

Uwe Post

Eine
Spiele-App
von
A bis Z

Android-Apps entwickeln

Ideal für Programmierneinsteiger geeignet



- ▶ Schritt für Schritt zu eigenen Apps und Spielen
- ▶ Inkl. Sprachgrundlagen von Java
- ▶ Animationen, Sounds, Zeichnen, Kamera, Bewegungssensoren, Highscores u. v. m.



Komplettes Startpaket mit der benötigten Software und allen Beispielen

Galileo Computing

Inhalt

Vorwort	11
1 Einleitung	13
1.1 Für wen ist dieses Buch?	13
Magie?	14
Große Zahlen	14
Technologie für alle	15
Die Grenzen der Physik	16
1.2 Unendliche Möglichkeiten	17
Baukasten	18
Spiel ohne Grenzen	19
Alles geht	22
1.3 Was ist so toll an Android?	22
MapDroyd	23
Google Sky Map	24
Bump	25
c:geo	27
barcoo	28
Öffi	29
Wikitude	30
Sprachsuche	32
Cut the Rope	33
Shaky Tower	35
2 Ist Java nicht auch eine Insel?	37
2.1 Warum Java?	37
2.2 Grundlagen	39
Objektorientierung: Klassen und Objekte	40
Konstruktoren	42
2.3 Pakete	43
Packages deklarieren	43
Klassen importieren	44
2.4 Klassen implementieren	45
Attribute	45
Methoden	48
Zugriffsbeschränkungen	50

	Eigene Konstruktoren	53
	Lokale Variablen	54
2.5	Daten verwalten	56
	Listen	56
	Schleifen	58
2.6	Vererbung	59
	Basisklassen	59
	Polymorphie	62
3	Vorbereitungen	65
3.1	Was brauche ich, um zu beginnen?	65
3.2	JDK installieren	67
3.3	Eclipse installieren	69
3.4	Tour durch Eclipse	71
3.5	Android Development Tool installieren	74
3.6	Android SDK installieren	76
3.7	SDK Tools installieren	77
3.8	Ein virtuelles Gerät erzeugen	78
3.9	Eclipse mit dem Handy verbinden	81
3.10	Was tun, wenn mein Eclipse verrücktspielt?	82
	Unerklärliche Unterstreichungen	82
	Ein Handy namens Fragezeichen	83
	Eclipse hängt sich auf	84
	Eclipse findet Resource-Dateien nicht	84
4	Die erste App	85
4.1	Sag »Hallo«, Android!	85
	Ein neues Android-Projekt erstellen	85
	Die StartActivity	87
	Der erste Start	92
4.2	Bestandteile einer Android-App	94
	Versionsnummern	95
	Activities anmelden	96
	Permissions	97
	Ressourcen	99
	Generierte Dateien	101
4.3	Benutzeroberflächen bauen	105
	Layout bearbeiten	105
	String-Ressourcen	109

	Layout-Komponenten	113
	Weitere visuelle Komponenten	116
4.4	Buttons mit Funktion	117
	Der OnClickListener	117
4.5	Eine App installieren	121
	Start mit ADT	121
	Installieren per USB	121
	Installieren mit ADB	122
	Drahtlos installieren	123
5	Ein Spiel entwickeln	127
5.1	Wie viele Stechmücken kann man in einer Minute fangen?	127
	Der Plan	127
	Das Projekt erzeugen	128
	Layouts vorbereiten	129
	Die GameActivity	130
5.2	Grafiken einbinden	133
	Die Mücke und der Rest der Welt	134
	Grafiken einbinden	135
5.3	Die Game Engine	137
	Aufbau einer Game Engine	137
	Ein neues Spiel starten	138
	Eine Runde starten	139
	Den Bildschirm aktualisieren	140
	Die verbleibende Zeit herunterzählen	146
	Prüfen, ob das Spiel vorbei ist	150
	Prüfen, ob eine Runde vorbei ist	152
	Eine Mücke anzeigen	152
	Eine Mücke verschwinden lassen	157
	Das Treffen einer Mücke mit dem Finger verarbeiten	160
	»Game Over«	161
	Der Handler	163
5.4	Der erste Mückenfang	167
	Retrospektive	168
6	Sound und Animation	173
6.1	Sounds hinzufügen	174
	Sounds erzeugen	174
	Sounds als Ressource	176
6.2	Sounds abspielen	177
	Der MediaPlayer	178

