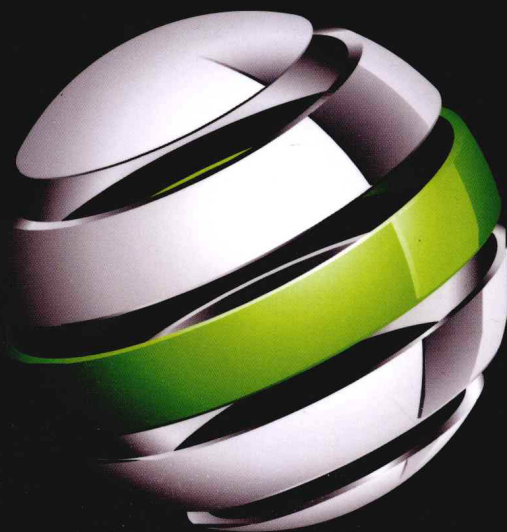


- 权威专家、精彩讲解
- 深入、细致、寓教于乐
- Android移动开发人员必备



Pro Android 2

精通 Android 2

[美] Sayed Y. Hashimi [印] Satya Komatineni [美] Dave MacLean 著

TURING 图灵程序设计丛书 移动开发系列

精通 Android 2

[美] Sayed Y. Hashimi [印] Satya Komatineni [美] Dave MacLean 著

人民邮电出版社
北 京

图书在版编目(CIP)数据

精通Android 2 / (美) 哈希米 (Hashimi, S.),
(印) 克曼特内尼 (Komatineni, S.), (美) 麦克莱恩
(MacLean, D.) 著; 杨越译. — 北京: 人民邮电出版社,
2010. 12

(图灵程序设计丛书)

书名原文: Pro Android 2

ISBN 978-7-115-24173-3

I. ①精… II. ①哈… ②克… ③麦… ④杨… III.
①移动通信—携带电话机—应用程序—程序设计 IV.
①TN929.53

中国版本图书馆CIP数据核字(2010)第207857号

内 容 提 要

本书全面介绍了针对Android平台的移动应用程序的开发, 涵盖从Android资源、Intent和Content-Provider等基本概念到OpenGL、文本到语言转换、多点触摸、主屏幕部件和Titanium Mobile的方方面面。本书按照由浅入深的顺序逐个剖析Android API, 指导读者开发Android应用程序。

本书囊括了Android开发人员所需的一切知识, 既可为Android开发人员夯实基础, 又能提高Android开发人员的技能。

图灵程序设计丛书

精通Android 2

-
- ◆ 著 [美] Sayed Y. Hashimi
[印] Satya Komatineni
[美] Dave MacLean
译 杨 越
责任编辑 杨海玲
执行编辑 李 静 毛倩倩
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京艺辉印刷有限公司印刷
- ◆ 开本: 800×1000 1/16
印张: 36.25
字数: 856千字 2010年12月第1版
印数: 1—3 000册 2010年12月北京第1次印刷
- 著作权合同登记号 图字: 01-2010-2368号

ISBN 978-7-115-24173-3

定价: 89.00元

读者服务热线: (010)51095186 印装质量热线: (010)67129223

反盗版热线: (010)67171154

版 权 声 明

Original English language edition, entitled *Pro Android 2*: by Sayed Y. Hashimi, Satya Komatineni, Dave MacLean, published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705 USA.

Copyright © 2010 by Sayed Y. Hashimi, Satya Komatineni, Dave MacLean. Simplified Chinese-language edition copyright © 2010 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由Apress L.P.授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

献给我的儿子，Sayed-Adieb。

——Sayed Y. Hashimi

献给我漂亮的妻子 AnnMarie，为她的精神支持；献给 Ashley，为她坚定的希望；献给 Nikolas，为他的友善帮助；献给 Kavitha，为她的智慧、机智和妙语；献给 Narayan，为他的聪明机智；最后献给我在印度和美国的所有家庭成员，为他们的爱。

