Making Everything Easier!™

JavaFX®

FOR DUMMIES®

A Wiley Brand

Learn to:

- **Install the JavaFX development tools, including the JavaFX SDK, the Java SE JDK, and NetBeans**
- **Develop desktop, mobile, and browser apps**
- **Use HTML5 and CSS3 with JavaFX**

Doug Lowe
Author of Java All-in-One For Dummies

Get More and Do More at Dummies.com®

Start with **FREE** Cheat Sheets

Cheat Sheets include

- Checklists
- Charts
- Common Instructions
- And Other Good Stuff!

To access the Cheat Sheet created specifically for this book, go to
www.dummies.com/cheatsheet/javafx

Get Smart at Dummies.com

Dummies.com makes your life easier with 1,000s of answers on everything from removing wallpaper to using the latest version of Windows.

Check out our

- Videos
- Illustrated Articles
- Step-by-Step Instructions

Plus, each month you can win valuable prizes by entering our Dummies.com sweepstakes. *

Want a weekly dose of Dummies? Sign up for Newsletters on

- Digital Photography
- Microsoft Windows & Office
- Personal Finance & Investing
- Health & Wellness
- Computing, iPods & Cell Phones
- eBay
- Internet
- Food, Home & Garden

Find out "HOW" at Dummies.com

**Sweepstakes not currently available in all countries; visit Dummies.com for official rules.*

by Doug Lowe

FOR
DUMMIES®
A Wiley Brand

JavaFX® For Dummies®

Published by: **John Wiley & Sons, Inc.,** 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

Published simultaneously in Canada

Library of Congress Control Number: 2014941051

ISBN 978-1-118-38534-0 (pbk); ISBN 978-1-118-41743-0 (ebk); ISBN 978-1-118-42166-6

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Table of Contents

Introduction

In the beginning there was *AWT,* the *Abstract Window Toolkit.* AWT was Java's first system for displaying window-based user interfaces in Java. AWT begat *Swing,* which soon became the preferred way to create user-friendly applications in Java.

But then there was *JavaFX,* the worthy successor to the GUI throne. JavaFX is designed to create stunning user interfaces that can run on a wide variety of devices, including traditional desktop and portable computers, tablets, smartphones, TV set-top boxes, game consoles, and many other types of devices.

Until recently, JavaFX was the red-headed stepchild of the Java world. It co-existed with Java, but wasn't an official part of Java. But beginning with Java version 8, JavaFX is now fully integrated into Java. And while JavaFX and Swing coexist today, Oracle has made it clear that Swing is in its twilight and JavaFX represents the future of user-interface programming.

So you're holding the right book in your hands. JavaFX is an essential skill for every Java programmer to have at his or her disposal, and this book will help you master that skill.

About This Book

This isn't the kind of book you pick up and read from start to finish, as if it was a cheap novel. If I ever see you reading it at the beach, I'll kick sand in your face. Beaches are for reading romance novels or murder mysteries, not programming books.

Assuming, then, that you have found a more suitable location to read this book, you can, if you want, read it straight through starting with Chapter 1 and finishing with Chapter 20. However, this sequence isn't necessary. If you are brand new to JavaFX programming, I suggest you read at least Part I in sequence so that you'll gain a basic understanding of how JavaFX works. But after you have the basics down, you can read the chapters in whatever sequence makes sense for you. If you need to know about adding effects to a shape, skip straight to Chapter 14. For information about about animation, skip ahead to Chapter 17.

You don't have to memorize anything in this book. It's a need-to-know book: You pick it up when you need to know something. Need a reminder on how to rotate a shape? Pick up the book. Can't remember the details of the `TilePane` class? Pick up the book. After you find what you need, put down the book and get on with your life.

This book works like a reference. Start with the topic you want to find out about. Look for it in the Table of Contents or in the index. The Table of Contents is detailed enough that you can find most of the topics you're looking for. If not, turn to the index, where you can find even more detail.

Of course, the book is loaded with information — so if you want to take a brief excursion into your topic, you're more than welcome. If you want to know the big picture on the scene graph, read Chapter 7. But if you just want a reminder on how to set the maximum scene size, read just the section on the `Scene` class.

