<<Android系统源代码情景分析>>

图书基本信息

书名：<<Android系统源代码情景分析>>

13位ISBN编号：9787121181085

10位ISBN编号：7121181088

出版时间：2012-10

出版时间：电子工业出版社

作者：罗升阳

页数：830

字数：1570000

前言

虽然Android系统自2008年9月发布第一个版本1.0以来，截至2011年10月发布最新版本4.0，一共存在十多个版本，但是据官方统计，截至2012年3月5日，占据首位的是Android 2.3，市场占有率达到66.5%；其次是Android 2.2，市场占有率为25.3%；第三位是Android 2.1，市场占有率为6.6%；而最新发布的Android 3.2和Android 4.0的市场占有率只有3.3%和2%。
因此，在本书中，我们选择了Android 2.3的源代码来分析Android系统的实现，一是因为从目前来说，它的基础架构是最稳定的；二是因为它是使用最广泛的。
本书内容全书分为初识Android系统篇、Android专用驱动系统篇和Android应用程序框架篇三个部分。
初识Android系统篇包含三个章节的内容，主要介绍Android系统的基础知识。
第1章介绍与Android系统有关的参考书籍，以及Android源代码工程环境的搭建方法；第2章介绍Android系统的硬件抽象层；第3章介绍Android系统的智能指针。
读者可能会觉得奇怪，为什么一开始就介绍Android系统的硬件抽象层呢？
因为涉及硬件，它似乎是一个深奥的知识点。
其实不然，Android系统的硬件抽象层无论是从实现上，还是从使用上，它的层次都是非常清晰的，而且从下到上涵盖了整个Android系统，包括Android系统在用户空间和内核空间的实现。
内核空间主要涉及硬件驱动程序的编写方法，而用户空间涉及运行时库层、应用程序框架层及应用程序层。
因此，尽早学习Android系统的硬件抽象层，有助于我们从整体上去认识Android系统，以便后面可以更好地分析它的源代码。
在分析Android系统源代码的过程中，经常会碰到智能指针，第3章我们就重点分析Android系统智能指针的实现原理，也是为了后面可以更好地分析Android系统源代码。
Android专用驱动系统篇包含三个章节的内容。
我们知道，Android系统是基于Linux内核来开发的，但是由于移动设备的CPU和内存配置都要比PC低，因此，Android系统并不是完全在Linux内核上开发的，而是在Linux内核里面添加了一些专用的驱动模块来使它更适合于移动设备。
这些专用的驱动模块同时也形成了Android系统的坚实基础，尤其是Logger日志驱动程序、Binder进程间通信驱动程序，以及Ashmem匿名共享内存驱动程序，它们在Android系统中被广泛地使用。
在此篇中，我们分别在第4章、第5章和第6章分析Logger日志系统、Binder进程间通信系统和Ashmem共享内存系统的实现原理，为后面深入分析Android应用程序的框架打下良好的基础。
Android应用程序框架篇包含十个章节的内容。
我们知道，在移动平台中，Android系统、iOS系统和Windows Phone系统正在形成三足鼎立之势，谁的应用程序更丰富、质量更高、用户体验更好，谁就能取得最终的胜利。
因此，每个平台都在尽最大努力吸引第三方开发者来为其开发应用程序。
这就要求平台必须提供良好的应用程序架构，以便第三方开发者可以将更多的精力集中在应用程序的业务逻辑上，从而开发出数量更多、质量更高和用户体验更好的应用程序。
在此篇中，我们将从组件、进程、消息和安装四个维度来分析Android应用程序的实现框架。
第7章到第10章分析Android应用程序四大组件Activity、Service、Broadcast Receiver和Content Provider的实现原理；第11章和第12章分析Android应用程序进程的启动过程；第13章到第15章分析Android应用程序的消息处理机制；第16章分析Android应用程序的安装和显示过程。
学习了这些知识之后，我们就可以掌握Android系统的精髓了。
本书特点本书从初学者的角度出发，结合具体的使用情景，在纵向和横向上对Android系统的源代码进行了全面、深入、细致的分析。
在纵向上，采用从下到上的方式，分析的源代码涉及了Android系统的内核层（Linux Kernel）、硬件抽象层（HAL）、运行时库层（Runtime）、应用程序框架层（Application Framework）以及应用程序层（Application），这有利于读者从整体上掌握Android系统的架构。
在横向上，从Android应用程序的组件、进程、消息以及安装四个角度出发，全面地剖析了Android系

统的应用程序框架层，这有利于读者深入地理解Android应用程序的架构以及运行原理。
作　者

内容概要

在内容上，本书结合使用情景，全面、深入、细致地分析Android系统的源代码，涉及到Linux内核层、硬件抽象层（HAL）、运行时库层(Runtime)、应用程序框架层(Application Framework)以及应用程序层(Application)。
在组织上，本书将上述内容划分为初识Android系统、Android专用驱动系统和Android应用程序框架三大篇章。
初识Android系统篇介绍了参考书籍、基础知识以及实验环境搭建；Android专用驱动系统篇介绍了Logger日志驱动程序、Binder进程间通信驱动程序以及Ashmem匿名共享内存驱动程序；Android应用程序框架篇从组件、进程、消息以及安装四个维度来对Android应用程序的框架进行了深入的剖析。
通过上述内容及其组织，本书能使读者既能从整体上把握Android系统的层次结构，又能从细节上去掌握每一个层次的要点。

