Kapitel 1
Hallo Android

Inhalt

- Hintergrundinformationen zu Android
- Warum Sie Android-Apps entwickeln sollten
- Einführung in das Android SDK und das Entwicklungsgerüst

Unabhängig davon, ob Sie nun ein erfahrener Konstrukteur mobiler Anwendungen, ein Desktop- oder Webentwickler oder ein absoluter Programmierneuling sind, Android stellt für jeden eine neue, aufregende Möglichkeit dar, innovative Anwendungen für eine immer größer werdende Zahl von Geräten zu schreiben.

Trotz des Namens hilft Android nicht dabei, eine unaufhaltbare Armee gefühlloser Robotkrieger zu erstellen, die sich auf einem unbarmherzigen Feldzug befinden, um die Erde von der Geißel Mensch zu befreien. Stattdessen handelt es sich bei Android um eine umfangreiche Open-Source-Software, die aus einem Betriebssystem, der Middleware und zentralen mobilen Anwendungen und einem Satz an API-Bibliotheken besteht, mit denen Sie Anwendungen schreiben können, die in der Lage sind, das Aussehen, das Verhalten und die Funktionalität von Geräten zu verbessern, auf denen diese Anwendungen dann laufen.

Kleine, modische und vielseitige moderne mobile Geräte sind zu leistungsstarken Werkzeugen geworden, die Touchscreens, Kameras, Geräte für die Medienwiedergabe, Empfänger für das Global Positioning System (GPS) und Hardware für die Near Field Communication (NFC) enthalten. Und parallel zur Weiterentwicklung der Technologie wurden aus Handys viel mehr als Geräte, mit denen nur Anrufe getätigt werden können. Die Einführung von Tablets und Google TV hat dazu geführt, dass sich Android weit über seine Ursprünge hinaus als Betriebssystem für Mobiltelefone etabliert hat und eine

einheitliche Plattform für die Entwicklung von Anwendungen für eine fast unüberschaubar große Vielfalt an Hardware bietet.

In Android werden systemeigene Anwendungen und Anwendungen Dritter mit denselben APIs geschrieben und in derselben Laufzeitumgebung ausgeführt. Diese APIs enthalten den Zugriff auf Hardware, das Aufzeichnen von Videos, standortbasierende Dienste, die Unterstützung im Hintergrund laufender Services, auf Landkarten basierende Dienste, relationale Datenbanken, Kommunikation zwischen Anwendungen, Bluetooth, NFC und die Anzeige von 2D- und 3D-Grafiken.

Dieses Buch beschreibt, wie Sie mit diesen APIs eigene Android-Anwendungen erstellen können. Sie lernen in diesem Kapitel Vorgehensweisen kennen, die Ihnen bei der Entwicklung für mobile und Android-fähige Hardware helfen, und erhalten eine Einführung in einige der für das Entwickeln mit Android zentralen Funktionen der Plattform.

Android kennt leistungsstarke APIs, eine ausgezeichnete Dokumentation, eine blühende Entwicklergemeinde und keine Kosten für Entwicklung und Vertrieb. Und während mobile Geräte immer beliebter werden und sich Android-Geräte rasant weiterentwickeln, haben Sie die Möglichkeit, bahnbrechende Anwendungen zu erstellen, und zwar unabhängig davon, über welche Kenntnisse Sie als Entwickler verfügen.

1.1 Ein wenig Hintergrundwissen

In den Tagen vor Twitter und Facebook, als Google nur als Idee im Kopf seines Gründers existierte und Dinosaurier über die Erde streiften, war ein mobiles Telefon dies: ein tragbares Telefon, das klein genug war, um in eine Jackentasche zu passen, mit Batterien, die für maximal sieben Stunden reichten. Aber diese Geräte sorgten für die Freiheit, Telefonate führen zu können, ohne an einer Leitung hängen zu müssen.

Heutzutage werden Mobiltelefone immer kleiner, moderner und leistungsfähiger und sie sind zu einem allgegenwärtigen, unverzichtbaren Bestandteil des Lebens geworden. Die Entwicklung der Hardware hat dafür gesorgt, dass mobile Geräte immer kleiner und wirkungsvoller geworden sind, während ihre Bildschirme immer größer und klarer werden und die Zahl der Peripheriegeräte immer weiter ansteigt.

Irgendwann erhielten mobile Geräte Kameras und Medienplayer und seitdem sind sie um GPS-Empfänger, Akzelerometer (Beschleunigungssensoren), NFC-Hardware und hochauflösende Touchscreens erweitert worden. Diese Neuerungen liefern einen fruchtbaren Nährboden für Softwareentwickler, aber bis heute hinkt die Software, die für mobile Geräte vorhanden ist, ständig den Weiterentwicklungen der Hardware hinterher.

1.1.1 Die noch nicht allzu ferne Vergangenheit

Früher mussten Entwickler, die C oder C++ verwendeten, die Hardware genau kennen, für die sie codierten. Bei dieser Hardware handelte es sich normalerweise um ein

einzelnes Gerät, bestenfalls um die Gerätegruppe eines einzelnen Herstellers. Als sich die Hardwaretechnologie und der mobile Zugriff auf das Internet weiterentwickelten, musste langsam, aber sicher vollständig auf diese Vorgehensweise verzichtet werden.

Es wurden dann Plattformen wie Symbian entwickelt, um Entwicklern ein breiteres Zielpublikum zu bieten. Diese Systeme schafften es erfolgreich, Entwickler mobiler Anwendungen zu ermutigen, sich um reichhaltige Anwendungen zu kümmern, die in der Lage waren, die verfügbare Hardware besser zu nutzen.

Auch wenn diese Plattformen den Zugriff auf die Gerätehardware boten – und immer noch bieten –, so verlangten sie von den Entwicklern, dass komplexer C/C++-Code geschrieben wurde, und sie griffen sehr stark auf systemeigene APIs zu, die fast immer nur sehr schwer zu handhaben waren. Dieses Problem wurde dann noch komplizierter, als es um Anwendungen ging, die auf unterschiedlicher Hardware laufen sollten. Und Anwendungen, die bestimmte Hardwarefunktionen wie GPS nutzen, konnten Entwicklern den letzten Nerv rauben.

In den letzten Jahren sah der größte Fortschritt bei der Entwicklung von Software für mobile Geräte so aus, dass durch Java gehostete MIDlets eingeführt wurden, die auf einer Java Virtual Machine (JVM) ausgeführt werden: Dabei handelt es sich um einen Vorgang, bei dem die zugrunde liegende Hardware abstrahiert wird und den Entwicklern die Möglichkeit gibt, Anwendungen zu schreiben, die auf den vielen Geräten laufen, die die Java-Laufzeitumgebung unterstützen. Unglücklicherweise gibt es diese Bequemlichkeit nur zu dem Preis, dass der Zugriff auf die Hardware des Gerätes stark eingeschränkt war.

Bei der Entwicklung für mobile Geräte (was zukünftig nur noch *mobile Entwicklung* genannt wird) galt es lange Zeit als normal, dass sich bei Anwendungen Dritter der Zugriff auf die Hardware und die Ausführungsrechte von denen unterschieden, die für systemeigene Anwendungen galten, wobei MIDlets häufig nicht alles an Rechten erhielten, was sie eigentlich zum Funktionieren benötigten.

Die Einführung der Java-MIDlets hat zwar dazu geführt, dass Entwickler mehr Publikum bekamen, aber das Fehlen von Hardwarezugriffen auf niedriger Ebene und einer geschützten Programmausführung führte dazu, dass es sich bei den meisten mobilen Anwendungen um Desktopprogramme oder Websites handelte, die so umgebaut worden waren, dass sie auf den kleineren Bildschirmen wiedergegeben werden konnten. Sie waren in der Regel nicht in der Lage, die Mobilität der Geräteplattform zu nutzen.

