

<<Android内核剖析>>

图书基本信息

书名：<<Android内核剖析>>

13位ISBN编号：9787121143984

10位ISBN编号：7121143984

出版时间：2011-9

出版时间：电子工业

作者：柯元旦

页数：595

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<Android内核剖析>>

内容概要

本书内容分别从基础、内核、系统、编译以及硬件驱动几个方面对Android内核相关知识进行深入剖析，详细分析了Android内核的内部机制，包括窗口管理系统、Activity管理系统、输入法框架、编译系统等，为Android内核定制以及高级应用程序开发提供技术参考。

本书适合于所有Android相关的工程师以及产品经理。

<<Android内核剖析>>

作者简介

柯元旦，1981年生于陕西咸阳，2003年毕业于西安电子科技大学通信工程学院；2003～2006年，创立了一家设计公司，提供便携式媒体播放器解决方案，基于TI DSP芯片自主开发了一套操作系统，包括任务管理与调度、文件系统及GUI(图形用户接口)等；2006～2009年，就职于联想研究院，先后从事嵌入式系统设计以及互联网应用开发；2009年至今，从事Android应用程序开发和Framework内核研究，对Android内核有较深刻的理解。

<<Android内核剖析>>

书籍目录

第1部分 基础篇

第1章 Linux基础

- 1.1 Linux文件系统概述
- 1.2 Linux启动过程
- 1.3 常用Linux 命令
- 1.4 Shell脚本备忘
 - 1.4.1 获取输入
 - 1.4.2 变量定义
 - 1.4.3 条件判断
 - 1.4.4 while []...do... done语句
 - 1.4.5 for循环
 - 1.4.6 函数
 - 1.4.7 常用内置符号常量
- 1.5 Make脚本备忘
 - 1.5.1 一个简单的Makefile文件
 - 1.5.2 变量的定义与赋值
 - 1.5.3 条件控制语句
 - 1.5.4 宏（函数）定义
 - 1.5.5 内置符号和变量
 - 1.5.6 模板目标（Pattern target）
 - 1.5.7 目标特定的变量赋值（Target-specific variable）
 - 1.5.8 常用选项

第2章 Java基础

- 2.1 类装载器DexClassLoader
 - 2.1.1 DexClassLoader的调用方法
 - 2.1.2 基于类装载器设计一种“插件”架构
- 2.2 JNI调用机制
 - 2.2.1 Java访问C
 - 2.2.2 C访问Java
 - 2.2.3 在C中使用持久对象
- 2.3 异步消息处理线程
 - 2.3.1 实现异步线程的一般思路
 - 2.3.2 Android中异步线程的实现方法

第3章 Android源码下载及开发环境配置

- 3.1 Mac系统的配置
 - 3.1.1 硬盘格式的配置
 - 3.1.2 port的用法
- 3.2 在Linux中配置USB连接
- 3.3 在Eclipse中调试Framework
 - 3.3.1 一段防止下载异常脚本
 - 3.3.2 调试Framework中的代码

第4章 使用git

- 4.1 安装git
- 4.2 git仓库管理
 - 4.2.1 仓库的组成

<<Android内核剖析>>

- 4.2.2 创建仓库
- 4.2.3 分支管理
- 4.3 git merge用法
- 4.4 git rebase用法
- 4.5 git cherry-pick用法
- 4.6 git reset用法
- 4.7 恢复到无引用提交
- 4.8 git remote用法
- 4.9 git 配置
 - 4.9.1 基本信息配置
 - 4.9.2 merge、diff工具配置
 - 4.9.3 .gitignore配置
- 4.10 同时使用git和svn
- 4.11 其他git常用命令示例
 - 4.11.1 git branch
 - 4.11.2 git checkout
 - 4.11.3 git log
 - 4.11.4 git commit --amend
 - 4.11.5 git cherry-pick sha-1
 - 4.11.6 git merge-base
 - 4.11.7 git diff master...dev
 - 4.11.8 git revert
 - 4.11.9 git diff
 - 4.11.10 git rm
 - 4.11.11 git tag

