

Kapitel 1

Hallo Android

Inhalt

- Hintergrund der Entwicklung mobiler Applikationen
- Was ist Android?
- Auf welchen Geräten Android läuft
- Warum Sie für mobile Geräte und Android entwickeln sollten
- Das Android SDK und Entwicklungs-Framework

1.1 Android-Anwendungsentwicklung

Ob Sie ein erfahrener Entwickler für mobile Geräte, ein Desktop- oder Web-Entwickler oder ein kompletter Programmieranfänger sind, Android stellt eine aufregende Möglichkeit dar, Anwendungen für ein Publikum von über zwei Milliarden Android-Gerätebenutzern zu schreiben.

Sie kennen wahrscheinlich schon Android, die am weitesten verbreitete Software für Mobiltelefone. Sollten Sie dagegen dieses Buch in der Hoffnung anschaffen, dass es ihnen hilft, eine unaufhaltsame Armee von emotionslosen Roboter-Kriegern zu schaffen, die dem unerbittlichen Wunsch folgen, die Erde von der Geißel der Menschheit zu reinigen, sollten Sie den Kauf dieses Buch (und Ihre Lebensziele) überdenken.

Bei der Vorstellung von Android im Jahre 2007 beschrieb Andy Rubin es mit folgenden Worten:

»Die erste wirklich offene und umfassende Plattform für mobile Geräte. Es umfasst ein Betriebssystem, eine Benutzeroberfläche und Anwendungen – allesamt Software für den

Betrieb eines Mobiltelefons, jedoch ohne die proprietären Hindernisse, die die mobile Innovation behindert haben.«

Where's My Gphone? (<http://googleblog.blogspot.com/2007/11/wheres-my-gphone.html>)

Seitdem hat sich Android über Mobiltelefone hinaus entwickelt und bietet eine Entwicklungsplattform für eine immer breitere Palette von Hardware wie Tablets, Fernseher, Uhren, Autos und das Internet of Things (IoT).

Android ist ein Open-Source-Software-Stack, der ein Betriebssystem, Middleware und wichtige Anwendungen für mobile und eingebettete Geräte enthält.

Entscheidend für uns als Entwickler ist, dass es auch eine Vielzahl von Bibliotheken enthält, die es ermöglichen, Anwendungen zu schreiben, die das Aussehen, das Gefühl und die Funktion der Android-Geräte prägen, auf denen sie laufen.

In Android werden System-, Bundle- und alle Anwendungen von Drittanbietern mit den gleichen APIs geschrieben und von der gleichen Laufzeitumgebung ausgeführt. Diese APIs bieten Hardwarezugriff, Videoaufzeichnung, standortbasierte Dienste, Unterstützung für Hintergrunddienste, Karten, Benachrichtigungen, Sensoren, relationale Datenbanken, Kommunikation zwischen Anwendungen, Bluetooth, NFC sowie 2D- und 3D-Grafik.

Dieses Buch beschreibt, wie Sie diese APIs verwenden, um Ihre eigenen Android-Anwendungen zu erstellen. In diesem Kapitel lernen Sie einige Richtlinien für die Entwicklung auf mobiler und eingebetteter Hardware kennen und lernen einige der Plattformfunktionen kennen, die Android-Entwicklern zur Verfügung stehen.

Android verfügt über leistungsfähige APIs, ein riesiges und vielfältiges Ökosystem von Benutzern, exzellente Dokumentation, eine blühende Entwickler-Community und erfordert keine Kosten für Entwicklung oder Vertrieb. Da das Ökosystem der Android-Geräte ständig wächst, haben Sie die Möglichkeit, innovative Anwendungen für Benutzer zu erstellen, unabhängig von Ihrer Entwicklungserfahrung.

1.2 Ein wenig Hintergrund

In den Tagen vor Instagram, Snapchat und Pokémon Go, als Google noch ein Zwinkern in den Augen seiner Gründer war und Dinosaurier die Erde durchstreiften, waren Mobiltelefone nur so kleine, tragbare Telefone, die in eine Aktentasche passen und über Batterien verfügen, die bis zu mehreren Stunden halten können. Sie boten jedoch die Freiheit, Anrufe zu tätigen, ohne physisch mit dem Festnetz verbunden zu sein.

In den Jahren seit der Markteinführung des ersten Android-Geräts sind Smartphones allgegenwärtig und unverzichtbar geworden. Die Weiterentwicklung der Hardware hat die Geräte leistungsfähiger gemacht, mit größeren, helleren Bildschirmen und fortschritt-

licher Hardware wie Beschleunigungssensoren, Fingerabdruckscannern und ultrahoch-auflösenden Kameras.