1.1.2 Ein Ausblick auf die Zukunft

Android gehört zu einer Reihe moderner mobiler Betriebssysteme, die entworfen worden sind, um die Entwicklung von Anwendungen auf immer leistungsfähigeren mobilen Geräten zu unterstützen. Auch Plattformen wie Microsofts Windows Phone und Apples iPhone sorgen für eine reichhaltige und einfach funktionierende Entwicklungsumgebung für mobile Anwendungen, aber sie bauen, anders als Android, auf proprietären Betriebssystemen auf. Und in einigen Fällen werden systemeigene Anwendungen stark bevorzugt, indem die Kommunikation zwischen Anwendungen und den internen Daten behindert

wird und der Vertrieb von Anwendungen Dritter auf den Plattformen Einschränkungen und einer strikten Kontrolle unterliegt.

Android sorgt bei mobilen Anwendungen für neue Möglichkeiten, indem es eine offene Entwicklungsumgebung anbietet, die auf einen Open-Source-Linux-Kernel aufbaut. Der Zugriff auf die Hardware wird über eine Reihe von API-Bibliotheken ermöglicht, und die Interaktion mit der Anwendung wird – vollständig kontrollierbar – unterstützt.

Unter Android haben alle Anwendungen die gleiche Wertigkeit. Android-Anwendungen Dritter und die zum System gehörenden Anwendungen werden mit denselben APIs geschrieben und in derselben Laufzeitumgebung ausgeführt. Benutzer können alle zum System gehörenden Anwendungen entfernen oder durch Alternativen Dritter ersetzen. Das gilt selbst für den Startbildschirm.

1.2 Was Android nicht ist

Android wurde zunächst nur als störender Eindringling auf der Spielwiese Erwachsener angesehen. Deshalb fällt es nicht schwer zu verstehen, warum es einige Verwirrung darüber gab, was Android wirklich ist. Android ist *nichts* hiervon:

- **Eine Implementierung von Java ME** – Android-Anwendungen werden in der Sprache Java geschrieben, aber sie laufen nicht innerhalb der Java ME (Mobile Edition) VM, und über Java kompilierte Klassen und ausführbare Dateien laufen nicht *per se* in Android.
- **Bestandteil des Linux Phone Standards Forums (LiPS) oder der Open Mobile Alliance (OMA)** – Android läuft auf einem Open-Source-Linux-Kernel, aber auch wenn sich die Ziele ähneln, so geht doch der Android-Ansatz eines vollständigen Softwarestapels weiter als das, worauf sich die Organisationen konzentrieren, die diese Standards festlegen.
- **Nur eine Anwendungsebene (wie UIQ oder S60)** – Obwohl auch Android eine Anwendungsebene enthält, beschreibt »Android« auch den gesamten Softwarestapel einschließlich des ihm zugrunde liegenden Betriebssystems, der API-Bibliotheken und der Anwendungen selbst.
- **Ein mobiles Telefon** – Android enthält zwar für die Hersteller mobiler Telefone eine Referenz für das Design von Telefonen, aber *das* Android-Telefon existiert nicht.
- **Googles Antwort auf das iPhone** – Beim iPhone handelt es sich um eine firmenspezifische Hard- und Softwareplattform, die von einer einzigen Firma (Apple) herausgegeben wird, während Android ein Open-Source-Softwarestapel ist, der von der Open Handset Alliance (OHA entwickelt und unterstützt wird, um auf beliebigen, kompatiblen Geräten zu laufen.

1.3 Android: Eine offene Plattform für die mobile Entwicklung

Andy Rubin von Google beschreibt Android so:

> *Die erste echte und umfassende Plattform für mobile Geräte. Sie enthält ein Betriebssystem, eine Benutzeroberfläche und Anwendungen – also die gesamte Software, damit ein mobiles Telefon funktioniert, aber ohne die proprietären Hürden, die bisher mobile Innovationen behindert haben. (*Where's My Gphone? – *http://googleblog.blogspot.com/2007/11/)*

Inzwischen hat sich Android zu mehr als einer reinen Plattform für Handys gemausert und bietet eine Entwicklungsumgebung für einen immer größer werdenden Kreis von Geräten, zu dem auch Tablets und Fernsehgeräte gehören. Man kann eigentlich sagen, dass Android ein Ökosystem ist, das aus drei Komponenten besteht:

- Einem kostenlosen Open-Source-Betriebssystem für unterstützte Geräte
- Einer Open-Source-Entwicklungsumgebung, um Anwendungen zu erstellen
- Geräten, insbesondere Mobiltelefone, die das Betriebssystem Android und die dafür erstellten Anwendungen verwenden

Natürlich besteht Android aus diversen voneinander abhängigen Teilen, zu denen auch die folgenden gehören:

- Ein Compatibility Definition Document (CDD) und eine Compatibility Test Suite (CTS), die die Anforderungen an ein Gerät beschreiben, damit es den Softwarestapel unterstützt
- Ein Linux-Betriebssystemkernel, der eine einfache Schnittstelle zur Hardware, dem Arbeitsspeicher und für die Verwaltung zur Verfügung stellt und der für die entsprechenden Kontrollen sorgt, wobei alles für unterstützte mobile Geräte optimiert worden ist
- Open-Source-Bibliotheken für die Anwendungsentwicklung, einschließlich SQLite, WebKit, OpenGL und einem Medienmanager
- Eine Laufzeitumgebung, die verwendet wird, um Android-Anwendungen auszuführen und aufzunehmen, und die die Dalvik Virtual Machine (VM und die zentralen Bibliotheken enthält, die Android-spezifische Funktionen bereitstellen. Die Laufzeitumgebung ist so klein und effizient entworfen worden, dass sie auf mobilen Geräten verwendet werden kann.
- Ein Framework für Anwendungen, das für die Anwendungsebene Services des Systems offenlegt, zu denen die Fenster- und die Standortverwaltung, Datenbanken, die Möglichkeit, zu telefonieren, und Sensoren gehören
- Ein Framework für die Benutzeroberfläche, das verwendet wird, um Anwendungen aufzunehmen und auszuführen

- Ein Satz zentraler, vorinstallierter Anwendungen
- Ein Software Development Kit (SDK), das verwendet wird, um Anwendungen zu erstellen, und das über die entsprechenden Plug-ins und Dokumentationen verfügt

Was Android letztendlich so unwiderstehlich macht, ist seine offene Philosophie. Die erlaubt es nämlich, dass Sie alle Defizite der Benutzeroberflächen oder am Design der integrierten Anwendungen beheben können, indem Sie eine Erweiterung oder eine Ersatzanwendung schreiben. Android gibt Ihnen, dem Entwickler, die Möglichkeit, Oberflächen und Anwendungen für mobile Geräte zu erstellen, die dann so aussehen und sich so verhalten, wie Sie es sich vorgestellt haben.

1.4 Systemeigene Android-Anwendungen

Android-Geräte werden normalerweise mit einer Reihe vorinstallierter Anwendungen ausgeliefert. Diese bilden einen Teil des Android Open Source Projects (AOSP). Zu diesen Anwendungen gehören unter anderem auch:

- Ein E-Mail-Client
- Eine Anwendung zum Verwalten von SMS-Nachrichten
- Eine komplette Suite für die Verwaltung persönlicher Daten, einschließlich eines Kalenders und einer Kontaktverwaltung
- Ein auf dem WebKit basierender Webbrowser
- Eine Anwendung für die Wiedergabe von Musik
- Eine Anwendung zum Aufzeichnen von Fotos und Videos
- Ein Taschenrechner
- Ein Startbildschirm
- Ein Wecker

In vielen Fällen werden Android-Geräte auch mit diesen Google-eigenen mobilen Anwendungen ausgeliefert:

- Der Google Play Store zum Herunterladen von Android-Anwendungen Dritter
- Ein mit allen Funktionen versehenes Google Maps, zu dem auch ein Routenplaner, die Satellitenansicht und ein Überblick über die Verkehrslage gehören
- Der E-Mail-Client Gmail
- Der Videoplayer YouTube

Die Daten, zum Beispiel Einzelheiten zu den Kontakten, die von vielen dieser systemeigenen (oder *nativen*) Anwendungen gespeichert und verwendet werden, stehen auch den Anwendungen Dritter zur Verfügung. Ähnliches gilt auch für Ereignisse wie eingehende Anrufe, auf die auch Ihre Anwendungen reagieren können. Mit ziemlicher Sicher-

heit sehen die Anwendungen auf neuen Android-Geräten nicht genau so aus, wie es oben beschrieben wird, weil das Erscheinungsbild letztendlich vom Hersteller der Hardware und/oder des Anbieters Ihres Telefondienstes abhängt.