Whenever I describe sample Java code, I present it as follows:

```
@override public void start(Stage primaryStage)
```

And Java class names, keywords, or other language elements are always shown in `monospace type`.

Foolish Assumptions

In this book, I make very few assumptions about what you already may or may not know about JavaFX. But I do have to make two basic assumptions:

- You own or have access to a computer on which Java JDK 8 has been installed or on which you have permission to install.

  JavaFX 8 is an integral part of JDK 8, so JDK 8 is a requirement for figuring out JavaFX. If you have not yet installed it, you'll find instructions on how to do so in Chapter 1.

- You know the basics of Java programming.

  If you're new to Java, may I suggest one of two books: my own *Java All-In-One For Dummies,* 4th Edition, or Barry Burd's *Java For Dummies*, 6th Edition. Both are published by Wiley.

There are no other prerequisites to this book.

How This Book Is Organized

This book is organized into five parts. Here's a brief description of what you find in each part.

Part I: Getting Started with JavaFX

This part contains the information you need to get started with JavaFX programming. After a brief introduction to what JavaFX is and why it's so popular, you discover the basics of creating simple JavaFX programs. You figure out how to create simple JavaFX scenes populated with common controls such as labels, text field, and buttons. Then, you find out how to write programs that respond to user input, such as when the user clicks a button or enters text into a text field. And finally, you read how to use basic layout managers to control the arrangement of controls in your JavaFX scene.

Part II: JavaFX Controls

The chapters in this part focus on the various types of controls you can use in a JavaFX application. Chapter 7 starts by explaining the details of how the JavaFX scene graph works and presents the details of the class hierarchy used by the various controls. Then, the remaining chapters in this part present information about specific types of controls, ranging from check boxes and radio buttons to tables and menus.

Part III: Enhancing Your Scenic Design

The chapters in this part help you improve the appearance of your applications. First, you read about additional types of layout managers that give you more precise control over the way your user interface is arranged. Then, you discover how to use CSS styles to apply formatting details. Next, you figure out how to incorporate simple shapes into your scenes. And finally, you can read about JavaFX's special effects, which let you embellish your display with shadows, motion blurs, and so on.

Part IV: Making Your Programs Come Alive

The chapters in this part focus on various ways to make your programs more responsive and engaging. You discover how to work with *properties,* which you can use to make one part of your user interface respond to changes in another part of your user interface. Then, you discover how to incorporate media including sound and video. Next, you figure out how to create sophisticated animations that make the objects on the screen dance about. And finally, you read how to create programs that respond to multi-finger gestures on touch-enabled devices.

Part V: The Part of Tens

This wouldn't be a *For Dummies* book without a Part of Tens. Each of the chapters here presents ten items of special interest. Chapter 19 presents ten additional JavaFX controls that didn't fit in Part II. And Chapter 20 presents ten steps to creating a JavaFX application that displays a three-dimensional scene.

Icons Used in This Book

Like any *For Dummies* book, this book is chock-full of helpful icons that draw your attention to items of particular importance. You find the following icons throughout this book:

Danger, Will Robinson! This icon highlights information that may help you avert disaster.

Did I tell you about the memory course I took?

Pay special attention to this icon; it lets you know that some particularly useful tidbit is at hand.

Hold it — overly technical stuff is just around the corner. Obviously, because this is a programming book, almost every paragraph of the next 400 or so pages could get this icon. So I reserve it for those paragraphs that go into greater depth, down into explaining how something works under the covers — probably deeper than you really need to know to use a feature, but often enlightening. You also sometimes find this icon when I want to illustrate a point with an example that uses some Java feature that hasn't been covered so far in the book, but that is covered later. In those cases, the icon is just a reminder that you shouldn't get bogged down in the details of the illustration, and instead focus on the larger point.

Beyond the Book

A lot of extra content that you won't find in this book is available at `www.dummies.com`. Go online to find the following:

- **Online articles covering additional topics at**

  ```
  www.dummies.com/extras/javafx
  ```

  Here you find articles covering additional features of JavaFX that didn't quite fit in the book.

- **The Cheat Sheet for this book is at**

  ```
  www.dummies.com/cheatsheet/javafx
  ```

  Here you'll find a convenient summary of some of the most important JavaFX classes.

