI design, and they can be covered in a variety of ways:

5. Arrays and Vectors
6. Class Hierarchies, Inheritance, and Interfaces
7. Graphical User Interfaces (GUIs)

Faculty who want to cover GUIs earlier can cover Chapter 7 first, omitting the few examples that involve arrays. Similarly, faculty who want to cover OOP details earlier can introduce the fundamentals of using arrays (Sections 5.1–5.3) and then cover Chapter 6 in detail. Then continue with the rest of Chapter 5 or Chapter 7.

Chapter 9, Recursion, could also be introduced earlier. Sections 9.1–9.3 could be covered after Chapter 4 and the rest of the chapter could be covered after Chapter 5.

Pedagogical Features

We employ several pedagogical features to enhance the usefulness of the book as a teaching tool. Discussion of some of these features follows.

End-of-Section Exercises: Most sections end with a number of self-check exercises, including exercises that require analysis of program fragments as well as short programming exercises. Answers to odd-numbered self-check exercises appear at the back of the book; answers to other exercises are provided in the Instructor's Manual.

End-of-Chapter Exercises: Each chapter ends with a set of quick-check exercises with answers. There are also chapter review exercises with solutions provided in the Instructor's Manual.

End-of-Chapter Projects: There are several projects at the end of each chapter that are suitable for programming assignments. Answers to selected projects appear in the Instructor's Manual.

Examples and Case Studies: The text contains a large number and variety of programming examples. Whenever possible, examples contain complete class or method definitions rather than incomplete fragments. Each chapter contains one or more case studies that are solved following the software development method.

Syntax Displays: The syntax displays describe the syntax and semantics of each new Java feature complete with examples.

Program Style Displays: The program style displays discuss issues of good programming style.

Error Discussions and Chapter Review: Each chapter ends with a section that discusses common programming errors. Chapter reviews include a table of new Java constructs.

Appendixes and Supplements

Appendixes: The text concludes with several appendixes covering the Java language, HTML, unicode, Borland JBuilder, resources for finding out more about Java, and a summary of Java class libraries.

Packages and Classes: Further information about this textbook can be found at www.aw.com/cssupport. You will be able to download package psJava and source code for all the classes provided in the textbook.

Instructor's Manual: Access to an online instructor's manual is available through your Addison-Wesley sales representative. The Instructor's Manual contains answers to selected exercises and projects and is available to qualified instructor's only.

Acknowledgments

There were many individuals without whose support this book could not have been written. These include the principal reviewers of this edition and the first edition:

Julia E. Benson, DeKalb College
Richard J. Botting, California State University, San Bernadino
Tom Cortina, SUNY at Stony Brook
Robert H. Dependahl, Jr., Santa Barbara City College
Bill Grosky, Wayne State University
Stanley H. Lipson, Kean University of New Jersey
David Mathias, Ohio State University
Bina Ramamurthy, SUNY at Buffalo
Stuart Reges, University of Arizona
James Svoboda, Clarkson University
John S. Zelek, University of Guelph

Several students at The College of New Jersey (TCNJ) and Temple University helped with the development of the textbook. They include Brice Behringer, Greg Bronevetsky, George Drayer, William Fenstermaker, Mark Nikolsky, Brian Robinson, and Michael Sipper from TCNJ and N.D. Brabham, Yadh El Afrit, Thaung Ngwe, Blossom Pinheiro, Brian Rubin, John Salmon, and Saritha Somasundaram from Temple.

We are also grateful to our colleagues who provided valuable insight and advice. In particular, we want to acknowledge Joseph Turner of Clemson University and Penny Anderson and Shawn Sivy from The College of New Jersey.

There are several individuals at Addison-Wesley who worked very hard to see this textbook completed. They include our sponsoring editor Susan Hartman and her very able assistant Elinor Actipis. Among their many contributions, Susan and Elinor worked with us and the principal reviewers to help refine our manuscript.

Finally, we would like to acknowledge the help and support of our families. The members of Ursula's household, her husband Jim, son Christopher, and father Henry, were all very understanding about her preoccupation with this book over the past year as were Elliot's wife Caryn and his children Deborah, Richard, and Robin and grandson Dustin. We are grateful to them all for their love and support.

