



jetzt lerne ich

# Android 4

## Programmierung



DIRK LOUIS / PETER MÜLLER

- Grafische Benutzeroberflächen erstellen
- Mehr als 50 fertige Beispiel-Apps entwickeln
- Mit der Eclipse-Entwicklungsumgebung arbeiten
- Alle Tools, Installationsanleitung und 300-seitiges Java-Tutorial auf DVD



# Inhaltsübersicht

<b>Vorwort</b> .....	<b>17</b>
<b>Teil A – Einführung</b> .....	<b>21</b>
<b>1 Der Rechner wird vorbereitet</b> .....	<b>23</b>
<b>2 Auf die Plätze, fertig ... App!</b> .....	<b>43</b>
<b>3 Was wann wofür</b> .....	<b>75</b>
<b>Teil B – Grundlagen</b> .....	<b>89</b>
<b>4 Code</b> .....	<b>91</b>
<b>5 Die Benutzeroberfläche (Layout)</b> .....	<b>115</b>
<b>6 Ressourcen</b> .....	<b>157</b>
<b>7 Mit dem Anwender interagieren</b> .....	<b>183</b>
<b>8 App-Grundlagen und Lebenszyklus</b> .....	<b>201</b>
<b>Teil C – Weiterführende Themen</b> .....	<b>213</b>
<b>9 In Views zeichnen</b> .....	<b>215</b>
<b>10 Menüs und Dialoge</b> .....	<b>231</b>
<b>11 Mehrseitige Apps</b> .....	<b>257</b>
<b>12 Daten speichern</b> .....	<b>269</b>
<b>13 Quiz-Apps</b> .....	<b>283</b>
<b>14 Multimedia</b> .....	<b>291</b>
<b>15 Sensoren</b> .....	<b>311</b>
<b>16 Einsatz der Datenbank SQLite</b> .....	<b>327</b>
<b>17 Geolokation</b> .....	<b>343</b>
<b>18 Brettspiel-Apps (TicTacToe)</b> .....	<b>353</b>
<b>19 Tablet-Unterstützung mit Fragmenten</b> .....	<b>365</b>
<b>20 Tipps und Tricks</b> .....	<b>379</b>

<b>Anhang A: Apps veröffentlichen oder weitergeben</b> .....	<b>399</b>
<b>Anhang B: Eclipse</b> .....	<b>409</b>
<b>Anhang C: Emulator, DDMS &amp; Debugger</b> .....	<b>421</b>
<b>Anhang D: Die DVD zum Buch</b> .....	<b>443</b>
<b>Anhang E: Lösungen</b> .....	<b>447</b>
<b>Stichwortverzeichnis</b> .....	<b>465</b>

# Inhaltsverzeichnis

<b>Vorwort</b> .....	17
<b>Teil A – Einführung</b> .....	21
<b>1 Der Rechner wird vorbereitet</b> .....	23
1.1 Die nötigen Hilfsmittel und Vorbereitungen .....	23
1.2 Das JDK für Java SE .....	24
1.2.1 Setup-Datei .....	25
1.2.2 Installation .....	25
1.2.3 Eintragung in den Systempfad .....	25
1.2.4 Test .....	26
1.3 Das Android-SDK .....	27
1.3.1 Setup-Datei .....	27
1.3.2 Installation .....	28
1.3.3 Dokumentation und API-Referenz .....	31
1.4 Eclipse .....	34
1.4.1 Setup-Datei .....	34
1.4.2 Installation .....	35
1.4.3 Erster Start .....	35
1.5 Das Android-Plugin .....	37
1.5.1 Installation .....	37
1.5.2 Konfiguration .....	38
1.6 Wo Sie weitere Hilfe finden .....	39
1.7 Nächste Schritte .....	40
1.8 Fragen und Antworten .....	40
1.9 Übungen .....	41
<b>2 Auf die Plätze, fertig ... App!</b> .....	43
2.1 Die Ruhe vor dem Sturm .....	43
2.2 Das Projekt .....	44
2.3 Das vorgegebene Codegerüst .....	54
2.3.1 Die package-Anweisung .....	55
2.3.2 Die import-Anweisungen .....	56
2.3.3 Die Klassendefinition .....	57

2.4	Layout und Ressourcen	59
2.4.1	XML-Layouts	59
2.4.2	Ressourcen	61
2.5	Die App erstellen (Build)	63
2.6	Die App im Emulator testen	65
2.6.1	AVD für Emulator anlegen	65
2.6.2	App testen	67
2.7	Die App auf dem Smartphone oder Tablet-PC testen	69
2.8	Nächste Schritte	73
2.9	Fragen und Antworten	73
2.10	Übungen	74
<b>3</b>	<b>Was wann wofür</b>	<b>75</b>
3.1	Was ist zu tun? – Die drei Pfeiler der App-Erstellung	75
3.2	Wer hilft uns? – Bausteine und Klassen	76
3.2.1	Bausteine für den App-Aufbau	76
3.2.2	Klassen zur Adressierung spezieller Aufgaben	80
3.3	Wo wird was gespeichert? – Dateitypen, die Sie kennen sollten	81
3.3.1	Quelldateien	81
3.3.2	Automatisch generierte Dateien	82
3.3.3	Die Android-Bibliothek	83
3.3.4	assets	84
3.3.5	Die Ressourcendateien	84
3.3.6	Die Manifestdatei	84
3.3.7	Die Properties-Datei	86
3.3.8	Die APK-Datei	87
3.4	Fragen und Antworten	87
3.5	Übungen	88
<b>Teil B – Grundlagen</b>		<b>89</b>
<b>4</b>	<b>Code</b>	<b>91</b>
4.1	Unterstützung durch den Eclipse-Editor	91
4.1.1	Syntaxhervorhebung	92
4.1.2	Gliederung (Folding)	92
4.1.3	QuickFix	94
4.1.4	QuickInfo statt API-Dokumentation	98
4.1.5	Klammernpaare identifizieren	101
4.1.6	Zeilennummern einblenden	102
4.1.7	Alle Vorkommen markieren	102
4.1.8	Definitionen finden	103
4.1.9	Code erweitern	103
4.1.10	Refactoring (Code umstrukturieren)	106

4.2	Klassen in eigene Quelldateien auslagern	107
4.2.1	Die Tuschstaffel-App	108
4.2.2	Quelldateien hinzufügen	111
4.3	Fragen und Antworten	114
4.4	Übungen	114
<b>5</b>	<b>Die Benutzeroberfläche (Layout)</b>	<b>115</b>
5.1	Ein paar einführende Gedanken zum Design von Benutzeroberflächen	115
5.2	Die zwei Gesichter der Layoutdateien: XML kontra Designer	118
5.2.1	Der XML-Code	119
5.2.2	Der Designer	124
5.3	Layout-Views	127
5.3.1	Die allgemeinen Layoutparameter	128
5.3.2	Die Layout-Views	131
5.3.3	Hintergrundfarbe (oder -bild)	139
5.3.4	Hierarchy Viewer	142
5.4	Widgets	143
5.5	Praxisbeispiel: eine Quiz-Oberfläche	147
5.6	Hoch- und Querformat	151
5.7	App-Symbol	153
5.8	Views im Code verwenden	153
5.8.1	Layouts laden	153
5.8.2	Zugriff auf UI-Elemente	154
5.9	Fragen und Antworten	156
5.10	Übungen	156
<b>6</b>	<b>Ressourcen</b>	<b>157</b>
6.1	Der grundlegende Umgang	157
6.1.1	Ressourcen anlegen	158
6.1.2	Ressourcen verwenden	161
6.1.3	Ressourcen aus dem Projekt entfernen	164
6.2	Welche Arten von Ressourcen gibt es?	165
6.2.1	Größenangaben	165
6.2.2	Farben	166
6.2.3	Strings	167
6.2.4	String-Arrays (Texte)	169
6.2.5	Bilder	171
6.2.6	Layouts	172
6.2.7	Menüs	173
6.2.8	Roh- und Multimediadaten	173
6.2.9	Stile	174

