

图书基本信息

书名：<<Android多媒体应用开发实战详解>>

13位ISBN编号：9787115284105

10位ISBN编号：7115284105

出版时间：2012-8

出版时间：人民邮电出版社

作者：王石磊，吴峥 编著

页数：560

字数：872000

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>

内容概要

Android凭借其强大的功能逐渐赢得了广大用户和开发者的青睐，已经成为移动开发平台上的翘楚。

全书分为17章，分别详细讲解了Android系统基础，深入底层基础，多媒体框架，音频系统框架，视频系统框架，Camera照相机系统，振动器系统和警报系统，2D应用开发，渲染二维图像，OpenGL ES基础，OpenGL

ES实战技术，音频开发基础，视频开发应用等。

全面涵盖多媒体开发与应用技术，在《Android多媒体应用开发实战详解：图像、音频、视频、2D和3D》最后，通过两个综合实例，分别介绍了开发屏保系统和音乐播放器的基本流程，帮助读者学以致用。

《Android多媒体应用开发实战详解：图像、音频、视频、2D和3D》适合Android程序员、研发人员及Android爱好者学习，也可以作为相关培训学校和大专院校相关专业的教学用书。

书籍目录

第1章 初识庐山真面目——Android概述

1.1 智能手机世界

1.1.1 何谓智能手机

1.1.2 当前主流的智能手机系统

1.2 Android的巨大优势

1.2.1 系出名门

1.2.2 强大的开发团队

1.2.3 诱人的奖励机制

1.3 搭建Android应用开发环境

1.3.1 安装Android SDK的系统要求

1.3.2 安装JDK、Eclipse、Android SDK

1.3.3 设定Android SDK Home

1.3.4 验证开发环境

1.3.5 创建Android虚拟设备 (AVD)

1.3.6 常见的几个问题

1.4 Android模拟器

1.4.1 Android模拟器简介

1.4.2 模拟器和真机的区别

1.4.3 模拟器简单总结

第2章 千里之行始于足下——多媒体开发前的准备工作

2.1 简析Android安装文件

2.1.1 Android SDK目录结构

2.1.2 android.jar及内部结构

2.1.3 SDK文档及阅读技巧

2.1.4 Android SDK工具集

2.2 解析Android SDK实例

2.3 Android系统架构

2.3.1 Android体系结构介绍

2.3.2 Android应用工程文件组成

2.3.3 应用程序的生命周期

第3章 底层分析基础

3.1 搭建Linux开发环境

3.1.1 安装

3.1.2 设置环境变量

3.1.3 安装编译工具

3.2 获取Android源代码

3.3 分析并编译Android源代码

3.3.1 Android源代码的结构

3.3.2 编译Android源代码

3.3.3 运行Android源代码

3.3.4 两种编译Android程序的方法

3.4 运行模拟器

3.4.1 Linux环境下运行模拟器的方法

3.4.2 模拟器辅助工具——adb

3.5 Android的启动过程

- 3.5.1 Init初始化进程
- 3.5.2 ServiceManager进程
- 3.5.3 Zygote进程
- 3.5.4 SystemService进程
- 3.6 进程间的通信
- 3.7 多核通信
 - 3.7.1 内存共享
 - 3.7.2 过程调用
- 第4章 多媒体框架
 - 4.1 Android多媒体系统介绍
 - 4.2 OpenMax框架
 - 4.2.1 分析OpenMax框架构成
 - 4.2.2 实现OpenMax IL层接口
 - 4.3 OpenCore框架
 - 4.3.1 OpenCore层次结构
 - 4.3.2 OpenCore代码结构
 - 4.3.3 OpenCore编译结构
 - 4.3.4 OpenCore OSCL
 - 4.3.5 实现OpenCore中的OpenMax部分
 - 4.3.6 OpenCore扩展
 - 4.4 Stagefright框架
 - 4.4.1 Stagefright代码结构
 - 4.4.2 通过Stagefright实现OpenMax接口
 - 4.4.3 Video Buffer传输流程
- 第5章 音频系统框架
 - 5.1 音频系统结构
 - 5.2 分析音频系统的层次
 - 5.2.1 层次说明
 - 5.2.2 Media库中的Audio框架
 - 5.2.3 本地代码
 - 5.2.4 JNI代码
 - 5.2.5 Java代码
 - 5.3 分析硬件抽象层
 - 5.4 分析编码/解码过程
 - 5.4.1 AMR编码
 - 5.4.2 AMR解码
 - 5.4.3 解码MP3
- 第6章 视频系统框架
 - 6.1 视频系统结构
 - 6.2 分析Overlay抽象层
 - 6.2.1 Overlay系统硬件抽象层的接口
 - 6.2.2 实现Overlay系统抽象层
 - 6.2.3 实现接口
 - 6.3 实现Overlay框架
- 第7章 Camera照相机系统
 - 7.1 Camera系统的结构
 - 7.2 分析Camera接口和驱动