Dieselben Fortschritte haben in jüngster Zeit zu einer starken Zunahme zusätzlicher Formfaktoren für Android-Geräte geführt, darunter eine Vielzahl von Smartphones, Tablets, Uhren und Fernsehgeräten.

Diese Hardware-Innovationen bieten einen fruchtbaren Boden für die Software-Entwicklung und bieten viele Möglichkeiten, innovative neue Anwendungen zu schaffen.

1.2.1 Die nicht allzu ferne Vergangenheit

In den frühen Tagen der Entwicklung nativer Telefonanwendungen mussten Entwickler, die im Allgemeinen in Low-Level C oder C++ programmierten, die spezifische Hardware verstehen, für die sie programmiert haben, typischerweise ein einzelnes Gerät oder möglicherweise eine Reihe von Geräten eines einzigen Herstellers. Die Komplexität dieses Ansatzes führte dazu, dass die Anwendungen, die für diese Geräte geschrieben wurden, oft hinter ihren Hardware-Gegenständen zurückblieben. Mit fortschreitender Hardware-technologie und mobilem Internetzugang ist dieser geschlossene Ansatz veraltet.

Der nächste bedeutende Fortschritt in der Entwicklung von Mobiltelefonanwendungen war die Einführung von Java-gehosteten MIDlets. MIDlets wurden auf einer Java Virtual Machine (JVM) ausgeführt, einem Prozess, der die zugrunde liegende Hardware abstrahiert und es Entwicklern ermöglicht, Anwendungen zu erstellen, die auf vielen Geräten laufen, die die Java-Laufzeitumgebung unterstützen.

Leider führte diese Bequemlichkeit dazu, dass der Zugriff auf die Gerätehardware stärker eingeschränkt wurde. Ebenso galt es als normal, dass Anwendungen von Drittanbietern andere Hardwarezugriffs- und Ausführungsrechte erhalten als solche, die von den Telefonherstellern an native Anwendungen vergeben wurden, wobei MIDlets oft nur wenige von beiden erhalten.

Die Einführung von Java MIDlets erweiterte die Zielgruppe der Entwickler, aber das Fehlen von Low-Level-Hardwarezugriff und Sandbox-Ausführung bedeutete, dass die meisten mobilen Anwendungen reguläre Desktop-Programme oder Websites waren, die für die Darstellung auf einem kleineren Bildschirm konzipiert waren, und nicht die Vorteile der inhärenten Mobilität der Handheld-Plattform nutzten.

1.2.2 Leben in der Zukunft

Bei seiner Einführung war Android Teil einer neuen Welle moderner mobiler Betriebssysteme, die speziell zur Unterstützung der Anwendungsentwicklung auf immer leistungsfähigerer mobiler Hardware entwickelt wurden.

Android bietet eine offene Entwicklungsplattform, die auf einem Open-Source-Linux-Kernel basiert. Der Hardwarezugriff steht allen Anwendungen über eine Reihe von API-Bibliotheken zur Verfügung, und die Interaktion mit den Anwendungen wird bei sorgfältiger Kontrolle vollständig unterstützt.

Unter Android sind alle Anwendungen gleichberechtigt. Anwendungen von Drittanbietern und systemeigene Android-Anwendungen werden mit den gleichen APIs geschrieben und in der gleichen Laufzeitumgebung ausgeführt. Benutzer können die meisten Systemanwendungen durch eine Alternative eines Drittanbieters ersetzen; sogar das Telefonprogramm und der Startbildschirm können ersetzt werden.

1.3 Das Android-Ökosystem

Das Android-Ökosystem besteht aus einer Kombination von drei Komponenten:

- Ein freies, quelloffenes Betriebssystem für Embedded-Geräte
- Eine Open-Source-Entwicklungsplattform für die Erstellung von Anwendungen
- Geräte, auf denen das Android-Betriebssystem läuft (und die dafür erstellten Anwendungen)

Genauer gesagt, besteht Android aus mehreren notwendigen und abhängigen Teilen, einschließlich der folgenden:

