

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

Copyright © 2012 by Lauren Darcey and Shane Conder

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

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

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an “as is” basis. The authors and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

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

Introduction	1
--------------------	---

Part I: Android Fundamentals

HOURL 1 Getting Started with Android	9
2 Mastering the Android Development Tools	31
3 Building Android Applications	47
4 Managing Application Resources	65
5 Configuring the Android Manifest File	83
6 Designing an Application Framework	99

Part II: Building an Application Framework

HOURL 7 Implementing an Animated Splash Screen	117
8 Implementing the Main Menu Screen	133
9 Developing the Help and Scores Screens	151
10 Building Forms to Collect User Input	171
11 Using Dialogs to Collect User Input	189
12 Adding Application Logic	205

Part III: Enhancing Your Application with Powerful Android Features

HOURL 13 Working with Images and the Camera	227
14 Adding Support for Location-Based Services	245
15 Adding Basic Network Support	269
16 Adding Additional Network Features	293
17 Adding Social Features	309
18 Creating a Home Screen App Widget	325

Part IV: Adding Polish to Your Android Application

19 Internationalizing Your Application	341
20 Developing for Different Devices	355

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

21 Diving Deeper into Android	371
22 Testing Android Applications	391

Part V: Publishing Your Application

HOUR 23 Getting Ready to Publish	409
24 Publishing on the Android Market	421

Part VI: Appendixes

A Configuring Your Android Development Environment	437
B Eclipse IDE Tips and Tricks	445
C Supplementary Materials	453
Index	459

Table of Contents

Introduction	1
Who Should Read This Book?	2
How This Book Is Structured	3
What Is (and Isn't) in This Book	4
What Development Environment Is Used?	5
What Conventions Are Used in This Book?	5
An Overview of Changes in This Edition	6
About the Short Links	7
Supplementary Tools Available	8

Part I: Android Fundamentals

HOURL 1: Getting Started with Android	9
Introducing Android	9
Google and the Open Handset Alliance	9
Android Makes Its Entrance	10
Cheap and Easy Development	11
Familiarizing Yourself with Eclipse	13
Creating Android Projects	14
Exploring the Android Project Files	16
Editing Project Resources	17
Running and Debugging Applications	21
Managing Android Virtual Devices	21
Creating Debug and Run Configurations in Eclipse	22
Launching Android Applications Using the Emulator	24
Debugging Android Applications Using DDMS	25
Launching Android Applications on a Device	26

HOURL 2: Mastering the Android Development Tools	31
Using the Android Documentation	31
Debugging Applications with DDMS	33
Managing Tasks	34
Browsing the Android File System	35
Interacting with Emulators	36
Taking Screenshots of the Emulator or Handset	38
Viewing Log Information	39
Working with the Android Emulator	39
Providing Input to the Emulator	40
Exploring the Android System	40
Using SD Card Images with the Emulator	42
Using Other Android Tools	43
HOURL 3: Building Android Applications	47
Designing a Typical Android Application	47
Designing Application Features	48
Determining Application Activity Requirements	49
Implementing Application Functionality	50
Using the Application Context	51
Retrieving Application Resources	51
Accessing Application Preferences	51
Accessing Other Application Functionality Using Contexts	52
Working with Activities	52
Launching Activities	53
Managing Activity State	54
Shutting Down Activities	56
Working with Intents	56
Passing Information with Intents	56
Using Intents to Launch Other Applications	57
Working with Dialogs	58
Working with Fragments	59
Logging Application Information	60

HOURL 4: Managing Application Resources	65
Using Application and System Resources	65
Working with Application Resources	66
Working with System Resources	68
Working with Simple Resource Values	69
Working with Strings	69
Working with Colors	70
Working with Dimensions	71
Working with Drawable Resources	72
Working with Images	72
Working with Other Types of Drawables	73
Working with Layouts	74
Designing Layouts Using the Layout Resource Editor	74
Designing Layouts Using XML	75
Working with Files	77
Working with XML Files	77
Working with Raw Files	78
Working with Other Types of Resources	79
HOURL 5: Configuring the Android Manifest File	83
Exploring the Android Manifest File	83
Using the Manifest Tab	84
Using the Application Tab	84
Using the Permissions Tab	85
Using the Instrumentation Tab	86
Using the AndroidManifest.xml Tab	86
Configuring Basic Application Settings	87
Naming Android Packages	88
Versioning an Application	88
Setting the Minimum Android SDK Version	89
Naming an Application	90
Providing an Icon for an Application	90

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

Providing an Application Description	90
Setting Debug Information for an Application	90
Setting Other Application Attributes	90
Defining Activities	91
Registering Activities	91
Designating the Launch Activity	92
Managing Application Permissions	93
Managing Other Application Settings	96
HOURL 6: Designing an Application Framework	99
Designing an Android Trivia Game	99
Determining High-Level Game Features	100
Determining Activity Requirements	100
Determining Screen-Specific Game Features	101
Implementing an Application Prototype	106
Reviewing the Accompanying Source Code	106
Creating a New Android Project	107
Adding Project Resources	107
Implementing Application Activities	109
Creating Application Preferences	110
Running the Game Prototype	111
Creating a Debug Configuration	112
Launching the Prototype in the Emulator	112
Exploring the Prototype Installation	113
Part II: Building an Application Framework	
HOURL 7: Implementing an Animated Splash Screen	117
Designing the Splash Screen	117
Implementing the Splash Screen Layout	118
Adding New Project Resources	120
Updating the Splash Screen Layout	122

Working with Animation	126
Adding Animation Resources	126
Animating Specific Views	128
Animating All Views in a Layout	129
Handling Animation Life Cycle Events	129
HOOR 8: Implementing the Main Menu Screen	133
Designing the Main Menu Screen	133
Determining Main Menu Screen Layout Requirements	134
Designing the Screen Header with RelativeLayout	135
Designing the ListView Control	135
Finishing Touches for the Main Menu Layout Design	135
Implementing the Main Menu Screen Layout	136
Adding New Project Resources	136
Updating the Main Menu Screen Layout Files	138
Working with the ListView Control	140
Filling a ListView Control	140
Listening for ListView Events	141
Customizing ListView Control Characteristics	143
Working with Other Menu Types	144
Adding an Options Menu to the Game Screen	145
HOOR 9: Developing the Help and Scores Screens	151
Designing the Help Screen	151
Implementing the Help Screen Layout	153
Adding New Project Resources	153
Updating the Help Screen Layout	154
Working with Files	155
Adding Raw Resource Files	156
Accessing Raw File Resources	156

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

Designing the Scores Screen	157
Determining Scores Screen Layout Requirements	158
Adding the TabHost Control	158
Implementing the Scores Screen Layout	160
Adding New Project Resources	160
Updating the Scores Screen Layout	161
Building a Screen with Tabs	163
Configuring the TabHost Control	163
Adding Tabs to the TabHost Control	164
Setting the Default Tab	164
Working with XML	165
Retrieving XML Resources	165
Parsing XML Files with XmlResourceParser	165
Applying Finishing Touches to the Scores Screen	166
HOOR 10: Building Forms to Collect User Input	171
Designing the Settings Screen	171
Implementing the Settings Screen Layout	175
Adding New Project Resources	175
Updating the Settings Screen Layout	176
Using Common Form Controls	178
Working with EditText Controls	178
Working with Button Controls	179
Working with Spinner Controls	182
Saving Form Data with SharedPreferences	184
Defining SharedPreferences Entries	184
Saving Settings to SharedPreferences	184
Reading Settings from SharedPreferences	185

HOURL 11: Using Dialogs to Collect User Input	189
Working with Activity Dialogs	189
Exploring the Different Types of Dialogs	190
Tracing the Life Cycle of an Activity Dialog	191
Using the DatePickerDialog Class	192
Adding a DatePickerDialog to a Class	193
Initializing a DatePickerDialog	194
Launching DatePickerDialog	195
Working with Custom Dialogs	196
Adding a Custom Dialog to the Settings Screen	196
HOURL 12: Adding Application Logic	205
Designing the Game Screen	205
Implementing the Game Screen Layout	208
Adding New Project Resources	208
Updating the Game Screen Layout	210
Working with ViewSwitcher Controls	211
Initializing Switcher Controls	212
Implementing Switcher Factory Classes	212
Updating the TextSwitcher Control	214
Updating the ImageSwitcher Control	214
Wiring Up Game Logic	215
Adding Game State Settings to the SharedPreferences	216
Retrieving, Parsing, and Storing Question Data	217
Part III: Enhancing Your Application with Powerful Android Features	
HOURL 13: Working with Images and the Camera	227
Designing the Avatar Feature	227
Adding an Avatar to the Settings Layout	229
Updating the Settings Screen Layout	230

Working with ImageButton Controls	231
Setting the Image of an ImageButton Control	231
Handling ImageButton Click Events	233
Choosing and Saving the Avatar Graphic	234
Working with Bitmaps	239
HOURL 14: Adding Support for Location-Based Services	245
Designing the Favorite Place Feature	245
Determining Favorite Place Feature Layout Updates	246
Designing the Favorite Place Dialog	247
Implementing the Favorite Place Feature	248
Adding New Project Resources	249
Updating the Settings Screen Layout	250
Implementing the Favorite Place Dialog Layout	250
Implementing the Favorite Place Dialog	252
Using Location-Based Services	254
Enabling Location Testing on the Emulator	255
Accessing the Location-Based Services	259
Using Geocoding Services	260
Using Geocoding Services with Android	261
Working with Maps	263
Launching a Map Application by Using an Intent	263
Working with Third-Party Services and Applications	265
HOURL 15: Adding Basic Network Support	269
Designing Network Applications	269
Working with an Application Server	270
Managing Lengthy Network Operations	271
Informing the User of Network Activity	271
Developing Network Applications	272
Enabling Network Testing on the Emulator	272
Testing Network Applications on Hardware	273

Accessing Network Services	274
Planning Been There, Done That! Network Support	274
Setting Network Permissions	275
Checking Network Status	275
Using HTTP Networking	276
Indicating Network Activity with Progress Bars	277
Displaying Indeterminate Progress	277
Displaying Determinate Progress	277
Displaying Progress Dialogs	278
Running Tasks Asynchronously	279
Using AsyncTask	279
Using Threads and Handlers	280
Downloading and Displaying Score Data	280
Extending AsyncTask for Score Downloads	281
Starting the Progress Indicator with <code>onPreExecute()</code>	282
Clearing the Progress Indicator with <code>onPostExecute()</code>	282
Handling Cancellation with <code>onCancelled()</code>	283
Handling Processing with <code>doInBackground()</code>	284
Handling Progress Updates with <code>onProgressUpdate()</code>	285
Starting the <code>ScoreDownloaderTask</code>	286
Downloading and Parsing Question Batches	287
Extending AsyncTask for Question Downloads	287
Starting the Progress Dialog with <code>onPreExecute()</code>	288
Dismissing the Progress Dialog with <code>onPostExecute()</code>	288
Handling the Background Processing	289
Starting <code>QuizTask</code>	289
HOOR 16: Adding Additional Network Features	293
Determining What Data to Send to the Server	293
Keeping Player Data in Sync	294
Uploading Settings Data to a Remote Server	295