- **Code listings for this book at**

  ```
  www.dummies.com/extras/javafx
  ```

  All the code listings used in this book are available for download.

- **Updates to this book, if I have any, are also available at**

  ```
  www.dummies.com/extras/javafx
  ```

Where to Go from Here

Yes, you *can* get there from here. With this book in hand, you're ready to dive right into to the cool and refreshing water of the JavaFX pool. Browse through the Table of Contents and decide where you want to start. Be bold! Be courageous! Be adventurous! And above all, have fun!

Part I

Getting Started with JavaFX

Visit www.dummies.com for great Dummies content online.

In this part . . .

- Figuring out a basic program
- Handling events
- Displaying simple scenes
- Arranging nodes
- Responding to input controls
- Visit `www.dummies.com` for great Dummies content online.

Chapter 1

Hello, JavaFX!

In This Chapter

- Getting a quick overview of what JavaFX is and what you can do with it
- Looking at a basic JavaFX program
- Downloading, installing, and configuring Java 8 so you can build your own JavaFX programs
- Building a JavaFX program the hard way, using nothing but Notepad and a command prompt
- Using TextPad to simplify JavaFX programming
- Using an IDE, such as Eclipse or NetBeans, for JavaFX programming

Welcome to the wonderful world of JavaFX programming!

This chapter offers a gentle introduction to JavaFX programming. In the next few pages, you find out what JavaFX is, where it came from, and where it's going. You see an example of the classic Hello, World! program implemented in JavaFX. And you discover how to set up your computer to develop your own JavaFX programs using several popular development tools for JavaFX.

Incidentally, I assume that you're already somewhat familiar with Java programming. You don't need to be a master programmer by any means, but you should have a solid understanding of the basics, such as creating programs that work with variables and statements (such as `if` and `for`) as well as creating your own classes and using the various classes that are part of the Java API (Application Programming Interface). I don't take the time to explain such basics in this book, so if you need an introduction to Java before you dive into the details of JavaFX, I suggest you get a copy of my masterpiece, *Java All-in-One For Dummies* (Wiley Publishing, Inc.).

The intent of this chapter is to get you ready to start learning how to write JavaFX programs. As such, you see a brief example of a simple JavaFX program in this chapter, which might not make complete sense at this early

stage of your JavaFX journey. Please don't become discouraged. In Chapter 2, I dissect that simple JavaFX program line-by-line so you can see what makes it tick. For this chapter, I focus on the high-level details of what JavaFX is, what you can do with it, and how to get your computer set up for JavaFX programming.

All the code listings used in this book are available for download at `www.dummies.com/extras/javafx`.

What Is JavaFX?

Simply put, JavaFX is a collection of Java packages that lets you add fancy graphical user interfaces to your Java applications. With JavaFX, you can create traditional windows-style user interfaces that include familiar controls such as labels, buttons, text boxes, check boxes, drop-down lists, and so on. But you can also adorn these user interfaces with fancy effects such as light sources, perspective, and animation. Hence the *FX* in *JavaFX*.

Prior to JavaFX, the main way to create graphical user interfaces in Java was through the Swing API. JavaFX is similar to Swing in many ways, so if you've ever used Swing to create a user interface for a Java program, you have a good head start at learning JavaFX.

JavaFX has been around as an add-on package for a while, but beginning with Java version 8, JavaFX is now an official standard part of the Java platform. Thus, after you install the Java 8 Development Kit (*JDK 8*), you can begin developing your own JavaFX applications with your favorite development tools. Later in this chapter, you discover how to download and install JDK 8, and you figure out how to create a simple JavaFX program using three popular Java development tools: TextPad, Eclipse, and NetBeans.

Because JavaFX is now a standard part of Java, you can run your JavaFX programs on any device that includes version 8 of the Java Runtime Environment (JRE). That includes computers, tablet devices, smartphones, and any other device that can support JDK8.

Oracle has announced that JavaFX will eventually replace Swing. Although Swing is still supported in Java 8 and will be supported for the foreseeable future, Oracle is concentrating new features on JavaFX. Eventually, Swing will become obsolete.

