



JavaFX 9 by Example

Rich-client applications for any platform

—

Third Edition

—

Carl Dea
Gerrit Grunwald
José Pereda, Ph.D
Sean Phillips
Mark Heckler

Apress®

JavaFX 9 by Example

Third Edition



Carl Dea

Gerrit Grunwald

José Pereda, Ph.D

Sean Phillips

Mark Heckler

Apress®

JavaFX 9 by Example

Carl Dea
Pasadena, Maryland, USA

José Pereda, Ph.D
Arroyo de la Encomienda, Spain

Mark Heckler
Godfrey, Illinois, USA

Gerrit Grunwald
Münster, Nordrhein-Westfalen, Germany

Sean Phillips
Bowie, Maryland, USA

ISBN-13 (pbk): 978-1-4842-1960-7
DOI 10.1007/978-1-4842-1961-4

ISBN-13 (electronic): 978-1-4842-1961-4

Library of Congress Control Number: 2017952397

Copyright © 2017 by Carl Dea, Gerrit Grunwald, José Pereda, Sean Phillips and Mark Heckler

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Cover image by Freepik (www.freepik.com)

Managing Director: Welmoed Spahr
Editorial Director: Todd Green
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Technical Reviewer: Brian Molt
Coordinating Editor: Jill Balzano
Copy Editor: Kezia Endsley

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit <http://www.apress.com/rights-permissions>.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <http://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484219607. For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

Contents at a Glance

About the Authors.....	xix
About the Technical Reviewer	xxi
Acknowledgments.....	xxiii
Introduction	xxv
■ Chapter 1: Getting Started	1
■ Chapter 2: JavaFX and Jigsaw	47
■ Chapter 3: JavaFX Fundamentals.....	69
■ Chapter 4: Lambdas and Properties	103
■ Chapter 5: Layouts and Scene Builder.....	135
■ Chapter 6: User Interface Controls	171
■ Chapter 7: Graphics	227
■ Chapter 8: JavaFX Printing.....	265
■ Chapter 9: Media and JavaFX.....	283
■ Chapter 10: JavaFX on the Web.....	327
■ Chapter 11: JavaFX 3D	367
■ Chapter 12: JavaFX and Arduino	391
■ Chapter 13: JavaFX on Mobile.....	431

■ **Chapter 14: JavaFX and Gestures..... 467**

■ **Chapter 15: Custom UIs 491**

■ **Chapter 16: Appendix A: References 525**

Index..... 551

Contents

- About the Authors.....xix
- About the Technical Reviewerxxi
- Acknowledgmentsxxiii
- Introductionxxv
- Chapter 1: Getting Started 1
 - Downloading Required Software..... 1
 - Installing the Java 9 Development Kit..... 3
 - Installing the JDK on Microsoft Windows 3
 - Installing the JDK on MacOS X 7
 - Installing the JDK on Linux 11
 - Setting Environment Variables 14
 - Setup Windows Environment Variables 16
 - Setting Up MacOS X/Linux Environment Variables 19
 - Installing Gradle 22
 - Installing the NetBeans IDE..... 23
 - Creating a JavaFX HelloWorld Application..... 29
 - Using the NetBeans IDE..... 30
 - Using an Editor and the Terminal (the Command-Line Prompt) 34
 - Using Gradle on the Command-Line Prompt 38
 - Walking Through the HelloWorld Source Code 41
 - JavaFX Scene Graph..... 42
 - JavaFX Node 42

Packaging a JavaFX Application.....	43
Downloading the Book's Source Code.....	44
Summary	45
■ Chapter 2: JavaFX and Jigsaw	47
What Is Project Jigsaw?	48
Benefits	48
Drawbacks.....	49
Java 9 Migration Path.....	49
History	52
JAR Hell	52
OSGi	53
Maven/Gradle	54
Getting Started	55
What Is the Module Path?.....	56
Module Definition	57
Module Types.....	58
An Example HelloWorld JavaFX 9 Modular Application	63
Create Project Structure	63
Create a Module Definition	63
Create Main Application Code.....	64
Compile Code (Module).....	65
Copy Resources	65
Run Application.....	66
Package Application as JAR	66
Run Application as JAR.....	67
Display Module Description	67
Summary	68

■ Chapter 3: JavaFX Fundamentals	69
JavaFX Lines	69
Drawing Lines	74
Drawing Shapes	78
Drawing Complex Shapes	79
A Complex Shape Example	79
The Cubic Curve	83
The Ice Cream Cone.....	84
The Smile.....	86
The Donut	86
Painting Colors	88
An Example of Color	88
Gradient Color	92
Stop Color	92
Linear Gradient	92
Radial Gradient	93
Semitransparent Gradients.....	94
Reflective Cycle Gradients.....	94
Drawing Text.....	95
Changing Text Fonts	97
Applying Text Effects	100
Summary.....	101
■ Chapter 4: Lambdas and Properties	103
Lambda	103
Lambda Expressions.....	104
Functional Interfaces.....	107
Aggregate Operations.....	108