——Satya Komatineni

献给我的妻子 Rosie 和儿子 Mike，感谢他们的鼎力支持；没有他们我将无法完成此书。献给 Max，感谢它一直在我脚边陪伴着我。

——Dave MacLean

序

无止境地构思、编码、写作和删除，这就是技术作家的咒语。技术发展如此之快，以致于在作者写完最后一句话的那一刻，就该重写了。作为技术读者，你可能已清楚认识到了这一事实，可能你也已经购买并阅读了本书，甚至可能你正在仔细阅读前言。这表明你不是一位不踏实的程序员，而是希望掌握技术背后的技术。做得不错，祝贺你做出了正确的投资。我将向你证明购买本书是明智之举。

这是市面上介绍 Android 的最优秀的图书。书中用大量篇幅介绍了 Android 的精髓，你会因为它们而对自己做出购买本书的决定感到欣慰。我是本书的技术审稿人，坦诚地讲，我希望自己有更多的内容可以编辑，但作者的工作非常扎实，我几乎找不到需要更正的问题。（不过，我曾多次埋怨他们将如此多的内容塞到一本书中，以致我的工作量增加了许多倍，直到最后一刻才忙完。）但我的损失就是你的收获：本书涵盖了你可能需要的一切与 Android 相关的知识。只需看一下目录就会知道。

根据惯例我需要介绍一下 Android 本身，也就是本书的主角。当然，你可能已经对 Android 有了一定的了解，它是来自 Google 的操作系统，Google 希望它能与 iPhone 一比高下，争夺市场领先地位，这也是你现在能读到本书的原因。作为一项技术，Android 已非常成熟，不再是最初在黑暗中摸索的无名之辈。现在随着 NexusOne（来自 Google 的基于 Android 的手机）的最新发布，Android 已成为一个强大的竞争者。2010 年将是 Google 与苹果公司在手机领域的争霸之年。两种技术有共存的空间，但 Google 强大的网络品牌力量使苹果公司的人不敢有一丝懈怠。

在巨大的 Android 市场面前，你就已经完成了两步：a) 你已选择了开发 Android 应用程序，b) 你已选择了市面上最优秀的 Android 图书。现在只剩最后一步了：翻过这一页，透彻理解 Android 的精髓。

Vikram Goyal
vikram@craftbits.com
www.craftbits.com
2010 年 1 月
澳大利亚布里斯班

致 谢

本书的编写不仅花费了作者的大量精力，而且得益于 Apress 一些才华横溢的成员以及技术审稿人投入的大量精力。衷心感谢来自 Apress 的 Steve Anglin、Douglas Pundick、Fran Parnell、Elizabeth Berry 和 Brigid Duffy，还要感谢技术审稿人 Vikram Goyal 为本书做出的贡献，他添加的注释和进行的更正是无价的。最后，作者深深感谢家人一直以来的宽容和支持。

目 录

第 1 章 Android 计算平台简介	1	2.2.2 Activity	24
1.1 面向新PC的全新平台	1	2.2.3 Intent	25
1.2 Android的历史	2	2.2.4 ContentProvider	25
1.3 Dalvik VM剖析	4	2.2.5 Service	25
1.4 比较Android与Java ME	5	2.2.6 AndroidManifest.xml	25
1.5 理解Android软件栈	8	2.2.7 AVD	26
1.6 使用Android SDK开发最终用户应用 程序	9	2.3 Hello World!	26
1.6.1 Android模拟器	9	2.4 AVD	30
1.6.2 Android UI	10	2.5 剖析Android应用程序的结构	32
1.6.3 Android基础组件	10	2.6 分析Notepad应用程序	33
1.6.4 高级UI概念	11	2.6.1 加载和运行Notepad应用程序	34
1.6.5 Android Service组件	13	2.6.2 分解应用程序	35
1.6.6 Android媒体和电话组件	13	2.7 了解应用程序生命周期	41
1.6.7 Android Java包	15	2.8 调试应用程序	43
1.7 利用Android源代码	18	2.9 小结	44
1.8 小结	19	第 3 章 使用资源、ContentProvider 和 Intent	45
第 2 章 初识 Android	20	3.1 资源	46
2.1 安装环境	20	3.1.1 字符串资源	46
2.1.1 下载JDK 6	20	3.1.2 布局资源	48
2.1.2 下载Eclipse 3.5	21	3.1.3 资源引用语法	49
2.1.3 下载Android SDK	21	3.1.4 定义资源ID供以后使用	50
2.1.4 安装ADT	22	3.1.5 已编译和未编译的Android 资源	51
2.2 了解基本组件	24	3.1.6 Android关键资源	52
2.2.1 View	24		

