

探索和品味Android大师们的内核设计艺术



张元亮◎编著

深入理解 Android系统

- 全面剖析进程/线程、内存管理、Binder机制、显示系统、多媒体管理、输入系统等核心知识在Android中的实现原理。
- 源码分析+全真示例+图片解析=更易于理解的思维路径。
- 由浅入深，由总体框架到细节实现，快速获取对Android系统的二次开发能力。
- 教授精髓，精讲精练。赠送源码，拿来就用。

清华大学出版社



深入理解 Android 系统

张元亮 编著

清华大学出版社

北 京

内 容 简 介

本书内容共 18 章,循序渐进地分析了整个 Android 系统的基本架构知识,从获取源码开始讲起,依次讲解了 Android 系统介绍,包括获取并编译 Android 源码,分析 JNI,内存系统架构详解,硬件抽象层架构详解,Binder 通信机制详解,init 启动进程详解,Zygote 进程详解,System 进程详解,应用程序进程详解,ART 机制架构详解,Sensor 传感器系统架构详解,蓝牙系统架构详解,Android 多媒体框架架构详解,音频系统框架架构详解,视频系统架构详解,WebKit 系统架构详解,Android 5.0 中的 WebView, Wi-Fi 系统架构详解等内容。本书几乎涵盖了所有 Android 系统架构的主要核心内容,讲解方法通俗易懂并且详细,不但适合应用高手们学习,也特别便于初学者学习和理解。

本书适合 Android 源码分析人员、Android 系统架构师、Linux 开发人员、Android 物联网开发人员、Android 爱好者、Android 底层开发人员、Android 驱动开发人员、Android 应用开发人员、Android 传感器开发人员、Android 智能家居开发人员、Android 可穿戴设备开发人员学习,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。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。
版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

深入理解 Android 系统/张元亮编著. —北京:清华大学出版社,2015
ISBN 978-7-302-40439-2

I. ①深… II. ①张… 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 122705 号

责任编辑:朱英彪
封面设计:刘超
版式设计:魏远
责任校对:王云
责任印制:宋林

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者:清华大学印刷厂

经 销:全国新华书店

开 本:203mm×260mm 印 张:44.5 字 数:1248 千字

版 次:2015 年 7 月第 1 版 印 次:2015 年 7 月第 1 次印刷

印 数:1~3000

定 价:88.00 元

产品编号:061537-01

前 言

2007年11月5日，谷歌公司宣布基于Linux平台的开源手机操作系统Android诞生，该平台号称是首个为移动终端打造的真正开放和完整的移动软件平台。本书将和广大读者一起深入理解Android系统的架构知识，共同领略这款神奇系统的奥妙之处。

市场占有率高居第一

截至2014年9月，Android在智能手机市场上的占有率从2013年第一季度的68.8%上升到85%。而iOS则从2013年的19.4%下降到15.5%，WP系统从原来的2.7%小幅上升到3.6%。从数据上看，Android平台占据了市场的主导地位。

由数据可知，iOS有所下降，WP市场小幅增长，但Android市场的占有率增加幅度较大。就目前来看，智能手机的市场已经饱和，大多数用户都在各个平台间转换。而就在这样一个市场上，Android还增长了10%左右的占有率确实不易。

为开发人员提供了发展的平台

(1) 保证开发人员可以迅速向Android应用开发转型

Android应用程序是使用Java语言开发的，只要具备Java开发基础，就能很快入门并掌握。作为单独的Android应用开发，对Java编程门槛的要求并不高，即使没有编程经验，也可以在突击学习Java之后学习Android。另外，Android完全支持2D、3D和数据库，并且和浏览器实现了集成。所以通过Android平台，程序员可以迅速、高效地开发出绚丽多彩的应用，例如常见的工具、管理软件、互联网应用和游戏等。

(2) 定期举办奖金丰厚的Android开发大赛

为了吸引更多的用户使用Android开发，已经成功举办了奖金为数千万美元的开发竞赛。鼓励开发人员创建出创意十足且实用的软件。对于开发人员来说，这种大赛不但能提升自己的开发水平，并且高额的奖金也是学员们学习的动力。

(3) 开发人员可以利用自己的作品赚钱

为了能让Android平台引起更多的关注，谷歌提供了一个专门下载Android应用的门店Android Market，地址是<https://play.google.com/store>。在这个门店里面允许开发人员发布应用程序，也允许Android用户下载自己喜欢的程序。作为开发者，需要申请开发者账号，申请后才能将自己的程序上传到Android Market，并且可以对自己的软件进行定价。只要所开发的软件程序足够吸引人，就可以获得很可观的金钱回报。这样实现了程序员学习和赚钱的两不误，所以吸引了更多开发人员加入到Android开发大军中来。

本书的内容

本书内容共 18 章，循序渐进地分析了整个 Android 系统的基本架构知识。本书从获取源码开始讲起，依次讲解了 Android 系统介绍，包括获取并编译 Android 源码，分析 JNI，内存系统架构详解，硬件抽象层架构详解，Binder 通信机制详解，init 启动进程详解，Zygote 进程详解，System 进程详解，应用程序进程详解，Sensor 传感器系统架构详解，蓝牙系统架构详解，Android 多媒体框架架构详解，音频系统框架架构详解，视频系统架构详解，WebKit 系统架构详解，Android 5.0 中的 WebView，Wi-Fi 系统架构详解，ART 机制架构详解等内容。本书几乎涵盖了所有 Android 系统架构的主要核心内容，讲解方法通俗易懂并且详细，不但适合应用高手们学习，也特别便于初学者学习和理解。

