

《移动开发经典丛书》

图书基本信息

书名：《移动开发经典丛书》

13位ISBN编号：9787302343012

出版时间：2014-1-1

作者：辛纳 (Onur Cinar)

页数：344

译者：于红,余建伟,冯艳红

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

作者简介

Onur Cinar有超过17年的移动和通信领域大规模复杂软件项目的设计、开发和管理经验。他的专业技能包括VoIP、视频通信、移动应用程序、网格计算和不同平台上的网络技术。从Android平台问世他就一直积极从事这方面的工作。他是Apress出版的Android Apps with Eclipse一书的作者。他在美国宾州费城Drexel大学获得计算机科学理学学士学位。现就职于微软Skype分部，任Android平台的Skype客户端高级产品工程经理。

书籍目录

第1章 Android平台上的C++入门

1

1.1 Microsoft Windows

1

1.1.1 在Windows平台上下载并安装JDK开发包

2

1.1.2 在Windows平台上下载并安装Apache ANT

5

1.1.3 在Windows平台上下载并安装Android SDK

7

1.1.4 在Windows平台上下载并安装Cygwin

8

1.1.5 在Windows平台上下载并安装Android NDK

11

1.1.6 在Windows平台上下载并安装Eclipse

13

1.2 Apple Mac OS X

14

1.2.1 在Mac平台上安装Xcode

14

1.2.2 验证Mac平台的Java开发包

15

1.2.3 验证Mac平台上的Apache ANT

15

1.2.4 验证GNU Make

16

1.2.5 在Mac平台上下载并安装Android SDK

16

1.2.6 在Mac平台上下载并安装Android NDK

18

1.2.7 在Mac平台上下载并安装Eclipse

19

1.3 Ubuntu Linux

20

1.3.1 检查GNU C库版本

20

1.3.2 激活在64位系统上支持32位的功能

21

1.3.3 在Linux平台上下载并安装Java开发工具包(JDK)

21

1.3.4 在Linux平台上下载并安装Apache ANT

22

1.3.5 在Linux平台上下载并安装GNU Make

22

1.3.6 在Linux平台上下载并安装Android SDK

23

1.3.7 在Linux平台上下载并安装Android NDK

24	
1.3.8	在Linux平台上下载并安装Eclipse
25	
1.4	下载并安装ADT
26	
1.4.1	安装Android平台包
29	
1.4.2	配置模拟器
30	
1.5	小结
33	
第2章	深入了解Android NDK
35	
2.1	Android NDK提供的组件
35	
2.2	Android NDK的结构
36	
2.3	以一个示例开始
36	
2.3.1	指定Android NDK的位置
37	
2.3.2	导入示例项目
37	
2.3.3	向项目中添加原生支持
39	
2.3.4	运行项目
40	
2.3.5	用命令行对项目进行构建
41	
2.3.6	检测Android NDK项目的结构
42	
2.4	构建系统
42	
2.4.1	Android.mk
43	
2.4.2	Application.mk
53	
2.5	使用NDK-Build脚本
54	
2.6	排除构建系统故障
55	
2.7	小结
56	
第3章	用JNI实现与原生代码通信
57	
3.1	什么是JNI
57	
3.2	以一个示例开始
57	

3.2.1 原生方法的声明	58
3.2.2 加载共享库	58
3.2.3 实现原生方法	59
3.3 数据类型	64
3.3.1 基本数据类型	64
3.3.2 引用类型	64
3.4 对引用数据类型的操作	65
3.4.1 字符串操作	65
3.4.2 数组操作	67
3.4.3 NIO 操作	68
3.4.4 访问域	69
3.4.5 调用方法	71
3.4.6 域和方法描述符	72
3.5 异常处理	75
3.5.1 捕获异常	75
3.5.2 抛出异常	75
3.6 局部和全局引用	76
3.6.1 局部引用	76
3.6.2 全局引用	76
3.6.3 弱全局引用	77
3.7 线程	78
3.7.1 同步	78
3.7.2 原生线程	79
3.8 小结	79
第4章 使用SWIG自动生成JNI代码	

81	
4.1 什么是SWIG	81
4.2 安装	82
4.2.1 Windows平台上SWIG的安装	82
4.2.2 在Mac OS X下安装	83
4.2.3 在Ubuntu Linux下安装	85
4.3 通过示例程序试用SWIG	86
4.3.1 接口文件	86
4.3.2 在命令行方式下调用SWIG	89
4.3.3 将SWIG集成到Android构建过程中	90
4.3.4 更新Activity	92
4.3.5 执行应用程序	93
4.3.6 剖析生成的代码	93
4.4 封装C语言代码	94
4.4.1 全局变量	94
4.4.2 常量	95
4.4.3 只读变量	96
4.4.4 枚举	97
4.4.5 结构体	100
4.4.6 指针	101
4.5 封装C++代码	101
4.5.1 指针、引用和值	102
4.5.2 默认参数	103
4.5.3 重载函数	104
4.5.4 类	104

