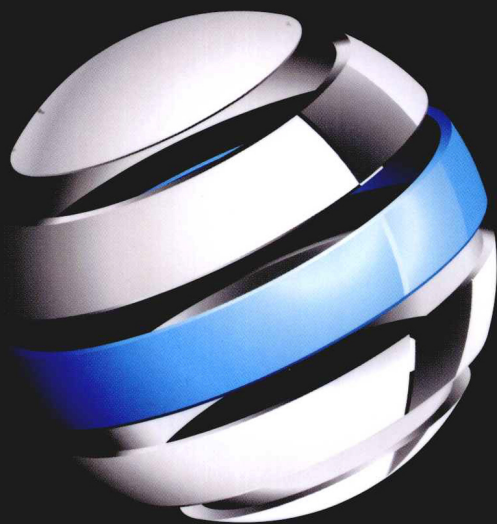


TURING

图灵程序设计丛书 移动开发系列

Apress®

- 汇集Android社区智慧
- 全面、详尽、示例支持
- 移动开发人员必读必备



Beginning Android 2

Begin the journey toward your own successful Android 2 applications

Android开发入门教程

 人民邮电出版社
POSTS & TELECOM PRESS

TURING

图灵程序设计丛书 移动开发系列

Android开发入门教程

[美] Mark L. Murphy 著
李雪飞 吴明晖 译

人民邮电出版社
北京

图书在版编目 (C I P) 数据

Android开发入门教程 / (美) 墨菲 (Murphy, M. L.)
著 ; 李雪飞, 吴明晖译. -- 北京 : 人民邮电出版社,
2010. 12

(图灵程序设计丛书)

书名原文: Beginning Android 2

ISBN 978-7-115-24116-0

I. ①A… II. ①墨… ②李… ③吴… III. ①移动通
信—携带电话机—应用程序—程序设计—教材 IV.
①TN929.53

中国版本图书馆CIP数据核字(2010)第201412号

内 容 提 要

本书是一部关于 Android 2 开发的基础教程。书中结合简单实例, 讲解了创建用户界面、内置的 Web 浏览器组件、菜单开发、SDK 工具、多媒体、ContentProvider、位置服务、地图 API、多点触摸, 以及 Android 2 新增特性。

本书适合对 Android 平台感兴趣的移动开发人员参考学习。

图灵程序设计丛书

Android开发入门教程

- ◆ 著 [美] Mark L. Murphy
- 译 李雪飞 吴明晖
- 责任编辑 朱 巍
- 执行编辑 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: 19.25
字数: 466千字 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: 01-2010-2362号

ISBN 978-7-115-24116-0

定价: 59.00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 *Beginning Android 2: Begin the journey toward your own successful Android 2 applications* by Mark L. Murphy, published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705 USA.

Copyright © 2010 by Mark L. Murphy. 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。

前 言

欢迎阅读本书

感谢你对开发 Android 应用程序怀有浓厚兴趣！毋庸置疑，越来越多的人在访问因特网服务时，都将使用所谓“非传统的”手段，如移动设备。现在我们在这个领域里做得越多，人们就会在这个领域投入更多的钱，从而使得将来构建更强大的移动应用程序会变得更加容易。Android 还很新（基于 Android 的设备在 2008 年底才刚刚出现在市场上），但相信在兼具深度和广度优势的开放手机联盟（Open Handset Alliance）的支持下，它一定能够迅速地发展壮大。

在此，首先感谢你选择这本书，由衷地希望本书能够对你有所帮助，或者至少能让你不时地露出会心一笑。

预备知识

如果你想学习为 Android 编写应用程序，那么你至少要理解 Java 编程的基本概念。Android 编程使用的是 Java 语法和一个作为 Java SE 子集类库（还有特定于 Android 的扩展）。如果此前你没有使用 Java 编写过程序，那在学习 Android 编程之前恐怕还得补上这一课。

本书不会详细介绍怎样下载或安装 Android 开发工具，包括基于 Eclipse IDE 的工具或者其他独立的工具。这些内容都可以在 Android 网站上查到。本书内容与你是否使用 IDE 应该没有什么关系^①。不过，要是你想试验本书中给出的任何示例，那么恐怕就得下载、安装和测试 Android 网站上列出的 Android 开发工具了。

