THE EXPERT'S VOICE® IN JAVA

# Beginning Java 8 Games Development

*LEARN THE FUNDAMENTALS OF JAVA 8 GAME PROGRAMMING*

Wallace Jackson

Apress®

*For your convenience Apress has placed some of the front matter material after the index. Please use the Bookmarks and Contents at a Glance links to access them.*


Apress®

# Contents at a Glance

# Introduction

The Java Programming Language is currently the most popular object-oriented (OOP) programming language in the world today. Java runs on everything from SmartWatches to HD Smartphones to Touchscreen Tablets to eBook Readers to Game Consoles to SmartGlasses to Ultra-High Definition (UHD) 4K Interactive Television Sets, with even more types of consumer electronics devices, such as those found in the automotive, appliances, health care, digital signage, security, and the home automation market, increasingly adopting the open source Java platform for use in their hardware devices as time goes on.

Since there are literally billions of Java compatible consumer electronics devices, owned by billions of users all over the world, it stands to reason that developing popular Java 8 Games for all of these people could be an extremely lucrative undertaking, given that you have the right game concept, artwork, game design, and optimization work process, of course.

Java 8 (and its multimedia engine, JavaFX 8) code can run on just about every operating system out there, including Windows XP; Vista, 7, 8, and 9; all Linux distributions; 32-bit Android 4 and 64-bit Android 5; Open Solaris; Macintosh OS/X, iOS; Symbian, and Raspberry Pi – it's only a matter of time before the other popular OSes add support for this popular open source programming language. Additionally, every popular Internet browser has Java built in! Java provides the ultimate flexibility in installing software, as an application, or in the browser as an applet. You can even drag a Java application right out of the browser, and have it install itself on that user's desktop! Java 8 is a truly remarkable technology.

There are a plethora of embedded and desktop hardware support levels currently for Java 8 (and for JavaFX 8.0) including the full Java SE 8, Java SE 8 Embedded, Java ME (Micro Edition) 8, and Java ME 8 Embedded, as well as Java EE 8 for Enterprise Application Development. Talk about being able to "code once, deliver everywhere!" That is the dream of every programmer, and Oracle is making it a reality with the powerful Java 8 multimedia programming platform.

This book will go a long way toward helping you to learn exactly how to go about developing Java 8 games, using the Java programming language in conjunction with the recently added JavaFX 8.0 multimedia engine. These Java 8 game applications will be able to run across a plethora of Java compatible consumer electronics devices. Developing Java 8 game applications that play smoothly across all of these different types of consumer electronics devices requires a very specific work process, including asset design, game code design, and optimization, all of which I will be covering during this book.

I wrote the Beginning Java 8 Game Development title from scratch, using a real-world client game project that I am actually working on, and will be delivering to the public sometime in 2015. I am targeting those readers who are Beginning Game Developers, and who had not coded in Java 8 and JavaFX 8.0. These readers are technically savvy, but they are not that familiar with object-oriented computer programming concepts and techniques. Since Java is now at Version 8u40, this book will be more advanced than many of the other Java books out there. Java 8 has added some very advanced features, such as the JavaFX 8.0 API, which gives Java 8 its own multimedia engine, supporting SVG, 2D, 3D, audio, and video media.

I designed this book to contain a comprehensive overview of the optimal Java 8 game development work process. Most beginning Java application development books only cover the language, however. If you really want to become that well-known Java game application developer that you seek to become, you will have to understand as well as master all of the areas of game design, including multimedia asset creation, user interface design, Java 8 Programming, JavaFX 8.0 class usage, and data footprint, memory, and CPU usage optimization. Once you've mastered these areas – hopefully, by the end of this book, you will be able to create the memorable user experience that will be required to create popular, best-selling Java 8 games. You can do it; I know you can!

Java 8 games are not only developed using the NetBeans 8.0 Integrated Development Environment (IDE) alone, but also in conjunction with the use of JavaFX 8 and several other different types of new media content development software packages. For this reason, this book covers the installation and use of a wide variety of other popular open source software packages, such as GIMP 2.8 and Audacity 2.0.6, in conjunction with developing Java 8 game applications using the NetBeans 8.0 IDE and the JavaFX new media engine, which brings the "wow factor" to the Java programming language.

I am architecting this book in this fashion so that you can ascertain precisely how your usage of new media content development software will fit into your overall Java 8 game development work process. This comprehensive approach will serve to set this unique book title distinctly apart from all of those other Java 8 game application development titles that are currently out on the market. The book starts out in Chapter 1 with downloading and installing the latest Java 8 JDK as well as the NetBeans 8.0 IDE, along with several popular open source content development applications.