- Ein Compatibility Definition Document (CDD) und eine Compatibility Test Suite (CTS), die die Fähigkeiten beschreiben, die ein Gerät benötigt, um den Android-Software-Stack zu unterstützen.
- Ein Linux-Betriebssystemkern, der eine Low-Level-Schnittstelle mit Hardware, Speicherverwaltung und Prozesssteuerung bietet, die für mobile und Embedded-Geräte optimiert ist.
- Open-Source-Bibliotheken für die Anwendungsentwicklung, einschließlich SQLite, WebKit, OpenGL und eines Medienmanagers.
- Ein Laufzeitsystem, das zum Ausführen und Bereitstellen von Android-Anwendungen verwendet wird, einschließlich der Android Run Time (ART) und der Kernbibliotheken, die Android-spezifische Funktionen bereitstellen. Das Laufzeitsystem ist so ausgelegt, dass es klein und effizient für den Einsatz auf kleinen Geräten läuft.
- Ein Anwendungs-Framework, das Systemdienste der Anwendungsschicht, einschließlich Window Manager und Location Manager, Datenbanken, Telefonie und Sensoren, unabhängig voneinander zur Verfügung stellt.
- Ein Framework für die Benutzeroberfläche, das zum Bereitstellen und Starten von Apps verwendet wird.

- Eine Reihe von vorinstallierten Kern-Apps.
- Ein Software Development Kit (SDK) zur Erstellung von Apps, einschließlich der zugehörigen Tools, IDE, Beispielcode und Dokumentation.

Was Android wirklich überzeugend macht, ist seine offene Philosophie, die sicherstellt, dass Sie alle Mängel in der Benutzeroberfläche oder im systemeigenen Anwendungsdesign beheben können, indem Sie eine Erweiterung oder einen Ersatz schreiben. Android bietet Ihnen als Entwickler die Möglichkeit, Anwendungen zu erstellen, die so gestaltet sind, dass sie genauso aussehen, sich anfühlen und funktionieren, wie Sie es sich vorstellen.

Mit mehr als 2 Milliarden aktiven Nutzern von Geräten mit dem Betriebssystem Android, die allein 2016 über 82 Milliarden Apps und Spiele von Google Play installierten, stellt das Android-Ökosystem eine einmalige Chance dar, Apps zu erstellen, die Milliarden von Menschenleben beeinflussen und verbessern können.

1.4 Vorinstallierte Android-Anwendungen

Android-Geräte verfügen in der Regel über eine Reihe vorinstallierter Apps, die von den Benutzern erwartet werden. Auf Smartphones sind dies typischerweise:

- Ein Wahlprogramm (Dialer) für das Telefon
- Eine SMS-Verwaltungsanwendung
- Ein Webbrowser
- Ein E-Mail-Client
- Ein Kalender
- Eine Kontaktliste
- Ein Musik-Player und eine Bildergalerie
- Kamera- und Videoaufzeichnungsanwendung
- Ein Taschenrechner
- Ein Startbildschirm
- Ein Wecker

In vielen Fällen werden Android-Geräte auch mit den folgenden proprietären Google Mobile-Anwendungen ausgeliefert:

- Der Google Play Store zum Herunterladen von Android-Anwendungen von Drittanbietern
- Die Google Maps-Anwendung, einschließlich StreetView, Wegbeschreibungen und Navigation, Satellitenansichten und Verkehrsinformationen
- Der Gmail-E-Mail-Client
- Der YouTube-Video-Player

- Der Google Chrome-Browser
- Der Google-Startbildschirm und der Google-Assistent

Die Daten, die von vielen dieser systemeigenen Anwendungen gespeichert und verwendet werden, wie beispielsweise Kontaktdetails, stehen auch Anwendungen von Drittanbietern zur Verfügung.

Die genaue Zusammensetzung der Anwendungen, die auf neuen Android-Geräten verfügbar sind, kann je nach Hardwarehersteller, Netzbetreiber oder Distributor und Gerätetyp variieren.

Da Android Open-Source ist, können Netzbetreiber und OEMs die Benutzeroberfläche und die mit jedem Android-Gerät gebündelten Anwendungen anpassen.

Bei kompatiblen Geräten müssen die zugrunde liegende Plattform und das SDK über OEM- und Netzbetreiber-Varianten hinweg konsistent bleiben. Das Aussehen der Benutzeroberfläche mag variieren, aber Ihre Anwendungen funktionieren auf allen kompatiblen Android-Geräten auf die gleiche Weise.

1.5 Android SDK-Features

Für uns Entwickler liegt der wahre Reiz von Android in seinen APIs.

