JavaTech

An Introduction to Scientific and Technical Computing with Java

JavaTech is a practical introduction to the Java programming language with an emphasis on the features that benefit technical computing, such as platform independence, extensive graphics capabilities, multi-threading, and tools to develop network and distributed computing software and embedded processor applications.

The book is divided into three parts. The first presents the basics of object-oriented programming in Java and then examines topics such as graphical interfaces, thread processes, I/O, and image processing. The second part begins with a review of network programming and develops Web client-server examples for tasks such as monitoring of remote devices. The focus then shifts to distributed computing with RMI, which allows programs on different platforms to exchange objects and call each other's methods. CORBA is also discussed and a survey of web services is presented. The final part examines how Java programs can access the local platform and interact with hardware. Topics include combining native code with Java, communication via serial lines, and programming embedded processors.

JavaTech demonstrates the ease with which Java can be used to create powerful network applications and distributed computing applications. It can be used as a textbook for introductory or intermediate level programming courses, and for more advanced students and researchers who need to learn Java for a particular task. *JavaTech* is up to date with Java 5.0.

Clark S. Lindsey received his Ph.D. in physics from the University of California at Riverside and has held research positions at Iowa State University, Fermilab, and the Royal Institute of Technology, Sweden. This book grew out of a course in Java programming he developed with Professor Lindblad. He now runs his own company that develops Java applications, Web publications, and educational tools and materials.

Johnny S. Tolliver holds a Ph.D. in Computational Plasma Physics and has worked in fusion energy research, computer security, and trusted operating systems. He is a Sun Certified Java Programmer and has been actively using Java since 1997. He is currently at Oak Ridge National Laboratory, developing Web services software and a GPS vehicle tracking application using GPS-enabled wireless phone handsets and other GPS devices.

Thomas Lindblad received his Ph.D. in physics at the University of Stockholm in 1972 and became associate professor two years later. He is currently a professor in the Department of Physics at the Royal Institute of Technology, Stockholm, and also serves part time as Director of Undergraduate Studies. His research currently concentrates on techniques in image and data analysis in high data rate systems.

JavaTech

An Introduction to Scientific and Technical Computing with Java

**Clark S. Lindsey, Johnny S. Tolliver
and Thomas Lindblad**

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 2RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521821131

First published 2005

Printed in the United Kingdom at the University Press, Cambridge

A catalog record for this publication is available from the British Library

Library of Congress Cataloging in Publication data

ISBN-13 978-0-521-82113-1 hardback
ISBN-10 0-521-82113-4 hardback

Cambridge University Press has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication, and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

Contents

Part I Introduction to Java

Part II Java and the network

Part III Out of the sandbox

Cambridge University Press
0521821134 - JavaTech: An Introduction to Scientific and Technical Computing with Java
Clark S. Lindsey, Johnny S. Tolliver and Thomas Lindblad
Frontmatter
More information

Preface

Java is a serious language suitable for demanding applications in science and engineering. Really, we promise! Java offers a lot more than just those little applets in your Web browser.

In *JavaTech* we focus on how Java can perform useful tasks in technical computing. These tasks might involve an animated simulation to demonstrate a scientific principle, a graphical user interface for an existing C or C++ computational engine, a distributed computing project, controlling and monitoring an experiment remotely via the Internet, or programming an embedded Java hardware processor in a device such as a remote sensor. While other Java books intended for the science and engineering audience concentrate primarily on numerical programming, we take a much broader approach and examine ways that Java can benefit programmers working on many different types of technical applications.

This project grew out of a course given by two of us (C.S.L. and Th.L.) at the Royal Institute of Technology in Stockholm, Sweden in which students of diverse backgrounds followed the class via the Internet. For this type of *distance learning* situation, we developed hypertext instructional material for delivery via the Web browser that allows for a high degree of self-study. This approach works especially well with Java since many of the demonstration programs run as applets within the browser.

This book provides a handy print companion to this hypertext course, which is available online at `www.javatechbook.com`. The book includes additional material that deals with distributed computing techniques based on work done by one of us (J.T.) at Oak Ridge National Laboratory in the USA. Throughout the book we refer to the hypertext materials as the *Web Course*.

Who should use this book

JavaTech targets primarily those who want to learn the Java programming language so as to apply it to practical applications in science and engineering. From the freshman science major to the experienced programmer in a technical field, we believe this book and the Web Course will be helpful.

For those unfamiliar with the language and with object-oriented programming, we begin with a compact introduction to Java. Since Java has grown into a very big field we only touch on the essential elements needed to begin doing useful programming. We include examples of how Java can apply to technical tasks such as histogramming of data and image analysis.

While familiarity with C and C++ will hasten a reader's understanding of Java programming, we do not assume the reader knows these languages.

After the Java introduction we discuss network programming, which we consider to be one of Java's strongest features. We focus particularly on how to build client/server systems for distributed computing applications. If you have a network application, such as the need to monitor remote devices or to give distant users access to a complex simulation running on a central server, the survey here should help you get started. Our aim is to show that you can create powerful network software with Java without needing first to become an authority on all the arcane intricacies of network systems. Java's networking tools and platform portability allow you to focus more on your application than on the underlying mechanisms.

