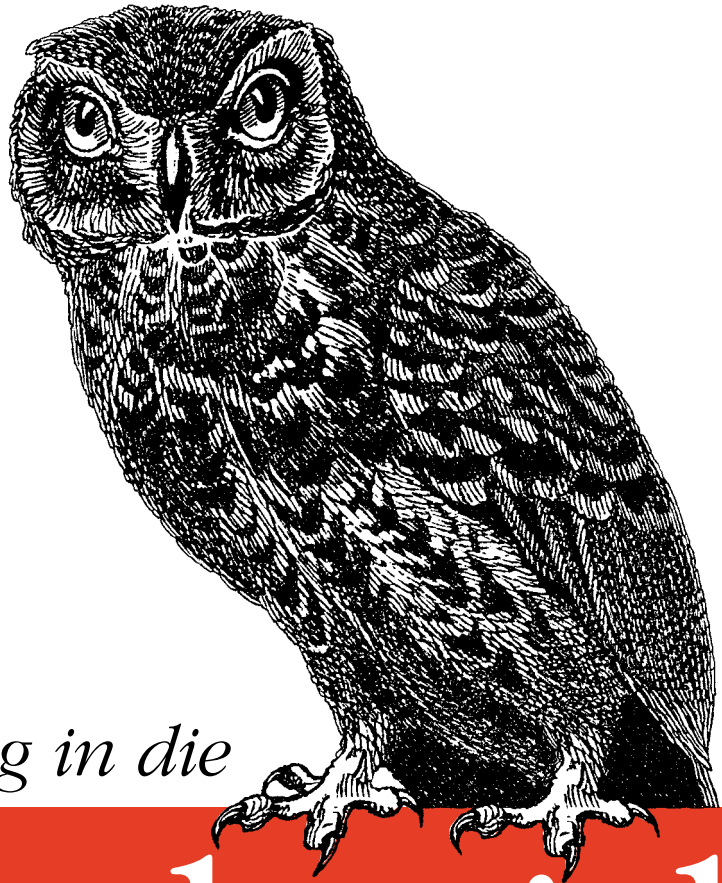


*Apps für den Android Market entwickeln*



*Einführung in die*

# Android Entwicklung

**O'REILLY®**

*Marko Gargenta*  
*Deutsche Übersetzung von Lars Schulten*

<b>Vorwort</b> .....	<b>XIII</b>
<b>1 Android im Überblick</b> .....	<b>1</b>
Was ist Android? .....	1
Was »umfassend« bedeutet .....	1
Open Source-Plattform .....	2
Speziell für Mobilgeräte .....	2
Geschichte .....	3
Googles Motivation .....	3
Open Handset Alliance .....	4
Android-Versionen .....	4
Zusammenfassung .....	6
<b>2 Die Systemstruktur</b> .....	<b>7</b>
Architekturüberblick .....	7
Linux .....	7
Portabilität .....	7
Sicherheit .....	8
Funktionsumfang .....	8
Native Bibliotheken .....	9
Dalvik .....	9
Android und Java .....	10
Das Application-Framework .....	11
Die Apps .....	12
Die APK-Datei .....	12
App-Signierung .....	12
App-Vertrieb .....	13
Zusammenfassung .....	13
<b>3 Kurzeinführung</b> .....	<b>15</b>
Das Android-SDK installieren .....	15
Einen PATH zu den Werkzeugen einrichten .....	16
Eclipse installieren .....	16

Eclipse-Arbeitsbereiche .....	17
Die Android-Entwicklungswerkzeuge einrichten .....	17
Hallo Welt .....	18
Ein neues Projekt erstellen .....	18
Die Manifestdatei .....	20
Der Layout-XML-Code .....	21
Die Datei strings .....	21
Die R-Datei .....	21
Der Java-Quellcode .....	22
Der Emulator .....	22
Emulator vs. echtes Gerät .....	25
Zusammenfassung .....	25
<b>4 Die grundlegenden Bausteine .....</b>	<b>27</b>
Was sind die grundlegenden Bausteine? .....	27
Ein praxisnahes Beispiel .....	27
Activities .....	28
Der Lebenszyklus von Activities .....	28
Intents .....	31
Services .....	32
Content-Provider .....	33
Broadcast-Receiver .....	34
Der Application-Context .....	35
Zusammenfassung .....	35
<b>5 Überblick über das Yamba-Projekt .....</b>	<b>37</b>
Die Yamba-App .....	37
Die Entwurfsstrategie .....	39
Der Projektentwurf .....	39
Teil 1: Android-Benutzerschnittstelle .....	40
Eine Activity erstellen .....	41
Netzwerk und Multithreading .....	41
Android-Apps debuggen .....	41
Teil 2: Einstellungen, Dateisystem, Optionsmenü und Intents .....	41
Die Activity .....	42
Das Menüsystem und Intents .....	42
Das Dateisystem .....	42
Teil 3: Android-Services .....	42
Services .....	42
Das Application-Objekt .....	42
Teil 4: Mit Datenbanken arbeiten .....	43
Androids SQLite-Unterstützung .....	43
Erneute Umgestaltung des Codes .....	43