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

Working with Android Services	296
Implementing UploadTask	298
Uploading Player Data with the HTTP GET Method	299
Uploading Avatar Data with the HTTP POST Method	301
Uploading Score Data to a Remote Server	304
Downloading Friends' Score Data	305
HOURL 17: Adding Social Features	309
Enhancing Applications with Social Features	309
Tailoring Social Features to Your Application	310
Supporting Basic Player Relationships	310
Adding Friend Support to Your Application	311
Enabling Friend Requests on the Settings Screen	311
Implementing the Friend Request Feature	314
Enhancing Player Relationships	318
Integrating with Social Networking Services	319
Adding Facebook Support	320
Adding Twitter Support	320
Working with the OpenSocial Initiative	320
HOURL 18: Creating a Home Screen App Widget	325
Designing an App Widget	325
Developing an App Widget	326
Configuring App Widget Properties	326
Working with RemoteViews	327
Working with Styles	328
Designing the App Widget Layout	329
Implementing an App Widget Provider	331
Handling App Widget Background Tasks	331
Updating the Android Manifest File	335

Part IV: Adding Polish to Your Android Application

HOURL 19: Internationalizing Your Application	341
General Internationalization Principles	341
How Android Localization Works	343
How the Android Operating System Handles Locale	345
How Applications Handle Locales	346
How the Android Market Handles Locales	348
Android Internationalization Strategies	349
Forgoing Application Internationalization	349
Limiting Application Internationalization	350
Implementing Full Application Internationalization	350
Using Localization Utilities	351
Determining System Locale	351
Formatting Date and Time Strings	351
Handling Currencies	352
HOURL 20: Developing for Different Devices	355
Configuration Management for Android	355
Handling Different Screen Orientations	357
Handling Orientation Changes Programmatically	362
Supporting Different Screen Characteristics	363
Supporting Different Device Features	364
Developing for Different Android SDKs	365
HOURL 21: Diving Deeper into Android	371
Exploring More Core Android Features	371
Declaring and Enforcing Application Permissions	372
Alerting the User with Notifications	372
Designing Advanced User Interfaces	373
Using Styles and Themes	373
Designing Custom View and ViewGroup Controls	374
Working with Input Methods	374

Handling User Gestures	375
Converting Text to Speech	376
Converting Speech to Text	377
Working with Multimedia	377
Playing and Recording Audio	377
Playing and Recording Video	378
Working with 2D and 3D Graphics	378
Using the Android Graphics Libraries	379
Using the OpenGL ES Graphics API	379
Personalizing Android Devices	380
Setting the Ringtone	380
Setting the Wallpaper	380
Creating a Live Wallpaper	381
Managing and Sharing Data	381
Working with Files and Directories	382
Storing Structured Data in a SQLite Database	383
Sharing Data with Other Applications	383
Integrating with Global Search	385
Accessing Underlying Device Hardware	386
Reading Raw Sensor Data	386
Working with Wi-Fi	387
Working with Bluetooth	387
Managing Power Settings and Battery Life	387
Hour 22: Testing Android Applications	391
Testing Best Practices	391
Developing Coding Standards	392
Performing Regular Versioned Builds	393
Using a Defect Tracking System	393
Developing Good Test Plans	393

Maximizing Test Coverage	395
Managing the Testing Environment	395
Testing on the Emulator	397
Testing on Target Devices	398
Performing Automated Testing	398

Part V: Publishing Your Application

HOURL 23: Getting Ready to Publish **409**

Understanding the Release Process	409
Preparing the Release Candidate Build	411
Preparing the Android Manifest File for Release	411
Protecting Your Application from Software Pirates	412
Readying Related Services for Release	413
Testing the Application Release Candidate	413
Packaging and Signing an Application	414
Digitally Signing Applications	414
Exporting and Signing the Package File	415
Testing the Signed Application Package	417
Installing the Signed Application Package	417
Verifying the Signed Application	418

HOURL 24: Publishing on the Android Market **421**

Selling on the Android Market	421
Signing Up for a Developer Account	422
Uploading an Application to the Android Market	423
Publishing on the Android Market	427
Using Other Developer Account Benefits	429
Exploring Other Android Publishing Options	429
Selling Your Application on Your Own Site	429
Selling Your Application on Other Markets	430

Part VI: Appendixes

APPENDIX A: Configuring Your Android Development Environment	437
Development Machine Prerequisites	437
Supported Operating Systems	437
Available Space	438
Installing the Java Development Kit	438
Installing the Eclipse IDE	438
Notes on Windows Installations	439
Notes on Mac OS X Installations	439
Installing the Android SDK Starter Package	439
Notes on Windows Installations	440
Notes on Mac OS X Installations	440
Notes on Linux OS Installations	440
Installing and Configuring the Android Plug-in for Eclipse (ADT)	440
Configuring Development Hardware for Device Debugging	443
Configuring Android Devices for Development Purposes	443
Configuring Your Operating System for Device Debugging	443
APPENDIX B: Eclipse IDE Tips and Tricks	445
Creating New Classes and Methods	445
Organizing Imports	445
Documenting Code	446
Using Auto-Complete	446
Editing Code Efficiently	447
Renaming Almost Anything	448
Formatting Code	448
Organizing Code	448
Fun with Refactoring	449
Resolving Mysterious Build Errors	450
Creating Custom Log Filters	451
Moving Panes Around in a Workspace	451

Customizing Panes in a Workspace	452
Integrating Source Control	452
APPENDIX C: Supplementary Materials	453
Using the Source Code for This Book	453
Accessing the Android Developer Website	454
Accessing the Publisher's Website	454
Accessing the Authors' Website	455
Contacting the Authors	456
Leveraging Online Android Resources	457
INDEX	459

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, geo-cached 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 back-packing, 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@sampublishing.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/web-search/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 walk-through, 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

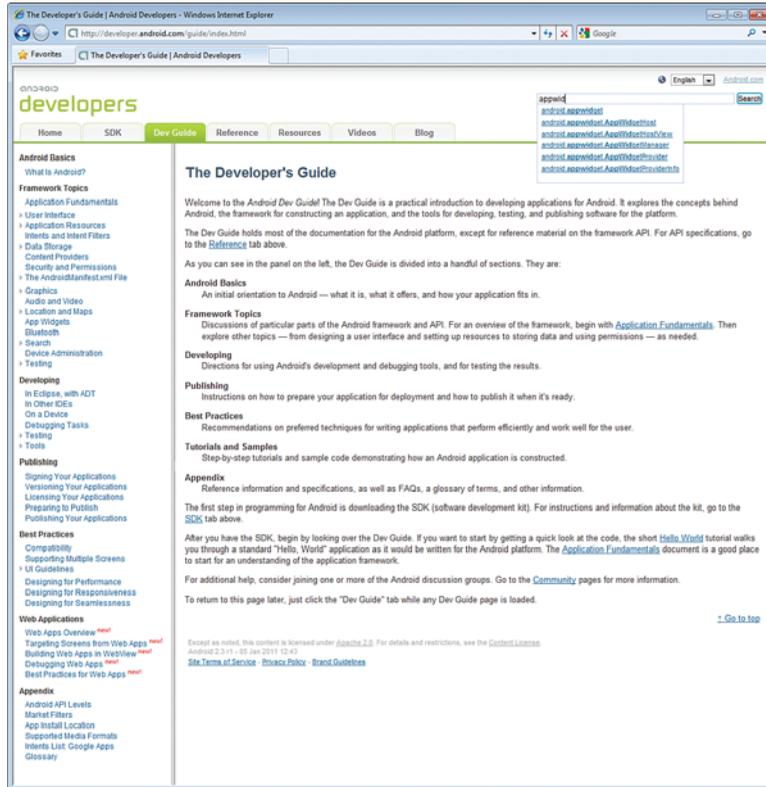
Android developers are fortunate to have more than a dozen development tools at their disposal to help facilitate the design of quality applications. Understanding what tools are available and what they can be used for is a task best done early in the Android learning process, so that when you are faced 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

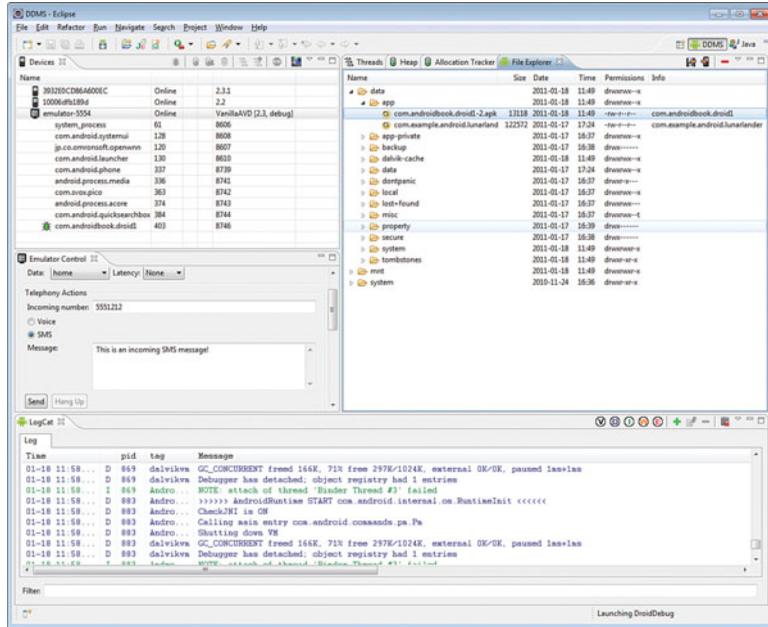
The Dalvik Debug Monitor Service (DDMS) is a debugging utility that is integrated into Eclipse through a special Eclipse perspective. The DDMS perspective provides a number of useful features for interacting with emulators and handsets and debugging applications (Figure 2.2).

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 ().