In Chapter 2, you will learn about NetBeans 8.0, and create your first Java 8 game application, and look at useful NetBeans features, such as code completion and code profiling. In Chapter 3, you will learn about the fundamentals of the Java 8 programming language, which you'll be implementing to create a Java 8 game during the remainder of the book.

In Chapter 4, you will learn all about the JavaFX 8.0 new media engine (API) and how its impressive features can take your Java 8 game development and place it into the stratosphere. In Chapter 5, you will learn all about the JavaFX 8 FXML (Java FX Markup Language) and about the underlying concepts of developing new media assets such as digital audio, digital images, digital video, 2D scalable vector graphics (SVG), and 3D geometry, for use with Java 8 games. In Chapter 6, you will learn about game design concepts, and create the foundation for your Java 8 game, its user interface, and a splashscreen. Thus the first third of this book is foundational material, which you'll need to be able to understand how NetBeans 8.0, Java 8, JavaFX 8.0, and various new media asset types supported by the JavaFX engine function together as a platform.

In Chapter 7 we will start to create the various game engines, starting with the game play loop 60 FPS timing engine, and we will learn about the JavaFX 8 Animation, Timeline, KeyFrame, KeyValue, Interpolator, and AnimationTimer classes, which allow the Java 8 game to tap into the JavaFX pulse event timing engine that gives Java 8 its multimedia power.

In Chapter 8, we will create your game Actor and Hero Java abstract classes, the Actor engine, if you will, which will allow us to create the different types of game play components that we will need for the Java 8 game. This will teach you how to create custom foundational classes for a game project, and you will look at the Node, SVGPath, Shape, Image, and ImageView classes as we incorporate these JavaFX class (object) types into our Java 8 Game Actor design.

In Chapter 9, you will learn how to add interactivity to your Java 8 Game projects, using event handling. We will add an event processing engine, which will process all of the different types of action, key, mouse, and drag events that you are likely to utilize in your Java 8 game development work process in the future when you create your own custom games.

In Chapter 10, you will learn about Java List, Set, and Array classes. These are called Java collections, and we will create a custom Actor management engine, which we will call the CastingDirector class, during this chapter. This will allow you to automate the task of keeping track of the cast of your game for each level, and will be used for collision detection.

In Chapter 11 we will start coding our primary Actor class for the InvinciBagel character, and add Java 8 code that controls movement on the screen, so that we can start to work on fusing character animation with game player key use so that we can allow our game players to control the InvinciBagel character completely. This involves "wiring up" the Bagel class to the GamePlayLoop (game play timing class created in Chapter 7) class, so we can start working in the fourth dimension of time.

In Chapter 12 you will use your Actor and Hero abstract classes that you created in Chapter 8 to create the InvinciBagel primary character and his Bagel.java class, as well as learn how to implement code that sets the boundaries for your Java 8 game, so that the Actor does not go off the screen, forcing him stay inside of the field of play for the game.

In Chapter 13 you will add different InvinciBagel sprite image states into your Java 8 game, and when these are combined with the movement you coded in Chapters 11 and 12, allow your InvinciBagel character to run, jump, fly, land, wait impatiently to be moved, and even turn sideways to evade bullets.

In Chapter 14, you will create a series of Prop classes that will allow you to place fixed props and obstacles into your Java 8 game levels. You will learn how to use one digital image asset to create four different scenery props, using the JavaFX ability to flip and mirror your image assets around either (or both of) their X and Y axes.

In Chapter 15, you'll implement your Java 8 game audio engine, using the JavaFX AudioClip class, which allows digital audio sequencing to be integrated into your Java 8 game play, taking it an order of magnitude higher, by stimulating the aural senses of your game player. You'll learn how to optimize digital audio assets so well, that you will not have to use any lossy compression, giving you perfect audio samples, and showing you exactly how much of the system's memory your audio assets will be using.

In Chapter 16, we'll start getting into advanced topics, such as designing collision polygons using SVG data and the GIMP 2.8 and PhysicsEditor software packages. We will also learn about the JavaFX Bounds and Node classes, and how collision detection is accomplished for Java 8 game development, using the .getBoundsInLocal() and .getBoundsInParent() method calls, in conjunction with the Node.intersects() and Shape.intersect() method calls.