3.1.7 使用任意XML资源文件	58	4.4 布局管理器	124
3.1.8 使用原始资源	59	4.4.1 LinearLayout布局管理器	124
3.1.9 使用资产	60	4.4.2 TableLayout布局管理器	127
3.1.10 了解资源目录结构	60	4.4.3 RelativeLayout布局管理器	131
3.2 ContentProvider	61	4.4.4 FrameLayout布局管理器	132
3.2.1 探索Android内置的 ContentProvider	62	4.4.5 为各种设备配置自定义布局	134
3.2.2 ContentProvider的架构	67	4.5 适配器	136
3.2.3 实现ContentProvider	78	4.5.1 SimpleCursorAdapter	137
3.3 Intent	88	4.5.2 了解ArrayAdapter	137
3.3.1 Android中可用的Intent	89	4.5.3 创建自定义适配器	138
3.3.2 Intent和数据URI	90	4.6 使用Hierarchy Viewer调试和 优化布局	138
3.3.3 一般操作	91	4.7 小结	141
3.3.4 使用extra信息	92	第5章 使用菜单和对话框	142
3.3.5 使用组件直接调用活动	93	5.1 Android菜单	142
3.3.6 组件设计人员最佳实践	94	5.1.1 创建菜单	143
3.3.7 Intent类别	95	5.1.2 使用菜单组	144
3.3.8 将Intent解析为组件的规则	97	5.1.3 响应菜单项	145
3.3.9 练习使用ACTION_PICK	97	5.1.4 创建测试工具来测试菜单	146
3.3.10 练习使用GET_CONTENT操作	99	5.2 使用其他菜单类型	152
3.4 本章的更多资源	100	5.2.1 展开的菜单	152
3.5 小结	101	5.2.2 使用图标菜单	152
第4章 构建用户界面和使用控件	102	5.2.3 使用子菜单	153
4.1 Android中的UI开发	102	5.2.4 配置系统菜单	154
4.2 Android中的常见控件	107	5.2.5 使用上下文菜单	154
4.2.1 文本控件	107	5.2.6 使用交替菜单	157
4.2.2 按钮控件	110	5.2.7 使用菜单响应数据变化	160
4.2.3 列表控件	116	5.3 通过XML文件加载菜单	160
4.2.4 网格控件	119	5.3.1 XML菜单资源文件的结构	160
4.2.5 日期和时间控件	121	5.3.2 填充XML菜单资源文件	161
4.3 Android中其他有趣的控件	122	5.3.3 响应基于XML的菜单项	161
4.3.1 MapView控件	123	5.3.4 其他XML菜单标记简介	162
4.3.2 Gallery控件	123	5.4 使用Android中的对话框	163
4.3.3 Spinner控件	123	5.4.1 设计提醒对话框	164