EBK
UW

Contents

Chapter 1 Introduction to Computers, Problem Solving, and Programming 1

1.1 Overview of Computers 2
- Mainframe and Minicomputers 2
- Sharing Computer Resources 2
- Microcomputers 3

1.2 Computer Components 3
- Memory 5
- Bytes and Bits 5
- Central Processor Unit 8
- Input/Output Devices 9
- Computer Networks 10
- The World Wide Web 10
- Programming for the Web 11

1.3 Computer Software 12
- Operating Systems 12
- Application Software 14
- Programming Languages 14
- Object-Oriented Programming 16

1.4 Processing a High-Level Language Program 18
- Cross-Platform Transparency and the Java Virtual Machine 18
- Applets and Applications 20

1.5 The Software Development Method 21
- Problem 22
- Analysis 22
- Design the Classes 22
- Implementation 23
- Testing 23
- Maintenance 23
- Applying the Software Development Method 24
- CASE STUDY: Showing Your Name in Stars 24

1.6 Professional Ethics for Computer Programmers 27
Privacy and Misuse of Data 27
Hacking Is Irresponsible and Illegal 28
Plagiarism and Software Piracy 28
Misuse of a Computer Resource 29

Chapter 2 Using Primitive Data Types and Using Classes 35

2.1 Primitive Data Types 36
Data Types,Variables, and Declarations 36
Syntax Displays 37
Primitive Type **`int`** 38
Primitive Type **`double`** 38
Primitive Type **`boolean`** 39
Primitive Type **`char`** 39
Data Types Enable Error Detection 39
2.2 Processing Numeric Data 41
The Division and Remainder Operators 41
Assignment Statements 42
Data Type of an Arithmetic Operation 43
Mixed-Type Assignment Statement 44
Casting Operations for Type Conversion 45
Expressions with Multiple Operators 45
Writing Mathematical Formulas in Java 48
2.3 Introduction to Methods 52
Calling or Invoking Methods Using Dot Notation 52
Method Arguments Are Like Function Arguments 53
Instance Methods versus Class Methods 54
2.4 The **`String`** Class 55
Declaring **`String`** Variables 55
Using Constructors to Create **`String`** Objects 55
Combining Variable Declaration and Object Creation 56
Reference Variables 56
Special Properties of Strings 57
`String` Operator **`+`** and **`String`** Assignment 57
Comparison of **`String`** and **`Character`** Data Types 58
Concatenating Strings with Primitive Types 59
`String` Methods 60
Storing the Result of a Method Call 63
Displaying the Result of a Method Call 64

此为试读，需要完整PDF请访问：www.ertongbook.com

2.5 Input/Output with Class `JOptionPane` and Method `println()` 65
- `import` Statement 65
- Reading Data Using `JOptionPane.showInputDialog()` 66
- Displaying Results in the Console Window 68
- Displaying Results Using `JOptionPane.showMessageDialog()` 69

2.6 Problem Solving in Action 73
- CASE STUDY: A Pig Latin Translator 73

2.7 Anatomy of a Java Program 76
- Comments 76
- Class Definitions 77
- Definition of Method `main()` 78
- The Body of method `main()` 78
- Program Style and Programmer Conventions 79
- Line Breaks in Programs 80
- Blank Spaces and Lines in Programs 81
- Java Keywords Revisited 81
- Identifiers 81
- Syntax Rules for Identifiers 82
- Valid Identifiers 82

2.8 Numerical Computations with Class `Math` 84
- Methods in Class `Math` 85

2.9 Common Errors and Debugging 92
- Syntax Errors 92
- System Errors 94
- Run-Time Errors 95
- Logic Errors 95
- Debugging Using Method `println()` 96

Chapter 3 Object-Oriented Design and Writing Worker Classes 105

3.1 A First Worker Class: Class `FoodItem` 106
- Review of Class Definitions 106
- Giving a Data Field an Initial Value 107
- Method Definitions 107
- Methods for Class `FoodItem` 108
- Constructor Method 111
- `Void` Methods 112
- The `return` Statement 112
- Calling Methods 113
- Postconditions 113
- Locality of Method Parameter and Variable Declarations 114