4.6 异常处理	106
4.7 内存管理	107
4.8 从原生代码中调用Java	108
4.8.1 异步通信	108
4.8.2 启用Directors	109
4.8.3 启用RTTI	109
4.8.4 重写回调方法	109
4.8.5 更新HelloJni Activity	110
4.9 小结	110
第5章 日志、调试及故障处理	111
5.1 日志	111
5.1.1 框架	111
5.1.2 原生日志API	112
5.1.3 受控制的日志	114
5.1.4 控制台日志	118
5.2 调试	119
5.2.1 预备知识	119
5.2.2 调试会话建立	120
5.2.3 建立调试示例	121
5.2.4 启动调试器	121
5.3 故障处理	126
5.3.1 堆栈跟踪分析	127
5.3.2 对JNI的扩展检查	128
5.3.3 内存问题	130
5.3.4 strace	

133	
5.4 小结	134
第6章 Bionic API入门	135
6.1 回顾标准库	135
6.2 还有另一个C库	136
6.2.1 二进制兼容性	136
6.2.2 提供了什么	136
6.2.3 缺什么	137
6.3 内存管理	137
6.3.1 内存分配	137
6.3.2 C语言的动态内存管理	138
6.3.3 C++的动态内存管理	139
6.4 标准文件I/O	141
6.4.1 标准流	141
6.4.2 使用流I/O	141
6.4.3 打开流	142
6.4.4 写入流	143
6.4.5 流的读取	145
6.4.6 搜索位置	148
6.4.7 错误检查	149
6.4.8 关闭流	149
6.5 与进程交互	150
6.5.1 执行shell命令	150
6.5.2 与子进程通信	150
6.6 系统配置	151

6.6.1 通过名称获取系统属性值	152
6.6.2 通过名称获取系统属性	152
6.7 用户和组	153
6.7.1 获取应用程序用户和组ID	153
6.7.2 获取应用程序用户名	154
6.8 进程间通信	154
6.9 小结	154
第7章 原生线程	155
7.1 创建线程示例项目	155
7.1.1 创建Android项目	155
7.1.2 添加原生支持	157
7.1.3 声明字符串资源	157
7.1.4 创建简单的用户界面	157
7.1.5 实现Main Activity	159
7.1.6 生成C/C++头文件	162
7.1.7 实现原生函数	163
7.1.8 更新Android.mk构建脚本	165
7.2 Java 线程	165
7.2.1 修改示例应用程序使之能够使用Java线程	165
7.2.2 执行Java Threads示例	166
7.2.3 原生代码使用Java线程的优缺点	167
7.3 POSIX线程	168
7.3.1 在原生代码中使用POSIX线程	168
7.3.2 用pthread_create创建线程	168
7.3.3 更新示例应用程序以使用POSIX线程	

169
7.3.4 执行POSIX线程示例
174
7.4 从POSIX线程返回结果
174
7.5 POSIX线程同步
176
7.5.1 用互斥锁同步POSIX线程
176
7.5.2 使用信号量同步POSIX线程
180
7.6 POSIX线程的优先级和调度策略
180
7.6.1 POSIX的线程调度策略
181
7.6.2 POSIX Thread优先级
181
7.7 小结
181
第8章 POSIX Socket API：面向连接的通信
183
8.1 Echo Socket示例应用
183
8.1.1 Echo Android应用项目
184
8.1.2 抽象echo activity
184
8.1.3 echo应用程序字符串资源
188
8.1.4 原生echo模块
188
8.2 用TCP sockets实现面向连接的通信
191
8.2.1 Echo Server Activity的布局
192
8.2.2 Echo Server Activity
193
8.2.3 实现原生TCP Server
194
8.2.4 Echo客户端Activity布局
206
8.2.5 Echo客户端Activity
208
8.2.6 实现原生TCP客户端
210
8.2.7 更新Android Manifest
213
8.2.8 运行TCP Sockets示例
214

8.3 小结	217
第9章 POSIX Socket API：无连接的通信	219
9.1 将UDP Server方法添加到Echo Server Activity中	219
9.2 实现原生UDP Server	220
9.2.1 创建UDP Socket：socket	220
9.2.2 从Socket接收数据报：recvfrom	221
9.2.3 向Socket发送数据报：sendto	223
9.2.4 原生UDP Server方法	224
9.3 将原生UDP Client方法加入Echo Client Activity中	225
9.4 实现原生UDP Client	226
9.5 运行UDP Sockets示例	228
9.5.1 连通UDP的模拟器	228
9.5.2 启动Echo UDP Client	229
9.6 小结	229
第10章 POSIX Socket API：本地通信	231
10.1 Echo Local Activity布局	231
10.2 Echo Local Activity	232
10.3 实现原生本地Socket Server	237
10.3.1 创建本地Socket：socket	237
10.3.2 将本地socket与Name绑定：bind	238
10.3.3 接受本地Socket：accept	240
10.3.4 原生本地Socket Server	240
10.4 将本地Echo Activity添加到Manifest中	242
10.5 运行本地 Sockets示例	