6.3	Alternative Ressourcen vorsehen	178
6.3.1	Das Grundprinzip	179
6.3.2	Wie stellt man konfigurationsspezifische Ressourcen bereit?	180
6.4	Fragen und Antworten	182
6.5	Übungen	182
<b>7</b>	<b>Mit dem Anwender interagieren</b>	<b>183</b>
7.1	Das Grundprinzip	183
7.1.1	Auf ein Ereignis reagieren	184
7.1.2	Welche Ereignisse gibt es?	187
7.1.3	Hintergrund der Ereignisverarbeitung	188
7.2	Vereinfachte Ereignisbehandlung	190
7.2.1	Ereignisbehandlung mit anonymen Listener-Klassen	190
7.2.2	Ereignisbehandlung mit anonymen Listener-Objekten	191
7.2.3	Ereignisbehandlung mithilfe der Activity-Klasse	192
7.3	Eine Behandlungsmethode für mehrere Views	193
7.4	Auf Tipp- und Wischereignisse reagieren	194
7.4.1	Tippereignisse	194
7.4.2	Wischereignisse	196
7.5	Auf Tastendrucke reagieren	197
7.6	Ereignisverarbeitung in selbst geschriebenen View-Klassen	199
7.7	Fragen und Antworten	200
7.8	Übungen	200
<b>8</b>	<b>App-Grundlagen und Lebenszyklus</b>	<b>201</b>
8.1	Die Android-Architektur	201
8.2	Der App-Lebenszyklus	203
8.3	Der Activity-Lebenszyklus	205
8.4	Lebenszyklus-Demo	206
8.5	Fragen und Antworten	211
8.6	Übungen	212
<b>Teil C – Weiterführende Themen</b>		<b>213</b>
<b>9</b>	<b>In Views zeichnen</b>	<b>215</b>
9.1	Das Grundprinzip	215
9.1.1	Die Leinwand	215
9.1.2	Das Atelier	215
9.1.3	Die Zeichenmethoden und -werkzeuge	216
9.1.4	Wie alles zusammenwirkt	216

<b>12</b>	<b>Daten speichern</b>	269
12.1	Preferences	269
12.2	Dateizugriffe	270
12.2.1	In Dateien schreiben	270
12.2.2	Aus Dateien lesen	271
12.2.3	Textdateien	272
12.2.4	Welche Dateien sind vorhanden?	273
12.2.5	Dateien als Ressourcen verwalten	274
12.3	Zugriff auf die SD-Karte	274
12.4	Die Reaktions-App	276
12.5	Fragen und Antworten	281
12.6	Übungen	281
<b>13</b>	<b>Quiz-Apps</b>	283
13.1	Aufbau und Benutzeroberfläche	283
13.2	Die Activity (QuizActivity.java)	284
13.3	Die Fragen (Frage.java)	286
13.4	Die Spielsteuerung (Spiellogik.java)	287
13.5	Verbesserungen	289
13.6	Fragen und Antworten	290
13.7	Übungen	290
<b>14</b>	<b>Multimedia</b>	291
14.1	Audioressourcen	291
14.2	Sound-Effekte mit SoundPool	292
14.3	Das Universalgenie: MediaPlayer	294
14.3.1	Audioressourcen abspielen	294
14.3.2	Audiodateien vom Dateisystem abspielen	295
14.3.3	Audiodateien aus dem Internet abspielen	295
14.3.4	Auf das Abspielende reagieren	296
14.3.5	MediaPlayer-Objekte wiederverwenden	297
14.3.6	Ressourcen freigeben	299
14.3.7	Audiodateien wiederholt abspielen	300
14.4	Piepen und andere Töne	300
14.5	Bilddateien anzeigen	302
14.6	Videos abspielen	303
14.7	Videos aufnehmen	305
14.8	Fotos aufnehmen und speichern	305
14.9	Fragen und Antworten	310
14.10	Übungen	310



9.2	Grafikprimitive zeichnen	220
9.3	Bilder bewegen	224
9.4	Verbesserungen	229
9.5	Fragen und Antworten	229
9.6	Übungen	230
<b>10</b>	<b>Menüs und Dialoge</b>	<b>231</b>
10.1	Menüs	231
10.1.1	Menü-Verwirrungen	232
10.1.2	Menü-Ressourcen	233
10.1.3	Menüeinträge in der ActionBar	235
10.1.4	Das Optionen-Menü	236
10.1.5	Das Kontextmenü	238
10.1.6	Popup-Menü	240
10.1.7	Untermenüs	241
10.1.8	Auf die Auswahl eines Menüeintrags reagieren	241
10.2	Dialoge	244
10.2.1	Dialoge erzeugen	244
10.2.2	Dialoge anzeigen	245
10.2.3	Standarddialoge mit AlertDialog	246
10.2.4	Dialoge für Datum- und Zeitauswahl	247
10.2.5	Der Fortschrittsdialog	250
10.2.6	Eigene Dialoge definieren	252
10.3	Benachrichtigungen mit Toasts	254
10.3.1	Toasts im Hintergrund-Thread	254
10.4	Fragen und Antworten	255
10.5	Übungen	256
<b>11</b>	<b>Mehrseitige Apps</b>	<b>257</b>
11.1	Intents	257
11.1.1	Was sind Intents?	257
11.1.2	Explizite und implizite Intents	259
11.1.3	Intent-Filter	259
11.2	Activities starten mit Intents	260
11.2.1	Intent-Objekte erzeugen	261
11.3	Intents empfangen	262
11.4	Ein Demo-Beispiel	263
11.5	Ergebnisse zurücksenden	266
11.6	Fragen und Antworten	267
11.7	Übungen	267

<b>15 Sensoren</b>	311
15.1 Zugriff	311
15.1.1 Was Sie benötigen	312
15.1.2 Welche Sensoren sind verfügbar?	312
15.1.3 Anmeldung beim Sensor	313
15.2 Sensordaten auslesen	315
15.2.1 Beschleunigungswerte ermitteln	317
15.2.2 Lagedaten ermitteln	320
15.3 Fragen und Antworten	324
15.4 Übungen	325
<b>16 Einsatz der Datenbank SQLite</b>	327
16.1 Was ist eine relationale Datenbank?	327
16.2 Datenbank anlegen/öffnen	328
16.2.1 onCreate()	329
16.2.2 onUpgrade()	331
16.2.3 close()	331
16.2.4 Datenbanken als Ressourcen mitgeben	331
16.3 Datenzugriffe	332
16.4 Datenbankinhalte mit ListView anzeigen	337
16.5 Fragen und Antworten	341
16.6 Übungen	342
<b>17 Geolokation</b>	343
17.1 Zugriff	343
17.1.1 Verfügbarkeit feststellen	343
17.1.2 Daten empfangen	344
17.1.3 Empfänger abmelden	345
17.2 Geokoordinaten	346
17.2.1 Sexagesimale und dezimale Darstellung	346
17.2.2 Das Location-Objekt	346
17.3 Die Demo-App	347
17.4 Fragen und Antworten	352
17.5 Übungen	352
<b>18 Brettspiel-Apps (TicTacToe)</b>	353
18.1 Aufbau und Benutzeroberfläche	353
18.2 Die Start-Activity (TicTacToeActivity)	355
18.3 Spielfeld und Logik (TicTacToeView)	356
18.3.1 Vorbereitungen	356
18.3.2 Spielfeld zeichnen	358

18.3.3	Spielerzug durchführen	359
18.3.4	Computerzug mit AsyncTask durchführen	360
18.4	Verbesserungen	363
18.5	Fragen und Antworten	364
18.6	Übungen	364
<b>19</b>	<b>Tablet-Unterstützung mit Fragmenten</b>	<b>365</b>
19.1	Was ist ein Fragment?	365
19.2	Ein Fragment erzeugen	366
19.3	Fragment zur Activity hinzufügen	366
19.4	Ein Fragment-Beispiel	368
19.4.1	Das Layout der Activity	369
19.4.2	Definition der Fragment-Klassen	370
19.4.3	Die Activity	372
19.5	Fragmente für Dialoge	375
19.6	Fragen und Antworten	376
19.7	Übungen	377
<b>20</b>	<b>Tipps und Tricks</b>	<b>379</b>
20.1	Mehrere AVDs und Emulator-Konfigurationen einrichten	379
20.2	Das Smartphone vibrieren lassen	382
20.3	UI-Code periodisch ausführen lassen	383
20.4	Bildergalerien mit GridView und BaseAdapter	386
20.4.1	Die Bildressourcen	386
20.4.2	Die Adapter-Klasse	387
20.4.3	Die GridView	390
20.4.4	Angeklickte Bilder als Vollbild anzeigen	390
20.5	Spinner verwenden (Listenfelder)	393
20.5.1	Den Spinner mit Daten füllen	393
20.5.2	Ereignisbehandlung	395
20.6	Mehrsprachige Apps	396
20.7	Fragen und Antworten	398
20.8	Übungen	398
<b>Anhang A:</b>	<b>Apps veröffentlichen oder weitergeben</b>	<b>399</b>
A.1	Die App vorbereiten	399
A.2	Digitales Signieren	400
A.3	Die App exportieren und signieren	401
A.4	Bei Google Play registrieren	404
A.4.1	Steuerliche Aspekte bei App-Verkauf	405
A.5	App hochladen	405
A.6	Weitergabe an Bekannte	406

