Lauren Darcey
Shane Conder

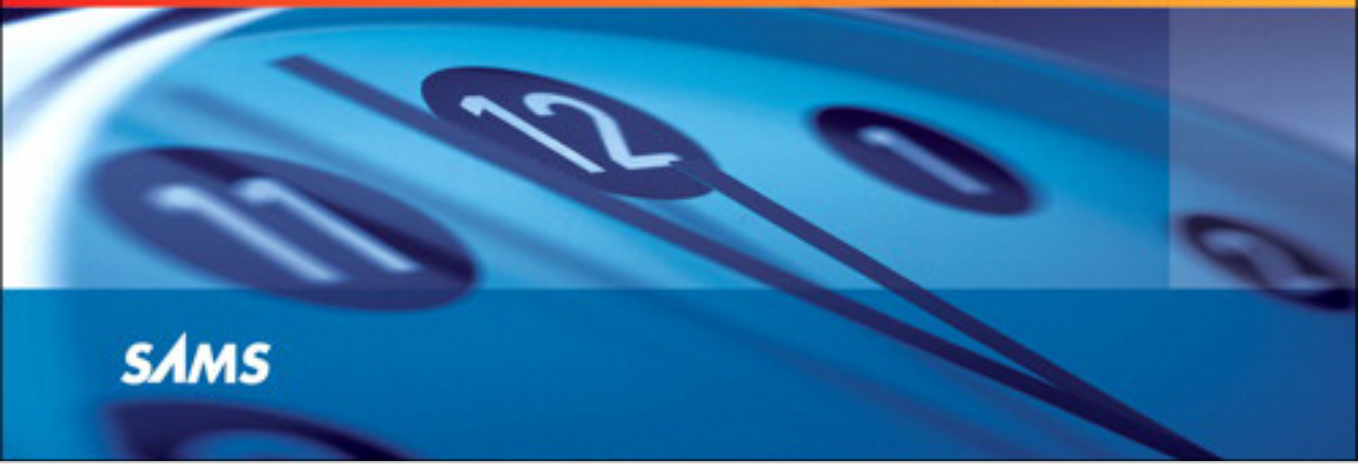
Second Edition

Full Color
Sample code provided on enclosed CD

Sams Teach Yourself

Android™ Application Development

in 24 Hours



SAMS

FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Lauren Darcey
Shane Conder

Sams **Teach Yourself**

Android™ Application Development

Second Edition

in **24 Hours**

SAMS 800 East 96th Street, Indianapolis, Indiana, 46240 USA

Sams Teach Yourself Android Application Development in 24 Hours, Second Edition

ISBN-13: 978-0-672-33569-3
ISBN-10: 0-672-33569-7

Library of Congress Cataloging-in-Publication Data
Darcey, Lauren, 1977-
Sams teach yourself Android application development in 24 hours / Lauren Darcey, Shane Conder. – 2nd ed.
p. cm.
ISBN 978-0-672-33569-3 (pbk. : alk. paper)
1. Application software–Development. 2. Android (Electronic resource) 3. Mobile computing. I. Conder, Shane, 1975- II. Title. III. Title: Teach yourself Android application development in twenty-four hours.
QA76.76.A65D26 2012
004–dc23

2011025487

Printed in the United States of America

Second Printing: September 2012

Trademarks

Warning and Disclaimer

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales
international@pearson.com

Editor in Chief
Mark Taub

Acquisitions Editor
Trina MacDonald

Development Editor
Sheri Cain

Managing Editor
Sandra Schroeder

Project Editor
Mandie Frank

Copy Editor
Charlotte Kughen, The Wordsmithery LLC

Indexer
Larry Sweazy

Proofreader
Williams Woods Publishing Services

Technical Editor
Jim Hathaway

Publishing Coordinator
Olivia Basegio

Designer
Gary Adair

Compositor
Bronkella Publishing

Contents at a Glance

Table of Contents

Part IV: Adding Polish to Your Android Application

Part VI: Appendixes

About the Authors

Lauren Darcey is responsible for the technical leadership and direction of a small software company specializing in mobile technologies, including Android, iPhone, BlackBerry, Palm Pre, BREW, and J2ME, and consulting services. With more than two decades of experience in professional software production, Lauren is a recognized authority in enterprise architecture and the development of commercial-grade mobile applications. Lauren received a B.S. in Computer Science from the University of California, Santa Cruz.

She spends her copious free time traveling the world with her geeky mobile-minded husband. She is an avid nature photographer, and her work has been published in books and newspapers around the world. In South Africa, she dove with 4-meter-long great white sharks and got stuck between a herd of rampaging hippopotami and an irritated bull elephant. She's been attacked by monkeys in Japan, gotten stuck in a ravine with two hungry lions in Kenya, gotten thirsty in Egypt, narrowly avoided a coup d'état in Thailand, geocached her way through the Swiss Alps, drank her way through the beer halls of Germany, slept in the crumbling castles of Europe, and gotten her tongue stuck to an iceberg in Iceland (while being watched by a herd of suspicious wild reindeer).

Shane Conder has extensive development experience and has focused his attention on mobile and embedded development for the past decade. He has designed and developed many commercial applications for Android, iPhone, BREW, BlackBerry, J2ME, Palm, and Windows Mobile—some of which have been installed on millions of phones worldwide. Shane has written extensively about the mobile industry and evaluated mobile development platforms on his tech blogs and is well known within the blogosphere. Shane received a B.S. in Computer Science from the University of California.

A self-admitted gadget freak, Shane always has the latest phone, laptop, or other mobile device. He can often be found fiddling with the latest technologies, such as cloud services and mobile platforms, and other exciting, state-of-the-art technologies that activate the creative part of his brain. He also enjoys traveling the world with his geeky wife, even if she did make him dive with 4-meter-long great white sharks and almost get eaten by a lion in Kenya. He admits that he has to take at least two phones and a tablet with him when backpacking, even though there is no coverage, that he snickered and whipped out his Android phone to take a picture when his wife got her tongue stuck to that iceberg in Iceland, and that he is catching on that he should be writing his own bio.