本书中的有些章节可能会引用前面章节的内容。而且，也不是书中的每个示例都会给出完整的源代码，否则本书就太厚了。如果读者想要编写示例，可以从 Apress 网站（www.apress.com）下载完整的源代码^②。

① 原文似乎有误。——译者注

② 可以从图灵公司网站（www.turingbook.com）免费注册下载。——译者注

本书版本说明

本书是 Apress 和 CommonsWare 合作的产物。你现在看到的是 Apress 的版本，是纸质的，此外还有针对不同的数字图书服务的电子版，例如 Safari 版。

CommonsWare 会继续在最初文本内容的基础上不断保持更新，并随时提供给 Warescription 项目的成员，而书名则是 *The Busy Coder's Guide to Android Development*。

CommonsWare 的网站上有关于这一合作关系的 FAQ（问答），地址为 <http://commonsware.com/apress>。

源代码及许可

读者可以从 www.apress.com 下载本书的源代码。如果你想修改或将其中的示例用作其他用途，请注意源代码中的所有 Android 项目都必须遵循 Apache 2.0 许可，参见 www.apache.org/licenses/LICENSE-2.0.html。

致谢

首先要感谢 Android 开发团队，不仅谢谢他们奉献了如此优秀的产品，更因为他们在 Android Google Groups 中提供了无私帮助。特别是，我要感谢 Romain Guy、Justin Mattson、Dianne Hackborn、Jean-Baptiste Queru、Jeff Sharkey 和 Xavier Ducrohet。

本书示例用到的图标由 Nuvola 图标集^①提供：www.icon-king.com/?p=15。

^① Nuvola（名称来自意大利语的“云”，是 SKY 图标主题的改进版）是一套自由软件图标，并在 GNU LGPL 2.1 下发布。Nuvola 的作者是 David Vignoni。这套图标最初是为像 KDE 和 GNOME 之类的桌面环境所开发，现在也已经有 Windows 和 Macintosh 的版本。图标的最终版本 1.0 包含了将近 600 个图标。默认的图标为 PNG 格式，但也提供了 SVG 版本。这套图标用丰富的色彩来表现了大量常用并易于辨认的物体。大部分图标颜色为蓝色，但其他颜色也很好看。（摘自：<http://zh.wikipedia.org/zh-cn/Nuvola>。）——编者注

目 录

第 1 章 Android 开发概述1	第 4 章 基于 XML 的布局 18
1.1 智能手机编程的挑战.....1	4.1 何谓基于XML的布局..... 18
1.2 Android由哪些部分构成.....2	4.2 为什么使用基于XML的布局..... 18
1.3 你能够控制什么.....3	4.3 举个例子..... 19
第 2 章 项目和目标4	4.4 什么时候加@符号.....20
2.1 基本概念.....4	4.5 怎样在Java中使用布局文件.....20
2.2 创建项目.....5	4.6 把故事讲完.....20
2.3 项目结构.....5	第 5 章 使用基本的部件22
2.3.1 根目录.....5	5.1 标签.....22
2.3.2 主Activity.....6	5.2 按钮.....23
2.3.3 资源.....6	5.3 图像.....23
2.3.4 编译结果.....7	5.4 字段.....24
2.4 AndroidManifest.xml文件.....7	5.5 复选框.....25
2.4.1 一开始是根元素.....8	5.6 单选按钮.....27
2.4.2 权限、编排和应用程序.....8	5.7 视图.....28
2.4.3 应用程序总要做点什么.....9	5.7.1 特性.....28
2.4.4 确保最大兼容性.....10	5.7.2 方法.....29
2.4.5 版本 = 控制.....10	5.7.3 颜色.....29
2.5 模拟器和目标.....11	第 6 章 使用容器30
2.5.1 虚拟设备.....11	6.1 线性布局.....30
2.5.2 设定目标.....13	6.1.1 LinearLayout的概念和特性.....30
第 3 章 简单的应用程序14	6.1.2 LinearLayout示例.....33
3.1 创建项目.....14	6.2 相对布局.....36
3.2 剖析Activity.....15	6.2.1 RelativeLayout的概念和属性.....36
3.3 构建和运行Activity.....17	6.2.2 RelativeLayout示例.....37