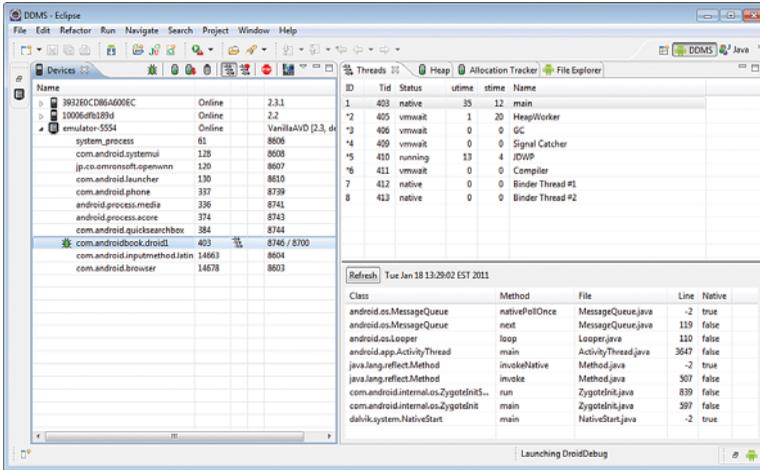


FIGURE 2.3 Using DDMS to examine thread activity for the Droid1 application.

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.

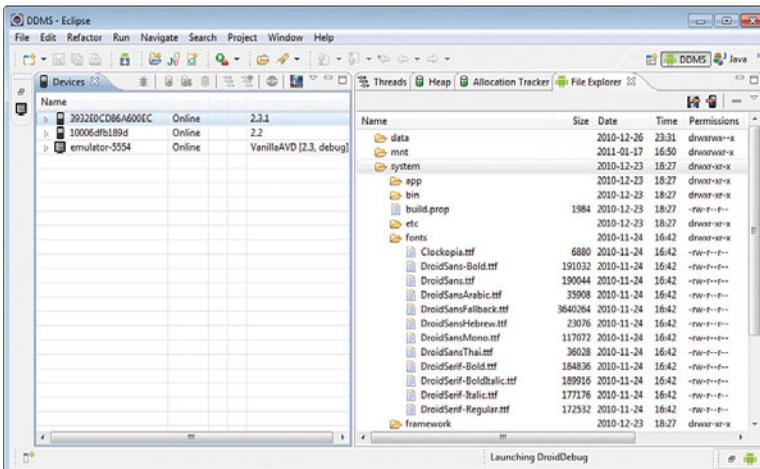


FIGURE 2.4 Using the DDMS File Explorer to browse system fonts on the handset.

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**

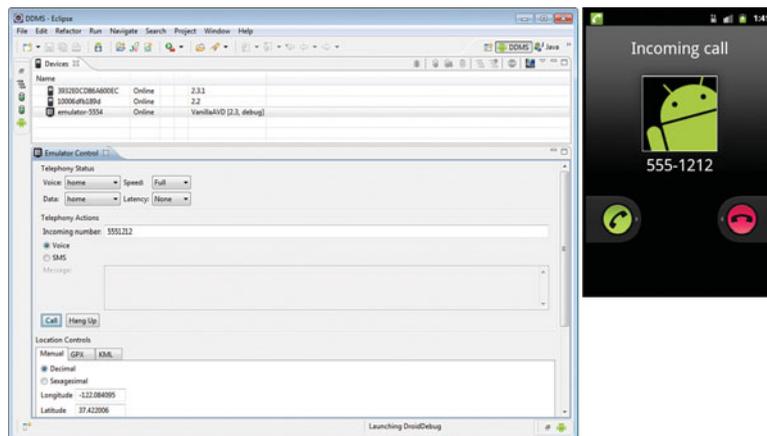
These features generally work for emulators only, not for handsets. For handsets, you must use real calls and real messages, which may incur fees (depending upon your plan).

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.

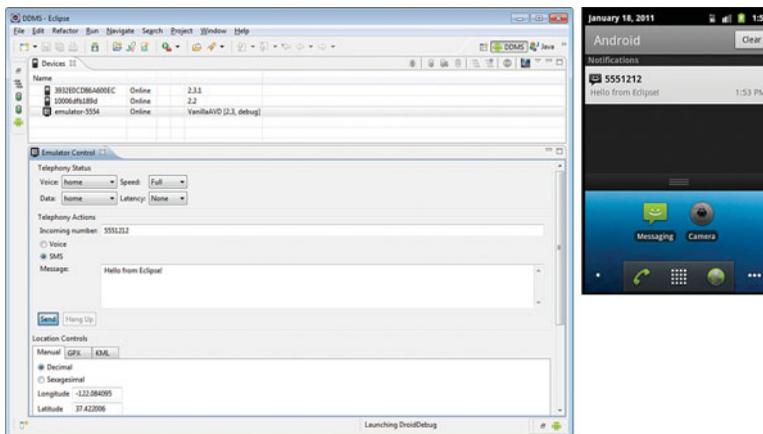


FIGURE 2.6
Using the DDMS Emulator Control tab (left) to send an SMS message to the emulator (right).

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

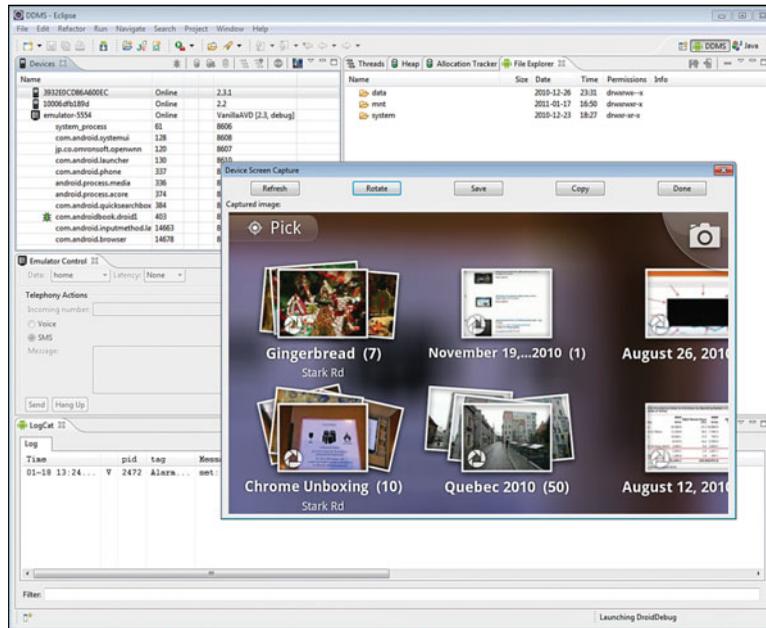
1. In the DDMS perspective, choose the emulator instance you want to send an SMS message to.
2. On the Emulator Control tab, navigate to the Telephony Actions section and input the Incoming number (for example, 5551212).
3. Select the SMS radio button.
4. Type an SMS message in the Message textbox.
5. 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

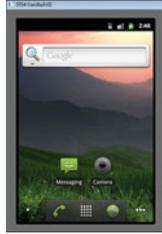
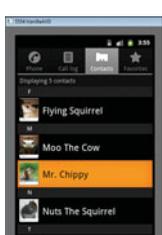
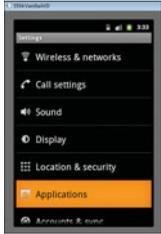
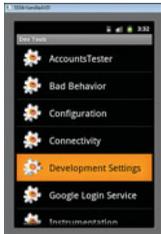
Feature	Description	Appearance
Home screen	Default screen. This is a common location for app widgets and live folders. You will also find a quick launch bar for the Dialer (📞) and Browser (🌐) applications as well as the Application menu.	
Dialer application	Built-in application for making and receiving 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

- Which features are available in the DDMS perspective?
 - Taking screenshots of emulator and handset screens
 - Browsing the file system of the emulator or handset
 - Monitoring thread and heap information on the Android system
 - Stopping processes
 - Simulating incoming phone calls and SMS messages to emulators
 - All of the above
- True or False: You must use the Android emulator for debugging.
- Which target platforms can Android applications be written for?
- 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

acceleration, 386

accessing

Android Developer website,
454

applications

functionality, 52

preferences, 51

author's website, 455

hardware, 386

applying Wi-Fi, 387

Bluetooth, 387

managing power, 387

reading raw sensor data,
386-387

LBS applications, 259-260

network applications,
274-276

publisher's website, 454

raw resource files, 156-157

accounts, developers

benefits of, 429

registering (Android Market),
422-423

**acquiring target devices,
395-396**

**ACTION_CREATE_LIVE_FOLDER,
385**

ACTION_IMAGE_CAPTURE, 241

**ACTION_RINGTONE_PICKER,
380**

activities

applications

applying, 52-56

implementing, 109-110

defining, 91-92

dialogs, 189

customizing, 196-201

DatePickerDialog class,
192-195

tracing life cycles of,
191-192

types of, 190-191

activities

- launching, 53, 234-235
- networks, 271
- options menus, adding, 146
- progress bars, 277-279
- remote, launching, 238
- shutting down, 56
- trivia game requirements, 100
- Activity class, 51**
- activity classes, implementing, 49**
- Activity.startActivityForResult() method, 54**
- adb command-line tool, 418**
- Add Friend dialog box, 313-314**
- adding**
 - animation, 126-128
 - Apache libraries, 293
 - application logic, 205
 - game screen design, 205-211
 - implementing, 215-223
 - ViewSwitcher controls, 211-215
 - avatars, 229-230
 - comments, 446
 - custom dividers, 143
 - custom selectors, 143
 - dialogs, settings screens, 196-201
 - drawable resources, 108
 - Facebook support, 320
 - filters, 451
 - JAR files, 302
 - layouts, 108
 - Listview template layouts, 139
 - OpenSocial support, 320
 - options menus, 145-147
 - password dialogs to classes, 198-199
 - raw resource files, 156
 - resources, 107, 120-122
 - colors, 175
 - friend requests, 311
 - game screens, 208
 - help screens, 153-154
 - main menu screens, 136-137
 - menus, 145
 - scores screens, 160-161
 - settings screens, 175-176
 - strings, 175-176
 - social features, 309-310
 - friend support, 311-318
 - integrating, 319-320
 - strings, 108
 - TabHost controls, 158
 - tests, 404
 - Twitter support, 320
 - XML files, 70
- addresses, translating coordinates, 262-263**
- addTab() method, 164**
- addView() method, 166**
- ADTs (Android Development Tools), 13, 21, 25**
- AlertDialog control, 196, 315**
- alerts, 372-373**
- Amazon Appstore, 431**
- anddev.org, 457**
- Android Developer Website, 454, 457**
- Android Development Tools. See ADTs**
- Android Hierarchy Viewer layout utility, 13**
- Android LogCat logging utility, 13**
- Android Market, 457**
 - applications
 - removing from, 429
 - uploading, 423-427
 - billing, 428
 - locales, handling, 348
 - publishing on, 421, 427-429
 - return policies, 428
 - selling on, 421, 423-427
- Android plug-in for Eclipse, installing, 440-442**
- Android Project Wizard, 13-20**
- Android SDK Starter Package, installing, 439**
- Android Virtual Devices. See AVDs**
- android.gesture package, 376**
- android.speech.RecognizerIntent, 377**
- AndroidManifest.xml tab, 86-87**
- animation, 379**
 - life cycle events, 129-130
 - resources, adding, 126-128
 - splash screens
 - adding resources, 120-122