第2部分 内核篇

第5章 Binder

- 5.1 Binder框架
- 5.2 设计Servier端
- 5.3 Binder客户端设计
- 5.4 使用Service类
 - 5.4.1 获取Binder对象
 - 5.4.2 保证包裹内参数顺序aidl工具的使用
- 5.5 系统服务中的Binder对象
 - 5.5.1 ServiceManager管理的服务
 - 5.5.2 理解Manager

第6章 Framework概述

- 6.1 Framework框架
 - 6.1.1 服务端
 - 6.1.2 客户端
 - 6.1.3 Linux驱动
- 6.2 APK程序的运行过程
- 6.3 客户端中的线程
- 6.4 几个常见问题
 - 6.4.1 Acitivity之间如何传递消息（数据）
 - 6.4.2 窗口相关的概念

第7章 理解Context

<<Android内核剖析>>

- 7.1 Context是什么
- 7.2 一个应用程序中包含多少个Context对象
- 7.3 Context相关类的继承关系
- 7.4 创建Context
 - 7.4.1 Application对应的Context
 - 7.4.2 Activity对应的Context
 - 7.4.3 Service对应的Context
 - 7.4.4 Context之间的关系
- 第8章 创建窗口的过程
 - 8.1 窗口的类型
 - 8.2 token变量的含义
 - 8.2.1 Activity中的mToken
 - 8.2.2 Window中的mAppToken
 - 8.2.3 WindowManager.LayoutParams中的token
 - 8.2.4 View中的token
 - 8.3 创建应用窗口
 - 8.4 创建子窗口
 - 8.4.1 Dialog的创建
 - 8.4.2 PopupWindow的创建
 - 8.4.3 ContextMenu的创建
 - 8.4.4 OptionMenu的创建
 - 8.5 系统窗口Toast的创建
 - 8.5.1 Toast调用流程
 - 8.5.2 Toast添加窗口
 - 8.6 创建窗口示例
- 第9章 Framework的启动过程
 - 9.1 Framework运行环境综述
 - 9.2 Dalvik虚拟机相关的可执行程序
 - 9.2.1 dalvikvm
 - 9.2.2 dvz
 - 9.2.3 app_process
 - 9.3 zygote的启动
 - 9.3.1 在init.rc中配置zygote启动参数
 - 9.3.2 启动Socket服务端口
 - 9.3.3 加载preload-classes
 - 9.3.4 加载preload-resources
 - 9.3.5 使用folk启动新的进程
 - 9.4 SystemServer进程的启动
 - 9.4.1 启动各种系统服务线程
 - 9.4.2 启动第一个Activity
- 第10章 AmS内部原理
 - 10.1 Activity调度机制
 - 10.1.1 几个重要概念
 - 10.1.2 AmS中的一些重要调度相关变量
 - 10.1.3 startActivity()的流程
 - 10.1.4 stopActivityLocked()停止Activity
 - 10.1.5 按“ Home ”键回到桌面的过程