Argument/Parameter Correspondence in Method Calls 114
Arguments Are Passed by Value 117
Transfers of Control during Method Call and Return 118

3.2 A Worker Class That Manipulates **String** Objects 120
CASE STUDY: Finding the Words in a Sentence 120

3.3 A Worker Class That Processes Integers 127
CASE STUDY: Designing a Coin Changer 127

3.4 Review of Methods 134
Constructor Methods 134
Accessor Methods 135
String Objects Are Immutable 136
Calling One Instance Method from Another 137
Use of Prefix **this** with a Data Field 137
Class Methods versus Instance Methods 137
Problem Inputs, Problem Outputs, Method Inputs, Method Outputs 139

3.5 Simplifying a Solution Using Multiple Classes 141
CASE STUDY: Computing the Weight of Flat Washers 141

3.6 Formatting Output and Class **KeyIn** (Optional) 149
Formatting Numbers with Class DecimalFormat 149
Class **NumberFormat** 151
Class **KeyIn** Methods for Input Operations (Optional) 153
Specifying a Range of Values 154

3.7 Applets, AWT, and the Graphics Class (Optional) 156
The Drawing Surface 156
The AWT Class Library 156
Displaying Text and Drawing 160
Drawing Rectangles 161
Drawing Arcs and Circles 164
Drawing Pie Slices 168

3.8 Common Programming Errors 170

Chapter 4 Control Structures: Decisions and Loops 181

4.1 Control Structures 182

4.2 **boolean** Expressions 183
boolean Variables 183
Relational Operators in **boolean** Expressions 183
Reading **boolean** Data 185
boolean Operators 186
Short-Circuit Evaluation of **boolean** Expressions 189
Writing English Conditions in Java 190
DeMorgan's Theorem (Optional) 192
Comparing Characters 192
Comparing Strings for Equality 193

Method `equalsIgnoreCase()` 194
Lexicographic Comparisons of Strings 194
4.3 The `if` Statement 197
`if` Statement with One Consequent 197
`if` Statement with Two Alternatives 197
Methods That Return `boolean` Results 201
4.4 Decision Steps in Algorithms 204
Searching a String 204
CASE STUDY: Payroll Problem 205
4.5 Multiple-Alternative Decisions: Nested `if` and `switch` 215
Comparison of Nested if and Sequence of `ifs` 216
Java Rule for Matching `else` with `if` 217
Multiple-Alternative Decision Form of Nested `if` 218
Order of Conditions Matters 220
`switch` Statement 222
4.6 Counting Loops, `while` and `for` Statements 225
Counting Loops 225
The `while` Statement 226
Comparison of `if` and `while` Statements 228
Loop-Control Variable 228
The `for` Statement 229
Accumulating a Sum 231
Increment, Decrement, and Compound Operators 233
4.7 State-Controlled Loops 238
Sentinel-Controlled Loops 239
Loops and Event-Driven Programming 242
4.8 Putting It All Together 243
CASE STUDY: Arithmetic Drill and Practice 243
4.9 Debugging and Testing Programs with Decisions and Loops 252
Using a Debugger 252
Debugging without a Debugger 253
Testing 254
4.10 Common Programming Errors 254
`if` Statement Errors 254
Loop Errors 255
Off-by-One Loop Errors 256
Checking Loop Boundaries 257

Chapter 5 Arrays and Vectors 271

5.1 Array Declarations 272
Declaring Arrays 272
Separating Array Declaration and Storage Allocation 273
Array Index 274
Arrays of Strings 277