The authors have also published an intermediate/advanced book on Android development called *Android Wireless Application Development*, Second Edition, part of the Addison-Wesley Developer's Library series. Lauren and Shane have also published numerous articles on mobile software development for magazines, technical journals, and online publishers of educational content. You can find dozens of samples of their work in *Smart Developer* magazine (Linux New Media), Developer.com, *Network World*, Envato (MobileTuts+ and CodeCanyon), and InformIT, among others. They also publish articles of interest to their readers at their own Android website, http://androidbook.blogspot.com. You can find a full list of the authors' publications at http://goo.gl/f0Vlj.

Dedication

For Chickpea.

Acknowledgments

This book would never have been written without the guidance and encouragement we received from a number of very patient and supportive people, including our editorial team, co-workers, friends, and family.

Throughout this project, our editorial team at Pearson (Sams Publishing) has been top notch. Special thanks go to Trina MacDonald, Olivia Basegio, and Sheri Cain. Our technical reviewer, Jim Hathaway, helped us ensure that this book provides accurate information. With each edition, this book gets better. However, it wouldn't be here without the help of many folks on past editions. Thanks go out to past reviewers, technical editors, and readers for their valuable feedback. Finally, we'd like to thank our friends and family members who supported us when we needed to make our book deadlines.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger.

Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message.

When you write, please be sure to include this book's title and author as well as your name and phone or email address. I will carefully review your comments and share them with the author and editors who worked on the book.

Email: feedback@samspublishing.com

Mail: Mark Taub
Editor in Chief
Sams Publishing
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

This page intentionally left blank

Introduction

The Android platform is packing some serious heat these days in the mobile marketplace and gaining traction worldwide. The platform has seen numerous advancements in terms of SDK functionality, handset availability, and feature set. A wide diversity of Android handsets and devices are now in consumers' hands—and we're not just talking about smartphones: The Android platform is used by tablets, netbooks, e-book readers (such as the Barnes & Noble nook), the much-hyped Google TV, digital photo frames, and a variety of other consumer electronics. Mobile operators and carriers are taking the platform seriously and spending big bucks on ad campaigns for Android devices.

In the past two years, the Android platform has transitioned from an early-adopter platform to providing some serious competition to more established platforms. (Yes, we're talking about platforms such as the iPhone and BlackBerry.) Not only is Android the number one global smartphone platform, having surpassed Symbian by the end of 2010 (http://goo.gl/EDrgz), but it's also gained standing among consumers as the most desired smartphone operating system in the U.S. (http://goo.gl/pVRgy)—a claim supported by 50% of all new smartphone sales (double the sales rate of second place iOS, with 25%) and 37% of all smartphones in the U.S. (second place is iOS, with 27%).

But let's not digress into an argument over which platform is better, okay? Because, honestly, you're wasting your time if you think there's one platform to rule them all. The reality is that people the world over use different phones, in different places, for different reasons—reasons such as price, availability, coverage quality, feature set, design, familiarity, compatibility. There is no one-size-fits-all answer to this debate.

Having developed for just about every major mobile platform out there, we are keenly aware of the benefits and drawbacks of each platform. We do not presume to claim that one platform is better than another in general; each platform has distinct advantages over the rest, and these advantages can be maximized. The trick is to know which platform to use for a given project. Sometimes, the answer is to use as many platforms as possible. Lately, we've been finding that the answer is the Android platform. It's inexpensive and easy to develop for; it's available to millions of potential users worldwide; and it has fewer limitations than other platforms.

Still, the Android platform is relatively young and has not yet reached its full-fledged potential. This means frequent SDK updates, an explosion of new devices on the market, and a nearly full-time job keeping track of everything going on in the Android world. In other words, it might be a bit of a bumpy ride, but there's still time to jump on this bandwagon, write some kick-butt applications, and make a name for yourself.

So let's get to it.

Who Should Read This Book?

There's no reason anyone with an Android device, a good idea for a mobile application, and some programming knowledge couldn't put this book to use for fun and profit. Whether you're a programmer looking to break into mobile technology or an entrepreneur with a cool app idea, this book can help you realize your goals of making killer Android apps.

We make as few assumptions about you as a reader of this book as possible. No wireless development experience is necessary. We do assume that you're somewhat comfortable installing applications on a computer (for example, Eclipse, the Java JDK, and the Android SDK) and tools and drivers (for USB access to a phone). We also assume that you own at least one Android device and can navigate your way around it, for testing purposes.

Android apps are written in Java. Therefore, we assume you have a reasonably solid understanding of the Java programming language (classes, methods, scoping, OOP, and so on), ideally using the Eclipse development environment. Familiarity with common Java packages such as `java.lang`, `java.net`, and `java.util` will serve you well.

Android can also be a fantastic platform for learning Java, provided you have some background in object-oriented programming and adequate support, such as a professor or some really good Java programming references. We have made every attempt to avoid using any fancy or confusing Java in this book, but you will find that with Android, certain syntactical Java wizardry not often covered in your typical beginner's Java book is used frequently: anonymous inner classes, method chaining, templates, reflection, and so on. With patience, and some good Java references, even beginning Java developers should be able to make it through this book alive; those with a solid understanding of Java should be able to take this book and run with it without issue.