本书的版本

Android 系统自 2008 年 9 月发布第一个版本 1.1 以来，截至 2014 年 10 月发布最新版本 5.0，一共存在十多个版本。由此可见，Android 系统升级频率较快，一年之中最少有两个新版本诞生。如果过于追求新版本，会造成力不从心的结果。所以在此建议广大读者不必追求最新的版本，只需关注最流行的版本即可。据官方统计，截至 2014 年 11 月 10 日，占据前 3 位的版本分别是 Android 4.4、Android 4.3 和 Android 5.0，其实这 3 个版本的区别并不是很大，只是在某些领域的细节上进行了更新。为了在市场普及率和新版本之间做好兼顾，本书将以最新的 Android 5.0 作为讲解主线，并且结合了 Android 4.4 的架构知识。

本书特色

本书内容十分丰富，讲解细致。我们的目标是通过一本图书，提供多本图书的价值，读者可以根据自己的需要有选择地阅读。在内容的编写上，本书具有以下特色。

（1）内容全面，讲解细致

本书几乎涵盖了 Android 系统架构所需要的所有主要知识点，详细讲解了每一个 Android 系统的具体实现过程。每一个知识点都力求用详实易懂的语言展现在读者面前。

（2）遵循合理的主线进行讲解

为了使广大读者彻底弄清楚 Android 系统架构的各个知识点，在讲解每一个知识点时，从 Linux 内核开始讲起，依次剖析了底层架构、API 硬件抽象层和顶层应用的具体知识。遵循了从底层到顶层的顺序，实现了 Android 系统架构大揭秘的目标。

（3）章节独立，自由阅读

本书中的每一章内容都可以独自成书，读者既可以按照本书编排的章节顺序进行学习，也可以根据自己的需求对某一章节进行针对性的学习。并且和传统古板的计算机书籍相比，阅读本书会带来很大的快乐。

（4）版本新颖，代表性强

本书以最新的 Android 5.0 作为讲解主线，结合 Android 4.4 的架构知识进行讲解，这样可以涵盖大多数读者群体，代表性更强。

本书的读者对象

- ☑ Android 源码分析人员。
- ☑ Android 系统架构师。
- ☑ Linux 开发人员。
- ☑ Android 物联网开发人员。
- ☑ Android 爱好者。
- ☑ Android 底层开发人员。
- ☑ Android 驱动开发人员。
- ☑ Android 应用开发人员。
- ☑ Android 传感器开发人员。
- ☑ Android 智能家居开发人员。
- ☑ Android 可穿戴设备开发人员。
- ☑ 相关培训学校的学生。
- ☑ 相关大专院校的学生。

售后服务

参与本书编写的人员还有周秀、付松柏、邓才兵、钟世礼、谭贞军、张加春、王教明、万春潮、郭慧玲、侯恩静、程娟、王文忠、陈强、何子夜、李天祥、周锐、朱桂英、张元亮、张韶青、秦丹枫。本书在编写过程中，得到了清华大学出版社工作人员的大力支持，正是各位编辑的求实、耐心和效率，才使得本书在这么短的时间内出版。另外也十分感谢我的家人在我写作时给予的巨大支持。

由于编者水平有限，如有纰漏和不尽如人意之处，诚请读者提出意见或建议，以便修订并使之更臻完善。另外我们提供了售后支持网站（<http://www.chubanbook.com/>）和 QQ 群（192153124），读者朋友如有疑问可以在此提出，一定会得到满意的答复。

编 者

目 录

第 1 章 获取并编译 Android 源码.....	1	2.5 JNIEnv 接口	61
1.1 获取 Android 源码	1	2.6 开发 JNI 程序	62
1.1.1 在 Linux 系统获取 Android 源码	1	2.6.1 开发 JNI 程序的步骤	62
1.1.2 在 Windows 平台获取 Android 源码	3	2.6.2 开发一个自己的 JNI 程序.....	63
1.2 分析 Android 源码结构	6	第 3 章 内存系统架构详解.....	66
1.2.1 总体结构.....	6	3.1 分析 Android 的进程通信机制	66
1.2.2 应用程序部分	7	3.1.1 IPC 机制介绍.....	66
1.2.3 应用程序框架部分.....	9	3.1.2 Service Manager 是 Binder 机制的上下文管 理者	67
1.2.4 系统服务部分	10	3.1.3 Service Manager 服务	84
1.2.5 系统程序库部分	12	3.2 分析匿名共享内存子系统	87
1.2.6 系统运行库部分	15	3.2.1 Ashmem 系统基础.....	87
1.2.7 硬件抽象层部分	16	3.2.2 基础数据结构	88
1.3 分析源码中提供的接口	17	3.2.3 初始化处理	89
1.3.1 暴露接口和隐藏接口	17	3.2.4 打开匿名共享内存设备文件	90
1.3.2 调用隐藏接口	23	3.2.5 实现内存映射	93
1.4 编译源码	26	3.2.6 实现读/写操作	94
1.4.1 搭建编译环境.....	26	3.2.7 实现锁定和解锁	96
1.4.2 在模拟器中运行	29	3.2.8 回收内存块	102
1.5 编译源码生成 SDK	30	3.3 分析 C++ 访问接口层	103
第 2 章 分析 JNI	35	3.3.1 接口 MemoryHeapBase	103
2.1 JNI 基础	35	3.3.2 接口 MemoryBase.....	112
2.1.1 JNI 的功能结构	35	3.4 分析 Java 访问接口层	115
2.1.2 JNI 的调用层次	36	第 4 章 硬件抽象层架构详解.....	120
2.1.3 分析 JNI 的本质	36	4.1 HAL 基础	120
2.2 分析 MediaScanner	38	4.1.1 推出 HAL 的背景	120
2.2.1 分析 Java 层.....	38	4.1.2 HAL 的基本结构	121
2.2.2 分析 JNI 层	45	4.2 分析 HAL module 架构	123
2.2.3 分析 Native (本地) 层	46	4.2.1 hw_module_t	124
2.3 分析 Camera 系统的 JNI	54	4.2.2 hw_module_methods_t.....	124
2.3.1 Java 层预览接口	54	4.2.3 hw_device_t	125
2.3.2 注册预览的 JNI 函数	56	4.3 分析文件 hardware.c.....	126
2.3.3 C/C++ 层的预览函数	59		
2.4 Java 与 JNI 基本数据类型转换	60		