Android ist Open Source, was bedeutet, dass Anbieter und OEMs die Oberfläche und die Anwendungen, die zusammen mit einem Android-Gerät ausgeliefert werden, beliebig anpassen können. Etliche Anbieter haben dies auch getan, zum Beispiel HTC mit *Sense*, Motorola mit *MotoBlur* und Samsung mit *TouchWiz*. Es ist wichtig, sich immer daran zu erinnern, dass die allem zugrunde liegende Plattform und das SDK für alle kompatiblen Geräte bei allen OEMs und Dienstanbietern identisch sind. Das Aussehen und die Bedienung der Benutzeroberflächen können unterschiedlich sein, aber Ihre Anwendungen funktionieren auf allen kompatiblen Android-Geräten gleich.

1.5 Funktionen des Android SDKs

Die Attraktivität von Android als Entwicklungsumgebung basiert auf seinen APIs. Android ist eine anwendungsunabhängige Plattform. Sie ermöglicht es Ihnen deshalb, Anwendungen zu erstellen, die zu einem so festen Bestandteil des Gerätes wie die Anwendungen werden, die bei der Auslieferung bereits vorhanden sind. Die folgende Liste hebt einige der bemerkenswertesten Android-Funktionen hervor:

- GSM-, EDGE-, 3G-, 4G- und LTE-Netzwerke zum Telefonieren und für den Datenaustausch ermöglichen es Ihnen, Anrufe zu tätigen und SMS-Nachrichten zu empfangen und zu versenden oder Daten über Mobilfunknetze zu übertragen und zu empfangen.
- Umfangreiche APIs für standortbasierende Dienste wie GPS und netzwerkbasierende Standorterkennung
- Volle Unterstützung von Anwendungen, die Elemente von Landkarten als Teil ihrer Oberfläche verwenden
- Zugriff auf Wi-Fi-Hardware und Peer-to-Peer-Verbindungen
- Volle Kontrolle von Multimediahardware einschließlich der Möglichkeit, mit der Kamera aufzunehmen und wiederzugeben und dabei auch das Mikrofon zu benutzen
- Medienbibliotheken für die Wiedergabe und das Aufnehmen von Audio/Video oder Standbildern in vielen Formaten
- APIs, um Sensor-Hardware wie Akzelerometer, Kompasse und Barometer zu nutzen
- Bibliotheken, um Bluetooth- und NFC-Hardware für einen Peer-to-Peer-Datenaustausch nutzen zu können
- IPC-Nachrichtenweitergabe
- Gemeinsam genutzte Datenablagen und APIs für Kontakte, soziale Netzwerke, Kalender und Multimedia

- Im Hintergrund arbeitende Dienste, Anwendungen und Prozesse
- Widgets und Live Wallpapers für den Startbildschirm
- Die Möglichkeit, Suchergebnisse von Anwendungen in die Systemsuche einzubinden
- Einen auf dem Open-Source-HTML5-WebKit basierenden Webbrowser
- Für mobile Geräte optimierte, mit Hardwarebeschleunigung arbeitende Grafik, einschließlich einer pfadbasierenden 2D-Grafikbibliothek und der Unterstützung von 3D-Grafik durch OpenGL ES 2.0
- Anpassung an die Landessprache durch ein dynamisches Programmiergerüst
- Ein Anwendungsframework, das die wiederholte Benutzung von Anwendungskomponenten und das Ersetzen systemeigener Anwendungen ermöglicht

1.5.1 Zugriff auf Hardware, einschließlich Kamera, GPS und Sensoren

Android enthält API-Bibliotheken, um das Entwickeln – zum Beispiel Zugriffe auf die verwendete Hardware – zu vereinfachen. Dies sorgt dafür, dass Sie Ihre Software nicht an die einzelnen Geräte anpassen müssen und somit Android-Anwendungen erstellen können, die auf jedem Gerät funktionieren, das den Android-Softwarestapel unterstützt. Zum Android SDK gehören APIs für standortbasierende Hardware (wie GPS), die Kamera, Audio, Netzwerkverbindungen, Wi-Fi, Bluetooth, Sensoren (einschließlich Akzelerometer), NFC, der Touchscreen und die Energieverwaltung. Sie können in den Kapiteln 12 und 15 bis 17 mehr über die Möglichkeiten erfahren, die einige der Android-APIs bieten.

1.5.2 Datenaustausch mit Wi-Fi, Bluetooth und NFC

Android bietet eine vielseitige Unterstützung für den Datenaustausch zwischen Geräten an. Hierzu gehören Bluetooth, Wi-Fi Direct und Android Beam. Diese Technologien bieten eine Vielzahl von Möglichkeiten, um Daten – abhängig von den verwendeten Geräten – gemeinsam zu nutzen. Dadurch sind Sie in der Lage, Anwendungen zu schreiben, die sich genau darauf spezialisieren. Außerdem kennt Android APIs, um Netzwerk- und Bluetooth-Verbindungen zu verwalten und NFC-Tags zu lesen. Sie finden in Kapitel 16, *Bluetooth, NFC, Netzwerke und Wi-Fi*, Einzelheiten zur Nutzung der Android-APIs für die Kommunikation.

1.5.3 Maps, Geocoding und standortbasierende Dienste

Die vorhandene Unterstützung für Landkarten (englisch *Maps*) ermöglicht es Ihnen, auf Landkarten basierende Anwendungen zu erstellen, die sich die Mobilität von Android-Geräten zunutze machen. Sie können Oberflächen entwerfen, zu denen auch ein interaktives Google Maps gehört, das Sie programmtechnisch steuern und mit Anmerkungen versehen können, indem Sie die reichhaltige grafische Bibliothek von Android verwenden.

Die standortbasierenden Dienste von Android bedienen sich technischer Errungenschaften wie GPS und Googles netzwerkbasierender Technologie, um die aktuelle Position des

Gerätes herauszufinden. Diese Dienste sorgen dafür, dass Sie sich weniger um eine Technologie kümmern müssen, die Ihren Standort ausfindig macht, als darum, welche Minimalanforderungen (zum Beispiel bei der Genauigkeit) erfüllt werden sollen. Dies bedeutet auch, dass Ihre standortbasierende Anwendung unabhängig davon funktioniert, welche Technologie das Hostgerät unterstützt.

Um Karten mit Standorten zu verbinden, enthält Android eine API für Geocoding (*forward* und *reverse*), über die Sie auf Landkarten Koordinaten für Adressen und die Adresse einer Kartenposition finden können. (Wie Sie Landkarten, das Geocoding und die standortbasierenden Dienste verwenden können, erfahren Sie in Kapitel 13, *Maps, Geocoding und standortbasierende Dienste*.)

1.5.4 Services für den Hintergrund

Android unterstützt Anwendungen und Services, die so entwickelt wurden, dass sie im Hintergrund ablaufen, während Ihre Anwendung nicht aktiv genutzt wird. (Sie können Services mit Diensten unter Windows vergleichen.)

Moderne mobile Handys und Tablets sind von Haus aus multifunktional, wobei Sie aber berücksichtigen müssen, dass die Größe des Bildschirms und die Art, wie die Geräte agieren, dafür sorgen, dass im Allgemeinen immer nur eine »handelnde« Anwendung gleichzeitig sichtbar ist. Plattformen, die es nicht zulassen, dass Anwendungen im Hintergrund ausgeführt werden, schränken damit die Brauchbarkeit von Anwendungen ein, die eigentlich nicht Ihrer ständigen Aufmerksamkeit bedürfen.

