Welcome to the dynamic world of Android smartphone and tablet app development with the Android Software Development Kit (SDK), the Java™ programming language and the rapidly evolving Android Studio Integrated Development Environment (IDE). 

*Android How to Program, 3/e* presents leading-edge mobile computing technologies for upper-level college courses. Many of the Android techniques we present also apply to Android Wear and Android TV app development, so after reading this book, you’ll be well prepared to investigate how to develop apps for these platforms.

The opportunities for Android app developers are enormous. Sales of Android devices and app downloads have been growing exponentially. The first-generation Android phones were released in October 2008. According to IDC, as of June 2015, Android had 82.8% of the global smartphone market share, compared to 13.9% for Apple and 2.6% for Microsoft.¹ Over one billion Android devices shipped in 2014 alone.² At the 2015 Google I/O conference, Google announced that in the prior 12 months there had been 50 billion app installs from Google Play™—Google’s marketplace for Android apps.³ Fierce competition among popular mobile platforms and carriers is leading to rapid innovation and falling prices. In addition, competition among the hundreds of Android device manufacturers is driving hardware and software innovation within the Android community.

**Architecture of Android How to Program, 3/e**

*Android How to Program, 3/e* was formed by merging

- our professional book *Android 6 for Programmers: An App-Driven Approach, 3/e*, published by the Pearson Technology Group
- condensed, introductory object-oriented Java programming content from our college textbook *Java How to Program, 9/e*, published by Pearson Higher Education
- hundreds of Android short-answer questions and app-development exercises we created for this book—most are in the book and many of the short-answer questions are in the test-item file for instructors.

We scoured the Android material, especially the fully coded Android apps, and enumerated the Java features that you’ll need to build these and similar apps. Then we extracted the corresponding Java content from *Java How to Program, 9/e*. That’s a 1500-page book, so it was challenging to whittle down that much content and keep it friendly for programming novices.

---

When you study the Android content, you’ll be thinking like a developer from the start. You’ll build real stuff and you’ll face the kinds of challenges professional developers must deal with. We’ll point you to the online documentation and forums where you can find additional information and get answers to your questions. We also encourage you to read, modify and enhance open-source code as part of your learning process.

**App-Driven Approach**

At the heart of the book is our *app-driven approach*—we present concepts in the context of *eight complete working Android apps*. We begin each of the app chapters with an *introduction* to the app, an *app test-drive* showing one or more *sample executions*, and a *technologies overview*. We build the app’s *GUI* and *resource files*. Then we proceed with a detailed *code walkthrough* of the app’s source code in which we discuss the programming concepts and demonstrate the functionality of the Android APIs used in the app. All the source code is available at [http://www.deitel.com/books/AndroidHTP3](http://www.deitel.com/books/AndroidHTP3) and at the book’s Companion Website [http://www.pearsonhighered.com/deitel](http://www.pearsonhighered.com/deitel). We recommend that you have the source code open in the IDE as you read the book. Figure 1 lists the book’s apps and the key technologies we used to build each.

<table>
<thead>
<tr>
<th>App</th>
<th>Technologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 2, Welcome App</td>
<td>Android Studio, visual GUI design, layouts, TextViews, ImageViews, accessibility, internationalization.</td>
</tr>
<tr>
<td>Chapter 3, Tip Calculator App</td>
<td>GridLayout, EditText, SeekBar, event handling, NumberFormat, customizing themes, defining app functionality with Java.</td>
</tr>
<tr>
<td>Chapter 4, Flag Quiz App</td>
<td>Fragments, Menus, Preferences, explicit Intents, Handler, AssetManager, tweened animations, Animators, Toasts, color state lists, layouts for multiple device orientations, logging error messages for debugging.</td>
</tr>
<tr>
<td>Chapter 5, Doodlz App</td>
<td>Two-dimensional graphics, Canvas, Bitmap, accelerometer, SensorManager, multitouch events, MediaStore, printing, Android 6.0 permissions.</td>
</tr>
<tr>
<td>Chapter 6, Cannon Game App</td>
<td>Manual frame-by-frame animation, graphics, sound, threading, SurfaceView and SurfaceHolder, immersive mode and full-screen layouts.</td>
</tr>
<tr>
<td>Chapter 8, Twitter® Searches App</td>
<td>SharedPreferences, SharedPreferences.Editor, implicit Intents, Intent Choosers, RecyclerView, RecyclerView.Adapter, RecyclerView.ViewHolder, RecyclerView.ItemDecoration.</td>
</tr>
<tr>
<td>Chapter 9, Address Book App</td>
<td>FragmentTransactions and the Fragment back stack, SQLite, SQLiteDatabase, SQLiteOpenHelper, ContentProvider, ContentResolver, Loader, LoaderManager, Cursor and GUI Styles.</td>
</tr>
</tbody>
</table>