Default Methods	111
An Example Case: Cats Large and Small	111
Code for the Example	112
Explanation of the Code	116
Properties and Binding	117
UI Patterns	117
Properties	117
Types of JavaFX Properties	118
JavaFX JavaBean	120
Property Change Support	121
Binding	122
Bidirectional Binding	123
High-Level Binding	123
Low-Level Binding	124
A Logon Dialog Example	124
Login Dialog Source Code	126
Explanation of the Code	130
Summary	134
■ Chapter 5: Layouts and Scene Builder	135
Layouts	135
HBox	136
VBox	140
FlowPane	144
BorderPane	144
GridPane	145
Scene Builder	150
Download and Installing Scene Builder	151
Launching Scene Builder	151
A Code Walkthrough	169
Summary	170

■ Chapter 6: User Interface Controls	171
Labels.....	171
Custom Fonts.....	173
Fonts as Icons.....	174
Example: Working with Third-Party Font Packs as Icons.....	174
How It Works.....	181
Buttons	185
Button	185
Check Box.....	186
Hyperlink	187
Radio Button	187
Example: Button Fun.....	189
Button Fun Instructions	189
Source Code of ButtonFun.java	190
How It Works.....	196
Menus.....	198
Creating Menus and Menu Items.....	198
Invoking a Selected MenuItem	199
Example: Working with Menus	200
How It Works.....	202
Additional Ways to Select Menus and Menu Items.....	203
The ObservableList Collection Class	204
Working with ListViews.....	204
Example: Hero Picker.....	205
How It Works.....	208
Working with TableViews	209
What Is a Cell Factory?	209
Making Table Cells Editable.....	210
Example: Bosses and Employees Working with Tables	212

Generating a Background Process	221
Creating a Background Task	221
Example: File Copy Progress Dialog (BackgroundProcesses)	222
How It Works	225
Summary	226
■ Chapter 7: Graphics	227
Working with Images	227
Loading Images	228
Displaying Images	230
A Photo Viewer Example	231
Features/Instructions	231
UML: Class Diagram	233
File Descriptions	234
Source Code	235
Animation	255
What Are Key Values?	255
What Are Keyframes?	255
What Is a Timeline?	256
JavaFX Transition Classes	256
Point-and-Click Game Example	257
Source Code	257
How It Works	261
Compound Transitions	261
PhotoViewer2 Example	262
Summary	264
■ Chapter 8: JavaFX Printing	265
JavaFX Printing	265
JavaFX Print APIs	268

Printer and PrinterJob	268
Query Printer Attributes	270
Configuring a Print Job	272
Printing a Web Page	274
Example WebDocPrinter Application	275
Source Code	277
How Does It Work?	281
Summary	282
■ Chapter 9: Media and JavaFX	283
Media Events	283
Playing Audio	285
An MP3 Player Example	285
MP3 Audio Player Source Code	287
How It Works	301
Playing Video	314
MPEG-4	314
VP6 .flv	314
A Video Player Example	315
Video Player Source Code	316
How It Works	320
Simulating Closed Captioning: Marking a Position in a Video Media	323
Closed Captioning Video Example	323
How It Works	325
Summary	325
■ Chapter 10: JavaFX on the Web	327
JavaFX Web and HTTP2 APIs	328
Web Engine	330
WebEngine's load() Method	330
WebEngine's loadContent() Method	331

HTML DOM Content	331
Obtaining an org.w3c.dom.Document (DOM) Object	331
Using Raw XML Content as a String	332
The JavaScript Bridge	332
Communicating from Java to JavaScript.....	333
Communicating from JavaScript to Java.....	333
Java 9 Module jdk.incubator.httpclient.....	335
Making RESTful Requests.....	339
The HTTP GET Request	340
HTTP POST Request.....	341
WebSockets	342
Viewing HTML5 Content (WebView).....	345
Example: An HTML5 Analog Clock	345
Analog Clock Source Code.....	346
How It Works.....	349
Inkscape and SVG.....	349
WebEvents	350
Weather Widget Example.....	351
One-Liner: Reading an Input Stream into a String.....	353
Source Code	354
How It Works.....	363
Enhancements.....	364
Summary.....	365
■ Chapter 11: JavaFX 3D	367
Basic 3D Scenes in JavaFX	367
A Very Basic 3D Scene Example.....	367
Primitives	369
Adding a Primitive Example	369
Simple Translate and Rotate Example	371
Multiple Primitive Transformation Example	372
All Together Now: Grouped Primitives	373