5.4.2 设计提示对话框.....	166	7.2.2 声明和使用权限	209
5.4.3 Android对话框的特性	170	7.2.3 理解和使用自定义权限.....	211
5.4.4 重新设计提示对话框.....	170	7.2.4 理解和使用URI权限.....	216
5.5 使用托管对话框	171	7.3 使用基于位置的服务.....	216
5.5.1 理解托管对话框协议.....	171	7.3.1 地图包	216
5.5.2 将非托管对话框转换为托管对话框.....	172	7.3.2 位置包	226
5.5.3 简化托管对话框协议.....	173	7.4 小结	241
5.6 小结	180	第 8 章 构建和使用服务	242
第 6 章 2D 动画揭秘	181	8.1 使用HTTP服务	242
6.1 逐帧动画	181	8.1.1 将HttpClient用于HTTP GET请求	242
6.1.1 计划逐帧动画	182	8.1.2 将HttpClient用于HTTP POST请求	244
6.1.2 创建活动	183	8.1.3 处理异常	247
6.1.3 将动画添加到活动.....	184	8.1.4 解决多线程问题	249
6.2 布局动画	186	8.2 执行进程间通信	252
6.2.1 基本的补间动画类型.....	187	8.2.1 创建简单服务	252
6.2.2 计划布局动画测试工具.....	188	8.2.2 Android中的服务	253
6.2.3 创建活动和ListView.....	188	8.2.3 本地服务	254
6.2.4 将ListView制作成动画.....	190	8.2.4 AIDL服务.....	258
6.2.5 使用插值器	193	8.2.5 在AIDL中定义服务接口	258
6.3 视图动画	195	8.2.6 实现AIDL接口	261
6.3.1 理解视图动画	195	8.2.7 从客户端应用程序调用服务.....	262
6.3.2 添加动画	197	8.2.8 向服务传递复杂的类型.....	266
6.3.3 使用Camera实现2D图像的 深度效果	200	8.3 小结	274
6.3.4 探索AnimationListener类.....	201	第 9 章 使用媒体框架和电话 API	275
6.3.5 关于变换矩阵的一些说明.....	202	9.1 使用媒体API	275
6.4 小结	203	9.1.1 了解和使用SD卡	275
第 7 章 探索安全性和基于位置的服务	204	9.1.2 播放音频内容	278
7.1 理解Android安全性模型	204	9.1.3 setDataSource方法	281
7.1.1 安全性概念概述.....	204	9.1.4 播放视频内容	282
7.1.2 为部署签名应用程序.....	205	9.1.5 MediaPlayer的一些注意事项	284
7.2 执行运行时安全性检查.....	209	9.1.6 音频录制	284
7.2.1 进程边界上的安全性.....	209	9.1.7 视频录制	288

9.1.8	MediaStore类	293	11.1.1	ListPreference	354
9.1.9	将媒体内容添加到媒体存储	297	11.1.2	以编程方式操作首选项	361
9.2	使用电话API	299	11.1.3	CheckBoxPreference	361
9.2.1	使用SMS	299	11.1.4	EditTextPreference	363
9.2.2	使用电话管理器	305	11.1.5	RingtonePreference	365
9.3	小结	306	11.2	组织首选项	366
第 10 章	使用 OpenGL 进行 3D 图形编程	307	11.3	小结	368
10.1	了解OpenGL的历史和背景	308	第 12 章	活动文件夹	369
10.1.1	OpenGL ES	308	12.1	探索活动文件夹	369
10.1.2	OpenGL ES与Java ME	309	12.1.1	用户如何使用活动文件夹	370
10.1.3	M3G: 另一种Java ME 3D图形标准	310	12.1.2	构建活动文件夹	372
10.2	OpenGL的基本原理	310	12.2	小结	382
10.2.1	使用OpenGL ES进行基本绘制	311	第 13 章	主屏幕部件	383
10.2.2	OpenGL照相机和坐标	315	13.1	主屏幕部件的架构	383
10.3	在Android中使用OpenGL ES	318	13.1.1	什么是主屏幕部件	383
10.3.1	使用GLSurfaceView和相关类	319	13.1.2	主屏幕部件的用户体验	384
10.3.2	绘制三角形的简单测试框架	320	13.1.3	部件的生命周期	386
10.3.3	更改照相机设置	324	13.2	示例部件应用程序	390
10.3.4	使用索引添加另一个三角形	325	13.2.1	定义部件提供程序	391
10.3.5	为简单的OpenGL三角形制作动画	326	13.2.2	定义部件尺寸	392
10.4	OpenGL的运用: 形状和纹理	329	13.2.3	与部件布局相关的文件	394
10.4.1	演示中使用的简单菜单技巧	329	13.2.4	实现部件提供程序	395
10.4.2	绘制矩形	333	13.2.5	实现部件模型	397
10.4.3	使用形状	335	13.2.6	实现部件配置活动	403
10.4.4	使用纹理	345	13.3	部件局限性和扩展	407
10.4.5	绘制多个图形	349	13.4	资源	407
10.4.6	OpenGL资源	352	13.5	小结	408
10.5	小结	353	第 14 章	Android 搜索	409
第 11 章	管理和组织首选项	354	14.1	Android搜索体验	409
11.1	探索首选项框架	354	14.1.1	探索Android全局搜索	410
			14.1.2	为全局搜索启用建议提供程序	413
			14.1.3	QSB与建议提供程序交互	415
			14.2	活动与搜索键交互	416