**Fig. 1** | *Android How to Program, 3/e* apps.
Intended Audiences

There are several audiences for this book. Most commonly, it will be used in upper-level elective college courses and industry professional courses for people familiar with object-oriented programming but who may or may not know Java and want to learn Android app development.

Uniquely, the book can also be used in introductory courses like CS1, intended for programming novices. We recommend that schools typically offering many sections of CS1 in Java consider designating one or two sections for ambitious students who have at least some prior programming experience and who want to work hard to learn a good amount of Java and Android in an aggressively paced one-semester course. The schools may want to list the courses with “honors” or “accelerated” designations. The book will work especially well in two-semester introductory programming sequences where the introduction to Java is covered first.

Instructor Resources

The following supplements are available to qualified college instructors only through Pearson Education’s Instructor Resource Center (http://www.pearsonhighered.com/irc):

- PowerPoint® slides containing the code and figures in the text.
- Test Item File of short-answer questions.
- Solutions Manual with solutions to the end-of-chapter short-answer exercises for both the Java and Android content. For the Java content, solutions also are provided for most of the programming exercises.

The suggested Android app-development project exercises are not typical homework problems. These tend to be substantial projects—many of which could require weeks of effort, possibly with students working in teams, as is common in industry. We do not provide solutions for these exercises. Please contact us at deitel@deitel.com if you have any questions.

Please do not write to us requesting access to the Pearson Instructor’s Resource Center. Access is restricted to qualified college instructors teaching from the book. Instructors may obtain access only through their Pearson representatives. If you’re not a registered faculty member, contact your Pearson representative or visit

http://www.pearsonhighered.com/educator/replocator/

Before You Begin: Software Used in Android How to Program, 3/e

For information on configuring your computer so that you can develop apps with Java and Android, see the Before You Begin section that follows this Preface.

Getting up to Speed in Java and XML

The Android portion of this book assumes that you already know Java and object-oriented programming. We use only complete, working apps, so if you don’t know Java but have object-oriented programming experience in a C-based language such as C++, C#, Swift or Objective-C you should be able to master the material quickly, learning a good amount of Java and Java-style object-oriented programming along the way.
If you do not know Java or another object-oriented programming language, the appendices provide a friendly introduction to Java and the object-oriented programming techniques you’ll need to develop Android apps. If you’re interested in learning Java in more depth, you may want to check out the comprehensive treatment in our textbook *Java How to Program, 10/e* (http://www.deitel.com/books/jhtp10). If you’re not familiar with XML, many free online tutorials are available, including:

- [Deitel XML Basics](http://bit.ly/DeitelXMLBasics)
- [Structure XML Data](http://bit.ly/StructureXMLData)

**App-Development Courses**

In 2007, Stanford University offered a new course called Creating Engaging Facebook Apps. Students worked in teams developing apps, several of which landed in Facebook’s top 10, earning some of the student developers millions of dollars. This course gained wide recognition for encouraging student creativity and team-oriented app development. Scores of colleges now offer app-development courses across many social media and mobile platforms such as Android and iOS. We encourage instructors to read the online mobile app development syllabi and check out the YouTube™ videos created by instructors and students for many of these courses.