Finally, regardless of your specific skill set, we do expect you to use this book in conjunction with other supplementary resources, specifically the Android SDK reference and the sample source code that accompanies each coding chapter. The Android SDK reference provides exhaustive documentation about each package, class, and method of the Android SDK. It's searchable online. If we were to duplicate this data in book form, this book would weigh a ton, literally. Secondly, we provide complete, functional code projects for each lesson in this book. If you're having trouble building the tutorial application as you go along, compare your work to the sample code for that lesson. The sample code is not intended to be the "answers," but it is the complete code listings that could not otherwise be reproduced in a book of this length.

How This Book Is Structured

In 24 easy one-hour lessons, you design and develop a fully functional network-enabled Android application, complete with social features and LBS (location-based services) support. Each lesson builds on your knowledge of newly introduced Android concepts, and you iteratively improve your application from hour to hour.

This book is divided into six parts:

- **Part I, "Android Fundamentals"**—Here, you get an introduction to Android, become familiar with the Android SDK and tools, install the development tools, and write your first Android application. Part I also introduces the design principles necessary to write Android applications, including how Android applications are structured and configured, as well as how to incorporate application resources such as strings, graphics, and user interface components into your projects.
- **Part II, "Building an Application Framework"**—In this part, you begin developing an application framework that serves as the primary teaching-tool for the rest of the book. You start by developing an animated splash screen, followed by screens for the main menu, settings, help, and scores. You review basic user interface design principles, such as how to collect input from the user, and how to display dialogs to the user. Finally, you implement the core application logic of the game screen.
- **Part III, "Enhancing Your Application with Powerful Android Features"**—Here, you dive deeper into the Android SDK, adding more specialized features to the sample application. You learn how to work with graphics and the built-in camera, how to leverage LBS, how to network-enable your application, and how to enhance your application with social features.
- **Part IV, "Adding Polish to Your Android Application"**—In this part, you learn how to customize your application for different handsets, screen sizes, and foreign languages. You also review different ways to test your mobile applications.
- **Part V, "Publishing Your Application"**—Here, you find out what you need to do to prepare for and publish your Android applications to the Android Market.
- **Part VI, "Appendixes"**—In this part you can find several helpful references for setting up your Android development environment, using the Eclipse IDE, and accessing supplementary book materials, like the book website and downloadable source code.

What Is (and Isn't) in This Book

First and foremost, this book aims to provide a thorough introduction to the Android platform by providing a detailed walk-through of building a real application from start to finish. We begin with the fundamentals, try to cover the most important aspects of development, and provide information on where to go for more information. This is not an exhaustive reference on the Android SDK. We assume you are using this book as a companion to the Android SDK documentation, which is available for download as part of the SDK and online at http://developer.android.com.

We only have 24 "hours" to get you up to speed on the fundamentals of Android development, so forgive us if we stay strictly to the topic at hand. Therefore, we take the prerequisites listed earlier seriously. This book does not teach you how to program, does not explain Java syntax and programming techniques, and does not stray too far into the details of supporting technologies often used by mobile applications, such as algorithm design, network protocols, developing web servers, graphic design, database schema design, and other such peripheral topics; there are fantastic references available on each of these subjects.

The Android SDK and related tools are updated very frequently (every few months). This means that no matter how we try, some minor changes in step-by-step instructions may occur if you choose to use versions of the tools and SDK that do not exactly match those listed later in this introduction in the "What Development Environment Is Used?" section. When necessary, we point out areas where the Android SDK version affects the features and functionality available to the developer. Feel free to contact us if you have specific questions; we often post addendum information or tool change information on our book website, http://androidbook.blogspot.com.

Although we specifically targeted Android SDK Version 2.3.3 and 3.0 for the tutorial in this book, many of the examples were tested on handsets running a variety of Android SDK versions, as far back as Android 1.6. We have made every effort to make the content of this book compatible with all currently used versions of Android, as well as work smoothly regardless of what version of the Android SDK you want to target.

This book is written in a tutorial style. If you're looking for an exhaustive reference on Android development, with cookbook-style code examples and a more thorough examination of the many features of the Android platform, we recommend our more advanced Android book, *Android Wireless Application Development*, Second Edition, which is part of the Addison-Wesley Developer's Library series.

What Development Environment Is Used?

The code in this book was written using the following development environments:

- Windows 7 and Mac OS X 10.6.7.
- Eclipse Java IDE Version 3.6 (Helios).
- Android ADT Plugin for Eclipse, 10.0.1.
- Android SDK tools, Release 10.
- Sun Java SE Development Kit (JDK) 6 Update 21.
- Android SDK Version 2.3.3 and 3.0 (developed and tested on a variety of SDK versions).
- Various Android devices including smartphones and tablets (Android SDK 2.2, 2.3.3, 3.0). (Note: Tablet optimization is discussed in Hour 20.)
- The network portions of the sample application leverage Google App Engine, but you won't need these tools.

What Conventions Are Used in This Book?

This book presents several types of sidebars for special kinds of information:

- **Did You Know?** messages provide useful information or hints related to the current text.
- **By the Way** messages provide additional information that might be interesting or relevant.
- **Watch Out!** messages provide hints or tips about pitfalls that may be encountered and how to avoid them.

This book uses the following code-related conventions:

- Code and programming terms are set in a `monospace` font.
- ➥ is used to signify that the code that follows should appear on the same line as the preceding code.

- Exception handling and error checking are often removed from printed code samples for clarity and to keep the book a reasonable length.

This book uses the following conventions for step-by-step instructions and explanations:

- The core application developed in this book is developed iteratively. Generally, this means that the first time a new concept is explained, every item related to the new concept is discussed in detail. As we move on to more advanced topics in later lessons, we assume that you have mastered some of the more rudimentary aspects of Android development from previous hours, and we do not repeat ourselves much. In some cases, we instruct you to implement something in an early lesson and then help you improve it in a later hour.
- We assume that you'll read the hours of this book in order. As you progress through the book, note that we do not spell out each and every step that must be taken for each and every feature you implement to follow along in building the core application example. For example, if three buttons must be implemented on a screen, we walk you step-by-step through the implementation of the first button but leave the implementation of the other two buttons as an exercise for you. In a later hour on a different topic, we might simply ask you to implement some buttons on another screen.
- Where we tell you to navigate through menu options, we separate options using commas. For example, when we instruct you on how to open a new document, we might say "Select File, New Document."

