本书基于Android 5.0版本，对Android开发进阶要点进行深入讲解，是高级工程师成长之路上的必备利器！

Broadview®
www.broadview.com.cn

Android 开发艺术探索

任玉刚 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Android 开发艺术探索

任玉刚 著

電子工業出版社
Publishing House of Electronics Industry

内 容 简 介

本书是一本 Android 进阶类书籍，采用理论、源码和实践相结合的方式来阐述高水准的 Android 应用开发要点。本书从三个方面来组织内容。第一，介绍 Android 开发者不容易掌握的一些知识点；第二，结合 Android 源代码和应用层开发过程，融会贯通，介绍一些比较深入的知识点；第三，介绍一些核心技术和 Android 的性能优化思想。

本书侧重于 Android 知识的体系化和系统工作机制的分析，通过本书的学习可以极大地提高开发者的 Android 技术水平，从而更加高效地成为高级开发者。而对于高级开发者来说，仍然可以从本书的知识体系中获益。

图书在版编目（CIP）数据

Android 开发艺术探索/任玉刚著. —北京：电子工业出版社，2015.9
ISBN 978-7-121-26939-4

Ⅰ. ①A… Ⅱ. ①任… 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（2015）第 189069 号

责任编辑：陈晓猛
印　　刷：中国电影出版社印刷厂
装　　订：三河市华成印务有限公司
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：32.75 字数：733 千字
版　　次：2015 年 9 月第 1 版
印　　次：2015 年 9 月第 2 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。

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。

服务热线：（010）88258888。

序 言

与玉刚共事两年，其对技术的热情和执著让人敬佩，其技术进步之快又让人惊叹。如今，他把所掌握的知识与经验成书出版，是一件大幸之事：于作者，此书是他的心血所成，可喜可贺；于读者，可解“工作视野”之困与“百思不得其解”之惑，或许有“啊哈，原来如此”之效，又或许有“技能+1”之得意一笑。

玉刚拥有丰富的 Android 开发经验，对 Android 开发的很多知识点都有深入研究，我相信此书定能为读者带来惊喜。书的内容，大抵有如下几方面：基础知识点之深入理解（例如，Activity 的生命周期和启动模式、Android 的消息机制分析、View 的事件体系、View 的工作原理等章节）；不常见知识点的分析（例如，IPC 机制、理解 Window 和 WindowManager 等章节）；工程实践中的经验（例如，综合技术、Android 性能优化等章节）。因此，此书读者需要有一定的 Android 开发基础和工程经验，否则读起来会比较吃力或者感觉云里雾里。对于想成长为高级或者资深 Android 研发的工程师，书中的知识点都是需要掌握的。

最后，希望读者能够从此书获益，接触到一些工作中未曾了解或者思考的知识点。更进一步，希望读者能够活学活用，并学习此书背后的钻研精神。

涂勇策

百度手机卫士　资深工程师

前 言

从目前的形势来看，Android 开发相当火热，但是高级 Android 开发人才却比较少，当然在国内，不仅仅是 Android，其他技术岗位同样面临这个问题。试想下，如果有一本书能够切实有效地提高开发者的技术水平，那该多好啊！纵观市场上的 Android 书籍，很多都是入门类书籍，还有一些 Android 源码分析、系统移植、驱动开发、逆向工程等系统底层类书籍。入门类书籍是目前图书市场中的中坚力量，它们在帮助开发者入门的过程中起到了非常重要的作用，但开发者若想进一步提高技术水平，还需要阅读更深入的书籍。底层书籍包括源码分析、驱动开发、逆向工程等书籍，它们从底层或者某一个特殊的角度来深入地分析 Android，这是很值得称赞和学习的，通过这些书可以极大地提高开发者底层或者相关领域的技术水平。但美中不足的是，系统底层书籍比较偏理论，部分开发者阅读起来可能会有点晦涩难懂。更重要的一点，由于它们往往侧重原理和底层机制，导致它们不能直接为应用层开发服务，毕竟绝大多数 Android 开发岗位都是应用层开发。由于阅读底层类书籍一般只能够加深对底层的认识，而在应用层开发中，还是不能形成直接有效的战斗力，这中间是需要转化过程的。但是，由于部分开发者缺乏相应的技术功底，导致无法完成这个转化过程。