14.2.1 常规活动上的搜索键行为	417	15.2 使用语段跟踪语音	467
14.2.2 禁用了搜索的活动的行为	422	15.3 使用音频文件代替语音	468
14.2.3 通过菜单调用搜索	423	15.4 TTS引擎的高级功能	474
14.2.4 本地搜索和相关活动	425	15.4.1 设置音频流	475
14.2.5 启用键入搜索	429	15.4.2 使用耳标	475
14.3 实现简单建议提供程序	430	15.4.3 播放静音	476
14.3.1 计划简单建议提供程序	431	15.4.4 使用语言方法	476
14.3.2 简单建议提供程序实现文件	431	15.5 将文本翻译为不同语言	477
14.3.3 实现SimpleSuggestion- Provider类	431	15.6 小结	486
14.3.4 简单建议提供程序搜索活动	434	第 16 章 触摸屏	487
14.3.5 搜索调用方活动	438	16.1 了解MotionEvent	487
14.3.6 简单建议提供程序用户体验	439	16.1.1 使用VelocityTracker	498
14.4 实现自定义建议提供程序	441	16.1.2 探索拖放操作	499
14.4.1 计划自定义建议提供程序	442	16.2 多点触摸	502
14.4.2 SuggestURLProvider项目 实现文件	442	16.3 触摸地图	508
14.4.3 实现SuggestUr1Provider类	443	16.4 手势	510
14.4.4 实现自定义建议提供程序 的搜索活动	450	16.5 小结	516
14.4.5 自定义建议提供程序描述 文件	454	第 17 章 Titanium Mobile: 基于 WebKit 的 Android 开发方法	517
14.4.6 自定义建议用户体验	455	17.1 Titanium Mobile概述	518
14.5 使用操作键和应用程序特有的搜索 数据	458	17.1.1 架构	518
14.5.1 在Android搜索中使用 操作键	458	17.1.2 Titanium生态系统	521
14.5.2 使用应用程序特定的搜索 上下文	460	17.1.3 下载和安装Titanium Developer	521
14.6 资源	461	17.2 熟悉规则: 第一个项目	528
14.7 小结	462	17.2.1 创建Titanium Mobile项目	528
第 15 章 文本到语音转换和 翻译 API	463	17.2.2 设计“Hello World”	530
15.1 Android中的文本到语音转换	463	17.2.3 配置应用程序以供调试	531
		17.2.4 封装应用程序	533
		17.2.5 将.apk文件安装到自己的 模拟器上	535
		17.3 规划真实的应用程序	536
		17.3.1 JQuery必备知识	536
		17.3.2 高级JavaScript的必备知识	539

17.3.3 理解微型模板引擎.....	541	18.3 上传应用程序.....	553
17.3.4 其他Titanium Mobile API.....	544	18.4 Android Market上的用户体验.....	554
17.4 小结.....	544	18.5 小结.....	556
第 18 章 使用 Android Market.....	546	第 19 章 前景展望和资源.....	557
18.1 成为发布者.....	546	19.1 Android的最新状态.....	557
18.1.1 遵守规则.....	547	19.1.1 基于Android的移动设备 制造商.....	557
18.1.2 开发人员控制台.....	548	19.1.2 Android应用程序商店.....	559
18.2 准备销售应用程序.....	549	19.2 Android前景展望.....	560
18.2.1 针对不同设备进行测试.....	549	19.2.1 移动操作系统概览.....	560
18.2.2 支持不同的屏幕尺寸.....	549	19.2.2 对比Android与其他移动 操作系统.....	562
18.2.3 准备上传 AndroidManifest.xml.....	550	19.2.3 HTML 5支持及其意义.....	563
18.2.4 本地化应用程序.....	550	19.3 Android资源.....	564
18.2.5 准备应用程序图标.....	551	19.3.1 核心Android资源.....	564
18.2.6 付费应用程序考虑因素.....	551	19.3.2 与Android新闻相关的资源.....	564
18.2.7 将用户引导至Market.....	552	19.4 小结.....	565
18.2.8 准备上传.apk文件.....	552		