Interacting with Your Scene	374
Primitive Picking for Primitives	375
First Person Movement Using the Keyboard.....	376
First Person Camera Movement Using the Mouse.....	377
Beyond the Basics.....	379
Custom 3D Objects Using the TriangleMesh Class	380
“Winding” and Wuthering	380
MeshViews and DrawMode	383
Roll Camera!	388
Hit the Lights	389
Summary	390
■ Chapter 12: JavaFX and Arduino	391
The Arduino Board.....	391
Programming the Arduino	393
Arduino Web Editor	394
Arduino IDE	396
Windows	396
MacOS X or Linux	398
Running the IDE	398
The Blink Example	400
Orientation Visualizer Example.....	401
How It Works.....	406
Serial Reading	406
Java Simple Serial Connector.....	406
JavaFX, the Charting API, and Orientation	406
Creating the Module Project.....	407
Serial Communications.....	409
How It Works.....	412
Testing Serial Comms	414
The JavaFX Charts API.....	414
Building and Running the Project	420

How It Works.....	421
Adding More Functionality.....	425
Building and Running the Project.....	427
How It Works.....	428
More Examples.....	430
Summary.....	430
■ Chapter 13: JavaFX on Mobile.....	431
JavaFXPorts: The Port to Mobile.....	431
JavaFXPorts Under the Hood	431
Getting Started with JavaFXPorts.....	432
Hello Mobile World Example.....	433
How Does It Work?	435
Submitting the App to the Stores.....	437
Gluon Mobile	438
The Gluon IDE Plug-Ins	438
Charm Glisten	439
License	441
Example: The BasketStats App.....	441
Creating the Project.....	441
Adding the Model.....	447
Adding the Service	450
Modifying the Main View	453
Modifying the Board View.....	457
Deploy to Mobile.....	464
More Examples.....	465
Summary.....	465
■ Chapter 14: JavaFX and Gestures.....	467
Recognizing Gestures in Your Application	467
Example: Animating Shapes Along a Path Using Touch Events	469
How Does It Work?	472

Touching, Rotating, and Zooming in 3D	473
The Leap Motion Controller	476
How It Works.....	477
Getting Started with the Leap SDK	478
Adding the Leap SDK to a JavaFX Project	479
The Hands Tracking Example.....	479
The LeapListener Class.....	480
The 3D Model Classes	483
The Application Class	485
Building and Running the Project.....	487
More Examples.....	489
Summary	489
■ Chapter 15: Custom UIs	491
Theming	491
Native Look and Feels	493
Web and Mobile Look and Feels.....	495
Applying the JavaFX CSS Theme	497
An Example of Switching Themes	500
JavaFX CSS Styling	505
What Are Selectors?	506
How to Define -fx- Based Styling Properties (Rules).....	511
Obeying the JavaFX CSS Rules.....	512
Custom Controls	513
The LED Custom Control.....	514
Structure of the LED Custom Control Example Code	515
The Properties of the LED Control.....	516
The Initialization Code of the LED Control.....	519
Other Ways to Create a Custom Control.....	523
Summary	523

■ **Chapter 16: Appendix A: References 525**

 Java 9 SDK 525

 Java 9 API Documentation..... 525

 Java 9 Features 525

 Java 9 Jigsaw 526

 IDEs 526

 Deploying Applications 526

 JavaFX 2D Shapes..... 527

 JavaFX Color..... 527

 JavaFX 2.x Builder Classes 527

 JavaFX Printing 527

 Project Lambda 528

 Nashorn..... 529

 Properties and Bindings 529

 Layouts..... 530

 JavaFX Tools..... 530

 Enterprise GUI Frameworks..... 531

 Domain-Specific Languages 532

 Custom UIs 532

 Operating System Style Guidelines 535

 JavaFX Media 535

 JavaFX on the Web 536

 JavaFX 3D 537

 JavaFX Gaming..... 538

 Java IoT and JavaFX Embedded..... 539

 Software and Device Manufacturers..... 540

 JavaFX Communities..... 540

Applications.....	541
Java/JavaFX Books and Magazines	543
Author Blogs.....	544
Tutorials, Courses, Consulting Firms, and Demos	544
Tools, Applications, and Libraries	545
Videos and Presentations on JavaFX	547
Index.....	551

About the Authors



Carl Dea is a principal software engineer. He has been developing software for over 20 years, for many clients from Fortune 500 companies to nonprofit organizations. He has written software ranging from mission-critical applications to ecommerce-based Web applications. His passion for software development started when his middle school science teacher showed him a TRS-80 computer. Carl has been using Java since the very beginning, and he has been a huge JavaFX enthusiast since the early days, when it was its own language called JavaFX script. His current software development interests are UI/UX, game programming, data visualizations, embedded systems, smartphones, AI, and robotics. When Carl is not working, he and his wife enjoy canoeing and day trips to the beach. Carl and his wife are proud parents of their younger daughter who attends Salisbury University. Carl and his wife are also proud of their

older daughter, who is a high school teacher in Anne Arundel County Maryland. Carl loves to code socially at <http://github.com/carldea>. He blogs at <http://carlfx.wordpress.com/>. Carl is also connected to LinkedIn at <http://www.linkedin.com/in/carldea>, and his Twitter handle is @carldea. Carl lives in Pasadena, Maryland, USA.