可以发现，目前市场上既能够极大地提高开发者的应用层技术经验，又能够将上层和系统底层的运行机制结合起来的书籍还是比较少的。对企业来说，在业务上有很强的技术能力，同时对 Android 底层也有一定理解的开发人员，是企业比较青睐的技术高手。为了完成这一愿望，笔者写了这本书。通过对本书的深入学习，开发者既能够极大地提高应用层的开发能力，又能够对 Android 系统的运行机制有一定的理解，但如果要深入理解 Android 的底层机制，仍然需要查看相关源码分析的书籍。

本书适合各类开发者阅读，对于初、中级开发者来说，可以通过本书更加高效地达到高级开发者的技术水平。而对于高级开发者，仍然可以从本书的知识体系中获益。本书的书名之所以采用艺术这个词，这是因为在笔者眼中，代码写到极致就是一种艺术。

本文内容

本书共 15 章，所讲述的内容均基于 Android 5.0 系统。

第 1 章介绍 Activity 的生命周期和启动模式以及 IntentFilter 的匹配规则。

第 2 章介绍 Android 中常见的 IPC 机制，多进程的运行模式和一些常见的进程间通信方式，包括 Messenger、AIDL、Binder 以及 ContentProvider 等，同时还介绍 Binder 连接池的概念。

第 3 章介绍 View 的事件体系，并对 View 的基础知识、滑动以及弹性滑动做详细的介绍，同时还深入分析滑动冲突的原因以及解决方法。

第 4 章介绍 View 的工作原理，首先介绍 ViewRoot、DecorView、MeasureSpec 等 View 相关的底层概念，然后详细分析 View 的测量、布局和绘制三大流程，最后介绍自定义 View 的分类以及实现思想。

第 5 章讲述一个不常见的概念 RemoteViews，分别描述 RemoteViews 在通知栏和桌面小部件中的使用场景，同时还详细介绍 PendingIntent，最后深入分析 RemoteViews 的内部机制并探索性地指出 RemoteViews 在 Android 中存在的意义。

第 6 章对 Android 的 Drawable 做一个全面性的介绍，除此之外还讲解自定义 Drawable 的方法。

第 7 章对 Android 中的动画做一个全面深入的分析，包含 View 动画和属性动画。

第 8 章讲述 Window 和 WindowManager，首先分析 Window 的内部工作原理，包括 Window 的添加、更新和删除，其次分析 Activity、Dialog 等类型的 Window 对象的创建过程。

第 9 章深入分析 Android 中四大组件的工作过程，主要包括四大组件的运行状态以及它们主要的工作过程，比如启动、绑定、广播的发送和接收等。

第 10 章深入分析 Android 的消息机制，其中涉及的概念有 Handler、Looper、MessageQueue 以及 ThreadLocal，此外还分析主线程的消息循环模型。

第 11 章讲述 Android 的线程和线程池，首先介绍 AsyncTask、HandlerThread、IntentService 以及 ThreadPoolExecutor 的使用方法，然后分析它们的工作原理。

第 12 章讲述的主题是 Bitmap 的加载和缓存机制，首先讲述高效加载图片的方式，接着介绍 LruCache 和 DiskLruCache 的使用方法，最后通过一个 ImageLoader 的实例来将它们综合起来。

第 13 章是综合技术，讲述一些很重要但是不太常见的技术方案，它们是 CrashHandler、multidex、插件化以及反编译。

第 14 章的主题是 JNI 和 NDK 编程，介绍使用 JNI 和 Android NDK 编程的方法。

第 15 章介绍 Android 的性能优化方法，比如常见的布局优化、绘制优化、内存泄露优化等，除此之外还介绍分析 ANR 和内存泄露的方法，最后探讨如何提高程序的可维护性这一话题。