An Overview of Changes in This Edition

When we first began writing the first edition of this book, there were few Android devices on the market. Today there are hundreds of devices shipping all over the world—smartphones, tablets, e-book readers, and specialty devices such as the Google TV. The Android platform has gone through extensive changes since the first edition of this book was published. The Android SDK has many new features and the development tools have received many much-needed upgrades. Android, as a technology, is now on solid footing within the mobile marketplace.

Within this new edition we took the opportunity to overhaul the content of this book based upon reader feedback—but don't worry, it's still the book readers loved the first time, just leaner, clearer, and more up-to-date. In addition to adding new content, we've retested and upgraded all existing content (text and sample code) for use

with the newest Android SDKs, tools, and devices. Here are some of the highlights of the additions and enhancements we've made to this edition:

- Coverage of the latest and greatest Android tools and utilities
- Updates to all existing chapters, often with entirely new sections
- Improved all code listings, making them more complete and clear
- Ensured that each time a new class is discussed, its full package is specified for easy reference
- New, improved exercises based upon tremendously helpful reader feedback
- Completely overhauled sample code in a new companion CD
- Clarified several tricky areas where readers of the first edition struggled
- Coverage of hot topics such as tablet design, services, App Widgets, Android Market updates, and more
- Even more tips and tricks from the trenches to help you design, develop, and test applications for different device targets, including an all-new chapter on tackling compatibility issues

We didn't take this review lightly; we touched every chapter and appendix to make this book the most painless way possible to get started developing Android applications. Finally, we included many additions, clarifications, and, yes, even a few fixes based upon the feedback from our fantastic (and meticulous) readers. Thank you!

About the Short Links

We've chosen to make most links in the book short links. This benefits the readers of the print book by making typing links in far easier and far less prone to error. These links are all shortened with the goo.gl link shortener, a service provided by Google. If the target of the link goes away, neither the original link nor the shortened link will work. We're confident this is the easiest way for readers to effectively use the links we've provided. In addition, as authors, we get to see which links readers are actually using.

Sometimes link shorteners are used as a way to hide nefarious links. Please be assured that we have only included shortened links we believe to be good (and thoroughly tested). In addition, Google provides screening of the target URLs for malware, phishing, and spam sites. Should a target link change hands and become a bad link, using the shortened link provides you, the reader, with an extra layer of protection.

For more information on this subject, see http://www.google.com/support/websearch/bin/answer.py?answer=190768 (http://goo.gl/iv8c7).

Supplementary Tools Available

This book has an accompanying CD with all the sample source code for each lesson.

This source code is also available for download on the publisher website: http://www.informit.com/store/product.aspx?isbn=0672335697.

Shane Conder and Lauren Darcey also run a blog at http://androidbook.blogspot.com, where you can always download the latest source code for their books as well. This website also covers a variety of Android topics as well as reader discussions, questions, clarifications, the occasional exercise walkthrough, and lots of other information about Android development. You can also find links to their various technical articles online and in print.

HOUR 2

Mastering the Android Development Tools

What You'll Learn in This Hour:

- Using the Android documentation
- Debugging applications with DDMS
- Working with the Android Emulator
- Using the Android Debug Bridge (ADB)
- Working with Android virtual devices

Android developers are fortunate to have more than a dozen development tools at their disposal to help facilitate the design of quality applications. Understanding what tools are available and what they can be used for is a task best done early in the Android learning process, so that when you are faced with a problem, you have some clue as to which utility might be able to help you find a solution. Most of the Android development tools are integrated into Eclipse using the ADT plug-in, but you can also launch them independently—you can find the executables in the `/tools` subdirectory of the Android SDK installation. During this hour, we walk through a number of the most important tools available for use with Android. This information will help you develop Android applications faster and with fewer roadblocks.

Using the Android Documentation

Although it is not a tool, per se, the Android documentation is a key resource for Android developers. An HTML version of the Android documentation is provided in the `/docs` subfolder of the Android SDK documentation, and this should always be your first stop when you encounter a problem. You can also access the latest help documentation online at the

Android Developer website, http://developer.android.com (http://goo.gl/K8GgD, see Figure 2.1 for a screenshot of the Dev Guide tab of this website).

FIGURE 2.1
Android developer documentation (online version).

The Android documentation is divided into seven sections:

- **Home**—This tab provides some high-level news items for Android developers, including announcements of new platform versions. You can also find quick links for downloading the latest Android SDK, publishing your applications on the Android Market, and other helpful information.
- **SDK**—This tab provides important information about the SDK version installed on your machine. One of the most important features of this tab is the release notes, which describe any known issues for the specific installation. This information is also useful if the online help has been upgraded but you want to develop to an older version of the SDK.

- **Dev Guide**—This tab links to the Android Developer's Guide, which includes a number of FAQs for developers, best practice guides and a useful glossary of Android terminology for those new to the platform. The appendix section also lists all Android platform versions (API Levels), supported media formats, and lists of intents.
- **Reference**—This tab includes, in a Javadoc-style format, a searchable package and class index of all Android APIs provided as part of the Android SDK.
- **Resources**—This tab includes links to articles, tutorials, and sample code. It also acts as a gateway to the Android developer forums. There are a number of Google groups you can join, depending on your interests.
- **Videos**—This tab, which is available online only, is your resource for Android training videos. Here, you can find videos about the Android platform, developer tips, and the Google I/O conference sessions.
- **Blog**—This tab links to the official Android developer blog. Check here for the latest news and announcements about the Android platform. This is a great place to find how-to examples, learn how to optimize Android applications, and hear about new SDK releases and Android Developer Challenges.