	MediaPlayer initialisieren	179
	Zurückspulen und Abspielen	179
6.3	Einfache Animationen	181
	Views einblenden	182
	Wackelnde Buttons	184
	Interpolation	186
6.4	Fliegende Mücken	191
	Grundgedanken zur Animation von Views	191
	Geschwindigkeit festlegen	191
	Mücken bewegen	193
	Bilder programmatisch laden	195
	If-else-Abfragen	197
	Zweidimensionale Arrays	198
	Resource-IDs ermitteln	200
	Retrospektive	201
7	Internet-Zugriff	205
7.1	Highscores speichern	205
	Highscore anzeigen	205
	Activities mit Rückgabewert	207
	Werte permanent speichern	207
	Rekordhalter verewigen	209
7.2	Bestenliste im Internet	214
	Ein App Engine-Projekt	215
	URL-Parameter entgegennehmen	217
	Daten im High Replication Datastore speichern	218
	Highscores aus dem Datastore auslesen	220
	Die Internet-Erlaubnis	222
	Der Android-HTTP-Client	223
	Background-Threads	228
	Die Oberfläche aktualisieren	230
	Highscores zum Server schicken	232
	HTML darstellen	234
	HTML mit Bildern	237
7.3	Listen mit Adaptern	240
	ListViews	240
	ArrayAdapter	244
	Eigene Adapter	247
	Recyceln von Views	251

8	Kamera und Augmented Reality	253
8.1	Die Kamera verwenden	253
	Der CameraView	254
	CameraView ins Layout integrieren	258
	Die Camera-Permission	260
8.2	Bilddaten verwenden	261
	Bilddaten anfordern	261
	Bilddaten auswerten	263
	Tomaten gegen Mücken	265
9	Sensoren und der Rest der Welt	271
9.1	Himmels- und sonstige Richtungen	271
	Der SensorManager	272
	Rufen Sie nicht an, wir rufen Sie an	272
	Die Kompassnadel und das Canvas-Element	274
	View und Activity verbinden	278
9.2	Wo fliegen sie denn?	278
	Sphärische Koordinaten	279
	Die virtuelle Kamera	281
	Mücken vor der virtuellen Kamera	282
	Der Radarschirm	286
9.3	Beschleunigung und Erschütterungen	292
	Ein Schrittzähler	293
	Mit dem SensorEventListener kommunizieren	295
	Schritt für Schritt	297
9.4	Hintergrund-Services	300
	Eine Service-Klasse	300
	Service steuern	303
	Einfache Service-Kommunikation	304
9.5	Arbeiten mit Geokoordinaten	307
	Der Weg ins Büro	307
	Koordinaten ermitteln	309
	Karten und Overlay	311
10	Tipps und Tricks	317
10.1	Fehlersuche	317
	Einen Stacktrace lesen	318
	Logging einbauen	321
	Schritt für Schritt debuggen	323

10.2	Views mit Stil	325
	Hintergrundgrafiken	325
	Styles	326
	Themes	327
	Button-Zustände	329
	9-Patches	330
10.3	Dialoge	332
	Standard-Dialoge	332
	Eigene Dialoge	337
	Toasts	340
10.4	Layout-Gefummel	341
	RelativeLayouts	341
	Layout-Gewichte	343
10.5	Troubleshooting	344
	Eclipse installiert die App nicht auf dem Handy	344
	App vermisst existierende Ressourcen	345
	LogCat bleibt stehen	345
11	Apps veröffentlichen	347
11.1	Vorarbeiten	347
	Zertifikat erstellen	347
	Das Entwickler-Konto	349
	Die Entwicklerkonsole	350
11.2	Hausaufgaben	353
	Updates	353
	Statistiken	355
	Fehlerberichte	357
11.3	In-App-Payment	359
	In-App-Produkte	361
	Der BillingService-Apparat	363
	BillingReceiver und BillingResponseHandler	365
11.4	Alternative Markets	367
	Amazon AppStore	368
	AppSLib	368
	AndroidPIT App Center	370
	SlideME.org	371
	Die Buch-DVD	373
	Index	375

»Wir irren uns nie.«
(HAL 9000)

4 Die erste App

Software installieren, Java-Crashkurs ... Jetzt wird es Zeit für Ihre erste App. Starten Sie Eclipse, schließen Sie Ihr Handy an, stellen Sie Kaffee (oder Tee) und Kekse bereit. Fertig? Auf in den Kampf!

4.1 Sag »Hallo«, Android!

Üblicherweise ist das erste Programm, das Sie in Lehrbüchern kennenlernen, eines, das den Text »Hallo, Welt« auf den Bildschirm schreibt. Mit etwas Glück lässt es Sie die Mehrwertsteuer berechnen oder D-Mark in Gulden umrechnen.