In Chapter 17, we will pull everything together, and focus solely on implementing your game play. You will create Actor subclasses for Treasure, Projectile, and Enemy, and create an auto-attack engine that will turn a game player's PC or mobile device into his or her adversary. We look at the most advanced topics, such as physics and AI, during this chapter, after which you will have enough of a foundation to create your own Java 8 games, using your own intellectual property!

This book attempts to be the most comprehensive Java 8 game application development programming title on the market, by covering most, if not all, of the major Java 8 and JavaFX classes that will need to be used to create Java 8 Game Applications. Some of these include the Image, ImageView, Group, Node, StackPane, Scene, Stage, Application, ListArray, HashSet, Arrays, AudioClip, MediaPlayer, URL, Button, Shape, HBox, SVGPath, Insets, AnimationTimer, and more.

If you're looking for the most comprehensive, up-to-date overview of the Java 8 programming language for games, including JavaFX 8.0 and NetBeans 8.0 IDE all seamlessly integrated with new media content development work processes, as well as a "soup to nuts" knowledge about how to optimally use these technologies in conjunction with the leading open source new media game content design and development tools, then this book will really be of significant interest to you.

It is the intention of this book to take you from being a Beginner in Java 8 game application development to a solid intermediate knowledge level regarding Java 8, NetBeans 8, and JavaFX 8.0 game application development. Be advised that this book, even though it's ostensibly a Beginner title, contains a significant amount of technical knowledge. All of the work processes that are described during the book may well take more than one read through to assimilate into an application development knowledge base (your quiver of technical knowledge). It will be well worth your time, however, rest assured.

CHAPTER 1

# Setting Up a Java 8 Game Development Environment

Welcome to the book *Beginning Java 8 Games Development*! Let's get started by creating a solid development software foundation for use with this book. The core of this foundation will be **Java SDK** (**Software Development Kit**) **8**, also called **JDK (Java Development Kit) 8**. I will also set you up with **NetBeans IDE 8.0** (**Integrated Development Environment**), which will make coding Java 8 games much easier. After that, I will introduce you to the latest open-source new media content creation software packages for digital illustration (Inkscape), digital imaging (GIMP [GNU Image Manipulation Program]), digital video (EditShare Lightworks), digital audio (Audacity), and 3D modeling and animation (Blender). At the end of the chapter, I will also suggest some other professional-level software packages that you should consider adding to the professional game development workstation that you will be creating over the course of this chapter.

To get the best results from all this free, professional-level software, you will want to have a modern, **64-bit** workstation with at least 4GB of system memory (6GB or 8GB would be even better) and a **multicore** processor (central processing unit [CPU]), such as an AMD FX-6300 (hexa-core), AMD FX-8350 (octa-core), or Intel i7 (quad-core). Workstations such as these have become commodity items and can be purchased at Walmart or `Pricewatch.com` at an affordable price.

The first thing that you will do in this chapter is make sure that you have **removed** any of the **outdated versions** of Java, such as Java 7 or Java 6, or any outdated versions of NetBeans, such as NetBeans 7 or NetBeans 6. This involves **uninstalling** (removing or deleting completely) these older development software versions from your workstation.

You will do this using the Windows program management utility **Programs and Features**, which can be found in the Windows operating system (OS) **Control Panel** suite of **Windows OS Management Utilities**. There are similar utilities on the Linux and Mac platforms, if you happen to be using one of these less commonly used OSs. Because most developers use Windows 7, 8, or 9, you will be using the Windows 64-bit platform for the examples in this book.

Next, I will show you where exactly to go on the Internet to get these software packages, so get ready to fire up your speedy Internet connection so that you can download nearly a gigabyte of all-new game content production software! After you download the latest versions of all this software, you will install the programming and content development packages and configure them for use with this book.

The order in which you perform these software installations is important, because Java JDK 8 and Java 8 Runtime Environment (JRE) form the foundation of NetBeans IDE 8.0. This is because NetBeans IDE 8.0 was originally coded using the Java programming language, so you will see just how incredibly professional a piece of software can be using this language. Thus, the Java 8 software will be the first software you install.

After you install Java 8, you will then install NetBeans 8.0, so that you have a graphical user interface (GUI), on top of the Java programming language, which will make the Java software development work process easier. After you have these two primary software development tools installed, you will get a plethora of new media content creation software packages, which you can use in conjunction with Java 8 and NetBeans 8.0 to create 2D and 3D games.