Teil 5: Listen und Adapter .....	43
Die Timeline-Activity .....	43
Noch mehr Refactoring? .....	44
Teil 6: Broadcast-Receiver .....	44
Boot- und Netzwerk-Receiver .....	44
Der Timeline-Receiver .....	44
Berechtigungen .....	44
Teil 7: Content-Provider .....	44
Die Nachrichtendaten .....	45
Android-Widgets .....	45
Teil 8: System-Services .....	45
Kompass und Position .....	45
Intent-Service, Alarm und Notifikationen .....	45
Zusammenfassung .....	46
<b>6 Android-Benutzerschnittstelle .....</b>	<b>47</b>
Zwei Verfahren zur Erstellung von Benutzerschnittstellen .....	47
Deklarative Benutzerschnittstellen .....	47
Benutzerschnittstellen programmieren .....	48
Das Beste aus beiden Welten .....	48
Views und Layouts .....	48
LinearLayout .....	49
TableLayout .....	50
FrameLayout .....	50
RelativeLayout .....	50
AbsoluteLayout .....	50
Erste Schritte im Yamba-Projekt .....	51
Das Layout der StatusActivity .....	52
Wichtige Widget-Eigenschaften .....	54
String-Ressourcen .....	55
Die StatusActivity-Java-Klasse .....	56
Den App-spezifischen Objekt- und Initialisierungscode erstellen .....	56
Code kompilieren und Projekte erstellen: Dateien speichern .....	59
Die jtwitter.jar-Bibliothek einfügen .....	59
Die Manifestdatei für die Internetberechtigungen aktualisieren .....	61
Logging unter Android .....	62
LogCat .....	62
Threading unter Android .....	64
Nur ein Thread .....	65
Ausführung mit mehreren Threads .....	66
AsyncTask .....	67
Andere UI-Events .....	70
Farben und Bilder ergänzen .....	74
Bilder hinzufügen .....	74
Etwas Farbe reinbringen .....	77

Alternative Ressourcen .....	79
Die Benutzerschnittstelle optimieren .....	80
Der Hierarchy Viewer .....	81
Zusammenfassung .....	81
<b>7 Einstellungen, das Dateisystem, das Optionsmenü und Intents .....</b>	<b>83</b>
Einstellungen .....	83
Prefs-Ressource .....	84
Die PrefsActivity .....	87
Die Manifestdatei aktualisieren .....	88
Das Optionsmenü .....	89
Die Menüressource .....	89
Android-Systemressourcen .....	91
Die StatusActivity aktualisieren, um das Menü zu laden .....	91
Eine Verarbeitung der Menüevents in die StatusActivity einbauen .....	92
Strings-Ressource .....	93
SharedPreferences .....	94
Das Dateisystem .....	96
Expedition ins Dateisystem .....	96
Dateisystempartitionen .....	96
Systempartition .....	97
SD-Karten-Partition .....	98
Die Benutzerdatenpartition .....	98
Dateisystemsicherheit .....	99
Zusammenfassung .....	99
<b>8 Services .....</b>	<b>101</b>
Das Yamba-Application-Objekt .....	102
Die Klasse YambaApplication .....	102
Die Manifestdatei aktualisieren .....	104
StatusActivity vereinfachen .....	105
UpdaterService .....	106
Die Java-Klasse UpdaterService erstellen .....	106
Die Manifestdatei aktualisieren .....	108
Menüelemente einfügen .....	108
Die Verarbeitung des Optionsmenüs aktualisieren .....	109
Den Service testen .....	110
Schleifen im Service .....	110
Den Service testen .....	113
Daten abrufen .....	114
Den Service testen .....	117
Zusammenfassung .....	117

<b>9</b>	<b>Die Datenbank</b>	<b>119</b>
	Über SQLite	119
	SQLiteOpenHelper	120
	Das Datenbankschema und seine Erstellung	121
	Die vier Grundoperationen	121
	Cursor	122
	Erstes Beispiel	123
	Den UpdaterService anpassen	125
	Den Service testen	128
	Datenbank-Constraints	130
	Die Nachrichtendaten isolieren	131
	Zusammenfassung	136
<b>10</b>	<b>Listen und Adapter</b>	<b>139</b>
	Die TimelineActivity	139
	Elementares TimelineActivity-Layout	140
	Der ScrollView	140
	Die Klasse TimelineActivity erstellen	141
	Über Adapter	144
	Der TimelineActivity einen ListView hinzufügen	144
	Ein RowLayout erstellen	145
	In TimelineActivity.java einen Adapter erstellen	147
	TimelineAdapter	149
	ViewBinder: eine bessere Alternative für TimelineAdapter	152
	Die Manifestdatei aktualisieren	153
	Anfängliche App-Einrichtung	155
	Eine Activity-Unterklasse als gemeinsame Basis	155
	Service-An/Aus-Schalter	156
	Zusammenfassung	161
<b>11</b>	<b>Broadcast-Receiver</b>	<b>163</b>
	Über Broadcast-Receiver	163
	Der BootReceiver	164
	Den BootReceiver in der Android-Manifestdatei registrieren	165
	Den BootReceiver testen	165
	Der TimelineReceiver	165
	Intents per Broadcast absetzen	167
	Der Network-Receiver	169
	Eigene Berechtigungen zum Senden und Empfangen von Broadcasts einführen	172
	Berechtigungen in der Manifestdatei deklarieren	172
	Den Service anpassen, um die Berechtigung einzufordern	173
	Den TimelineReceiver anpassen, um Berechtigungen einzufordern	175
	Zusammenfassung	175