Aus Sicht eines Smartphones kommt das einer tödlichen Beleidigung ziemlich nahe, finden Sie nicht? Daher werden wir eine standesgemäße App vorziehen. Als kleine Vorbereitung, die schon ziemlich viel verrät, öffnen Sie bitte den Android Market und suchen nach »Text to Speech«. Installieren Sie die App, falls Sie sie noch nicht auf Ihrem Gerät haben. Sie stellt, wie Sie unschwer erraten können, Sprachausgabe-Funktionen bereit. Auf dem Emulator funktioniert das leider nicht ohne Weiteres, halten Sie daher Ihr Handy samt USB-Kabel bereit.

Jede App entspricht im Arbeitsbereich von Eclipse einem **Project**. Als ersten Schritt legen Sie ein neues Projekt an. Dank des installierten ADT gibt es die Möglichkeit, gleich ein Android-Projekt mit den nötigen Voreinstellungen anzulegen. Wählen Sie im Eclipse-Menü NEW • OTHER, oder drücken Sie + .

Ein neues Android-Projekt erstellen

Zum Anlegen von neuen Projekten oder Dateien gibt es Wizards. Wir benötigen den Wizard mit dem Namen ANDROID PROJECT. Es genügt, wenn Sie die ersten drei Buchstaben in das obere Textfeld eintippen: Daraufhin erscheinen darunter nur noch die Wizards mit dazu passenden Namen. Einer davon ist der gesuchte Wizard. Doppelklicken Sie darauf, um ihn zu starten (Abbildung 4.1).

Geben Sie dem neuen Projekt den Namen SAGHALLO. Wählen Sie in der Liste der Build Targets ANDROID 2.2 aus. Sie erinnern sich sicher, dass Sie diese SDK-Version zuvor mit dem SDK and AVD Manager installiert haben.

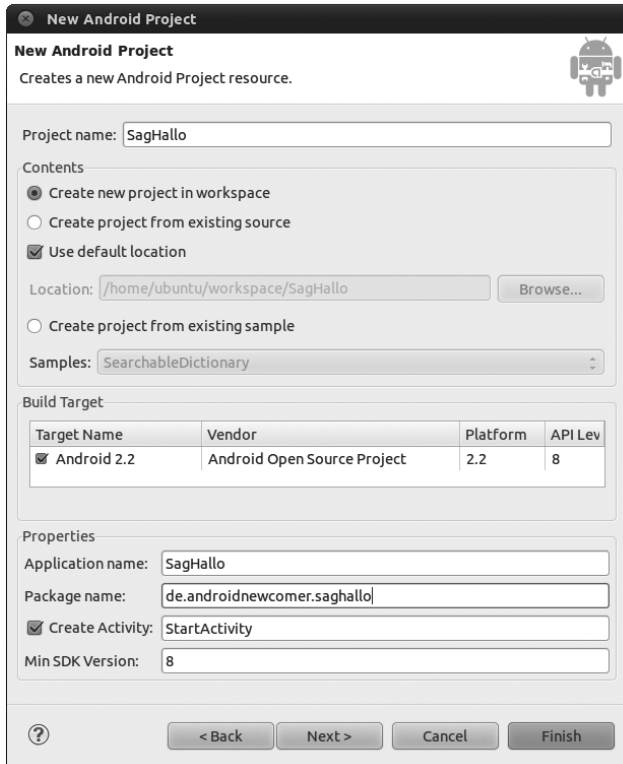


Abbildung 4.1 Der Wizard namens »New Android Project« erzeugt alle Dateien, die wir brauchen.

Auch als APPLICATION NAME tragen Sie SAGHALLO ein. Der PACKAGE NAME bleibt Ihnen überlassen, ich wähle in allen Beispielen `de.androidnewcomer` und hänge den Projektnamen in Kleinbuchstaben an, in diesem Fall heißt mein Package also: `de.androidnewcomer.saghallo`.

Lassen Sie den Wizard auch gleich eine Activity erzeugen. Activities sind Klassen, die jeweils App-Bildschirme verwalten, und ohne macht eine App nicht viel her. Nennen Sie die Activity `StartActivity`.

Es ist eine Konvention, den Namen jeder Activity-Klasse mit diesem Begriff enden zu lassen. Sie werden noch sehen, dass Activities entscheidende Bestandteile von Android-Apps sind, daher sollten die Klassen auf den ersten Blick als Activities erkennbar sein.

Tragen Sie zum Schluss als MIN SDK VERSION eine »8« ein. Diese Versionsnummer entspricht üblicherweise der Spalte API LEVEL des ausgewählten Build Targets. Damit bestimmen Sie, dass Ihre App nur auf Geräten ab Android 2.2 läuft.