5.2 Processing Arrays and Array Elements 279
Data Field **`length`** 279
Displaying Array Elements 280
CASE STUDY: Calculating Exam Statistics 283
5.3 Operations on Whole Arrays 290
Declaring Arrays 291
Copying Arrays 291
Array Assignment 292
Passing Arrays to Methods 293
Argument Correspondence for Array Arguments 293
Returning an Array of Values 295
5.4 Searching and Sorting an Array 297
Array Search 298
Sorting an Array 299
Method **`findPosMin()`** 302
Loop Invariants 302
Finding the Median Value in an Array 302
5.5 Arrays of Objects 304
Arrays of Strings 304
Menus 306
Array of **`Employee`** Objects 308
CASE STUDY: Computing Payroll Using an Array of Employees 309
CASE STUDY: Phone Directory 311
5.6 Multidimensional Arrays—Arrays of Arrays 323
Declaring Two-Dimensional Arrays 323
Initializing a Two-Dimensional Array 325
Processing Two-Dimensional Arrays Using Nested Loops 326
Use of **`length`** Attribute with Two-Dimensional Arrays 328
Arrays with More Than Two Dimensions 328
5.7 Vectors 329
Casting Objects in a Vector 331
Storing Different Type Objects in a Vector 333
Class **`PhoneBook`** as a Vector 333
5.8 Wrapper Classes for Primitive Type Data 337
5.9 **`Arrays`** and **`ArrayList`** Collection Classes (Optional) 340
`Arrays` Class 340
Class **`ArrayList`** 341
5.10 Common Programming Errors 346

Chapter 6 Class Hierarchies, Inheritance, and Interfaces 359

6.1 Class Hierarchies and Inheritance 360
is a versus *has a* Relationships 361
CASE STUDY: A Hierarchy of Employee Classes 361
6.2 Operations in a Class Hierarchy 376
Method Overriding 378
Method Overloading 379
Protected Visibility 379
Shadowing Data Fields 380
Misnomer of Superclass and Subclass 381
Assignment of a Class Hierarchy 381
Casting in a Class Hierarchy 384
The `instanceof` Operator 386
6.3 Polymorphism 388
CASE STUDY: A Company with an Array of Employees of Different Types 389
Polymorphism and Late Binding of Method Calls 390
6.4 Interfaces 394
Declaring Constants in Interfaces 397
The Comparable Interface and Method `compareTo()` 397
Sorting Comparable Objects 398
6.5 Abstract Classes 404
CASE STUDY: Areas of Geometric Figures 405
6.6 Drawing Figures Using an Abstract Class and an Interface 414
CASE STUDY: Drawing Geometric Figures 414
Abstract Classes, Multiple Inheritance, and Interfaces 421
6.7 Packages and Visibility (Optional) 425
The `import` Statement 425
Default Visibility 426
Visibility Supports Encapsulation 427
The No Package Declared Environment 427
6.8 Testing a Program System 428
Top-Down Testing and Stubs 428
Bottom-Up Testing and Drivers 429
Identifier Scope and Watch Window Variables 430
Black-Box versus White-Box Testing 430
6.9 Common Programming Errors 431

Chapter 7 **Graphical User Interfaces (GUIs)** **439**

7.1 AWT, `Swing`, and Browser-Applet Interaction 440
- Categories of Classes in `Swing` and AWT 441
- GUI Component Classes 441
- Overview of AWT and `Swing` Hierarchy 442
- Applets and Panels Are Containers 444
- Browser and Applet Interaction 444

7.2 Designing a First GUI 446
- The Containment Hierarchy for a First GUI 447
- Declaring the GUI Components 448
- Placing the Components in the Applet 451
- Adding Color and Borders to Components 452

7.3 The Java Event Model 455
- Registering as an Event Listener 455
- The `ActionListener` Interface 456
- Method `actionPerformed()` 456
- Handling Other Events 459
- Running the Applet 459
- Comparison of Event-Driven Programming and "Stop-and-Wait" Programming 460

7.4 Using a GUI in an Application 461

7.5 Components for Making Choices 465
- Check Boxes 465
- Radio Buttons 468
- Combo Boxes 472

7.6 Designing a GUI for an Existing Class 476
- CASE STUDY: A GUI for the `PhoneBook` Class 476

7.7 Listener Classes as Inner Classes 486

7.8 Layout Managers 489
- Flow Layout 490
- Border Layout 490
- Box Layout 492
- Grid Layout 496
- CASE STUDY: A GUI for a Payroll Application 499

7.9 Common Programming Errors 509

Chapter 8 **Exceptions, Streams, and Files** **515**

8.1 Exceptions 516
- Using `if` Statements to Prevent Errors 516
- Exception Classes 516
- `try-catch` Statement 516
- Order of `catch` Blocks 521

此为试读，需要完整PDF请访问：www.ertongbook.com