6.3 表格布局	40	9.6.2 动态添加内容	87
6.3.1 TableLayout的概念和特性	40	9.6.3 自动翻转	88
6.3.2 TableLayout示例	42	9.7 滑动的抽屉	89
6.4 滚动	42	9.8 其他容器	91
第7章 使用选择部件	45	第10章 输入法框架	92
7.1 适配器	45	10.1 键盘, 硬还是软	92
7.2 列表	46	10.2 按需定制	92
7.3 微调控件	49	10.3 修改附属键	95
7.4 网格	51	10.4 适应布局	96
7.5 自动完成字段 (至少减少35%的输入)	53	10.5 释放创造力	97
7.6 画廊	55	第11章 使用菜单	98
第8章 使用列表	57	11.1 选项菜单	98
8.1 初步改进	57	11.1.1 创建选项菜单	98
8.2 动态列表	58	11.1.2 添加菜单项和子菜单	99
8.3 更好, 更快, 更强	60	11.2 上下文菜单	100
8.3.1 使用convertView	61	11.3 简单的示例	100
8.3.2 使用持有者模式	62	11.4 扩展的示例	104
8.4 交互式列表	64	11.4.1 菜单的XML结构	104
8.5 可重用列表	68	11.4.2 菜单项与XML	105
8.6 选用其他适配器	74	11.4.3 创建菜单	106
第9章 高级部件和容器	75	第12章 字体	107
9.1 选择日期和时间	75	12.1 珍惜已有字体	107
9.2 时钟	78	12.2 更多字体	108
9.3 进度条	79	12.3 字形介绍	109
9.4 滑动选择	79	第13章 嵌入WebKit浏览器	111
9.5 选项卡	80	13.1 小型浏览器	111
9.5.1 构建	80	13.2 加载内容	112
9.5.2 规则	80	13.3 导航内容	113
9.5.3 使用	81	13.4 扩展应用程序	114
9.5.4 增强	83	13.5 设置、首选项和选项	115
9.5.5 Intent和View	84	第14章 显示弹出消息	116
9.6 翻转	85	14.1 弹出Toast	116
9.6.1 手工翻转	86	14.2 提醒框	117

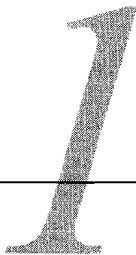
14.3 检查效果	117	18.3 多标签浏览	142
第 15 章 处理线程	120	第 19 章 处理旋转	145
15.1 了解处理程序	120	19.1 销毁问题	145
15.1.1 消息	120	19.2 异同	145
15.1.2 Runnable	123	19.3 更多保存	149
15.2 就地运行	123	19.4 DIY 旋转	151
15.3 我的 UI 线程到哪去了	123	19.5 强制解决问题	152
15.4 异步观感	123	19.6 综述	154
15.4.1 原理	124	第 20 章 处理资源	156
15.4.2 AsyncTask、泛型和 Vararg	124	20.1 资源	156
15.4.3 AsyncTask 的各个阶段	125	20.2 字符串理论	156
15.4.4 示例任务	125	20.2.1 纯文本字符串	157
15.5 附加说明	129	20.2.2 字符串格式	157
第 16 章 处理 Activity 生命周期事件	130	20.2.3 样式文本	157
16.1 Activity 的状态	130	20.2.4 样式字符串格式	158
16.2 Activity 的生命周期	131	20.3 获取图片	160
16.2.1 onCreate() 和 onDestroy()	131	20.4 XML: 资源之路	162
16.2.2 onStart()、onRestart() 和 onStop()	131	20.5 杂项	164
16.2.3 onPause() 和 onResume()	131	20.5.1 维度	164
16.3 优美的状态	132	20.5.2 颜色	165
第 17 章 创建 Intent 过滤器	133	20.5.3 数组	165
17.1 你有什么意图	133	20.5.4 因人而异	166
17.1.1 Intent 组成	133	第 21 章 使用首选项	171
17.1.2 Intent 路由	134	21.1 获取想要的内容	171
17.2 叙述 Intent	135	21.2 编辑首选项	172
17.3 缩小接收器范围	136	21.3 目前的框架	172
17.4 暂停警告	137	21.4 让用户自己选择	173
第 18 章 启动活动和子活动	138	21.5 添加“分层”结构	176
18.1 对等活动和子活动	138	21.6 弹出对话框	177
18.2 启动	139	第 22 章 管理和访问本地数据库	180
18.2.1 制作 Intent	139	22.1 数据库示例	180
18.2.2 进行调用	139	22.2 SQLite 快速入门	181
		22.3 从头开始	182