<b>12</b>	<b>Content-Provider</b>	<b>177</b>
	Einen Content-Provider erstellen	177
	Den URI definieren	178
	Daten einfügen	179
	Daten aktualisieren	180
	Daten löschen	181
	Daten abfragen	182
	Den Datentyp ermitteln	182
	Die Android-Manifestdatei aktualisieren	184
	Den Content-Provider in einem Widget nutzen	184
	Die Klasse YambaWidget implementieren	184
	Das XML-Layout erstellen	187
	Die AppWidgetProviderInfo-Datei erstellen	188
	Die Manifestdatei aktualisieren	188
	Das Widget testen	189
	Zusammenfassung	189
<b>13</b>	<b>System-Services</b>	<b>191</b>
	Kompass-Demo	191
	Allgemeine Schritte bei der Verwendung von System-Services	192
	Benachrichtigungen durch den Kompass	192
	Kompass-Haupt-Activity	194
	Das Rose-Widget	196
	Der Location-Service	198
	Wo bin ich?-Demo	198
	Den Location-Service in Yamba integrieren	202
	Die Einstellungen anpassen	203
	Die YambaApplication anpassen	204
	Die StatusActivity anpassen	204
	Intent-Service	208
	Alarme	211
	Den Einstellungen ein Intervall hinzufügen	211
	Den BootReceiver anpassen	213
	Benachrichtigungen senden	214
	Zusammenfassung	217
<b>14</b>	<b>Die Android Interface Definition Language</b>	<b>219</b>
	Den entfernten Service implementieren	220
	Die AIDL schreiben	220
	Den Service implementieren	221
	Ein Parcel implementieren	223
	In der Manifestdatei registrieren	224

Den entfernten Client implementieren .....	225
An den entfernten Service binden .....	226
Testen, ob alles funktioniert .....	228
Zusammenfassung .....	229
<b>15 Das Native Development Kit (NDK) .....</b>	<b>231</b>
Wozu das NDK gedacht ist (und wozu nicht) .....	231
Probleme, die das NDK löst .....	232
Die Toolchain .....	232
Die Bibliotheken verpacken .....	232
Dokumentation und Header-Dateien .....	232
Ein NDK-Beispiel: Fibonacci .....	233
FibLib .....	233
Die JNI-Header-Datei .....	235
C-Implementierung .....	237
Das Makefile .....	238
Die Shared-Library erstellen .....	238
Die Fibonacci-Activity .....	239
Testen, ob alles funktioniert .....	241
Zusammenfassung .....	241
<b>Index .....</b>	<b>243</b>



---

# Android im Überblick

In diesem Kapitel werden Sie erfahren, wie Android das Licht der Welt erblickte. Wir werden uns seiner Geschichte zuwenden, um einen besseren Blick auf seine Zukunft zu entwickeln. Android steht vor einem entscheidenden Jahr; deshalb wollen wir uns die Führungsspieler des Ökosystems und ihre Triebkräfte ansehen und die Stärken und Schwächen ausloten, die sie ins Ensemble einbringen.

Wenn wir dieses Kapitel abgeschlossen haben, sollten Sie die Mitspieler bei Android kennen und einen Überblick über Geschichte, Gegenwart und die mögliche Zukunft der Android-Welt gewonnen haben.

## Was ist Android?

Android ist eine umfassende Open Source-Plattform für Mobilgeräte. Es wird von Google protegiert, ist aber Eigentum der Open Handset Alliance (<http://www.openhandsetalliance.com/>). Das Ziel dieser Gruppe ist »eine beschleunigte Innovation im Mobilbereich, um dem Benutzer ein reicheres, kostengünstigeres und besseres Mobilerlebnis zu bieten«. Android ist das Mittel, mit dem das erreicht werden soll.

Also revolutioniert Android die Welt der Mobilgeräte. Erstmals gibt es eine wirklich offene Plattform, die die Hardware von der Software trennt, die darauf läuft. Die Folge ist, dass die gleichen Anwendungen auf einer großen Anzahl an Geräten ausgeführt werden kann, was das Ökosystem für Entwickler und Konsumenten gleichermaßen bereichert.

Zerlegen wir einige der relevanten Schlagwörter und schauen wir uns an, was sie zu sagen haben.

## Was »umfassend« bedeutet

Android ist eine umfassende Plattform. Das bedeutet, dass es eine vollständige Softwareausstattung für Mobilgeräte darstellt.

Entwicklern bietet Android alle Werkzeuge und Frameworks, die für die schnelle und einfache Entwicklung von Mobilanwendungen erforderlich sind. Für den Einstieg in die An-

droid-Entwicklung brauchen Sie nur das Android-SDK. Sie brauchen nicht einmal ein richtiges Android-Gerät.

Beim Benutzer funktioniert Android bereits im Auslieferungszustand. Zusätzlich können die Anwender ihr Gerät aber auch grundlegend anpassen.

Herstellern bietet Android eine vollständige Lösung für den Betrieb auf ihren Geräten. Sieht man von einigen hardwarespezifischen Treibern ab, bringt Android alles mit, was man braucht, um darunter ein Gerät in Betrieb zu nehmen.

## Open Source-Plattform

Android ist eine Open Source-Plattform. Der gesamte Software-Stack, von den elementaren Linux-Modulen bis zu den nativen Bibliotheken, vom Anwendungs-Framework bis zu vollständigen Anwendungen, ist absolut offen.

Außerdem steht Android unter unternehmensfreundlichen Lizenzen (Apache/MIT) und kann von anderen deswegen beliebig erweitert und zu einer Vielzahl von Zwecken verwendet werden. Es wurden sogar einige externe Bibliotheken, die in den Android-Software-Stack eingetacht wurden, neu geschrieben und unter einer neuen Lizenz veröffentlicht.

Sie haben als Entwickler also Zugriff auf den Quellcode der gesamten Plattform. Das ermöglicht Ihnen, sich anzusehen, wie das Android-Betriebssystem im Inneren funktioniert. Als Hersteller können Sie Android leicht auf Ihre spezifischen Hardwareanforderungen anpassen. Sie können sogar Ihre eigenen proprietären und geheimen Zutaten hineinmischen, ohne dass Sie diese an die Entwicklungsgemeinschaft zurückgeben müssen, wenn Sie das nicht wollen.