<b>Anhang B: Eclipse</b> .....	409
B.1 Android-Projekt anlegen .....	409
B.2 Projekte erstellen .....	411
B.3 Projekte deaktivieren .....	412
B.4 Projekte löschen .....	412
B.5 Neuen Workspace einrichten .....	413
B.6 Bestehendes Projekt in Workspace aufnehmen (Import) .....	415
B.7 Launch-Konfigurationen anpassen oder einrichten .....	415
B.8 Properties-Fenster anzeigen .....	416
B.9 Formatierung von XML-Layoutdateien .....	417
B.10 Apps exportieren .....	418
B.11 Kleines Eclipse-Wörterbuch .....	419
<b>Anhang C: Emulator, DDMS &amp; Debugger</b> .....	421
C.1 Der Emulator .....	421
C.1.1 Emulator starten .....	423
C.1.2 Die Emulator-Bedienung .....	428
C.1.3 Apps installieren und deinstallieren .....	428
C.2 Das DDMS .....	429
C.3 Der Debugger .....	434
C.3.1 Debug-Lauf starten .....	434
C.3.2 Debug-Möglichkeiten .....	436
C.4 Debugging-Beispiel .....	439
<b>Anhang D: Die DVD zum Buch</b> .....	443
<b>Anhang E: Lösungen</b> .....	447
<b>Stichwortverzeichnis</b> .....	465

## 3 Was wann wofür

- SIE LERNEN IN DIESEM KAPITEL,
- WIE APPS AUFGEBAUT SIND,
  - WELCHE BIBLIOTHEKS-KLASSEN FÜR DIE APP-PROGRAMMIERUNG WICHTIG SIND UND
  - ERFAHREN, WOFÜR MANIFEST-, JAR-, APK- UND ANDERE DATEIEN BENÖTIGT WERDEN.

Nachdem die ersten Schritte getan sind, die erste App erstellt und hoffentlich auch erfolgreich im Emulator getestet wurde, wir also die ersten Hürden erfolgreich genommen haben – und dies gilt ganz besonders für diejenigen Leser, die nebenbei auch noch das Java-Tutorium auf der Buch-DVD durchgearbeitet haben –, werden wir in diesem Kapitel eine kurze Zwischenpause einlegen, die Entwicklerwerkzeuge und den Compiler für eine Weile ruhen lassen und diese Unterbrechung dazu nutzen, uns geistig auf die nächsten Aufgaben vorzubereiten.

Konkret werden wir uns eine Übersicht darüber verschaffen, was bei der App-Programmierung eigentlich von uns erwartet wird, mit welchen Komponenten (Android-Klassen) wir es dabei zu tun haben und wie Android-Projekte im Detail aufgebaut sind. Wir werden dabei etlichem Bekannten begegnen, aber auch viel Neues entdecken.

### 3.1 Was ist zu tun? – Die drei Pfeiler der App-Erstellung

Zur App-Programmierung gehört, dass Sie

- den **Code schreiben**, der festlegt, wie sich die App verhält,
- das **Layout festlegen**, das bestimmt, wie die App auf dem Anzeigegerät dargestellt wird,
- die **Ressourcen bereitstellen**, die für die Anzeige und Funktion der App benötigt werden.

Haben Sie die Aufgabenkomplexe wiedererkannt? Es sind die gleichen Aufgaben, die wir bereits in Kapitel 2.3 und 2.4 angesprochen haben.

Der Code ist naturgemäß das eigentliche, ureigene Metier des Programmierers. Wie das Codegerüst einer App aussieht, haben Sie ja bereits in Kapitel 2.3 gesehen. In Kapitel 4 werden wir uns etwas näher mit dem Code befassen, ein paar Fingerübungen machen und uns vor allem ansehen, wie uns Eclipse bei der Codebearbeitung unterstützt.

App-Benutzeroberflächen werden üblicherweise über XML-Layoutdateien definiert (siehe Kapitel 2.4). Da sie für den Erfolg und die Bedienbarkeit einer App von großer Bedeutung sind, werden wir uns in Kapitel 5 etwas ausführlicher mit ihnen beschäftigen. Wir werden uns in den XML-Code einarbeiten und uns ansehen, wie uns der Eclipse-Designer bei der visuellen Bearbeitung der Layouts unterstützt, und uns nebenbei mit Hintergrundbildern, Orientierungen und App-Symbolen befassen.

Apps arbeiten viel mit Ressourcen: Bilder, anzuzeigende Texte und Beschriftungen, Farben, Stile, Mediendateien etc. In Kapitel 6 werden wir die verschiedenen Ressourcentypen vorstellen. Vor allem die Arbeit mit Strings und Bildern, inklusive der Unterstützung unterschiedlicher Geräteauflösungen, werden wir etwas eingehender betrachten.

## 3.2 Wer hilft uns? – Bausteine und Klassen

Android-Apps werden in der Umgebung eines Android-Betriebssystems ausgeführt. Das Betriebssystem stellt, unterstützt von passender Hardware, den Apps viele interessante Optionen zur Verfügung (wie z.B. das Abspielen von Sounddateien, das Aufnehmen von Fotos, das Starten fremder App-Komponenten, das Speichern von Dateien etc.), stellt umgekehrt aber auch Anforderungen an die Apps (vorgegebener Aufbau, vorgegebene Kommunikationswege, DEX-Bytecode etc.).

Um dem Programmierer die Arbeit zu erleichtern, sodass er mit möglichst geringem Aufwand Android-konforme Apps erstellen und die vielen technischen Möglichkeiten nutzen kann, stellt uns Google die Klassen der Android-Bibliothek zur Verfügung.

### 3.2.1 Bausteine für den App-Aufbau

Apps bestehen aus diversen Bausteinen, hinter denen naturgemäß Klassen aus der Android-Bibliothek stehen. Sehen wir uns einige dieser Bausteine etwas genauer an.

#### Activities

Während Windows-Anwendungen üblicherweise **ein** Hauptfenster besitzen, in dem der Anwender **eine Vielzahl** von Aktionen durchführen kann, bestehen Apps aus **einer oder mehreren** Bildschirmseiten, die jede **einer** bestimmten Aktivität gewidmet sind. Womit wir beim Thema Aktivitäten, oder, wie der Android-Programmierer auch sagt, »Activities« wären.

Eine Activity ist eine Kombination aus Bildschirmseite und zugehörigem Code. Eine Activity sollte einer in sich abgeschlossenen Aufgabe (Aktivität) gewidmet sein, sie wird als Klasse implementiert, die von der Bibliotheksklasse `android.app.Activity` abzuleiten ist, und sie muss in der Manifestdatei der App aufgeführt werden.

Eine Besonderheit der App-Activities ist, dass sie in sich geschlossene Bausteine darstellen und nur lose an ihre App gebunden sind. Dies hat zwei Konsequenzen:

- Eine Activity kann grundsätzlich von jeder App auf dem Android-Gerät aufgerufen werden.
- Für den Aufruf von Activities gibt es einen globalen Aufrufmechanismus: der Aufruf über Intents. Diesen Mechanismus müssen Sie verwenden, gleichgültig, ob Sie eine Activity der eigenen oder einer fremden App aufrufen möchten.

#### Intents

Betrachtet man Apps als lose verbundene Activities, so sind es die Intents, zu Deutsch »Absichten«, die die lose Verbindung zwischen den Activities herstellen.

### Wiederverwendbare Komponenten

Ist es nicht seltsam, eine Anwendung als eine Sammlung eigenständiger binärer Software-Komponenten zu definieren? Ganz und gar nicht! Microsoft betreibt seit Jahren einen enormen Aufwand, um über diverse Technologien (COM, DCOM, COM+, .NET Framework) seiner Windows-Entwicklergemeinschaft das zu bieten, was Android von vornherein mitbringt: die einfache anwendungsübergreifende Wiederverwendung von auf dem System installierten Software-Bausteinen.

Konkret bedeutet dies: Wenn Sie aus einer Activity heraus eine andere Activity starten möchten, müssen Sie

- einen Intent erzeugen, der angibt, welche Activity aufzurufen ist und welche Daten dieser Activity gegebenenfalls übergeben werden sollen,
- den Intent mit einer passenden Android-Methode abschicken.

Das Android-System empfängt den Intent und sucht nach der auszuführenden Activity. Gibt es auf dem System eine Activity, die zur Beschreibung in dem Intent passt, wird die Activity gestartet.

