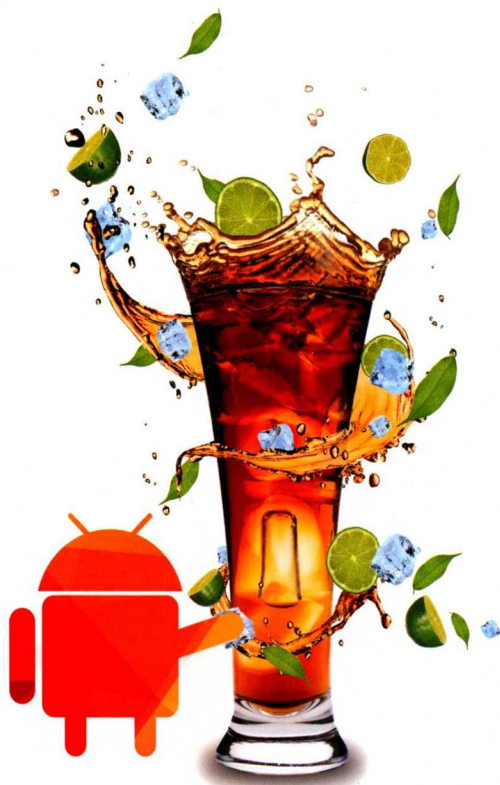


深入分析Android系统新特性
掌握独立阅读Android系统源码的能力

Broadview[®]
www.broadview.com.cn



深入剖析 Android新特性

强波 著



中国工信出版集团



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

深入剖析 Android 新特性

强波 著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

Android 系统发布于 2008 年,到 2018 年已经有十年的时间。经过十年的发展,Android 已经成为全球第一大操作系统。目前,Android 拥有几十亿用户,几百万的应用程序,更有无数的开发者。在过去的十年里,Android 一直没有停止更新的步伐,最近几年,Android 以每年一个大版本的速度向前演进。并且,在一年内还会有若干的小版本发布。很自然,开发者需要不断地了解这些新增的功能和特性。

本书对最近几个版本(5.0~8.0)的主要新增功能进行了整理和解析,本书的重点不仅仅是讲解这些新增功能特性的外部行为,而是在结合 AOSP 的源码基础上,解析这些功能特性的内部实现。任何已有的功能都是固定的,随着行业的发展,Android 系统在未来可能加入的功能是不确定的,只有掌握了阅读和分析 AOSP 源码的能力,才能应对不确定的变化,并具备今后能够自行研究 Android 系统的能力。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

深入剖析 Android 新特性 / 强波著. —北京: 电子工业出版社, 2018.5

ISBN 978-7-121-33933-2

I. ①深… II. ①强… III. ①移动终端—应用程序—程序设计 IV. ①TN929.53

中国版本图书馆 CIP 数据核字(2018)第 060672 号

责任编辑: 陈晓猛

印 刷: 三河市君旺印务有限公司

装 订: 三河市君旺印务有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱

邮编: 100036

开 本: 787×980 1/16 印张: 27.75

字数: 532.8 千字

版 次: 2018 年 5 月第 1 版

印 次: 2018 年 5 月第 1 次印刷

定 价: 79.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888, 88258888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式: 010-51260888-819, faq@phei.com.cn。

前言

本书介绍

本书的书名是《深入剖析 Android 新特性》。“新”和“旧”是相对的，这样的词是有时效性的，今天新的东西，很快就不新了，本书介绍的是 Android 系统 5.0 至 8.0 的新增特性。

Android 作为一个跨越了多种设备的软件平台，由于各家厂商对于设备支持的速度不一，市场上的碎片化现象很严重。在 Android 7.0、8.0 推出的时候，很多用户可能还在使用 4.4 甚至更早的版本。而 Android 的新版本从推出到最终真正普及，需要一个较长的过渡时期。