4.3.1	寻找动态链接库的地址.....	126	5.1.17	释放物理页面.....	173
4.3.2	数组 variant_keys.....	126	5.1.18	分配内核缓冲区.....	174
4.3.3	载入相应的库.....	127	5.1.19	释放内核缓冲区.....	176
4.3.4	获得 hw_module_t 结构体.....	127	5.1.20	查询内核缓冲区.....	179
4.4	分析硬件抽象层的加载过程.....	128	5.2	Binder 封装库.....	179
4.5	分析硬件访问服务.....	132	5.2.1	Binder 的 3 层结构.....	180
4.5.1	定义硬件访问服务接口.....	132	5.2.2	类 BBinder.....	181
4.5.2	具体实现.....	133	5.2.3	类 BpRefBase.....	183
4.6	分析官方实例.....	134	5.2.4	类 IPCThreadState.....	185
4.6.1	获取实例工程源码.....	135	5.3	初始化 Java 层 Binder 框架.....	188
4.6.2	直接调用 service()方法的实现代码.....	136	5.3.1	搭建交互关系.....	188
4.6.3	通过 Manager 调用 service 的实现代码.....	141	5.3.2	实现 Binder 类的初始化.....	188
4.7	HAL 和系统移植.....	144	5.3.3	实现 BinderProxy 类的初始化.....	190
4.7.1	移植各个 Android 部件的方式.....	144	5.4	实体对象 binder_node.....	190
4.7.2	设置设备权限.....	144	5.4.1	定义实体对象.....	191
4.7.3	init.rc 初始化.....	148	5.4.2	增加引用计数.....	192
4.7.4	文件系统的属性.....	148	5.4.3	减少引用计数.....	193
4.8	开发自己的 HAL.....	150	5.5	本地对象 BBinder.....	194
4.8.1	封装 HAL 接口.....	150	5.5.1	引用了运行的本地对象.....	195
4.8.2	开始编译.....	153	5.5.2	处理接口协议.....	201
第 5 章	Binder 通信机制详解.....	155	5.6	引用对象 binder_ref.....	205
5.1	分析 Binder 驱动程序.....	155	5.7	代理对象 BpBinder.....	208
5.1.1	数据结构 binder_work.....	155	5.7.1	创建 Binder 代理对象.....	208
5.1.2	结构体 binder_node.....	156	5.7.2	销毁 Binder 代理对象.....	209
5.1.3	结构体 binder_ref.....	157	第 6 章	init 启动进程详解.....	213
5.1.4	通知结构体 binder_ref_death.....	158	6.1	什么是 init 进程.....	213
5.1.5	结构体 binder_buffer.....	158	6.2	入口函数.....	214
5.1.6	结构体 binder_proc.....	159	6.3	init 配置文件.....	217
5.1.7	结构体 binder_thread.....	160	6.3.1	init.rc 基础.....	217
5.1.8	结构体 binder_transaction.....	161	6.3.2	init.rc 解析.....	219
5.1.9	结构体 binder_write_read.....	162	6.4	解析 Service.....	223
5.1.10	BinderDriverCommandProtocol.....	162	6.4.1	Zygote 对应的 service action.....	224
5.1.11	枚举 BinderDriverReturnProtocol.....	163	6.4.2	init 组织 Service.....	224
5.1.12	结构体 binder_ptr_cookie 和 binder_transaction_data.....	164	6.4.3	解析 Service 用到的函数.....	226
5.1.13	结构体 flat_binder_object.....	164	6.5	解析 on.....	230
5.1.14	设备初始化.....	165	6.5.1	Zygote 对应的 on action.....	230
5.1.15	打开 Binder 设备文件.....	167	6.5.2	结构体 action.....	232
5.1.16	实现内存映射.....	168	6.5.3	解析 on 字段所在的 option.....	232
			6.6	init 控制 Service.....	233