**Key Topics**

Here are some of this book’s other key topics:

**Android 6 SDK.** We cover various new Android 6 Software Development Kit (SDK) features.

**Android Studio IDE.** The free Android Studio (based on IntelliJ IDEA Community Edition) is now Google’s preferred IDE for Android app development (the original Android development tools were based on the Eclipse IDE). Android Studio, combined with the free Android Software Development Kit (SDK) and the free Java Development Kit (JDK), provide all the software you’ll need to create, run and debug Android apps, export them for distribution (e.g., upload them to Google Play™) and more. See the Before You Begin section after this Preface for download and installation instructions for all this software.

**Material Design.** With Android 5, Google introduced its new Android look-and-feel, based on their material design specification:

[http://www.google.com/design/spec/material-design/introduction.html](http://www.google.com/design/spec/material-design/introduction.html)

In the specification, Google presents material design’s goals and principles, then provides details on animation techniques, styling on-screen elements, positioning elements, uses of specific user-interface components, user-interaction patterns, accessibility, internationalization and more. Google uses material-design principles in its mobile and browser-based apps.

Material design is a massive topic. In this book, we focus on the following aspects:

- Using Android’s built-in *Material* themes—these give Android’s built-in user-interface components a look-and-feel consistent with material design principles.

---

Key Topics

- Using built-in Android Studio *app templates*—these are designed by Google to adhere to material design principles.

- Using *user-interface components*, as appropriate, that are recommended by the material design guidelines for specific purposes, such as `FloatingActionButton`, `TextInputLayout` and `RecyclerView`.

In addition to Google’s material design specification, you may want to read the book *Android User Interface Design: Implementing Material Design for Developers, 2nd Edition* by our professional colleague and past *Android for Programmers* reviewer Ian Clifton. From Ian: “Google announced the material design guidelines in 2014, creating a design system that suggested how an app should look as well as behave. The goal was to provide a design framework that would improve the visual appearance of all apps and create a behavioral consistency that did not exist previously across apps. *Android User Interface Design: Implementing Material Design for Developers, 2nd Edition* covers material design in detail, making user-centered design, color theory, typography, interaction patterns and other aspects of design accessible to all developers.”

**Support and App Compatibility Libraries.** A big challenge developers face when using new Android features is backward compatibility with earlier Android platforms. Many new Android features are now introduced via support libraries. These enable you to use new features in apps targeting current *and* past Android platforms. One such library is the `AppCompat` library. Android Studio’s app templates have been updated to use the `AppCompat` library and its themes, enabling the new apps you create to run on most Android devices. By creating apps with the `AppCompat` library from the start, you avoid having to reimplement your code if you decide to support older Android versions to target a wider audience.

In addition, at the 2015 Google I/O developer conference, Google introduced the Android Design Support Library for using material design in Android 2.1 and higher. Material design support also is built into most of Android Studio’s app templates.

**REST Web Services and JSON.** Chapter 7 presents our *Weather Viewer* app, which demonstrates how to invoke REST (Representational State Transfer) web services—in this case, the 16-day weather-forecast web service from OpenWeatherMap.org. This service returns the weather forecast in JSON (JavaScript Object Notation)—a popular text-based data-interchange format used to represent objects as key–value pairs of data. The app also uses classes from the `org.json` package to process the web service’s JSON response.

**Android 6.0 Permissions.** Android 6.0 has a new permissions model that’s designed for a better user experience. Before Android 6.0, a user was required at installation time to grant *in advance* all permissions that an app would ever need, which often discouraged users from installing apps. With the new model, the app is installed without asking for any permissions. Instead, the user is asked to grant a permission only the *first* time the corresponding feature is used. Chapter 5 introduces the new permissions model and uses it to request permission from the user to store an image on the device’s external storage.
Preface

Fragments. Starting with Chapter 4, we use Fragments to create and manage portions of each app’s GUI. You can combine several fragments to create user interfaces that take advantage of tablet screen sizes. Also, you can easily interchange fragments to make your GUIs more dynamic, as you’ll do in Chapter 9.