- customizing, 126-130
 - design, 117-118
 - layouts, 118-119
 - updating layouts, 122, 125
 - views, 128-129
- Apache libraries, adding, 293**
- APIs (application programming interfaces)**
 - Fragments, 373
 - OpenSocial, adding, 320
- App Widgets**
 - creating, 325-336
 - Provider, implementing, 331
- Apple iPhone, 10**
- Application Manager, 113**
- Application tab, 84**
- applications**
 - activities, 52-56, 91-92, 109-110
 - Android Market
 - publishing on, 427-429
 - removing from, 429
 - selling on, 421-427
 - uploading, 423-427
 - attributes, configuring, 90-91
 - building, 47-50
 - Camera, 234
 - configuring, 87-91
 - context, 51-52
 - data management, 381
 - content providers, 384-385
 - files and directories, 382-383
 - integrating global searches, 385
 - DDMS, 25-26, 33-39
 - debugging, 21-22, 90
 - descriptions, 90
 - Dev Tools, 113
 - devices, 26-28
 - dialogs, 58
 - digital signatures, 414
 - emulators, 24
 - fragments, 59
 - frameworks
 - design, 99
 - implementing prototypes, 106-111
 - running prototypes, 111-113
 - trivia games, 99-105
 - functionality, 52
 - Gallery, 234
 - “Hello, World”, 13
 - icons, 90
 - information, 398
 - intents, 56-57
 - Java, 12
 - LBS
 - accessing, 259-260
 - applying, 254-257
 - design, 245
 - favorites, 246-254
 - geocoding services, 260-263
 - maps, 263-266
 - locales (internationalization), 346
 - logging, 60
 - logic
 - adding, 205
 - game screen design, 205-211
 - implementing, 215-223
 - ViewSwitcher controls, 211-215
 - maps, 263-264
 - naming, 90
 - networks
 - accessing, 274-276
 - design, 269-271
 - developing, 272-273
 - downloading score data, 280-286
 - parsing question batches, 287-289
 - progress bars, 277-279
 - running tasks asynchronously, 279-280
 - packaging, 414-416
 - permissions
 - declaring, 372
 - managing, 93-95
 - preferences
 - accessing, 51
 - creating, 110
 - projects, 14-20
 - publishing, 429-431
 - release processes, 409-410
 - prerelease practices, 411-413
 - testing, 413

applications

- resources
 - applying, 65-69
 - defining SDKs, 367
 - files, 77-79
 - layouts, 74-77
 - managing, 65
 - retrieving, 51
 - values, 69-73
 - return policies, 428
 - running, 21-22
 - settings, 96
 - signing, 414-418
 - social features
 - adding, 309-310
 - friend support, 311-318
 - integrating, 319-320
 - support, 366
 - testing
 - best practices, 391
 - developing coding standards, 392
 - implementing build processes, 393
 - maximizing coverage, 395-405
 - planning, 393-395
 - tracking bugs, 393
 - theft, protecting against, 412-413
 - uploading, 293
 - versions, 88
 - applying**
 - activity dialogs, 189
 - customizing, 196-201
 - DatePickerDialog class, 192-195
 - tracing life cycles of, 191-192
 - types of, 190-191
 - AndroidManifest.xml tab, 86-87
 - animation, 126-130
 - Application tab, 84
 - applications
 - activities, 52-56
 - context, 51-52
 - dialogs, 58
 - fragments, 59
 - intents, 56-57
 - logging, 60
 - avatars
 - bitmaps, 239-241
 - ImageButton controls, 231-234
 - selecting, 234-238
 - Bluetooth, 387
 - callbacks, 54
 - cameras, 235-236
 - colors, 70
 - defect tracking systems, 393
 - dimensions, 71
 - documentation, 31-33
 - drawable resources, 72-73
 - emulators, 39
 - files, help screens, 155-157
 - form controls, 178-183
 - graphics, 378
 - libraries, 379
 - OpenGL ES, 379
 - handlers, 280
 - images, 72
 - Instrumentation tab, 86
 - LBS applications, 254-257
 - ListView controls, 140-143
 - localization utilities, 351-352
 - Manifest tab, 84
 - maps, 263-266
 - MIME messages, 301
 - multimedia, 377
 - audio, 377
 - video, 378
 - Permission tab, 85
 - RemoteViews interfaces, 327-328
 - SD card images with emulators, 42
 - strings, 69
 - styles, App Widgets, 328-329
 - textEdit controls, 178
 - threads, 280
 - ViewSwitcher controls, 211-215
 - Wi-Fi, 387
 - XML, 165-167
- asynchronous tasks, friend requests, 315-317**
 - AsyncTask class, 279-281, 287**
 - attributes, configuring applications, 90-91**
 - audio, 377**
 - authors**
 - contacting, 456
 - websites, accessing, 455

Auto-complete, 446

automating

- builds, 13
- testing, 398-404

availability

- servers, 276
- versions, 12

available space, 438

avatars

- adding, 229-230
- bitmaps, 239-241
- design, 227
- ImageButton controls, 231-234
- posting, 302-304
- selecting, 234-238
- uploading, 301

AVDs (Android Virtual Devices), 13, 21

- configuring, 21-22
- Google APIs, 256

B

backgrounds

- processing, 289
- tasks, App Widgets, 331

backward compatibility, 366

batches, parsing questions, 287-289

batteries, managing, 387

BATTERY_STATS permission, 387

benefits of developer

accounts, 429

best practices, testing applications, 391

- developing coding standards, 392
- implementing build processes, 393
- planning, 393-395
- tracking bugs, 393

billing, Android Market, 428

bitmaps, avatars, 239-241

Bluetooth, applying, 387

brightness, 386

bugs, tracking, 393

building

- applications, 47-50
- activities, 52-56
- context, 51-52
- dialogs, 58
- fragments, 59
- frameworks, 99
- implementing prototypes, 106-111
- intents, 56-57
- logging, 60
- running prototypes, 111-113
- trivia games, 99-105
- errors, resolving, 450
- forms, 171
 - applying controls, 178-183
 - designing settings screens, 171-173

implementing settings

screen layouts, 175-178

saving data with SharedPreferences, 184-186

processes, implementing, 393

screens with tabs, 163-164

button controls, 179-180

C

Cadenhead, Rogers, 12

callbacks, activities, 54

calls, simulating incoming, 36-37

cameras, applying, 234-236

cases

- test, creating, 400-403
- edge, 222-223

classes

- activities, 49-51
- AsyncTask, 279, 287
- Context, 382
- creating, 445
- DatePickerDialog, 192-195
- GestureDetector, 375
- HttpClient, 302
- HttpPost, 302
- MyImageSwitcherFactory, 212-213
- MyTextSwitcherFactory, 212-213

classes

- password dialogs, adding to, 198-199
- QuizHelpActivity, 103
- QuizSettingsActivity, 295
- QuizTask, starting, 289
- ScoreDownloaderTask, 286
- Service, 296-297
- UploaderService, 297
- UploadTask, implementing, 298
- ViewFactory, 207
- ViewSwitcher, 206
- WidgetUpdateTask, 333-334
- clearAnimation() method, 128**
- clicks**
 - button, 180-181, 219-221
 - events, ImageButton controls, 233
- client/server testing, 394**
- code**
 - documenting, 446
 - editing, 447
 - formatting, 448
 - managing, 448
 - optimization, 13
 - repeating, 449-450
 - reviewing, 106
 - source, 453-454
 - standards, developing, 392
 - versions, 88
- code-signing tools, 13**
- colors**
 - applying, 70
 - resources, adding, 175
- comments, adding, 446**
- compatibility, backward, 366**
- compress() method, 240**
- concatenation, qualifiers, 356**
- configuring**
 - App Widgets, 325-336
 - applications
 - activities, 52-56
 - attributes, 90-91
 - building, 47-50
 - context, 51-52
 - defining activities, 91-92
 - dialogs, 58
 - fragments, 59
 - intents, 56-57
 - logging, 60
 - permissions, 93-95
 - settings, 87-96
 - AVDs, 21-22
 - button controls, 179-180
 - Debug configurations, 22, 112
 - development environments, 437-444
 - devices, debugging, 26
 - EditText controls, 178
 - locations, emulators, 256-257
 - management, 355-357
 - customizing screen orientations, 357-362
 - default settings, 363-364
 - developing SDK versions, 365-367
 - feature support, 364
 - modifying screen orientations, 362
 - network permissions, 275
 - projects, 14-20
 - Run configurations, 22
 - settings
 - synchronizing, 294
 - uploading, 295-304
 - Spinner controls, 182
 - TabHost controls, 163
 - textEdit controls, 178
- conformance testing, 395**
- contacting authors, 456**
- content providers, 384-385**
- context**
 - applications
 - applying, 51-52
 - launching activities using, 53
 - menus, 144
- Context class, 382**
- controls**
 - AlertDialog, 315
 - button, 179-180
 - EditText, 178
 - configuring, 178
 - listening for keystrokes, 199-201
 - forms, 178-183
 - ImageButton, 179, 230-234
 - ImageSwitcher, updating, 214-215
 - Layout, 119
 - LinearLayout, 173, 230, 250