Android muss nicht lizenziert werden. Sie können es einfach nutzen und modifizieren, es gibt keine irgendwie gearteten Haken. Außerdem bietet Android auf unterschiedlichen Ebenen der Plattform viele Punkte, in die sich jeder einklinken kann, um es auf ungeahnte Weise zu erweitern.



Es gibt einige kleinere elementare Codeteile, die den verschiedenen Herstellern gehören, beispielsweise der Software-Stack für die Netzwerk-, WLAN- und Bluetooth-Schnittstellen. Android gibt sich große Mühe, diesen Komponenten abstrakte Schnittstellen vorzulagern, damit sich herstellerspezifischer Code leichter verwalten lässt.

## Speziell für Mobilgeräte

Android ist eine speziell für Mobilgeräte entwickelte Plattform. Bei der Entwicklung von Android prüfte das Entwicklungsteam, welche Einschränkungen mobiler Geräte sich in naher Zukunft wahrscheinlich nicht ändern werden. Eine davon ist, dass Mobilgeräte batteriebetrieben sind und sich die Leistung von Batterien vermutlich so bald nicht deutlich verbessern wird. Auch bedeutet die kleine Größe von Mobilgeräten, dass Speicherplatz und Geschwindigkeit immer beschränkt sein werden.

Diese Einschränkungen wurden von Anfang an mitbedacht und werden von der Plattform auf allen Ebenen berücksichtigt. Das Ergebnis ist ein in allen Aspekten besseres Benutzererlebnis.

Android wurde so entworfen, dass es auf Geräten unterschiedlicher Art laufen kann. Es setzt keine bestimmte Bildschirmgröße oder -auflösung, keinen Chipsatz und so weiter voraus. Der Kern wurde portabel entworfen.

## Geschichte

Die Geschichte von Android ist interessant und bietet einige Ausblicke auf das, was die Zukunft bringen könnte.

Das sind die Schlüsselereignisse der letzten paar Jahre:

- 2005 kauft Google Android, Inc. Alle erwarten, dass es bald ein »gPhone« geben wird.
- Dann wird es eine Zeit lang still.
- 2007 wird die Open Handset Alliance angekündigt. Android wird offiziell Open Source.
- 2008 wird das Android-SDK 1.0 veröffentlicht. Das von HTC hergestellte und vom Netzbetreiber T-Mobile USA vertriebene G1-Handy folgt kurz darauf.
- 2009 sieht eine starke Zunahme von Android-basierten Geräten. Neue Versionen des Betriebssystems werden veröffentlicht: Cupcake (1.5), Donut (1.6) und Eclair (2.0 und 2.1). Mehr als 20 Geräte nutzen Android.
- 2010 ist Android die sich nach Blackberry am zweitbesten verkaufende Smartphone-Plattform. Froyo (Android 2.2) wird veröffentlicht. Es sind mehr als 60 Geräte auf dem Markt, die Android nutzen.
- 2011 wird Gingerbread (2.3) veröffentlicht und mit Honeycomb (3.0 und 3.1) der Schritt zum Tablet-PC vollzogen. Android-Geräte erobern den ersten Rang auf dem Smartphone-Markt. Die ersten Tablet-PCs mit Honeycomb erscheinen, Dutzende werden angekündigt.

Als Google 2005 Android, Inc. erwarb, dachte die Welt, dass Google im Begriff wäre, den Smartphone-Markt zu betreten: Es gab weitverbreitete Spekulationen über ein Gerät, das auf den Namen gPhone hören sollte.

Googles CEO, Eric Schmidt, machte von Anfang an klar, dass Androids Ambitionen weit über ein einziges Gerät hinausgingen. Stattdessen dachte Google an eine Plattform, die viele Handys und andere Geräte unterstützen kann.

## Googles Motivation

Googles Motivation bei der Unterstützung des Android-Projekts scheint gewesen zu sein, Android-Geräte allgegenwärtig zu machen und damit einen ausgeglichenen Markt für Mobilgeräte zu schaffen. Google ist eigentlich ein Medienunternehmen, und Googles Geschäftsmodell basiert auf dem Verkauf von Werbeflächen. Wenn alle Android nutzen,

kann Google zusätzlich weitere Dienstleistungen anbieten und konkurrenzfähig operieren. Das unterscheidet sich vom Geschäftsmodell anderer Softwarehersteller, die von Lizenzgebühren leben.

Obwohl Google einige proprietäre Apps wie Gmail und Maps lizenziert und mit dem Android-Markt etwas Geld verdient, bleiben die Werbeeinnahmen, die diese Apps bringen, die wesentliche Motivation.

## Open Handset Alliance

Damit Android mehr ist als nur Google, ist es Eigentum der Open Handset Alliance, einer gemeinnützigen Gruppe, zu der sich wichtige Netzbetreiber, Hersteller und andere zusammgefunden haben. Die Alliance will mit Offenheit und Innovation den Umgang mit Mobilgeräten für der Nutzer angenehmer gestalten.

Tatsächlich ist die Allianz jedoch noch recht jung. Viele der Mitglieder lernen gerade erst, wie man miteinander arbeitet. Zurzeit ist Google das Unternehmen, das am meisten in das Android-Projekt investiert.



Die erste Version des Android-SDK wurde veröffentlicht, bevor das erste Gerät auf dem Markt war. Und auch heute brauchen Sie kein Handy, wenn Sie für Android entwickeln wollen. Es gibt einige Ausnahmen (Hardware-sensoren, Telefonfunktionen usw.), aber im Prinzip bietet das Android-SDK alles, was Sie benötigen, um für diese Plattform zu entwickeln.

## Android-Versionen

Wie bei Software üblich, wurde auch Android mit der Zeit verbessert, wie man an den zahlreichen Versionsnummern erkennen kann. Aber die Beziehungen zwischen den unterschiedlichen Versionsnummern können verwirrend sein. Tabelle 1-1 soll eine kleine Orientierungshilfe geben.