这个现象对开发来说既有好处也有坏处。好处是开发者有足够多的时间为新版本的功能和特性做准备，坏处是开发者在开发应用时需要兼顾太多的版本。

本书在 Android 8.0 推出不久后上市，就是希望能给开发者们对近几年 Android 的新特性做一个梳理。因为笔者觉得，随着时间的推移，以及 Android 新版本的逐步普及，这些内容是开发者在最近几年正好需要的。

另外，这不是一本仅仅告诉读者 Android SDK 中的 API 如何使用的书。在本书中，我们会花更多的精力在这些特性的内部实现上，通过解析 AOSP 的源码，让读者不仅知其然，也知其所以然。

Android 是一个开源的操作系统，任何人都可以获取和阅读其源码。笔者认为，阅读 Android 源码有如下好处：

- 加深对 Android 系统的理解；
- 对 Android SDK 提供的 API 有更深入的理解；
- 提升自己的设计和架构水平；
- 在应对一些 Android 定制版本上的特有问题时，能够明白背后可能发生了什么。

合适的读者

本书适合以下读者群：

- Android 应用程序开发者；
- Android 系统工程师；
- 对 Android 系统内部实现感兴趣的读者；
- 对 Android 系统最新功能感兴趣的读者。

本书会包含什么

本书会在 AOSP 源码的基础上讲解 Android 系统的新增特性，主要集中在 5.0~8.0 版本。本书首先会对 Android 系统做一个整体的介绍，并对最关键的知识做一些说明。之后，会将大部分精力集中在讲解 Android 系统 5.0 至 8.0 的新增特性上。

在讲解这些特性的时候，我们会结合 AOSP 的源码，尽可能深入到系统的内部实现中，让读者不仅知其然，也知其所以然。

任何一本书的内容都凝聚了很多人的经验，本书在写作过程中也参考了很多的资料，为了对这些资料的原作者表示尊重，也为了让读者可以在更大的范围内去进行探索和学习，在每个章节的结尾，都尽可能会包含“参考资料与推荐读物”，这些内容是笔者在写作过程中参考的资料，或者是笔者认为对读者有帮助的信息。

任何操作系统的实现都是一个极其庞大的工程，Android 系统尤其如此。因此，任何一本书都不可能穷其所有细节，本书自然也不例外。就连 5.0 至 8.0 的新增功能中，我们也只能介绍其凤毛麟角。

但本书最大的目的在于：希望通过对新增功能的解析，让读者掌握 Android 的系统架构模型，以及阅读 AOSP 源码的能力，并最终具有能够自行研究 Android 系统的能力。

“授人以鱼不如授人以渔”，任何已有的功能都是固定的，随着行业的发展，Android 系统在未来可能加入的功能是不确定的，只有掌握了自己阅读和分析 Android 源码的能力，才能应对不确定的变化。

本书不会包含什么

下面这些内容，由于它们本身都是非常大的话题，可以单独写成一本书（甚至几本书），因

此这些内容在本书中不会讲解。

- 任何编程语言方面的知识;
- Linux/UNIX 系统开发知识;
- Linux 内核开发知识;
- 浏览器内核开发知识 (Webkit、Chromium);
- Android App 基础开发知识。

阅读本书, 你不必懂 Linux 内核、浏览器的相关知识。但本书期待读者拥有 C/C++、Java 语言代码的阅读能力, 熟悉 Linux 环境, 并有基本的 Android App 开发知识。否则, 你可能要选择其他书来做一些准备了。

为什么要写这本书

是的, 市面上已经有太多的 Android 书籍了, 为什么还需要这一本呢? 在我决定写这本书之前, 我也这样问自己。

本书的编辑陈晓猛先生最初与我联系的时候, 是因为看到了我的个人博客。这是在我的博客上线仅仅三个月不到的时候。

说实话, 因为我本身工作很忙, 写博客都要使劲才能挤出时间, 所以就更加没有时间去推广。我写博客的目的仅仅是为了自己积累一些东西。当然也希望我所记录下来的东西, 能对别人有帮助。