Gerrit Grunwald is a software engineer with more than 10 years of experience in software development. He has been involved in development of Java desktop applications and controls. His current interests include JavaFX, HTML5, and Swing, especially development of custom controls in these technologies. Gerrit is also interested in Java-driven embedded technologies like JavaSE Embedded on Raspberry Pi, i.MX6, BeagleBone Black, CubieBoard2, and similar devices. He is a true believer in open source software and has participated in popular projects like JFXtras.org as well as his own projects (Enzo, SteelSeries Swing, and SteelSeries Canvas). Gerrit is an active member of the Java community, where he founded and leads the Java User Group Münster (Germany), co-leads the JavaFX and IoT community at Oracle, and is a JavaOne RockStar and Java Champion. He is also a speaker at conferences and user groups internationally.



José Pereda, Ph.D in Structural Engineering, works as a software engineer at Gluon. Being on Java since 1999, he is a JavaFX advocate, developing JavaFX applications for mobile platforms and embedded platforms. He also works on open source projects (JFXtras, FXyz, <https://github.com/jperedadnr>), co-authoring a JavaFX book (JavaFX 8 Introduction by Example), blogging (<http://jperedadnr.blogspot.com.es/>), tweeting (@JPeredaDnr), and speaking at conferences (such as JavaOne, JAX, Jfokus, JavaLand, and Coding Serbia). José lives with his wife and four kids in Valladolid, Spain.



Sean Phillips is a Java software architect currently working on ground systems for Space Science missions at the NASA Goddard Space Flight Center. He has developed embedded systems, along with modeling simulation, and visualization software for projects from commercial and military RADAR, heavy manufacturing, Battlespace Awareness, and Orbital Flight Dynamics. Sean is the lead developer of F(X)yz (<http://birdasaur.github.io/FXyz/>), a free third-party library for JavaFX 3D components and data visualization tools. Sean enjoys sharing his experiments through blog entries on various sites, including Java.net (<https://weblogs.java.net/blogs/sean.mi.phillips>), The NetBeans DZone (<http://netbeans.dzone.com/users/seanmiphillips>), and personal media (Twitter @SeanMiPhillips).

Sean lives in Bowie, Maryland, USA, with his very supportive wife Zulma and sons Sebastian and Sean Alexander.



Mark Heckler is a Principal Technologist & Developer Advocate at Pivotal, a conference speaker, and published author focusing on software development for the Internet of Things and the cloud. He has worked with key players in the manufacturing, retail, medical, scientific, telecom, and financial industries, and with various public sector organizations to develop and deliver critical capabilities on time and on budget. Mark is an open source contributor, author/curator of a developer-focused blog, and on Twitter (@MkHeck). Mark lives with his very understanding wife, three kids, and dog in St. Louis, Missouri, USA.

About the Technical Reviewer



Brian Molt has been a software developer at a manufacturing company in Nebraska for close to 14 years. During his time there, he has programmed with multiple languages on a variety of platforms, including RPGLE on an AS/400, web developing with ASP.NET, desktop application development with JavaFX, and recently REST services with JAX-RS.

Acknowledgments

I would like to thank my amazing wife, Tracey, and my daughters, Caitlin and Gillian, for their loving support and sacrifices. A special thanks to my daughter Caitlin, who helped with illustrations and brainstorming fun examples for the first edition. A big thanks to Jim Weaver for being a great mentor and friend. I would also thank Josh Juneau for his advice and guidance throughout this journey. A big thank you to Brian Molt, for providing excellent feedback and, most of all, for testing my code examples. Thanks also to David Coffin for tech reviewing the second edition. Hats off to my co-authors Gerrit Grunwald, José Pereda, Sean Phillips, and Mark Heckler, for their amazing talents. My co-authors are truly superheroes who came to my rescue to make this book possible.

A huge shout-out to the wonderful people at Apress for their professionalism and support. A special thanks to Jonathan Gennick for believing in me and whipping me into shape again and again. Thanks to Jill Balzano for keeping us on track and getting us across the finish line.

Thanks to all who follow me on Twitter, especially the topics (hash tags) related to JavaFX, UI/UX, and IoT. Also, thanks to the authors (Jim Weaver, Weiqi Gao, Stephen Chin, Dean Iverson, and Johan Vos) of the *Pro JavaFX* book for allowing me to tech review chapters in the first edition. Thanks also to Stephen Chin and Keith Combs for heading up the awesome JavaFX User Group. I want to also thank Rajmahendra Hegde for helping me with the eWidgetFX open source framework effort. Thanks to Hendrik Ebberts for always wanting to team up with me on talks and fun projects at past JavaOne conferences. Most importantly, thanks to all the past JavaFX 1.x book authors that helped inspire me.