Tabelle 1-1: Android-Versionen bis Android 3.0

Android-Version	API-Level	Name
Android 1.0	1	
Android 1.1	2	
Android 1.5	3	Cupcake
Android 1.6	4	Donut
Android 2.0	5	Eclair
Android 2.01	6	Eclair
Android 2.1	7	Eclair
Android 2.2	8	Froyo (Frozen Yogurt)
Android 2.3	9	Gingerbread

Abbildung 1-1: Android-Versionen bis Android 3.0 (Fortsetzung)

Android-Version	API-Level	Name
Android 2.3.3	10	Gingerbread
Android 3.0	11	Honeycomb

Die Android-Versionsnummer allein erklärt die großen und kleinen Versionssprünge der Softwareplattform nur teilweise. Wichtiger ist das API-Level. Versionsnummern ändern sich permanent, manchmal weil sich die APIs geändert haben und manchmal weil kleinere Fehler behoben oder kleinere Leistungsverbesserungen eingebaut wurden.

Als Anwendungsentwickler müssen Sie darauf achten, welches API-Level Ihre App für die Ausführung anvisiert. Das API-Level bestimmt, welche Geräte Ihre App ausführen können und welche nicht.

Normalerweise werden Sie wollen, dass Ihre App auf so vielen Geräten wie eben möglich läuft. Deswegen sollten Sie auch ein API-Level anvisieren, das so klein wie möglich ist. Beachten Sie dabei die Verteilung der Android-Versionen auf den Geräten, die bereits in der Hand von Benutzern sind. Abbildung 1-1 zeigt einen Schnappschuss des Android Device Dashboard (<http://developer.android.com/resources/dashboard/platform-versions.html>) (Mai 2011).

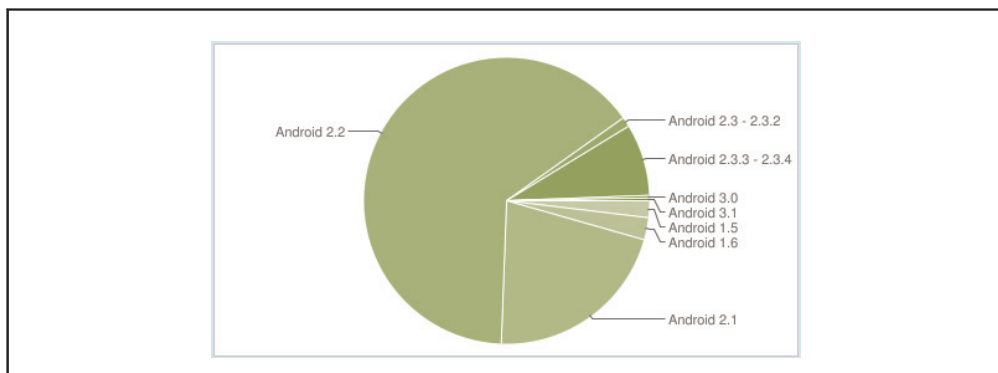


Abbildung 1-1: Verteilung von Android-Versionen im Mai 2011

Wahrscheinlich wird Ihnen aufgefallen sein, dass nicht mehr viele Leute Android 1.5 und 1.6 nutzen. Wahrscheinlich wird Ihnen auch aufgefallen sein, dass noch nicht viele Leute das neueste und beste Android 2.3 haben, die meisten inzwischen aber eine der 2.x-Versionen nutzen. Die Versionen 3.0 und 3.1 sind natürlich noch neuer, spielen aber in einer eigenen Liga, da sie nur für den Einsatz auf Tablet-PCs gedacht sind.

Wie Sie aus dieser Grafik erkennen können, müssen Sie bei der Android-Entwicklung also immer einen Kompromiss eingehen. Wollen Sie die neuen Funktionen der jüngeren Plattform-Versionen einsetzen, müssen Sie in Kauf nehmen, dass Ihre App auf Geräten mit älteren Android-Versionen nicht funktioniert. Wollen Sie, dass Ihre App auf so vielen Geräten

## A

AbsoluteLayout 50  
Activities 28, 39  
adb shell 99  
addPreferencesFromResource()-Methode 88  
afterTextChanged()-Methode 73  
AIDL (Android Interface Definition Language) 219  
Alarm-Service 208, 211  
Alpha-Kanal 77  
Android Device Dashboard 5  
Android Interface Definition Language (AIDL) 219  
Android UI (siehe UI (Benutzerschnittstelle))  
Android Virtual Device (AVD) 23, 98  
Android, Geschichte 3  
Android-Komponenten (siehe Activities, siehe Broadcast-Receiver, siehe Content-Provider, siehe System-Services)  
Android-Projekt 18  
AndroidManifest.xml-Beispiel 61, 165  
anfängliche App-Einrichtung 155  
angehaltener Zustand 30  
Apache Harmony 9  
API  
    Content-Provider 180, 186  
    Level 5, 19, 51  
    Location 198  
    Twitter-kompatibel 58–59, 114, 127  
    und AIDL-Schnittstelle 220  
    Wurzel 83, 85, 95, 99  
APK-Datei (Application Package-Datei) 12, 232, 239  
App-Ressourcen 91  
App-Signierung 12  
Application-Context 35  
Application-Framework 11  
Application-Objekte 102, 114, 134

AppWidgetProvider-Klasse 184  
Architektur 7  
ARGB-Farbkodierung 77  
asInterface()-Methode 228  
AsyncTask 67  
Ausführung mit mehreren Threads 66  
ausgesetzter Zustand 30  
AVD (Android Virtual Device) 23, 98