View-Holder Pattern, ListView and RecyclerView. The apps in Chapters 7–9 each display scrollable lists of data. Chapter 7 presents the data in a ListView and introduces the view-holder pattern, which improves scrolling performance by reusing GUI components that scroll off-screen. With ListViews, using the view-holder pattern is recommended. Chapters 8 and 9 each present a list of data in the more flexible and more efficient RecyclerView for which the view-holder pattern is required.

Printing. We demonstrate class PrintHelper from Android’s printing framework for printing from an app (Chapter 5). Class PrintHelper provides a user interface for selecting a printer, has a method for determining whether a given device supports printing and provides a method for printing a Bitmap. PrintHelper is part of the Android Support Library.

Immersive Mode. Chapter 6’s Cannon Game app uses immersive mode to hide the system bars at the screen’s top and bottom, allowing the game to fill the screen. Users can access the system bars by swiping down from the top of the screen.

Testing on Android Smartphones, Tablets and the Android Emulator. For the best app-development experience and results, you should test your apps on actual Android smartphones and tablets. You can still have a meaningful experience using just the Android emulator (see the Before You Begin section); however, it’s processor intensive and can be slow, particularly with games that have a lot of moving parts. In Chapter 1, we mention some Android features that are not supported on the emulator.

Cloud Test Lab. Google is working on a new Cloud Test Lab—an online site for testing your apps across a wide range of devices, device orientations, locales, spoken languages and network conditions. You’ll be able to run automated tests and receive detailed reports containing screenshots and videos of your app in action, as well as error logs to help you find problems and improve your apps. For more information and to sign up to be notified when Cloud Test Lab becomes available, visit:

http://developers.google.com/cloud-test-lab/

Android Wear and Android TV. Android Wear runs on smart watches. Android TV runs directly on some smart TVs and on media players that you can connect to your TV (typically via HDMI cables). Many Android techniques we present also apply to Android Wear and Android TV app development. The Android SDK provides Android Wear and Android TV emulators, so you can test your apps for these platforms, even if you don’t have devices. To learn more about these technologies from the developer perspective, visit:


for Android Wear and


for Android TV.
Multimedia. The book’s apps use a range of Android multimedia capabilities, including graphics, images, frame-by-frame animation and audio.

Uploading Apps to Google Play. Chapter 10, Google Play and App Business Issues, discusses Google Play and setting up a merchant account so you can sell your apps. You’ll learn how to prepare apps for submission to Google Play, find tips for pricing your apps, and find resources for monetizing them with in-app advertising and in-app sales of virtual goods. You’ll also find resources for marketing your apps. Chapter 10 can be read after Chapter 1.

Java Exception Handling. We integrate basic exception handling early in the Java content then present a richer treatment in Appendix H; we use exception handling throughout the Android chapters.

Classes Arrays and ArrayList; Collections. Appendix E covers class Arrays—which contains methods for performing common array manipulations—and generic class ArrayList—which implements a dynamically resizable array-like data structure. Appendix J introduces Java’s generic collections that are used frequently in our Android treatment.

Java Multithreading. Maintaining app responsiveness is a key to building robust Android apps and requires extensive use of Android multithreading. Appendix J introduces multithreading fundamentals so that you can understand our use of the Android AsyncTask class in Chapter 7.

GUI Presentation. Appendix I introduces Java GUI development. Android provides its own GUI components, so this appendix presents a few Java GUI components and focuses on nested classes and anonymous inner classes, which are used extensively for event-handling in Android GUIs.

Pedagogic Features

Syntax Coloring. For readability, we syntax color the code, similar to Android Studio’s use of syntax coloring. Our syntax-coloring conventions are as follows:

- comments appear like this
- keywords appear like this
- constants and literal values appear like this
- all other code appears like this

Code Highlighting. We emphasize the key code segments in each program by enclosing them in light yellow rectangles.