Tauchen wir noch ein wenig tiefer in den Intent-Mechanismus ein. Grundsätzlich gibt es zwei Wege, einen Intent zu adressieren:

- als expliziten Intent – in diesem Fall wird als Adressat die Activity angegeben, die aufgerufen werden soll.
- als impliziten Intent – in diesem Fall wird als Adressat keine konkrete Activity angegeben. Stattdessen werden bestimmte Informationen über die gewünschte Aktion mitgeliefert (*action*, *type* und *category*), und das Android-System bestimmt, welche der auf dem System vorhandenen Activities zu den Informationen passt. (Zur Unterstützung dieses Mechanismus definieren die Activities in der Manifestdatei sogenannte Intent-Filter, deren Daten mit den Informationen im Intent-Objekt abgeglichen werden, siehe auch weiter unten die Ausführungen zur Manifestdatei.)

**Aufsteiger** Activities sind nicht die einzigen Komponenten, die über Intents aufgerufen werden können. Auch Services und Broadcast Receiver sind ausführbare binäre Komponenten, die über Intents gestartet werden.

### Broadcast Intents

Nicht nur Apps können Intents abschicken. Auch das Android-System selbst kann Intents versenden, die von interessierten Apps über Broadcast Receiver abgefangen werden können. Wir sprechen in diesem Fall von Broadcast Intents.

### Views

Kommen wir noch einmal auf die grafischen Benutzeroberflächen der Apps zurück. Diese sind, wie Sie wissen, auf Bildschirmseiten verteilt und werden üblicherweise über XML-Layoutdateien definiert (siehe auch Kapitel 2.4).

Aufgebaut werden diese Bildschirmseiten aus Views. Eine View, zu Deutsch »Ansicht«, ist einfach ein rechteckiges Element einer UI-Oberfläche, das sich selbst zeichnet und grundsätzlich mit dem Anwender interagieren kann.

Drei Arten von Views sind für uns besonders interessant:

- Zeichenflächen – Instanzen der Klassen `View`, `ImageView` oder `SurfaceView`, die rechteckige Bereiche auf einer Bildschirmseite repräsentieren, in die wir zeichnen können.
- Widgets – spezialisierte Views, die einer bestimmten Aufgabe gewidmet sind. So sind z.B. die typischen Steuerelemente wie Eingabefelder, Schaltflächen, Listenfelder etc. als Widgets im Paket `android.widget` definiert.
- Viewgroups – Container-Views, die andere Views in sich aufnehmen können. Viewgroups, die andere Views nicht nur aufnehmen, sondern auch noch nach bestimmten Regeln anordnen, bezeichnen wir als Layout-Views (siehe Kapitel 5.3).

### Sonstige Bausteine

Es gibt noch eine Reihe weiterer App-Bausteine, die in diesem Buch zwar keine besondere Rolle spielen, von denen Sie aber dennoch schon einmal gehört haben sollten.

- Services

Ein Service, zu Deutsch »Dienst«, ist eine Arbeit, die im Hintergrund erledigt wird. Im Gegensatz zu Activities besitzen Services daher keine eigene Benutzeroberfläche. Mithilfe von Services kann man Aufgaben parallel zur laufenden App ausführen (beispielsweise eine Hintergrundmusik abspielen) oder langwierige Aktionen, wie z.B. das Herunterladen großer Multimediadateien aus dem Internet, im Hintergrund erledigen, ohne dass die App dadurch lahmgelegt wird. Services werden als Unterklassen der Klasse `Service` implementiert.

- Broadcast Receiver

Broadcast Receiver sind Komponenten, die auf Meldungen des Android-Systems reagieren, wie z.B. »niedriger Batteriestand« (`ACTION_BATTERY_LOW`) oder »Der Kameraknopf wurde gedrückt« (`ACTION_CAMERA_BUTTON`). Broadcast Receiver besitzen keine eigene Benutzeroberfläche, können aber Nachrichten und Optionen zur Reaktion auf das Ereignis in die System-Statusleiste ausgeben.

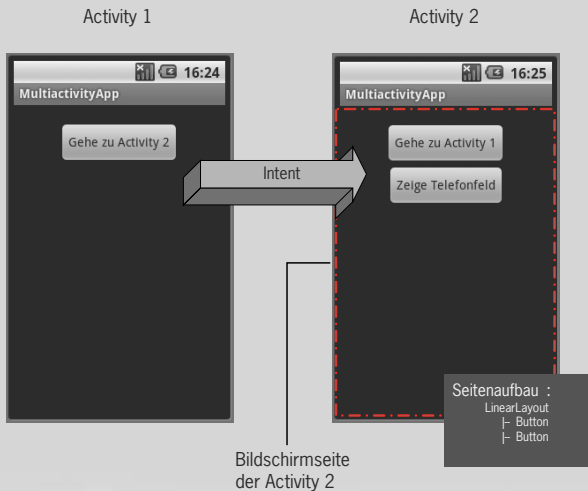
- Content Provider

Content Provider sind Komponenten, die Ihnen bei der Verwaltung externer Daten helfen. Sie können sie für Daten benutzen, die nur von einer App verwendet werden. Sie können über einen Content Provider aber auch Daten verwalten, die von mehreren oder allen Apps auf einem Android-Gerät genutzt werden können.



### Das Activity-View-Intent-Geflecht

Eine Android-App besteht aus einer oder mehreren Activities. Jede dieser Activities steht für einen in sich abgeschlossenen Aufgabenbereich inklusive zugehöriger Bildschirmseite. Apps können aus prinzipiell beliebig vielen Activities (Bildschirmseiten) bestehen. Der Wechsel von einer Activity zur anderen erfolgt stets durch Absendung eines Intents.



**Abbildung 3.1:** Drückt der Anwender den Button auf der 1. Bildschirmseite, wird ein Intent abgesetzt, der die Bildschirmseite der 2. Activity aufruft. Will der Anwender zurück zur 1. Seite, muss er auf den zugehörigen Button drücken, der natürlich ebenfalls einen Intent absetzt (hier nicht dargestellt). Die gestrichelte Linie markiert den Bereich, den die Bildschirmseite umfasst.

- Fragmente

Mit Android 3.0 (API-Level 11) eingeführte Komponente. Fragmente erlauben dem Programmierer, den Code umfangreicher Activities aufzuteilen, indem er Teile der Activity in Fragmenten auslagert. Fragmente definieren ihre eigene Benutzeroberfläche und haben einen eigenen Lebenszyklus. Ihr wohl größter Vorzug: auf Fragmente basierende grafische Benutzeroberflächen können so aufgebaut werden, dass funktionell zusammen gehörende Fragmente auf einem Tablet-PC nebeneinander, auf einem Smartphone aber nacheinander angezeigt werden. Eine so konzipierte App bietet auf Tablet-PCs einen größeren Bedienungskomfort, ist aber auch auf Smartphones ausführbar.

### App-Komponenten

Activities, Services, Content Provider, Broadcast Receiver und Fragmente werden in der Android-Terminologie als »Android-Komponenten« bezeichnet – essentielle Bausteine, die Aufgaben definieren, die vom Anwender oder vom System gestartet werden können.