243	
10.6 异步I/O	
243	
10.7 小结	
244	
第11章 支持C++	
245	
11.1 支持的C++运行库	
245	
11.1.1 GAbi++ C++运行库	
246	
11.1.2 STLport C++运行库	
246	
11.1.3 GNU STL C++运行库	
246	
11.2 指定C++运行库	
246	
11.3 静态运行库与动态运行库	
247	
11.4 C++异常支持	
247	
11.5 C++ RTTI支持	
248	
11.6 C++标准库入门	
249	
11.6.1 容器	
249	
11.6.2 迭代器	
250	
11.6.3 算法	
251	
11.7 C++运行库的线程安全	
251	
11.8 C++运行库调试模式	
251	
11.8.1 GNU STL调试模式	
251	
11.8.2 STLport调试模式	
252	
11.9 小结	
253	
第12章 原生图形API	
255	
12.1 原生图形API的可用性	
255	
12.2 创建一个AVI视频播放器	
256	
12.2.1 将AVILib作为NDK的一个导入模块	
256	

12.2.2 创建AVI播放器Android应用程序	258
12.2.3 创建AVI Player的Main Activity	258
12.2.4 创建Abstract Player Activity	262
12.3 使用JNI图形API进行渲染	269
12.3.1 启用JNI Graphics API	269
12.3.2 使用JNI Graphics API	270
12.3.3 用Bitmap渲染来更新AVI Player	271
12.3.4 运行使用Bitmap渲染的AVI Player	278
12.4 使用OpenGL ES渲染	279
12.4.1 使用OpenGL ES API	279
12.4.2 启用OpenGL ES 1.x API	279
12.4.3 启用OpenGL ES 2.0 API	280
12.4.4 用OpenGL ES渲染来更新AVI Player	280
12.5 使用原生Window API进行渲染	290
12.5.1 启用原生Window API	290
12.5.2 使用原生Window API	291
12.5.3 用原生window渲染器来更新AVI Player	293
12.5.4 EGL图形库	301
12.6 小结	301
第13章 原生音频API	303
13.1 使用OpenSL ES API	303
13.1.1 与OpenSL ES标准的兼容性	304
13.1.2 音频许可	304
13.2 创建WAVE音频播放器	304
13.2.1 将WAVELib作为NDK导入模块	

304	
13.2.2	创建WAVE播放器Android应用程序
306	
13.2.3	创建WAVE播放器主Activity
306	
13.2.4	实现WAVE Aduio播放
310	
13.3	运行WAVE Audio Player
327	
13.4	小结
328	
	第14章 程序概要分析和NEON优化
329	
14.1	用GNU Profiler度量性能
329	
14.1.1	安装Android NDK Profiler
329	
14.1.2	启用Android NDK Profiler
330	
14.1.3	使用GNU Profiler分析gmon.out文件
331	
14.2	使用ARM NEON Intrinsics进行优化
332	
14.2.1	ARM NEON技术概述
333	
14.2.2	给AVI Player添加一个亮度过滤器
333	
14.2.3	为AVI播放器启用Android NDK Profiler
336	
14.2.4	AVI Player程序概要分析
337	
14.2.5	使用NEON Intrinsics优化Brightness Filter
338	
14.3	自动向量化
342	
14.3.1	启用自动向量化
343	
14.3.2	自动向量化问题的发现和排除
344	
14.4	小结
344	

精彩短评

- 1、第4章干货较多，其他章节基本上大杂烩，面面俱到，点到为止。但是OOP的混合编程也是一个重要的发展方向。
 - 2、虽然有Android.mk的部分没太详细讲，但是也不失为一本Android的NDK开发的入门好书，通过开发一些小功能，用几种不同的方法来实现，让你了解C++开发Android程序，代码也不复杂，书上也都是完整的实例。
 - 3、工作中NDK用的很多，借这本书系统的梳理一遍，也发现了新东西
1. native code依然可用gdb、valgrind，好棒！！
 2. 理解Java和native代码的相互调用，本质要理解native和JVM如何协作的。
 3. NDK系统库是针对移动平台简化版的C/C++库。提供四种C++运行库：system、GAbi++、STLport、GNU STL C++
 - 4、native层渲染图像、视频的方法：bitmap、openGLES、native window
 - 5、native层播放音频的方法：openSLES
 - 6、native多线程：基于java线程——不能通信、不同能同步，太傻比了；基于native posix线程——常用。
- 4、好棒很详细。从JNI开始讲到用libavi这个C解码库来结合surfaceview做播放器。。就差实践一下啦==
 - 5、知识分散，还有出版印刷质量太差...图片看不清...

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:www.tushu000.com