Using Fonts for Emphasis. We use various font conventions:

- The defining occurrences of key terms appear in bold maroon for easy reference.
- On-screen IDE components appear in bold Helvetica (e.g., the File menu).
- Program source code appears in Lucida (e.g., int x = 5;).

In this book you’ll create GUIs using a combination of visual programming (point-and-click, drag-and-drop) and writing code. We use different fonts when we refer to GUI elements in program code versus GUI elements displayed in the IDE:

- When we refer to a GUI component that we create in a program, we place its class name and object name in a Lucida font—e.g., Button saveContactButton.
Preface

- When we refer to a GUI component that’s part of the IDE, we place the component’s text in a bold Helvetica font and use a plain text font for the component’s type—e.g., “the File menu” or “the Run button.”

**Using the > Character.** We use the > character to indicate selecting a menu item from a menu. For example, we use the notation File > New to indicate that you should select the New menu item from the File menu.

**Source Code.** The book’s source code is available for download from the following sites

- http://www.deitel.com/books/AndroidHTP3
- http://www.pearsonhighered.com/deitel

**Documentation.** The Android documentation, which we reference frequently, is available at

- http://developer.android.com

An overview of Android Studio is available at


**Chapter Objectives.** Each chapter begins with a list of learning objectives.

**Figures.** Numerous tables, source-code listings and screenshots are included.

**Software Engineering.** We stress program clarity and performance, and we concentrate on building well-engineered, object-oriented software.

**Self-Review Exercises and Answers.** Extensive self-review exercises and answers are included for self study.

**Exercises with a Current Flair.** We’ve worked hard to create topical Android app-development exercises. You’ll create apps using a broad array of current technologies. All of the Android programming exercises require the implementation of complete apps. You’ll be asked to enhance the existing chapter apps, develop similar apps, use your creativity to develop your own apps that use the chapter technologies and build new apps based on open-source apps available on the Internet (be sure to read and comply with the open-source code-license terms for any app you modify). The Android exercises also include short-answer fill-in and true/false questions.

In the Java exercises, you’ll be asked to recall important terms and concepts; indicate what code segments do; indicate what's wrong with a portion of code; write Java statements, methods and classes; and write complete Java programs.

**Index.** We include an extensive index for reference. The page number of the defining occurrence of each key term in the book is highlighted in the index in bold maroon.

**Working with Open-Source Apps**

The numerous free, open-source Android apps available online are excellent resources for learning Android app development. We encourage you to download open-source apps and read their source code to understand how they work. Some of the apps in the programming exercises can be implemented by modifying or enhancing existing open-source apps. Our goal is to give you handles on interesting problems that may also inspire you to create new apps using the same technologies.
Caution: The terms of open source licenses vary considerably. Some allow you to use the app’s source code freely for any purpose, while others stipulate that the code is available for personal use only—not for creating for-sale or publicly available apps. Be sure to read the licensing agreements carefully. If you wish to create a commercial app based on an open-source app, you should consider having an intellectual property attorney read the license; be aware that these attorneys charge significant fees.

Copyright Notice and Code License

All of the Android code and Android apps in the book are copyrighted by Deitel & Associates, Inc. The sample Android apps in the book are licensed under a Creative Commons Attribution 3.0 Unported License (http://creativecommons.org/licenses/by/3.0), with the exception that they may not be reused in any way in educational tutorials and textbooks, whether in print or digital format.

Additionally, the authors and publisher make no warranty of any kind, expressed or implied, with regard to these programs or to the documentation contained in this book. The authors and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs. You’re welcome to use the apps in the book as shells for your own apps, building on their existing functionality (within the terms of the preceding license). If you have any questions, contact the authors at deitel@deitel.com.