Im Hintergrund laufende Services machen es möglich, unsichtbare Anwendungskomponenten zu erstellen, die für eine automatische Verarbeitung sorgen, ohne dass der Benutzer selbst etwas machen muss. Die Ausführung im Hintergrund lässt es zu, dass Ihre Anwendungen ereignisgesteuert reagieren und regelmäßig aktualisiert werden können. Damit sind Sie zum Beispiel in der Lage, Spielstände oder Marktpreise zu beobachten, standortbasierende Warnungen zu erzeugen und eingehende Anrufe und SMS-Nachrichten vorzusortieren.

Seit jeher werden Benutzer über *Benachrichtigungen* auf Ereignisse aufmerksam gemacht, die in einer im Hintergrund laufenden Anwendung geschehen. Indem Sie den Notification Manager verwenden (*Notification* ist die englische Bezeichnung für *Benachrichtigung*), können Sie Warnungen auslösen, Vibrationen hervorrufen, die LED des Gerätes aufleuchten lassen und Benachrichtigungssymbole in Statusleisten steuern. Wenn Sie mehr über Benachrichtigungen erfahren und alles aus Hintergrunddiensten herausholen wollen, schauen Sie sich die Kapitel 9 und 10 an.

1.5.5 SQLite, die Datenbank zum Speichern und Abfragen von Daten

Für ein Gerät, dessen Speicherkapazität ziemlich begrenzt ist, ist es wichtig, dass Daten schnell und effizient gespeichert und abgefragt werden können. Android sorgt mit SQLite für eine leichtgewichtige relationale Datenbank, die jede Anwendung nutzen kann. Ihre Anwendungen sind damit in der Lage, Daten sicher und effizient abzulegen.

Standardmäßig läuft die Datenbank einer Anwendung in einer geschützten Umgebung – ihr Inhalt steht nur der Anwendung zur Verfügung, die sie angelegt hat. Content Provider kennen nun einen Mechanismus, über den sie nicht nur die anwendungsspezifischen Datenbanken gezielt freigeben können, sondern auch in der Lage sind, zwischen Ihrer Anwendung und der Datenquelle für eine Abstraktion zu sorgen. Datenbanken und Content Provider werden im Einzelnen in Kapitel 8. *Datenbanken und Content Provider*, behandelt.

1.5.6 Gemeinsam genutzte Daten und die Kommunikation zwischen Anwendungen

Android kennt Techniken – in erster Linie Intents und Content Provider –, um die Daten Ihrer Anwendungen auch woanders nutzen zu können.

Intents sorgen für einen Mechanismus, um Nachrichten innerhalb von und unter Anwendungen weiterzuleiten. Indem Sie Intents verwenden, können Sie eine bestimmte Aktion (zum Beispiel eine Telefonnummer wählen oder einen Kontakt bearbeiten) systemweit versenden, damit sie dann von anderen Anwendungen ausgeführt wird. Mit demselben Mechanismus sind Sie in der Lage, Ihre eigene Anwendung so zu registrieren, dass auch sie diese Nachrichten empfangen oder die verlangte Aktion ausführen kann. Bei Intents handelt es sich um zentrale Bestandteile von Android. Sie werden ausführlich in Kapitel 5, *Intents und Broadcast Receiver*, behandelt.

Sie können *Content Provider* verwenden, um für einen sicheren, kontrollierten Zugriff auf die »privaten« Datenbanken von Anwendungen zuzugreifen. Die Datenablagen der systemeigenen Anwendungen wie die der Kontaktverwaltung sind als Content Provider angelegt, was es Ihnen ermöglicht, diese Daten auch über Ihre eigenen Anwendungen auszulesen oder zu bearbeiten. Kapitel 8 geht detailliert auf Content Provider ein. Es deckt nicht nur die systemeigenen Provider ab, sondern zeigt auch, wie Sie selbst Provider erstellen und verwenden können.

1.5.7 Den Startbildschirm durch Widgets und Live Wallpaper erweitern

Widgets und *Live Wallpaper* geben Ihnen die Möglichkeit, dynamische Anwendungskomponenten zu erstellen, die in Ihrer Anwendung ein Fenster erzeugen und direkt auf dem Startbildschirm nützliche Informationen bereitstellen. Benutzer können damit direkt vom Startbildschirm aus mit Ihrer Anwendung interagieren. Das wiederum steigert das Interesse der Benutzer an Ihrer Anwendung, weil sie einen unmittelbaren Zugriff auf Informationen erhalten. Wie Sie Anwendungskomponenten für den Startbildschirm erstellen, erfahren Sie in Kapitel 14, *Den Startbildschirm erobern*.

1.5.8 Umfangreiche Unterstützung von Medien und 2D/3D-Grafiken

Größere Bildschirme und hellere und höher aufgelöste Anzeigegeräte haben dazu geführt, dass es heutzutage mobile Multimediageräte gibt. Damit Sie so viel wie möglich aus der Hardware herausholen können, stellt Android Grafikbibliotheken für zweidimensionales

Malen und dreidimensionale Grafiken (in diesem Fall über OpenGL) zur Verfügung. (2D- und 3D-Grafiken werden im Einzelnen in Kapitel 11, *Fortgeschrittene User Experience*, behandelt.)

Außerdem bietet Android umfangreiche Bibliotheken für den Umgang mit Standbildern, Videos und Audiodateien an, wobei die Formate MPEG4, H.264, HTTP Live Streaming, VP8, WEBP, MP3, AAC, AMR, HLS, JPG, PNG und GIF unterstützt werden. (Auf die Android-Bibliotheken zur Medienverwaltung wird in Kapitel 15, *Audio, Video und die Verwendung der Kamera*, eingegangen.)

1.5.9 Cloud to Device Messaging

Der Android-Service *Cloud to Device Messaging* (C2DM, was auf Deutsch den Datentransfer von der Cloud zum Gerät bezeichnet) sorgt für einen wirkungsvollen Mechanismus, den Entwickler nutzen können, um ereignisgesteuerte Anwendungen zu erstellen, die auf serverseitigen Push-Vorgängen beruhen. Indem Sie C2DM verwenden, sind Sie in der Lage, eine leichtgewichtige, immer aktive Verbindung zwischen Ihrer mobilen Anwendung und Ihrem Server herzustellen, die es Ihnen erlaubt, kleinere Datenmengen in Echtzeit direkt auf Ihr Gerät zu senden.

Der Service C2DM wird normalerweise verwendet, um Anwendungen darüber zu informieren, dass es auf dem Server neue Daten gibt. Dies verringert die Notwendigkeit, regelmäßige Abfragen zu starten, reduziert die Auswirkungen, die eine Programmaktualisierung auf die Batterien des Gerätes hat, und sorgt dafür, dass diese Aktualisierungen zeitnah erfolgen.

1.5.10 Die Verwaltung des Arbeitsspeichers und der Prozesse optimieren

Android verwendet wie Java und .NET eine eigene Laufzeitumgebung und eine eigene VM, um den Arbeitsspeicher von Anwendungen zu verwalten. Aber anders als bei diesen beiden anderen Programmierumgebungen verwaltet die Laufzeitumgebung von Android auch die Lebensdauer von Prozessen. Android sorgt dafür, dass Anwendungen angesprochen werden können, indem Prozesse bei Bedarf angehalten oder »abgeschossen« werden, um Ressourcen für Anwendungen mit einer höheren Priorität freizumachen.

Nebenbei bemerkt, die höchste Priorität erhält immer die Anwendung, mit der sich der Benutzer gerade aktiv beschäftigt. In einer Umgebung, die es nicht zulässt, dass Anwendungen ihre eigene Lebensdauer steuern, ist es wichtig, daran zu denken, dass Ihre Anwendungen zwar immer auf einen schnellen Tod vorbereitet sind, dass sie aber auch weiterhin in der Lage bleiben, zu reagieren, sich zu aktualisieren oder bei Bedarf im Hintergrund neu zu starten. Sie erfahren in Kapitel 3, *Anwendungen und Aktivitys erstellen*, mehr über den Lebenszyklus von Android-Anwendungen.