在陈晓猛先生与我联系之前, 我并没有想过自己会出一本书。但在看到他的留言之后, 我便在想, 如果我整理出来的知识能够让更多的人看到, 能够对更多的人产生帮助, 那不是很好吗? 于是我便尝试给他回了邮件。

出于以下理由, 让我决定写这本书:

- Android 平台拥有非常多的开发者, 开发者们需要更多的资料;
- AOSP 的源码是完全开放的, 但却缺少内部实现的设计文档和说明资料;
- 目前已有的书籍大部分是针对 Android 4.4 前后的版本, 需要有一本书针对新版本做一些整理;
- 每个人看问题和分析问题的方式不一样, 给更多的人分享自己的经验是很有意义的;
- 作为一个操作系统工程师, 对于同行的分析和研究也是我的工作内容之一。

“术业有专攻, 闻道有先后”, 做技术的过程本身就是一个互相学习和互相交流的过程。

由于笔者水平有限，文中若有错漏之处，也希望读者不吝赐教。

我的邮箱：paulquei@gmail.com。

我的个人主页：<http://qiangbo.space>。

本书的主要结构

本书包含的章节及每一章的内容介绍如下。

- 第 1 章：介绍 Android 系统的整体架构，并讲解如何获取 AOSP 源码，以及自己动手编译出可以运行的 Android 系统。
- 第 2 章：讲解 Android 系统中的进程管理，包括进程的创建、优先级管理，以及系统对于内存的管理。
- 第 3 章：讲解 Android 系统中的虚拟机，包括 Dalvik 虚拟机、ART 虚拟机。
- 第 4 章：讲解 Android 系统上用户界面的改进，包括多窗口功能和 App Shortcuts。
- 第 5 章：讲解 Android 系统上的 SystemUI 改进，包括 System Bar、Notification，以及 Quick Settings。
- 第 6 章：讲解 Android 系统上功耗方面的改进，包括 Project Volta、Doze 模式与 App StandBy。
- 第 7 章：讲解 Android 系统上设备管理方面的改进，包括对于多用户的支持和面向企业环境的 Android。
- 第 8 章：讲解 Android 系统安全方面的改进。
- 第 9 章：讲解 Android 系统在图形方面的改进，包括整个架构、主要组件，以及 Project Butter 等知识。
- 第 10 章：讲解 Android O 系统架构的改进——Project Treble。

致谢

这本书得以出版，我第一个要感谢的是南京富士通南大软件技术有限公司的朱清森部长。他曾是我的领导，正是他的引导，我才第一次尝试写作并投稿给 IBM DeveloperWorks 站点 (<https://www.ibm.com/developerworks/cn/java/j-lo-asm/>)，之后才有了我写的其他文章，以及我的个人博客，当然还有这本书。

第二个要感谢的是博文视点的编辑陈晓猛先生，是他的鼓励和辛苦编辑才使本书得以面世。

最后要感谢的是我在 AliOS 的同事和领导，从这群优秀的工程师身上我学到太多太多。在我写作本书的期间内，也得到了他们的很多帮助。

强波

2018 年 3 月于杭州

----- 读者服务 -----

轻松注册成为博文视点社区用户 (www.broadview.com.cn), 扫码直达本书页面。

- **提交勘误：**您对书中内容的修改意见可在 [提交勘误](#) 处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动：**在页面下方 [读者评论](#) 处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：<http://www.broadview.com.cn/33933>



目录

第 1 章 预备知识	1
1.1 Android 系统架构	1
1.2 关于 AOSP	3
1.3 理解 Android Binder 机制	3
1.3.1 Binder 机制简介	3
1.3.2 整体架构	4
1.3.3 驱动层	7
1.3.4 Binder Framework C++ 部分	26
1.3.5 Binder Framework Java 部分	54
1.3.6 参考资料与推荐读物	70
第 2 章 Android 系统中的进程管理	72
2.1 关于进程	72
2.2 系统进程与应用进程	73
2.2.1 init 进程与 init 语言	75
2.2.2 Zygote 进程	81
2.2.3 system_server 进程	84
2.3 应用进程的创建	87
2.3.1 关于应用组件	87
2.3.2 进程与线程	88
2.3.3 ActivityManagerService	89
2.3.4 Activity 与进程创建	92
2.3.5 Service 与进程创建	94