Android 6 App-Development Fundamentals LiveLessons Video Training

Our Android 6 App-Development Fundamentals, Parts I and II LiveLessons videos show you what you need to know to start building robust, powerful Android apps with Android 6, the Java programming language and Android Studio. Included are approximately 16–20 hours of expert training synchronized with this book’s Before You Begin section, Section 1.9 and the apps presented in Chapters 2–9. For additional information about Deitel LiveLessons video products, visit

http://www.deitel.com/livelessons

or contact us at deitel@deitel.com. You also can access our LiveLessons videos if you have a subscription to SafariBooksOnline.com. For a free 10-day trial, register at

http://www.safaribooksonline.com/register

Join the Deitel & Associates, Inc. Social Media Communities

To receive updates on this and our other publications, online Resource Centers, instructor-led on-site training courses and more, join the Deitel social media communities on

- Facebook®—http://facebook.com/DeitelFan
- Twitter®—http://twitter.com/deitel
- LinkedIn®—http://bit.ly/DeitelLinkedIn
- YouTube®—http://youtube.com/DeitelTV
- Google+™—http://google.com/+DeitelFan
and subscribe to the Deitel® Buzz Online newsletter

http://www.deitel.com/newsletter/subscribe.html

**Contacting the Authors**

We’d sincerely appreciate your comments, criticisms, corrections and suggestions for improvement. Please address all questions and other correspondence to

deitel@deitel.com

We’ll respond promptly and post corrections and clarifications as Android evolves at:

http://www.deitel.com/books/AndroidHTP3

and on Facebook, LinkedIn, Twitter, Google+ and the Deitel® Buzz Online.

Visit http://www.deitel.com to

- download code examples
- check out our growing list of online programming Resource Centers
- receive updates for this book by subscribing to the free Deitel® Buzz Online e-mail newsletter at http://www.deitel.com/newsletter/subscribe.html
- receive information on our instructor-led programming-language training courses offered at customer sites worldwide.

**Acknowledgments**

Thanks to Barbara Deitel for long hours devoted to this project—she created all of our Android Resource Centers and patiently researched hundreds of technical details.

We’re fortunate to have worked with the dedicated team of publishing professionals at Pearson Higher Education. We appreciate the guidance, wisdom and energy of Tracy Johnson, Executive Editor, Computer Science. Tracy and her team publish our college textbooks. We also appreciate the efforts and 20-year mentorship of our friend and professional colleague Mark L. Taub, Editor-in-Chief of the Pearson Technology Group. Mark and his team publish all of our professional books and LiveLessons video products. Michelle Housley recruited distinguished members of the Android community to review the manuscript. We selected the cover art and Chuti Prasertsith designed the cover. Bob Engelhardt did a wonderful job bringing the book to publication.

We’d like to thank Alexander Wald, a Deitel summer intern and co-author of our professional book, *Android 6 for Programmers: An App-Driven Approach*. He helped us convert the previous edition of that book and our Android apps from Android 4.3 and 4.4 using Eclipse to Android 6 using Android Studio. Alexander is currently pursuing a B.S. in Computer Science at Worcester Polytechnic Institute with a minor in Electrical Engineering. He became interested in mathematics and the sciences at an early age and has been writing code for approximately nine years. He’s motivated by his passion to be creative and innovative and his interest in sharing his knowledge with others.

We’d also like to thank Michael Morgano, a former colleague of ours at Deitel & Associates, Inc., now an Android developer at PHHHOTO, who co-authored the first editions of this book and our book, *iPhone for Programmers: An App-Driven Approach*. Michael is an extraordinarily talented software development professional.
Finally, thanks to Abbey Deitel, former President of Deitel & Associates, Inc., and a graduate of Carnegie Mellon University’s Tepper School of Management where she received a B.S. in Industrial Management. Abbey managed the business operations of Deitel & Associates, Inc. for 17 years, along the way co-authoring a number of our publications, including the previous *Android How to Program* editions’ versions of Chapters 1 and 10.