6.6.1 启动 Zygote	233	8.7 分析实现性能统计	292
6.6.2 启动 Service	234	8.7.1 构造函数	292
6.6.3 总结 4 种启动 Service 的方式	238	8.7.2 进行性能统计	293
6.7 启动属性服务	243	8.7.3 输出统计文件	295
6.7.1 引入属性	243	8.8 剪贴板服务	302
6.7.2 设置内核变量	245	8.8.1 复制数据到剪贴板	302
6.7.3 初始化属性服务	246	8.8.2 从剪贴板粘贴数据	304
6.7.4 实现具体启动工作	247	8.8.3 管理 CBS 中的权限	306
6.7.5 获取属性值	249	第 9 章 应用程序进程详解	309
6.7.6 处理请求	251	9.1 创建应用程序	309
第 7 章 Zygote 进程详解	253	9.1.1 发送创建请求	309
7.1 Zygote 基础	253	9.1.2 保存启动参数	312
7.2 启动 Zygote	254	9.1.3 创建指定的应用程序	314
7.2.1 init.c 启动脚本	254	9.1.4 创建本地对象 LocalSocket	315
7.2.2 创建一个 Socket	258	9.1.5 接收创建新应用程序的请求	316
7.2.3 入口函数 main()	260	9.2 启动线程池	320
7.2.4 启动函数创建一个虚拟机实例	262	9.3 创建信息循环	322
7.2.5 和 Zygote 进程中的 Socket 实现连接	264	第 10 章 ART 机制架构详解	324
第 8 章 System 进程详解	271	10.1 分析 ART 的启动过程	324
8.1 启动前的准备	271	10.1.1 运行 app_process 进程	325
8.1.1 获取创建的 Socket	271	10.1.2 准备启动	329
8.1.2 启动 System 进程	272	10.1.3 创建运行实例	336
8.2 分析 SystemServer	272	10.1.4 注册本地 JNI 函数	338
8.2.1 分析主函数 main()	272	10.1.5 启动守护进程	339
8.2.2 分析函数 init2()	275	10.1.6 解析参数	340
8.3 第一个启动的 ServiceEntropyService	275	10.1.7 初始化类、方法和域	350
8.3.1 将内容写到 urandom 设备	276	10.2 进入 main() 主函数	357
8.3.2 将和设备相关的信息写到 urandom 设备	277	10.3 查找目标类	358
8.3.3 读取 urandom 设备的内容	277	10.3.1 函数 LookupClass()	359
8.3.4 发送 ENTROPY_WHAT	278	10.3.2 函数 DefineClass()	361
8.4 生成并管理日志文件	278	10.3.3 函数 InsertClass()	365
8.4.1 分析 DBMS 构造函数	278	10.3.4 函数 LinkClass()	366
8.4.2 添加 dropbox 日志文件	280	10.4 类操作	368
8.4.3 DBMS 和 settings 数据库	284	10.5 实现托管操作	370
8.5 分析 DiskStatsService	285	第 11 章 Sensor 传感器系统架构详解	376
8.6 监测系统内部存储空间的状态	289	11.1 Android 传感器系统概述	376
8.6.1 构造函数	289	11.2 Java 层详解	377
8.6.2 内存检查	290		

11.3 Frameworks 层详解	383	12.7.1 初始化蓝牙芯片	458
11.3.1 监听传感器的变化	383	12.7.2 蓝牙服务	458
11.3.2 注册监听	384	12.7.3 管理蓝牙电源	459
11.4 JNI 层详解	396	12.8 Android 系统的低功耗蓝牙协议栈	459
11.4.1 实现 Native (本地) 函数	396	12.8.1 Android 低功耗蓝牙协议栈基础	460
11.4.2 处理客户端数据	401	12.8.2 低功耗蓝牙 API 详解	460
11.4.3 处理服务端数据	403	第 13 章 Android 多媒体框架架构详解	498
11.4.4 封装 HAL 层的代码	417	13.1 Android 多媒体系统介绍	498
11.4.5 处理消息队列	422	13.2 OpenMax 框架详解	499
11.5 HAL 层详解	425	13.2.1 分析 OpenMax 框架构成	500
第 12 章 蓝牙系统架构详解	435	13.2.2 实现 OpenMax IL 层接口	504
12.1 短距离无线通信技术概览	435	13.3 OpenCore 框架详解	512
12.1.1 ZigBee——低功耗、自组网	435	13.3.1 OpenCore 层次结构	512
12.1.2 Wi-Fi——大带宽支持家庭互联	435	13.3.2 OpenCore 代码结构	513
12.1.3 蓝牙——4.0 进入低功耗时代	436	13.3.3 OpenCore 编译结构	514
12.1.4 NFC——必将逐渐远离历史舞台	436	13.3.4 操作系统兼容库	518
12.2 蓝牙技术基础	437	13.3.5 实现 OpenCore 中的 OpenMax 部分	520
12.2.1 蓝牙技术的发展历程	437	13.4 StageFright 框架详解	532
12.2.2 低功耗蓝牙的特点	437	13.4.1 StageFright 代码结构	533
12.2.3 低功耗蓝牙的架构	438	13.4.2 StageFright 实现 OpenMax 接口	533
12.2.4 低功耗蓝牙分类	439	13.4.3 分析 Video Buffer 传输流程	537
12.2.5 集成方式	439	第 14 章 音频系统框架架构详解	554
12.2.6 BLE 和传统蓝牙 BR/EDR 技术的对比	440	14.1 硬件架构的发展趋势	554
12.3 蓝牙规范详解	440	14.1.1 原始架构模式	554
12.3.1 Bluetooth 系统中的常用规范	441	14.1.2 移动处理器的解决方案	554
12.3.2 蓝牙协议体系结构	441	14.1.3 升级版高通骁龙 801	555
12.3.3 低功耗 (BLE) 蓝牙协议	443	14.2 音频系统基础	557
12.3.4 现有的基于 GATT 的协议/服务	443	14.3 音频系统的层次	559
12.3.5 双模协议栈	444	14.3.1 层次说明	559
12.3.6 单模协议栈	445	14.3.2 Media 库中的 Audio 框架	559
12.4 低功耗蓝牙协议栈详解	445	14.3.3 本地代码	562
12.4.1 低功耗蓝牙协议栈基础	445	14.3.4 分析 JNI 代码	564
12.4.2 蓝牙协议体系中的协议	446	14.3.5 分析 Java 层代码	565
12.5 TI 公司的低功耗蓝牙	448	14.4 Audio 系统的硬件抽象层	567
12.5.1 获取 TI 公司的低功耗蓝牙协议栈	448	14.4.1 Audio 硬件抽象层基础	568
12.5.2 分析 TI 公司的低功耗蓝牙协议栈	450	14.4.2 AudioFlinger 中的 Audio 硬件抽象层的实现	569
12.6 分析 Android 系统中的蓝牙模块	456		
12.7 分析蓝牙模块的源码	458		