A massive thank you to the JavaFX community at large, which there are way too many individuals to name. Lastly, I want to give a big kudos and acknowledgment to the people at Oracle and ex-employees who helped me (directly or indirectly) as JavaFX 2.x, 8, and 9 were being released: Nandini Ramani, Jonathan Giles, Jasper Potts, Richard Bair, Angela Caicedo, Stuart Marks, John Yoon, David Grieve, Michael Heinrichs, David DeHaven, Nicolas Lorain, Kevin Rushforth, Sheila Cepero, Gail Chappell, Cindy Castillo, Scott Hommel, Joni Gordon, Alexander Kouznetsov, Irina Fedortsova, Dmitry Kostovarov, Alla Redko, Nancy Hildebrandt, and all the Java, JavaFX, and NetBeans teams involved.

Whether, then, you eat or drink or whatever you do, do all to the glory of God (1 Corinthians 10:31, NASB).

—Carl Dea

Introduction

Welcome to *JavaFX 9 by Example*.

What is JavaFX?

JavaFX is Java's next-generation graphical user interface (GUI) toolkit that allows developers to rapidly build rich cross-platform applications. Built from the ground up, JavaFX takes advantage of modern GPUs through hardware-accelerated graphics while providing well-designed programming interfaces enabling developers to combine graphics, animation, and UI controls. The new JavaFX 9 is a pure Java language application programming interface (API). The goal of JavaFX is to be used across many types of devices, such as embedded devices, smartphones, TVs, tablet computers, and desktops.

Nandini Ramani, former VP of Development at Oracle, plainly states the intended direction of JavaFX as a platform in the following excerpt from the screencast *Introducing JavaFX*:

The industry is moving toward multi-core/multi-threading platforms with GPUs. JavaFX leverages these attributes to improve execution efficiency and UI design flexibility. Our initial goal is to give architects and developers of enterprise applications a set of tools and APIs to help them build better data-driven business applications.

—Nandini Ramani, former Oracle Corp. VP of Development, Java Client Platform

Before the creation of JavaFX, the development of rich client-side applications involved the gathering of many separate libraries and APIs to achieve highly functional applications. These separate libraries include media, UI controls, Web, 3D, and 2D APIs. Because integrating these APIs together can be rather difficult, the talented engineers at Oracle created a new set of JavaFX libraries that roll up all the same capabilities under one roof. JavaFX is the “Swiss Army Knife” of GUI toolkits (Figure FM-1). JavaFX 9 is a pure Java (language) API that allows developers to leverage existing Java libraries and tools.



Figure FM-1. *JavaFX*

Depending on who you talk to, you are likely to encounter different definitions of “user experience” (or in the UI world, UX). But one fact remains—the users will always demand better content and increased usability from GUI applications. In light of this fact, developers and designers often work together to craft applications to fulfill this demand. JavaFX provides a toolkit that helps both the developer and designer (in some cases, the same person) create functional, yet aesthetically pleasing, applications. Another thing to acknowledge, when developing a game, media player, or the usual enterprise application, is that JavaFX will not only assist in developing richer UIs, but you’ll also find that the APIs are extremely well-designed to greatly improve developer productivity (we’re all about the user of the API’s perspective).

Although this book doesn’t go through an exhaustive study of all of JavaFX 8 and 9’s capabilities, you will find common use cases that can help you build richer applications. Hopefully, this book will lead you in the right direction by providing practical, real-world examples.

Some History

In 2005 Sun Microsystems acquired the company SeeBeyond, where a software engineer named Chris Oliver created a graphics-rich scripting language known as F3 (Form Follows Function). F3 was later unveiled by Sun Microsystems at the 2007 JavaOne conference as JavaFX. On April 20, 2009, Oracle Corporation announced the acquisition of Sun Microsystems, making Oracle the new steward of JavaFX.

At JavaOne 2010, Oracle announced the JavaFX roadmap, which included its plans to phase out the JavaFX scripting language and re-create the JavaFX platform for the Java platform as Java-based APIs. As promised based on the 2010 roadmap, JavaFX 2.0 SDK was released at JavaOne, in October 2011. In addition to the release of JavaFX 2.0, Oracle announced its commitment to take steps to open source JavaFX, thus allowing the community to help move the platform forward. Open sourcing JavaFX will increase its adoption, enable a quicker turnaround time on bug fixes, and generate new enhancements.

