移动开发实用丛书

MOBILE DEVELOPMENT PRACTICAL SERIES

Android

移动开发一本就够

【美】Lauren Darcey 【美】Shane Conder 著　　李卉 张魏 祝延彬 译

本书提供 Android 开发从入门到进阶的全程指导，囊括详尽的开发实践技术和前瞻的市场应用技巧！

本书源代码可通过此网址下载：
Informit.com/title/9780321743015

人民邮电出版社
POSTS & TELECOM PRESS

Android

移动开发一本就够

Addison Wesley

本书涵盖了大量具有实践价值的信息、示例代码和最佳案例，它将成为你构造和发布Android移动应用程序，并将其推广到市场的良师益友。凭借在移动和无线开发领域所积累的丰富经验，Shane Conder 和 Lauren Darcey 为你提供了打造成功的 Android 项目所需的全部知识和技术：从概念到设计，从代码编写、测试到应用程序打包、发布，一切囊括其中。作者详细阐述了 Android 开发平台，如何高效地设计 Android 应用程序，以及时下最为流行的用户交互方式实践。本书还为 Android 大多数重要的 API 提供了详尽的代码，包括数据存储、网络、通话、位置服务（LBS）、多媒体、3D 图形以及硬件。

此次改版后的每一个章节的相关内容都升级为可以匹配最新版本的 Android SDK、工具、应用以及硬件设备。所有示例代码全部经过更新，并且在各大制造商的主流机型上进行了测试，包括 HTC、摩托罗拉和 ARCHOS。新增了包括完整应用在内的示例代码。还新增了以下内容：

- 9 个全新的章节，涵盖了网络 API、Android NDK、应用程序扩展、用户管理、数据同步、备份和高级用户输入等内容；
- 极大地丰富了部分章节，包括 Android Manifest 文件、内容提供器、应用程序设计与测试；
- 新增热点话题关注：蓝牙、手势、语音识别、应用程序小插件、动态文件夹、动态壁纸和全局搜索；
- 升级 3D 图形编程，包括 OpenGL ES 2.0 的使用；
- 关于跟踪不同设备开发兼容性问题的全新章节，帮助你设计适用于最简单的手机和最新潮的平板电脑的应用程序；
- 更多适用于不同设备的应用程序设计、开发和测试的小技巧；
- 新增一个关于 Eclipse 使用技巧的附录。

本书是 Android 开发团队中每一位成员不可或缺的资源。本书适合不同层次的软件开发人员、团队主管、项目经理、测试人员、QA 专业人士、软件架构师甚至市场销售人员。

Lauren Darcey 是一家专注于 Android、iPhone、黑莓以及其他主流移动平台技术和咨询的软件公司的技术主管。凭借近 20 年专业软件产品的研发经验，Darcey 被公认为企业架构和商业级移动应用开发的权威。

Shane Conder 是一位富有经验的开发人员，他从事移动和嵌入式开发 10 余年，设计和开发了针对 BREW、J2ME、Palm、BlackBerry、Windows Mobile、iPhone 和 Android 等诸多平台的许多商业应用程序，以及若干世界范围内流行的小应用。

在 androidbook.blogspot.com 上可以找到这两位作者。

封面设计：王建国

ISBN 978-7-115-25181-7

分类建议：计算机／程序设计／移动开发
通信技术／移动通信

人民邮电出版社网址：www.ptpress.com.cn

ISBN 978-7-115-25181-7

定价：75.00 元

移动开发实用丛书
MOBILE DEVELOPMENT PRACTICAL SERIES

Android 移动开发一本就够

【美】Lauren Darcey 【美】Shane Conder 著　　李卉 张魏 祝延彬 译

人民邮电出版社
北京

Download at http://www.pin5i.com/

图书在版编目（CIP）数据

Android移动开发一本就够 / (美) 达西(Darcey,L.), (美) 康德尔 (Conder,S.) 著 ; 李卉, 张魏, 祝延彬译. -- 北京 : 人民邮电出版社, 2011.7
(移动开发实用丛书)
ISBN 978-7-115-25181-7