DatePickerDialog class

- ListView, 135
 - main menu screens, 140-143
 - templates, 139
 - ProgressBar, 277
 - RelativeLayout, 135, 206
 - ScrollView, 173
 - Spinner, 182, 250
 - configuring, 182
 - events, 183
 - selections, 183
 - TabHost
 - adding, 158
 - configuring, 163
 - TextSwitcher, updating, 214
 - TextView, 20, 173, 230
 - VideoView, 378
 - View, 119
 - ViewGroup, 374
 - ViewSwitcher, 211-215
 - converting text to speech, 376**
 - coordinates, translating addresses, 262-263**
 - Copy button, 38**
 - copying files, 35-36**
 - costs, development, 11-12**
 - coverage, test, 395-405**
 - createScaledBitmap()**
 - method, 241
 - currencies, handling, 352**
 - customizing**
 - avatars, 227
 - adding, 229-230
 - bitmaps, 239-241
 - ImageButton controls, 231-234
 - selecting, 234-238
 - dialogs, 196-201
 - dividers, adding, 143
 - help screens, 151-152
 - applying files, 155-157
 - implementing, 153-155
 - interfaces, 373
 - input methods, 374
 - styles and themes, 373
 - user gestures, 375
 - views, 374
 - log filters, 451
 - main menu screens, 133-136
 - adding resources, 136-137
 - ListView control, 140-143
 - types of menu mechanisms, 144-147
 - updating, 138-139
 - network applications, 269-271
 - accessing, 274-276
 - developing, 272-273
 - downloading score data, 280-286
 - parsing question batches, 287-289
 - progress bars, 277-279
 - running tasks asynchronously, 279-280
 - panes, 452
 - password dialogs, launching, 201-202
 - preferences, 110
 - scores screens, 157
 - applying XML, 165-167
 - building with tabs, 163-164
 - implementing, 160-162
 - requirements, 158
 - screen orientations, 357-362
 - selectors, adding, 143
 - splash screens, 117-118
 - adding resources, 120-122
 - animation, 126-130
 - layouts, 118-119
 - updating layouts, 122, 125
- D**
- Dalvik Debug Monitor Service.**
 - See DDMS
 - data management, 381**
 - content providers, 384-385
 - files and directories, 382-383
 - integrating global searches, 385
 - databases, managing devices, 396**
 - DatePickerDialog class, 192-195**

dates**dates**

- date of birth, settings screens, 172
- formatting, 351-352

DDMS (Dalvik Debug Monitor Service), 25-26, 33-39**Debug button, 24****Debug configurations, creating, 22, 112****debugging, 13**

- applications, 21-22, 90
- DDMS, 25-26, 33-39
- Devices, configuring, 26
- emulators, 38-39
- handsets, 38-39
- hardware, configuring, 443-444

declaring

- application permissions, 372
- string literals, 218

defaults

- resources, internationalization, 346-347
- settings, managing, 363-364
- tabs, configuring, 164

defect tracking systems, applying, 393**defining**

- activities, 91-92
- dialogs, 191
- SDKs, 367

deleting

- dialogs, 192
- directories, 36
- files, 36

demographic information, 172**deployment, 13****descriptions of applications, 90****design**

- App Widgets, 325-336
- applications, 47-50
 - activities, 52-56
 - context, 51-52
 - dialogs, 58
 - fragments, 59
 - frameworks, 99
 - implementing prototypes, 106-111
 - intents, 56-57
 - logging, 60
 - running prototypes, 111-113
 - trivia games, 99-105
- avatars, 227
 - adding, 229-230
 - bitmaps, 239-241
 - ImageButton controls, 231-234
 - selecting, 234-238
- game screens, 205-211
- help screens, 151-152
 - applying files, 155-157
 - implementing, 153-155
- interfaces, 373-375
- landscape mode layouts, 361-362
- LBS, 245
 - accessing, 259-260
 - applying, 254-257
 - favorites, 246-254

geocoding services, 260-263

maps, 263-266

main menu screens, 133-136

adding resources, 136-137

landscape mode layouts, 360

ListView control, 140-143

types of menu mechanisms, 144-147

updating, 138-139

network applications, 269, 271

accessing, 274-276

developing, 272-273

downloading score data, 280-286

parsing question batches, 287-289

progress bars, 277-279

running tasks asynchronously, 279-280

scores screens, 157

applying XML, 165-167

building with tabs, 163-164

implementing, 160-162

requirements, 158

settings screens, 171-173

applying controls, 178-183

implementing layouts, 175-178

- saving data with SharedPreferences, 184-186
- splash screens, 117
 - adding resources, 120-122
 - customizing, 126-130
 - landscape mode layouts, 359
 - layouts, 118-119
 - updating layouts, 122, 125
- designating launch activities, 92**
- detecting SDKs, 367**
- Dev Tools application, 113**
- Developer.com, 457**
- developers, accounts**
 - benefits of, 429
 - registering, 422-423
- development**
 - ADT, 13
 - App Widgets, 326
 - code standards, 392
 - configuration management, 355-357
 - customizing screen orientations, 357-362
 - feature support, 364
 - managing default settings, 363-364
 - modifying screen orientations, 362
 - SDK versions, 365-367
 - cost of, 11-12
 - environments, configuring, 437-444
 - features, navigating, 371-373
 - hardware, configuring debugging, 443-444
 - network applications, 272-273
 - tools, 113
 - DDMS, 33-39
 - documentation, 31-33
 - managing tasks, 34-35
 - navigating files, 35-36
- Device Screen Capture tool, 38**
- devices, 10**
 - applications, launching, 26-28
 - configuration management, 355-357
 - customizing screen orientations, 357-362
 - developing SDK versions, 365-367
 - feature support, 364
 - managing default settings, 363-364
 - modifying screen orientations, 362
 - databases, managing, 396
 - debugging, 26, 443-444
 - fragmentation, 396
 - hardware
 - accessing, 386
 - applying Wi-Fi, 387
 - Bluetooth, 387
 - managing power, 387
 - reading raw sensor data, 386-387
 - multimedia, 377-378
 - navigating, 40-42
 - personalizing, 380
 - live wallpaper, 381
 - ringtones, 380
 - wallpaper, 380
 - target
 - identifying, 395-396
 - testing, 398
 - virtual, managing, 21-22
- diagnostic logging, 398**
- dialogs**
 - activities, 189
 - customizing, 196-201
 - DatePickerDialog class, 192-195
 - tracing life cycles of, 191-192
 - types of, 190-191
 - Add Friend, 313-314
 - applications, applying, 58
 - favorites, LBS applications, 247-254
 - input, 189
 - progress bars, 278
- digital signatures, 414**
- dimensions**
 - applying, 71
 - game screens, adding, 209
 - resources, Favorite Place feature, 249

directories

directories. *See also* files

- deleting, 36
- managing, 382-383
- projects, navigating, 16
- resources, 356

`dismiss()` method, 278

`dismissDialog()` method, 191

dismissing dialogs, 192

displays, 359. *See also* screens

distance, measuring, 386

dividers, adding customization, 143

documentation, 31-33

- code, 446
- internationalization, 343
 - Android Market, 348
 - applications, 346
 - default resources, 346-347
 - localization utilities, 351-352
 - operating systems, 345-346
 - strategies, 349-351
 - workflow, 393-395

`doFriendRequest()` method, 315

`doInBackground()` method, 284, 289, 298

downloading scores, 280-286, 305

drawable resources

- adding, 108
- applying, 72-73

driving quizzes forward, 219-221

E

Eckel, Bruce, 12

Eclipse

- Android plug-in for Eclipse, 440-442
- automated testing
 - with, 399
- debuggers, attaching, 24
- installing, 438
- navigating, 13
- optimizing, 445-452

edge cases, 222-223, 395

editing

- code, 447
- files
 - manifest, 18-19
 - resources, 19-20
- projects, 17-18
- string resources, 20

EditText control, 178, 199-201

email, settings screens, 172

emulators, 13, 36

- applications, launching, 24
- applying, 39
- AVDs, configuring, 21-22
- incoming
 - calls, 36-37
 - SMS messages, 37
- input, providing, 40
- locations
 - configuring, 256-257
 - testing, 255
- networks, testing, 272
- prototypes, launching in, 112-113

screenshots of, 38-39

SD card images, applying with, 42

testing, 397

enabling friend requests, 311-314

encapsulating images, 73

enforcing application permissions, 372

environments

- development, configuring, 437-444
- testing, managing, 395-396

events

- animation life cycles, 129-130
- clicks, ImageButton controls, 233
- ListView controls, listening for, 141-142
- Spinner controls, 183

exporting package files, 415-416

extending AsyncTask class, 287

external services, release processes, 413

Extract Local Variable tool, 449

F

Facebook support, adding, 320

Favorite Place feature

- implementing, 248-249
- updating, 250

- favorites, LBS applications, 246-254**
- features, 48**
 - high-level game, determining, 100
 - navigating, 371-373
 - support, 364
- FierceDeveloper, 457**
- files**
 - application package (.apk), 417-418
 - deleting, 36
 - editing, 17-18
 - help screens, 155-157
 - JAR, adding, 302
 - managing, 382-383
 - manifest
 - App Widgets, 335-336
 - application permissions, 93-95
 - application settings, 96
 - configuring application settings, 87-91
 - defining activities, 91-92
 - editing, 18-19
 - launching activities in, 53
 - managing, 83
 - navigating, 83-87
 - preparing for release, 411-412
 - updating, 110
 - navigating, 35-36
 - packages, exporting, 415-416
 - projects, navigating, 16
 - raw
 - accessing, 156-157
 - adding, 156
 - applying, 78
 - resources
 - applying, 77-79
 - editing, 19-20
 - XML
 - adding, 70
 - parsing, 165-166
- filling ListView controls, 140-141**
- filter logs, customizing, 451**
- findViewById() method, 140, 163**
- finish() method, 56**
- folder content, managing, 385**
- fonts, 379**
- forgoing application internationalization, 349**
- formatting**
 - App Widgets, 325-336
 - Auto-complete, 446
 - avatars, 227
 - adding, 229-230
 - bitmaps, 239-241
 - ImageButton controls, 231-234
 - selecting, 234-238
 - classes, 445
 - code, 448
 - colors, 70
 - dates, 351-352
 - files, 77-79
 - help screens, 151-152
 - applying files, 155-157
 - implementing, 153-155
 - images, 72
 - interfaces, 373-375
 - layouts, 74-77
 - LBS, 245
 - accessing, 259-260
 - applying, 254-257
 - favorites, 246-254
 - geocoding services, 260-263
 - maps, 263-266
 - main menu screens, 133-136
 - adding resources, 136-137
 - ListView control, 140-143
 - types of menu mechanisms, 144-147
 - updating, 138-139
 - manifest files
 - applications, 93-96
 - configuring application settings, 87-91
 - defining activities, 91-92
 - methods, 445
 - network applications, 269-271
 - accessing, 274-276
 - developing, 272-273
 - downloading score data, 280-286
 - parsing question batches, 287-289

formatting

- progress bars, 277-279
- running tasks asynchronously, 279-280
- preferences, 110
- projects, 14-15, 107
- ringtones, 380
- scores screens, 157
 - applying XML, 165-167
 - building with tabs, 163-164
 - implementing, 160-162
 - requirements, 158
- settings
 - synchronizing, 294
 - uploading, 295-304
- splash screens, 117
 - adding resources, 120-122
 - customizing, 126-130
 - layouts, 118-119
 - updating layouts, 122, 125
- time, 351-352
- wallpaper, 380-381

forms, building, 171

- applying controls, 178-183
- designing settings screens, 171-173
- implementing settings
 - screen layouts, 175-178
- saving data with
 - SharedPreferences, 184-186

fragments

- applications, applying, 59
- devices, 396

Fragments API, 373**frameworks, applications**

- design, 99
- implementing prototypes, 106-111
- running prototypes, 111-113
- trivia games, 99-105

friends

- downloading scores, 305
- requests, enabling, 311-314
- Scores of Friends tab, 318
- support, adding, 311-318

functionality

- applications, 50-52
- testing, 394

G**Gallery, 234, 237****games**

- logic, implementing, 215-223
- screens
 - adding options menus, 145-147
 - design, 205-211
 - feature, 105
 - landscape mode layouts, 361-362
 - settings, adding, 216-217

gender, settings screens, 172

geocoding services, 260-263

GestureDetector class, 375**gestures, user interfaces, 375**

- getApplicationContext()**
 - method, 51
- getAttributeValue()**
 - method, 166
- getConfiguration() method, 351**
- getFromLocationName()**
 - method, 262
- GetJar, 431**
- getLastKnownLocation()**
 - method, 259
- getResources() method, 51**
- getSharedPreferences()**
 - method, 51
- getSystemService() method, 387**
- getText() method, 179**
- global searches, integrating, 385**