The final part of the book looks at how Java can interact with the local platform, with code in other languages, and with embedded processors. For example, perhaps you have a legacy program in C that represents many years of development and tuning, but it lacks a graphical interface to make it interactive and flexible. We discuss the Java Native Interface (JNI) that allows you to connect your program to Java and to take advantage of the extensive graphical tools available in Java to build an interface around your computational engine in C (or in Fortran via intermediate C code as discussed in the Web Course). You can also add the networking capabilities of Java discussed above. For example, remote clients could connect with your legacy program that runs on a central server.

Hardware microprocessors designed especially to run Java are now widely available. Those who work on embedded processor applications will be interested in our survey of the field of Java processors. In a demonstration program, for example, we show how to connect via a serial port to a microcontroller that is programmed with Java and used to read a sensor.

We look at compact, low-cost platforms that contain Java processors, Ethernet connectors, analog-to-digital inputs, digital-to-analog outputs and other useful features. With such systems you can run servers that allow remote clients to monitor, control, and diagnose an instrument of some kind. This offers the opportunity to those who work with large complex installations, such as an elaborate scientific apparatus or a power plant, to access and control a system at a fine-grained level. We provide a demonstration of a server on such a Java processor platform in which the server responds to a Web browser with an HTML file containing a voltage reading.

Organization and topics

We attempted with this book and Web Course combination to create an innovative and highly flexible approach that allows readers with a diverse range of interests and backgrounds to find and use effectively the materials for their particular needs. The Web Course includes hypertext tutorial materials, many demonstration programs, and exercises. The book compliments the Web Course with more extensive discussions on a range of topics and with tables and diagrams for quick reference.

We follow an example-based teaching approach, using lots of applets and application programs to demonstrate the concepts and techniques described. In addition, we supply a large selection of *starter* programs that provide templates with which readers can quickly begin to develop their own programs.

The chapters in the book correspond directly to those in the Web Course. Note that while one of Java's strongest features is its extensive graphics capability, we do not discuss graphics programming in the first five chapters. Instead we focus on the components and structure of the language. We demonstrate techniques with stand-alone programs (referred to in Java as *applications*) that print to the console and applets that send output to the web browser's Java console window.

The book and Web Course are divided into three parts plus appendices.

Part I Introduction to Java

The 12 chapters in Part I provide an introduction to the Java language. These chapters focus on the Java language but also discuss various topics relevant to applying Java to technical areas. The Web Course expands the introductory material into three tracks:

The *Java Track* provides an introduction to Java programming. The reader can follow this track alone for a quick course in the basics of Java programming. *Supplements* provide additional information on both basic and advanced topics.

The *Tech Track* focuses on topics relevant to general math, science, and engineering applications of Java such as floating-point numbers, random number generators, and image processing.

The *Physics Track* provides an example of how to apply Java to a particular technical subject. The track corresponds to a short course for undergraduate students on the use of numerical computing, simulations, and data analysis in experimental physics.

Part II Java and the network

This part focuses on the application of Java to network programming and distributed computing. It begins with an introduction to TCP/IP programming and then looks at several topics including socket based client/server demonstration programs and distributed computing with RMI, CORBA, and other techniques. An introduction is given to Unified Modeling Language (UML), which leads to better object oriented code design and analysis. A brief overview of web services and XML is also provided.

Part III Out of the sandbox

This part deals with how Java programs can access information and resources on the underlying platforms on which the Java Virtual Machine (JVM) is installed and how the JVM can interact with its local environment. It also reviews implementations of Java in hardware rather than in a virtual machine. Topics include interfacing Java programs to C/C++ and Fortran codes with the Java Native Interface (JNI), communicating with devices via serial/parallel ports, and working with embedded Java processors.

Appendices

Appendices 1 and 2 provide tables of Java language elements and operators, respectively. Appendix 3 gives additional information about floating-point numbers in Java.

Topics not discussed

Java has grown into an enormous industry since it first appeared in the mid-1990s. No single book could possibly do justice to all of the Java classes, packages, tools, techniques, and applications of the language. In fact, there exist many books devoted to individual topics such as Java I/O, graphics, and multithreading. The Java industry expands further every day.

For this book we have chosen what we consider to be an important subset of Java topics relevant to technical applications. Some important topics not treated include:

- Java Enterprise techniques, such as database access and Java application servers
- Security topics such as the Java Cryptography Extension (JCE)
- Java 3D graphics

We do provide in the Web Course a large set of links to references and resources for these and other Java subjects. We also believe that this book provides the reader with a solid base of understanding on which to pursue further learning. All Java programmers must deal with the need to continually learn new classes and APIs (Application Program Interfaces). As we go to press, Sun is about to release Java 2 Standard Edition version 5.0, which contains significant additions to the language. We discuss the most important of these but some are beyond the scope of this book.

We emphasize the use of the web for access to language specifications, online tutorials, and other resources needed to tackle new Java techniques. We include references and web links in each chapter and in the Web Course. You can also find many online resources at `http://java.sun.com`, `java.net`, and `www.ibm.com/developerworks/java/`.