Ⅰ. ①A… Ⅱ. ①达… ②康… ③李… ④张… ⑤祝… 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(2011)第051604号

版 权 声 明

Authorized translation from the English language edition, entitled ANDROID WIRELESSAPPLICATION DEVELOPMENT, 2E, 9780321743015 by CONDER, SHANE; DARCEY, LAUREN, published by Pearson Education, Inc, publishing as Addison-Wesley Professional, Copyright © 2011.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and POSTS & TELECOMMUNICATIONS PRESS Copyright © 2011.

本书封面贴有 Pearson Education（培生教育出版集团）激光防伪标签，无标签不得销售。
北京市版权局著作权合同登记号 图字：01-2011-1388 号

移动开发实用丛书

Android 移动开发一本就够

◆ 著 ［美］Lauren Darcey ［美］Shane Conder
译 李 卉 张 魏 祝延彬
责任编辑 姚予疆
执行编辑 刘 洋
◆ 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 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
三河市海波印务有限公司印刷
◆ 开本：787×1092 1/16
印张：33
字数：824 千字 2011 年 7 月第 1 版
印数：1–4 000 册 2011 年 7 月河北第 1 次印刷
著作权合同登记号 图字：01-2011-1388 号
ISBN 978-7-115-25181-7

定价：75.00 元

读者服务热线：(010)67129264 印装质量热线：(010)67129223
反盗版热线：(010)67171154

Download at http://www.pin5i.com/

内容提要

本书涵盖了 Android 开发从概念、技术到市场推广应用的全部话题，内容包括：Android 概览、Android 应用程序设计精髓、Android 用户界面设计精髓、常用 Android API、更多 Android 应用程序设计原则、对外发布你的 Android 应用程序和附录 7 个部分。附录还包含了 Android 开发常用工具（如模拟器、DDMS、调试桥、Eclipse 和 SQLite 等）的使用指南。

本书不仅适合 Android 应用程序开发人员阅读，还能为 QA 测试人员提供指导。另外本书还可以帮助项目经理更好地管理项目进程和项目团队，帮助市场推广人员更好地把握市场，创造令人瞩目的销售业绩。

Download at http://www.pin5i.com/

致　谢

如若我们没有收到这许许多多人（他们是我们的编辑团队、同事、朋友以及家人）的指导和鼓励，此书将无缘问世。我们想要感谢 Android 开发人员社区、Google 和开放手机联盟（OHA）的眼光和信任，以及他们所提供的经验与支持。在整个编撰过程中，Pearson Education（Addison-Wesley）的编辑团队始终给予我们鼓励并提供着专业的指导。在此要特别感谢 Trina MacDonald、Olivia Basegio、Songlin Qiu 以及我们一流的技术支持团队：Doug Jones 和 Charles Stearns（还有 Dan Galpin、Tony Hillerson 和 Ronan Schwarz，他们为本书的第一版做出了评审）。Dan Galpin 还慷慨地为本书的所有 TIP、NOTE 和 WARNING 设计了生动而形象的 Android 图标。我们还将感谢 Ray Rischpater，感谢他的长期鼓励和专业的写作建议。必须提到的还有 Amy Badger，感谢她为本书所绘制的漂亮的“瀑布”插图，以及 Hans Bodlaender，感谢他允许我们使用其业余开发的漂亮的“象棋字”。

Download at http://www.pin5i.com/

译者序

Android 是一个真正意义上的开放性移动设备综合平台，它自发布以来就一直受到业界的极大关注。开放手机联盟（Open Handset Alliance，OHA）中诸多重量级企业和厂商的合作，不但打破了移动领域存在了很久的垄断问题，其开放的氛围也造就了更多样化的硬件设备。它为程序员和用户提供了前所未有的丰富资源和便捷体验，Android 平台对程序员完全免费开放，大大促进了 Android 平台上应用程序的发展。从 2009 年 7 月到 2011 年 3 月，不到两年的时间里，Android 市场中的应用程序发布数量有了大幅增长，从开始的每月 1669 款应用发展到最高每月 32323 款，增长速度相当惊人。