Now is a good time to get to know your way around the Android SDK documentation. First, check out the online documentation and then try the local documentation (available in the `/docs` subdirectory of your Android SDK installation).

Debugging Applications with DDMS

The Dalvik Debug Monitor Service (DDMS) is a debugging utility that is integrated into Eclipse through a special Eclipse perspective. The DDMS perspective provides a number of useful features for interacting with emulators and handsets and debugging applications (Figure 2.2).

The features of DDMS are roughly divided into five functional areas:

- Task management
- File management
- Emulator interaction
- Logging
- Screen captures

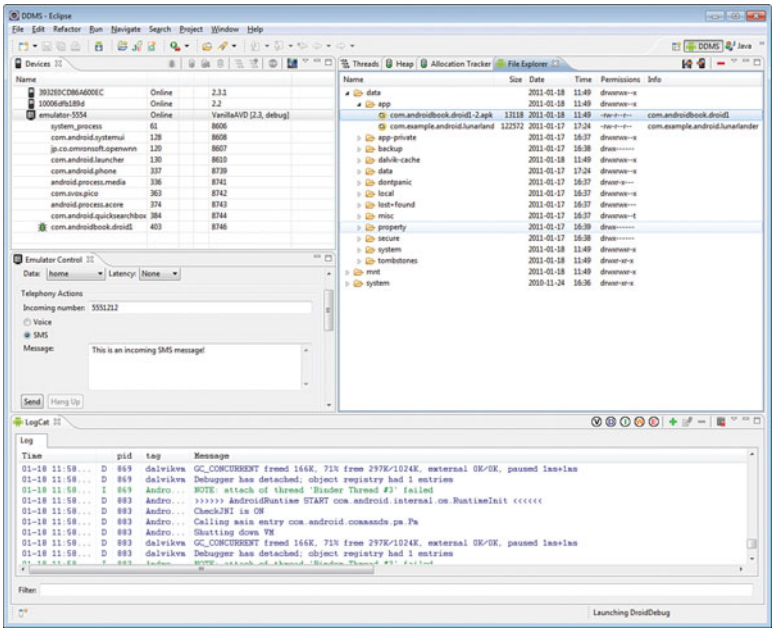


FIGURE 2.2
The DDMS perspective, with one emulator and two Android devices connected (the Nexus S running 2.3.1 and the Samsung Galaxy Tablet running 2.2).

DDMS and the DDMS perspective are essential debugging tools. Now let's take a look at how to use these features in a bit more detail.

> The DDMS tool can be launched separately from Eclipse. You can find it in the Android SDK `/tools` directory.

Managing Tasks

The top-left corner of the DDMS perspective lists the emulators and handsets currently connected. You can select individual instances and view its processes and threads. You can inspect threads by clicking on the device process you are interested in—for example, com.androidbook.droid1—and clicking the Update Threads button (), as shown in Figure 2.3. You can also prompt garbage collection on a process and then view the heap updates by clicking the Update Heap button (). Finally, you can stop a process by clicking the Stop Process button ().

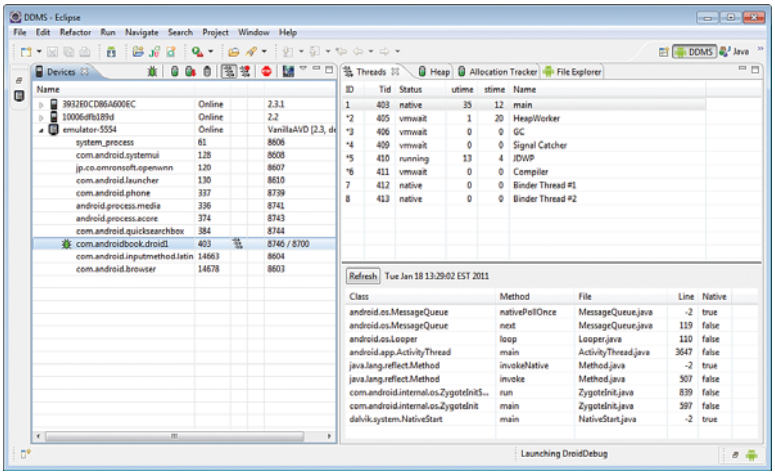


FIGURE 2.3
Using DDMS to examine thread activity for the Droid1 application.

Debugging from the DDMS Perspective

Within the DDMS perspective, you can choose a specific process on an emulator or a handset and then click the Debug button ( ) to attach a debugger to that process. You need to have the source code in your Eclipse workspace for this to work properly. This works only in Eclipse, not in the standalone version of DDMS.

Browsing the Android File System

You can use the DDMS File Explorer to browse files and directories on the emulator or a device (Figure 2.4). You can copy files between the Android file system and your development machine by using the Push ( ) and Pull ( ) buttons available in the top right-hand corner of the File Explorer tab.

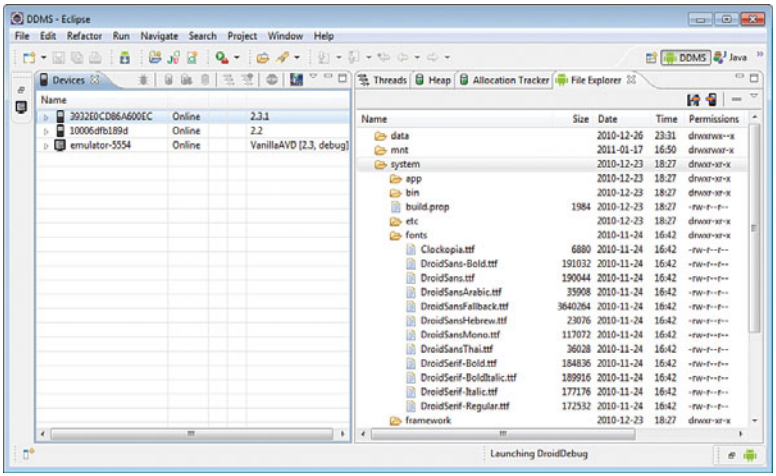


FIGURE 2.4
Using the DDMS File Explorer to browse system fonts on the handset.