# Prepare a Workstation for Java 8 Game Development

Assuming that you already have a professional-level workstation in place for new media content development and game development, you need to remove all the outdated JDKs and IDEs and make sure that you have the latest V8 (not the drink, silly!) Java and NetBeans software installed on your system and ready to go. If you are new to this and do not have a game-appropriate workstation, go to Walmart or Pricewatch.com, and purchase an affordable multicore (use a 4-, 6- or 8-core) 64-bit computer running Windows 8.1 (or 9.0 if it is available) that has 4GB, 6GB, or 8GB of DDR3 (1333 or 1600 memory access speed) system memory at the very least and a 750GB, or even 1TB, hard disk drive.

The way that you remove old software is through the Windows **Control Panel** and its set of utility icons, one of which is the **Programs and Features** icon (Windows 7 and 8), displayed in Figure 1-1. Note that in earlier versions of Windows, this utility icon may be labeled differently, probably as something like **Add or Remove Programs**.

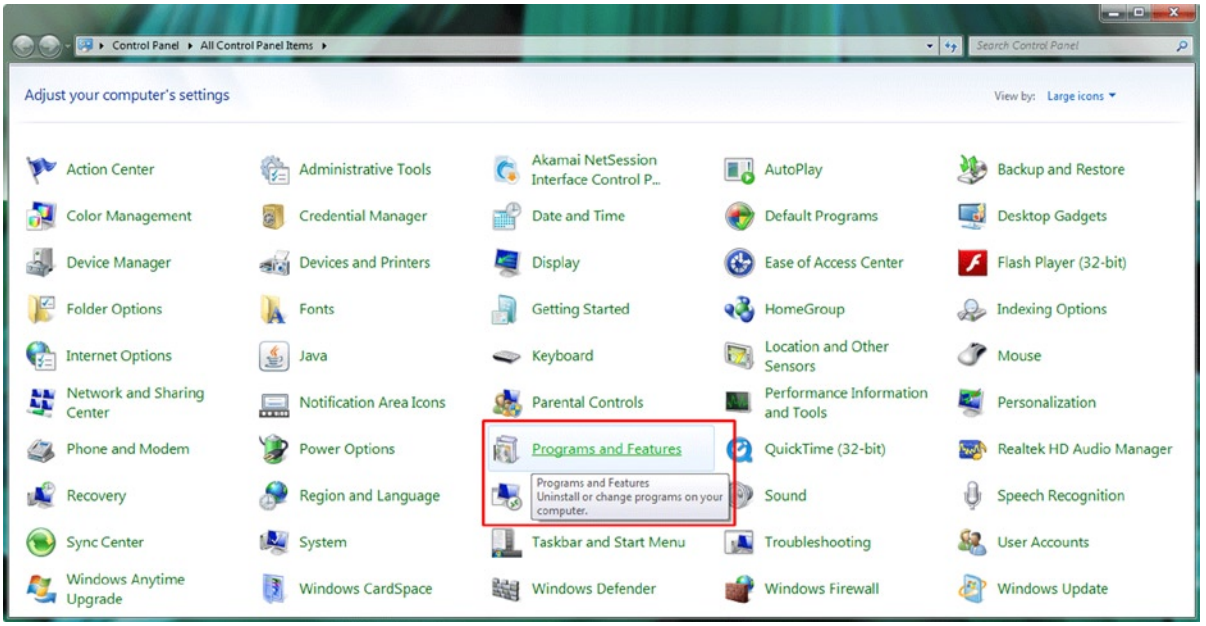


***Figure 1-1.*** *Use the Programs and Features utility icon to uninstall or change programs on your computer workstation*

Click the Programs and Features link, or double-click the icon in previous versions of Windows, to launch the utility. Then, scroll down to see if you have any old versions of the Java development tools (Java 5, Java 6, or Java 7) installed on your workstation. Note that if you have a brand new workstation, you should find no preinstalled versions of Java or NetBeans on your system. If you do find them, return the system, as it may have been used previously!

As you can see in Figure 1-2, on my Windows 7 HTML5 development workstation, I had an older version of Java, Java 7, installed (on November 29, 2013), taking up 344MB of space. To remove a piece of software, **select it** by clicking it (it will turn light blue), and then click the **Uninstall** button, shown at the top of the figure. I left the **tool tip**, which says, "**Uninstall this program,**" showing in the screenshot so that you can see that if you **hover** your mouse over anything in the Programs and Features utility, it will tell you what that feature is used for.