2.3.6	ContentProvider 与进程创建	96
2.3.7	BroadcastReceiver 与进程创建	97
2.3.8	参考资料与推荐读物	99
2.4	进程的优先级管理	100
2.4.1	优先级的依据	100
2.4.2	优先级的基础	102
2.4.3	优先级的更新	105
2.4.4	优先级的算法	108
2.4.5	优先级的生效	119
2.4.6	结束语	119
2.4.7	参考资料与推荐读物	120
2.5	进程与内存的回收	120
2.5.1	开发者 API	121
2.5.2	虚拟机的垃圾回收	122
2.5.3	开发者的内存回收	122
2.5.4	Linux OOM Killer	123
2.5.5	LowMemoryKiller	125
2.5.6	进程的死亡处理	126
2.5.7	参考资料与推荐读物	127
2.6	结束语	127
第 3 章 Android 系统上的虚拟机		129
3.1	Java 语言与 Java 虚拟机	130
3.1.1	Java 语言	130
3.1.2	Java 虚拟机	130
3.1.3	Java 虚拟机实现架构	130
3.1.4	类加载器 (Class loader)	132
3.1.5	垃圾回收	132
3.1.6	结束语	135
3.1.7	参考资料与推荐读物	135
3.2	Dalvik 虚拟机	135
3.2.1	Stack-based VS. Register-based	136

3.2.2	Dalvik Executable(dex)文件	137
3.2.3	Dalvik 指令	139
3.2.4	Dalvik 启动过程	144
3.2.5	程序的执行：解释与编译	153
3.2.6	Dalvik 上的垃圾回收	156
3.2.7	参考资料与推荐读物	159
3.3	Android Runtime (ART)	159
3.3.1	ART VS.Dalvik.....	159
3.3.2	OAT 文件格式.....	162
3.3.3	boot.oat 与 boot.art.....	165
3.3.4	Dalvik 到 ART 的切换.....	166
3.3.5	ART 虚拟机的启动过程.....	169
3.3.6	内存分配	173
3.3.7	垃圾回收	178
3.3.8	JIT 的回归	182
3.3.9	ART 虚拟机的演进与配置.....	185
3.3.10	参考资料与推荐读物	186
第 4 章	用户界面改进.....	187
4.1	多窗口功能	187
4.1.1	概述	188
4.1.2	开发者相关	189
4.1.3	内部实现	191
4.1.4	参考资料与推荐读物	205
4.2	App Shortcuts	206
4.2.1	开发者 API.....	206
4.2.2	内部实现	209
第 5 章	系统界面改进.....	215
5.1	SystemUI 整体介绍	215
5.1.1	SystemUI 简介	215
5.1.2	SystemUI 的初始化	216

5.1.3	System Bar 的初始化.....	221
5.1.4	参考资料与推荐读物	224
5.2	System Bar	224
5.3	Notification	234
5.3.1	开发者 API.....	234
5.3.2	通知栏与通知窗口	240
5.3.3	Notification 从发送到显示	243
5.4	Quick Settings.....	256
5.4.1	开发者 API.....	256
5.4.2	系统实现	260
5.4.3	参考资料与推荐读物	265
第 6 章	功耗的改进	266
6.1	Project Volta.....	266
6.1.1	JobScheduler API	267
6.1.2	电量消耗分析工具	277
6.1.3	在虚拟机层面减少电池消耗	279
6.1.4	省电模式	280
6.1.5	结束语	281
6.1.6	参考资料与推荐读物	281
6.2	Doze 模式与 App StandBy	282
6.2.1	概述	282
6.2.2	了解 Doze 模式.....	282
6.2.3	了解 App StandBy.....	284
6.2.4	对其他用例的支持	284
6.2.5	在 Doze 模式和 App StandBy 下进行测试.....	285
6.2.6	Doze 模式的实现.....	287
6.2.7	App StandBy 的实现.....	293
6.2.8	参考资料与推荐读物	297
6.3	Android 8.0 上的后台限制	297
6.3.1	概览	298
6.3.2	后台服务限制	299