Falls Sie ein älteres Handy haben sollten, müssen Sie hier die passende Version eingeben. Android 1.6 entspricht beispielsweise API-Level 4. Laden Sie sich mit dem AVD Manager die passende Umgebung herunter, falls nötig. Dann sehen Sie auch die zugehörige API-Level-Version.

Klicken Sie auf FINISH.

Die App, die der Wizard für uns erzeugt hat, kann natürlich noch nicht sprechen. Ihnen bleibt daher nichts anderes übrig, als es ihr beizubringen – indem Sie nun feierlich die ersten eigenen Java-Zeilen hinzufügen.

Die StartActivity

Klappen Sie im Package Explorer das dort entstandene Icon auf, das Ihr Android-Projekt **SagHallo** repräsentiert. Von den vielen Icons, die daraufhin auftauchen, ignorieren Sie zunächst alle bis auf das, neben dem SRC (Abkürzung für **Source Code**, also Quellcode) steht. In diesem Verzeichnis befinden sich alle Java-Dateien des Projekts. Der Wizard hat für den Anfang genau eine erzeugt, sie heißt *StartActivity.java* und befindet sich im Package `de.androidnewcomer.saghallo`, genau wie gewünscht.

Öffnen Sie das Package, und doppelklicken Sie auf *StartActivity.java*. Daraufhin zeigt Eclipse Ihnen im großen Fenster den vom Wizard erzeugten Java-Code (Abbildung 4.2).

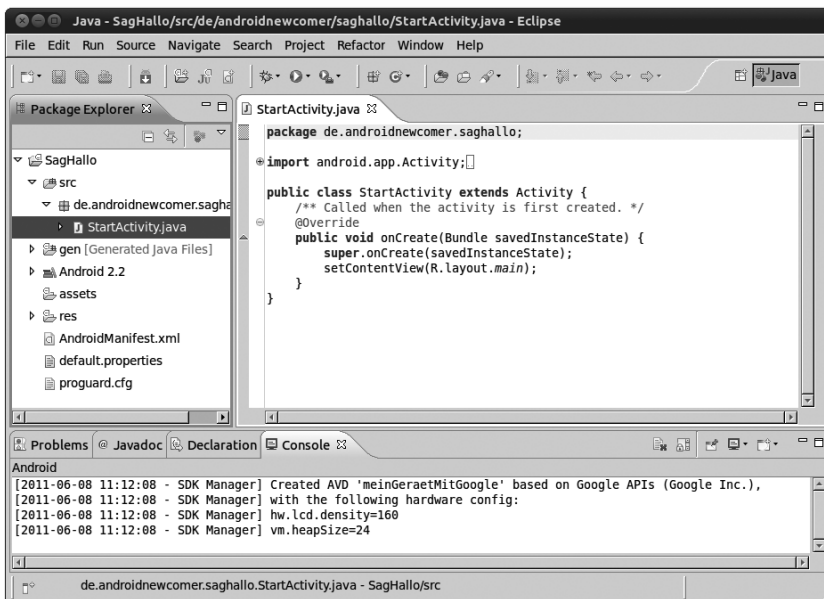


Abbildung 4.2 Der Wizard hat genau eine Java-Datei erzeugt.

Sie sehen, dass die Klasse `StartActivity` von einer Klasse namens `Activity` erbt und genau eine Methode enthält: `onCreate()`. Die Annotation `@Override` zeigt Ihnen, dass `onCreate()` offenbar eine gleichnamige Methode in der Elternklasse `Activity` überschreibt.

Wie der ebenfalls automatisch erzeugte Quellcodekommentar in freundlichem Blau erläutert, wird diese Methode beim ersten Start automatisch aufgerufen. Darum kümmert sich das Android-System ohne unser Zutun, nachdem es ein Objekt der Klasse erzeugt hat.

Kommentare

Es gibt zwei Möglichkeiten, Kommentare im Programmcode zu kennzeichnen, damit der Java-Compiler nicht versucht, den Text zu übersetzen.

Die eine Schreibweise schließt den Kommentar in Schrägstriche und Sterne ein:

```
/* Dies ist ein Kommentar.  
   Er kann sich über mehrere Zeilen erstrecken. */
```

Wie Sie sehen, können Sie auf diese Weise längere Kommentare schreiben. Oft verwenden Programmierer diese Notation, um kurze Codeschnipsel *auszukommentieren*.

```
if( bedingung1 /* && bedingung2 */ ) {
```

Diese Strategie dient dazu, auszuprobieren, wie sich ein Programm ohne einen bestimmten Teil verhält.