22.4	设置表	184	27.3.1	第一步: 创建提供程序类	212
22.5	数据	184	27.3.2	第二步: 提供URI	217
22.6	有因必有果	185	27.3.3	第三步: 声明属性	218
22.6.1	Raw查询	186	27.3.4	第四步: 更新清单文件	218
22.6.2	常规查询	186	27.4	更改通知支持	219
22.6.3	使用构造器进行构建	186	第 28 章	请求和要求许可	220
22.6.4	使用Cursor	187	28.1	请求许可	220
22.7	无所不在的数据	188	28.2	声明许可	221
第 23 章	访问文件	190	28.2.1	通过清单文件强制实施 许可	222
23.1	使用的数据	190	28.2.2	在其他地方强制实施许可	222
23.2	读取与写入	192	28.3	别忘了文档	223
第 24 章	充分利用 Java 库	196	第 29 章	创建服务	224
24.1	外部限制	196	29.1	通过类创建服务	224
24.2	Ant和JAR	197	29.2	单例	225
24.3	参照脚本	197	29.3	清单文件的作用	226
24.4	滴酒不沾	200	29.4	事件提醒	227
24.5	评审脚本	200	29.4.1	回调	227
第 25 章	通过 Internet 进行通信	201	29.4.2	广播Intent	228
25.1	REST和Relaxation	201	29.5	远程服务与其他代码	229
25.2	通过Apache HttpClient操作HTTP	201	第 30 章	调用服务	230
25.3	解析响应	203	30.1	联系的纽带	230
25.4	要考虑的问题	205	30.2	接收广播内容	232
第 26 章	使用内容提供程序	206	第 31 章	利用通知提醒用户	234
26.1	数据片段	206	31.1	发布通知的类型	234
26.2	获得句柄	207	31.1.1	硬件通知	235
26.3	查询	207	31.1.2	图标	235
26.4	适应环境	208	31.2	查看运行中的通知发布	235
26.5	舍与得	209	第 32 章	访问基于位置的服务	238
26.6	感知BLOB	210	32.1	位置提供程序: 它们知道你藏在 哪里	238
第 27 章	构建内容提供程序	211	32.2	自我定位	239
27.1	剖析	211	32.3	移动	240
27.2	类型	212			
27.3	创建内容提供程序	212			

32.4 我们到了吗	241	36.2 多合一	265
32.5 测试	241	36.2.1 考虑规则, 而不是位置	265
第 33 章 使用 MapView 和 MapActivity		36.2.2 考虑物理尺寸	266
显示地图	242	36.2.3 避免使用实际像素	266
33.1 条款无情	242	36.2.4 选择可缩放的Drawable	266
33.2 添加项问题	242	36.3 量身定制	267
33.3 基本要素	243	36.3.1 添加<supports-screens>	267
33.4 练习控制	244	36.3.2 资源和资源集	268
33.4.1 缩放	244	36.3.3 查找尺寸	268
33.4.2 居中	245	36.4 一切都是模拟的	269
33.5 地形起伏	245	36.4.1 密度不同	269
33.6 层上加层	245	36.4.2 调整密度	270
33.6.1 Overlay类	246	36.4.3 访问实际设备	270
33.6.2 绘制ItemizedOverlay	246	36.5 充分利用形势	271
33.6.3 处理屏幕单击	247	36.5.1 用按钮代替菜单	271
33.7 MyLocationOverlay	248	36.5.2 使用简单的Activity代替选	
33.8 关键所在	249	项卡	271
第 34 章 呼叫处理	250	36.5.3 整合多个Activity	272
34.1 向管理者报告	250	36.6 示例: EU4You	272
34.2 亲自进行呼叫	250	36.6.1 第一个版本	272
第 35 章 开发工具	253	36.6.2 固定字体大小	277
35.1 层次结构管理	253	36.6.3 固定大小的图标	278
35.2 令人愉快的Dalvik调试详细演示	257	36.6.4 使用空间	278
35.2.1 日志记录	259	36.6.5 不是浏览器会怎样	279
35.2.2 文件推拉	259	36.7 合作伙伴的错误有哪些	280
35.2.3 屏幕截图	260	第 37 章 手机的处理	281
35.2.4 位置更新	260	37.1 该应用程序包含显式指令	281
35.2.5 接入呼叫和消息	261	37.2 按钮	282
35.3 存储卡	262	37.3 有保障的市场	282
35.3.1 创建卡的映像	262	37.4 细枝末节	283
35.3.2 插入卡	263	37.4.1 Archos 5 Android Internet	
第 36 章 处理多种屏幕尺寸	264	Tablet	283
36.1 默认设置	264	37.4.2 Motorola CLIQ/DEXT	284
		37.4.3 Motorola DROID/Milestone	284