Between JavaFX 2.1 and 2.2, the number of new features grew rapidly. Table FM-1 shows the numerous features included between versions 2.1 and 2.2. JavaFX 2.1 was the official release of the Java SDK on a MacOS. JavaFX 2.2 was the official release of the Java SDK on the Linux operating system.

The Java 8 release was announced March 18, 2014. Java 8 has many new APIs and language enhancements, which includes lambdas, Stream API, Nashorn JavaScript engine, and JavaFX APIs. Relating to JavaFX 8, the following features include 3D graphics, rich text support, and printing APIs.

Java 9 is expected to be released some time in September of 2017. Java 9 has so many features, but the most important feature is modularization, better known as *Project Jigsaw*.

What You Will Learn in This Book

In this book, you will be learning JavaFX 9 capabilities by following practical examples. These examples will, in turn, provide you with the knowledge needed to create your own rich client applications. Following Java’s mantra, “Write once, run anywhere,” JavaFX also preserves this same sentiment. Because JavaFX 9 is written entirely in the language of Java, you will feel right at home.

Most of the examples can be compiled and run under Java 8. However, some of them are designed to take advantage of Java 9’s language enhancements. While working through this book with JavaFX 8 and 9, you will see that the JavaFX APIs and language enhancements will help you to become a more productive developer. Having said this, we encourage you to explore all the new Java 8 capabilities.

This book covers JavaFX 8 and 9’s fundamentals, modules, lambdas, properties, layouts, UI controls, printer, animation, custom UIs, charts, media, web, 3D, Arduino, touch events, and gestures.

Who This Book Is For

If you are a Java developer looking to take your client-side applications to the next level, you will find this book your guide to help you begin creating usable and aesthetically pleasing user interfaces. Also, if you are an experienced Java Swing, Flash/Flex, SWT, or web developer who wants to learn how to create high-performance rich client-side applications, then this is the book for you.

How This Book Is Structured

This book is arranged in a natural progression from beginner to intermediate and advanced level concepts. For the Java developer, none of the concepts mentioned in this book should be extremely difficult to figure out. This book is purely example-based and discusses major concepts before demonstrating applications. For each project, after showing the output from executing the code, we'll provide a detailed explanation walking through the code. Each example can be easily adapted to meet your own needs when developing a game, media player, or your usual enterprise application. The more experienced a Java UI developer you are, the more freedom you'll have to jump around to different chapters and examples throughout the book. However, any Java developer can progress through the book and learn the skills needed to enhance their everyday GUI applications.

CHAPTER 1



Getting Started

To get started using this book to learn JavaFX 9, you will want to set up your development environment to compile and run many of its examples. In this chapter you will learn how to install the required software, such as the Java development kit (JDK) and the NetBeans integrated development environment (IDE). After installing the required software, you will begin by creating a traditional JavaFX HelloWorld example. Once you are comfortable with the development environment, I will walk through the JavaFX HelloWorld source code. Finally, you will get a glimpse of how to package your application as a standalone application to be launched and distributed.

I realize many developers are partial to their IDEs and editors, so I've also provided three ways to compile and run the HelloWorld example. In the first way to compile and run code you will be using the NetBeans IDE, in the second way you will learn how to use the command line (terminal window), and thirdly you will be using a build tool called *Gradle* to compile and launch JavaFX applications. The second and third strategies are for those who are not fond of fancy IDEs. If you are more comfortable with the *Eclipse* and *IntelliJ* IDE, you will be happy to know that Gradle can generate the respective project files that will allow your projects to be easily loaded into them.

If you are already familiar with the installation of the Java Development Kit (JDK), Gradle, and the NetBeans IDE, you can skip to Chapter 2, which covers Java 9 Jigsaw. Let's get started!

Downloading Required Software

When using this book, you'll find that many examples may still be written using the Java 8 JDK; however Chapter 2 and a few other chapters will get into Java 9-specific concepts, making the Java 9 JDK a requirement. One of the primary changes to the Java platform is modularity (Jigsaw). Modularity will truly transform your client-side applications. You are probably wondering exactly, "how is it transformed?" Well, imagine building a Java application that occupies a smaller footprint in memory and on disk space. This means users will experience faster load times and a better user experience!

Having said this, go ahead and download Java 9 Java Development Kit (JDK) or later. In this chapter, you will see the steps on installing the JDK on Windows, MacOS X, and Linux. You can download the Java 9 JDK from the following URL:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Another tool that we have not mentioned in prior editions is *Gradle*. Gradle has become an industry standard in building projects and artifacts in the real world. Later, I will walk you through how to install Gradle, but for now head over to gradle.org and look for the download link. At the download link you will be presented with two binary distributions. One is just a binary distribution and the other is a full distribution that includes the source code and documentation. The binary distribution is required to build applications, so you'll want to download it. Go to the following URL to download Gradle's Build tool:

<https://gradle.org/gradle-download>

Installing an integrated development environment (IDE) is optional for the book; however if you choose to use an IDE you will be able to get all the benefits of code highlighting, code completion, incremental debugging, refactoring, and many modern developer conveniences. The top three IDEs used in the wild are NetBeans, IntelliJ, and Eclipse. Although this book walks you through installing the NetBeans IDE, it doesn't mean we are partial to NetBeans or any other IDEs. I've chosen NetBeans because release schedules are usually on par with the major JDK releases and it's quite mature, especially when it comes to tooling and JavaFX application development. NetBeans has numerous JavaFX demo projects (example templates) built-in and the Scene Builder tool. Scene Builder is a graphical tool used to build JavaFX UIs visually.