## B

BaseActivity-Beispiel 155  
Batterieverbrauch 29, 34, 50, 169, 192, 198, 207  
Bausteine, Überblick 27  
Benutzerdatenpartition 98  
Benutzereinstellungen (siehe Einstellungen)  
Benutzername 99  
Benutzerschnittstelle (siehe UI (Benutzerschnittstelle))  
Berechtigungen  
    Broadcasts senden/empfangen 172  
    eigene 172  
    für Nachrichten 173  
    in Manifestdatei deklarieren 61, 172  
    Internet 61  
    Location, fein und grob 201  
Bibliotheken  
    externe nutzen 59  
    native 9, 238  
    Shared-Libraries erstellen 238  
    SQLite als Sammlung von 120  
    verpacken 232  
Bilder hinzufügen 74  
bindService()-Methode 227  
findViewById()-Methode 149, 151–152  
Bionic 9  
BootReceiver 164, 213  
Bornstein, Dan 9

- BroadcastReceiver 34, 39, 163
  - BootReceiver 164, 213
  - NetworkReceiver 169
  - TimelineReceiver 165, 174
- Build-Target 19, 51
- Buttons 48, 52, 91, 108, 210

## C

- C-Implementierung 12, 231–232, 235, 237
- Canvas 196
- close()-Methode 133
- codebasierte Benutzerschnittstellen 48
- Content-Provider
  - erstellen 177
  - Überblick 33
- ContentProvider
  - über Widget nutzen 184
  - Überblick 39
- CRUD-Prinzip 33
- Cursor 122, 143, 182

## D

- .d()-Dringlichkeitsstufe 62
- Dalvik 9
- Dateien speichern 59
- Dateisystem 42, 96
- Datenbanken
  - Constraints 130
  - nutzen 141
  - Schema erstellen 121
  - Überblick 119
- DbHelper 120
- DDMS 17, 63, 96, 99, 128
- Debugging 41
- deklarative Benutzerschnittstellenerstellung 47
- delete()-Methode 122, 177
- doInBackground()-Methode 67
- Dringlichkeitsstufen, Log 62

## E

- .e()-Dringlichkeitsstufe 62
- Eclipse Android Development Tools (ADT) 47
- Eclipse IDE
  - Code bearbeiten in 87–88
  - installieren 16
  - Organize Imports-Werkzeug 62
  - WYSIWYG-Editor 75
- eigene Berechtigungen 172
- Einstellungen
  - anfängliche Einrichtung 155
  - Benutzer führen zu 161

- gemeinsame 94
- in Dateisystem zugreifen 96
- Menüsystem 89
- Positionsdaten 203
- Prefs-Ressource 84
- PrefsActivity-Klasse 87
- Sicherheit, Dateisystem 99
- Überblick 83

- Emulatoren 22
- entfernte Services 220
- entfernter Client 225
- Entwicklungswerkzeuge 17
- Entwurfsstrategie 39
- Events 70
- execSQL()-Methode 122, 124

## F

- Farben ergänzen 74
- feine Positionsdatenberechtigung 201
- fetchStatusUpdates()-Methode 135, 168
- Fibonacci-Demo 233
- format()-Methode 144
- FrameLayout 50

## G

- Garbage Collector 143
- gemeinsame Einstellungen 94
- Geocoder 198–199
- getApplication()-Methode 35, 105, 116
- getApplicationContext()-Methode 35
- getColumnIndex()-Methode 143
- getDefaultSharedPreferences()-Methode 94
- getFilesDir()-Methode 98
- getFriendsTimeline()-Methode 114, 117, 127, 130
- getID()-Methode 177, 181
- getLatestStatusCreatedAtTime()-Methode 134
- getReadableDatabase()-Methode 143
- getRelativeTimeSpanString()-Methode 150, 153
- getService()-Methode 211
- getStatusTextById()-Methode 134
- getStatusUpdates()-Methode 161
- getString()-Methode 95
- getSystemService()-Methode 192, 198, 204, 207
- getTwitter()-Methode 95, 102, 105, 117, 127
- getType()-Methode 177, 182
- getWritableDatabase()-Methode 143
- GNU libc 9
- Google 3
- gravity-Eigenschaft 55
- grobe Positionsdatenberechtigung 201

## H

Hallo Welt-Beispiel 18  
Header-Dateien erstellen 235  
Hexadezimalwerte 77  
Hierarchy-Viewer 81  
Hintergrund, Services laufend in 32  
HTC Sense 34

## I

.i()-Dringlichkeitsstufe 62  
id-Eigenschaft 55  
IDE (Integrated Development Environment) 16  
insert()-Methode 121, 128, 130, 177, 179  
insertOrThrow()-Methode 130  
insertWithOnConflict()-Methode 133  
Integrated Development Environment (IDE) 16  
Intents  
    Broadcast 163, 167  
    Filter 165  
    Menüs integrieren 92  
    Services 208  
    Überblick 31, 219  
    und Menüs 83  
Internet-Zugriff 172  
Internetberechtigung 61  
Internetzugriff 169  
Interprocess Communication (IPC) 219  
interrupt()-Methode 113  
invalidate()-Methode 197  
IPC (Interprocess Communication) 219