14.4.3 真正实现 Audio 硬件抽象层	575	16.3.5 BrowserFrame	639
14.5 Kernel Driver 实现	575	16.3.6 JWebCoreJavaBridge	639
14.6 实现编/解码过程	582	16.3.7 DownloadManagerCore	639
14.6.1 AMR 编码	583	16.3.8 其他类	639
14.6.2 AMR 解码	587	16.4 数据载入器架构	639
14.6.3 解码 MP3	591	16.5 Java 层对应的 C/C++ 类库	640
第 15 章 视频系统架构详解	594	16.6 分析 WebKit 的操作过程	642
15.1 视频输出系统	594	16.6.1 WebKit 初始化	642
15.1.1 基本层次结构	594	16.6.2 载入数据	644
15.1.2 硬件抽象层架构	595	16.6.3 刷新绘制	644
15.2 MediaPlayer 架构详解	602	16.7 WebViewCore 详解	645
15.2.1 MediaPlayer 架构图解	602	第 17 章 Android 5.0 中的 WebView	652
15.2.2 MediaPlayer 的接口与架构	603	17.1 WebView 架构基础	652
15.2.3 分析 Java 部分	610	17.2 WebView 类简介	654
15.2.4 分析 JNI 部分	614	17.3 WebViewProvider 接口	656
15.2.5 核心库 libmedia.so	618	17.4 WebViewChromium 详解	659
15.2.6 服务库 libmediaservice.so	621	17.5 WebViewChromiumFactoryProvider 详解	660
15.2.7 OpenCorePlayer 实现 libopencoreplayer.so	622	17.6 AwContents 架构	663
15.2.8 对 MediaPlayer 的总结	622	17.7 实现 Mixed Content 模式	666
15.3 VideoView 详解	628	17.8 引入第三方 Cookie	667
15.3.1 构造函数	628	第 18 章 Wi-Fi 系统架构详解	670
15.3.2 公共方法	629	18.1 Wi-Fi 系统基础	670
第 16 章 WebKit 系统架构详解	635	18.2 Wi-Fi 本地部分架构	672
16.1 WebKit 系统目录	635	18.3 Wi-Fi JNI 部分架构	676
16.2 Java 层的基本框架	636	18.4 Java FrameWork 部分的源码	677
16.3 Java 层的主要类	637	18.4.1 WifiManager 详解	678
16.3.1 WebView 简介	637	18.4.2 WifiService 详解	679
16.3.2 WebViewDatabase	638	18.4.3 WifiWatchdogService 详解	688
16.3.3 WebViewCore	638	18.5 Setting 设置架构	689
16.3.4 CallbackProxy	638		

第 1 章 获取并编译 Android 源码

在分析 Android 源码之前，需要先获取 Android 系统的源码，并在自己的机器上进行编译。本章将详细讲解获取并编译 Android 5.0 源码的基本知识。另外，因为 Android 系统源码的文件数量巨大，目录结构层次复杂，所以将在本章对 Android 5.0 源码的目录结构进行整体分析，并详细介绍从 SDK 中生成 SDK 的方法。

1.1 获取 Android 源码

要想研究 Android 系统的源码，需要先获取其源码。目前，市面中的主流操作系统是 Windows、Linux 和 Mac OS。因为 Mac OS 属于类 Linux 系统，所以本书将讲解在 Windows 系统和 Linux 系统中获取 Android 源码的知识。

1.1.1 在 Linux 系统获取 Android 源码

在 Linux 系统中，通常使用 Ubuntu 来下载和编译 Android 源码。由于 Android 的源码内容很多，Google 采用了 git 的版本控制工具，并对不同的模块设置不同的 git 服务器，可以用 repo 自动化脚本来下载 Android 源码，下面逐步介绍如何获取 Android 源码的过程。

(1) 下载 repo

在用户目录下，创建 bin 文件夹，用于存放 repo，并把该路径设置到环境变量中去，命令如下：

```
$ mkdir ~/bin
$ PATH=~/.bin:$PATH
```

下载 repo 的脚本，用于执行 repo，命令如下：

```
$ curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo > ~/bin/repo
```

设置可执行权限，命令如下：

```
$ chmod a+x ~/bin/repo
```

(2) 初始化一个 repo 的客户端

在用户目录下，创建一个空目录，用于存放 Android 源码，命令如下：

```
$ mkdir AndroidCode
$ cd AndroidCode
```

进入到 AndroidCode 目录，并运行 repo 下载源码，下载主线分支的代码，主线分支包括最新修改的 bug，以及并未正式发布版本的最新源码，命令如下：

```
$ repo init -u https://android.googlesource.com/platform/manifest
```