Android 计算平台简介

计算正逐渐变得“个人化”，对计算资源的访问不再受时间和地点的限制。而正在向计算平台转变的手持设备处于这一发展趋势的前沿。曾几何时，手机不再仅仅用于通话，它们已能够承载数据和视频。值得注意的是，移动设备正成为如此强大的通用计算平台，势必成为新一代的 PC（Personal Computer，个人计算机）。据预测，甚至华硕、惠普和戴尔等传统 PC 制造商都在计划推出基于 Android OS 的各种规格的设备。各种操作系统之间、计算平台之间、编程语言之间，以及开发框架之间的竞争正在转向并将在移动设备上重演。

我们还预测，随着越来越多的 IT 应用程序开始推出移动版本，IT 行业将掀起一场移动编程热潮。为了帮助读者从这一趋势中受益，本书将介绍如何使用 Java 为运行在 Google 公司 Android 平台（<http://developer.android.com/index.html>）上的设备开发应用程序。Android 是一个面向移动开发的开源平台，它令我们非常激动，因为它是一个引入了众多全新的框架设计模式的先进平台，即使受到移动平台的限制。

本章将概述 Android 及其 SDK，简短介绍重要的包，引出每章将涉及的主题，展示如何利用 Android 源代码，重点介绍一下为 Android 平台编程的美好前景。

1.1 面向新 PC 的全新平台

目前移动电话等专用设备已跻身于重要的通用计算平台行列（参见图 1-1），这对于开发人员

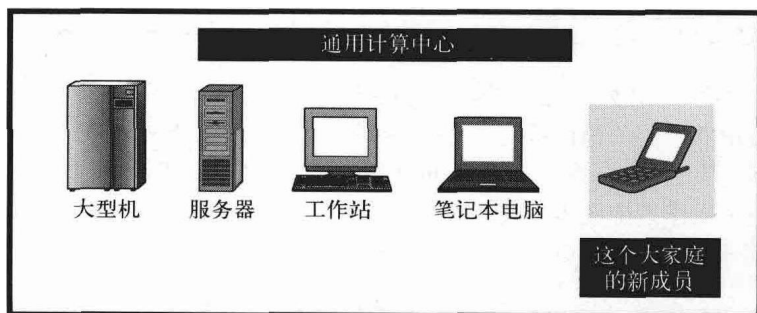


图 1-1 手持设备是全新的 PC

而言是件好事。这一新趋势使我们可通过通用计算语言使用移动设备,扩大了移动应用程序的应用范围和市场份额。

Android 平台融入了面向手持设备的通用计算理念。它是一个综合平台,包含一个基于 Linux 的操作系统,用于管理设备、内存和进程。Android 的库涵盖了电话、视频、图形、UI 编程和设备的其他许多方面。

说明 尽管针对移动设备而构建,但 Android 平台具备全功能桌面框架的所有特征。Google 公司通过 Android SDK (Software Development Kit, 软件开发工具包) 将此框架提供给 Java 程序员。在使用 Android SDK 时,很难感受到是在向移动设备编写应用程序,因为你能够访问在桌面或服务上使用的大部分类库,包括关系数据库。

Android SDK 支持大多数 Java SE (Java Platform, Standard Edition), 但 AWT (Abstract Window Toolkit, 抽象窗口工具包) 和 Swing 除外。对于 AWT 和 Swing, Android SDK 拥有自己的扩展现代 UI 框架 (extensive modern UI framework)。因为你在使用 Java 编写应用程序,所以可以想到,需要一个 JVM (Java Virtual Machine, Java 虚拟机) 来解释运行时 Java 字节码。JVM 通常会提供必要的优化,以使 Java 能够达到与 C 和 C++ 等编译语言相当的性能水平。Android 提供了自己的经过优化的 JVM 来运行已编译的 Java 类文件,以克服手持设备的内存、处理器速度和功率等限制。这个虚拟机称为 Dalvik VM (1.3 节将详细介绍)。