通过这 15 章的学习，可以让初、中级开发者的技术水平和把控能力提升一个档次，最终成为高级开发者。

本书特色

本书定位为进阶类图书，不会对一些基础知识从头说起，或者说每一章节都不涵盖各种入门知识，但是在向高级知识点过渡的时候，会稍微提及一下基础知识从而做到平滑过渡。开发者在掌握入门知识以后，通过本书可以极大地提高应用层开发的技术水平，同时还可以理解一定的 Android 底层运行机制，并且能够将它们进行升华从而更好地为应用层开发服务。除了这些，开发者还可以掌握一些核心技术和性能优化思想，本书涉及的知识，都是一个合格的高级工程师所必须掌握的。简单地说，本书的目的就是让初、中级开发者更有针对性地掌握高级工程师所应该掌握的技术，能够让初、中级开发者按照正确的道路快速地成长为高级工程师。

致谢

感谢本书的策划编辑陈晓猛，他的高效率是本书得以及时出版的一个重要原因；感谢我的妻子对我写书的支持，接近 1 年的写书时光是她一直陪伴在我身边；感谢百度手机卫士这款产品，它是本书的技术源泉；感谢和我一起奋斗的同事们，和你们在一起工作的时光，我不仅提高了技术水平而且还真正感受到了一种融洽的工作氛围；还要感谢所有关注我的朋友们，你们的鼓励和认可是我前进的动力。

由于技术水平有限，书中难免会有错误，欢迎大家向我反馈：singwhatiwanna@gmail.com，也可以关注我的 CSDN 博客，我会定期在上面发布本书的勘误信息。

本书互动地址

CSDN 博客：http://blog.csdn.net/singwhatiwanna

Github：https://github.com/singwhatiwanna

QQ 交流群：481798332

微信公众号：Android 开发艺术探索

书中源码下载地址：

https://github.com/singwhatiwanna/android-art-res

或者

www.broadview.com.cn/26939

任玉刚

2015 年 6 月于北京

目 录

第1章 Activity的生命周期和启动模式

作为本书的第1章，本章主要介绍Activity相关的一些内容。Activity作为四大组件之首，是使用最为频繁的一种组件，中文直接翻译为“活动”，但是笔者认为这种翻译有些生硬，如果翻译成界面就会更好理解。正常情况下，除了Window、Dialog和Toast，我们能见到的界面的确只有Activity。Activity是如此重要，以至于本书开篇就不得不讲到它。当然，由于本书的定位为进阶书，所以不会介绍如何启动Activity这类入门知识，本章的侧重点是Activity在使用过程中的一些不容易搞清楚的概念，主要包括生命周期和启动模式以及IntentFilter的匹配规则分析。其中Activity在异常情况下的生命周期是十分微妙的，至于Activity的启动模式和形形色色的Flags更是让初学者摸不到头脑，就连隐式启动Activity中也有着复杂的Intent匹配过程，不过不用担心，本章接下来将一一解开这些疑难问题的神秘面纱。

1.1 Activity的生命周期全面分析

本节将Activity的生命周期分为两部分内容，一部分是典型情况下的生命周期，另一部分是异常情况下的生命周期。所谓典型情况下的生命周期，是指在有用户参与的情况下，Activity所经过的生命周期的改变；而异常情况下的生命周期是指Activity被系统回收或者由于当前设备的Configuration发生改变从而导致Activity被销毁重建，异常情况下的生命周期的关注点和典型情况下略有不同。

1.1.1 典型情况下的生命周期分析

在正常情况下，Activity 会经历如下生命周期。

（1）onCreate：表示 Activity 正在被创建，这是生命周期的第一个方法。在这个方法中，我们可以做一些初始化工作，比如调用 setContentView 去加载界面布局资源、初始化 Activity 所需数据等。

（2）onRestart：表示 Activity 正在重新启动。一般情况下，当当前 Activity 从不可见重新变为可见状态时，onRestart 就会被调用。这种情形一般是用户行为所导致的，比如用户按 Home 键切换到桌面或者用户打开了一个新的 Activity，这时当前的 Activity 就会暂停，也就是 onPause 和 onStop 被执行了，接着用户又回到了这个 Activity，就会出现这种情况。

（3）onStart：表示 Activity 正在被启动，即将开始，这时 Activity 已经可见了，但是还没有出现在前台，还无法和用户交互。这个时候其实可以理解为 Activity 已经显示出来了，但是我们还看不到。