## J

Java (siehe auch Eclipse IDE)  
    Ausführung in mehreren Threads 111  
    BaseActivity.java 157, 210  
    Benachrichtigungen 192, 199  
    Bibliotheken für 59  
    Bibliotheken für Android 12  
    BootReceiver.java 164, 213  
    codebasierte Benutzerschnittstelle mit 48  
    Compass.java 194  
    Dalvik-Compiler 10, 12  
    Dateinamenskonventionen 51–52  
    DbHelper.java 123  
    Fehler 59  
    FibActivity.java 239  
    FibLib.java 234  
    gen/com/marakana/R.java 22  
    generiert aus XML 48, 56  
    HalloWelt.java 22

    im Vergleich zu nativem Code 233, 241  
    Klassen 51  
    Klassenpfade 60  
    LogActivity.java 226  
    LogService.java 221  
    mehrere Threads 66  
    Message.java 223  
    NetworkReceiver.java 170  
    Packages 19, 51, 98  
    Parcel 223, 226  
    PrefsActivity.java 87  
    Quellcode 22  
    R-Datei und 21, 91  
    Rose.java 196  
    Schleifen 117, 127  
    StatusActivity.java 56, 58, 67, 71, 109, 204  
    StatusData.java 131  
    synchronisierte Methoden 104  
    TimelineActivity.java 141, 147, 159, 166,  
        175  
    und AIDL 220, 222  
    UpdaterService.java 106, 111, 115, 125, 135,  
        168, 209, 215  
    WhereAml.java 199  
    Widget-ID 55  
    YambaApplication.java 103, 134, 204  
    YambaWidget.java 185  
Java Native Interface (JNI) 231  
javah-Werkzeug 235  
JNI (Java Native Interface) 231  
jtwitter.jar-Bibliothek 59

## K

Klassenpfade 60  
Kompass-Demo 191  
Kompassrosen-Widget 194  
Kompilieren von Code 59  
konfigurationsfreie Datenbank 120

## L

laufender Zustand 30, 192  
Layoutdatei 21  
Layouts und Views 48  
layout\_gravity 55  
layout\_height 54  
layout\_weight 54  
layout\_width 54  
LinearLayout 49  
Linux 7  
Listen und Adapter 43  
Lizenzen 2, 9



- LocationListener 207
- LocationService 198
- log()-Methode 221
- Log-Klasse 62
- LogCat 62, 113
- LogClient-Demo 225
- Logging 62
- LogService-Demo 220

## M

- make-System 238
- makeText()-Methode 69
- Malware 13
- Manager 12
- Manifestdatei
  - Activities registrieren 88, 153
  - Berechtigungen deklarieren in 61, 172
  - BootReceiver registrieren 165
  - ContentProvider registrieren 184
  - LocationListener-Beispiel 201
  - NetworkReceiver registrieren 169
  - Services registrieren 108, 224
  - Überblick 20
  - Widgets registrieren 188
  - YambaApplication 104
- Marshaling 219
- Media Store 33
- Menü
  - Elemente einfügen 108
  - Events 92
  - laden 91
  - Ressourcen 89
- moveToNext()-Methode 143
- Multithreading 41

## N

- Nachrichten 111, 127, 131, 179, 187
  - auf neue prüfen 110
  - Benachrichtigungen über 165, 214
  - Berechtigungen für 173
  - Fenster 53
  - IntentService-basiert 208
  - lokal speichern 119
  - senden 163
  - Widget zur Anzeige 184
- Name/Wert-Paare 83
- Namenskonventionen
  - CamelCase 19
  - Java-Klassen 19, 51
  - Java-Packages 51
  - JNI-Signaturen 236

- Ressourcen 56, 79
- Widgets 55
- native Bibliotheken 9, 12, 231
- Native Development Kit (NDK) 231
- NDK (Native Development Kit) 231
- NetworkReceiver 169
- Netzwerklatenz 41
- Netzwerkverbindungen 61
- Netzwerkverfügbarkeit 169, 172
- NotificationService 214

## O

- Observer-Muster 34, 163
- onAccuracyChanged()-Methode 192, 195
- onBind()-Methode 107, 221
- onClick()-Methode 95
- onCreate()-Methode 56, 88, 94, 106–107, 121, 123, 127, 141
- onCreateOptions()-Methode 158
- onCreateOptionsMenu()-Methode 89, 91
- onDeleted()-Methode 185
- onDestroy()-Methode 106, 108, 113, 141, 143
- onDisabled()-Methode 185
- onDraw() 196
- onEnabled()-Methode 185
- onHandleIntent()-Methode 208
- onLocationChanged()-Methode 201, 208
- onMenuOpened()-Methode 156, 159, 210
- onOptionsItemSelected()-Methode 89, 92, 109, 159, 210
- onPause()-Methode 167, 192, 195, 201
- onPostExecute()-Methode 67, 69
- onProgressUpdate()-Methode 67, 69
- onReceive()-Methode 164, 185, 187
- onResume()-Methode 141, 167, 175, 192, 195
- onSensorChanged()-Methode 192, 196
- onServiceConnected()-Methode 228
- onServiceDisconnected()-Methode 228
- onSharedPreferenceChanged()-Methode 94
- onStart()-Methode 192
- onStartCommand()-Methode 106, 108–109, 113
- onStop()-Methode 192
- onTerminate()-Methode 104
- onUpdate()-Methode 185, 187
- onUpgrade()-Methode 121, 123
- Open Handset Alliance 1, 4
- Open Source-Plattform 2
- OpenGL 9
- OpenJDK 10
- OpenSSL 9
- Optionsmenü 89, 155

Organize Imports-Werkzeug 62  
Override/Implement Methods-Werkzeug 107

## P

Package-Name 19  
Parameter übergeben 219  
Parcel implementieren 223  
Partition für Benutzerdaten 97  
Partitionen 96  
Passwort 99  
PATH-Variable 16  
PendingIntent 211  
Phishing-Angriff 13  
Portabilität 3, 7  
Projektentwurf 39  
Publisher 163  
Publisher/Subscriber-Muster 163  
putExtra()-Methode 168