Android 的开放性质决定了这是一个极具活力的平台，事实也正是如此。Android 平台的更新换代速度非常快，几乎每隔几个月就会有更新发布。例如本书主要针对的 Android 2.2（Froyo）版本，相较于之前版本的 Android，新增了对 Flash 10、Wi-Fi、蓝牙通信、动态壁纸（Live Wallpaper）、HTML 5、手势、来电照片等功能的支持。通过 Android，我们可以真切地感受到新技术强大的生命力，也不禁令人感叹移动生活发展的日新月异。

同时，随着业务的拓展，3C（计算机、通信与消费电子产品）市场也有了新的变化，平板电脑逐渐进入了人们的视野，肩负起了改变用户使用电子产品习惯的重任，而 Android 同样能够很好地胜任这一点——它的扩展性能够极大地满足未来平板电脑甚至平板电视等电子产品的需求。

从整个互联网的发展来看，越来越丰富的移动应用是大势所趋，快捷的生活方式呼唤同样快捷的游戏、应用和工具，Android 无疑是承载这些内容的极佳载体，Android 的发展速度有目共睹，支持 Android 的硬件设备数量也在迅猛增长着，市场之门才刚刚打开。

这一切足以证明 Android 是极具发展前景的平台。作为和两位作者身处同一领域的从业者，我们很幸运能够通过本书的翻译，和他们一起深入了解 Android 这一先进而优秀的平台，乃至见证它的成长过程。两位作者细致入微地讲解了 Android 开发涉及的方方面面的问题，并凭借自身丰富的开发经验给出了许多具有实战性的建议和技巧，我们非常乐于和大家分享。编辑邮箱：liuyang@ptpress.com.cn。

在本书的翻译过程中，我们还要特别感谢马煜坤、贾学来、陈勇、李科达、张皓博、王璁、罗一峰、褚彬池、王思敏、徐帆、张瑶、高龄、朱嘉明、张万东、舒文琼等人提供的支持和帮助，他们也为本书作出了贡献。

译者

2011 年 4 月于北京

Download at http://www.pin5i.com/

作者简介

Lauren Darcey

Lauren Darcey 是一家专注于移动开发的软件公司的技术主管，所涉及的技术包括 Android、iPhone、Blackberry、Palm Pre、BREW 和 J2ME，还包括咨询服务。拥有 20 年专业软件产品研发经验的 Lauren 被公认为企业架构和商业级移动应用开发的权威。Lauren 获得了加州大学圣克鲁兹分校授予的计算机科学专业学士学位。

Shane Conder

Shane Conder 拥有丰富的开发经验，在过去的 10 年中，他把主要精力集中在移动和嵌入式开发领域，迄今已设计并开发了基于 Android、iPhone、BREW、Blackberry、J2ME、Palm 和 Windows Mobile 的诸多商业软件。Shane 撰写了大量关于移动通信行业和评估移动开发平台的文章，这些文章发表在他的个人技术博客上，在业界受到了广泛的关注。Shane 获得了加州大学授予的计算机科学专业学士学位。

Download at http://www.pin5i.com/

前　言

在以开放手机联盟（Open Handset Alliance，OHA）和 Google 公司为先驱的开拓下，Android 作为一个热门、新潮、免费、开源的移动平台在无线通信领域引起了不小的波澜，而你手中的这本书正能够为软件开发团队提供设计、开发、测试、调试和发布 Android 应用程序的一系列指导。如果你是一位资深的移动开发人员，那么书中一些小提示和小技巧可以帮助你更好地利用 Android 的某些独特功能，从而更加顺利地完成整个开发过程；相反，如果你刚刚踏入移动开发领域，那么本书将为你提供所有需要掌握的知识和技能，从而帮助你平稳而顺利地从传统软件开发过渡到移动开发，尤其是过渡到其中最富有前景的新平台——Android。