作者简介

罗升阳，1984年出生，2007年毕业于浙江大学计算机系，取得学士学位，2010年毕业于上海交通大学计算机系，取得硕士学位。
毕业后一直从事于互联网软件开发，并且致力于移动平台的研究，特别是对Android平台有深入的理解和研究。
在国内知名IT技术社区CSDN上发表了数十篇高质量的Android系统原创性文章，并且开设博客专栏--《老罗的Android之旅》，积极与网友互动，深受大家喜爱，访问量一直居于前茅。

<<Android系统源代码情景分析>>

书籍目录

章节摘录

版权页： 插图： 成员变量proc指向一个Binder实体对象的宿主进程。
在Binder驱动程序中，这些宿主进程通过一个binder_proc结构体来描述。
宿主进程使用一个红黑树来维护它内部所有的Binder实体对象，而每一个Binder实体对象的成员变量rb_node就正好是这个红黑树中的一个节点。
如果一个Binder实体对象的宿主进程已经死亡了，那么这个Binder实体对象就会通过它的成员变量dead_node保存在一个全局的hash列表中。
由于一个Binder实体对象可能会同时被多个Client组件引用，因此，Binder驱动程序就使用结构体binder_ref来描述这些引用关系，并且将引用了同一个Binder实体对象的所有引用都保存在一个hash列表中。
这个hash列表通过Binder实体对象的成员变量refs来描述，而Binder驱动程序通过这个成员变量就可以知道有哪些Client组件引用了同一个Binder实体对象。
成员变量internal_strong_refs和local_Strong_refs均是用来描述一个Binder实体对象的强引用计数，而成员变量local_weak_refs用来描述一个Binder实体对象的弱引用计数。
当一个Binder实体对象请求一个Service组件来执行某一个操作时，会增加该Service组件的强引用计数或者弱引用计数，相应地，Binder实体对象会将其成员变量has_strong_ref和has_weak_ref的值设置为1。
当一个Service组件完成一个Binder实体对象所请求的操作之后，Binder实体对象就会请求减少该Service组件的强用计数或者弱引用计数。
Binder实体对象在请求一个Service组件增加或者减少强引用计数或者弱引用计数的过程中，会将其成员变量pending_strong_ref或者pending_weak_ref的值设置为1；而当该Service组件增加或者减少了强引用计数或者弱引用计数之后，Binder实体对象就会将这两个成员变量的值设置为0。

媒体关注与评论

强大的罗大师，感谢你的Android之旅，我已看完你的6篇教程，虽然我目前还不太懂，但是我相信你就是我打开它的那把钥匙，再次感谢你的教程。
——@mutex_js老罗，写的太精彩啦！
不仅仅是在Android 按键机制方面令人受益匪浅，给我在学习其他模块上也提供了一套很好的分析思路。
非常感谢！
——@yuleslie看你的文章，收获很多，分析得很透彻，思路清晰，前后呼应，成系统，对我帮助很大，非常感谢你的无私奉献！
——@kenen2006你的博客给了我一种非常刺激的体验，让我更深层次地认识Android。
非常感谢你的讲解，太棒了！
——@stevenhu_223我真的想放弃这个行业，可我看到你的博客，让我看到了希望，让我充满了信心和坚定！
——@zhudeqing看过几本Andriod方面的书，但还真比不上这里的博客。
罗老师的Linux内核知识及对软件架构的知识积累用功很深，代码阅读能力也很强！
膜拜！
——@herodie我是看你的博客开始学Android的，写的真的太好了。
每一个系列都自成一体，无需其他参考。
从顶到下都能串到一起，这样看起来最痛快了！
——@hellowolrd本人语言功底也不是很好，所以系统学习Android也经历了不少困难。
3个多月了，反反复复阅读你的博文；每一次都有进步，每一次的进步也都有不同的方向。
谢谢！
技术的道路上真没有捷径，更不可以偷懒。
我将不懈努力！
谢谢博主，期待您能出书！
——@tankai19880619刚看到老罗的这篇新作，还挺热乎的。
忍不住分N口气读完，膜拜！
读完后，有些有意思的想法，从文章本身来看，很精彩。
这篇文章老罗至少说明了一下问题：1. 两种最典型的使用场景及他们的区别（UI相关与否）；2. 引出来两种线程消息处理模型；3. 用实例解释两种模型是如何使用的；4. 最后介绍两种模型的是怎样实现的。
该文章不但告诉我们what（该文的主旨）、how (怎样使用），还有why（怎么实现）。
偶的神，一篇文章里写了这么多，还能写的这么清楚。
不容易啊！
——@rambo2188太牛了！
我是一个刚从事Android开发的新人，你的文章 真是入木三分，读了很有启发，你的这种分析Android方式很棒！
为了造福更多的Android学习者，建议你出书！
——@yang105我觉得也许学习这些知识并不算难，可是要做到时刻有一个清晰的思路去学并且能够把学过东西用通俗易懂的话语表达出来挺难的。
必尽自身知识有限，有些东西可能一下子无法深刻理解。
楼主是怎么做到的呢？
感觉楼主的每篇文章都讲的精练透彻，主要是能把问题全讲出来。
我想楼主的知识面一定很广！
——@wantianpei

编辑推荐

《Android系统源代码情景分析》能使读者既能从整体上把握Android系统的层次结构，又能从细节上去掌握每一个层次的要点。

版权说明