You can also delete files and directories by using the Delete button ( ) or just pressing the Delete key. There is no confirmation for this delete operation, nor can it be undone.

Interacting with Emulators

DDMS can send a number of events, such as simulated calls, SMS messages, and location coordinates, to specific emulator instances. These features are found under the Emulator Control tab in DDMS. These events are all "one way," meaning that they can be initiated from DDMS, not from the emulator to DDMS.

By the Way

These features generally work for emulators only, not for handsets. For handsets, you must use real calls and real messages, which may incur fees (depending upon your plan).

Simulating Incoming Calls to the Emulator

You can simulate incoming voice calls by using the DDMS Emulator Control tab (see Figure 2.5). This is not a real call; no data (voice or otherwise) is transmitted between the caller and the receiver.

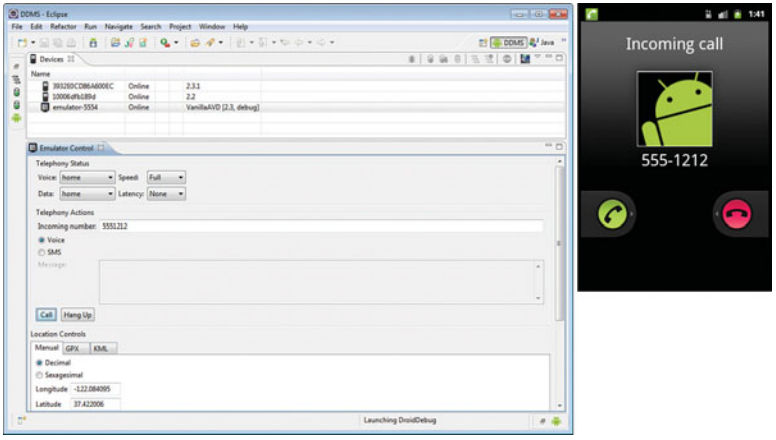


FIGURE 2.5 Using the DDMS Emulator Control tab (left) to place a call to the emulator (right).

To simulate an incoming call to an emulator running on your machine, follow these steps:

1. In the DDMS perspective, choose the emulator instance you want to call.
2. On the Emulator Control tab, navigate to the Telephony Actions section and input the incoming number (for example, 5551212).

3. Select the Voice radio button.
4. Click the Call button.
5. In the emulator, you should see an incoming call. Answer the call by clicking the Send button in the emulator or sliding the slider to the right.
6. End the call at any time by clicking the End button in the emulator or by clicking the Hang Up button in the DDMS perspective.

Simulating Incoming SMS Messages to the Emulator

You can simulate incoming SMS messages by using the Emulator DDMS Emulator Control tab (see Figure 2.6). You send an SMS much as you initiate a voice call.

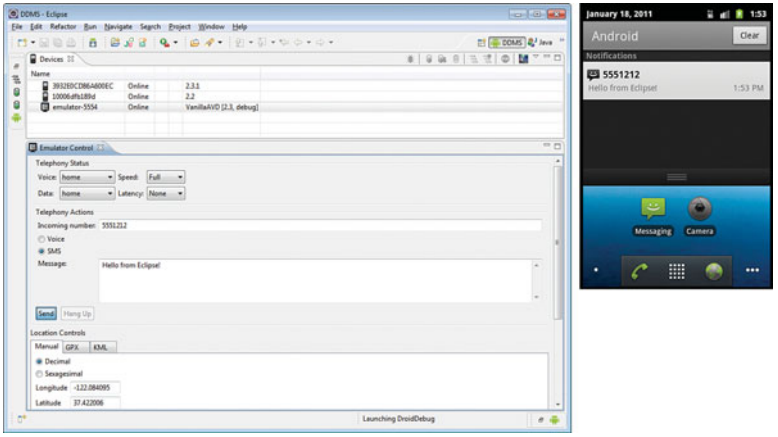


FIGURE 2.6 Using the DDMS Emulator Control tab (left) to send an SMS message to the emulator (right).

To send an SMS message to an emulator running on your machine, follow these steps:

1. In the DDMS perspective, choose the emulator instance you want a send an SMS message to.
2. On the Emulator Control tab, navigate to the Telephony Actions section and input the Incoming number (for example, 5551212).
3. Select the SMS radio button.
4. Type an SMS message in the Message textbox.
5. Click the Send button. In the emulator, you should see an incoming SMS notification on the notification bar. Pull down the bar to view the SMS message details.

Taking Screenshots of the Emulator or Handset

One feature that can be particularly useful for debugging both handsets and emulators is the ability to take screenshots of the current screen (see Figure 2.7).