37.4.4	Google/HTC Nexus One	284	38.3	处理API变更	287
37.4.5	Motorola BACKFLIP	284	38.3.1	检测版本	287
第 38 章	处理平台变更	285	38.3.2	包装API	287
38.1	品牌管理	285	第 39 章	未来何去何从	292
38.2	让人头疼的更多问题	286	39.1	问题——部分答案	292
38.2.1	视图层次结构	286	39.2	源代码	292
38.2.2	变更资源	287	39.3	获得最新的信息	293

Android 开发概述



大体来说，Android 设备指的就是手机。目前，虽然大家都在讨论 Android 技术在其他领域（如车载“电脑”中）的应用，但谈论的焦点还是手机应用程序。对开发人员来说，为手机开发应用程序既能体验到新鲜刺激，但又要应对诸多挑战。

1.1 智能手机编程的挑战

从好的方面说，基于 Android 的智能手机确实非常之炫。通过移动设备提供因特网服务可以追溯到 20 世纪 90 年代中期的 HDML（Handheld Device Markup Language，手持设备标记语言）。但是直到最近几年，手机上网才真正成为现实。由于手机短消息的风行和苹果 iPhone 的出现，手机作为能上网的设备而迅速走红。因此，开发 Android 应用程序能让你在日新月异的细分市场（能够上网的手机）中，充分体验新技术（Android）带来的刺激。

可是，在你不得不为一些麻烦事而绞尽脑汁时，问题就来了。

任何有 PDA 或手机编程经验的人，都对手机的问题心知肚明——各方面都太小了，例如：

- 屏幕小（你不可能听到这样问题：“你的口袋里装了一块 24 英寸的 LCD，还是……？”）；
- 键盘小（如果有键盘的话）；
- 定位设备（如果有）不是给人制造麻烦（想想那些丢了手写笔的人），就是不够精确（粗大的手指跟“多点触摸”LCD 可不容易和谐）；
- 与桌面电脑或服务器相比，CPU 的速度和内存有限；
- 你可以使用任何编程语言 and 开发框架，前提是设备制造商必须选中并且将它写入了手机的固件里。

更令人难受的是，运行在手机上的应用程序无法回避一个事实：它们运行在手机上。

人们在打电话遇到问题时，通常很容易着急上火，这也是为什么 Verizon Wireless 的“你听得见我说话吗？”系列广告流行那么多年的原因所在。同样还是这些人，他们在看到你的程序

让他的手机“中断”时，同样也会气急败坏的；程序为什么会导致手机“中断”？以下是可能的原因。

- 耗尽 CPU 资源导致电话接不进来。
- 在手机来电或需要接听时不能安静地隐藏到后台，因为程序与手机操作系统不能很好地协调运作。
- 导致手机的操作系统宕机，例如让内存像过筛子一样泄漏。

事实上，开发手机程序与开发桌面程序、开发网站或开发服务器端程序有很大不同：所用的工具不同、框架的行为不同，开发程序受到的限制更多。

而 Android 的目的就是把你解放出来。

- 让你可以使用常用的编程语言（Java），常用的库（例如，某些 Apache Commons API），甚至使用你熟悉的开发工具（Eclipse）。
- 让你有一个相当严格而且独立的框架，以便确保你的程序能够在手机里成为一个“好公民”，不会妨碍其他程序或手机本身的正常运行。