此书为谁而写

本书涵盖了我们数年来在移动开发领域所积累的经验，以及成功进行移动项目开发所需的全部概念和实用技术，包括移动开发过程与传统软件开发的区别，还包括帮助你节约宝贵时间和规避隐患的技巧。不管你的项目有多大规模，本书都能为你提供帮助。

本书主要针对以下几类人群。

（1）想要学习编写专业 Android 应用程序的软件开发人员

本书主体部分主要针对有 Java 程序开发经验而无需具有移动开发经验的程序员。但是，具有丰富移动开发经验的开发人员更能够学习到如何充分运用和发挥 Android 的优势，同时了解它与市面上其他移动开发技术的区别。

（2）担负 Android 程序测试任务的 QA 人员

无论 QA 工程师采用的是“黑盒”还是“白盒”的测试方法，这本书对于他们而言都具有非凡的价值。我们花了几章的篇幅来讲解有关 QA 的问题，例如制定稳健的测试计划，为应用程序开发漏洞跟踪系统，管理手机设备以及利用 Android 工具进行完整测试的方法。

（3）筹建和管理 Android 开发团队的项目经理

本书可以帮助项目经理完成从筹建开发团队、聘用开发人员到执行 Android 项目的整个过程。我们探讨了有关项目风险管理和顺利进行项目开发的有关话题。

（4）其他读者

这本书不仅适用于软件开发人员，而且适用于一个正在寻找潜在的垂直市场应用软件（Vertical Market Application）的公司，一个正在构思某个“新奇酷”手机软件的企业家，抑或一个正在探寻新手机乐趣的发烧友。同时，这本书也可以帮助某些正在为某个特定目的（例如可行性分析）而对 Android 进行评估的企业，或者任何拥有 Android 手机并且乐于创造新的移动应用软件的人。

书中解答的问题

本书为你解答了以下问题。

① Android 是什么？不同版本的 SDK 差别在哪里？

② Android 和其他移动技术有什么不同？开发人员如何利用 Android 的优势？

③ 开发人员如何使用针对 Java 的 Eclipse 开发环境来开发和调试模拟器和真机上的 Android 应用程序？

④ Android 应用程序是怎么架构起来的？

Download at http://www.pin5i.com/

⑤ 如何为移动应用程序（尤其是为 Android）设计强大的用户界面？
⑥ Android SDK 提供哪些功能？开发人员如何使用它们？
⑦ 移动应用开发过程同传统桌面应用程序开发有什么区别？
⑧ 对于 Android 开发而言，什么样的开发策略是最佳的？
⑨ 项目经理、开发人员和测试人员在策划、开发和测试移动应用程序时应该关注什么？
⑩ 开发团队如何设计能够供发布使用的“防弹（bulletproof）”的 Android 应用程序？
⑪ 开发团队如何封装 Android 应用程序以供部署（deployment）？
⑫ 开发团队如何利用 Android 应用程序谋求利润？
⑬ 本书第二版有些什么新的内容？

本书的结构

本书分为七个部分。前五个部分是开发人员感兴趣的主要问题，第六部分和第七部分则为项目经理、QA 人员和开发人员提供了更多有用的信息。

接下来对本书的各个部分作一个简单的介绍。

第一部分：Android 概览

第一部分简单介绍了 Android，阐释了它和其他移动平台的区别。你将会熟悉 Android SDK 和相关工具，了解如何搭建开发环境，以及怎样在模拟器及手机上编写和运行你的第一个 Android 应用程序。

第二部分：Android 应用程序设计精髓

第二部分介绍了设计一个 Android 应用程序所需要了解的原则。你将学习到如何构建 Android 应用程序，怎样在你的项目中包含诸如字符串、图形及其他用户界面组件等资源的方法。