Perusing the Possibilities of JavaFX

One of the basic strengths of JavaFX is its ability to let you easily create complicated graphical user interfaces with all the classic user interface gizmos everyone knows and loves. Thus, JavaFX provides a full range of controls — dozens of them in fact, including the classics such as buttons, labels, text boxes, check boxes, drop-down lists, and menus, as well as more exotic controls such as tabbed panes and accordion panes. Figure 1-1 shows a typical JavaFX user interface that uses several of these control types to create a form for data entry.

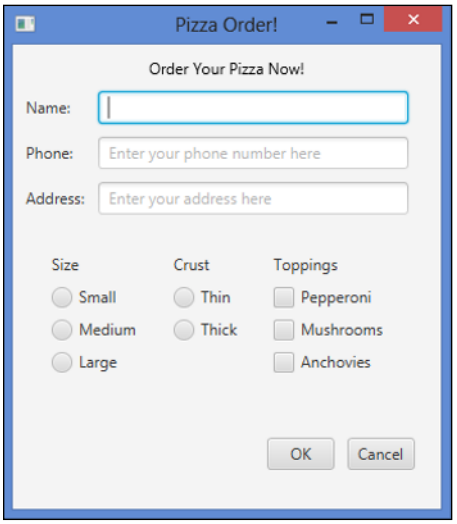

Figure 1-1: A typical JavaFX program.

Truthfully, the data-entry form shown in Figure 1-1 isn't very remarkable. In fact, you can easily create data-entry forms like this using Swing with about the same amount of effort. The real advantages of using JavaFX over Swing don't become apparent until you start using some of the more advanced JavaFX features.

For starters, consider the general appearance of the data-entry form shown in Figure 1-1. The appearance of the buttons, labels, text fields, radio buttons, and check boxes are a bit dated. The visual differences between the dialog box shown in Figure 1-1 and one you could've created in Visual Basic on a Windows 95 computer 20 years ago are minor.

Where JavaFX begins to shine is in its ability to easily allow you to improve the appearance of your user interface by using *Cascading Style Sheets (CSS)*. CSS makes it easy to customize many aspects of the appearance of your user

interface controls by placing all the formatting information in a separate file dubbed a style sheet. A *style sheet* is a simple text file that provides a set of rules for formatting the various elements of the user interface. You can use CSS to control literally hundreds of formatting properties. For example, you can easily change the text properties such as font, size, color, and weight, and you can add a background image, gradient fills, borders, and special effects such as shadows, blurs, and light sources.

Figure 1-2 shows a variation of the form that was shown in Figure 1-1, this time formatted with CSS. The simple CSS file for this form adds a background image, enhances the text formatting, and modifies the appearance of the buttons.

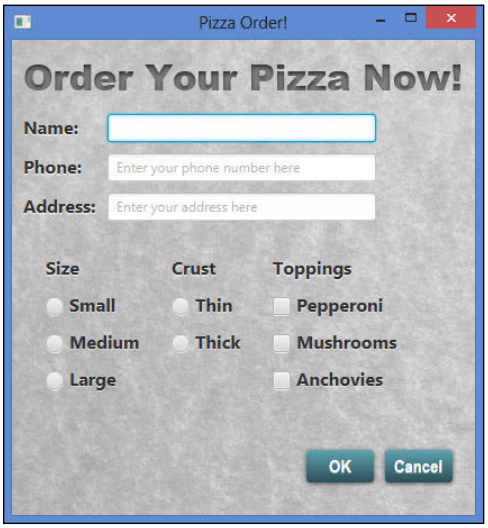

Figure 1-2: JavaFX lets you use CSS to specify formatting for user interface elements.

Besides CSS, JavaFX offers many other capabilities. These are the most important:

- **Visual effects:** You can add a wide variety of visual effects to your user interface elements, including shadows, reflections, blurs, lighting, and perspective effects.
- **Animation:** You can specify animation effects that apply transitions gradually over time.
- **Charts:** You can create bar charts, pie charts, and many other chart types using the many classes of the `javafx.scene.chart` package.
- **3-D objects:** You can draw three-dimensional objects such as cubes, cylinders, spheres, and more complex shapes.

- **Touch interface:** JavaFX can handle touchscreen devices, such as smartphones and tablet computers with ease.
- **Property bindings:** JavaFX lets you create *properties,* which are special data types that can be bound to user interface controls. For example, you can create a property that represents the price of an item being purchased and then bind a label to it. Then, whenever the value of the price changes, the value displayed by the label is updated automatically.