Die zweite Möglichkeit, Kommentare im Programmcode zu kennzeichnen, ist der doppelte Schrägstrich `//`.

```
Auto cabrio = new Auto(); // erzeugt mein neues Auto
```

Der Java-Compiler ignoriert alles, was hinter dem `//` steht, bis zum Ende der Zeile. Folglich können Sie damit keine einzelnen Elemente innerhalb einer Zeile auskommentieren, ebenso erfordert jede weitere Kommentarzeile einen weiteren einleitenden `//`.

Der Doppelschrägstrich wird gern am Zeilenanfang verwendet, um eine ganze Zeile auszukommentieren. Eclipse hilft Ihnen sogar dabei: Wenn Sie mehrere Zeilen markieren und `[Strg] + [⇧] + [7]` drücken (also sozusagen `[Strg] + [7]`), werden alle Zeilen mit `//` auskommentiert oder, wenn sie schon auskommentiert sind, wieder einkommentiert (d.h. die `//` entfernt). Natürlich klappt das auch über das Menü: Wählen Sie `SOURCE • TOGGLE COMMENT`.

Eclipse hebt Kommentare zwar farblich hervor, aber allzu viele kleine Kommentare verbessern nicht gerade die Übersicht.

Die Faustregel für Kommentare lautet: Schreiben Sie welche, wenn Sie es für möglich halten, dass Sie oder andere Programmierer eine Stelle sonst nicht auf Anhieb verstehen. Und überschätzen Sie dabei niemanden ...

Derzeit erledigt die `onCreate()`-Methode zwei Dinge: Erstens ruft sie die gleichnamige Methode der Elternklasse auf. Damit sie sich nicht selbst aufruft, steht

super davor. Selbstverständlich besitzt auch die Elternklasse eine Methode namens `onCreate()`, und sie erledigt wichtige organisatorische Aufgaben. Deshalb muss sie unbedingt aufgerufen werden.

Die zweite Codezeile in der Methode `onCreate()` ruft die Methode `setContentView()` auf. Da diese Methode offensichtlich nicht in `StartActivity` zu finden ist, können Sie davon ausgehen, dass sie in einer Elternklasse definiert ist. Die Methode erhält als Parameter einen Wert, über den später noch zu sprechen sein wird. Für den Moment genügt es, zu wissen, dass diese Zeile dafür sorgt, dass ein anderswo unter dem Namen `main` definierter Bildschirminhalt angezeigt wird.

Zur Erinnerung: Private Methoden

In einer Methode können Sie alle Methoden derselben Klasse aufrufen, außerdem alle Methoden der Elternklasse, die nicht mit dem Modifizierer `private` Ihren Blicken entzogen sind:

```
class Elternklasse {
    private void eineMethode() {
        ...
    }
}
...
class Kindklasse extends Elternklasse {
    public void testMethode() {
        eineMethode(); // Fehler
    }
}
```

Die Programmierer der Klasse `Activity` stellen Ihnen eine Menge hilfreicher Methoden zur Verfügung, die Sie in eigenen von `Activity` abgeleiteten Klassen verwenden können.

Lassen Sie uns nun die `StartActivity` um die gewünschte Sprachausgabe erweitern. Das wird dann dazu führen, dass die App beim ersten Start zu uns spricht.

Fügen Sie zunächst der Klasse ein Attribut hinzu:

```
private TextToSpeech tts;
```

Diese Eingabe quittiert Eclipse mit einer roten Unterstreichung, weil `TextToSpeech` unbekannt ist. Rote Unterstreichung heißt: Syntaxfehler, der Compiler kann dies nicht übersetzen, folglich kann er kein lauffähiges Programm erzeugen. Also müssen Sie `TextToSpeech` importieren.

Die gesuchte Klasse befindet sich in einem Paket des Android SDK, daher müssen Sie sie importieren. Glücklicherweise nimmt Ihnen Eclipse diese Arbeit ab, weil es im Gegensatz zu Ihnen schnell nachschauen kann, welches das zu importie-

Index

@Override 88
9-Patches 330

A

AAC+ 176
above 342
abstract 60, 301
Accelerometer 292
Activity 86
ActivityNotFoundException 319
Adapter 240
ADB 78
adb 84
add() 311
addView() 156
ADT 70, 74, 85
AlertDialog 333
AlertDialog.Builder 333
Alpha 182
Alpha-Transparenz 142
Amazon 368
andengine 34
Android Debug Bridge 78, 122
Android Debug-Bridge 83
Android Development Tool 70, 101
Android Market 23
Android Resource Manager 84, 102
Android SDK 76
Android Virtual Device 77
AndroidHttpClient 223
Android-ID 366, 367
Android-Manifest 94, 99, 130, 133, 302
AndroidPI 370
AndSMB 123
Animation 181
AnimationListener 188
Annotation 62
anonyme innere Klasse 190
Apache 124
API-Key 311
APK 102, 345, 347
apkbuilder 76
Apotheke 31