第三部分：Android 用户界面设计精髓

第三部分对在 Android 中应该怎样设计用户界面进行了更深入的探讨。你将学习到核心用户界面元素——视图的有关知识，以及 Android SDK 提供的基本的屏幕绘制和动画功能。

第四部分：使用常用 Android API

第四部分由一系列章节组成，每一章都对 Android SDK 中最重要的 API 进行了更深入的讲解，例如数据和存储 API（包括文件和数据库的使用，以及 Content Provider）、网络电话、定位服务（LBS）、多媒体和 3D 图形 API，以及可选硬件 API。

第五部分：更多 Android 应用程序设计原则

第五部分涵盖了更多的 Android 应用程序设计原则，例如通知和服务。

第六部分：对外发布你的 Android 应用程序

第六部分涵盖了移动软件开发的全过程，还为项目经理、软件开发人员和 QA 人员提供了不少小提示和小技巧。

第七部分：附录

第七部分针对 Android 开发工具提供了几个有用的快速入门引导：模拟器、ADB 和 DDMS、Eclipse 使用技巧，以及 SQLite 指南。

第二版的不同之处

当我们开始撰写本书第一版的时候，市面上还没有 Android 设备。当我们开始编写后不久才有了第一款 Android 设备，但仅仅在美国可以买到。而今天，全世界范围内已经有数十款不同的 Android 设备了。从本书第一版问世后，Android 平台已经经历了大量革新与进步。

Download at http://www.pin5i.com/

Android SDK 也有了许多新的特性，许多开发工具也进行了不少必要的升级。Android 作为一种技术，已经在移动市场上拥有坚实的基础，占领了不可撼动的一席之地。

在这一新版中，我们在书的内容上进行了一系列的改动，使之更好、更完整，相信读者会更加青睐。除了增添了许多新的内容外，我们还重新测试并升级了所有的内容（包括文字和示例代码），以便能够适应最新的 Android SDK。下面是此版书中引入的新内容以及有所强调的内容的提要。

① 涵盖了最新、最好的 Android 工具和功能包。

② 对原有章节的内容进行了提升，增添了新的小节。

③ 对示例代码和应用程序等进行了完整修改——它们以话题的方式被很好地组织了起来。

④ 新增了 9 章，涵盖了许多新的 SDK 特性，包括网络 API、Android NDK、应用程序范围扩展、用户管理、数据同步、备份、高级用户输入和兼容性等话题。

⑤ 诸如 Android Manifest 文件、Content Provider、应用程序设计和测试等话题都有了自己专门的章节。

⑥ 更新的 3D 图形编程，包括 OpenGL ES 2.0。

⑦ 涵盖许多热门话题，例如蓝牙、手势（gestures）、语音识别、App Widget、Live Folder、Live Wallpaper（动态壁纸）和全局搜索。

⑧ 更多来源于实践的技巧，它们可以帮助你更好地为不同的目标设备设计、开发和测试应用程序，其中包括一个专门讲述兼容性问题的新章节。

⑨ 新增一个专门讲解 Eclipse 使用技巧的附录。

正如你所看到的，我们将 Android 所提供的许多最热门、最激动人心的特性囊括进了本书。我们不是浅尝辄止，而是深入到每一个现有的章节，对内容进行了改进和提升，同时增添了不少新的章节。最后，我们还对其中的部分内容做出了更详尽的说明，除此之外，我们还根据读者的反馈修改了部分内容，在此对他们表示感谢！

本书使用的 Android 开发环境

本书所有 Android 应用程序代码均在如下环境中编写。

① Windows 7 和 Mac OS X 10.6.4。

② Eclipse Java IDE Version 3.5（Galileo）。

③ Eclipse JDT 插件和 Web Tools Platform（WTP）。

④ Sun Java SE 开发工具包（JDK）6 Update 20。

⑤ Android SDK Version 2.2，API Level 8（Fro Yo）：

- ADT Plug-in for Eclipse 0.9.9；
- NDK Tools Revision 4b；
- SDK Tools Revision 7。