You discover all these features and more in later chapters of this book. But for now, it's time to have a look at a simple JavaFX program so you can get a feel for what JavaFX programs look like.

Looking at a Simple JavaFX Program

Figure 1-3 shows the user interface for a very simple JavaFX program that includes just a single button. Initially, the text of this button says `Click me please!` When clicked, the text of the button changes to `You clicked me!` If you click the button again, the text changes back to `Click me please!` Thereafter, each time you click the button, the text cycles between `Click me please!` and `You clicked me!`

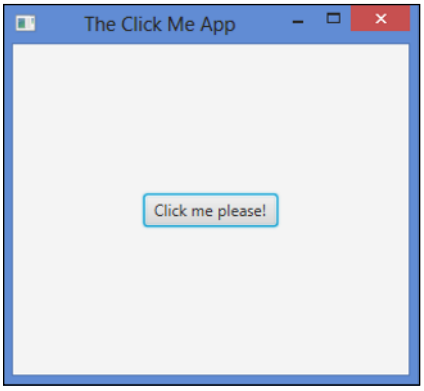

Figure 1-3: The Click Me program.

To give you an idea of what JavaFX programming looks like, Listing 1-1 shows the complete listing for this program. I won't explain the details of how this program works — I examine this program in painstaking detail in Chapter 2. For now, I just want you to get the big picture to give you a feel for what JavaFX programming looks like.

Listing 1-1: The Click Me Program

```
import javafx.application.*;
import javafx.stage.*;
import javafx.scene.*;
import javafx.scene.layout.*;
import javafx.scene.control.*;

public class ClickMe extends Application
{
    public static void main(String[] args)
    {
        launch(args);
    }

    Button btn;

    @Override public void start(Stage primaryStage)
    {
        // Create the button
        btn = new Button();
        btn.setText("Click me please!");
        btn.setOnAction(e -> buttonClick());

        // Add the button to a layout pane
        BorderPane pane = new BorderPane();
        pane.setCenter(btn);
        // Add the layout pane to a scene
        Scene scene = new Scene(pane, 300, 250);

        // Finalize and show the stage
        primaryStage.setScene(scene);
        primaryStage.setTitle("The Click Me App");
        primaryStage.show();
    }

    public void buttonClick()
    {
        if (btn.getText() == "Click me please!")
        {
              btn.setText("You clicked me!");
        }
        else
        {
              btn.setText("Click me please!");
        }
    }
}
```

The following paragraphs give a brief explanation of the key elements of the Click Me program:

- As with any other Java program, JavaFX programs begin with a slew of `import` statements that reference the various packages that will be used by the program.

  For this example, five packages are imported. Most JavaFX programs will require these five packages as well as additional packages that provide more advanced features.

- All JavaFX programs extend a core class named `Application`, which provides the basic functionality of the program. When you extend the `Application` class, you must override a `start` method; JavaFX calls this method when the application starts.

- Like any Java program, a JavaFX program must have a `main` method. In a JavaFX program, the `main` method simply calls the `launch` method of the `Application` class, which in turn launches the application and calls the `start` method.

- The user interface elements of a JavaFX program are arranged in a hierarchy of containers. At the highest level is a *stage,* which represents a window. Within the stage is a *scene,* which contains user interface controls. The controls themselves (such as buttons, labels, drop-down lists, and so on) are usually contained in one or more *layout panes* that govern the positional layout of the controls.

  If you study the code in the `start` method, you see that these elements are built from the bottom up:

  - A button is created.
  - The button is added to a layout pane (specifically, a `StackPane`, which is one of several types of layout panes available).
  - The layout pane is added to a scene and then the scene is added to the stage.
  - The stage's `show` method is called to display the application's GUI (Graphical User Interface).

- The `buttonClick` method is called whenever the user clicks the button. This method examines the current text displayed by the button and changes the text accordingly. Thus, each time the user clicks the button, the button's text changes from `Click me please!` to `You clicked me!` or vice-versa.

Please don't worry if you find some (or even all) of this program confusing at this point. My intent for this chapter is simply to give you a peek at a simple JavaFX program, but not to overwhelm you with the details of how this program works. As I mention earlier, I will review the details of this program line-by-line in Chapter 2.