Google

- APIs, 256, 265
- checkout, 428
- Open Handset Alliance, 9-10

gradients, 379**Graphical Layout editors, 13****graphics. See also images**

- 2D/3D, 278-279
- animation, 126-130
- applying, 378
- avatars
 - adding, 229-230
 - bitmaps, 239-241
 - design, 227

- ImageButton controls, 231-234
 - selecting, 234-238
- cameras, 235-236
- libraries, 379
- managing, 65
- OpenGL ES, 379
- grouping application resources, 66**

H

- Handango, 431**
- handlers, applying, 280**
- handling currencies, 352**
- handsets, 13. See also devices**
 - manufacturers, 9
 - screenshots of, 38-39
- hardware**
 - accessing, 386-387
 - debugging, 443-444
 - networks, 273
- hashtables, storing question data, 218-219**
- “Hello, World”, 13**
- help screens**
 - design, 151-152
 - applying files, 155-157
 - implementing, 153-155
 - features, 103
- HelpActivity, 49**
- high-level game features, determining, 100**

- home screens, adding App Widgets, 336-338**
- HTTP (Hypertext Transfer Protocol), 276-277**
 - GET method, 299-301
 - POST method, 301
- HttpClient class, 302**
- HttpPost class, 302**

I

- icons, creating applications, 90**
- identifying target devices, 395-396**
- IDEs (integrated development environments), 12**
- ImageButton controls, 179, 230-234**
- images**
 - applying, 72
 - avatars
 - adding, 229-230
 - bitmaps, 239-241
 - design, 227
 - ImageButton controls, 179, 231-234
 - selecting, 234-238
 - cameras, 235-236
 - managing, 65
 - programming, 73
 - SD cards, 42
- ImageSwitcher control, updating, 214-215**
- IME (input method editors), 374**

- implementing**
 - activity classes, 49
 - Add Friend dialog box, 313-314
 - App Widget Provider, 331
 - applications
 - activities, 109-110
 - internationalization, 350-351
 - prototypes, 106
 - build processes, 393
 - Favorite Place feature, 248-249
 - friend requests, 314-315
 - games
 - logic, 215-223
 - screen layouts, 208-209
 - help screen layouts, 153-157
 - main menu screens, 133-136
 - adding resources, 136-137
 - ListView control, 140-143
 - types of menu mechanisms, 144-147
 - updating, 138-139
 - password dialog layouts, 198
 - scores screens, 157
 - applying XML, 165-167
 - building with tabs, 163-164
 - layouts, 160-162
 - requirements, 158

implementing

- settings screen layouts, 175-178
- splash screens
 - adding resources, 120-122
 - customizing, 126-130
 - design, 117-118
 - layouts, 118-119
 - updating layouts, 122, 125
- UploadTask class, 298
- WidgetUpdateTask class, 333-334
- imports, managing, 445**
- incoming**
 - calls, simulating, 36-37
 - SMS message, simulating, 37
- initializing**
 - DatePickerDialog classes, 194
 - dialogs, 191
 - switcher controls, 212
- input**
 - dialogs
 - activities, 189
 - customizing, 196-201
 - DatePickerDialog class, 192-195
 - tracing life cycles of, 191-192
 - types of, 190-191
 - emulators, providing, 40
 - forms
 - applying controls, 178-183
 - building, 171
 - designing settings screens, 171-173
 - implementing settings screen layouts, 175-178
 - saving data with SharedPreferences, 184-186
 - interfaces, 374
 - text, 179
- input method editors (IMEs), 374**
- installing**
 - Android plug-in for Eclipse, 440-442
 - Android SDK Starter Packages, 439
 - Eclipse IDEs, 438
 - JDKs, 438
 - prototypes, navigating, 113
 - signed application packages, 417-418
- Instrumentation tab, 86**
- integrated development environments. See IDEs**
- integrating, 13**
 - global searches, 385
 - social networking services, 319-320
 - source controls, 452
 - testing, 394
- intents, applying applications, 56-57**
- interfaces, 13**
 - design, 48, 373
 - input methods, 374
 - styles and themes, 373
 - user gestures, 375
 - views, 374
 - formatting, 74-77
 - Google APIs, 256, 265
 - RemoteViews interfaces, 327-328
 - sizing, 71
 - splash screens
 - adding resources, 120-122
 - customizing, 126-130
 - design, 117-118
 - layouts, 118-119
 - updating layouts, 122, 125
 - trivia games, 101-105
- internationalization**
 - Android Market, 348
 - applications, 346
 - default resources, 346-347
 - localization utilities, 351-352
 - operating systems, 345-346
 - overview of, 343
 - principles, 341-342
 - strategies, 349-351
 - testing, 394

J

JAR files, adding, 302

Java, 12

Java Development Kit. *See* JDK

Java Runtime Environment, 438

Javadoc comments, adding, 446

JDK (Java Development Kit), installing, 438

JRE (Java Runtime Environment), 438

JUnit, automated testing with, 399

K-L

keys, private, 415

landscape mode

game screen layouts, 361-362

main menu screen layouts, 360

splash screen layouts, 359

languages, internationalization, 343

Android Market, 348

applications, 346

default resources, 346-347

localization utilities, 351-352

operating systems, 345-346

strategies, 349-351

last known locations, retrieving, 259

Launch Configuration screen, 24

launching

Android, 10

activities, 53, 92, 234-235

applications

devices, 26-28

from emulators, 24

intents, 57

custom password dialogs, 201-202

DatePickerDialog class, 195

dialogs, 192

maps, 263-264

prototypes in emulators, 112-113

remote activities, 238

layouts

Add Friend dialog box, 313-314

AppWidgets, designing, 329-331

avatars, adding, 229-230

friend requests, updating, 312-313

game screens

implementing, 208-209

landscape mode, 361-362

updating, 210-211

help screens

applying files, 155-157

implementing, 153-155

LBS applications, updating, 246

main menu screens

adding resources, 136-137

design, 134-136

landscape mode, 360

ListView control, 140-143

types of menu mechanisms, 144-147

updating, 138-139

panes

customizing, 452

moving, 451

passwords, implementing, 198

resources

adding, 108

applying, 74-77

screen orientations, 358-359

settings screens, 175-178, 250

splash screens, 118-119

adding resources, 120-122

customizing, 126-130

landscape mode, 359

updating, 122, 125

LBS (location-based services) design, 245

accessing, 259-260

applying, 254-257

favorites, 246-254

LBS (location-based services) design

- geocoding services, 260-263
- maps, 263-266
- libraries**
 - adding, 293
 - graphics, 379
- life cycles**
 - activity dialogs, tracing, 191-192
 - animation events, 129-130
- limiting application internationalization, 350**
- LinearLayout control, 173, 230, 250**
- Linux, 12, 440**
- listening**
 - for ListView events, 141-142
 - for Spinner control selection events, 183
- ListView control, 135**
 - main menu screens, 140-143
 - templates, 139
- literals, declaring strings, 218**
- live folders, managing content, 385**
- live wallpaper, creating, 381**
- localization utilities, 351-352**
- location-based services. See LBS**
- locations**
 - emulators, configuring, 256-257
 - systems, determining locales, 351

- testing, 255
- updating, 260
- log information, viewing, 39**
- LogCat logging tool, 26**
- logic**
 - application
 - adding, 205
 - game screen design, 205-211
 - implementing, 215-223
 - ViewSwitcher controls, 211-215
 - games, 215-223
- logs**
 - applications, 60, 398
 - filters, customizing, 451
- long-clicks, ImageButton controls, 233**

M**Mac OS X, 12**

- Android SDK Starter Packages, installing, 440
- Eclipse IDEs, installing, 439
- operating systems, debugging, 444

magnetism, 386**main menu screens**

- adding resources, 136-137
- features, 102
- implementing, 133-136
- landscape mode layouts, 360

- ListView control, 140-143
- types of menu mechanisms, 144-147
- updating, 138-139
- makeView() method, 213**
- managedQuery() method, 384**
- managing**
 - activity state, 54
 - App Widget update services, 334-335
 - code, 448
 - configuration, 355-357
 - customizing screen orientations, 357-362
 - developing SDK versions, 365-367
 - feature support, 364
 - managing default settings, 363-364
 - modifying screen orientations, 362
 - data, 381
 - content providers, 384-385
 - files and directories, 382-383
 - integrating global searches, 385
 - default settings, 363-364
 - devices, databases, 396
 - imports, 445
 - manifest files, 83
 - applications, 93-96
 - configuring application settings, 87-91