1.6 Die Open Handset Alliance

Die Open Handset Alliance (OHA) ist ein Konsortium aus mehr als 80 Technologiefirmen, zu denen Hardwarehersteller, Mobilfunkanbieter, Softwareentwickler, Unternehmen der Halbleiterbranche und andere gehören. Von besonderer Bedeutung sind bekannte Firmen der Mobilfunkbranche wie Samsung, Motorola, HTC, T-Mobile, Vodafone, ARM und Qualcomm. Die OHA sagt von sich selbst, dass sie Folgendes darstellt:

> *Eine Verpflichtung für Offenheit, eine gemeinsame Vision für die Zukunft und konkrete Pläne, damit die Visionen Realität werden. Neuerungen beim Mobilfunk sollen beschleunigt vorangetrieben werden, um Verbrauchern eine reichhaltigere, preiswertere und bessere mobile Umgebung zu schaffen*. (*Original: www.openhandsetalliance.com*)

Die OHA hofft, Verbrauchern eine bessere mobile Softwareumgebung bieten zu können, indem sie die Plattform für eine moderne mobile Entwicklung in kürzeren Abständen mit einer immer besseren Qualität liefert und dabei auf Lizenzgebühren für Softwareentwickler und Hersteller von Geräten verzichtet.

1.7 Worauf läuft Android?

Das erste Android-Handy, das T-Mobile G1, ist 2008 auf den Markt gekommen. Bis Anfang 2012 sind mehr als 300 Millionen Android-kompatible Geräte von mehr als 39 Herstellern in mehr als 123 Länder für 231 Mobilfunkanbieter verkauft worden. Android ist kein mobiles Betriebssystem, das nur für eine bestimmte Hardware zur Verfügung steht, sondern es ist entwickelt worden, um eine Vielzahl von Hardwareplattformen zu unterstützen – angefangen bei Smartphones, über Tablets, bis hin zu Fernsehgeräten.

Da es weder Lizenzgebühren noch eine proprietäre Software gibt, sind die Kosten, die Herstellern entstehen, wenn sie Android-Geräte bereitstellen, konkurrenzlos günstig. Viele erwarten nun, dass die Vorteile, die Android als Plattform für leistungsfähige Anwendungen bietet, dazu führen, dass die Hersteller von Geräten ermutigt werden, eine immer breitere Palette maßgeschneiderter Hardware herzustellen.

1.8 Warum für mobile Geräte entwickeln?

Das Aufkommen moderner mobiler Smartphones – multifunktionaler Geräte, mit denen Sie nicht nur telefonieren können, sondern die auch über einen voll funktionsfähigen Webbrowser, Kameras, Medienplayer, Wi-Fi und standortbasierende Dienste verfügen – hat fundamental die Weise geändert, wie die Menschen mit ihren mobilen Geräten umgehen und auf das Internet zugreifen. Die Zahl der Handybesitzer übersteigt in vielen Ländern die Zahl der Computerbesitzer, wobei es weltweit mehr als drei Milliarden

Nutzer von Handys gibt. 2009 kennzeichnet das Jahr, in dem mehr Handy- als Computerbenutzer zum ersten Mal das Internet besucht haben. Viele glauben, dass in den nächsten fünf Jahren mehr Menschen über Handys als über PCs auf das Internet zugreifen werden.

Die immer größer werdende Beliebtheit von Smartphones hat in Verbindung mit der ebenfalls immer größer werdenden Verfügbarkeit von Hochgeschwindigkeits-Hotspots für Mobilfunk- und Wi-Fi-Netze dazu geführt, dass es einen riesigen Markt für innovative mobile Anwendungen gibt. Die Allgegenwart von Handys und unsere Beziehung zu ihnen bewirken, dass es Entwickler hier mit einer grundsätzlich anderen Plattform zu tun haben als beim PC. Ein Handy kann heutzutage mit seinem Mikrofon, der Kamera, dem Touchscreen, der Standorterkennung und den Umgebungssensoren zu einem Gerät werden, das das Wahrnehmungsvermögen erweitert. Anwendungen für Smartphones haben die Art geändert, wie die Menschen ihre Telefone nutzen. Dies gibt Ihnen als Anwendungsentwickler die einzigartige Gelegenheit, dynamische, aufregend neue Anwendungen zu erstellen, die zu einem aktiven Bestandteil des Lebens vieler Menschen werden.

1.9 Warum für Android entwickeln?

Android stellt etwas Neues dar, ein mobiles Programmiergerüst, das auf modernen mobilen Geräten basiert und das von Entwicklern für Entwickler gemacht wird. Es gibt ein einfaches, leistungsfähiges und offenes SDK, keine Lizenzgebühren, eine ausgezeichnete Dokumentation und eine blühende Entwicklergemeinde. Dies alles trägt dazu bei, dass Android eine Möglichkeit darstellt, um Software zu erstellen, die eine Änderung bewirkt, wenn es darum geht, wie und warum Menschen ihre mobilen Geräte nutzen. Die Hürde, die neue Android-Entwickler überwinden müssen, ist sehr niedrig:

- Um Android-Entwickler zu werden, müssen Sie kein Zertifikat erwerben.
- Google Play bietet beim Vertrieb Ihrer Anwendungen für Verkäufe Vorkasse an oder sorgt für die Möglichkeit der In-App-Bezahlung, damit Sie mit Ihren Anwendungen Geld verdienen können.
- Es gibt für den Vertrieb von Anwendungen kein Genehmigungsverfahren.
- Entwickler behalten die vollständige Kontrolle über ihre Produkte.

Kommerziell bedeutsam ist, dass täglich mehr als 850.000 neue Android-Geräte aktiviert werden, und Studien zeigen, dass Android-Geräte den größten Teil der Smartphone-Verkäufe ausmachen. Im März 2012 unterstützte Google Play (früher *Android Marketplace*) den Vertrieb von Anwendungen in 131 Ländern; es gibt – mit steigender Tendenz – zehn Milliarden Installationen und pro Monat mehr als eine Milliarde Downloads.

1.9.1 Faktoren, die die Einführung von Android vorantreiben

Entwickler sind im Ökosystem von Android schon immer ein wichtiges Element gewesen und sowohl Google als auch die OHA wetten, dass Verbraucher nur dann mit besserer mobiler Software versorgt werden können, wenn es den Entwicklern einfacher gemacht wird, diese Anwendungen zu schreiben.

Android ist eine leistungsfähige und intuitiv zu bedienende Entwicklungsumgebung, die es Entwicklern, die nie zuvor für mobile Geräte programmiert haben, ermöglicht, schnell und einfach neuartige Anwendungen zu erstellen. Es ist nicht schwer zu erkennen, wie aufregende Android-Anwendungen zu einer Nachfrage nach Geräten geführt haben, auf denen diese Anwendungen laufen, und hierbei haben sich besonders Anwendungen hervorgetan, die von Entwicklern geschrieben wurden, die sich für Android entschieden haben, obwohl sie auch für andere Plattformen entwickeln könnten. Und weil sich Android in immer mehr Formfaktoren wie leistungsfähige Hardware, fortschrittliche Sensoren und neue APIs für Entwickler ausbreitet, wachsen auch die Möglichkeiten für Innovationen.

Der ungehinderte Zugriff auf alles, was einem System zugrunde liegt, ist schon immer das gewesen, was die Entwicklung von Software und das Interesse an Plattformen vorangetrieben hat. Die zum Internet gehörende Offenheit hat dazu geführt, dass es innerhalb von nur zehn Jahren zu einer mehrere Milliarden Euro schweren Industrie wurde. Davor waren es ein offenes System wie Linux und die leistungsfähigen APIs, die Teil des Betriebssystems Windows waren, die letztendlich zu einer explosionsartigen Ausbreitung des PC und der Anwendungsprogrammierung geführt haben. Diese Offenheit und Leistungsfähigkeit sorgen dafür, dass jeder, der die entsprechende Lust verspürt, seine Vorstellungen bei minimalem Kosteneinsatz zum Leben erwecken kann.