App Center 370
Application Nodes 96
AppsLib 368
AppStore 23
ArrayAdapter 244
ArrayList 56
ArrayWayOverlay 314
Atomreaktoren 39
Attribut 45
AttributeSet 287
Audacity 174
Audio-Formate 176
Augmented Reality 32, 253, 279
Auswahlliste 244
AVD 77–79
Azimuthwinkel 280

B

Background 135, 326
Background-Thread 229
barcoo 29
Basisklasse 60
Baumstruktur 72
Beans 37
Bedingung 50
Beschleunigungssensor 20, 26
Bildschirmausrichtung 97
BillingReceiver 365
BillingResponseHandler 365
BillingService 363–364
bin 121
Bit 45
boolean 45, 47, 151
Boolsche Operatoren 150
Breakpunkt 323
Browserspiele 39
Build Target 85
Bump 26
Button 107
byte 47
Bytecode 38

C

C 37
 C++ 37
 c:geo 27
 cacheColorHint 242
 Calendar 336
 Camera 254
 CameraView 254
 Canvas 275
 Cast 119
 Casting 153
 Caused by 320
 char 47
 CheckBox 107
 checked 330
 Checked Exceptions 228
 Checkout 349
 Children 157
 Chrominanz 264
 class 40
 clear() 311
 colors.xml 338
 Compiler 37
 Console View 121
 Constructor 42
 Content Assist 131
 Context 154
 Countdown 146
 CPU 37
 Culicidae 127
 Custom View 275, 291
 Cut the Rope 34

D

Dalvik VM 38
 Date 156
 DatePicker 117
 DatePickerDialog 333, 336–337
 DDMS 74
 Debug-Perspektive 323
 Debug-View 324
 Debug-Zertifikat 349
 Denglisch 41
 device independant pixels 136
 Dialog 161, 332
 DialogInterface 333

Digicam 29, 253
 Digitale Signatur 347
 dismiss() 334
 DisplayMetrics 145
 doGet() 217
 double 47
 draw9patch 331
 drawable 100, 103, 105
 drawArc() 289
 Dropbox 124

E

Eclipse 39, 45, 69, 81
 Editable 120
 EditText 107
 einblenden 182
 else 197
 Emulator 66, 70, 76, 78
 enabled 330
 Enterprise 32
 Entwicklerkonsole 350, 358
 Entwickler-Konto 349
 Entwicklungsumgebung 69
 Erdanziehungskraft 292
 Erdbeben 20
 Erdbeschleunigung 20
 Ereignis-Warteschlange 165
 Event 164
 Eventqueue 164
 Exceptions 226
 extends 60

F

Farb-Ressourcen 141
 Fehlerberichte 321
 FILL 276
 FILL_AND_STROKE 276
 final 103, 339
 findViewById() 119, 153, 249, 339
 finish() 132
 flickr 31
 float 47, 284
 focused 330
 Form Widgets 107
 Fortran 37

Fragezeichen 83
 FrameLayout 141
 fromHtml() 237

G

Galaxy of Fire 2 173
 Game Engine 137
 Garbage Collection 84
 Garbage Collector 42, 179
 Geocaching 21, 27
 Geokoordinaten 21
 getAnimation() 190
 getChildAt() 157
 getChildCount() 157
 getDefaultSensor() 272
 getDrawable() 238
 getIdentifier() 170, 200
 getRessources() 213
 getSharedPreferences() 208
 getSystemService() 272, 310
 getter 62
 getText() 212
 getWriter() 221
 GIMP 195
 GONE 211
 Google App Engine 214
 Google Maps 21, 23
 Google Sky Map 25
 GPS 21, 307
 Grafiken 100
 Gravity 114, 136, 156

H

handleMessage() 296
 Handler 163, 165, 295–296, 305
 Hanger 84
 Hexadezimale Farbwerte 141
 Hexadezimalzahl 104, 106
 Hierarchy Viewer 75
 High Replication Datastore 218
 Hintergrundfarbe 242
 HorizontalScrollView 116
 HTML 234
 HTML-Farbcodes 142
 HTTP-Client 223

HttpEntity 224
 HttpGet 223
 HttpServletResponse 218
 Hubble-Teleskop 25
 HVGA 79
 Hypertext Transfer Protocol 216