**Reviewers of the Content from Android How to Program and Android 6 for Programmers: An App-Driven Approach Recent Editions**

We’d like to thank the following academics and professionals who reviewed this book and/or its previous editions. They scrutinized the text and the code and provided countless suggestions for improving the presentation: Paul Beusterien (Principal, Mobile Developer Solutions), Eric J. Bowden, COO (Safe Driving Systems, LLC), Tony Cantrell (Georgia Northwestern Technical College), Ian G. Clifton (Independent Contractor, Android App Developer and author of *Android User Interface Design: Implementing Material Design for Developers, 2nd Edition*), Daniel Galpin (Android Advocate and author of *Intro to Android Application Development*), Jim Hathaway (Application Developer, Kellogg Company), Douglas Jones (Senior Software Engineer, Fullpower Technologies), Charles Lasky (Nagautuck Community College), Enrique Lopez-Manas (Lead Android Architect, Sixt, and Computer Science Teacher at the University of Alcalá in Madrid), Sebastian Nykopp (Chief Architect, Reactor), Michael Pardo (Android Developer, Mobiata), Luis Ramirez (Lead Android Engineer at Reverb), Ronan “Zer” Schwartz (CIO, OpenIntents), Arijit Sengupta (Wright State University), Donald Smith (Columbia College), Jesus Ubaldo Quevedo-Torrero (University of Wisconsin, Parkside), Dawn Wick (Southwestern Community College) and Frank Xu (Gannon University).

Well, there you have it! *Android How to Program, 3/e* will quickly get you started developing Android apps with Android 6 and Android Studio. We hope you enjoy reading the book as much as we enjoyed writing it!

*Paul Deitel*

*Harvey Deitel*

**About the Authors**

**Paul Deitel**, CEO and Chief Technical Officer of Deitel & Associates, Inc., has over 30 years experience in computing and is a graduate of MIT, where he studied Information Technology. He holds the Java Certified Programmer and Java Certified Developer designations and is an Oracle Java Champion. Paul was named as a Microsoft® Most Valuable Professional (MVP) for C# in 2012–2014. Through Deitel & Associates, Inc., he has delivered hundreds of programming courses worldwide to clients, including Cisco, IBM, Siemens, Sun Microsystems, Dell, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Rogue Wave Software, Boeing, SunGard, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey Deitel, are the world’s best-selling programming-language textbook/professional book/video authors.

**Dr. Harvey Deitel**, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has over 50 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees in Electrical Engineering from MIT and a Ph.D. in Mathematics from Boston Uni-
versity—he studied computing in each of these programs before they spun off Computer Science departments. He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Science Department at Boston College before founding Deitel & Associates, Inc., in 1991 with his son, Paul. The Deitels’ publications have earned international recognition, with translations published in Japanese, German, Russian, Spanish, French, Polish, Italian, Simplified Chinese, Traditional Chinese, Korean, Portuguese, Greek, Urdu and Turkish. Dr. Deitel has delivered hundreds of programming courses to corporate, academic, government and military clients.

**About Deitel & Associates, Inc.**

Deitel & Associates, Inc., founded by Paul Deitel and Harvey Deitel, is an internationally recognized authoring and corporate training organization, specializing in Android and iOS app development, computer programming languages and Internet and web software technology. The company’s clients include many of the world’s largest corporations, as well as government agencies, branches of the military, and academic institutions. The company offers instructor-led training courses delivered at client sites worldwide on major programming languages and platforms, including Android app development, iOS app development, Java™, Swift™, C++, C, Visual C#®, Visual Basic®, and Internet and web programming.

Through its 40-year publishing partnership with Prentice Hall/Pearson, Deitel & Associates, Inc., publishes leading-edge programming professional books, college textbooks, e-books, interactive e-learning and LiveLessons video courses. Deitel & Associates, Inc. and the authors can be reached at:

```plaintext
deitel@deitel.com
```

To learn more about Deitel’s Corporate Training curriculum, visit

```
http://www.deitel.com/training
```

To request a proposal for worldwide on-site, instructor-led training at your organization, send an e-mail to deitel@deitel.com.

Individuals wishing to purchase Deitel publications can do so via links posted at

```
http://www.deitel.com and through http://informit.com, http://amazon.com, http://barnesandnoble.com and other major booksellers. Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit
```

```
```