# Stichwortverzeichnis

## Symbole

@Override 58

## A

AbsoluteLayout 139

ActionBar 232, 235

Action-Item 236

Action-Menü 232

Activities 47, 50, 57, 76

– beenden 266

– Ergebnisse zurücksenden 266, 276

– Lebenszyklus 205

– Manifestdatei 265

– on-Ereignismethoden überschreiben 199

– Start-Activity 86

– starten 260

– Titel 448

Activity

– fileList() 273

– findViewById() 155

– finish() 208, 266

– getFilesDir() 272

– getIntent() 262

– getResources() 164

– getSystemService() 312

– onContextItemSelected() 242

– onCreate() 58

– onCreateContextMenu() 238

– onCreateDialog() 244

– onCreateOptionsMenu() 236

– onOptionsItemSelected() 242

– onPause() 306

– onPrepareDialog() 245

– onResume() 306

– openFileInput() 271

– openFileOutput() 270

– registerForContextMenu() 239

– setContentView() 58, 60, 153

– showDialog() 245

– startActivity() 262

– startActivityForResult() 276

Activity-Menü 231

Adapter 137

– ArrayAdapter 394

– BaseAdapter 387

– Bilddaten 371

– SimpleCursorAdapter 337

AdapterContextMenuInfo 242

adb 428

addView() (ViewGroup) 219

AlertDialog 246

android

– alpha 122

– background 122, 139

– checkedButton (RadioGroup) 145

– checked (CheckBox) 144

– checked (RadioButton) 145

– checked (Switch) 146

– checked (ToggleButton) 146

– columnCount (GridLayout) 136

– contentDescription 117

– contentDescription (ImageButton) 144

– contentDescription (ImageView) 145

– focusable 117

– gravity (LinearLayout) 131

– id 122

– inputType (EditText) 144

– layout\_above (RelativeLayout) 134

– layout\_align... (RelativeLayout) 134

– layout\_below (RelativeLayout) 134

– layout\_center... (RelativeLayout) 134

– layout\_columnWidth (GridView) 138

– layout\_gravity (GridView) 138

– layout\_gravity (LinearLayout) 132

– layout\_height 129

– layout\_horizontalSpacing (GridView) 138

– layout\_marginBottom 130

– layout\_marginLeft 130

– layout\_marginRight 130

– layout\_marginTop 130

– layout\_numColumns (GridView) 138

– layout\_stretchMode (GridView) 138

– layout\_toLeftOf (RelativeLayout) 134

– layout\_toRightOf (RelativeLayout) 134

– layout\_verticalSpacing (GridView) 138

– layout\_weight (LinearLayout) 132

– layout\_width 129

– max (ProgressBar) 145

– minLines (EditText) 144

– onItemSelected (Spinner) 146

– orientation (LinearLayout) 131

– orientation (RadioGroup) 145

– padding 122

– password (EditText) 144

– progress (ProgressBar) 145

– prompt (Spinner) 146

– rotationX 122

– rowCount (GridLayout) 136

– scaleType (ImageView) 145

– src (ImageButton) 144

– src (ImageView) 145

– style (ProgressBar) 145

- text (Button) 144
- text (CheckBox) 144
- text (EditText) 144
- textOff (Switch) 146
- textOff (ToggleButton) 146
- textOn (Switch) 146
- textOn (ToggleButton) 146
- text (RadioButton) 145
- textSize (TextView) 146
- textStyle (TextView) 146
- text (Switch) 146
- text (TextView) 146
- typeface (TextView) 146
- visibility 122
- Android
  - ausführliche Installationsbeschreibung 23
  - Google Play 399, 430
  - Hilfsmittel 23
  - Market Place 399
  - mit Java 7 verwenden 73
  - Plattformen 30
  - Referenz der API 32
  - SDK 27
  - Versionsnummern 41
- Android-Architektur 201
- Android-Bibliothek 80, 83
  - API-Dokumentation 96
  - API-Version einer Klasse 73
  - Info in Editor 98
  - nachträglich ändern 74
- android.permission.ACCESS\_COARSE\_LOCATION 345
- android.permission.ACCESS\_FINE\_LOCATION 345
- android.permission.CAMERA 305
- android.permission.INTERNET 100, 295
- android.permission.READ\_EXTERNAL\_STORAGE 274
- android.permission.VIBRATE 382
- android.permission.WRITE\_EXTERNAL\_STORAGE 274, 305
- Android-Plugin 37
- Android-SDK
  - Dokumentation 32
  - Installation 28
  - Komponenten herunterladen 28
  - Unterverzeichnisse 31
- API 30
  - Bezug zu Android-Version 41
  - Dokumentation 96
  - einer Klasse 73
  - Referenz 32
- APK-Datei 87, 203
- »Application Not Responding«-Meldung 80
- Apps
  - Activities 47, 50, 57, 76
  - Android-Bibliothek 56, 83
  - an Gerätekonfiguration anpassen 178
  - Anwendungsname 46
  - APK-Datei 87
  - Application Not Responding-Meldung 80
  - beenden (finish()) 208
  - beenden (Zurück-Taste) 69
  - Benutzeroberfläche 115
  - Bildschirmseiten 76
  - deinstallieren 428
  - Ereignisse 183
  - erstellen (Build) 63
  - exportieren 401, 418
  - Galerien 137
  - Grundgerüst 54
  - Hoch- und Querformat 151
  - Intents 76
  - Komponenten 79
  - Layout 59
  - Layoutdatei 60
  - Manifestdatei 84
  - mehrsprachige 396
  - Min-SDK 84
  - Paket 46, 56, 81
  - Präferenzen 269
  - Projekt anlegen 44
  - Projektname 46
  - Properties-Datei 86
  - Ressourcen 61, 157
  - Ressourcendateien 84, 158
  - R.java 82, 160, 163
  - R-Klasse 60
  - Screenshots für die Veröffentlichung 430
  - SDK-Version 46, 50
  - signieren 401
  - Startsymbol 153
  - Strings 61
  - strings.xml 61
  - Target-SDK 46, 84
  - testen, auf Smartphone 69
  - testen, im Emulator 65
  - Views 78
  - weitergeben 399
  - zeitraubende Operationen 80
  - Zugriff auf Dateisystem 270
  - Zugriff auf SD-Karte 274
- AppTheme 177
- Arbeitsthread 251
- ArrayAdapter 394
- AsyncTask 360
  - doInBackground() 362
  - execute() 361
- Attribute 121. *Siehe* android
  - allgemeine 122
  - Layoutparameter 128
  - Namespace 121
  - style 174
- Audio
  - Formate 294
  - MediaPlayer 294
  - Ressourcen 291
  - SoundPool 292
  - Töne abspielen 300
- AudioTrack 301
- Außenabstand (Margin) 130

- AVD 422
  - einrichten 65, 379
  - mehrere 379
- B**
- Back-Stack 203
  - Fragments 367
- Barrierefreiheit 161
- BaseAdapter 387
  - getCount() 388
  - getItem() 388
  - getView() 387
- Beispiele
  - auf der Buch-DVD 443
  - Bildergalerie 386
  - Geolokation 343
  - Quiz-App 283
  - Reaktions-App 276
  - Sensoren 311
  - TicTacToe-App 353
  - UFO-App 224
- Benutzeroberflächen
  - Design 115
  - erleichterte Bedienbarkeit 117
  - Hoch- und Querformat 151
  - Layout-Views 127
  - Widgets (Steuerelemente) 143
- Berechtigungen
  - android.permission.ACCESS\_COARSE\_LOCATION 345
  - android.permission.ACCESS\_FINE\_LOCATION 345
  - android.permission.CAMERA 305
  - android.permission.INTERNET 100, 295
  - android.permission.VIBRATE 382
  - android.permission.WRITE\_EXTERNAL\_STORAGE 274, 305
- Beschleunigungssensor 317
- Bibliotheken 84
- Bilder 171, 302
  - App-Symbol 153
  - Bildergalerien 386
  - Formate 172
  - Größe 172
  - Hintergrundbilder 140
  - per Code laden 302
  - Thumbnails 387
  - zeichnen 224
- Bildergalerien 386
- Bildschirmdichten 153
- Bildschirmseiten 76
  - Design 115
  - Hierarchie 126
  - Hoch- und Querformat 151
  - im Designer 124
  - Layout tauschen 120
  - Layout-Views 127
  - View-Elemente 78
  - View-Hierarchie 120
  - Widgets (Steuerelemente) 143
- Wurzelement 120
- XML-Code 119
- Bitmap 302
- BitmapFactory 302
  - decodeResource() 302
- Broadcast 267
- Broadcast Intents 77, 267
- Broadcast Receiver 78
- Buch-DVD 24
  - Beispiele 443
  - Eclipse 34
  - Java-Tutorium 443
  - JDK für Java SE 25
  - XML-Einführung 443
- Buch-Website 39
- Build (Erstellen) 63
- Build-SDK
  - Bedeutung 74
  - eines Projekts 46
  - nachträglich ändern 73
- Bundle 261, 262
- Button 144
  - onClick 144
  - text 144
- C**
- Calendar 248
- Callback 247
- Camera 305
- Canvas 215
  - drawBitmap() 221, 227
  - drawCircle() 222
  - drawColor() 221
  - drawLine() 222
  - drawLines() 222
  - drawOval() 222
  - drawPoints() 222
  - drawRect() 222
  - drawRGB() 221
  - drawRoundRect() 222
  - drawText() 222
  - fill...() 222
  - translate() 222
- CheckBox 144
  - checked 144
  - isChecked() 144
  - text 144
- Class-Literal 262
- close() (SQLiteDatabase) 331
- Color 223
- Console-Fenster 68
- Content Provider 78, 341
- ContentValues 333
- convert() (Location) 347
- create() (MediaPlayer) 294
- Cursor 334
  - getCount() 334
  - getInt() 335
  - getString() 335
  - moveToFirst() 335