Als neutrale Plattform bietet Ihnen Android die Möglichkeit, Apps zu erstellen, die genauso Teil des Smartphones sind wie alles, was out-of-the-box zur Verfügung gestellt wird. Die folgende Liste zeigt einige der bemerkenswertesten Android-Funktionen:

- Transparenter Zugriff auf Telefonie- und Internetressourcen durch Unterstützung von GSM-, EDGE-, 3G-, 4G-, LTE- und Wi-Fi-Netzwerken, so dass Ihre Anwendung Daten über Mobil- und Wi-Fi-Netzwerke senden und abrufen kann.
- Umfassende APIs für standortbasierte Dienste wie GPS und netzwerkbasierter Standorterkennung
- Volle Unterstützung für die Integration von Karten in die Benutzeroberfläche
- Volle Kontrolle über die Multimedia-Hardware, einschließlich Wiedergabe und Aufnahme mit Kamera und Mikrofon
- Medienbibliotheken zum Abspielen und Aufnehmen einer Vielzahl von Audio-, Video- oder Standbildformaten
- APIs für die Verwendung von Sensorhardware, einschließlich Beschleunigungsmesser, Kompass, Barometer und Fingerabdrucksensoren
- Bibliotheken für die Verwendung von Wi-Fi-, Bluetooth und NFC-Hardware
- Gemeinsame Datenspeicher und APIs für Kontakte, Kalender und Multimedia-dateien
- Hintergrunddienste und ein fortschrittliches Benachrichtigungssystem

- Integrierter Webbrowser
- Mobil-optimierte, hardware-beschleunigte Grafik, einschließlich einer pfadbasierten 2D-Grafikbibliothek und Unterstützung für 3D-Grafiken mit OpenGL ES 2.0
- Lokalisierung durch ein dynamisches Ressourcen-Framework

1.6 Worauf läuft Android?

Das erste Android-Handy, das T-Mobile G1, wurde im Oktober 2008 in den USA auf den Markt gebracht. Bis Ende 2017 gibt es weltweit mehr als 2 Milliarden monatlich aktive Android-Geräte, es ist damit das weltweit am weitesten verbreitete Smartphone-Betriebssystem.

Anstatt ein mobiles Betriebssystem zu sein, das für eine einzelne Hardware-Implementierung entwickelt wurde, unterstützt Android eine Vielzahl von Hardware-Plattformen, von Smartphones bis hin zu Tablets, Fernsehern, Uhren und IoT-Geräten.

Da keine Lizenzgebühren oder proprietäre Software anfallen, sind die Kosten für die Hersteller von Smartphones für die Bereitstellung von Android-Geräten vergleichsweise niedrig, was in Verbindung mit einem riesigen Ökosystem aus leistungsstarken Anwendungen die Gerätehersteller dazu veranlasst hat, immer vielfältigere und maßgeschneiderte Hardware zu produzieren.

Infolgedessen erstellen Hunderte von Herstellern, darunter Samsung, LG, HTC und Motorola, Android-Geräte. Diese Geräte werden über hunderte von Netzanbietern weltweit an die Anwender verteilt.

1.7 Warum für mobile Geräte entwickeln?

Smartphones sind für uns so fortschrittlich und persönlich geworden, dass sie für viele Menschen zu einer Erweiterung ihrer selbst geworden sind. Studien haben gezeigt, dass viele Handy-Nutzer ängstlich werden, wenn sie ihr Gerät verlegen, die Verbindung verlieren oder der Akku leer ist.

Die Allgegenwart der Smartphones und unsere Verbundenheit mit ihnen machen sie zu einer grundlegend anderen Plattform für die Entwicklung als PCs. Mit einem Mikrofon, einer Kamera, einem Touchscreen, einer Positionserkennung und Umgebungssensoren kann ein Smartphone effektiv zu einem außersinnlichen Wahrnehmungsgerät werden.

Der Besitz von Smartphones übertrifft den Besitz von Computern in vielen Ländern, mit mehr als 3 Milliarden Smartphone-Nutzern weltweit. 2009 war das Jahr, in dem zum

ersten Mal mehr Menschen von einem Smartphone aus auf das Internet zugegriffen als von einem PC aus.

Die zunehmende Popularität von Smartphones, kombiniert mit der zunehmenden Verfügbarkeit von mobilen Hochgeschwindigkeitsdaten und Wi-Fi-Hotspots, hat eine riesige Chance für fortschrittliche mobile Anwendungen geschaffen.

Smartphone-Apps haben die Art und Weise, wie Menschen ihre Telefone benutzen, verändert. Dies gibt Ihnen, dem Anwendungsentwickler, die einzigartige Gelegenheit, dynamische, überzeugende neue Anwendungen zu entwickeln, die zu einem wichtigen Bestandteil des Lebens der Menschen werden.