(2) 隔一段时间或者晚上、凌晨时下载，一般这个时段更容易下载 Android 源代码。
如果看到类似如图 1-3 所示的信息，则表示连接成功，正在初始化。

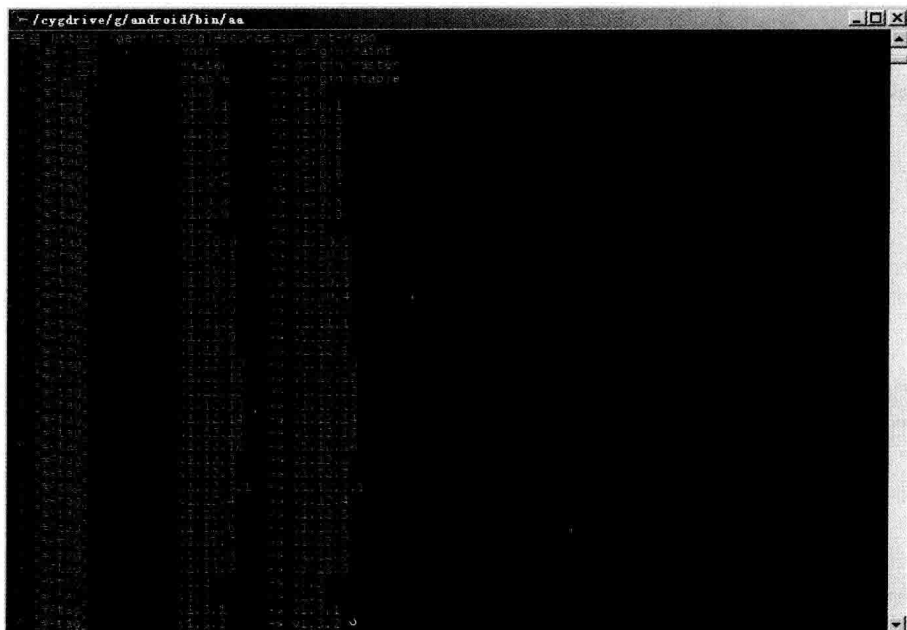


图 1-3 成功初始化

注意:

- (1) 在源代码下载过程中，源代码下载目录中看不到任何文件，打开“显示/隐藏”，会看到一个名为.repo的文件夹，这个文件夹用来保存 Android 源代码的“临时文件”。
- (2) 当文件最后下载接近完成时，会从.repo文件夹中导出 Android 源代码。
- (3) 当下载 Android 5.0 源代码完成后，可以看到 Android 源代码下载目录中会有 bionic、bootable、build、cts、dalvik 等文件夹目录，这些就是 Android 的源代码。
- (4) 如果不得不关闭计算机停止下载，那么可以在源代码下载的终端按下 Ctrl+C 或者 Ctrl+Z 快捷键停止源代码的下载，这样不会造成源代码的丢失或损坏。

1.1.2 在 Windows 平台获取 Android 源码

在 Windows 平台上获取 Android 源码的方式和在 Linux 中的获取原理相同，但是需要预先在 Windows 平台上面搭建一个 Linux 环境，此处需要用到 cygwin 工具。cygwin 的作用是构建一套在 Windows 上的 Linux 模拟环境，下载 cygwin 工具的地址是 <http://cygwin.com/install.html>。

下载成功后会得到一个名为 setup.exe 的可执行文件，经过此文件可以更新和下载最新的工具版本，具体流程如下所示。

- (1) 启动 cygwin，如图 1-4 所示。
- (2) 单击“下一步”按钮，选择第一个选项：从网络下载安装，如图 1-5 所示。
- (3) 单击“下一步”按钮，选择安装根目录，如图 1-6 所示。
- (4) 单击“下一步”按钮，选择临时文件目录，如图 1-7 所示。