In the remaining sections of this chapter, you figure out how to download, install, and configure the Java Development Kit and how to compile and test the Click Me program using popular Java development tools.

Downloading and Installing JavaFX

Actually, the above heading is a bit of a trick. Prior to Java 8, JavaFX was a separate entity from Java. Thus, to use JavaFX, you had to download and install a separate JavaFX package. But beginning with Java 8, JavaFX is now an integral part of Java. So if you've downloaded and installed Java 8, you already have JavaFX.

In the following sections, I discuss how to download, install, and configure the Java 8 Development Kit (JDK 8) so that you can code and test JavaFX programs. If you've already installed JDK 8, you can skip the rest of this section.

Downloading JDK 8

To get to the download page, point your browser to `http://java.oracle.com/technetwork/java` and then follow the appropriate links to download the JDK 8 for your operating system.

When you get to the Java download page, you find links to download the JDK or the JRE. Follow the JDK link; the JRE link gets you only the Java Runtime Environment, not the complete Java Development Kit.

The JDK download comes in two versions:

- The online version requires an active Internet connection to install the JDK.

  The offline version lets you download the JDK installation file to your computer and install it later.

I recommend that you use the offline version; it installs faster, and you can reinstall the JDK later if you need to without downloading it again.

Installing JDK 8

After you download the JDK file, you can install it by running the executable file you downloaded. The procedure varies slightly depending on your operating system, but basically, you just run the JDK installation program file after you download it, as follows:

- **On a Windows system,** open the folder in which you saved the installation program and double-click the installation program's icon.
- **On a Linux or Solaris system,** use console commands to change to the directory to which you downloaded the file and then run the program.
- **On a Mac,** open the Downloads window and double-click the JDK `.dmg` file you downloaded. A Finder window appears containing an icon of an open box. Double-click this icon to launch the installer.

After you start the installation program, it prompts you for any information that it needs to install the JDK properly, such as which features you want to install and what folder you want to install the JDK in. You can safely choose the default answer for each option.

Setting the path

After you install the JDK, you need to configure your operating system so that it can find the JDK command-line tools. To do that, you must set the `Path` environment variable — a list of folders that the operating system uses to locate executable programs. To do this on a Windows system, follow these steps. You must be logged in as an administrator to make the changes described in this procedure.

1. **Open the Control Panel.**
   - *On a Windows 7 or earlier system,* open the Start menu and choose Control Panel.
   - *On a Windows 8 or later system,* click the Start button or press the Windows key, type **Control Panel**, and then press Enter.
2. **Double-click the System icon.**

   The System Properties page appears.
3. **Click the Advanced System Settings link and then click the Environment Variables button.**

   The Environment Variables dialog box appears, as shown in Figure 1-4.

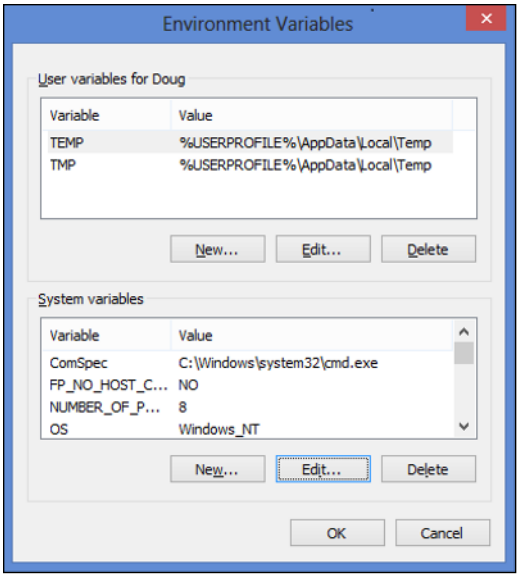

Figure 1-4: The Environment Variables dialog box.

4. **In the System Variables list, scroll to the `Path` variable, select it, and then click the Edit button.**

   A little dialog box pops up to let you edit the value of the `Path` variable.

5. **Add the JDK `bin` folder to the beginning of the `Path` value.**

   Use a semicolon to separate the `bin` folder from the rest of the information that may already be in the path.