1.8 Warum für Android entwickeln?

Neben dem Zugang zum größten Ökosystem der Smartphone-Benutzer stellt Android ein dynamisches Framework für die App-Entwicklung dar, das auf der Realität moderner mobiler Geräte basiert, die von Entwicklern für Entwickler entwickelt wurden.

Mit einem einfachen, leistungsstarken und offenen SDK, ohne Lizenzgebühren, exzellenter Dokumentation, einer Vielzahl von Geräten und Formfaktoren und einer blühenden Entwickler-Community bietet Android die Möglichkeit, Software zu entwickeln, die das Leben von Menschen verändern kann.

Die Einstiegshürde für neue Android-Entwickler ist gering:

- Es ist keine Zertifizierung erforderlich, um Android-Entwickler zu werden.
- Der Google Play Store bietet kostenlose Optionen für den Vorabkauf, die In-App-Abrechnung und Abonnement-Optionen für die Verteilung und Vermarktung Ihrer Anwendungen.
- Es gibt keinen Genehmigungsprozess für die Verteilung von Anwendungen.
- Entwickler haben die totale Kontrolle über ihre Marken.

Aus kommerzieller Sicht stellt Android das am weitesten verbreitete Smartphone-Betriebssystem dar und bietet Zugang zu über 2 Milliarden monatlich aktiven Android-Geräten weltweit und bietet eine beispiellose Reichweite, um Ihre Anwendungen für Benutzer auf der ganzen Welt verfügbar zu machen.

1.9 Einführung in das Development Framework

Android-Anwendungen werden in der Regel mit Java (oder Kotlin) als Programmiersprache geschrieben, aber nicht mit einer Java VM, sondern mit der Android Run Time (ART) ausgeführt.

Hinweis



In der Vergangenheit wurden Android-Anwendungen hauptsächlich mit Hilfe der Java-Syntax geschrieben. In jüngster Zeit hat Android Studio 3.0 die volle Unterstützung für Kotlin als offizielle Premium-Sprache für die Anwendungsentwicklung eingeführt. Kotlin ist eine JVM-Sprache, die mit bestehenden Android-Sprachen und der Android Run Time interoperabel ist und es Ihnen ermöglicht, sowohl die Java- als auch die Kotlin-Syntax innerhalb derselben Anwendungen zu verwenden.

Jede Android-Anwendung läuft in einem separaten Prozess und überträgt die Verantwortung für Speicher- und Prozessmanagement auf die Android-Laufzeitumgebung (Android Run Time, ART), die Prozesse bei Ressourcenknappheit wenn nötig beendet.

Das Android-Laufzeitsystem läuft auf der Basis des Linux-Kernels, der sich um das Zusammenspiel mit der Low-Level-Hardware kümmert, einschließlich Treiber und Speicherverwaltung, während eine Reihe von APIs den Zugriff auf alle zugrunde liegenden Dienste, Funktionen und Hardware ermöglicht.

1.9.1 Was kommt in der Box?

Das Android SDK enthält alles, was Sie zum Entwickeln, Testen und Debuggen von Android-Anwendungen benötigen:

- Die Android-API-Bibliotheken: Der Kern des SDK sind die Android-API-Bibliotheken, die Entwicklern den Zugriff auf den Android-Stack ermöglichen. Dies sind dieselben Bibliotheken, die Google verwendet, um systemeigene Android-Anwendungen zu erstellen.
- Entwicklungswerkzeuge: Das SDK enthält die Android Studio IDE und verschiedene andere Entwicklungswerkzeuge, mit denen Sie Ihre Anwendungen kompilieren und debuggen können, um Android-Quellcode in ausführbare Anwendungen zu verwandeln. Mehr über die Entwicklertools erfahren Sie in Kapitel 2.
- Der Android Virtual Device Manager und der Emulator: Der Android-Emulator ist ein vollständig interaktiver Emulator für mobile Geräte mit mehreren alternativen Skins. Der Emulator läuft in einem Android Virtual Device (AVD), das eine Hardwarekonfiguration des Geräts simuliert. Mit dem Emulator können Sie sehen, wie Ihre Anwendungen auf einem echten Android-Gerät aussehen und sich verhalten werden. Alle Android-Anwendungen laufen innerhalb von ART, so dass der Software-Emulator eine ausgezeichnete Entwicklungsumgebung ist. Weil er hardwareneutral ist, bietet er eine bessere unabhängige Testumgebung als jede einzelne Hardware-Implementierung.
- Vollständige Dokumentation: Das SDK enthält umfangreiche Referenzinformationen auf Code-Ebene, die genau beschreiben, was in jedem Paket und jeder Klasse

enthalten ist und wie man sie verwendet. Zusätzlich zur Code-Dokumentation erklären die Referenzdokumentation und das Entwicklerhandbuch von Android den Einstieg, geben detaillierte Erklärungen zu den Grundlagen der Android-Entwicklung, heben die optimale Vorgehensweise hervor und bieten tiefe Einblicke in Framework-Themen.