- defining activities, 91-92
- navigating, 83-87
- power, 387
- resources, 65
 - applying, 65-69
 - files, 77-79
 - layouts, 74-77
 - values, 69-73
- tasks, 34-35
- testing environments, 395-396
- virtual devices, 21-22
- manifest files**
 - activities, 53
 - App Widgets, 335-336
 - editing, 18-19
 - managing
 - applications, 93-96
 - configuring application settings, 87-91
 - defining activities, 91-92
 - navigating, 83-87
 - release processes, preparing for, 411-412
 - updating, 110
- Manifest tab, 84**
- maps, applying, 263-266**
- markets, selling applications, 430-431**
- master layouts, main menu screens, 138**
- maximizing test coverage, 395-405**
- measurements, units of, 71**
- MediaRecorder, 378**
- MenuActivity, 49**
- menus**
 - context, 144
 - main
 - adding resources, 136-137
 - implementing screen, 134
 - implementing screens, 133-136
 - ListView control, 140-143
 - types of menu mechanisms, 144-147
 - updating, 138-139
 - options, 144-147
 - resources, adding, 145
- messages**
 - incoming SMS, simulating, 37
 - MIME, 301
- methods**
 - Activity.startActivityForResult(), 54
 - addTab(), 164
 - addView(), 166
 - clearAnimation(), 128
 - compress(), 240
 - createScaledBitmap(), 241
 - creating, 445
 - dismiss(), 278
 - dismissDialog(), 191
 - doFriendRequest(), 315
 - doInBackground(), 284, 289, 315
 - findViewById(), 140, 163
 - finish(), 56
 - getApplicationContext(), 51
 - getAttributeValue(), 166
 - getConfiguration(), 351
 - getFromLocationName(), 262
 - getLastKnownLocation(), 259
 - getResources(), 51
 - getSharedPreferences(), 51
 - getSystemService(), 387
 - getText(), 179
 - HTTP, 299-301
 - Input, interfaces, 374
 - makeView(), 213
 - managedQuery(), 384
 - notify(), 373
 - onActivityResult(), 54
 - onBind(), 297
 - onCancelled(), 283-284
 - onCreate(), 140
 - onCreateDialog(), 191, 315
 - onCreateOptionsMenu(), 146
 - onDateSet(), 193
 - onLongClick(), 233
 - onOptionsItemSelected(), 146
 - onPause(), 128, 298
 - onPickDateButtonClicked(), 180, 195
 - onPostExecute(), 282-283, 288
 - onPreExecute(), 282, 288, 298
 - onPrepareDialog(), 191, 194

methods

onProgressUpdate(), 285
 onSetPasswordButtonClick(), 180
 onStartCommand(), 297
 onTouchEvent(), 375
 onUpdate(), 334
 postAvatarToServer(), 303
 removeDialog(), 191
 requestLocationUpdates(), 260
 requestRouteToHost(), 276
 setCurrentTabByTag(), 164
 setImageBitmap(), 231
 setImageDrawable(), 231
 setImageResource(), 231
 setImageURI(), 231-232
 setProgress(), 277
 setSelection(), 183
 setText(), 179
 setup(), 164
 showDialog(), 191, 201
 startActivity(), 53
MiKandi.com, 431
MIME, 301
modes, landscape
 game screen layouts, 361-362
 main menu screen layouts, 360
 splash screen layouts, 359
modifying screen orientations, 362
monitoring, 13, 26
monkey tools, 395

moving
 files, 35-36
 panes, 451
multimedia, 377-378
MyImageSwitcherFactory class, 212-213
MyTextSwitcherFactory class, 212-213

N

naming
 applications, 90
 filters, 451
 packages, 88
 settings screens, 171
 versions, 88
navigating
 devices, 40-42
 documentation, 32
 Eclipse, 13
 features, 371-373
 files, 35-36
 manifest files, 83-87
 project files, 16
 prototypes, installing, 113

networks

applications
 accessing, 274-276
 design, 269-271
 developing, 272-273
 downloading score data, 280-286

 parsing question batches, 287-289
 progress bars, 277-279
 running tasks asynchronously, 279-280
 emulators, testing on, 272
 hardware, testing on, 273
 HTTP, 276-277
 permissions, configuring, 275
 servers
 determining data to send, 293
 scores, 304
 settings, 295-304
 synchronizing applications, 294
 social features, integrating, 319-320
 status, 275

nicknames, settings screens, 171

Notification object, 373

NotificationManager system service, 373

notifications, 372-373

notify() method, 373

O

obfuscation, 13

objects

Notification, 373
 SensorManager, 387

onActivityResult() method, 54

- onBind() method, 297
- onCancelled() method, 283-284
- onCreate() method, 140
- onCreateDialog() method, 191, 315
- onCreateOptionsMenu()
method, 146
- onDataSet() method, 193
- onLongClick() method, 233
- onOptionsItemSelected()
method, 146
- onPause() method, 128, 298
- onPickDateButtonClick()
method, 180, 195
- onPostExecute() method,
282-283, 288
- onPreExecute() method, 282,
288, 298
- onPrepareDialog() method,
191, 194
- onProgressUpdate() method,
285
- onSetPasswordButtonClick()
method, 180
- onStartCommand() method,
297
- onTouchEvent() method, 375
- onUpdate() method, 334
- Open Handset Alliance, 9-10,
457
- OpenGL ES, graphics, 379
- OpenIntents, 457
- OpenSocial support, adding,
320
- operating systems, 12
 - debugging, 443
 - locales (internationaliza-
tion), 345-346
 - requirements, 437-438
 - support, 437
- optimizing
 - applications
 - adding social features,
309-310
 - friend support, 311-318
 - integrating social fea-
tures, 319-320
 - code, 13
 - Eclipse, 445-452
 - player relationships,
318-319
 - scores screens, 166-167
- options. *See also* customizing
 - adding, 145-147
 - menus, 144
- organizing. *See* managing
- orientations, 386
 - customizing, 357-362
 - developing SDK versions,
365-367
 - feature support, 364
 - managing default settings,
363-364
 - modifying, 362
- P**
- packages
 - files, exporting, 415-416
 - naming, 88
- packaging, 13
 - applications, 414-416
 - avatars, 302-304
- paints, 379
- panes
 - customizing, 452
 - moving, 451
- parsing
 - question batches, 287-289
 - question data, 217
 - XML
 - declaring string literals,
218
 - files, 165-166
- passing information with
intents, 56
- passwords
 - layouts, implementing, 198
 - settings screens, 172
- performance, testing, 395
- Permission tab, 85
- permissions
 - applications, managing,
93-95
 - networks, configuring, 275
- personal websites, selling appli-
cations on, 429-430
- personalizing devices, 380-381
- perspective, adding DDMS, 25
- piracy, ProGuard, 412-413
- planning
 - publishing
 - prerelease practices,
411-413
 - release processes,
409-410
 - testing, 413
 - testing, 393-395
- platforms, overview of, 9

PlayActivity

PlayActivity, 49

players

- relationships
 - optimizing, 318-319
 - support, 310
- uploading data, 299-301

playing

- audio, 377
- video, 378

plug-ins

- ADT, 13
- Android plug-in for Eclipse, 440-442

policies, return, 428

postAvatarToServer() method, 303

posting avatars, 302-304

power, managing, 387

preferences. See also customizing

- applications
 - accessing, 51
 - creating, 110
 - retrieving, 111
 - saving, 110

prerelease practices, 411-413

prerequisites, 437-438

principles, internationalization, 341-342

private keys, 415

processes

- background, 289
- builds, implementing, 393
- releases, 409-410
 - packaging and signing applications, 414-416

prerelease practices, 411-413

testing, 413

programming

- code, reviewing, 106
- images, 73
- layouts, 77
- SDKs, detecting, 367

ProgressBar control, 277

ProGuard, 13, 412-413

projects

- Android Project Wizard, 13-20
- creating, 14-20, 107
- Debug configurations, creating, 22
- editing, 17-18
- files, navigating, 16
- JAR files, adding, 302
- resources
 - adding, 107, 120-122
 - friend requests, 311
- Run configurations, creating, 22
- testing, 399-400

properties, App Widgets, 326

prototypes

- emulators, launching in, 112-113
- implementing, 106-111
- installing, 113
- running, 111, 113

providers

- App Widget Provider, implementing, 331
- content, 384-385
- LBS, applying, 260

publisher websites, accessing, 454

publishing

- on Android Market, 421, 427-429
- options, 429-431
- releases, 409-410
 - packaging and signing applications, 414-416
 - prerelease practices, 411-413
 - testing, 413

Push files, moving, 35-36

Q

qualifiers, resource directories, 356

questions, parsing batches, 287-289

QuizHelpActivity class, 103

QuizSettingsActivity class, 295

QuizTask class, starting, 289

quizzes, driving forward, 219-221

R

raw files

- accessing, 156-157
- adding, 156
- applying, 78

raw sensor data, reading, 386-387

- reading raw sensor data, 386-387**
 - recording**
 - audio, 377
 - video, 378
 - refactoring, 449-450**
 - references, application resources, 68**
 - Refresh button, 38**
 - registering**
 - activities, 91
 - developer accounts (Android Market), 422-423
 - relationships, players**
 - optimizing, 318-319
 - support, 310
 - RelativeLayout control, 135, 206**
 - release processes, 409-410**
 - deployment, 13
 - packaging and signing applications, 414-416
 - prerelease practices, 411-413
 - testing, 413
 - remote activities, launching, 238**
 - remote servers**
 - scores, uploading, 304
 - settings, uploading, 295-304
 - RemoteViews interfaces, applying, 327-328**
 - removeDialog() method, 191**
 - removing**
 - applications from Android Market, 429
 - dialogs, 192
 - Rename tool, 448**
 - repeating code, 449-450**
 - requestLocationUpdates() method, 260**
 - requestRouteToHost() method, 276**
 - requests, friends**
 - enabling, 311-314
 - implementing, 314-315
 - requirements**
 - activities, 49, 100
 - main menu screen layout, 134
 - operating systems, 437-438
 - scores screen layouts, 158
 - resolving build errors, 450**
 - resources**
 - adding, 120-122
 - animation, 126-128
 - applications
 - defining SDKs, 367
 - retrieving, 51
 - colors, 175
 - directories, 356
 - documentation, 31-33
 - drawable
 - adding, 108
 - applying, 72-73
 - Favorite Place feature, 249
 - files, editing, 19-20
 - friend requests, enabling, 311
 - game screens, 208
 - help screens, adding, 153-154
 - internationalization, 346-347
 - layouts, adding, 108
 - main menu screens, adding, 136-137
 - managing
 - applying, 65-69
 - files, 77-79
 - layouts, 74-77
 - values, 69-73
 - menus, adding, 145
 - projects
 - adding, 107
 - editing, 17-18
 - raw files
 - accessing, 156-157
 - adding, 156
 - scores screens, adding, 160-161
 - settings screens, adding, 175-176
 - strings
 - adding, 108, 175-176
 - editing, 20
 - websites, 457
- results, activities, 53**
- retrieving**
 - application resources, 51
 - avatars, 302-304
 - last known locations, 259
 - network status, 275
 - preferences, 111

retrieving

- question data, 217
- XML resources, 165
- return policies, Android Market, 428
- reviewing
 - manifest files, 411-412
 - source code, 106
- RIM BlackBerry, 10
- ringtones, formatting, 380
- Rotate button, 38
- Run configurations, creating, 22
- running
 - applications, 21-22
 - automated tests, 403-404
 - prototypes, 111-113
 - tasks asynchronously, 279-280

S

- Sam's Teach Yourself Java in 24 Hours*, 12
- satellite triangulation, 265. *See also* locations
- saving
 - activities, 55
 - application resources, 66
 - avatars, 234-238
 - bitmaps, 240
 - data with
 - SharedPreferences, 184-186
 - preferences, 110
 - question data, 217-219