⑥ Android 手机设备：T-Mobile G1、HTC Nexus One、HTC Evo 4G、摩托罗拉 Droid 和 ARCHOS 5 internet tablet。

可用的补充材料

本书中的源代码可以在网站 www.informit.com/title/9780321743015 上找到。

我们还建立了一个博客：http://androidbook.blogspot.com，其中包括各种类型的 Android 话题，同时还提供了读者的反馈、问题和更深入的信息。你还可以在上面找到各种技术文章的链接。

Download at http://www.pin5i.com/

获取更多信息

读者在互联网上可以找到许多活跃而有用的 Android 开发人员社区。另外，下面的一系列网站可供 Android 开发人员获取更多资源。

- **Android Developer Website：** Android SDK 和开发人员参考网站
  http://developer.android.com/
- **Stack Overflow：** 一个包含许多 Android 技术信息的网站和为开发人员而建立的官方支持论坛
  http://stackoverflow.com/questions/tagged/android
- **Open Handset Alliance：** 供 Android 制造商、运营商和开发人员参考
  http://www.openhandsetalliance.com/
- **Android Market：** 购买和销售 Android 应用程序
  http://www.android.com/market/
- **Mobiletuts+：** 移动开发指南，包括 Android
  http://mobile.tutsplus.com/category/tutorials/android/
- **anddev.org：** 一个 Android 开发人员论坛
  http://www.anddev.org
- **Google Team Android Apps：** 开源 Android 应用程序
  http://apps-for-android.googlecode.com/
- **FierceDeveloper：** 为无线开发人员提供技术周刊
  http://www.fiercedeveloper.com/
- **Wireless Developer Network：** 提供无线领域的每日新闻
  http://www.wirelessdevnet.com/
- **Developer.com：** 面向开发者提供移动技术文章的网站
  http://www.developer.com/

本书约定

本书遵循如下约定。

① ➥用于指示代码接续，即该行代码应与上行代码保持在同一行。出现该符号的代码行略去了缩进。

② 代码和程序术语使用等宽（monospace）字符表示。

同时，本书提供如下类型的信息框：

Tip

提供同当前内容相关的有用信息和提示。

Note

为读者补充感兴趣的相关信息。

Warning

提供可能遇到的陷阱的有关信息，以及避免它们的方法。

联系作者

我们欢迎您的评论、提问以及反馈。请访问本书博客：http://androidbook.blogspot.com 或通过电子邮件联系我们：androidwirelessdev+awad2e@gmail.com。

Download at http://www.pin5i.com/

目　录

第一部分　Android概览

第1章　Android简介 …… 2

1.1　移动软件开发简史 …… 2
1.1.1　遥想当年 …… 2
1.1.2　“砖头” …… 3
1.1.3　无线应用协议（WAP） …… 4
1.1.4　专用移动平台 …… 6
1.2　开放手机联盟（OHA） …… 7
1.2.1　无线Google …… 8
1.2.2　成立开放手机联盟 …… 8
1.2.3　制造商：设计Android手机 …… 9
1.2.4　移动运营商：传播Android体验 …… 9
1.2.5　内容提供商：开发Android应用程序 …… 9
1.2.6　充分发挥Android的优势 …… 10
1.3　Android平台的不同之处 …… 10
1.3.1　Android：下一代平台 …… 11
1.3.2　免费开放源代码 …… 11
1.3.3　熟悉和廉价的开发工具 …… 12
1.3.4　合理的开发人员学习曲线 …… 12
1.3.5　强大的应用程序开发支持 …… 12
1.3.6　丰富、安全的应用集成 …… 13
1.3.7　无障碍发布 …… 13
1.3.8　为应用程序提供的“自由市场” …… 13
1.3.9　新生并逐渐成长的平台 …… 14
1.4　Android平台 …… 14
1.4.1　Android的基本体系结构 …… 14
1.4.2　安全与许可权限 …… 16
1.4.3　开发Android应用程序 …… 17
小结 …… 18
参考资料和更多信息 …… 18