- **Beispielcode:** Das Android SDK enthält eine Auswahl von Beispielanwendungen, die einige der Möglichkeiten von Android demonstrieren, sowie einfache Programme, die die Verwendung einzelner API-Funktionen verdeutlichen.
- **Online-Support:** Android hat lebendige Entwickler-Communities in den meisten sozialen Netzwerken, Slack (*slack.com*) und vielen Entwicklerforen. Stack Overflow (*www.stackoverflow.com/questions/tagged/android*) ist ein sehr beliebtes Forum für Android-Fragen und ein großartiger Ort, um Antworten auf Fragen von Anfängern zu finden. Viele Android-Ingenieure von Google sind auf Stack Overflow und Twitter aktiv.

1.9.2 Den Android-Software-Stack verstehen

Der Android-Software-Stack ist ein Linux-Kernel und eine Sammlung von C/C++-Bibliotheken, die über ein Anwendungs-Framework zugänglich sind, das Dienste für die Laufzeit und Anwendungen bereitstellt und diese verwaltet, wie in Abbildung 1.1 dargestellt.

- **Linux-Kernel:** Kerndienste (einschließlich Hardwaretreiber, Prozess- und Speicherverwaltung, Sicherheits-, Netzwerk- und Energiemanagement) werden von einem Linux-Kernel verwaltet (die spezifische Kernelversion hängt von der Version der Android-Plattform und der Hardwareplattform ab).
- **Hardware Application Layer (HAL):** Der HAL stellt eine Abstraktionsschicht zwischen der zugrunde liegenden Hardware des physischen Geräts und dem Rest des Stacks dar.
- **Bibliotheken:** Android läuft auf dem Kernel und HAL und umfasst verschiedene C/C++-Kernbibliotheken wie libc und SSL sowie die folgenden:
 - Medienbibliothek zur Wiedergabe von Audio- und Videomedien
 - Ein Oberflächen-Manager für die Anzeigeverwaltung
 - Grafikbibliotheken mit SGL und OpenGL für 2D- und 3D-Grafiken
 - SQLite für eingebaute Datenbankunterstützung
 - SSL und WebKit für integrierten Webbrowser und Internetsicherheit
- **Android Run Time (ART):** Das Laufzeitsystem ist das, was ein Android-Smartphone zu einem Android-Smartphone macht und nicht zu einer mobilen Linux-Implementierung. Einschließlich der Kernbibliotheken ist die Android Run Time die Engine, die Ihre Anwendungen laufen lässt und die Grundlage für das Application Framework bildet.
- **Kernbibliotheken:** Obwohl die meisten Android-Anwendungen mit Hilfe der Java- oder Kotlin-JVM-Sprachen entwickelt wurden, ist ART keine Java-VM. Die Android