Java 编程语言的简单性和人们对它的熟悉,再加上 Android 丰富的类库,使 Android 成为了富有吸引力的编程平台。图 1-2 简单展示了 Android 软件栈 (1.5 节将更详细地介绍它)。

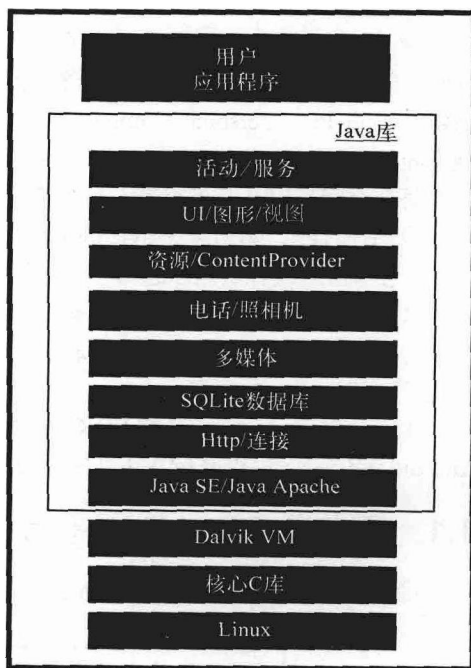


图 1-2 Android 软件栈的总体结构

1.2 Android 的历史

我们看一下 Android 是如何进入移动操作系统领域的。各种手机使用着不同的操作系统,比如 Symbian OS、微软公司的 Windows Mobile、Mobile Linux、iPhone OS (基于 Mac OS X)、Moblin (来自英特尔公司) 以及许多其他专用操作系统。迄今为止,没有一个操作系统成为事实标准。可用于开发移动应用程序的 API 和环境具有诸多限制,似乎远远落后于桌面框架。Google 公司就是在这样的环境下进军移动领域的。Android 平台具有开放、经济和代码开源的特点,还包含一个高端开发框架。

Google 于 2005 年收购了新兴公司 Android, 开始开发 Android 平台 (参见图 1-3)。Android

公司的重要成员包括 Andy Rubin、Rich Miner、Nick Sears 和 Chris White。

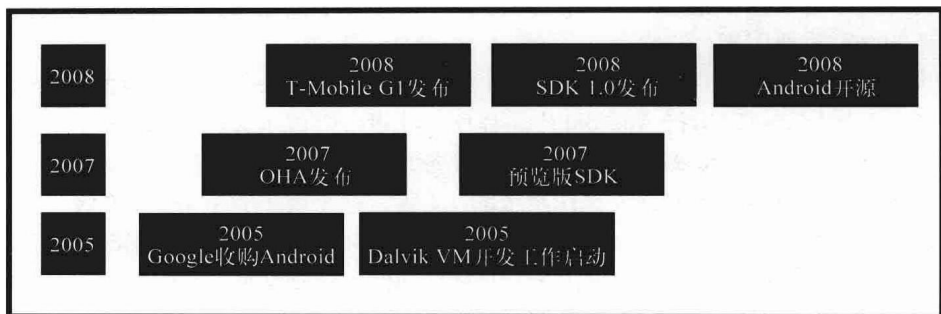


图 1-3 Android 的发展历程

2007 年年底，一些行业领先企业围绕 Android 平台成立了开放手机联盟（Open Handset Alliance, <http://www.openhandsetalliance.com>）。该联盟的一些重要成员如下。

- ❑ Sprint Nextel
- ❑ T-Mobile
- ❑ 摩托罗拉
- ❑ 三星
- ❑ 索尼爱立信
- ❑ 东芝
- ❑ 沃达丰
- ❑ Google
- ❑ 英特尔
- ❑ 德州仪器

该联盟的一个目标是迅速创新和更好地应对用户需求，它的第一个重要成果就是 Android 平台。Android 旨在满足移动运营商、手机制造商和应用程序开发人员的需求。联盟成员已承诺通过开源的 Apache 许可证 2.0 版发布重要的知识产权。