I

Icon 97, 100, 351
 icon.png 100
 ids.xml 156
 if 50
 IllegalArgumentException 228
 ImageButton 116
 ImageGetter 238
 ImageView 116
 IMarketBillingService 363
 implements 91, 118
 importieren 44, 89
 In-App-Payment 359
 initialisieren 46
 Inkscape 134, 325, 351
 innere Klasse 184
 InputStreamReader 224
 Installation 355
 Instanz 40
 instanziierten 41
 int 47
 Integer 46
 Intent 132
 Intent Filter 96–97
 Interface 91, 118
 interface 91
 Interpolation 186
 Interpolator 187
 invalidate() 275
 INVISIBLE 211
 IOException 226, 256, 345

J

jar 312
 Java 37
 Java Development Kit 67
 Java Runtime Environment 38
 Java-Kompatibilitat 83

Java-Perspective 71
Java-Runtime 40–41
JDK 67
JRE 38

K

Kaffeemaschine 67
kartesisches Koordinatensystem 279
Keystore 348
Klasse 40
Kommentar 88
Konstante 156
Konstruktor 42
Kopierschutzmechanismus 353
Kriegshammer 14
Kugelkoordinatensystem 279

L

Labyrinth 20
Lagesensoren 35
landscape 97
Laufvariable 157
Launch Options 80
Launcher 97
Layout 106, 129, 131, 342
Layout gravity 141
Layout Weight 143
layout_weight 343
Layout-Datei 161
Layout-Editor 328
Layouteditor 105
LayoutInflater 248, 251, 341
LayoutParams 145, 155, 194
LinearLayout 113
lineTo() 277
ListView 116, 240
Lizensierungsservice 353
loadAnimation() 183
Location 310
LocationManager 310
Log Level 318
Log.e 321
LogCat 318
Logcat 257
Log-Filter 322

Logging 257
LOGTAG 322
lokale Klasse 296
long 47
Loop 157
Luminanz 264

M

Magnetfeldsensor 21, 271
main.xml 106
MapActivity 312
MapController 315
MapDroyd 23
mapsforge 311
MapView 313
MapViewMode 313
Maßstab 145
match_parent 114
Math 145, 153
Math.min() 145
MediaPlayer 178
Methode 49
Mikrofon 21, 33, 174
Mindestpreise 353
modifier 44
Modifizierer 44
moveTo() 277
mp3 176
Mücke 18, 128
Multitasking 163

N

Namespace-Attribut 183
network based location 307
netzwerkbasierte Ortsbestimmung 307
notifyDataSetChanged() 250
NullPointerException 228, 306, 320
NV21 263
NV21Image.java 269

O

Objekte 40
objektorientiert 40

Öffi 30
 Ogg Vorbis 176
 onActivityResult() 207
 onClick() 333
 onClickListener 118, 155, 294, 303, 308,
 339
 onClickListener() 132
 onCreate() 88
 onDateSet() 336
 onDateSetListener 336
 onDestroy() 179
 onDraw() 275
 onInit() 90
 onKeyDown() 251
 onLocationChanged() 310
 onPreviewFrame() 262
 onResume() 206
 onSensorChanged() 274, 283, 295
 Openstreetmaps.org 311
 Operator 48
 Organize Imports 90, 94
 OSM 24
 Outline 72, 113, 171
 OutOfMemoryError 359
 Overlays 314
 OverlayWay 314
 Override 62

P

Package 43
 Package Builder 102
 Package Explorer 72, 87, 94, 99, 105
 Padding 136
 Paint 276
 Panoramico 31
 Parameter 42, 50
 parseInt() 218
 Pascal 37
 Path 276
 PayPal 370–372
 Permission 97, 222
 Perspective 71
 Pfad 276
 Physik-Engine 34
 Pivotpunkt 185
 Pizzeria 31
 Plugins 70

PNG 100, 326
 Polarwinkel 280
 Polymorphie 63
 portrait 97
 postDelayed() 166, 335
 PreviewCallback 262
 primitive Datentypen 46–47
 private 51, 61, 89
 Privater Schlüssel 348
 Problems 72
 Programmbibliothek 312
 ProgressBar 107
 ProgressDialog 333–334
 protected 61
 public 44
 purchaseResponse() 365
 Pythagoras 201

Q

qualified name 44
 qualifizierter Name 44
 Query 220

R

R.java 102, 119
 Radar 286
 Random 148
 raw 176
 Receiver 365
 Refactor 151, 169, 246
 Refactoring 152
 Reference Chooser 135
 RegionalExpress 42
 registerListener() 273
 RelativeLayout 341
 removeCallbacks() 186
 removeUpdates() 311
 removeView() 160
 replace() 226
 Request 216
 requestLocationUpdates() 310
 requestPurchase() 364
 res 99, 103
 Resource Chooser 108
 Resource Manager 102, 106