<<Android内核剖析>>

- 10.1.6 按“ Back ”键回到上一个Activity
- 10.1.7 长按“ Home ”键
- 10.1.8 Activity生命期的代码含义
- 10.2 内存管理
 - 10.2.1 关闭而不退出
 - 10.2.2 Android与Linux的配合
 - 10.2.3 各种关闭程序的过程
 - 10.2.4 释放内存详解
- 10.3 对AmS中数据对象的理解
 - 10.3.1 常见的对象操作
 - 10.3.2 理解Activity
 - 10.3.3 Android多进程吗，是同时在运行多个应用程序吗
- 10.4 ActivityGroup的内部机制
 - 10.4.1 TabActivity使用时的类关系结构
 - 10.4.2 LocalActivityManager的内部机制
 - 10.4.3 ActivityGroup内部的Activity生命期控制
- 第11章 从输入设备中获取消息
 - 11.1 Android消息获取过程概述
 - 11.2 与消息处理相关的源码文件分布
 - 11.3 创建InputDispatcher线程
 - 11.4 把窗口信息传递给InputDispatcher线程
 - 11.5 创建InputChannel
 - 11.6 在WmS中注册InputChannel
 - 11.7 在客户进程中注册InputChannel
 - 11.8 WmS中处理消息的时机
 - 11.9 客户窗口获取消息的时机
- 第12章 屏幕绘图基础
 - 12.1 绘制屏幕的软件架构
 - 12.2 Java客户端绘制调用过程
 - 12.3 C客户端绘制过程
 - 12.4 Java客户端绘制相关类的关系
- 第13章 View工作原理
 - 13.1 导论
 - 13.2 用户消息类型
 - 13.2.1 按键消息
 - 13.2.2 触摸消息
 - 13.3 按键消息派发过程
 - 13.3.1 KeyEvent.DispatcherState中的长按监测
 - 13.3.2 按键消息总体派发过程
 - 13.3.3 根视图内部派发过程
 - 13.3.4 Activity内部派发过程
 - 13.3.5 View类内部的onKeyDown()和onKeyUp()
 - 13.3.6 Activity中的onKeyDown()和onKeyUp()
 - 13.3.7 PhoneWindow内部消息派发过程
 - 13.4 按键消息在WmS中的派发过程
 - 13.5 触摸消息派发过程
 - 13.5.1 触摸消息总体派发过程

<<Android内核剖析>>

- 13.5.2 根视图内部消息派发过程
 - 13.5.3 ViewGroup内部消息派发过程
 - 13.5.4 各种消息监测的基本实现方法
 - 13.5.5 View内默认消息派发过程
 - 13.6 导致View树重新遍历的时机
 - 13.6.1 状态的分类
 - 13.6.2 导致View树重新遍历的总体诱因图
 - 13.6.3 refreshDrawableList()
 - 13.6.4 onFocusChanged()
 - 13.6.5 ensureTouchMode()
 - 13.6.6 setVisibility()
 - 13.6.7 setEnable()
 - 13.6.8 setSelected()
 - 13.6.9 invalidate()
 - 13.6.10 requestFocus()
 - 13.6.11 requestLayout()
 - 13.7 遍历View树performTraversals()的执行过程
 - 13.8 计算视图大小 (measure) 的过程
 - 13.8.1 measure内部设计思路
 - 13.8.2 ViewGroup中的measureChildWithMargins()
 - 13.8.3 LinearLayout中的onMeasure()过程举例
 - 13.9 布局 (layout) 过程
 - 13.9.1 layout过程的设计思路
 - 13.9.2 LinearLayout中onLayout()内部过程
 - 13.9.3 TextView中gravity与layout的关系
 - 13.10 绘制 (draw) 过程
 - 13.10.1 视图中可绘制的元素
 - 13.10.2 绘制过程的设计思路
 - 13.10.3 ViewRoot中draw()的内部流程
 - 13.10.4 View类中draw()函数内部流程
 - 13.10.5 ViewGroup类中绘制子视图dispatchDraw()内部流程
 - 13.10.6 ViewGroup类中drawChild()过程
 - 13.10.7 绘制滚动条
 - 13.11 动画的绘制
 - 13.11.1 动画的设计思路
 - 13.11.2 ViewGroup类中drawChild()函数中视图动画绘制过程
 - 13.11.3 ViewGroup中dispatchDraw()中布局动画绘制流程
- 第14章 WmS工作原理
- 14.1 概述
 - 14.1.1 窗口的定义
 - 14.1.2 窗口管理要解决的核心问题
 - 14.1.3 解决核心问题所使用的相关的变量列表
 - 14.1.4 几个操作的概念
 - 14.1.5 什么是Policy, 以及其与WmS的关系
 - 14.1.6 WmS接口结构
 - 14.2 WmS主要内部类
 - 14.2.1 表示窗口的数据类

