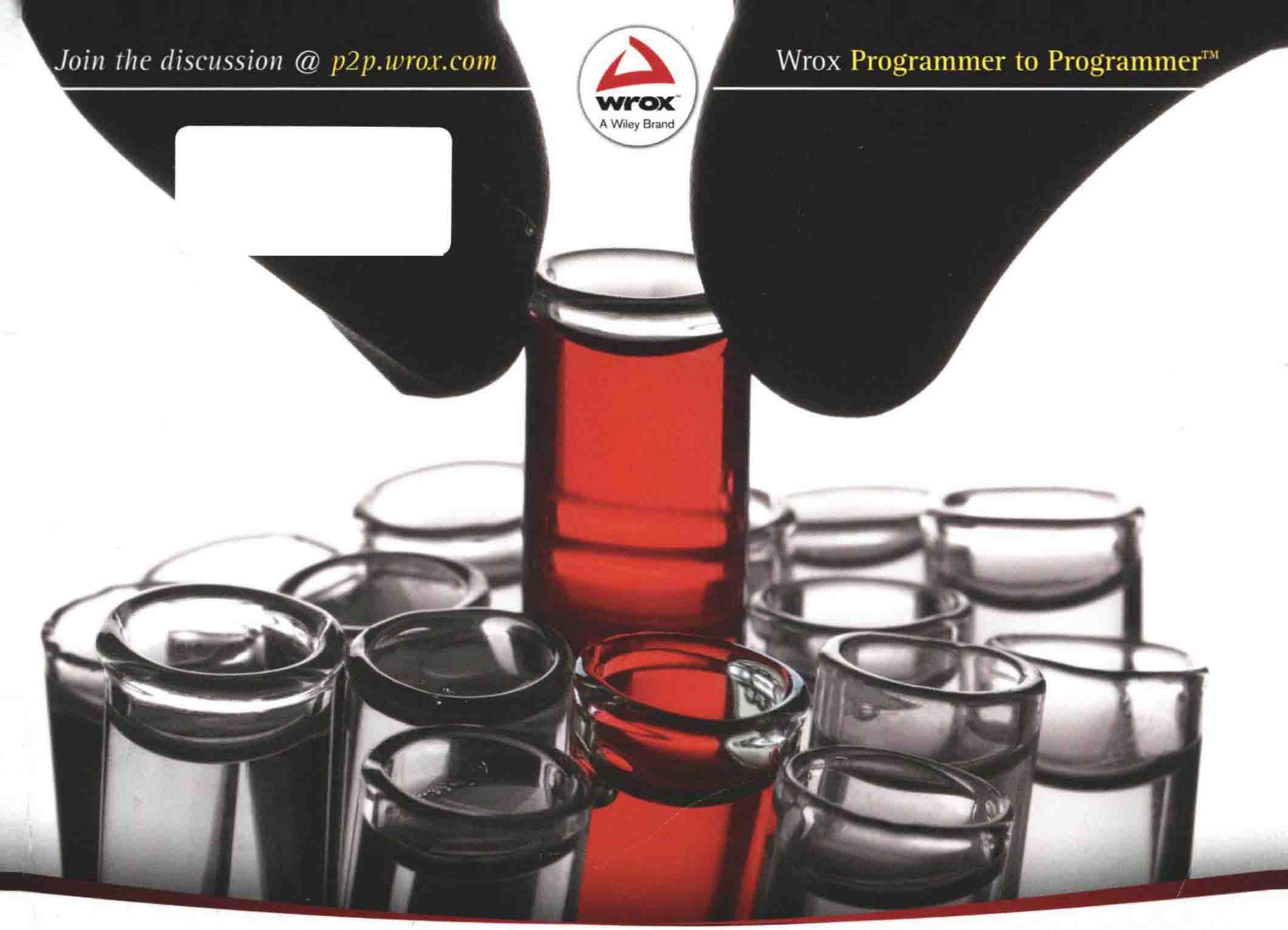
Join the discussion @ p2p.wrox.com

wrox
A Wiley Brand

Wrox Programmer to Programmer™



Beginning Java® Programming

The Object-Oriented Approach

Bart Baesens, Aimée Backiel, Seppe vanden Broucke

BEGINNING

Java® Programming

THE OBJECT-ORIENTED APPROACH

Bart Baesens
Aimée Backiel
Seppe vanden Broucke

wrox™
A Wiley Brand

Beginning Java® Programming: The Object-Oriented Approach

Published by
John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Published simultaneously in Canada

ISBN: 978-1-118-73949-5

ISBN: 978-1-118-73951-8 (ebk)

ISBN: 978-1-118-73935-8 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Web site may provide or recommendations it may make. Further, readers should be aware that Internet Web sites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at http://booksupport.wiley.com. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2013948012

Trademarks: Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc., and/or its affiliates, in the United States and other countries, and may not be used without written permission. Java is a registered trademark of Oracle, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc., is not associated with any product or vendor mentioned in this book.