聪明的读者想必已经猜到了，本书很在程度上就是要讲述怎么使用这个框架，怎么编写在其权限范围内运行的程序，以及怎样挖掘它的潜力。

1.2 Android 由哪些部分构成

在开发桌面应用程序时，你是“自己领域的主宰”。你可以启动主窗口以及任意多个子窗口，如对话框。从开发人员的角度看，一切都是你说了算，你可以想方设法地去利用操作系统提供的各种特性。而且很大程度上，你都不必考虑同一台计算机上还会同时运行其他应用程序。如果要与其他程序交互，一般都是通过 API，如 JDBC（Java Database Connectivity，Java 数据库连接），或者构建于该 API 之上的构架，来与 MySQL 或者其他数据库通信。

开发 Android 程序的概念也类似，但是不同的封装和组织方式，确保了手机不会轻易崩溃。以下是 Android 应用程序中会用到的主要组件。

- **Activity**：用户界面是由 Activity 构建而成的。可以将 Android 程序中的 Activity 想象成桌面应用程序中的窗口或对话框。虽然 Activity 可以没有自己的用户界面，但多数情况下那些“没头没脑”的代码更适合以 ContentProvider 或 Service 的形式实现。
- **ContentProvider**：ContentProvider 用于为设备中存储的数据提供了一个抽象层，以便不同的应用程序访问。Android 开发模型鼓励你将自己的数据公开给自己的和其他的应用程序。通过创建 ContentProvider 来实现数据共享，可以对别人访问数据的方式拥有完全的控制权。

- **Service:** Activity 和 ContentProvider 都是短命的，随时可能被关掉。但 Service 则是可以独立于 Activity 而长期运行的（如果有必要的话）。因此，可以使用 Service 来检查更新 RSS 新闻源，或者在启动它的 Activity 退出之后仍然通过它来播放音乐。
- **Intent:** Intent 是系统消息，在设备内部运行，向应用程序发布各种事件，包括硬件状态变化（例如，插入了 SD 卡）、收到数据（例如，收到一条短信），以及应用程序事件（例如，用户通过设备的主菜单打开了 Activity）。除了响应 Intent 之外，你还可创建自定义 Intent 以打开其他 Activity，或者让它在特定的情形下给你发送通知（例如，当用户走到离某个地点 100 米范围内时，就触发某个 Activity）。

1.3 你能够控制什么

Android 提供了很多特性，用来辅助你开发应用程序。

- **存储:** 可以在应用程序中封装不会发生变化的数据，比如图标或帮助文件等。也可以占用设备中的一部分空间，用于保存包含用户输入或其他有用数据的文件或数据库。在用户提供大块存储空间的情况下（如 SD 卡），也可以从中读取或向其中写入文件。
- **网络:** Android 设备通常随时都可以上网，而且接入网络的方式不止一种。你可以在任何级别上利用因特网访问，从使用原始的 Java 套接字，到在应用程序中嵌入内置的基于 WebKit 的浏览器的小部件。
- **多媒体:** Android 设备具有播放和录制音频和视频的功能。虽然具体的实现方式因设备而异，但能够通过请求设备可以获知它的功能，然后再随心所欲地利用其多媒体功能。不管是播放音乐、拍照片，还是通过麦克风来录音。
- **GPS (Global Positioning System, 全球定位系统):** Android 设备会频繁访问位置服务（如 GPS），以便应用程序知道当前设备在地球上的什么位置。进而，可以显示该位置的地图，或者以其他方式来利用位置数据，例如在设备丢失的情况下跟踪设备的移动。
- **手机服务:** 当然，Android 设备通常都是手机。因此，你的程序可以拨打电话、发送和接收 SMS (Short Message Service, 短信息服务) 消息，以及实现最新的电话技术所能实现的一切功能。

下载并安装好最新的 Android SDK (Android Software Development Kit, Android 软件开发包) 和基于 Eclipse 的 ADT (Android Developer Tools, Android 开发人员工具) 插件^①之后, 就可以动手开发 Android 应用程序了。本章介绍与开发 Android 应用程序有关内容。