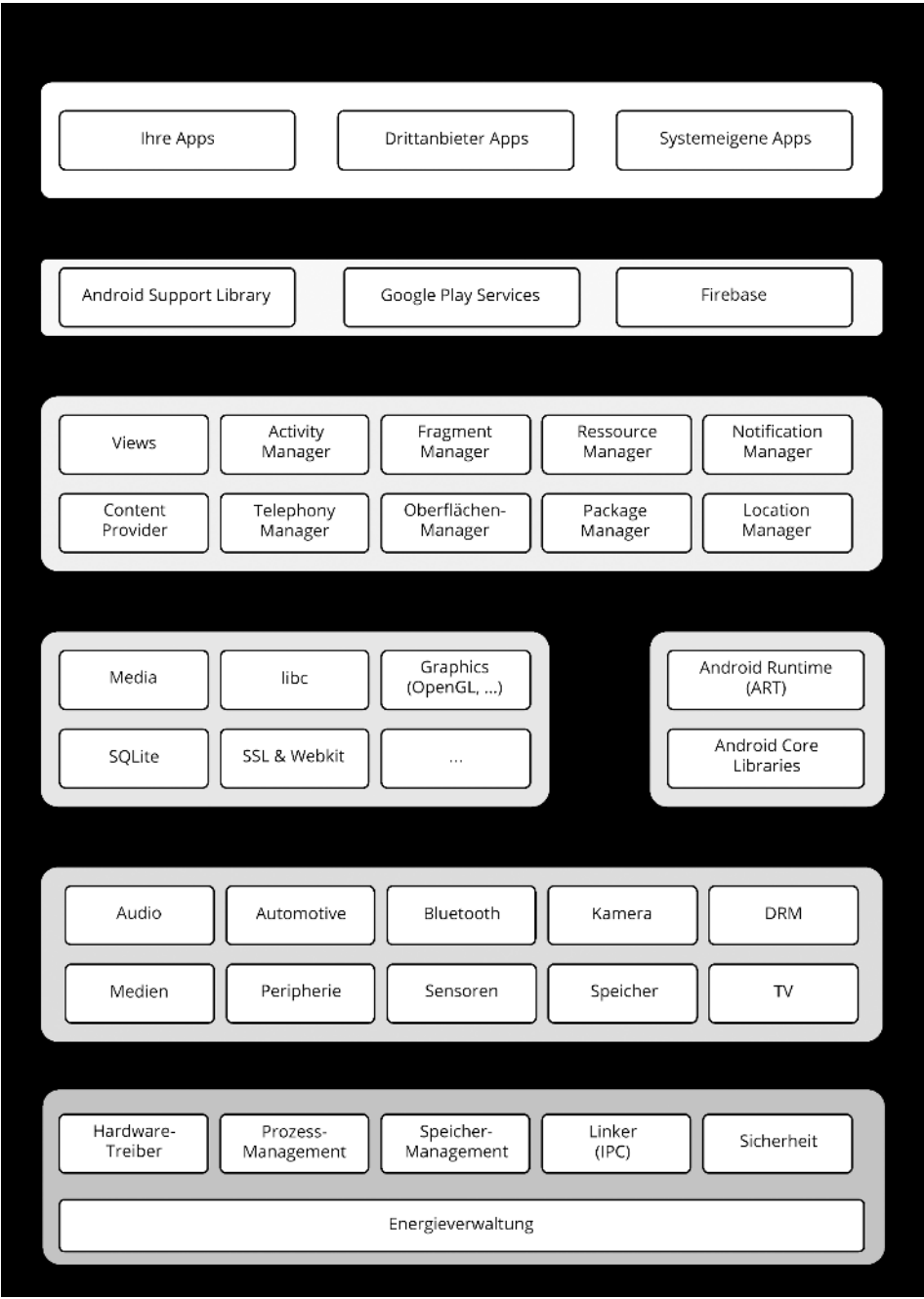


Abbildung 1.1 Android Software Stack

Kernbibliotheken stellen die meisten der in den Java-Kernbibliotheken verfügbaren Funktionen sowie die Android-spezifischen Bibliotheken zur Verfügung.

- **Application Framework:** Das Application Framework stellt die Klassen zur Verfügung, die zur Erstellung von Android-Anwendungen verwendet werden. Es bietet auch eine generische Abstraktion für den Hardwarezugriff und verwaltet die Benutzeroberfläche und die Anwendungsressourcen.
- **Application Layer (Anwendungsschicht):** Alle Anwendungen, sowohl systemeigene als auch die von Drittanbietern, werden auf der Anwendungsschicht und damit auf der Basis derselben API-Bibliotheken erstellt. Die Anwendungsschicht läuft innerhalb des Android-Laufzeitsystems (Android Run Time ART) unter Verwendung der Klassen und Dienste, die vom Application Framework zur Verfügung gestellt werden.

1.9.3 Das Android-Laufzeitsystem (Android Run Time ART)

Eines der Schlüsselemente von Android ist das Android-Laufzeitsystem, die Android Run Time (ART). Anstatt eine traditionelle Java-VM wie Java ME zu verwenden, setzt Android seine eigene, angepasste Laufzeit ein, um sicherzustellen, dass mehrere Instanzen effizient auf einem einzigen Gerät laufen.

ART verwendet den dem Gerät zugrunde liegenden Linux-Kernel, um Low-Level-Funktionalität, einschließlich Sicherheit, Threading, Prozess- und Speicherverwaltung, zu bewerkstelligen. Es ist auch möglich, C/C++-Anwendungen zu schreiben, die näher an das zugrunde liegende Linux-Betriebssystem heranreichen. Obwohl Sie dies tun können, gibt es dafür in den meisten Fällen keinen Grund.