6.3.3	广播限制	300
6.3.4	系统实现	301
6.3.5	结束语	308
第 7 章 面向设备管理的改进		310
7.1	多用户的支持	310
7.1.1	术语	311
7.1.2	支持多用户	312
7.1.3	多用户的实现	313
7.1.4	参考资料与推荐读物	333
7.2	设备管理	334
7.2.1	Device Administration API 介绍	334
7.2.2	开发设备管理程序	335
7.3	面向企业环境的 Android	339
7.3.1	企业环境解决方案	340
7.3.2	受管理资料 (Managed Profiles)	340
7.3.3	受管理资料的内部实现	342
7.3.4	参考资料与推荐读物	349
第 8 章 Android 系统安全改进		350
8.1	Android 系统安全概览	350
8.2	运行时权限	351
8.2.1	功能介绍	351
8.2.2	新增 API	354
8.2.3	参考资料与推荐读物	370
第 9 章 图形系统改进		371
9.1	整体架构	371
9.1.1	Android 图形组件	371
9.1.2	组件	373
9.1.3	Android 如何绘制视图	374
9.1.4	关于硬件加速	376

9.1.5	参考资料与推荐读物	377
9.2	图形系统组件	377
9.2.1	Activity 与 Surface	377
9.2.2	Gralloc	379
9.2.3	BufferQueue	382
9.2.4	Surface	384
9.2.5	参考资料与推荐读物	388
9.3	Project Butter	388
9.3.1	FPS	389
9.3.2	VSYNC	391
9.3.3	Choreographer 与 VSYNC	392
9.3.4	Triple Buffer	398
9.3.5	参考资料与推荐读物	400
9.4	SurfaceFlinger	400
9.4.1	SurfaceFlinger 介绍	401
9.4.2	Hardware Composer	402
9.4.3	SurfaceFlinger 的启动	403
9.4.4	SurfaceFlinger 的对外接口	405
9.4.5	VSYNC 的传递	407
9.4.6	SurfaceFlinger 的事件	409
9.4.7	图层的合成	411
9.4.8	刷新	412
9.4.9	参考资料与推荐读物	413
9.5	Vulkan 简介	414
9.5.1	Vulkan 组件	414
9.5.2	修改的组件	415
9.5.3	Vulkan API	415
9.5.4	参考资料与推荐读物	415
第 10 章	系统架构改进	416
10.1	Project Treble 整体介绍	417
10.2	HIDL	420

10.2.1	语法介绍	421
10.2.2	HIDL 与 Binder	422
10.3	ConfigStore HAL	423
10.3.1	概述	423
10.3.2	内部实现	424
10.4	Vendor Native Development Kit	426
10.5	Vendor Interface Object	427
10.5.1	概述	427
10.5.2	Manifest	429
10.5.3	内部实现	429

1 chapter

第 1 章 预备知识

本章是全书正文的第 1 章，会对后文需要的知识做一些必要性的准备。其中包括：

- 对 Android 系统架构做一个整体性的介绍；
- 对 AOSP 做一定的说明；
- 对 Android 系统中大量使用的 Binder 机制做深入的讲解。

