12.1 Introduction

A **graphical user interface** (**GUI**) presents a user-friendly mechanism for interacting with an app. A GUI (pronounced “GOO-ee”) gives an app a distinctive “look-and-feel.” GUIs are built from **GUI components**—also called *controls* or *widgets* (short for window gadgets). A GUI component is an object with which the user interacts via the mouse, the keyboard or another form of input, such as voice recognition.

Look-and-Feel Observation 12.1

Providing different apps with consistent, intuitive user-interface components gives users a sense of familiarity with a new app, so that they can learn it more quickly and use it more productively.

History of GUI in Java

Java’s original GUI library was the Abstract Window Toolkit (AWT). Swing was added to the platform in Java SE 1.2. Until recently, Swing was the primary Java GUI technology. Swing

will remain part of Java and is still widely used. We discuss Swing in online Chapters 26 and 35.

JavaFX is Java's GUI, graphics and multimedia API of the future. Sun Microsystems (acquired by Oracle in 2010) announced JavaFX in 2007 as a competitor to Adobe Flash and Microsoft Silverlight. JavaFX 1.0 was released in 2008. Prior to version 2.0, developers wrote JavaFX apps in JavaFX Script, which compiled to Java bytecode, allowing JavaFX apps to run on the Java Virtual Machine. Starting with version 2.0 in 2011, JavaFX was reimplemented as Java libraries that could be used directly in Java apps. Some of the benefits of JavaFX over Swing include:

- JavaFX is easier to use—it provides one API for client functionality, including GUI, graphics and multimedia (images, animation, audio and video). Swing is only for GUIs, so you need to use other APIs for graphics and multimedia apps.
- With Swing, many IDEs provided GUI design tools for dragging and dropping components onto a layout; however, each IDE produced different code (such as different variable and method names). JavaFX Scene Builder (Section 12.2) can be used standalone or integrated with many IDEs and it produces the same code regardless of the IDE.
- Though Swing components could be customized, JavaFX gives you complete control over a JavaFX GUI's look-and-feel (Chapter 13) via Cascading Style Sheets (CSS)—the same technology used to style web pages.
- JavaFX has better threading support, which is important for getting the best application performance on today's multi-core systems.
- JavaFX uses the GPU (graphics processing unit) for hardware-accelerated rendering.
- JavaFX supports transformations for repositioning and reorienting JavaFX components, and animations for changing the properties of JavaFX components over time. These can be used to make apps more intuitive and

easier to use.

- JavaFX provides multiple upgrade paths for enhancing existing GUIs—Swing GUI capabilities may be embedded into JavaFX apps via class `SwingNode` and JavaFX capabilities may be embedded into Swing apps via class `JFXPanel`.

This chapter introduces JavaFX GUI basics—we present a more detailed treatment of Java FX GUI in the next chapter. Chapter 22 discusses graphics and multimedia. We placed the Java How to Program, 10/e Swing and Java 2D chapters on the book's Companion Website—see the inside front cover for Companion Website access instructions.