As mentioned in the introduction, we do not delve into numerical programming with Java. We only touch on this subject here while the Web Course *Tech* and

Physics tracks contain several introductory level sections. See the reference list at the end of Chapter 1 for a list of several books that deal extensively with numerical programming in Java.

How to use this book and Web Course

We designed the book and Web Course in a way that lets readers follow individualized paths through the materials. Part I, in particular, allows for a variety of different approaches. You could, for example, study only the Java sections of each chapter and get a fast introduction to the basics of Java programming. You could also study the sections with particular relevance to technical applications (the Web Course expands on these in its *Tech Track*) or, alternatively, you could skip these tech topics in a first pass and return to them later. Those already familiar with Java basics could focus just on the tech-related topics.

You can proceed through the book and Web Course at your own pace and experiment with the many applets and application demonstration programs. There is an emphasis on coding by the reader since ultimately you can only learn Java or any other language by writing lots of programs yourself.

Part II and Part III deal with specialized topics. If you are already familiar with the basics of Java programming, you could proceed directly to the chapter or sub-section of interest in those parts.

One of the most important features of Java is its extensive network programming capability. So we designed the course around the assumption that the reader has easy access to the Internet. Most of the Web Course pages include links to reference and resource materials, especially the tutorials and language specifications on the `http://java.sun.com` website. Rather than reinvent the wheel we try to incorporate resources such as the Sun tutorials in a way that takes best advantage of what is already available.

The Web Course hypertext materials and demonstration codes, along with updates and corrections to the book, are available at the website `www.JavaTechBook.com`. (A mirror site is available at `www.particle.kth.se/~lindsey/JavaCourse/Book/`.)

Note that if we included in the book the source codes for all the demonstration programs, it would be a very long book indeed. Since the source codes are easily available from the Web Course, we often print only “code snippets” rather than entire classes or programs.

Conventions

`Fixed width` style indicates:

- code samples such as: `for (i=0; i < 4; i++) j++;`
- Java class names, variable names, and other code-related terms
- console commands such as: `c:\> java HelloWorld`
- web addresses such as `http://java.sun.com`

In code listings, italicized *`fix width`* indicates that the text is not actually in the code but included to emphasize some aspect of the code or to summarize code that was skipped. We also put the class name in **bold** in the code listings. (Coding style conventions are discussed in Section 5.9.) When discussing a method in the text we may often ignore the argument list for the sake of brevity. So `aMethod (int x, float y, double z)` is abbreviated as `aMethod()`.

In the main text, new terms of particular importance are italicized. The book name and Web Course sections are also italicized.

In Chapter 22 on the Java Native Interface, we use the notation `Xxx` and `xxx` as placeholders to represent the many possible names that can replace the `Xxx` or `xxx`. For example, JNI has a `GetIntField()` method. It also has `GetFloatField()`, `GetDoubleField()`, etc. methods. We refer to these as a group with the `GetXxxField()` notation. Similarly, the `xxx` in `jxxxArray` can be replaced with `int`, `float`, `double`, etc. to produce `jintArray`, `jfloatArray`, `jdoubleArray`, etc.

Java version

The code in *JavaTech* primarily follows that of Java version 1.4 released in 2002, but we discuss the significant enhancements available in the Java 5.0 release where relevant. (This release was under development for at least two years and became available in beta form near the end of the writing of this book.) Since many web browsers currently in use only run Java 1.1 applets and also since some small platforms (e.g. embedded processors) with limited resources only run Java 1.1, we also include in the Web Course some discussion of programming techniques for this version and provide sample codes.

The programs do not usually assume a particular platform and should run on MS Windows, Mac OS X, Linux, as well as Solaris and most Unix platforms.

Acknowledgements

We would like to thank our editors Simon Capelin and Vince Higgs for their help and patience. We thank Roger Sundman and Carl Wedlin of Imsys Technologies for their review of the Java hardware discussion and helpful suggestions. Thanks also to Michele Cianciulli for his comments on the manuscript.

We thank the many students who took our Web Course over the years and gave us a great amount of useful feedback. We especially want to thank one of our first students, Conny Carlberg, who encouraged us by quickly applying Java to his research. Several students at the Royal Institute of Technology (KTH) have used parts of the manuscript of this book, and many of them have come with interesting and useful comments, especially Bruno Janvier and Jaakko Pajunen. At a very early stage, when the Java course was introduced, and when Java was not generally too well known, we received encouraging support from many professors at KTH, the University of Stockholm, and the Manne Siegbahn Institute of Physics.

One of us (C. S. L.) would like to dedicate this book to his wife Kerima who provided great support and encouragement.

One of us (J. S. T.) would like to thank his wife Janey and children Kevin and Chelsea for their enduring patience with a too-often absent or preoccupied husband and father during many months on a project that grew to be longer and more difficult than anyone expected. Thank you.

One of us (Th. L.) makes a dedication to whoever said "do not write any more books, it is a much bigger undertaking than you recall from writing the previous one."