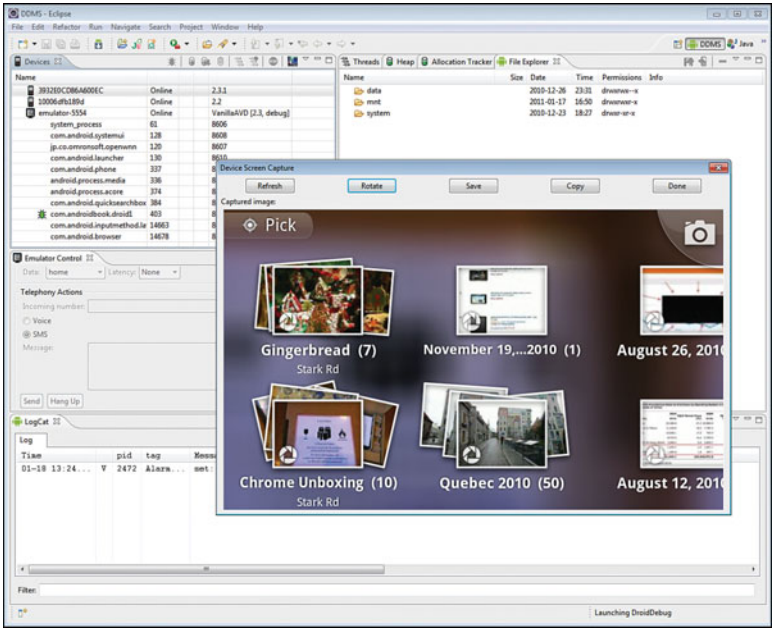


FIGURE 2.7 Using the DDMS Screen Capture button to take a screenshot of the Nexus S handset, which happens to be displaying some old photo albums in the Gallery.

The screenshot feature of the DDMS perspective is particularly useful when used with real devices. To take a screen capture of what's going on at this very moment on your device, follow these steps:

1. In the DDMS perspective, choose the device (or emulator) you want a screenshot of. The device must be connected via USB.
2. On that device or emulator, make sure you have the screen you want. Navigate to it, if necessary.
3. Press the Screen Capture button ( ) to take a screen capture. This launches a capture screen dialog.
4. Within the capture screen, click the Save button to save the screenshot to your local hard drive. The Rotate button rotates the Device Screen Capture tool to display in landscape mode. This tool does not show a live view, just a snapshot; click the Refresh button to update the capture view if you make changes on the device. The Copy button places the image on your system's clipboard

for pasting into another application, such as an image editor. Click the Done button to exit the tool and return to the DDMS perspective.

Viewing Log Information

The LogCat logging utility that is integrated into the DDMS perspective enables you to view the Android logging console. You might have noted the LogCat logging tab, with its diagnostic output, in Figure 2.2 earlier in this chapter. We talk more about how to implement your own custom application logging in Hour 3, "Building Android Applications."

Filtering Log Information

Eclipse has the ability to filter logs by log severity. You can also create custom log filters by using tags. For more information on how to do this, see Appendix B, "Eclipse IDE Tips and Tricks."

Working with the Android Emulator

The Android emulator is probably the most powerful tool at a developer's disposal. It is important for developers to learn to use the emulator and understand its limitations. The Android emulator is integrated with Eclipse, using the ADT plug-in for the Eclipse IDE.

Emulator Limitations

The Android emulator is a convenient tool, but it has a number of limitations:

- The emulator is not a device. It simulates general handset behavior, not specific hardware implementations or limitations.
- Sensor data, such as satellite location information, battery and power settings, and network connectivity, are all simulated using your computer.
- Peripherals such as camera hardware are not fully functional.
- Phone calls cannot be placed or received but are simulated. SMS messages are also simulated and do not use a real network.
- No USB or Bluetooth support is available.
- Using the Android emulator is not a substitute for testing on a true Android device.

Providing Input to the Emulator

As a developer, you can provide input to the emulator in a number of ways:

- Use your computer mouse to click, scroll, and drag items (for example, sliding volume controls) onscreen as well as on the emulator skin.
- Use your computer keyboard to input text into controls.
- Use your mouse to simulate individual finger presses on the soft keyboard or physical emulator keyboard.
- Use a number of emulator keyboard commands to control specific emulator states.

Try It Yourself

Try out some of the methods of interacting with the emulator:

1. In Eclipse, launch the Droid1 application you created in Hour 1, "Getting Started with Android."
2. While your application is running, press Ctrl+F11 and Ctrl+F12 to toggle the emulator between portrait and landscape modes. Note how your application redraws the simple application screen to accommodate different screen orientations.
3. Press Alt+Enter to enter full screen mode with the emulator. Then press Alt+Enter again to return to exit full screen mode.

Many useful commands are available for the emulator. For an exhaustive list, see the official emulator documentation that was installed with the Android SDK documentation or online at http://goo.gl/aDnxD.

Exploring the Android System

If you're not already familiar with how Android devices work, now is a good time to learn your way around Android devices as users see them. Keep in mind that we're focusing on the "Google experience" or the "Google Android" user interface here, as opposed to the specific user interface changes and additions made by some device manufacturers and carriers.

Table 2.1 lists some important features of Android devices. The features described in this table apply to the traditional smartphone UI most users are familiar. The Android 3.0/3.1 release (which was tablet-centric) introduced a new holographic UI design, which has similar features.

TABLE 2.1 Android System Screens and Features

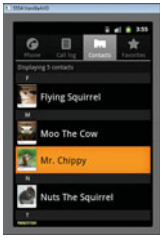
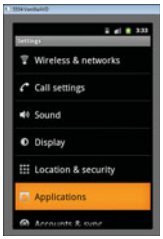
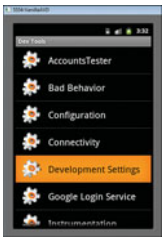
Feature	Description	Appearance
Home screen	Default screen. This is a common location for app widgets and live folders. You will also find a quick launch bar for the Dialer ( ) and Browser ( ) applications as well as the Application menu.	
Dialer application	Built-in application for making and receiving phone calls. Note: The emulator has limited phone features.	
Messaging application	Built-in application for sending and receiving SMS messages. Note: The emulator has limited messaging features.	
Browser application	Built-in web browser. Note that the emulator has an Internet connection, provided that your machine has one.	
Contacts application	Database of contact information. Leveraged by many applications on the platform for sharing purposes. Consider adding some “test contacts” to your favorite emulator AVD instance for easy development and testing.	