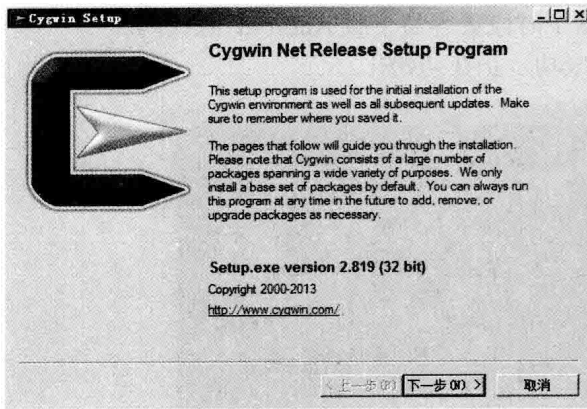


图 1-4 启动 cygwin

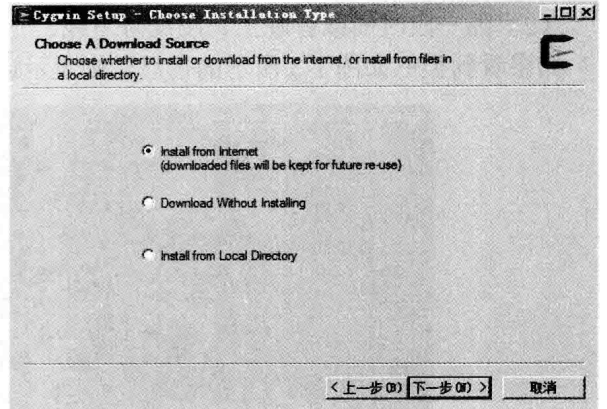


图 1-5 选择从网络下载安装

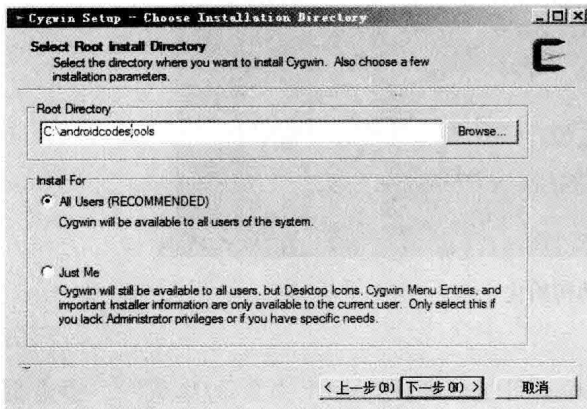


图 1-6 选择安装根目录

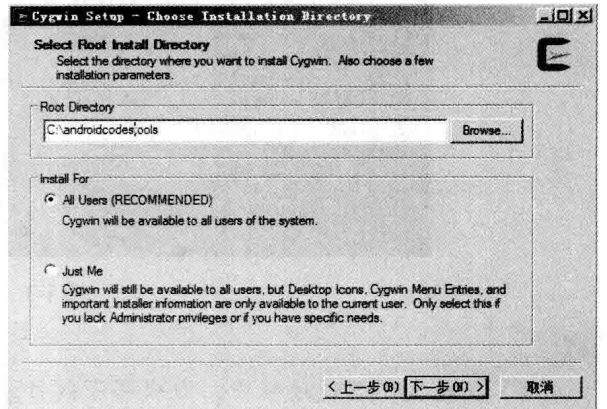


图 1-7 选择临时文件目录

(5) 单击“下一步”按钮，设置网络代理。如果所在网络需要代理，则在这一步进行设置，如果不用代理，则选择直接下载，如图 1-8 所示。

(6) 单击“下一步”按钮，选择下载站点。一般选择比较近的站点速度会比较快，这里选择的是台湾站点，如图 1-9 所示。

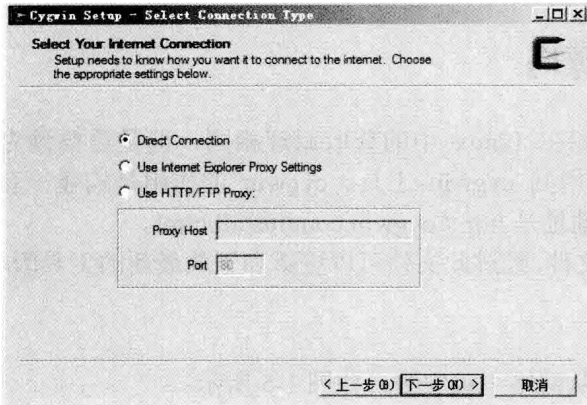


图 1-8 设置网络代理

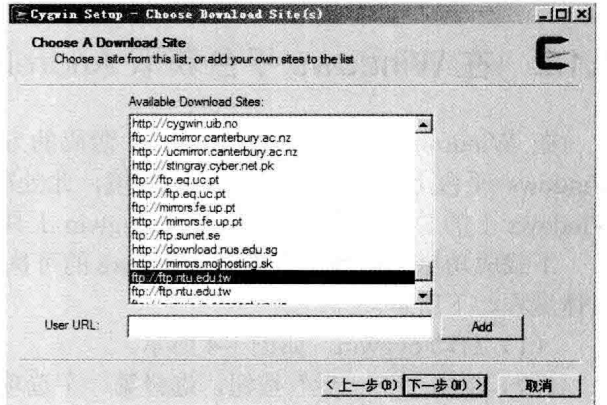


图 1-9 选择下载站点

(7) 单击“下一步”按钮，开始更新工具列表，如图 1-10 所示。

(8) 单击“下一步”按钮，选择需要下载的工具包。在此需要依次下载 curl、git、python 这些工具，如图 1-11 所示。

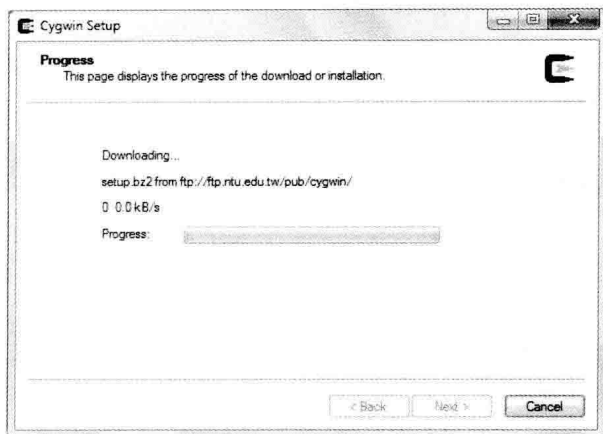


图 1-10 更新工具列表

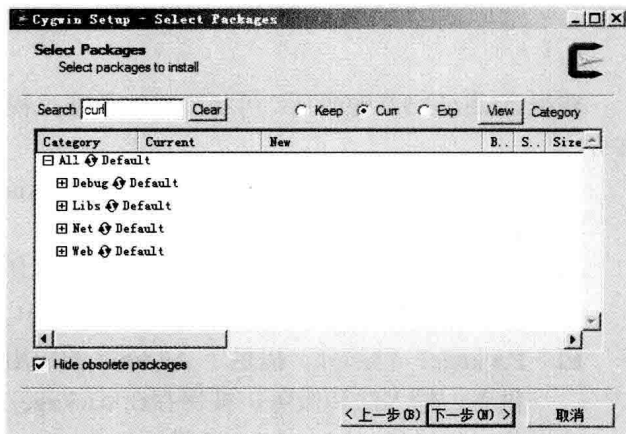


图 1-11 依次下载工具

为了确保能够安装上述工具，一定要用鼠标双击变为 Install 形式，如图 1-12 所示。

(9) 单击“下一步”按钮，开始下载安装，如图 1-13 所示。

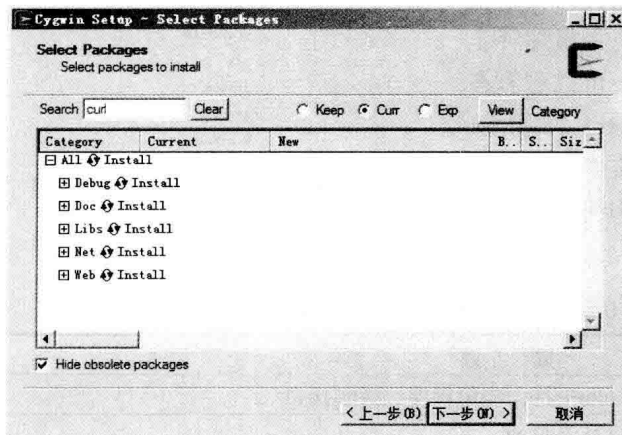


图 1-12 设置为 Install 形式

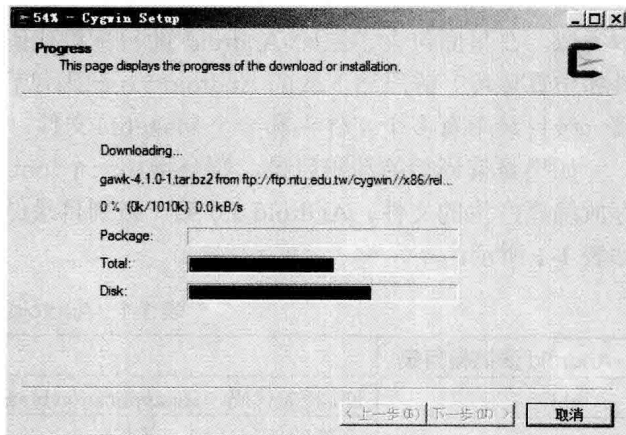


图 1-13 下载进度条

下载安装成功会出现提示信息，单击“完成”按钮即完成安装。当安装好 cygwin 后，打开 cygwin，会模拟出一个 Linux 的工作环境，然后按照 Linux 平台的源码下载方法即可下载 Android 源码。

建议读者在下载 Android 源码时，严格按照官方提供的步骤进行，地址是 <http://source.android.com/source/downloading.html>，这一点对初学者来说尤为重要。另外，整个下载过程比较漫长，需要大家耐心等待。如图 1-14 所示是笔者机器中的命令截图。

```
Cloning into 'deb'...
remote: Counting objects: 22, done
remote: Finding sources: 100% (22/22)
remote: Total 1818 (delta 416), reused 1818 (delta 416)
Receiving objects: 3% (408/1818)
Receiving objects: 100% (1818/1818), 272.73 KiB | 0 bytes/s, done.
Resolving deltas: 100% (416/416), done.
Checking connectivity... done.
Cloning into 'flo'...