## Q

QEMU 23  
query()-Methode 122, 143, 177, 182

## R

R-Datei 21  
Refactoring 39, 94, 105, 131, 191  
Registrieren 94, 96, 153, 165, 192, 224  
RelativeLayout 50  
requery()-Methode 166  
requestLocationUpdates()-Methode 201  
res/layout-Ordner 53  
Ressourcen 12, 79, 91  
RGB-Farbsatz 77  
run()-Methode 113, 135

## S

Schema, Datenbank 121  
Schleifen im Service 110  
Schmidt, Eric 3  
Schnittstellen 220  
SDCard-Partition 97  
SDK (Software Development Kit) 15, 19, 23, 91  
sendBroadcast()-Methode 169, 174, 211  
sendTimelineNotification()-Methode 216  
SensorManager 192  
Service-An/Aus-Schalter 156  
Services (siehe System-ervices)  
setContentview()-Methode 53, 57  
setDirection()-Methode 197  
setInexactRepeating()-Methode 214  
setMyLocation()-Methode 208

Settings-Provider 33  
setupList()-Methode 161  
setViewBinder()-Methode 152  
setViewValue()-Methode 152  
Sicherheit 8, 61, 99, 120, 122, 172, 231  
Signieren von Apps 12  
Simulatoren 23  
Single-Thread 65  
Software Development Kit (SDK) (siehe SDK  
(Software Development Kit))  
Spyware 13  
SQL-Injektion 122  
SQLite 9, 43, 119  
sqlite3-Werkzeug 128  
startActivity()-Methode 92, 155, 211  
startender Zustand 29  
startManagingCursor()-Methode 143, 182  
startService()-Methode 109, 211  
status.xml 52  
StatusActivity 52, 56, 67, 91, 105, 204  
stopService()-Methode 109, 170  
String-Ressource 55  
Stub()-Methode 222, 228  
Subscriber 163  
System-Services  
    allgemeine Schritte bei der Verwendung 192  
    Android-Services vs. native Services 32  
    erstellen 106  
    in Manifestdatei definieren 108  
    Intents 208  
    Kompass-Demo 191  
    LocationService 198  
    Menüs einfügen und verarbeiten 108  
    NotificationService 214  
    Schleifen 110  
    testen 110, 113  
    Überblick 101, 191  
    und Projektentwurf 39

## T

TableLayout 50  
TableRow 50  
Testen 110, 113, 117, 228  
text-Eigenschaft 55  
TextWatcher 70  
Thread.sleep()-Methode 110, 113  
Threading 64  
TimelineActivity-Beispiel 139, 159, 165  
TimelineAdapter-Beispiel 149  
TimelineReceiver 165, 174

## U

- Überschreiben/Implementieren von Methoden 93
- UI (Benutzerschnittstelle) 40
  - Android-Objekte 57
  - optimieren 80
  - zwei Verfahren zur Erstellung 47
- umfassende Plattform 1, 37
- Uniform Resource Identifier (URI) 178
- Unmarshaling 219
- update()-Methode 122, 177
- updateAppWidget()-Methode 187
- UpdaterService 106, 115, 125, 164–165, 167, 173
- updateStatus()-Methode 65
- URI (Uniform Resource Identifier) 178

## V

- Verpacken von Bibliotheken 232
- Vertrieb von Apps 13
- ViewBinder 152
- Views und Layouts 48
- Viruse 13
- vorbereitete Anweisungen 128

## W

- .w()-Dringlichkeitsstufe 62
- .wtf()-Dringlichkeitsstufe 62
- Webkit 9
- Widgets
  - Android-UI-Widgets vs. App-Widgets 49
  - App-Widgets 184
  - Auswahl 139
  - Auswahl/Ansicht 198
  - Content-Provider über 184
  - Kompassrosen-Beispiel 194
  - und UI-Trägheit 80
  - wichtige Eigenschaften von 54
- withAppendedID()-Methode 180
- Wo bin ich?-Demo 198
- writeToParcel()-Methode 224
- WYSIWYG-Editor 75

## X

- XML
  - android:-Schlüsselwort 91
  - AndroidManifest.xml 20, 61, 89, 154, 165, 171, 201
  - anzeigen 61
  - deklarative Benutzerschnittstelle mit 47

- direkt modifizieren 76
- Eclipse, Dateien benennen/umbenennen 52, 79, 84
- für Einstellungsressourcen 86
- generieren zu Java 48, 56
- Intent-Filter 153
- Layout-Code 146, 187
- Layoutcode 21
- main.xml 52
- Manifestdatei 88, 108, 184, 188
- menu.xml 108
- Menüressource 89
- Möglichkeiten der Bearbeitung 88
- prefs.xml 212
- res/layout/main.xml 21, 198
- res/layout/row.xml 146
- res/layout/status.xml 78
- res/layout/status2.xml 70
- res/layout/timeline.xml 145
- res/layout/timeline\_basic.xml 140
- res/layout/yamba\_widget.xml 187
- res/menu/menu.xml 91, 157
- res/values/strings.xml 21, 56, 93
- res/xml/menu.xml 210
- res/xml/prefs.xml 86, 203
- res/xml/yamba\_widget\_info.xml 188
- Strings 21
- strings.xml 155, 212
- Strukturansicht 90
- visualisieren 53

## Y

- Yamba
  - 140-Zeichen-Zähler 70
  - Anfänge 51
  - Application-Objekt 102
  - Beispiel-App 27, 35
  - Daten abrufen von 106, 114, 121, 127
  - Datenbank öffnen 123
  - Location-Service integrieren 202
  - Twitter-kompatible Apps erstellen 56, 67, 71, 94, 139
  - Überblick 37
  - und Twitter 37
- YambaWidget-Klasse 184

## Z

- Zeichnungen ergänzen 74
- zerstörter Zustand 30, 14