<<Android内核剖析>>

- 14.2.2 DimAnimator
- 14.2.3 FadeInOutAnimation
- 14.2.4 InputMonitor类
- 14.2.5 PolicyThread
- 14.2.6 Session
- 14.2.7 Watermark
- 14.2.8 WMThread
- 14.3 窗口的创建和删除
 - 14.3.1 创建窗口的时机和过程
 - 14.3.2 assignLayersLocked()的执行过程
 - 14.3.3 addWindowToListInOrderLocked()的执行过程
 - 14.3.4 删除窗口的时机
 - 14.3.5 删除窗口的过程
 - 14.3.6 removeWindowInnerLocked()
- 14.4 计算窗口的大小
 - 14.4.1 描述窗口尺寸的变量
 - 14.4.2 窗口大小的变化过程
 - 14.4.3 Policy中layoutWindowLw()的执行过程
 - 14.4.4 输入法窗口如何影响应用窗口的大小
- 14.5 切换窗口
 - 14.5.1 切换要解决的问题
 - 14.5.2 InputManager和WmS的接口
 - 14.5.3 AmS与WmS的接口
 - 14.5.4 从A到B的切换
 - 14.5.5 从B回到A的过程
 - 14.5.6 A中长按“ Home ”键切换到B
 - 14.5.7 setAppVisiblity()与销毁Surface
 - 14.5.8 computeFocusedWindowLocked()
- 14.6 performLayoutAndPlaceSurfacesLockedInner()的执行过程
 - 14.6.1 总体过程
 - 14.6.2 第一大步骤：计算窗口的大小
 - 14.6.3 第二大步骤：计算窗口的可视状态
 - 14.6.4 第三大步骤：通知SurfaceFlinger进行窗口重绘
- 14.7 窗口动画
- 14.8 屏幕旋转及Configuration的变化过程

第3部分 系统篇

第15章 资源访问机制

- 15.1 定义资源
- 15.2 存储资源
- 15.3 styleable、style、attr、theme的意义
- 15.4 AttributeSet与TypedArray类
- 15.5 获取Resources的过程
 - 15.5.1 通过Context获取
 - 15.5.2 通过PackageManager获取
- 15.6 Framework资源
 - 15.6.1 加载和读取
 - 15.6.2 添加

<<Android内核剖析>>

15.6.3 实现真正主题切换的两种思路

第16章 程序包管理 (Package Manager Service)

16.1 包管理概述

16.2 packages.xml文件格式

16.2.1 last-platform-version标签

16.2.2 permissions标签

16.2.3 cert标签

16.2.4 sigs标签

16.2.5 perms标签

16.2.6 package标签

16.2.7 shared-user标签

16.3 包管理服务的启动过程

16.3.1 各主要功能类的关系

16.3.2 PmS主体启动过程

16.3.3 readPermission()内部过程

16.3.4 mSettings.readLP()

16.3.5 scanPackageLI()内部过程

16.3.6 mSettings.writeLP()