Another pro tip is learning about Gradle *tasks*, which are capable of generating Java projects for either IntelliJ or Eclipse IDE with a single command. To learn more about Gradle tasks that can automatically set up IntelliJ and Eclipse projects, visit the following links:

- IntelliJ: https://docs.gradle.org/current/userguide/idea_plugin.html
- Eclipse: https://docs.gradle.org/current/userguide/eclipse_plugin.html

The following are URLs to download your favorite IDE:

- NetBeans: <https://netbeans.org/downloads>
- IntelliJ: <https://www.jetbrains.com/idea/download>
- Eclipse: <https://eclipse.org/downloads>

Currently, JavaFX 9 from Oracle corp. runs on the following operating systems:

- Windows OS (Vista, 7, 8, 10) 32- and 64-bit
- MacOS X (64-bit)
- Ubuntu Linux 12.x - 13.x (32- and- 64-bit)

To see a detailed listing of all the supported operating systems, visit the following link:

<http://jdk.java.net/9/supported>

While you may not see your supported operating system or hardware, you will be happy to know about the OpenJDK and OpenJFX projects, which allow you to build Java and JavaFX yourself! To learn more about OpenJDK or OpenJFX, visit the following URLs:

- OpenJDK: <http://openjdk.java.net>
- OpenJFX: <https://wiki.openjdk.java.net/display/OpenJFX/Main>

After downloading the appropriate software versions for your operating system, you will install the Java 9 JDK, Gradle, and/or your chosen IDE.

Installing the Java 9 Development Kit

Once you have downloaded the correct version of Java for your particular operating system, follow the steps outlined in this section to install the Java 9 JDK. I assume you've downloaded Oracle's Java 9 JDK, but if not, go to following location:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

In the following sections, you will follow steps to install Java 9 on the three popular operating systems: Microsoft Windows, MacOS X, and Linux.

Installing the JDK on Microsoft Windows

The following steps use the Java 9 JDK 64-bit version for the Windows 10 operating system. If your Windows operating system is different, refer to the following link for further details.

<http://www.oracle.com/technetwork/java/javase/overview/index.html>

The following are steps to install the Java 9 JDK on the Windows OS:

1. Install the Java 9 JDK by right-clicking the installer's executable (jdk-9-windows-x64.exe) to run as an administrator. Typically, other operating systems will pop up a similar warning dialog box to alert you of the risks of installing or running binary files. Click the Run button to proceed.
2. Begin the installation process by clicking on the Next button, as shown in Figure 1-1.

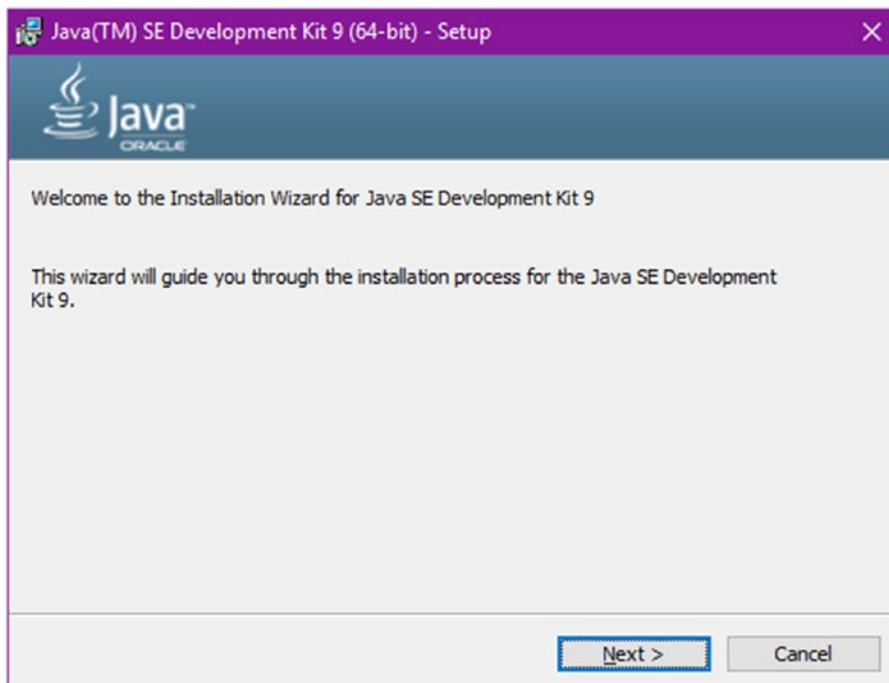


Figure 1-1. Java SE Development Kit 9 installation process

- 3. Next, you'll be presented with the Custom Setup window allowing you to select optional features shown in Figure 1-2. Here, you'll just accept the default selections and click the Next button to continue.