1.9.2 Was zwar Android hat, andere Plattformen aber nicht

Viele der Funktionen, die schon aufgelistet wurden, zum Beispiel 3D-Grafik und die Unterstützung einer systemeigenen Datenbank, stehen nicht nur in anderen nativen mobilen SDKs, sondern auch auf mobilen Browsern zur Verfügung. Die Geschwindigkeit, die mobile Plattformen wie Android und seine Mitbewerber aufgenommen haben, macht es schwierig, die verfügbaren Funktionen miteinander zu vergleichen. Die folgende unvollständige Liste führt einige der Funktionen auf, die es in Android gibt, die aber nicht in allen modernen mobilen Entwicklungsumgebungen vorhanden sind:

- **Die Anwendung *Google Maps*** – Google Maps für Handys ist sehr beliebt, und Android enthält ein Google Maps als unabhängiges, wieder verwendbares Bedienelement, das Sie in Ihren Anwendungen einsetzen können, um kartenbasierende Anwendungen zu schreiben, die die bekannte Oberfläche von Google Maps verwenden.
- **Im Hintergrund laufende Services und Anwendungen** – Die volle Unterstützung von im Hintergrund laufenden Anwendungen und Services lässt es zu, dass Sie Anwendungen erstellen, die auf einem ereignisgesteuerten Modell basieren und still vor sich hin arbeiten, während andere Anwendungen genutzt werden oder während Ihr

Handy irgendwo herumliegt, bis es klingelt, aufblitzt oder vibriert, um Ihre Aufmerksamkeit zu erregen. Vielleicht handelt es sich um einen Musikplayer, der streamt, um eine Anwendung, die sich um Aktien kümmert und Sie vor wichtigen Änderungen an Ihrem Portfolio warnt, oder um einen Dienst, der abhängig von Ihrem aktuellen Standort, der Tageszeit oder dem Anrufer entsprechend Ihren Klingelton oder dessen Lautstärke ändert. Android stellt allen Anwendungen und Entwicklern die gleichen Möglichkeiten zur Verfügung.

- **Daten und die Kommunikation zwischen Prozessen gemeinsam nutzen** – Indem Android Intents und Content Provider verwendet, sind Ihre Anwendungen in der Lage, Nachrichten auszutauschen, Verarbeitungen durchzuführen und Daten gemeinsam zu nutzen. Sie können diese Mechanismen auch dazu verwenden, die Daten und Funktionen der systemeigenen Android-Anwendungen auszuhebeln. Um die durch die offene Strategie hervorgerufenen Risiken abzufedern, sind alle Prozesse, Datenablagen und Dateien einer Anwendung so lange privat, bis sie ausdrücklich über auf Berechtigungen basierende Sicherheitsmaßnahmen freigegeben werden. Diese Maßnahmen werden detailliert in Kapitel 18, *Android-Entwicklung für Fortgeschrittene*, beschrieben.
- **Alle Anwendungen werden auf die gleiche Weise erstellt** – Android unterscheidet nicht zwischen systemeigenen Anwendungen und denen, die von dritter Seite erstellt werden. Damit erhalten Verbraucher die noch nie da gewesene Macht, das Aussehen und das Verhalten ihrer Geräte zu ändern, indem alle systemeigenen Anwendungen durch Alternativen Dritter ersetzt werden, die trotzdem Zugriff auf alle zugrunde liegenden Daten und die gesamte Hardware haben.
- **Wi-Fi Direct und Android Beam** – Indem Sie die neuartigen Kommunikations-APIs für die Kommunikation zwischen Geräten nutzen, sind Sie in der Lage, Funktionen wie die unmittelbare Freigabe von Medien und das Streaming in Anwendungen einzubinden. *Android Beam* ist eine auf NFC basierende API, die es Ihnen ermöglicht, auf Annäherung basierende Interaktionen zu unterstützen. Bei *Wi-Fi Direct* handelt es sich um ein breit gefächertes Peer-to-Peer-Netz, das für eine zuverlässige Hochgeschwindigkeitskommunikation zwischen Geräten sorgt.
- **Widgets, Live Wallpaper und die Schnellsuche** – Durch die Nutzung von Widgets und Live Wallpaper können Sie in Ihren Anwendungen Fenster des Startbildschirms verwenden. Die Schnellsuche lässt es zu, dass Sie Suchergebnisse Ihrer Anwendungen direkt in die Suchfunktion Ihres mobilen Gerätes aufnehmen.

1.9.3 Die Landschaft der mobilen Entwicklung ändert sich

Die bisher vorhandenen mobilen Entwicklungsumgebungen haben um das mobile Entwickeln eine Aura der Exklusivität gelegt. Den krassen Gegensatz dazu bildet Android. Android-Geräte werden zwar mit einem Satz der Standardanwendungen ausgeliefert, die Benutzer auf einem neuen Gerät erwarten, aber die echte Stärke liegt darin, dass Benutzer das Aussehen, das Verhalten und die Funktionen eines Gerätes vollständig an die eigenen Bedürfnisse anpassen können – was wiederum Anwendungsentwicklern ungeahnte Möglichkeiten verschafft.

Alle Android-Anwendungen sind ein originärer Bestandteil des Gerätes und nicht nur Software, die in einem geschützten Bereich abläuft. Und statt Softwareversionen für kleine Bildschirme zu schreiben, die auf schwachbrüstigen Geräten laufen kann, sind Sie in der Lage, mobile Anwendungen zusammenzubauen, die die Art ändern, wie die Menschen ihre Handys nutzen.

Die mobile Entwicklung erfreut sich gerade einer Periode schneller Erneuerungen und eines unglaublichen Wachstums. Dies sorgt bei Entwicklern sowohl für Herausforderungen als auch für Chancen, wobei jeder für sich herausfinden muss, welche Möglichkeiten da auf ihn warten. Android wird ständig weiterentwickelt und verbessert, um mit aktuellen und zukünftig entstehenden mobilen Entwicklungsumgebungen mithalten zu können, aber als Open-Source-Entwicklungsframework hängt viel von der Stärke des SDKs ab. Sein kostenloser und offener Ansatz der mobilen Anwendungsentwicklung und der vollständige Zugriff auf die Ressourcen des mobilen Gerätes bieten jedem Entwickler die Möglichkeit, sich eine eigene Entwicklungsumgebung maßzuschneidern.

1.10 Das Entwicklungsgerüst

Nachdem das »Warum« abgehandelt worden ist, lassen Sie uns einen Blick auf das »Wie« werfen. Android-Anwendungen werden normalerweise in der Programmiersprache Java geschrieben, während die Ausführung der Dateien nicht in der herkömmlichen Java VM, sondern in einer VM erfolgt, die *Dalvik* genannt wird.

Tipp

Weiter hinten in diesem Kapitel erhalten Sie eine Einführung in das Entwicklungsgerüst, das hier *Framework* genannt wird. Dies beginnt bei einer technischen Definition des Android-Softwarestapels, beschreibt, woraus das SDK besteht, und endet mit einer Einführung in die Android-Bibliotheken und einem näheren Blick auf die Dalvik VM.

Jede Android-Anwendung läuft in ihrer eigenen Dalvik-Instanz in einem eigenen Prozess und gibt dabei die Kontrolle über den Arbeitsspeicher und die Prozessverwaltung an die Android-Laufzeitumgebung ab. Diese hält dann Prozesse im Rahmen der Ressourcenverwaltung an und beendet sie gegebenenfalls. Dalvik und die Android-Laufzeitumgebung befinden sich im Linux-Kernel, der auf niedriger Ebene für die Interaktion mit der Hardware verantwortlich ist (zum Beispiel für Treiber und die Speicherverwaltung), während APIs für den Zugriff auf die zugrunde liegenden Services, Funktionen und die Hardware sorgen.