BEGINNING
Java® Programming

To my lovely wife Katrien and kids, Ann-Sophie,
Victor, and Hannelore. To my parents and
parents-in-law.

—Bart

To my parents my siblings, Karen, Jocelyn, and
Matthew; and my greatest supporter Jing.

—Aimée

To Xinwei, my parents, and sister.

—Seppe

ABOUT THE AUTHORS

PROFESSOR BART BAESENS is a professor at KU Leuven (Belgium) and a lecturer at the University of Southampton (United Kingdom). He has been teaching courses on programming for more than 10 years. His research focuses on big data and analytics, customer relationship management, web analytics, fraud detection, and credit risk management. His findings have been published in well-known international journals and presented at top international conferences. He is also the author of the books *Credit Risk Management: Basic Concepts* (Oxford University Press, 2008) and *Analytics in a Big Data World* (Wiley, 2014). His research is summarized at `www.dataminingapps.com`. He also regularly tutors, advises, and provides consulting support to international firms.

AIMÉE BACKIEL is a PhD researcher in applied economics at KU Leuven in Belgium. Her focus of research is survival analysis, a statistical modeling technique. She was first introduced to Java in 2003 while studying mathematics at the University of Southern Maine. After working some years in data analysis and assessment, she followed up with an Object-Oriented Programming course at KU Leuven, earning a perfect 20/20 for the electronic agenda system she pair-programmed with a partner. Currently, she incorporates Java daily in her own research. In addition, she leads basic programming exercise sessions for students of various disciplines, giving her insight into teaching and learning Java from the ground up and providing a multitude of practical examples for learners who may or may not be technically oriented. Her background provides a balance between a theoretical foundation and an applied business-oriented approach.

SEPPE VANDEN BROUCKE is a PhD researcher in the Department of Decision Sciences and Information Management at KU Leuven. His interest in programming began when he was six years old, starting with languages such as BASIC and Logo. Throughout the following years, Seppe gained experience in a large number of languages, including PHP, Python, Ruby, and Java. He is applying the latter extensively in his day-to-day research activities in order to extract knowledge and insights from business-process-oriented data. His research interests include business process management, process mining, data mining, and machine learning. His work has been published in well-known international journals and presented at top conferences. Seppe has also contributed to and participated in a number of open-source projects. His research is summarized at `www.dataminingapps.com`, and his personal page is located at `www.seppe.net`.

CREDITS

Project Editor
Kelly Talbot

Technical Editors
Andres Almiray
Alan Williamson
Christian Ullenboom

Production Manager
Kathleen Wisor

Copy Editor
Kezia Endsley

Manager of Content Development & Assembly
Mary Beth Wakefield

Marketing Director
David Mayhew

Marketing Manager
Carrie Sherill

Professional Technology & Strategy Director
Barry Pruett

Business Manager
Amy Knies

Associate Publisher
Jim Minatel

Project Coordinator, Cover
Patrick Redmond

Proofreader
Kathy Pope, Word One

Indexer
John Sleeva

Cover Designer
Wiley

Cover Image
©istockphoto.com/rrocio

ACKNOWLEDGMENTS

THIS BOOK WOULD NOT HAVE BEEN POSSIBLE without the support and assistance of many dedicated, helpful, bright, and supportive individuals. Therefore, the authors wish to acknowledge the contributions and assistance of various colleagues, friends, and fellow Java lovers to the writing of this book.

This book is the result of many years of teaching programming courses and working with Java in a research context. We first would like to thank our publisher, Wiley, for accepting our book proposal less than one year ago. Next, we would like to thank our editor, Kelly Talbot, for the excellent collaboration, close follow-up, and timely replies to the many email exchanges we had, as well as the copy, technical, and other assistant editors for the many comments and positive feedback relayed to us, which immensely improved the quality of this book. We are also grateful to the active and lively Java community for providing various user fora, blogs, online lectures, and tutorials, which proved very helpful.

We would also like to acknowledge the direct and indirect contributions of the many colleagues, fellow professors, students, researchers, and friends with whom we have collaborated during the past 10 years.

Last but not least, we are grateful to our partners, parents, and families for their love, support, and encouragement. We hope the readers will enjoy the book and welcome any feedback or suggestions for improvement.