### D

- Dalvik Virtual Machine 202
- Dateien 270
  - auf SD-Karte 274
  - lesen 271
  - Ressourcen 274
  - schreiben 270
  - Textdateien 272
- Daten
  - als Preferences speichern 269
  - Persistenz 269
- Datenbanken
  - als Ressourcen 331
  - anlegen 328
  - Datensatz 327
  - Datensätze aktualisieren 336
  - Datensätze einfügen 333
  - Datensätze lesen 334
  - Datensätze löschen 336
  - Fremdschlüssel 328
  - Groß- und Kleinschreibung 331
  - öffnen 328
  - Primärschlüssel 328, 330
  - relationale 327
  - schließen 331
  - SQL 328
  - Treiber 328
- DatePickerDialog 247
- Datum, Auswahl über Dialog 247
- DDMS 210, 429
  - Devices-Fenster 430
  - Emulator Control-Fenster 433
  - File Explorer-Fenster 432
  - LogCat-Fenster 431
  - LogCat-Filter anlegen 210, 431
  - starten 429
- Debugging
  - DDMS 429
  - Debugger 434
  - Haltepunkte 437
  - Logausgabe 208
  - starten 434
  - Variablen inspizieren 439
- Debug-Monitor 210
- decodeResource() (BitmapFactory) 302
- delete() (SQLiteDatabase) 336
- deprecated 245
- Designer 124
  - Arbeitsschritte 124
  - Endgeräte simulieren 127
  - UI-Elemente ausrichten 124
  - UI-Elemente konfigurieren 126
  - UI-Hierarchie 126
- Dialog 244
  - dismiss() 245
  - setCancelable() 245
  - setOwnerActivity() 256
  - show() 256
- Dialoge 244
  - AlertDialog 246

- anzeigen 245
- anzeigen (ab Android 3.0) 375
- eigene 252
- erzeugen 244
- DialogFragment 366, 375
- dismiss() (Dialog) 245
- distanceBetween() (Location) 347
- distanceTo() (Location) 347
- dolnBackground() (AsyncTask) 362
- DPAD 199
- Drawable 216
- drawBitmap() (Canvas) 221
- drawCircle() (Canvas) 222
- drawColor() (Canvas) 221, 227
- drawLine() (Canvas) 222
- drawLines() (Canvas) 222
- drawOval() (Canvas) 222
- drawPoints() (Canvas) 222
- drawRect() (Canvas) 222
- drawRGB() (Canvas) 221
- drawRoundRect() (Canvas) 222
- drawText() (Canvas) 222
- drawTextOnPath () (Canvas) 229
- DVD, zum Buch 24, 443

### E

- Eclipse 34
  - Android-Plugin installieren 37
  - Apps erstellen 63
  - Apps exportieren 418
  - Code Assist 103
  - Desktop-Verknüpfung 37
  - Dialogfeld New Android App 45
  - Emulator-Startoptionen 424
  - erster Start 35
  - Exception-Behandlung ergänzen 99
  - Folding 92
  - Formatierung von XML-Layoutdateien 417
  - Hilfe zu Methodenargumenten 99
  - import-Anweisungen ergänzen lassen 97
  - Installation 35
  - Java-Kompatibilität einstellen 73
  - Klammernpaare identifizieren 101
  - Klasselemente hinzufügen 104
  - Launch-Konfigurationen 415
  - Layout-Designer 124
  - Liste der Klasselemente 98
  - Outline-Ansicht 126
  - Package Explorer 51
  - Package Explorer aktualisieren 159
  - Probleme mit der App-Erstellung 412
  - Projekte anlegen 44, 409
  - Projekte ausführen 411
  - Projekte deaktivieren 412
  - Projekte erstellen (Build) 411
  - Projekte importieren 415
  - Projekte löschen 412
  - Properties-Fenster 416
  - Quelldateien hinzufügen 111

- Quelldateien laden 81
- QuickFix 94
- QuickInfo 98
- Refactoring 106
- Syntaxhervorhebung 92
- Verbindung zu Android SDK 414
- Vorkommen markieren 102
- Warnsymbole 94
- Workspaces 35, 36, 53, 412, 413
- Wörterbuch 419
- Zeilennummern 102
- zu Definition wechseln 103
- EditText 144
  - getText() 144
  - inputType 144
  - minLines 144
  - password 144
  - text 144
- Eingabeaufforderung 27
- Emulator 65, 421
  - AVD bei App-Ausführung auswählen 380
  - AVD einrichten 65
  - AVD-Gerät 422
  - einrichten 379
  - Hoch- und Querformat 152
  - konfigurieren 380
  - Launch-Konfigurationen 381
  - SD-Karte 423, 425
  - Startoptionen 423
  - zurücksetzen 423, 424
  - Zurück-Taste 69
- encode() (Uri) 295
- Environment 275
- Erdanziehung 317
- Ereignisse 183
  - Activity-Klasse 192
  - anonyme Listener-Klassen 190
  - anonyme Listener-Objekte 191
  - Behandlungscode einrichten 184
  - Klickereignisse 185
  - Listener-Interfaces 185, 188
  - Listener-Methoden implementieren 185
  - Listener-Objekt registrieren 186
  - Menüs 241
  - OnClickListener 185, 187
  - OnDragListener 187
  - onEreignismethoden überschreiben 199
  - onFocusChangeListener 188
  - OnKeyListener 188
  - OnLongClickListener 188
  - onTouchListener 188, 194
  - Sender ermitteln 193
  - Spinner 395
  - Tastaturreignisse 197, 228
  - Tippereignisse 194
  - View-Parameter 193
  - Wischereignisse 196
- Erstellen 63
- execSql() (SQLiteDatabase) 330
- execute() (AsyncTask) 361
- Exportieren
  - Apps 418
- F**
- Farben 139, 166, 223
- Fehlermeldungen
  - analysieren 94
  - beheben 64, 94
  - R-Fehler 94
  - verschwinden nicht 64
  - Warnungen 64, 95
  - Warnungen unterdrücken 74
- FileInputStream 271
- fileList() (Activity) 273
- FileOutputStream 270
- fill...() (Canvas) 222
- fill\_parent 129
- Filter 318
  - Hochpass 319
  - Tiefpass 319
- findViewById() (Activity) 155
- finish() (Activity) 208, 266
- Fokus
  - Tastatureingaben 198
  - Views 117
- Folding 92
- Fotos 305
- Fragment 366
- FragmentManager 367
- Fragments 79, 365
  - Back-Stack 367
- FragmentTransaction 367
- FrameLayout 138
- G**
- Gebietsschema 396
- Geokoordinaten
  - dezimal 346
  - sexagesimal 346
- Geolokation
  - Daten empfangen 344
  - Empfänger abmelden 345
  - GPS 343
  - Netzwerk 343
  - Provider 343
  - Verfügbarkeit 343
- getAccuracy() (Location) 352
- getAction() (MotionEvent) 195
- getAltitude() (Location) 346
- getBearing() (Location) 346
- getCount() (BaseAdapter) 388
- getCount() (Cursor) 334
- getExternalStorageDirectory() (Environment) 275
- getFilesDir() (Activity) 272
- getInt() (Cursor) 335
- getIntent() (Activity) 262
- getItem() (BaseAdapter) 388
- getItemId() (MenuItem) 242
- getLatitude() (Location) 346

getLongitude() (Location) 346  
 getMenuInfo() (MenuItem) 242  
 getReadableDatabase() (SQLiteOpenHelper) 329  
 getResources() (Activity) 164  
 getSensorList() (SensorManager) 312  
 getSpeed() (Location) 346  
 getString() (Cursor) 335  
 getSystemService() (Activity) 312  
 getText() (EditText) 144  
 getTime() (Location) 346  
 getView() (BaseAdapter) 387  
 getWritableDatabase() (SQLiteOpenHelper) 329  
 getX() (MotionEvent) 197  
 getY() (MotionEvent) 197  
 Gliederung 92  
 Google Play 399  
 GPS 343  
 GPX 350  
 Gradientenfüllung 229  
 Grafik 215
 

- Bilder zeichnen 224
- Canvas 215
- Farben 223
- Füllung 223
- Koordinaten 223
- onDraw() 215, 219
- Sprites 224
- Umrisse 223
- Zeichenwerkzeuge 216
- zeichnen 220

 Gravitation
 

- Somigliana 318
- Vektor ermitteln 324

 GridLayout 136
 

- columnCount 136
- rowCount 136

 GridView 137, 386
 

- layout\_columnWidth 138
- layout\_gravity 138
- layout\_horizontalSpacing 138
- layout\_numColumns 138
- layout\_stretchMode 138
- layout\_verticalSpacing 138