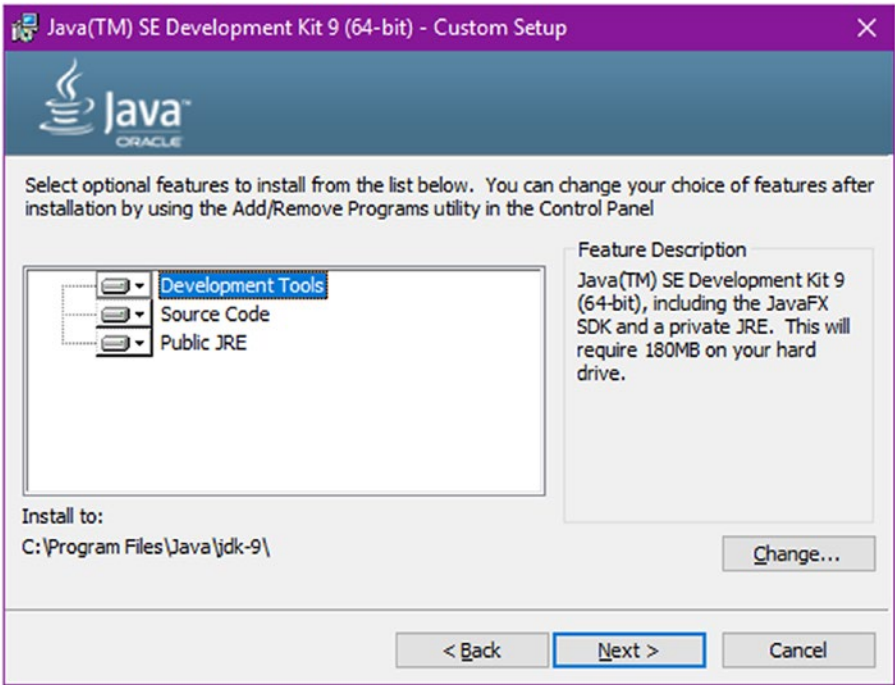


Figure 1-2. Java SE Development Kit 9 optional features

4. Afterward, the installation process will display a progress bar, as shown in Figure 1-3.

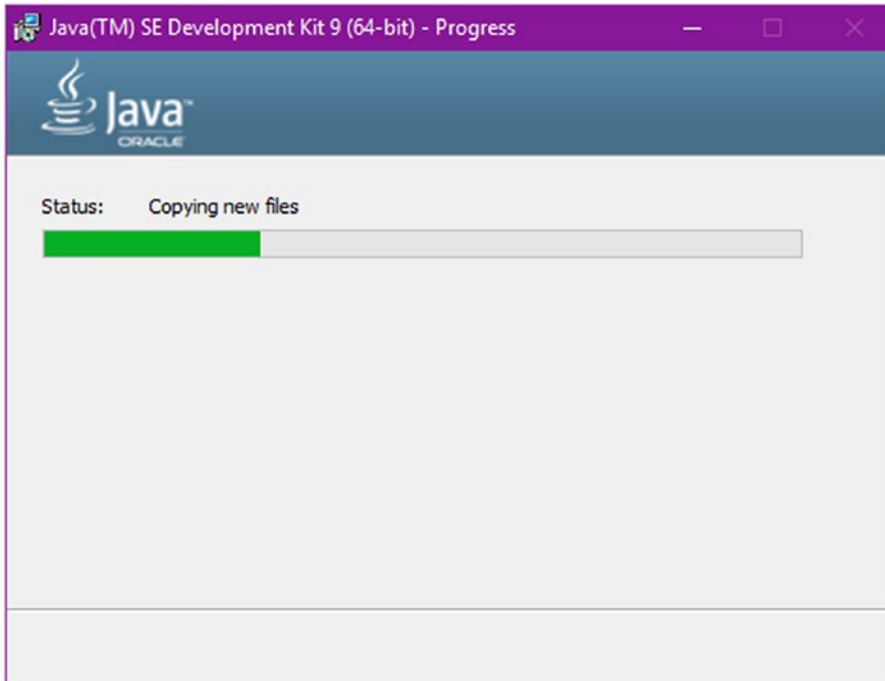


Figure 1-3. Java SE Development Kit 9 installation in progress