Bart Baesens

Aimée Backiel

Seppe vanden Broucke

January 2015

INTRODUCTION

CONGRATULATIONS! By picking up this book you have made the first step in your voyage toward learning Java. Java is a programming language with a long history, starting with its inception in 1991, when it was still named "Oak," through the first public release (Java 1.0) in 1995 and the newly released Java 8. Its "write once, run anywhere" approach, together with robust language features and numerous libraries led to a spectacular adoption rate. Java is one of the most popular languages in use today, and has been especially successful in enterprise and business environments.

Note, however, that Java is not without its criticism (no programming language is). You might have picked upon the often-repeated criticism that Java is verbose, unsecure, suffering from a slow release cycle, and that it is fading in popularity compared to the interest in new, more exciting languages (Ruby, Erlang, and Haskell, to name a few) by the computer science and programmer communities. The reality, however, tells a different story. Java remains widely taught in schools and universities and is regarded as the language of choice in many organizations. The introduction of Java 7 in 2011 made many tasks simpler, and the availability of many seasoned and stable libraries, tools, and feature-complete IDEs is unmatched by the ecosystem found around other languages. In 2014, Java 8 introduced lambda expressions to streamline code and a reengineered date and time interface that simplifies and improves the internationalization of applications. Java's strong architectural foundations make the language ideally suited for both newcomers and experienced programmers who want to expand their knowledge of Object-Oriented Programming.

Before reaching the end of this book, you will agree that being proficient in Java is a strong skill to possess indeed. We aim to get you started and up to speed as quickly as possible, without making sacrifices in terms of depth and breadth of topics. The goal is not to guide you in simply adding yet another (or first) language to your repertoire, but also to familiarize you with Java's underlying approach toward robust and structured Object-Oriented Programming. As you will see, Java's "verbosity" makes it ideally suited to teach programming best practices in an explicit manner. Providing step-by-step explanations together with many examples—inspired by real-life environments rather than toy exercises—will help you quickly appreciate Java's design and usefulness, and learn that programming in Java can even be great fun!

WHO THIS BOOK IS FOR

Java is a great language to learn for both new and experienced programmers. As such, this book is geared towards a broad audience, including practitioners, analysts, programmers, and students wanting to apply Java in a pragmatic context. It doesn't matter whether you are new to programming and have chosen Java as a place to start or whether you come in from another programming language; Java is a great choice. Many books exist on the topic of Java (just look at the shelf where you found this book!), but we feel that existing offerings reuse the same approach to demonstrate concepts. Therefore, you will not find the archetypical (and honestly, completely useless) "Hello World" example in this book. Instead, we delve into concrete, thought-out examples that illustrate how Java can be useful and used in real life. Whether you are an analyst struggling with spreadsheet formulas to perform a somewhat advanced calculation (there has to be a smarter way, right?), a student wondering how your future employer is using Java, or a hobbyist programmer trying to keep track of stock quotes or a book database, this book aims to familiarize you with all the necessary concepts.

TOPICS COVERED IN THIS BOOK

The topics discussed in this book can, broadly speaking, be outlined in the following three categories. First is a general introduction to programming and Java. The first chapters briefly discuss programming in general, before moving on to a high-level description of Java's history and language features. We also make sure to set up everything you need to get started with Java.

The second part deals with Object-Oriented Programming in Java. The goal is to help readers acquire a strong knowledge of how Object-Oriented Programming works and how Java programs are structured.

The third part is more focused and practical, and shows how you can leverage Java to talk to data sources (such as files, databases, and even web services) and how you can create a graphical user interface around your program logic.

Note that we have not structured the book around the aforementioned three parts. Instead of splitting the book into "theoretical" and "practical" parts, we chose to introduce new concepts step-by-step as they are needed so that you can quickly move on to examples and exercises. The best way to learn is by doing, and this saying particularly holds true when learning to program.

In Chapter 1, we provide a brief general introduction to programming geared toward newcomers and novice programmers. In Chapter 2, we start introducing Java by providing an overview of the language's history, the different technological components that make up the language, the general language structure, and data types. At that point, you will know enough to get started, so that in Chapter 3, we will guide you toward setting up your development environment and trying out some basic examples and exercises. In Chapter 4, we introduce Object-Oriented Programming basics; flow-control statements are covered in Chapter 5. At that point, you will be able to create simple but functional programs. Chapter 6 explains how to catch errors and debug your programs, which will come in handy as you start using resources and coding more advanced programs. At this point, you will have all the necessary components to start looking at some more advanced Object-Oriented Programming concepts in Chapter 7. After this, you'll be ready to tackle more complex interactions with files, external sources, and users. In Chapter 8 you will look at dealing with file-based input and output, including how to load files, perform basic operations, and save the results back to disk. Chapters 9 and 10 build on this by explaining how to interact with databases and web sources. At this point, you will be itching to move away from command-line based applications, and Chapter 11 explains in-depth how to build graphical user interfaces. Chapter 12 concludes the theory by providing an overview of some common architectural patterns (best practices, if you will) used by seasoned Java developers.