 Größenangaben 130, 165  
 Groß- und Kleinschreibung
 

- Datenbanken 331
- Klassennamen 57, 82

 GUI. Siehe Benutzeroberflächen

## H

Haltepunkte 437  
 Handler 254, 383
 

- handleMessage() 385
- sendMessage() 384
- sendMessageDelayed() 385

 hasAccuracy() (Location) 352  
 Hierarchy Viewer 142  
 Hintergrund 139  
 Hintergrundbilder 140  
 Hochpass 319

## I

Icon-Menü 231  
 ID 60  
 ImageButton 144
 

- contentDescription 144
- onClick() 144
- src 144

 ImageView 145
 

- contentDescription 145
- scaleType 145
- setImageBitmap() 302
- setImageResource() 302
- src 145

 import 56  
 Importieren
 

- Klassen 56
- Projekte 415

 Innenabstand (Padding) 122  
 insert() (SQLiteDatabase) 333  
 Installation
 

- Android-Plugin 37
- Android-SDK 28
- ausführliche Beschreibung 23
- Eclipse 35
- JDK für Java SE 25

 Intent (Klasse) 258  
 Intents 76, 257
 

- Action 258
- Broadcast Intents 77, 267
- Bundle-Daten 261, 262
- Category 258
- Component 258
- Data 258
- Daten auslesen 262
- empfangen 262
- erzeugen 261
- explizite 259
- Extras 258
- implizite 259
- Intent-Filter 259
- senden 262
- Start-Activity 260
- zusätzliche Daten mitgeben 261

 isChecked() (CheckBox) 144  
 isProviderEnabled() (LocationManager) 344

## J

JAR-Dateien 84  
 jarsigner 401  
 Java
 

- Android-Kompatibilität 73
- JDK 24
- JRE 40
- Versionsnummern 40

 JDK für Java SE 24
 

- Eintrag in Systempfad (PATH) 25
- Installation 25

 JRE 40

- K**  
 Kamera 305  
 KeyEvent 198  
 keystore 401  
 KillableAfter-Flag 206  
 Klassen  
 – innere 113  
 – Namen 57  
 Klickereignisse 185  
 Konsole 27  
 Kontextmenüs 231, 238  
 Koordinaten, Grafik 223
- L**  
 Lagesensor 320  
 Launch-Konfigurationen 415  
 Layoutdateien  
 – formatieren 417  
 – im Designer 124  
 – selbst definierte View-Klassen 216  
 – XML-Code 119  
 Layout-Designer. Siehe Designer  
 Layoutparameter, allg. 128  
 – layout\_height 129  
 – layout\_marginBottom 130  
 – layout\_marginLeft 130  
 – layout\_marginRight 130  
 – layout\_marginTop 130  
 – layout\_width 129  
 Layouts 59, 172  
 – Attribute 121  
 – Designrichtlinien 115  
 – Größenangaben 130  
 – Hierarchie 126  
 – Hoch- und Querformat 151  
 – IDs zuweisen 154  
 – im Hierarchy Viewer 142  
 – laden 153  
 – per Code 59  
 – per XML 59  
 – setContentView() 60  
 – Stile 174  
 – tauschen 120  
 – View-Hierarchie 120  
 – XML-Code 119  
 – XML-Dateien 60  
 Layout-Views 78, 127  
 – AbsoluteLayout 139  
 – FrameLayout 138  
 – GridLayout 136  
 – GridView 137  
 – Layoutparameter 128  
 – Layoutregeln 127  
 – LinearLayout 131  
 – RelativeLayout 133  
 – TableLayout 135  
 Lebenszyklus, App 203  
 LIFO-Prinzip 204  
 LinearGradient 229  
 LinearLayout 131  
 – gravity 131  
 – layout\_gravity 132  
 – layout\_weight 132  
 – orientation 131  
 Listener-Interfaces 185, 188  
 Listenfelder 393  
 ListFragment 366, 370  
 ListView 337  
 load() (SoundPool) 293  
 Location  
 – convert() 347  
 – distanceBetween() 347  
 – distanceTo() 347  
 – getAccuracy() 352  
 – getAltitude() 346  
 – getBearing() 346  
 – getLatitude() 346  
 – getLongitude() 346  
 – getSpeed() 346  
 – getTime() 346  
 – hasAccuracy() 352  
 LocationListener 344  
 – onLocationChanged() 346  
 LocationManager 343  
 – isProviderEnabled() 344  
 – removeUpdates() 345  
 – requestLocationUpdates() 345  
 Log 208  
 Logging 208, 431  
 Lokale 396  
 Lösungen  
 – zu den Übungen 447
- M**  
 makeText() (Toast) 254  
 Manifestdatei 84  
 – Activities eintragen 265  
 – Berechtigungen (Permissions) 295  
 Margin (Außenabstand) 116, 130  
 Market Place 399  
 match\_parent 129  
 MediaController 303  
 MediaPlayer 294  
 – Audiodateien abspielen 295  
 – Audiodateien aus dem Internet abspielen 295  
 – Audioressourcen abspielen 294  
 – create() 294  
 – Endlosschleife 300  
 – pause() 295  
 – prepare() 298  
 – release() 299  
 – setDataSource() 298  
 – setLooping() 300  
 – start() 295  
 – stop() 295  
 – Systemressourcen freigeben 299  
 – wiederverwenden 297  
 MediaRecorder 305  
 MediaStore 305, 310  
 Mehrsprachigkeit 396



- Menüeinträge ActionBar 235
- MenüInflater 236
- MenuItem 242
  - getItemId() 242
  - getMenuInfo() 242
- MenuItem.OnMenuItemClickListener 243
- Menü 231
  - Action-Menü 232
  - Ereignisbehandlung 241
  - Kontextmenüs 231, 238
  - MenüInflater 236
  - Optionen-Menü 231, 236
  - Popup-Menü 232, 240
  - Ressourcen 173, 233
  - Submenüs 231
  - Untermenüs 231, 241
- Methoden
  - Callback 247
  - überschreiben 210
- Min-SDK 84
  - an Smartphone anpassen 72
- MotionEvent 195
  - ACTION\_DOWN 195
  - ACTION\_UP 195
  - getAction() 195
  - getX() 197
  - getY() 197
- moveToFirst() (Cursor) 335
- Multimedia
  - Audiodateien 294
  - Bilder 302
  - Fotos 305
  - Kamera 305
  - Ressourcen 173
  - Sound-Effekte 292
  - Video 303
- O**
  - onClick() (ImageButton) 144
  - onClick() (OnClickListener) 185, 187
  - onClick() (RadioButton) 145
  - onClick() (ToggleButton) 146
  - onClose() (SQLiteOpenHelper) 331
  - OnCompletionListener 296
    - onCompletion() 296
  - onContextItemSelected() (Activity) 242
  - onCreate() (Activity) 58
  - onCreateContextMenu() (Activity) 238
  - onCreateDialog() (Activity) 244
  - onCreateOptionsMenu() (Activity) 236
  - onCreate() (SQLiteOpenHelper) 329
  - OnDragListener 187
    - onDrag() 187
  - onDraw() (View) 215, 219
  - OnFocusChangeListener 188
    - onFocusChange() 188
  - OnItemClickListener 391
    - onItemClick() 391
  - OnItemSelectedListener 395
    - onItemSelected() 395
    - onNothingSelected() 395
  - onKeyDown() (View) 228
  - OnKeyListener 188, 310
    - onKey() 188
  - OnLoadCompleteListener 293
    - onLoadComplete() 293
  - onLocationChanged() (LocationListener) 346
  - OnLongClickListener 188
    - onLongClick() 188
  - onOptionsItemSelected() (Activity) 242
  - onPause() (Activity) 306
  - onPrepareDialog() (Activity) 245
  - onResume() (Activity) 306
  - onSensorChanged() (SensorEventListener) 314, 315
  - onTouchEvent() (View) 197
  - OnTouchListener 188, 194
    - onTouch() 188, 194
  - onTouch() (Switch) 146
  - onUpgrade() (SQLiteOpenHelper) 331
  - openFileInput() (Activity) 271
  - openFileOutput() (Activity) 270
  - Optionen-Menü 231, 236
  - Outline-Ansicht 126
  - Overflow-Menü 236
- P**
  - package 55
  - Package Explorer 51
  - PackageManager 267
  - Padding (Innenabstand) 116, 122
  - Paint 216
    - setAlpha() 229
    - setColor() 220
    - setStrokeWidth() 220
    - setStyle() 223
  - Pakete 46, 56, 81
  - parse() (Uri) 295
  - Path 229
  - PATH-Umgebungsvariable 25
  - pause() (MediaPlayer) 295
  - Permissions. *Siehe* Berechtigungen
  - Plattformen (Android) 30
  - play() (SoundPool) 294
  - Popup-Menü 232, 240
  - postInvalidate() (View) 280
  - Preferences 269
  - prepare() (MediaPlayer) 298
  - ProgressBar 145
    - max 145
    - progress 145
    - style 145
  - ProgressDialog 250
  - Projekte
    - anlegen 44
    - auf der Festplatte 52
    - Console-Fenster 68
    - Dateien 81
    - Grundgerüst 54
    - Java 7 73