- 7.2.1 Camera驱动接口
- 7.2.2 硬件抽象层
- 7.3 实现Camera的硬件抽象层
 - 7.3.1 Java程序部分
 - 7.3.2 Camera的Java本地调用部分
 - 7.3.3 Camera的本地库libui.so
 - 7.3.4 Camera服务libcameraservice.so
- 第8章 振动器系统和警报系统
 - 8.1 振动器系统
 - 8.1.1 硬件抽象层
 - 8.1.2 JNI框架部分
 - 8.1.3 实现硬件抽象层
 - 8.2 Alarm警报器系统
 - 8.2.1 Alarm系统的结构
 - 8.2.2 RTC驱动程序
 - 8.2.3 模拟器环境的具体实现
- 第9章 绘制二维图像
 - 9.1 使用Color类设置文本颜色
 - 9.2 使用Paint类绘制图像
 - 9.3 使用Canvas画布
 - 9.4 使用Rect矩形类
 - 9.5 NinePatch类
 - 9.6 使用Matrix类
 - 9.7 Bitmap类
 - 9.8 使用BitmapFactory类
 - 9.9 使用Region类
 - 9.10 使用Typeface类
 - 9.11 使用Shader类
- 第10章 二维动画应用
 - 10.1 使用Drawable实现动画效果
 - 10.1.1 Drawable基础
 - 10.1.2 使用Drawable实现动画效果
 - 10.2 实现Tween Animation动画效果
 - 10.2.1 Tween动画基础
 - 10.2.2 定义动画效果
 - 10.2.3 Tween应用实例1
 - 10.2.4 Tween应用实例2
 - 10.2.5 Tween Animation总结
 - 10.3 实现Frame Animation动画效果
 - 10.3.1 Frame动画基础
 - 10.3.2 Frame动画应用实例
 - 10.4 播放GIF动画
 - 10.5 实现EditText动画特效
 - 10.6 全新的Property Animation动画
- 第11章 渲染二维图像
 - 11.1 Android GDI系统之SurfaceFlinger
 - 11.1.1 SurfaceFinger基础

- 11.1.2 Surface和Canvas
- 11.1.3 Surface渲染
- 11.2 Skia引擎
 - 11.2.1 Skia基础
 - 11.2.2 Skia介绍
 - 11.2.3 Skia中的类
 - 11.2.4 使用Skia绘图
 - 11.2.5 Skia的其他功能
- 11.3 通过Skia绘制文字分析原理
- 第12章 OpenGL ES基础
 - 12.1 OpenGL ES介绍
 - 12.2 OpenGL ES的基本应用
 - 12.2.1 点线法绘制三角形
 - 12.2.2 索引法绘制三角形
 - 12.2.3 顶点法绘制三角形
 - 12.3 实现投影效果
 - 12.3.1 正交投影
 - 12.3.2 透视投影
 - 12.3.3 实现投影效果
 - 12.4 实现光照效果
 - 12.4.1 光照基础
 - 12.4.2 实例应用——开启/关闭光照
 - 12.4.3 实例应用——实现定位光效果
 - 12.5 实现纹理映射效果
 - 12.5.1 纹理映射基础
 - 12.5.2 实例应用——实现三角形纹理贴图效果
 - 12.5.3 实例应用——实现地月模型效果
 - 12.5.4 实例应用——实现纹理拉伸效果
- 第13章 OpenGL ES进阶
 - 13.1 绘制基本的三维形状
 - 13.1.1 绘制一个圆柱体
 - 13.1.2 绘制一个圆环
 - 13.1.3 绘制一个抛物面效果
 - 13.1.4 绘制一个螺旋面效果
 - 13.2 实现坐标变换操作
 - 13.2.1 实现缩放变换效果
 - 13.2.2 实现平移变换效果
 - 13.3 实现混合效果
 - 13.3.1 基本知识
 - 13.3.2 实现混合效果
 - 13.4 实现摄像机和雾特效效果
 - 13.4.1 摄像机
 - 13.4.2 雾特效
 - 13.4.3 实现雾特效和摄像机效果
 - 13.5 粒子系统
 - 13.5.1 粒子系统基础
 - 13.5.2 实现粒子系统效果

