

GLOBAL
EDITION

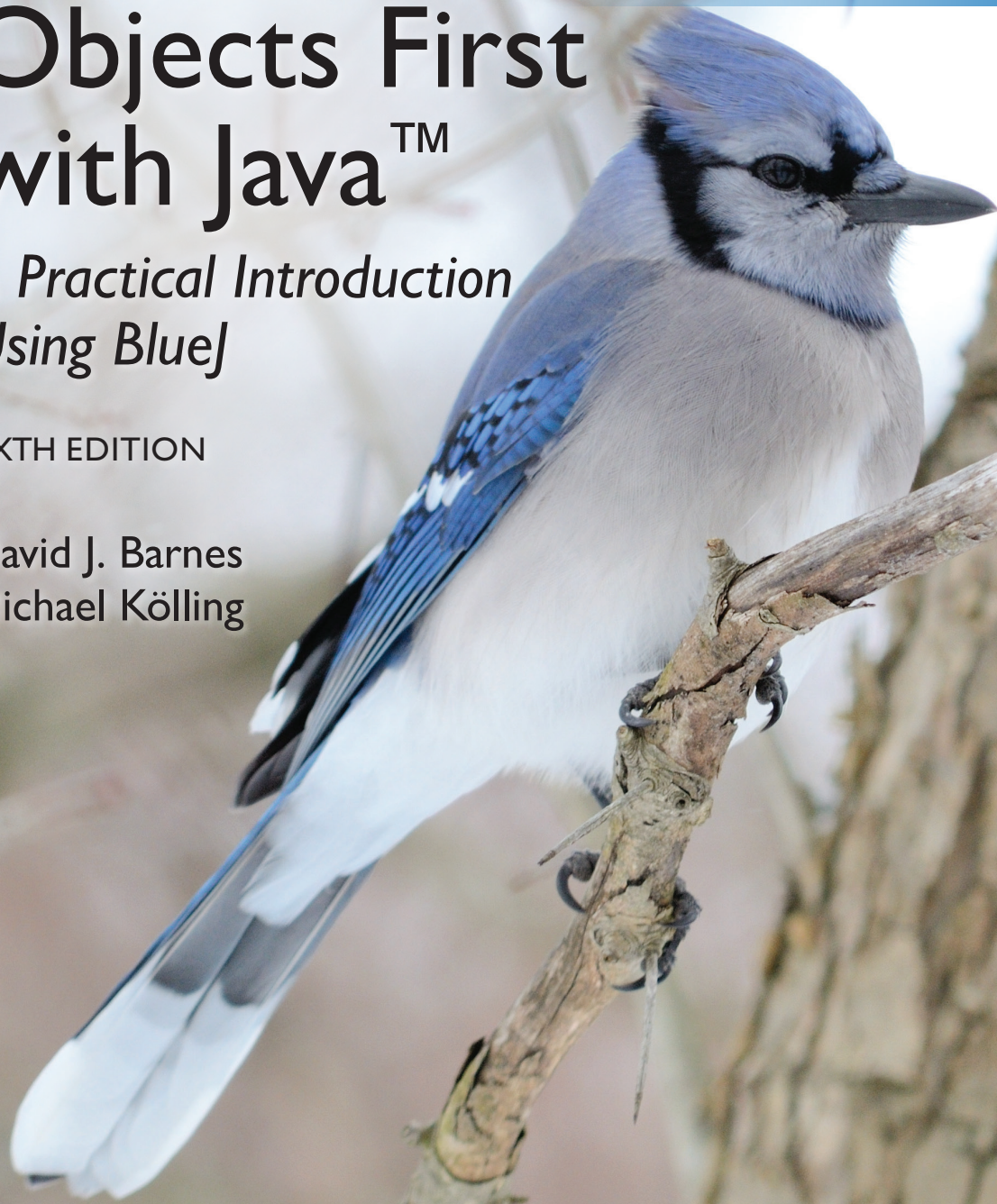


Objects First with JavaTM

*A Practical Introduction
Using BlueJ*

SIXTH EDITION

David J. Barnes
Michael Kölling



ALWAYS LEARNING

PEARSON



Objects First with Java™

A Practical Introduction Using BlueJ

David J. Barnes and Michael Kölling

University of Kent

Sixth Edition

Global Edition

PEARSON

Boston Columbus Indianapolis New York San Francisco Hoboken
Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto
Delhi Mexico City Sao Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo

Objects First with Java: A Practical Introduction Using BlueJ, Global Edition

Table of Contents

Cover

Title Page

Copyright Page

Contents

Foreword

Preface

List of Projects Discussed in Detail in This Book

Acknowledgments

Part 1 Foundations of Object Orientation

Chapter 1 Objects and Classes

1.1 Objects and classes

1.2 Creating objects

1.3 Calling methods

1.4 Parameters

1.5 Data types

1.6 Multiple instances

1.7 State

1.8 What is in an object?

1.9 Java code

1.10 Object interaction

1.11 Source code

1.12 Another example

Table of Contents

1.13 Return values

1.14 Objects as parameters

1.15 Summary

Chapter 2 Understanding Class Definitions

2.1 Ticket machines

2.2 Examining a class definition

2.3 The class header

2.4 Fields, constructors, and methods

2.5 Parameters: receiving data

2.6 Assignment

2.7 Methods

2.8 Accessor and mutator methods

2.9 Printing from methods

2.10 Method summary

2.11 Summary of the naïve ticket machine

2.12 Reflecting on the design of the ticket machine

2.13 Making choices: the conditional statement

2.14 A further conditional-statement example

2.15 Scope highlighting

2.16 Local variables

2.17 Fields, parameters, and local variables

2.18 Summary of the better ticket machine

2.19 Self-review exercises

2.20 Reviewing a familiar example

2.21 Calling methods

2.22 Experimenting with expressions: the Code Pad

2.23 Summary

Chapter 3 Object Interaction

3.1 The clock example

Table of Contents

- 3.2 Abstraction and modularization
- 3.3 Abstraction in software
- 3.4 Modularization in the clock example
- 3.5 Implementing the clock display
- 3.6 Class diagrams versus object diagrams
- 3.7 Primitive types and object types
- 3.8 The NumberDisplay class
- 3.9 The ClockDisplay class
- 3.10 Objects creating objects
- 3.11 Multiple constructors
- 3.12 Method calls
- 3.13 Another example of object interaction
- 3.14 Using a debugger
- 3.15 Method calling revisited
- 3.16 Summary

Chapter 4 Grouping Objects

- 4.1 Building on themes from Chapter 3
- 4.2 The collection abstraction
- 4.3 An organizer for music files
- 4.4 Using a library class
- 4.5 Object structures with collections
- 4.6 Generic classes
- 4.7 Numbering within collections
- 4.8 Playing the music files
- 4.9 Processing a whole collection
- 4.10 Indefinite iteration
- 4.11 Improving structurethe Track class
- 4.12 The Iterator type
- 4.13 Summary of the music-organizer project

Table of Contents

4.14 Another example: an auction system

4.15 Summary

Chapter 5 Functional Processing of Collections (Advanced)

5.1 An alternative look at themes from Chapter 4

5.2 Monitoring animal populations

5.3 A first look at lambdas

5.4 The forEach method of collections

5.5 Streams

5.6 Summary

Chapter 6 More-Sophisticated Behavior

6.1 Documentation for library classes

6.2 The TechSupport system

6.3 Reading class documentation

6.4 Adding random behavior

6.5 Packages and import

6.6 Using maps for associations

6.7 Using sets

6.8 Dividing strings

6.9 Finishing the TechSupport system

6.10 Autoboxing and wrapper classes

6.11 Writing class documentation

6.12 Public versus private

6.13 Learning about classes from their interfaces

6.14 Class variables and constants

6.15 Class methods

6.16 Executing without BlueJ

6.17 Further advanced material

6.18 Summary

Chapter 7 Fixed-Size Collections Arrays

Table of Contents

- 7.1 Fixed-size collections
- 7.2 Arrays
- 7.3 A log-file analyzer
- 7.4 The for loop
- 7.5 The automaton project
- 7.6 Arrays of more than one dimension (advanced)
- 7.7 Arrays and streams (advanced)
- 7.8 Summary

Chapter 8 Designing Classes

- 8.1 Introduction
- 8.2 The world-of-zuul game example
- 8.3 Introduction to coupling and cohesion
- 8.4 Code duplication
- 8.5 Making extensions
- 8.6 Coupling
- 8.7 Responsibility-driven design
- 8.8 Localizing change
- 8.9 Implicit coupling
- 8.10 Thinking ahead
- 8.11 Cohesion
- 8.12 Refactoring
- 8.13 Refactoring for language independence
- 8.14 Design guidelines
- 8.15 Summary

Chapter 9 Well-Behaved Objects

- 9.1 Introduction
- 9.2 Testing and debugging
- 9.3 Unit testing within BlueJ
- 9.4 Test automation

Table of Contents

- 9.5 Refactoring to use with streams (advanced)
- 9.6 Debugging
- 9.7 Commenting and style
- 9.8 Manual walkthroughs
- 9.9 Print statements
- 9.10 Debuggers
- 9.11 Debugging streams (advanced)
- 9.12 Choosing a debugging strategy
- 9.13 Putting the techniques into practice
- 9.14 Summary