1.10.1 Was zum Paket gehört

Das Android SDK enthält alles, was Sie benötigen, um mit dem Entwickeln, Testen und Debuggen von Android-Anwendungen zu beginnen:

- **Die Android-APIs** – Den Kern des SDKs bilden die Android-API-Bibliotheken, die Entwicklern den Zugriff auf den Android-Stapel ermöglichen. Dabei handelt es sich um dieselben Bibliotheken, die auch Google verwendet, um systemeigene Android-Anwendungen zu erstellen.
- **Werkzeuge für die Entwicklung** – Das SDK enthält verschiedene Werkzeuge für die Entwicklung, die es Ihnen ermöglichen, Anwendungen zu kompilieren und zu debuggen, wodurch Sie Android-Quellcode in ausführbare Anwendungen umwandeln können. Sie erfahren in Kapitel 2, *Los geht's*, mehr über die Entwicklungswerkzeuge.
- **Der Android Virtual Device Manager und der Emulator** – Der Android-Emulator ist eine voll funktionsfähige Emulation eines interaktionsfähigen mobilen Gerätes, für das alternative Skins zur Verfügung stehen. Der Emulator läuft in einem *Android Virtual Device* (AVD), das die Hardwarekonfiguration eines Gerätes simuliert. Wenn Sie den Emulator verwenden, können Sie sehen, wie Ihre Anwendung auf einem echten Android-Gerät aussieht und sich verhält. Alle Android-Anwendungen laufen in der Dalvik VM, was den Softwareemulator zu einer ausgezeichneten Entwicklungsumgebung macht – sie ist hardwareunabhängig und sorgt für eine unabhängigere Testumgebung, als es eine einzelne Hardware könnte.
- **Eine ausführliche Dokumentation (in Englisch)** – Zum SDK gehört eine ausführliche (englischsprachige) Information als Referenz auf Codeebene, die genau aufführt, was sich in den einzelnen Paketen und Klassen befindet und wie diese verwendet werden. Zusätzlich zur Dokumentation des Codes erklärt ein Android-Referenz- und Entwicklungsleitfaden (ebenfalls in Englisch), was Sie am Anfang beachten müssen, liefert ausführliche Erklärungen über die Grundsätze, die hinter der Entwicklung mit Android stecken, hebt bewährte Vorgehensweisen hervor und sorgt dafür, dass Sie tief in die Themen des Frameworks eintauchen.
- **Beispielcode** – Das Android SDK enthält eine Auswahl an Beispielanwendungen, die einige der Möglichkeiten von Android aufzeigen, und einfache Programme, die die Verwendung bestimmter API-Funktionen verdeutlichen.
- **Online-Support** – Android hat sehr schnell eine blühende Entwicklergemeinde entstehen lassen. Sie finden in den Google Groups (*http://developer.android.com/resources/community-groups.html#ApplicationDeveloperLists*) aktive Foren von Android-Entwicklern. Dort gibt es auch regelmäßig Beiträge des Google-Entwicklerteams zu technischen und entwicklungsspezifischen Themen. Auch Stack Overflow (*www.stackoverflow.com/questions/tagged/android*) ist ein sehr beliebtes Ziel, um Fragen zu Android loszuwerden und die Antworten dazu zu finden. Ein deutschsprachiges Forum finden Sie unter Android-Hilfe (*http://www.android-hilfe.de*); hier sind gute Englischkenntnisse kein Muss.

Für diejenigen von Ihnen, die Eclipse verwenden, hat Android die *Android Development Tools* (ADT) herausgegeben. Dabei handelt es sich um ein Plug-in, das das Erstellen von Projekten vereinfacht und Eclipse eng mit dem Android-Emulator und den Build- und Debuggingwerkzeugen verbindet. Die Funktionen des ADT-Plug-ins werden im Einzelnen in Kapitel 2 beschrieben.

1.10.2 Der Android-Softwarestapel

Beim Android-Softwarestapel handelt es sich – einfach ausgedrückt – um einen Linux-Kernel und eine Sammlung von C/C++-Bibliotheken, die über ein anwendungsbezogenes Framework bereitgestellt werden, das Services und Verwaltungsmöglichkeiten für die Laufzeitumgebung und die Anwendungen zur Verfügung stellt. Der Android-Softwarestapel setzt sich aus den Elementen zusammen, die Abbildung 1.1 zeigt.

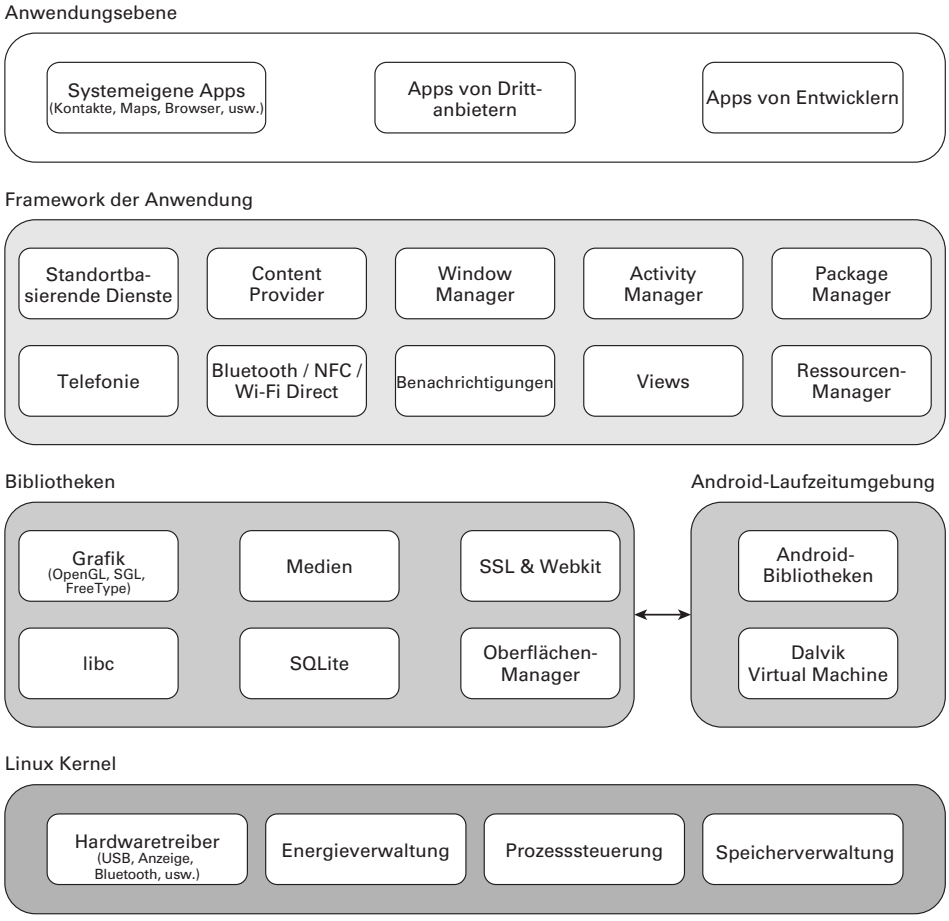

Abbildung 1.1 Elemente des Android-Softwarestapels

- **Linux-Kernel** – Zentrale Dienste (einschließlich Hardwaretreiber, Prozess- und Speicherverwaltung, Sicherheit, Netzwerk und Energieverwaltung) werden von einem Linux-2.6-Kernel bedient. Der Kernel sorgt auch für eine Abstraktionsebene zwischen der Hardware und dem Rest des Stapels.