第2章　搭建你的Android开发环境 …… 19

2.1　配置你的开发环境 …… 19
2.1.1　配置操作系统以适应设备调试 …… 20
2.1.2　配置Android硬件以供调试 …… 20
2.1.3　升级Android SDK …… 20
2.1.4　Android软件开发包的问题 …… 21
2.2　探索Android SDK …… 21
2.2.1　理解Android SDK许可协议 …… 21
2.2.2　阅读Android SDK文档 …… 22
2.2.3　探索Android应用程序框架 …… 23
2.2.4　了解Android工具 …… 24
2.2.5　探索Android示例应用程序 …… 27
小结 …… 28
参考资料和更多信息 …… 28

第3章　编写第一个Android应用程序 …… 29

3.1　测试你的开发环境 …… 29
3.1.1　在Eclipse工程中导入Snake应用程序 …… 29
3.1.2　为Snake工程创建Android虚拟设备（AVD） …… 31
3.1.3　为Snake工程创建运行配置参数 …… 31
3.1.4　在Android模拟器中运行Snake应用程序 …… 32
3.2　构建你的第一个Android应用程序 …… 33
3.2.1　创建和配置新的Android工程 …… 33
3.2.2　Android应用程序的核心文件和目录 …… 34
3.2.3　为新工程创建Android虚拟设备 …… 35

Download at http://www.pin5i.com/

3.2.4 为工程配置运行参数 …… 35
3.2.5 在模拟器中运行Android应用程序 …… 36
3.2.6 在模拟器中调试Android应用程序 …… 37
3.2.7 为Android应用程序添加日志记录 …… 39
3.2.8 为应用程序添加媒体支持 …… 40
3.2.9 为应用程序添加位置服务（LBS） …… 42
3.2.10 在硬件上调试应用程序 …… 43
小结 …… 45
参考资料和更多信息 …… 45

第二部分 Android应用程序设计精髓

第4章 剖析Android应用程序 …… 48
4.1 掌握重要的Android术语 …… 48
4.2 使用应用程序Context …… 48
4.3 使用Activity执行应用程序任务 …… 49
4.3.1 Android Activity的生命周期 …… 50
4.3.2 使用Intent管理Activity跳转 …… 53
4.4 使用服务 …… 56
4.5 接收和广播Intent …… 57
小结 …… 57
参考资料和更多信息 …… 58
第5章 使用Android Manifest文件定义应用程序 …… 59
5.1 配置Android Manifest文件 …… 59
5.2 管理你的应用程序标识 …… 63
5.2.1 为你的应用程序指定版本 …… 63
5.2.2 给你的应用程序加上名称和图标 …… 63
5.3 强制指定应用程序的系统需求 …… 64
5.3.1 针对特定SDK版本 …… 64
5.3.2 强制指定应用程序平台需求 …… 66
5.3.3 使用外部库 …… 67
5.4 注册Activity和其他应用程序组件 …… 68
5.4.1 使用Intent过滤器为应用程序指派主入口点Activity …… 68
5.4.2 配置其他Intent过滤器 …… 68
5.5 使用许可权限 …… 69
5.5.1 注册应用程序所需的许可权限 …… 69
5.5.2 注册应用程序授予其他应用程序的许可权限 …… 70
5.6 探索Manifest文件的其他参数 …… 71
小结 …… 71
参考资料和更多信息 …… 71
第6章 管理应用程序资源 …… 72
6.1 什么是资源 …… 72
6.1.1 存储应用程序资源 …… 72
6.1.2 理解资源目录的层级结构 …… 72
6.1.3 资源的类型 …… 73
6.1.4 存储不同的资源类型 …… 74
6.1.5 使用程序访问资源 …… 76
6.2 使用Eclipse设置简单资源 …… 77
6.3 使用资源 …… 79
6.3.1 使用String（字符串）资源 …… 79
6.3.2 使用格式化的字符串资源 …… 80
6.3.3 使用字符串数组 …… 81
6.3.4 使用Boolean（布尔）资源 …… 82
6.3.5 使用Integer（整型）资源 …… 82
6.3.6 使用颜色 …… 83
6.3.7 使用尺寸 …… 83
6.3.8 使用简单Drawable资源 …… 84
6.3.9 使用图像 …… 85
6.3.10 使用动画 …… 87
6.3.11 使用菜单 …… 89
6.3.12 使用XML文件 …… 90
6.3.13 使用原始文件 …… 91
6.3.14 资源引用 …… 91