1.1 Android 系统架构

大型的软件通常都采用分层的架构模式，Android 系统也不例外。图 1-1 是 Android 系统的整体架构。

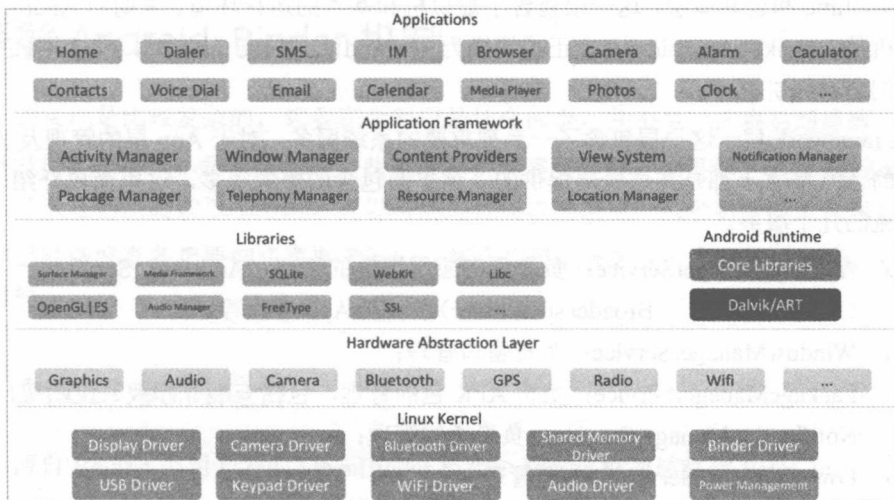


图 1-1 Android 系统架构

从这幅图中可以看出，Android 操作系统架构共分为五层。我们从下至上来了解一下：

- 内核层，这一层包含了 Linux 内核以及 Android 在 Linux 内核上增加的驱动。这些驱动通常与硬件无关，而是为了上层软件服务的，它们包括以下内容。
 - Binder：进程间通信（IPC）基础设施。Binder 在 Android 系统中大量使用，几乎所有的 Framework 层的服务都是通过 Binder 的形式暴露出接口供外部使用的。
 - Ashmem 匿名共享内存。共享内存的作用是，当多个进程需要访问同一块数据时，可以避免数据复制。例如，经由 ContentProvider 接口获取数据的客户端和 ContentProvider 之间就是通过共享内存的方式来访问的。
 - lowmemorykiller 进程回收模块。在 Framework 层，所有的应用进程都是由 ActivityManagerService 来管理的，它会根据进程的重要性设置一个优先级，这个优先级会被 lowmemorykiller 读取。在系统内存较低时，lowmemorykiller 会根据进程的优先级排序，将优先级低的进程杀死，直到系统恢复到合适的内存状态。
 - logger 日志相关。开发人员经常会使用 logcat 读取日志来帮助分析问题。而无论是 logcat 工具，还是通过日志 API 写入日志，最终都是由底层的 logger 驱动进行处理的。
 - wakelock 电源管理相关。Android 系统通常运行在以电池供电的移动设备上，因此专门增加了该模块来管理电源。
 - Alarm 闹钟相关，为 AlarmManager 服务。
- 硬件抽象层，该层为硬件厂商定义了一套标准的接口。有了这套标准接口之后，可以在不影响上层的情况下，调整内部实现。
- Runtime 和公共库层，这一层包含了虚拟机和基本的运行环境。早期的 Android 的虚拟机是 Dalvik，Android 5.0 上正式切换为 Android RunTime（ART），在虚拟机一章中我们会详细讲解。
- Framework 层，这一层包含了一系列重要的系统服务。对于 App 层的管理及 App 使用的 API 基本上都是在这一层提供的。这里面包含的服务很多，这里简单介绍一下最常见的几个服务。
 - ActivityManagerService：负责所有应用组件的管理（Activity、Service、ContentProvider、BroadcastReceiver），以及 App 进程管理；
 - WindowManagerService：负责窗口管理；
 - PackageManagerService：负责 APK 包的管理，包括安装、卸载、更新等；
 - NotificationManagerService：负责通知管理；
 - PowerManagerService：电源管理；
 - LocationManagerService：定位相关。