   Note: The name of the `bin` folder may vary on your system, as in this example:

   ```
   c:\Program Files\Java\jdk1.8.0\bin;other directories...
   ```

6. **Click OK three times to exit.**

   The first OK gets you back to the Environment Variables dialog box; the second OK gets you back to the System Properties page; and the third OK closes the System Properties page.

For Linux or Solaris, the procedure depends on which shell you're using. For more information, consult the documentation for the shell you're using. Note that this step is not necessary on Mac systems.

Developing the Click Me Program with Notepad

After you install JDK 8, JavaFX is at your disposal. Strictly speaking, the only other tools besides JDK 8 you need to develop Java programs is a text editor and access to a command prompt. With the text editor, you create the Java source file, saving the file with the extension `.java`. Then, at the command prompt, you use Java's command-line tools to compile and run the program.

Windows comes with the free text-editor Notepad that is adequate enough for creating simple Java source files. *Notepad* is a generic text editor that doesn't know anything about the peculiarities of Java source code. As a result, Notepad doesn't give you any assistance with details such as indenting, matching up left and right braces, or drawing your attention to syntax errors.

Nor will Notepad give you any help with compiling, running, or debugging a Java program. But Notepad does have the advantage of being free and simple to use. And, it's already on your computer, so there's nothing else to install.

Here are the steps for creating the Click Me program using Notepad and Java's command-line tools:

1. **Start Notepad.**

   To do that in Windows 7 or 8:

   a. *Click the Start button (or press the Windows key on your keyboard).*

   b. *Type **notepad** and then press Enter.*

   Notepad comes to life, presenting you with an empty text editing window.

2. **Type the Click Me program text shown in Listing 1-1 into the editing window.**

   Be sure to type the text exactly as it appears in the listing. When you're done, carefully review your work to make sure you typed it correctly.

   Figure 1-5 shows how the Click Me program appears when correctly entered into Notepad. (Note that the Notepad window shown in the figure is not large enough to display the entire source file; you must scroll the window to see the entire file.)

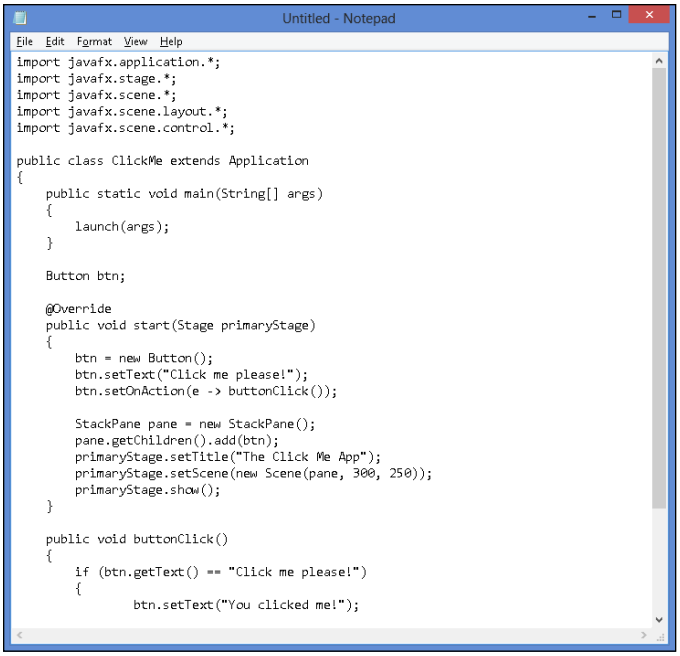

Figure 1-5: The Click Me program in Notepad.

3. **Choose File➪Save to save the file using the name `ClickMe.java`.**

   You can save the file in any folder you wish, but it is very important that the name be exactly `ClickMe.java`.

   The name of a Java source file must exactly match the name of the class it contains, right down to capitalization. Thus, if you save the file as `clickme.java` instead of `ClickMe.java`, the Click Me program won't work.

4. **Open a command prompt window.**

   In Windows 7 or 8, to open the window:

   a. *Click the Start button or press the Windows key*

   b. *Type* cmd *and press Enter.*

5. **Use the `cd` command to change to the folder in which you saved the source file in Step 3.**

   For example, if you saved the file in `C:\Java`, enter the following command:

```
cd C:\Java
```