remote: Counting objects: 22, done
remote: Finding sources: 100% (22/22)
remote: Total 2827 (delta 1340), reused 2827 (delta 1340)
Receiving objects: 99% (2827/2827), 1.22 MiB | 1.30 MiB/s, done.
Resolving deltas: 100% (1340/1340), done.
Checking connectivity... done.
Cloning into 'flo-keens'...
remote: Sending approximately 284.72 MiB ...
remote: Counting objects: 22, done
remote: Finding sources: 100% (22/22)
Receiving objects: 97% (882/820), 879.35 MiB | 1.39 MiB/s
```

图 1-14 在 Windows 中用 cygwin 工具下载 Android 源码

1.2 分析 Android 源码结构

获得 Android 5.0 源码后，可将源码的全部工程分为如下 3 个部分。

- ☑ **Core Project:** 核心工程部分，这是建立 Android 系统的基础，被保存在根目录的各个文件夹中。
- ☑ **External Project:** 扩展工程部分，可以使其他开源项目具有扩展功能，被保存在 external 文件夹中。
- ☑ **Package:** 包部分，提供了 Android 的应用程序、内容提供者、输入法和服务，被保存在 package 文件夹中。

本节将详细讲解 Android 5.0 源码的目录结构。

1.2.1 总体结构

无论是 Android 1.5 还是 Android 5.0，各个版本的源码目录基本类似。在里面包含了原始 Android 的目标机代码、主机编译工具和仿真环境。解压缩下载的 Android 5.0 源码包后，可以看到在第一级目录中有多个文件夹和一个 Makefile 文件，如图 1-15 所示。

如果是编译后的源码目录，则会增加一个 out 文件夹，用来存放编译产生的文件。Android 5.0 第一级别目录结构的具体说明如表 1-1 所示。



图 1-15 下载的 Android 5.0 源码

表 1-1 Android 5.0 源码的根目录

Android 源码根目录	描 述
abi	abi 相关代码，abi:application binary interface，应用程序二进制接口
art	全新的运行环境，需要和 Dalvik VM 区分开来
bionic	bionic C 库
bootable	启动引导相关代码
build	存放系统编译规则及 generic 等基础开发配置包
cts	Android 兼容性测试套件标准
dalvik	dalvik Java 虚拟机
development	应用程序开发相关
device	设备相关代码
docs	介绍开源的相关文档
external	Android 使用的一些开源的模组
frameworks	核心框架——Java 及 C++ 语言，是 Android 应用程序的框架
gdk	即时通信模块
hardware	主要是硬件适配层 HAL 代码