16.4 应用程序的安装和卸载

16.4.1 各主要功能类关系

16.4.2 应用程序安装过程

16.4.3 应用程序的卸载过程

16.5 intent匹配框架

16.5.1 主要功能类的关系

16.5.2 主体调用过程

第17章 输入法框架

17.1 输入法框架组成概述

17.2 输入法中各Binder对象的创建过程

17.2.1 InputConnection

17.2.2 IInputMethodClient

17.2.3 InputMethodSession

17.2.4 InputMethod

17.3 输入法主要操作过程

17.3.1 输入法相关模块的启动过程

17.3.2 切换输入法

17.3.3 启动输入法

17.3.4 显示输入法

17.3.5 输入法操作过程中的重要变量总结

17.4 输入法窗口内部的显示过程

17.4.1 IMS中的showWindow()的内部执行过程

17.4.2 标准布局的IMS

17.4.3 自定义布局的IMS

17.5 向编辑框传递字符

17.6 输入法相关源码清单

第4部分 编译篇

第18章 Android编译系统

18.1 Android源码文件结构

<<Android内核剖析>>

- 18.2 从调用make命令开始说起
 - 18.2.1 编译命令
 - 18.2.2 编译结构猜想
 - 18.3 编译所需脚本文件之间的协同关系
 - 18.3.1 编译系统内部功能模块图
 - 18.3.2 脚本文件的包含关系
 - 18.3.3 从子项目中提取编译目标
 - 18.3.4 生成编译规则
 - 18.3.5 设置编译输出目录
 - 18.3.6 生成最终的Image文件
 - 18.4 如何增加一个product
 - 18.4.1 什么是一个product
 - 18.4.2 如何增加一个product
 - 18.5 如何增加一个项目
 - 18.5.1 项目类别和项目路径
 - 18.5.2 添加一个C项目
 - 18.5.3 添加一个APK项目
 - 18.6 APK编译过程
 - 18.6.1 总体编译过程概述
 - 18.6.2 生成R.java
 - 18.6.3 编译aidl文件
 - 18.6.4 包含Java静态库
 - 18.6.5 编译Java源文件生成Jar包
 - 18.6.6 将Jar包转换为dex文件
 - 18.6.7 编译资源文件生成APK包
 - 18.6.8 将dex文件添加到APK包中
 - 18.6.9 添加JNI所需的动态库文件
 - 18.6.10 对APK文件进行签名
 - 18.6.11 使用zipalign优化APK内部存储
 - 18.7 Framework的编译
 - 18.7.1 总体编译过程
 - 18.7.2 framework/core/ext三个Jar文件的区别
 - 18.8 编译android.jar
 - 18.8.1 资源文件
 - 18.8.2 aidl文件
 - 18.8.3 Java文件
 - 18.9 编译adt插件
 - 18.10 总结
- 第19章 编译自己的Rom
- 19.1 嵌入式系统的内存地址空间
 - 19.2 各种映像 (Image) 文件的作用
 - 19.3 编译Nexus S (NS) 的Image文件
 - 19.3.1 编译Linux Kernel
 - 19.3.2 提取NS的私有驱动文件
 - 19.3.3 编译system.img文件
 - 19.3.4 创建ramdisk.img
 - 19.3.5 创建boot.img文件

<<Android内核剖析>>

- 19.4 使用fastboot写入Image文件
- 19.5 最后验证
 - 19.5.1 解决触摸按键问题
 - 19.5.2 解决音量和电源键
 - 19.5.3 WIFI问题
 - 19.5.4 安装Google Mobile Service (GMS)

第5部分 硬件驱动篇

第20章 基于TI OMAP处理器的 Techshine 开发板介绍

- 20.1 Techv-35XX开发板概述
- 20.2 交叉编译环境配置
- 20.3 x-loader编译
- 20.4 u-boot编译
- 20.5 Techv-35XX Linux驱动和内核配置及编译
 - 20.5.1 Touchscreen驱动配置
 - 20.5.2 KeyBoard驱动配置
 - 20.5.3 Audio驱动配置
 - 20.5.4 MMC/SD驱动配置
 - 20.5.5 NandFlash驱动配置
 - 20.5.6 LCD驱动配置
 - 20.5.7 内核编译
- 20.6 Techv-35XX Android驱动编写
- 20.7 Techv-35XX Android开发环境建立
- 20.8 编译Android Donut
- 20.9 Android根文件系统的制作
- 20.10 相关Image文件的烧写
- 20.11 Android 根文件系统安装

<<Android内核剖析>>

编辑推荐

“内核剖析”乍一听起来挺吓唬人的，但这个词语存在两个问题，第一个是什么才能称为内核？另一个是“谁”才有能力或者有机会写一本“内核剖析”的书？

由柯元旦编著的《Android内核剖析》之所以在前言中提出这个问题，就是为了不吓唬大家，并给大家一种信心，相信自己有能力理解本书的内容。

<<Android内核剖析>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>