说明 手机制造商不需要支付任何许可费用，就可以在他们的手机或设备上加载Android。

2007 年 11 月，Android SDK 首次发布“预览”版。2008 年 9 月，T-Mobile 发布了 T-Mobile G1，这是第一部基于 Android 平台的智能手机。几天后，Google 公司发布了 Android SDK Release Candidate 1.0。2008 年 10 月，Google 依据 Apache 的开源许可公开了 Android 平台的源代码。

在 Android 发布时，它的一个重要架构目标是使应用程序能够彼此交互，重用彼此的组件。这种重用不仅适用于服务（service），还适用于数据和用户界面（UI）。最终，Android 平台的众多架构功能使这一开放性得以实现。第 3 章将深入介绍其中的一些功能。

由于功能完善成熟, Android 也吸引了一些早期追随者利用 Web 资源所提供的云计算模型, 使用手机自身的本地数据存储增强云计算体验。Android 对手机上的关系数据库的支持也是吸引早期采用者的一个重要因素。

2008 年年底, Google 公司发布了一款名为 Android Dev Phone 1 的手持设备, 它无需与任何手机提供商网络绑定就可以运行 Android 应用程序。此设备(成本大约为 400 美元)的目的是让开发人员试用能够运行 Android 操作系统的真实设备, 而这无需任何协议。与此同时, Google 还发布了一个 bug 修复版本, 也就是该操作系统的 1.1 版, 这个版本完全基于 1.0 版。1.0 和 1.1 版的 Android 不支持软键盘, 设备需要配备物理键盘。2009 年 4 月发布的 1.5 SDK 修复了这个问题, 该版本包含许多其他功能, 比如高级媒体录制功能、部件和活动文件夹(live folder)。(活动文件夹将在第 12 章介绍, 部件将在第 13 章介绍。)

2009 年 9 月发布了 Android 操作系统的 1.6 版, 在一个月內又发布了 Android 2.0, 这有力地推动了 2009 年圣诞节众多 Android 设备的上市。这个版本引入了高级搜索功能和文本到语音的转换功能。(Android 搜索将在第 14 章介绍, 文本到语音的转换将在第 15 章介绍。)这个版本还引入了手势和多点触控支持, 这些主题将在第 16 章介绍。

借助对 HTML 5 的支持, Android 2.0 引入了利用 HTML 实现的有趣功能。这些新的编程方式将在第 17 章介绍, 该章还将介绍 Titanium Mobile。每天诞生的基于 Android 的应用程序越来越多, 以及新型的独立在线应用程序商店。这些应用程序商店, 以及 Google 公司经营的在线 Android Market, 将在第 18 章介绍。第 19 章将分析 Android 在移动设备领域的发展前景。

1.3 Dalvik VM 剖析

作为 Android 平台的一部分, Google 公司花了大量时间思考针对低功耗手持设备的优化设计。与桌面设备相比, 手持设备在内存和速度方面落后 8~10 年。它们的计算能力也有限, 一个手持设备的总 RAM 可能只有 64 MB, 可供应用程序使用的空间可能只有 20 MB。

说明 2008 年年底发布的 T-Mobile G1 电话拥有 192 MB RAM、一张 1 GB 的 SD 卡和 528 MHz Qualcomm MSM7201A 处理器。摩托罗拉公司 2009 年年底发布的 Droid 拥有 256 MB RAM、一张 16 GB 的 microSD 卡和 550 MHz Arm Cortex 处理器。而戴尔公司最廉价的笔记本电脑都拥有 2.1 GHz 双核处理器和 4 GB RAM。

因此, 手机的性能要求非常高, 这需要手机设计人员优化一切要素。如果看一下 Android 中的包, 将会看到它们完善而丰富。据 Google 公司介绍, 即使使用经过优化的 JVM, 这些系统库也可能使用 10 MB~20 MB RAM。

这些问题使 Google 公司在许多方面重新审视标准 JVM 实现。(Dan Bornstein 对 Google 公司的这个 JVM 实现作出了重要贡献, 他编写了 Dalvik VM, Dalvik 是冰岛一个小镇的名称。)首先, Dalvik VM 获取生成的 Java 类文件, 将它们组合为一个或多个 Dalvik 可执行文件(.dex)。它重用多个类文件中的重复信息, 与未压缩的传统.jar 文件相比, 有效地将空间需求减少了一半。例