Wenn die Geschwindigkeit und Effizienz von C/C++ für Ihre Anwendung erforderlich sind, stellt Android ein systemnahes Development Kit (NDK) zur Verfügung. Das NDK wurde entwickelt, um es Ihnen zu ermöglichen, C++-Bibliotheken mit Hilfe der lib- und libm-Bibliotheken zu erstellen, zusammen mit dem direkten Zugriff auf OpenGL.

Hinweis



Dieses Buch konzentriert sich ausschließlich auf das Schreiben von Anwendungen, die unter ART mit dem SDK laufen; die NDK-Entwicklung fällt nicht in den Rahmen dieses Buches. Wenn Ihre Neigungen in Richtung NDK-Entwicklung gehen, den Linux-Kernel und C/C++-Unterbau von Android erforschen, ART modifizieren oder anderweitig an Dingen unter der Haube herumbasteln wollen, dann schauen Sie sich das Android Open Source Project unter source.android.com an.

Alle Android-Hardware- und System-Service-Zugänge werden mit ART als Middle Tier verwaltet. Durch die Nutzung dieses Laufzeitsystems für die Ausführung von Anwendungen haben Entwickler eine Abstraktionsschicht, die sicherstellt, dass sie sich nie um eine bestimmte Hardware-Implementierung kümmern müssen.

Die ART führt Dalvik Executable Dateien (.dex) aus – benannt nach einer früheren Implementierung einer virtuellen Maschine namens »Dalvik« – ein Format, das optimiert wurde, um möglichst geringen Speicherbedarf zu gewährleisten. Sie erstellen .dex-Executables, indem Sie kompilierte Java-Klassen mit Hilfe der im SDK enthaltenen Tools transformieren.

Hinweis



Mehr darüber, wie man Dalvik-Executables erstellt, erfahren Sie in Kapitel 2.

1.9.4 Android-Anwendungsarchitektur

Die Architektur von Android fördert die Wiederverwendung von Komponenten und ermöglicht es Ihnen, Activities, Services und Daten mit anderen Anwendungen zu veröffentlichen und gemeinsam zu nutzen, wobei der Zugriff über die von Ihnen definierten Sicherheitseinschränkungen kontrolliert wird.

Mit dem gleichen Mechanismus, der es Ihnen ermöglicht, einen neuen Kontaktmanager oder Telefonwähler zu erstellen, können Sie die Komponenten Ihrer Anwendung freigeben, damit andere Entwickler darauf aufbauen können, indem sie neue Oberflächen-Frontends oder Funktionserweiterungen erstellen.

Die folgenden Anwendungsdienste sind die architektonischen Eckpfeiler aller Android-Anwendungen und bieten das Grundgerüst, das Sie für Ihre eigene Software verwenden werden:

- Activity-Manager und Fragment-Manager: Activities und Fragmente werden verwendet, um die Benutzeroberfläche Ihrer Anwendungen zu definieren. Die Activity- und Fragment-Manager steuern den Lebenszyklus Ihrer Activities beziehungsweise Fragmente, einschließlich der Verwaltung des Activity-Stacks (beschrieben in den Kapiteln 3 und 4).
- Views: Views werden verwendet, um die Steuerelemente der Benutzeroberfläche innerhalb Ihrer Activities und Fragmente zu konstruieren, wie in Kapitel 4 beschrieben.
- Notification-Manager: Bietet einen konsistenten und unaufdringlichen Mechanismus zur Benachrichtigung Ihrer Benutzer, wie in Kapitel 10 beschrieben.
- Content-Provider: Ermöglicht die gemeinsame Nutzung von Daten durch Ihre Anwendungen, wie in Kapitel 8 beschrieben.
- Resource-Manager: Ermöglicht die Auslagerung von nicht-kodierten Ressourcen wie Strings und Grafiken, wie in Kapitel 3 gezeigt.
- Intents: Bieten einen Mechanismus für den Datentransfer zwischen Anwendungen und ihren Komponenten, wie in Kapitel 5 beschrieben.

1.9.5 Android-Bibliotheken

Android bietet eine Reihe von APIs für die Entwicklung Ihrer Anwendungen. Anstatt sie alle hier aufzulisten, lesen Sie die Dokumentation unter developer.android.com/reference/packages.html, die eine vollständige Liste der Pakete enthält, die im Android SDK enthalten sind.

Android zielt auf eine breite Palette mobiler Hardware ab. Beachten Sie daher, dass die Eignung und Implementierung einiger der erweiterten oder optionalen APIs je nach Endgerät variieren kann.