第14章 音频开发应用

14.1 音频应用接口类介绍

14.2 AudioManager类

14.2.1 AudioManager基础

14.2.2 AudioManager基本应用——设置短信提示铃声

14.2.3 AudioManager基本应用——调节手机音量的大小

14.3 录音处理

14.3.1 使用MediaRecorder接口录制音频

14.3.2 使用AudioRecord接口录制音频

14.4 播放音频

14.4.1 使用AudioTrack播放音频

14.4.2 使用MediaPlayer播放音频

14.4.3 使用SoundPool播放音频

14.4.4 使用Ringtone播放铃声

14.4.5 使用JetPlayer播放音频

14.4.6 使用AudioEffect处理音效

14.5 语音识别技术

14.5.1 Text-To-Speech技术

14.5.2 谷歌的Voice Recognition技术

14.6 实现振动效果

14.6.1 Vibrator类基础

14.6.2 使用Vibrator实现振动效果

14.7 设置闹钟

14.7.1 AlarmManage基础

14.7.2 开发一个闹钟程序

第15章 视频开发应用

15.1 使用MediaPlayer播放视频

15.2 使用VideoView播放视频

15.2.1 VideoView基础

15.2.2 使用VideoView播放手机中的影片

15.2.3 使用VideoView播放手机中的MP4

15.3 使用Camera拍照

15.3.1 Camera基础

15.3.2 总结Camera拍照的流程

15.3.3 使用Camera预览并拍照

第16章 开发一个屏保程序

16.1 屏幕保护程序介绍

16.1.1 屏幕保护程序的作用

16.1.2 手机中的屏幕保护程序

16.2 开发屏保程序的原理

16.3 开发一个屏保程序

16.3.1 准备素材图片

16.3.2 编写布局文件

16.3.3 编写主程序文件

第17章 开发一个音乐播放器

17.1 项目介绍

17.1.1 项目背景介绍

- 17.1.2 项目的目的
- 17.2 系统需求分析
 - 17.2.1 构成模块
 - 17.2.2 系统流程
 - 17.2.3 功能结构图
 - 17.2.4 系统功能说明
 - 17.2.5 系统需求
- 17.3 数据库设计
 - 17.3.1 字段设计
 - 17.3.2 E-R图设计
 - 17.3.3 数据库连接
 - 17.3.4 创建数据库
 - 17.3.5 操作数据库
 - 17.3.6 数据显示
- 17.4 具体编码
 - 17.4.1 设置服务信息
 - 17.4.2 播放器主界面
 - 17.4.3 播放列表功能
 - 17.4.4 菜单功能模块
 - 17.4.5 播放设置界面
 - 17.4.6 设置显示歌词
 - 17.4.7 文件浏览器模块
 - 17.4.8 数据存储
- 17.5 总结

章节摘录

版权页：插图：程序也如同自然界的生物一样，有自己的生命周期。

应用程序的生命周期即程序的存活时间，即在啥时间内有效。

Android是构建在Linux之上的开源移动开发平台，在Android中，多数情况下每个程序都是在各自独立的Linux进程中运行的。

当一个程序或其某些部分被请求时，它的进程就“出生”了。

当这个程序没有必要再运行下去且系统需要回收这个进程的内存用于其他程序时，这个进程就“死亡”了。

可以看出，Android程序的生命周期是由系统控制而非程序自身直接控制的。

这和我们编写桌面应用程序时的思维有一些不同，一个桌面应用程序的进程也是在其他进程或用户请求时被创建，但是往往是在程序自身收到关闭请求后执行一个特定的动作（如从main函数中返回）而导致进程结束的。

要想做好某种类型的程序或者某种平台下程序的开发，最关键的就是要弄清楚这种类型的程序或整个平台下的程序的一般工作模式，并熟记在心。

在Android中，程序的生命周期控制就是属于这个范畴。

开发者必须理解不同的应用程序组件，尤其是Activity、Service和Intent Receiver。

了解这些组件是如何影响应用程序的生命周期的，这非常重要。

如果不正确地使用这些组件，可能会导致系统终止正在执行重要任务的应用程序进程。

一个常见的进程生命周期漏洞的例子是Intent Receiver（意图接收器），当Intent Receiver在onReceive方法中接收到一个Intent（意图）时，它会启动一个线程，然后返回。

一旦返回，系统将认为Intent Receiver不再处于活动状态，因而Intent Receiver所在的进程也就不再有用了（除非该进程中还有其他组件处于活动状态）。

因此，系统可能会在任意时刻终止该进程以回收占有的内存。

这样进程中创建出的那个线程也将被终止。

解决这个问题的方法是从Intent Receiver中启动一个服务，让系统知道进程中还有处于活动状态的工作

。为了使系统能够正确决定在内存不足时应该终止哪个进程，Android根据每个进程中运行的组件及组件的状态把进程放入一个“Importance Hierarchy（重要性分级）”中。

进程的类型按重要程度排序。

它有一个可以被用户从屏幕上看到的活动，但不在前台（它的onPause方法被调用）。

例如，如果前台的活动是一个对话框，以前的活动就隐藏在对话框之后，就会出现这种进程。

可见进程非常重要，一般不允许被终止，除非是为了保证前台进程的运行而不得不终止它。

编辑推荐

《Android多媒体应用开发实战详解:图像、音频、视频、2D和3D》适合Android程序员、研发人员及Android爱好者学习,从入门到深入,从底层到应用,知识点环环相扣,实用性强,深入分析总结了音频、视频、三维和图像应用的知识,也可以作为相关培训学校和大专院校相关专业的教学用书。

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>