Response 216
 Ressource Chooser 136
 Ressourcen 99
 Restore 72
 rotate() 277
 round() 145, 196
 Router 21
 run() 165, 229
 Runescape 39
 Runnable 165, 229
 runOnUiThread() 231
 RuntimeException 228

S

Sampling 176
 scale-independant pixels 136
 Schatztruhen 16
 Schleife 58, 157
 Screen Orientation 294
 Screenshots 350
 ScrollView 116
 SDK Platform Tools 76
 SD-Karte 79
 selected 330
 selector 329
 sendEmptyMessage() 306
 SensorEvent 274
 SensorEventListener 273, 295
 SensorManager 272, 294
 Serveranwendung 27, 30, 39
 Service 300, 307
 Servlet 216
 setAnimationListener() 190
 setAntiAlias() 276
 setBackgroundResource() 170
 setColor() 276
 setContentView 106
 setContentView() 131, 243, 278, 313, 339
 setDisplayOrientation() 256
 setImageResource() 170, 198
 setLayoutParams() 194
 setOneShotPreviewCallback() 262
 setPreviewDisplay() 255
 setProgress() 336
 setResult() 207
 setStrokeWidth() 287
 setStyle() 276
 setTag() 280
 setter 62
 setVisibility() 211
 setWayData() 315
 Shaky Tower 35
 Shared Preferences 207
 SharedPreferences 366
 SharesFinder 123
 short 47
 SimpleStringSplitter 235, 249
 Single Processing 163
 SlideME 371
 SMB 123
 SO-8859-1 225
 Software Development Kit. Dahinter 76
 Software Sites 74
 Sound 173
 Sound-Formate 176
 Speicher 41
 sphärische Koordinaten 279
 Spiel 34
 Spielregeln 137
 Spinner 245
 Sprachausgabe 85, 90
 Sprachsuche 32
 src 87
 Stacktrace 319, 358
 Standard-Dialoge 333
 Standard-Konstruktor 54
 startActivity 87, 106
 startActivity() 132
 startActivityForResult() 207, 246
 startAnimation() 184
 startPreview() 258
 startService() 304
 state_pressed 330
 static 304, 306
 statische Attribute 304
 Statische Methoden 145
 Statistik 355
 Steuerrad 20
 stopService() 304
 Stream 224
 String 48
 String-Konstante 111
 String-Ressource 109
 strings.xml 111, 213
 String-Verweise 111
 STROKE 276

Stromverschwendung 50
 Styles 326
 super 89
 surfaceChanged() 256
 surfaceCreated() 255
 SurfaceHolder 254
 SurfaceView 117, 254
 svg 134
 Synonymwörterbuch 18

T

Tags 156
 Task List 72
 Telepathie 106
 Text Style 141
 Text to speech 85
 TextColor 141, 327
 TextToSpeech 90
 TextView 107
 Theme 327
 Thesaurus 18
 this 90
 Threads 229
 TimePicker 117
 TimePickerDialog 333, 336
 Toast 340
 Traceview 75
 translate() 277
 Transparenz 100
 Tricorder 19, 297
 trim() 212
 try-catch 227
 Tutorials 73
 Type-Casting 153

U

Überschreiben 62
 Ubuntu 71
 UI-Thread 229, 231
 Unchecked Exceptions 228
 Unicode 225
 Update 355
 URLEncoder 233
 URL-Parameter 217
 USB-Debugging 81

USB-Kabel 65, 81, 85, 92
 User Agent 223
 Uses Permission 98
 UTF-8 225

V

Vampir 127
 Variable 54
 Variablennamen 41–42, 46
 Vektor 191
 Venus 25
 Verantwortung 51
 Vererbung 59–60
 Vergleichsoperator 51
 Versionscode 95
 Versionsname 95
 Versionsnummer 354
 VideoView 117
 View 119, 153
 ViewGroup 157
 Views bewegen 191
 virtuelle Kamera 281
 virtueller Raum 279
 void 49
 Vollbild-Modus 170

W

Wahrheitswert 50
 Wahrscheinlichkeit 147
 Wasserkocher 67
 WAV 176
 Webcams 31
 WebView 116
 while() 267
 while(bedingung) 157
 Wikipedia 32
 Wikitude 31
 Wizard 128
 workspace 70
 Wrap in Container 258
 wrap_content 114, 136

X

XE 124, 188
XML 258, 329
xmlns 183

Y

YCbCr_420_SP 263
Youtube 351

Z

Zeitmaschine 16
Zentrifugalkraft 292
Zertifikat 348
Zufallsgenerator 147–148, 192
Zugspitzbahn 17
Zuweisungsoperator 46, 51
zweidimensionales Array 198