- Package Explorer 51
- Projektverzeichnis 52
- Wizards 44
- Workspace 53
- Properties 126
- Properties-Datei 86

**Q**

- query() (SQLiteDatabase) 334
- QuickFix 94
- QuickInfo, zur API 98
- Quiz-App 283

**R**

- RadioButton 145
  - checked 145
  - onClick() 145
  - text 145
- RadioGroup 145
  - checkedButton 145
  - orientation 145
- random() (Math) 342
- Reaktions-App 276
- RectF 223
- Referenz, der Android-API 32
- registerForContextMenu() (Activity) 239
- register() (Sensor) 313
- RelativeLayout 133
  - layout\_above 134
  - layout\_align... 134
  - layout\_below 134
  - layout\_center... 134
  - layout\_toLeftOf 134
  - layout\_toRightOf 134
- release() (MediaPlayer) 299
- removeUpdates() (LocationManager) 345
- requestLocationUpdates() (LocationManager) 345
- Ressourcen 61, 157
  - als Objekte laden 164
  - alternative Ressourcenversionen 178
  - an Attribute zuweisen 162
  - anlegen 158
  - anlegen (im Designer) 182
  - an View-Eigenschaften zuweisen 161
  - Audiodateien 291
  - Bilder 171
  - Dateien 274
  - Dateinamen 158
  - Datenbanken 331
  - entfernen 164
  - Farben 166
  - Format 158
  - Größenangaben 165
  - im Code 163
  - Layouts 172
  - Mehrsprachigkeit 396
  - Menüs 173, 233
  - Multimedia 173
  - Rohdaten 173
  - Speicherort 158

- Stile 174
- String-Arrays 169
- Strings 167
- verwenden 161
- Videodateien 291
- Ressourcendateien 158
- R-Fehler 94
- R.java 82, 160, 163
- R-Klasse 60
- Rohdaten 173
- Root-Activity 204

**S**

- Schlüssel 400
- SD-Karte
  - Emulator 423, 425
  - Test auf Existenz 275
  - Zugriff 274
- SDK für Android 27
- SDK-Version 46, 50
- sendMessageDelayed() (Handler) 385
- sendMessage() (Handler) 384
- Sensor 312, 313
  - register() 313
  - Typen-Konstanten 311
- Sensoren
  - bei Sensor registrieren 313
  - Beschleunigungssensor 317
  - Daten auslesen 315
  - Filter 318
  - Lagesensor 320
  - Sensortypen 311
  - verfügbare Sensoren 312
  - Werte 316
- SensorEvent 312, 315
- SensorEventListener 312, 313
  - onSensorChanged() 314, 315
- SensorManager 312
  - getDefaultSensor() 313
  - getSensorList() 312
- Services 78
- setAlpha() (Paint) 229
- setCancelable() (Dialog) 245
- setColor() (Paint) 220
- setContentView() (Activity) 58, 60, 153
- setDataSource() (MediaPlayer) 298
- setGravity() (Toast) 254
- setImageBitmap() (ImageView) 302
- setImageResource() (ImageView) 302
- setLooping() (MediaPlayer) 300
- setOwnerActivity() (Dialog) 256
- setStrokeWidth() (Paint) 220
- setStyle() (Paint) 223
- SharedPreferences 269
- show() (Dialog) 256
- showDialog() (Activity) 245
- show() (Toast) 254
- Signieren 400
- SimpleCursorAdapter 337

- Somigliana 318
  - Sound 291
    - Audiodateien 294
    - MediaPlayer 294
    - Sound-Effekte 292
    - SoundPool 292
    - Töne 300
  - SoundPool 292
    - load() 293
    - play() 294
  - Spinner 146, 393
    - Ereignisbehandlung 395
    - konfigurieren 393
    - mit Daten füllen 393
    - onItemSelected 146
    - prompt 146
  - Sprites 224
  - SQL 328
  - SQLiteDatabase 329
    - close() 331
    - delete() 336
    - execSql() 330
    - insert() 333
    - query() 334
    - update() 336
  - SQLiteOpenHelper 328
    - getReadableDatabase() 329
    - getWritableDatabase() 329
    - onClose() 331
    - onCreate() 329
    - onUpgrade() 331
  - Start-Activity 86
    - startActivity() (Activity) 262
    - startActivityForResult() (Activity) 276
    - start() (MediaPlayer) 295
  - Startsymbol 153
  - startTone() (ToneGenerator) 300
  - StatFs 275
  - Stile 174
    - an Activities zuweisen 177
    - an Views zuweisen 174
    - definieren 174
    - parent-Attribut 176
    - Themes 177
    - Vererbung 176
  - stop() (MediaPlayer) 295
  - stopTone() (ToneGenerator) 301
  - String-Arrays 169
  - Strings 62, 167
    - strings.xml 61
    - style-Attribut 174
  - Support-Library 376
  - SurfaceView 306
  - Switch 146
    - checked 146
    - onTouch() 146
    - text 146
    - textOff 146
    - textOn 146
  - Syntaxhervorhebung 92
- 
- T**
  - TableLayout 135
  - TableRow 135
  - Target-SDK 84
  - Task 203
  - Tastaturereignisse 197, 228
  - Testen
    - auf Smartphone 69
    - im Emulator 65
  - TextView 146
    - text 146
    - textSize 146
    - textStyle 146
    - typeface 146
  - Themes 177
  - Threads 251, 278
  - Thumbnails 387
  - Tiefpass 319
  - TimePickerDialog 247, 248
  - Timer 110
  - TimerTask 109
  - Tippereignisse 194
  - Toast 254
    - makeText() 254
    - setGravity() 254
    - show() 254
  - Toasts 184, 254
  - ToggleButton 146
    - checked 146
    - onClick() 146
    - textOff 146
    - textOn 146
  - ToneGenerator 300
    - startTone() 300
    - stopTone() 301
  - Tools
    - jarsigner 401
    - keystore 401
    - translate() (Canvas) 222
- 
- U**
  - UFO-App 224
    - Screenshots der App 430
  - Ul. *Siehe* Benutzeroberflächen
  - Untermenüs 231, 241
  - update() (SQLiteDatabase) 336
  - Uri 295
    - encode() 295
    - parse() 295
- 
- V**
  - Veröffentlichung 399
    - Screenshots der App 430
  - Vibrator 382
    - vibrate() 382
  - Vibrieren 382
  - Video 291
    - Formate 294
    - MediaPlayer 303
    - Ressourcen 291
  - VideoView 303

## View

- eigene View-Klassen erzeugen 216
- eigene View-Klassen in Code 218
- eigene View-Klassen in XML 216
- onDraw() 215
- onKeyDown() 228
- onTouchEvent() 197
- postInvalidate() 280
- ViewGroup
  - addView() 219
- Viewgroups 78, 131
- Views 78
  - Attribute 121
  - Drehung 122
  - Eigenschaften 126
  - Fokussierbarkeit 117
  - Hierarchie 126
  - Hintergrund 122, 139
  - Hintergrundbild 140
  - Hintergrundfarbe 139
  - ID 122
  - Innenabstand (Padding) 122
  - Kontextmenüs 238
  - Layout-Views 78, 127
  - mit ID verbinden 154
  - on-Ereignismethoden überschreiben 199
  - Sichtbarkeit 122
  - Transparenz 122

- Viewgroups (Container) 78, 131
- Widgets 78, 143
- Zeichenflächen 78
- zeichnen 215
- Zugriff in Code 155

## W

- Warnungen 64, 95
- Website, zu Buch 39
- WebView 146
- WebViewFragment 366
- Widgets 78, 143
- Wischereignisse 196
- Wizards 44
- Workspace 53
- wrap\_content 129

## X

- xml-Layouts 60

## Z

- Zeichenflächen 78
- Zeichnen 220
- Zeit, Auswahl über Dialog 248
- Zertifikat, digitales 401
- Zufallsgenerators 280
- Zurück-Taste 69, 117, 245