2.1 基本概念

在开发 Android 应用程序之前, 必须创建一个相应的 Android 项目。如果你使用 Eclipse 开发的话, 那就是创建一个 Eclipse 项目。这个项目用来保存所有源代码、资源 (例如国际化字符串)、第三方 JAR 及相关内容。Android 构建工具 (集成在 Eclipse 中的或独立的构建工具) 会将项目的内容转换成 APK (Android Package, Android 包) 文件, 这也就是 Android 应用程序。这些工具还可以帮你把 APK 文件部署到 Android 模拟器, 或者部署到真实的 Android 设备中, 以便测试。

项目中的最关键的一个组成部分是描述文件 (manifest), 即 AndroidManifest.xml。这个文件中包含着应用程序的“目录”, 列出了主要的应用程序组件、权限, 等等。Android 在运行时会通过这个描述文件将应用程序与操作系统关联起来。此外, Android Market (或其他独立的“应用程序商店”) 也会用到这些描述文件, 以便使 Android 2.0 的应用程序不会被安装到运行 Android 1.5 的设备上。

要使用模拟器来测试应用程序, 必须先创建一个 AVD (Android Virtual Device, Android 虚拟设备)。多数情况下, 都需要创建很多 AVD, 每个 AVD 分别模拟一种基于特硬件的 Android 设备。此外, 还可以为不同屏幕尺寸、不同 Android 版本创建不同的 AVD。

在创建项目和 AVD 时, 需要为 Android 指定要使用的 API 级别。API 级别是一个整数值, 不

^① Android SDK 和 ADT 都可以从 Android 开发人员站点 (<http://developer.android.com/>) 下载。如果官方站点打不开, 可以访问 <http://tinyurl.com/y66sfw7> 或自行搜索下载; ADT 可以通过 Eclipse 更新管理器安装, 安装说明参见 <http://www.cn-cuckoo.com/2010/05/08/eclipse-adt-1618.html>。——译者注

同的整数值映射到不同的 Android 版本（例如，API 3 对应 Android 1.5）。创建项目时，可以告诉 Android 你的应用程序支持的最小和最大 API 级别。而在创建 AVD 时，则是告诉 Android 新建的这个 AVD 要模拟哪个级别的 API。这样，就可以看到自己的应用程序在支持不同 Android 版本的不同（模拟）设备上运行的效果了。

上面提到的所有概念都会在本章中详细介绍。

2.2 创建项目

要想在命令行中创建项目（以便使用 ant 等构建工具），需要运行 `android create project` 命令。这个命令带有一些参数，用于指定应用程序代码所属的 Java 包、应用程序针对的 API 级别，等等。成功运行命令后，会得到一个文件夹，其中包含构建基本的 Android 应用程序所需的全部文件。

下面是一个 `android create project` 命令的示例：

```
android create project --target 2 --path ./FirstApp --activity FirstApp --package  
apt.tutorial
```

如果你想使用 Eclipse 开发 Android 应用程序，而不是使用 `android create project` 方式，那么就需要使用 Eclipse 的新项目向导来创建一个新 Android 应用程序。

注意 本书的源代码是按照使用命令行构建工具构建应用程序而创建的。如果你要使用 Eclipse，可以创建空 Eclipse 项目，再将代码导入到项目中。^①

2.3 项目结构

每个 Android 项目都包含一套特定的目录树结构（非常类似 Java 项目），而 Android 构建系统就是基于该目录树结构组织的。不过，Android 的这套目录结构还是有点与众不同。下面我们就来简单介绍一下 Android 项目的目录结构，以便读者有个大概的印象，特别是让你能够轻松地看明白本书给出的示例代码。

2.3.1 根目录

在创建了新 Android 项目后（例如通过 `android create project` 命令），你会在项目的根目录下发现一些文件和子目录，以下是对它们的简要说明。

^① 译者是这样做的：在 New Android Project 对话框中，选择 create project from existing source，通过 Browse... 按钮找到源代码目录，打开具体示例项目的目录，Eclipse 即可自动填写有关新项目的信息。——译者注