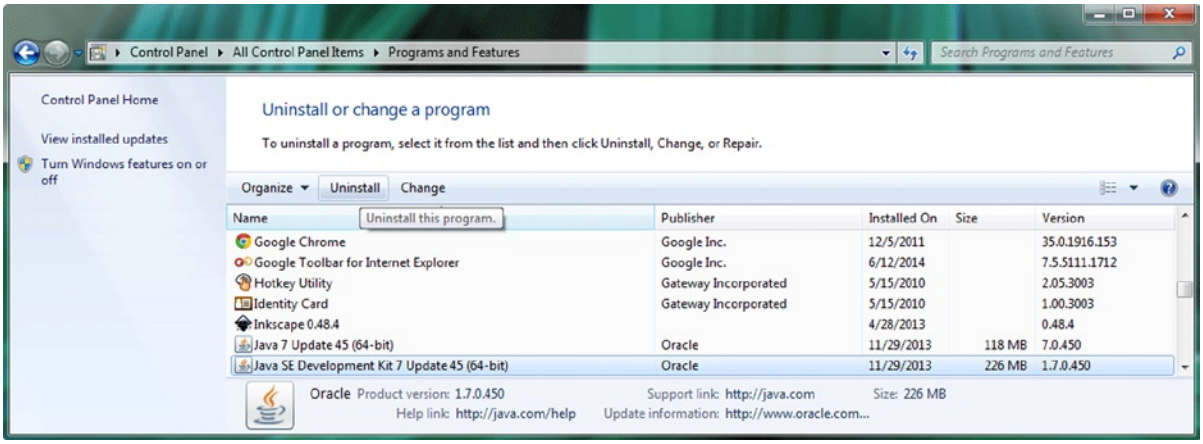


***Figure 1-2.*** *Select any version of Java older than the current version (Java 8), and click the Uninstall button at the top*

Once you click the Uninstall button, the utility will remove the older version of Java. If you want to keep your old Java project files, make sure to back up your Java project files folder (if you have not done so already, that is). Make sure that you back up your workstation's hard disk drive regularly so that you do not lose any of your work.

Also make sure that you uninstall all versions of Java; in my case, there were 64-bit Java 7 update 45 and Java SDK 7u45, used to run or execute IDEs, such as NetBeans (or Eclipse), that were coded using the Java programming language.

Next, you will want to ascertain if there are any older versions of the NetBeans IDE on your workstation. In my case, as you can see in Figure 1-3, there was indeed a NetBeans 7 IDE installation currently on my 64-bit Windows 7 workstation. I selected this for removal and then clicked the **Uninstall/Change** button, shown at left, which brought up a custom **Uninstall Summary** dialog, shown at right.

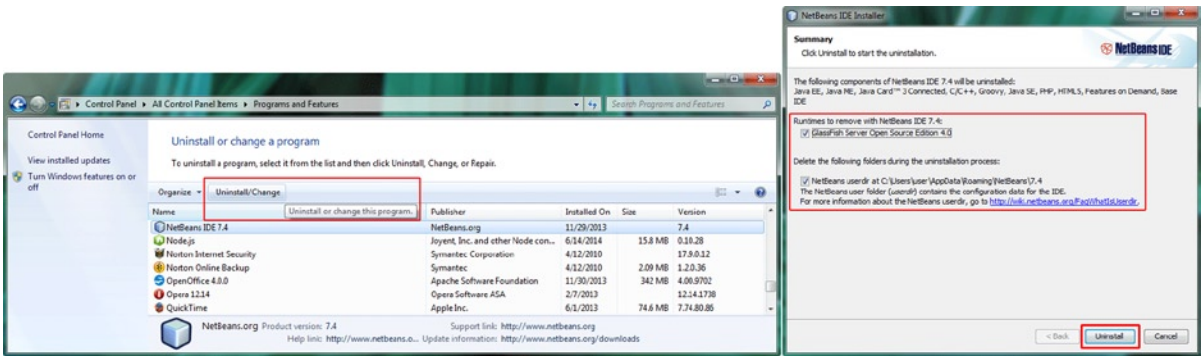


***Figure 1-3.*** *Find and select any version of NetBeans that is older than version 8.0; also, uninstall old GlassFish versions*

Manufacturers (in this case, the NetBeans development team) can create custom Uninstall Summary dialogs for their products to use during the uninstall process, as you can see here. This dialog allows you to select whether you want to uninstall GlassFish Server 4 and the NetBeans **UserDir Configuration** folder. Because you are installing new versions of NetBeans and GlassFish, select both check boxes, and then click the **Uninstall** button.