TABLE 2.1 Continued

Feature	Description	Appearance
Application menu	Shows all installed applications. From the Home screen, click the Application menu button ( ) to see all installed applications.	
Settings application	Built-in application to configure a wide variety of "phone" settings for the emulator, such as application management, sound and display settings, and localization.	
Dev Tools application	Built-in application to configure development tool settings.	

Using SD Card Images with the Emulator

If you want to transfer files to your emulator instance (running a specific AVD) then you likely want to use the SD card image associated with that AVD to store those files. The same holds true for downloading content such as images using the Browser application.

To copy file data to a specific instance of the emulator, use the File Explorer tab of the DDMS perspective to push or pull files. For developers, most file transfers occur either between the `/mnt/sdcard` directories, or to and from specific application's directory (for example, `/data/data/com.androidbook.droid1`).

If you've added media files (for example, images, audio, and so on) to the device, you might need to force the Android operating system to rescan for new media. The most convenient way to do this is by using the Dev Tools application to run the Media Scanner. After you force a scan, you should see any new images you copied to the `/mnt/sdcard/download` directory, for example, show up in the Gallery application.

Using Other Android Tools

Although we've already covered the most important tools, a number of other special-purpose utilities are included with the Android SDK. A list of the tools that come as part of the Android SDK is available on the Android developer website at http://goo.gl/yzFHz. Here you can find a description of each tool as well as a link to its official documentation.

Summary

The Android SDK ships with a number of powerful tools to help with common Android development tasks. The Android documentation is an essential reference for developers. The DDMS debugging tool, which is integrated into the Eclipse development environment as a perspective, is useful for monitoring emulators and devices. The Android emulator can be used for running and debugging Android applications virtually, without the need for an actual device. There are also a number of other tools for interacting with handsets and emulators in a variety of situations.

Q&A

Q. ***Is the Android documentation installed with the Android SDK the same as the documentation found at http://developer.android.com (http://goo.gl/K8GgD)?***

A. No. The documentation installed with the SDK was "frozen" at the time the SDK was released, which means it is specific to the version of the Android SDK you installed. The online documentation is always the latest version of the Android SDK. We recommend using the online documentation, unless you are working offline or have a slow Internet connection, in which case the local SDK documentation should suffice.

Q. *Do you have to develop Android applications with Eclipse?*

A. No. Eclipse is the preferred development environment for Android (and the IDE used by this book), but it is not required for Android development. The ADT plug-in for Eclipse provides a convenient entry point for many of the underlying development tools for creating, debugging, packaging, and signing Android applications. Developers who do not use Eclipse (or simply want access to these tools outside of the IDE) can run the underlying tools directly from the command line. For more information about developing using other IDEs, see the Android developer website at http://goo.gl/KXcZj.

Q. *Is testing your application on the emulator alone sufficient?*

A. No. The Android emulator simulates the functionality of a real device and can be a big time- and cost-saving tool for Android projects. It is a convenient tool for testing, but it can only pretend at real device behavior. The emulator cannot actually determine your real location or make a phone call. Also, the emulator is a generic device simulation and does not attempt to emulate any quirky details of a specific device or user experience. Just because your application runs fine on the emulator does not guarantee that it will work on the device.

Workshop

Quiz

1. Which features are available in the DDMS perspective?

   A. Taking screenshots of emulator and handset screens

   B. Browsing the file system of the emulator or handset

   C. Monitoring thread and heap information on the Android system

   D. Stopping processes

   E. Simulating incoming phone calls and SMS messages to emulators

   F. All of the above

2. True or False: You must use the Android emulator for debugging.

3. Which target platforms can Android applications be written for?

4. True or False: The Android emulator is a generic device that supports only one screen configuration.

Answers

1. F. All of the above. The DDMS perspective can be used to monitor, browse, and interact with emulators and handsets in a variety of ways.
2. False. The Android emulator is useful for debugging, but you can also connect the debugger to an actual device and directly debug applications running on real hardware.
3. There are a number of target platforms available and more are added with each new SDK release. Some important platform targets include Android 1.6, Android 2.1, Android 2.2, Android 2.3, and Android 3.0. Targets can include the Google APIs, if desired. These targets map to the AVD profiles you must create in order to use the Android emulator.
4. False. The Android emulator is a generic device, but it can support several different skins. For a complete list of skins supported, see the Android SDK and AVD Manager.

Exercises

1. Launch the Android emulator and customize your home screen. Change the wallpaper. Install an AppWidget. Get familiar with how the emulator tries to mimic a real handset. Note the limitations, such as how the dialer works.
2. Launch the Android emulator and browse the Settings application. Try changing a setting and see what happens. Uninstall an application (Settings, Applications, Manage Applications, click on an application and press the UnInstall button, then confirm with the OK button to uninstall an application). Under the About phone submenu, check the Android version.
3. Launch the Android emulator and browse the Dev Tools application. Review the settings available, especially those within the Development Settings submenu. Check out the documentation for this tool on the Android Developer website at http://goo.gl/QcScV.
4. Launch the Android emulator and add a few test contacts to your Contacts database for this AVD. If you give a contact the phone number you like to use for incoming calls from the DDMS perspective, the contact's name and picture display whenever that phone number is used for testing purposes.

5. Add a new image file to your emulator instance. Find a JPG graphic file, such as a photo, and use the DDMS perspective's File Explorer to push the file to the `/mnt/sdcard/download` directory of the emulator. Launch the Gallery application and if the image does not immediately appear, then use the Dev Tools application to perform a media scan and re-launch the Gallery application. After the graphic is visible in the Gallery, go create a contact and set the contact's photo to that photo.

Index

A

B

C

D

E

F

G

H

I

J

K-L

M

N

O

P

Q

R

S

T

U

V

W

X-Z