By the end of this book, you will have gained a strong knowledge of Java's internals, you will know what is meant by Object-Oriented Programming, know how to debug and deal with errors in your programs, know how to handle file-based input and output, talk to databases, talk to web services, make a full-fledged graphical application, and be familiar with some common and well-known programming "patterns," which are best practices to structure and organize a program's architectural setup.

TOPICS NOT COVERED IN THIS BOOK

This book is not a reference manual. The goal is to get readers acquainted with the basics of Java and Object-Oriented Programming to use within practical applications, not to provide a full overview of Java's API. As such, given the scope of this book, there are some concepts that are not discussed in detail. However, we have taken care to avoid elements you can live without at this point in your Java career.

Working with generics in Java, for example, is not discussed explicitly, but instead explained briefly where needed (when we talk about collections in Java, such as lists or sets, for example). Working with generic classes can be daunting for novice Java programmers, and the Object-Oriented Programming concepts discussed suffice to cover the multitude of use cases. That said, familiarizing yourself with generic types after going through this book should not prove difficult.

Other topics that are not discussed in-depth include networking in Java (socket programming), multi-threaded and concurrent programming, reflection, and the lambda expressions introduced with Java 8. Networking aspects, however, are dealt with from a "higher-level" view. We discuss how to interact with web services, which provides a great starting point for practitioners to load data coming from the web. Concurrent, multi-threaded programming is a beneficial practice when performance and speed becomes an issue in applications, but for most use cases in a practical context, Java performs just fine without having to deal with multi-threading. Additionally, programming in a concurrent fashion introduces some particular challenges and "gotchas" that are unfit for beginning Java programmers to deal with. Reflection is a part of the Java API that allows programmers to examine and "reflect" on Java programs while they are running and perform changes to programs while they are being executed. While very helpful in some cases, it also is out of scope for a beginner's book on Java. Finally, the recently released Java 8 introduces some new concepts, most notably lambda expressions. Java 7 also provides functionality to work with so-called "anonymous classes," which are ad hoc implementations of a base class without a specific name or definition. Other than the fact that these classes can appear somewhat verbose, they are perfectly fine to use instead of lambda expressions (which provide the same functionality in a more concise manner) and are still widely used. We do, however, provide short notes on developments within Java 8 whenever appropriate.

Finally, Java is composed of a certain number of "technologies" (also called platform components). The "standard" Java is denoted as "Standard Edition" (SE), and is the one we tackle here. An Enterprise Edition (EE) also exists, as well as a number of extensions to develop embedded and mobile applications (Embedded Java and Java ME), applications for smart TVs (Java TV), and graphic-intensive applications (Java FX and Java 2D). We do not discuss these extensions in this book, as they have no place in a beginner's book on Java. Readers wanting to apply Java in these specific areas should, however, be ready to move on to these more focused topics after reading this book.

CONVENTIONS

This book applies a number of styles and layout conventions to differentiate between the different types of information.

TRY IT OUTS What Are They?

This book comes with many code examples and exercises that introduce and explain new concepts step-by-step. These are called "Try It Outs" and can be executed directly while reading the chapters or revisited at a later time. Each Try It Out is followed by a How It Works, which is designed to help you understand exactly what's happening in the Try It Out exercise.

In addition to providing extensive examples in the form of Try It Outs, you will also encounter frequent tips, hints, advice, and background information, which are formatted like this:

> ***NOTE*** *Tips, hints, advice, and background information are formatted like this. Reading these segments can help to make concepts clearer. You may also notice a Warning or Common Mistakes heading on these boxes to bring your attention to particular things you should avoid.*

Discussions that extend beyond a short tip or note will look a little bit different.

> **SIDEBAR**
>
> We use this format to explain concepts that require more than a simple note. The details are not necessary to the understanding of the book but offer a more complete discussion on a particular topic.

Finally, program code within this book is formatted in two ways. The first way is as code type: (`/* Like this comment */`). Variables (like *`iAmAVariable`*) appear in italic code type.

Larger code blocks look like the following:

```
public class NoHelloWorld {
    public static void main(String[] args) {
        System.out.println("This is not an Hello World program...");
    }
}
```

> ***NOTE*** *Eclipse is the Integrated Development Environment (IDE) we will be using throughout this book. Don't worry what this means for now. Everything you need to set up in order to follow along is explained in Chapter 3.*