- **Bibliotheken** – Android enthält einige C/C++-Bibliotheken , die im Kernel laufen. Zu diesen Bibliotheken gehören nicht nur *libc* und *SSL*, sondern auch:
  - Eine Medienbibliothek für die Wiedergabe von Audio- und Videomedien
  - Eine Oberflächenverwaltung, um die Anzeige steuern zu können
  - Grafikbibliotheken, die *SGI* und *OpenGL* für 2D- und 3D-Grafiken bereitstellen
  - *SQLite* für eine native Datenbankunterstützung
  - *SSL* und *WebKit* für integrierte Webbrowser und Sicherheit im Internet
- **Android-Laufzeitumgebung** – Die Laufzeitumgebung ist es, die aus einem Android-Gerät ein Android-Gerät und nicht nur eine mobile Linux-Implementierung macht. Die Android-Laufzeitumgebung ist die *Engine*, die Ihre Anwendungen mit Energie versorgt und zusammen mit den Bibliotheken die Basis des Anwendungsframeworks bildet.
  - **Zentrale Bibliotheken** – Obwohl der größte Teil der Android-Anwendungsentwicklung in der Sprache Java erfolgt, handelt es sich bei Dalvik nicht um eine Java VM. Die zentralen Android-Bibliotheken sorgen für die Funktionen, auf die Sie in den zentralen Java-Bibliotheken und den Android-spezifischen Bibliotheken zugreifen können.
  - **Dalvik VM** – Dalvik ist eine auf Register basierende *Virtual Machine*, die so optimiert wurde, dass auf einem Gerät mehrere Instanzen effizient laufen können. Diese virtuelle Maschine verlässt sich für die Threadverarbeitung auf den Linux-Kernel und auf eine einfache Speicherverwaltung.
- **Entwicklungsgerüst für Anwendungen** – Das Entwicklungsgerüst für Anwendungen (das *Application Framework*) sorgt für die Klassen, die verwendet werden, um Android-Anwendungen zu erstellen. Außerdem stellt es für den Zugriff auf die Hardware eine generische Abstraktion zur Verfügung, verwaltet die Benutzeroberfläche und weist Ressourcen zu.
- **Anwendungsebene** – Alle Anwendungen, also sowohl die systemeigenen als auch die Dritter, werden mit denselben API-Bibliotheken auf der Anwendungsebene (auch *Application Layer* genannt) zusammengebaut. Die Anwendungsebene läuft in der Android-Laufzeitumgebung und verwendet dabei die Klassen und Services, die vom Application Framework bereitgestellt werden.

1.10.3 Die Dalvik Virtual Machine

Eines der Schlüsselelemente von Android ist die Dalvik VM. Statt eine herkömmliche Java-VM wie Java ME zu verwenden, nutzt Android eine eigene, angepasste VM, die so entwickelt worden ist, dass mehrere Instanzen davon effizient auf einem einzelnen Gerät laufen können.

Die Dalvik VM verwendet den dem Gerät zugrunde liegenden Linux-Kernel, um auf unterer Ebene Funktionen auszuführen, zu denen die Sicherheit, die Threadsteuerung und die Prozess- und Speicherverwaltung gehören. Außerdem ist es möglich, C/C++-

Anwendungen zu schreiben, die sehr nahe am zugrunde liegenden Linux-Betriebssystem ablaufen. Aber auch wenn Sie so etwas machen *können*, gibt es keinen echten Grund dafür.

Wenn es bei Ihren Anwendungen auf Geschwindigkeit und den Wirkungsgrad von C/C++ ankommt, gibt es unter Android das *Native Development Kit* (NDK). Das NDK ist entwickelt worden, um Ihnen die Möglichkeit zu geben, C++-Bibliotheken zu erstellen, indem Sie zusammen mit dem Zugriff auf OpenGL die Bibliotheken *libc* und *libm* nutzen.

Tipp

Dieses Buch konzentriert sich auf das Schreiben von Anwendungen, die in Dalvik laufen und das SDK verwenden. Das Entwickeln mit dem NDK gehört nicht zum Inhalt dieses Buches. Wenn sich Ihre Interessen aber darauf richten und Sie den Linux-Kernel- und den C/C++-Unterbau von Android erforschen, Dalvik modifizieren oder Sie sonst wie unter der Motorhaube an den Dingen herumfrickeln wollen, sollten Sie unter *http://groups.google.com/group/android-internals* die Google Group Android Internals besuchen.

Der Zugriff auf die gesamte Android-Hardware und die Systemdienste werden verwaltet, indem Dalvik als mittlere Schicht (*Middle Tier*) genutzt wird. Indem eine VM verwendet wird, die die Anwendungen ausführt, verfügen Entwickler über eine Abstraktionsebene, die dafür sorgt, dass sich niemand über die Einbindung bestimmter Hardware Gedanken machen muss.

Die Dalvik VM führt Dalvik-Dateien aus. Dabei handelt es sich um ein Format, das optimiert wurde, um im Arbeitsspeicher so gut wie keine Spuren zu hinterlassen. Sie erstellen die ausführbaren Dateien, die die Erweiterung *.dex* haben, indem Sie in Java kompilierte Klassen mit Werkzeugen umwandeln, die Bestandteil des SDKs sind. Sie erfahren in Kapitel 2 mehr darüber, wie ausführbare Dalvik-Dateien erstellt werden.

1.10.4 Die Architektur von Android-Anwendungen

Die Architektur von Android ermutigt dazu, Komponenten wiederzuverwenden, und lässt es zu, dass Sie Activitys, Services und Daten mit anderen Anwendungen teilen, wobei der Zugriff auf diese Komponenten die von Ihnen vergebenen Sicherheitseinschränkungen beachtet.

Derselbe Mechanismus, der es Ihnen ermöglicht, die Kontaktverwaltung oder die Telefonanwendung durch ein selbst geschriebenes Programm zu ersetzen, kann dafür sorgen, dass Sie Komponenten Ihrer Anwendungen offenlegen, damit andere Entwickler darauf aufbauen und neue UI-Frontends oder Erweiterungen von Funktionen erstellen können. (*UI* ist die Abkürzung für *User Interface* und bedeutet – je nach Kontext – *Benutzeroberfläche* oder *Benutzerschnittstelle*.) Die folgenden Services sind die architektonischen Ecksteine aller Android-Anwendungen. Sie sorgen für das Framework, das Sie für Ihre eigene Software verwenden:

- **Activity Manager und Fragment Manager** – Sie steuern den Lebenszyklus Ihrer *Activitys* beziehungsweise *Fragments*, einschließlich der Verwaltung des Activity-Stapels (die in den Kapiteln 3 und 4 beschrieben wird).
- **Views** – Werden so für die Konstruktion der Benutzeroberfläche für Ihre Activitys und Fragments verwendet, wie es in Kapitel 4 beschrieben wird.
- **Notification Manager** – Sorgt für einen einheitlichen und unaufdringlich arbeitenden Mechanismus, um Benutzer so auf etwas aufmerksam zu machen, wie es in Kapitel 10 beschrieben wird.
- **Content Provider** – Ermöglicht es Ihren Anwendungen, Daten so gemeinsam zu nutzen, wie es in Kapitel 8 beschrieben wird.
- **Resource Manager** – Macht es möglich, dass nicht aus Code bestehende Ressourcen wie *Strings* (Zeichenfolgen) und Grafiken eingebunden werden können (siehe Kapitel 3).
- **Intents** – Sorgen für einen Mechanismus, um Daten zwischen Anwendungen und deren Komponenten zu übertragen, was in Kapitel 5 beschrieben wird.

1.10.5 Android-Bibliotheken

Android bietet für die Entwicklung von Anwendungen eine Vielzahl von APIs an. Statt diese hier alle aufzuführen, sollten Sie sich unter *http://developer.android.com/reference/packages.html* mit der entsprechenden Liste beschäftigen, weil Sie dort immer eine vollständige Übersicht über die Pakete erhalten, die zum Android SDK gehören. Android versucht, ein möglichst breites Spektrum mobiler Hardware anzusprechen. Sie sollten aber immer davon ausgehen, dass es auch vom Hostgerät abhängt, ob erweiterte oder optionale APIs genutzt und/oder implementiert werden können.