- SAX (Simple API for XML), 165
- scaling bitmaps, 241
- ScoreDownloaderTask class, 286
- scores
 - design, 157
 - applying XML, 165-167
 - building with tabs, 163-164
 - implementing, 160-162
 - requirements, 158
 - downloading, 280-286, 305
 - features, 103
 - uploading, 304
- Scores of Friends tab, 318
- ScoresActivity, 49
- screens
 - design, 48
 - game
 - design, 205-211
 - landscape mode layouts, 361-362
 - help
 - applying files, 155-157
 - design, 151-152
 - implementing, 153-155
 - Home, adding App Widgets, 336-338
 - main menu
 - adding resources, 136-137
 - implementing, 133-136
 - landscape mode layouts, 360
 - ListView control, 140-143

- types of menu mechanisms, 144-147
- updating, 138-139
- options, 145-147
- orientations
 - customizing, 357-362
 - developing SDK versions, 365-367
 - feature support, 364
 - managing default settings, 363-364
 - modifying, 362
- scores. *See also* scores
 - applying XML, 165-167
 - building with tabs, 163-164
 - design, 157
 - implementing, 160-162
 - requirements, 158
- settings
 - adding dialogs, 196-201
 - avatars, 229-230
 - enabling friend requests, 311-314
 - Favorite Place feature, 250
- splash
 - adding resources, 120-122
 - customizing, 126-130
 - design, 117-118
 - landscape mode layouts, 359
 - layouts, 118-119
 - updating layouts, 122, 125
- trivia games, 101-105

- screenshots, emulators, 38-39
- ScrollView control, 173
- SD cards, 42
- SDKs (Software Development Kits), 10
 - defining, 367
 - detecting, 367
 - localization utilities, 351-352
 - operating systems, 345-346
 - overview of, 343
 - strategies, 349-351
 - internationalization
 - Android Market, 348
 - applications, 346
 - default resources, 346-347
 - versions, 89, 365-367
 - searching, integrating global searches, 385
 - security, ProGuard, 412-413
 - selecting
 - avatars, 234-238
 - options menus, 146-147
 - Spinner controls, 183
 - selectors, customizing, 143
 - selling applications
 - on markets, 430-431
 - on personal websites, 429-430
 - SensorManager object, 387
 - sequences, animation, 128-129
 - servers
 - applications, 270-271
 - availability, 276
 - networks
 - determining data to send, 293
 - scores, 304
 - synchronizing applications, 294
 - uploading settings, 295-304
 - Service class, 296-297
 - services
 - App Widgets, 332-335
 - release processes, 413
 - social networking, integrating, 319-320
 - setCurrentTabByTag() method, 164
 - setImageBitmap() method, 231
 - setImageDrawable() methods, 231
 - setImageResource() method, 231
 - setImageURI() method, 231-232
 - setProgress() method, 277
 - setSelection() method, 183
 - setText() method, 179
 - settings. *See also* formatting
 - applications, managing, 96
 - default, 363-364
 - games, adding, 216-217
 - power, 387
 - ringtones, 380
 - screens
 - avatars, 229-230
 - design, 171-181
 - dialogs, 196-201
 - Favorite Place feature, 250
 - features, 104
 - friend requests, 311-314
 - synchronizing, 294
 - uploading, 295-304
 - wallpaper, 380-381
 - setup() method, 164
 - shapes, 379
 - SharedPreferences
 - settings, adding, 216-217
 - saving data with, 184-186
 - sharing data, 381
 - content providers, 384-385
 - files and directories, 382-383
 - integrating global searches, 385
 - showDialog() method, 191, 201
 - shutting down activities, 56
 - signing
 - applications, 414-416
 - testing, 417-418
 - Simple API for XML. *See* SAX
 - simulating
 - incoming calls, 36-37
 - incoming SMS messages, 37
 - sizing
 - bitmaps, 241
 - interfaces, 71
 - SMS messages, incoming, 37
 - social features
 - adding, 309-310
 - friend support, 311-318
 - integrating, 319-320

software developers

software developers, 9

Software Development Kits.

See SDKs

source code, 106, 453-454

source control, integrating, 452

space, available, 438

specifying

language-specific resources,
346

region-specific resources,
347

speech, converting to text, 376

Spinner control, 182-183, 250

splash screens

customizing, 126-130

design, 117-118

features, 102

landscape mode layouts,
359

layouts, 118-119, 122, 125

resources, 120-122

SplashActivity, 49

Stack Overflow, 457

Standards, developing code,
392

startActivity() method, 53

starting QuizTask class, 289

state

activities, 54-55

settings, 216-217

status, networks, 275

storage

application resources, 66

question data, 217-219

strategies

internationalization,
349-351

screen orientations, 357

strings, 20

applying, 69

date/time, formatting,
351-352

game screens, adding
resources, 208

literals, declaring, 218

managing, 65

resources

adding, 108, 175-176

Favorite Place feature,
249

styles. See also customizing;
formatting

App Widgets, 328-329

interfaces, 373

subdirectories, 356

support

applications, 366

colors, 70

dimensions, 71

Facebook, 320

features, 364

friends, 311-318

images, 72

internationalization, 343

Android Market, 348

applications, 346

default resources,
346-347

localization utilities,
351-352

operating systems,
345-346

strategies, 349-351

LBS

accessing, 259-260

applying, 254-257

design, 245

favorites, 246-254

geocoding services,
260-263

maps, 263-266

MIME, 301

networks, accessing ser-
vices, 274-276

OpenSocial, adding, 320

operating systems, 437

player relationships, 310

Twitter, adding, 320

synchronizing applications, 294

systems

locales, determining, 351

resources, applying, 65-69

T

TabHost control

adding, 158

configuring, 163

tablets, 364

tabs

AndroidManifest.xml, 86-87

Application, 84

Instrumentation, 86

Manifest, 84

- Permission, 85
- screens, building with, 163-164
- target devices**
 - identifying, 395-396
 - testing, 398
- target platforms, selecting applications, 366**
- tasks**
 - applications, building, 47-50
 - asynchronous friend requests, 315-317
 - backgrounds, App Widgets, 331
 - managing, 34-35
 - running, 279-280
- temperatures, 386**
- templates, ListView controls, 139**
- testing**
 - applications
 - best practices, 391
 - developing coding standards, 392
 - implementing build processes, 393
 - maximizing coverage, 395-405
 - planning, 393-395
 - release candidates, 413
 - tracking bugs, 393
 - automating, 398-404
 - cameras, 236
 - emulators, 397
 - environments, managing, 395-396
 - locations, 255
 - networks
 - on emulators, 272
 - on hardware, 273
 - signed applications, 417-418
 - target devices, 398
 - test cases, creating, 400-403
- text**
 - Auto-complete, 446
 - EditText controls, 178, 199-201
 - input, 179
 - speech, converting, 376
- text-to-speech (TTS) engines, 373**
- TextSwitcher control, updating, 214**
- TextView control, 20, 173, 230**
- theft, protecting against, 412-413**
- themes, interfaces, 373**
- Thinking in Java, 12**
- third-party services, 265**
- threads, applying, 280**
- time, formatting, 351-352**
- tools, 12, 43, 437-444**
 - adb command-line, 418
 - code-signing, 13
 - development, 31, 113
 - DDMS, 33-39
 - documentation, 31-33
 - managing tasks, 34-35
 - navigating files, 35-36
 - Device Screen Capture, 38
 - emulators, applying, 39
 - Extract Local Variable, 449
 - LogCat logging, 26
 - monkey, 395
 - ProGuard, 412-413
 - Rename, 448
- tracing life cycles of activity dialogs, 191-192**
- tracking bugs, 393**
- translating addresses into coordinates, 262-263**
- triggering alerts, 373**
- trivia games, designing, 99-105**
- troubleshooting build errors, 450**
- TTS (text-to-speech) engines, 373**
- Twitter support, adding, 320**
- typefaces, 379**
- types**
 - of dialogs, 190-191
 - of graphics animation, 126
 - of menu mechanisms, 144-147
 - of testing, 394-395

U

underlying device hardware, 386-387

units of measurement, 71

updating

- App Widgets, 332-335
- applications, signing, 415

updating

- Favorite Place feature, 250
- game screen layouts, 210-211
- help screens, 154
- ImageSwitcher controls, 214-215
- layouts, 122, 125
 - friend requests, 312-313
 - main menu screens, 138-139
- LBS applications, 246
- locations, 260
- manifest files, 110
- scores screens, 161-162
- settings screens
 - avatars, 230
 - layouts, 177-178
- TextSwitcher controls, 214
- upgrade testing, 394**
- UploaderService class, 297**
- uploading**
 - applications, 293
 - Android Market, 423-427
 - settings, 295-304
 - synchronizing, 294
 - avatars, 301
 - scores, 304
- UploadTask class, implementing, 298**
- usability testing, 394**
- user interface design, 373. See also interfaces**
 - input methods, 374
 - styles and themes, 373

- user gestures, 375
- views, 374
- users**
 - alerts, 372-373
 - gestures, 375
- utilities, localization, 351-352.**
See also tools

V

- V CAST Apps, 431**
- validating passwords, 172**
- values, 69-73**
- verifying**
 - age, 172
 - signed applications, 418
- versions**
 - applications, 88
 - availability, 12
 - build processes, implementing, 393
 - code, 88
 - naming, 88
 - SDKs, 89, 365-367
- video, 378**
- VideoView control, 378**
- View controls, 119**
- ViewFactory class, 207**
- ViewGroup controls, 374**
- viewing**
 - files, 35-36
 - log information, 39
 - scores, 280-286
 - Scores of Friends tab, 318

- views**
 - animation, 128-129
 - interfaces, 374
 - RemoteViews interfaces, 327-328
- ViewSwitcher class, 206, 211-215**
- virtual devices, managing, 21-22**

W

- wallpaper, 380-381**
- websites**
 - accessing, 454-455
 - applications, selling, 429-430
 - resources, 457
- Wi-Fi, applying, 387**
- WidgetUpdateTask class, 333-334**
- Windows, 12**
 - Android SDK Starter Packages, installing, 440
 - Eclipse IDEs, installing, 439
 - operating systems, debugging, 444
- wireless carriers, 9**
- Wireless Developer Network, 457**
- wizards, Android Project Wizard, 13-20**

workflow documentation,
393-395

workspace panes, 451-452

X-Z

XML (Extensible Markup Language)

applying, 165-167

files, 70, 77

game screens, 209-210

layout design, 75

parsing, declaring string lit-
erals, 218

XmlResourceParser, 165-166