Part 2 Application Structures

Chapter 10 Improving Structure with Inheritance

- 10.1 The network example
- 10.2 Using inheritance
- 10.3 Inheritance hierarchies
- 10.4 Inheritance in Java
- 10.5 Network: adding other post types
- 10.6 Advantages of inheritance (so far)
- 10.7 Subtyping
- 10.8 The Object class
- 10.9 The collection hierarchy
- 10.10 Summary

Chapter 11 More about Inheritance

- 11.1 The problem: networks display method
- 11.2 Static type and dynamic type
- 11.3 Overriding
- 11.4 Dynamic method lookup
- 11.5 super call in methods
- 11.6 Method polymorphism

Table of Contents

- 11.7 Object methods: toString
- 11.8 Object equality: equals and hashCode
- 11.9 Protected access
- 11.10 The instanceof operator
- 11.11 Another example of inheritance with overriding
- 11.12 Summary

Chapter 12 Further Abstraction Techniques

- 12.1 Simulations
- 12.2 The foxes-and-rabbits simulation
- 12.3 Abstract classes
- 12.4 More abstract methods
- 12.5 Multiple inheritance
- 12.6 Interfaces
- 12.7 A further example of interfaces
- 12.8 The Class class
- 12.9 Abstract class or interface?
- 12.10 Event-driven simulations
- 12.11 Summary of inheritance
- 12.12 Summary

Chapter 13 Building Graphical User Interfaces

- 13.1 Introduction
- 13.2 Components, layout, and event handling
- 13.3 AWT and Swing
- 13.4 The ImageViewer example
- 13.5 ImageViewer 1.0: the first complete version
- 13.6 ImageViewer 2.0: improving program structure
- 13.7 ImageViewer 3.0: more interface components
- 13.8 Inner classes
- 13.9 Further extensions

Table of Contents

13.10 Another example: MusicPlayer

13.11 Summary

Chapter 14 Handling Errors

14.1 The address-book project

14.2 Defensive programming

14.3 Server error reporting

14.4 Exception-throwing principles

14.5 Exception handling

14.6 Defining new exception classes

14.7 Using assertions

14.8 Error recovery and avoidance

14.9 File-based input/output

14.10 Summary

Chapter 15 Designing Applications

15.1 Analysis and design

15.2 Class design

15.3 Documentation

15.4 Cooperation

15.5 Prototyping

15.6 Software growth

15.7 Using design patterns

15.8 Summary

Chapter 16 A Case Study

16.1 The case study

16.2 Analysis and design

16.3 Class design

16.4 Iterative development

16.5 Another example

16.6 Taking things further

Table of Contents

Appendix A: Working with a BlueJ Project

- A.1 Installing BlueJ
- A.2 Opening a project
- A.3 The BlueJ debugger
- A.4 Configuring BlueJ
- A.5 Changing the interface language
- A.6 Using local API documentation
- A.7 Changing the new class templates

Appendix B: Java Data Types

- B.1 Primitive types
- B.2 Casting of primitive types
- B.3 Object types
- B.4 Wrapper classes
- B.5 Casting of object types

Appendix C: Operators

- C.1 Arithmetic expressions
- C.2 Boolean expressions
- C.3 Short-circuit operators

Appendix D: Java Control Structures

- D.1 Control structures
- D.2 Selection statements
- D.3 Loops
- D.4 Exceptions
- D.5 Assertions

Appendix E: Running Java without BlueJ

- E.1 Executing without BlueJ

Table of Contents

E.2 Creating executable .jar files

E.3 Developing without BlueJ

Appendix F: Using the Debugger

F.1 Breakpoints

F.2 The control buttons

F.3 The variable displays

F.4 The Call Sequence display

F.5 The Threads display

Appendix G: JUnit Unit-Testing Tools

G.1 Enabling unit-testing functionality

G.2 Creating a test class

G.3 Creating a test method

G.4 Test assertions

G.5 Running tests

G.6 Fixtures

Appendix H: Teamwork Tools

H.1 Server setup

H.2 Enabling teamwork functionality

H.3 Sharing a project

H.4 Using a shared project

H.5 Update and commit

H.6 More information

Appendix I: Javadoc

I.1 Documentation comments

I.2 BlueJ support for javadoc

Appendix J: Program Style Guide

Table of Contents

J.1 Naming

J.2 Layout

J.3 Documentation

J.4 Language-use restrictions

J.5 Code idioms

Appendix K: Important Library Classes

K.1 The java.lang package

K.2 The java.util package

K.3 The java.io and java.nio.file packages

K.4 The java.util.function package

K.5 The java.net package

K.6 Other important packages

Index