（4）onResume：表示 Activity 已经可见了，并且出现在前台并开始活动。要注意这个和 onStart 的对比，onStart 和 onResume 都表示 Activity 已经可见，但是 onStart 的时候 Activity 还在后台，onResume 的时候 Activity 才显示到前台。

（5）onPause：表示 Activity 正在停止，正常情况下，紧接着 onStop 就会被调用。在特殊情况下，如果这个时候快速地再回到当前 Activity，那么 onResume 会被调用。笔者的理解是，这种情况属于极端情况，用户操作很难重现这一场景。此时可以做一些存储数据、停止动画等工作，但是注意不能太耗时，因为这会影响到新 Activity 的显示，onPause 必须先执行完，新 Activity 的 onResume 才会执行。

（6）onStop：表示 Activity 即将停止，可以做一些稍微重量级的回收工作，同样不能太耗时。

（7）onDestroy：表示 Activity 即将被销毁，这是 Activity 生命周期中的最后一个回调，在这里，我们可以做一些回收工作和最终的资源释放。

正常情况下，Activity 的常用生命周期就只有上面 7 个，图 1-1 更详细地描述了 Activity 各种生命周期的切换过程。

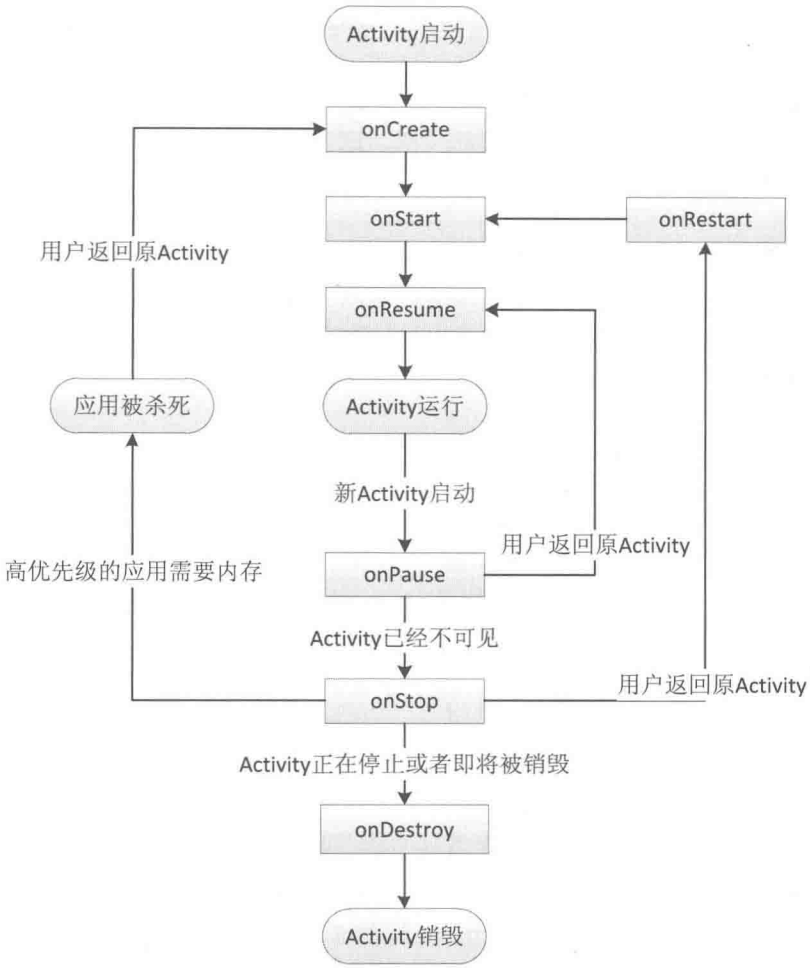

图 1-1 Activity 生命周期的切换过程

针对图 1-1，这里再附加一下具体说明，分如下几种情况。

（1）针对一个特定的 Activity，第一次启动，回调如下：onCreate -> onStart -> onResume。

（2）当用户打开新的 Activity 或者切换到桌面的时候，回调如下：onPause -> onStop。这里有一种特殊情况，如果新 Activity 采用了透明主题，那么当前 Activity 不会回调 onStop。

（3）当用户再次回到原 Activity 时，回调如下：onRestart -> onStart -> onResume。