Download at http://www.pin5i.com/

6.3.15 使用布局（Layout）…… 92
6.3.16 使用样式（Style）…… 96
6.3.17 使用主题（Theme）…… 98
6.4 引用系统资源 ……98
小结 ……99
参考资料和更多信息 ……99

第三部分 Android用户界面设计精髓

第7章 探索用户界面屏幕元素 …… 102
7.1 Android视图和布局简介 …… 102
7.2 使用TextView显示文本 …… 103
7.2.1 配置布局和尺寸 …… 104
7.2.2 在文本中创建上下文链接 …… 104
7.3 获取用户数据 …… 105
7.3.1 使用EditText控件获取用户输入文本 …… 106
7.3.2 使用Spinner控件为用户提供可选输入 …… 109
7.4 使用按钮、多选框和单选组 …… 110
7.4.1 使用基本按钮 …… 110
7.4.2 使用多选框和开关（Toggle）按钮 …… 111
7.4.3 使用RadioGroup和RadioButton …… 112
7.5 获取用户输入的日期和时间 …… 114
7.6 使用指示器向用户显示数据 …… 115
7.7 使用SeekBar调整进度 …… 116
7.7.1 使用RatingBar显示评分数据 …… 117
7.7.2 使用Chronometer显示时间推移 …… 118
7.7.3 显示时间 …… 119
7.8 为用户提供选项和上下文菜单 …… 119
7.8.1 使用选项菜单 …… 119
7.8.2 使用ContextMenu …… 121
7.9 处理用户事件 …… 123
7.9.1 监听触摸模式状态改变 …… 123
7.9.2 监听全屏幕上的事件 …… 124
7.9.3 监听长时单击 …… 124
7.9.4 监听焦点改变 …… 125
7.10 使用对话框 …… 126
7.10.1 探索不同类型的对话框 …… 126
7.10.2 跟踪对话框的生命周期 …… 127
7.10.3 使用自定义对话框 …… 128
7.11 使用样式 …… 128
7.12 使用主题 …… 130
小结 …… 131

第8章 使用布局设计用户界面 …… 132
8.1 创建Android用户界面 …… 132
8.1.1 使用XML资源创建布局 …… 132
8.1.2 使用代码创建布局 …… 133
8.2 组织用户界面 …… 135
8.3 使用内建的布局类 …… 138
8.3.1 使用FrameLayout …… 140
8.3.2 使用LinearLayout …… 141
8.3.3 使用RelativeLayout …… 142
8.3.4 使用TableLayout …… 144
8.3.5 在屏幕上使用多布局 …… 145
8.4 使用内建的View容器类 …… 146
8.4.1 使用数据驱动容器 …… 147
8.4.2 使用标签组织视图 …… 150
8.4.3 添加滚动支持 …… 152
8.4.4 探索其他视图容器 …… 153
小结 …… 154

第9章 绘图与动画 …… 155
9.1 屏幕绘图 …… 155
9.2 使用文字 …… 158
9.2.1 使用默认字体 …… 158
9.2.2 使用自定义字体 …… 159
9.2.3 测量文字所需的屏幕尺寸 …… 160
9.3 使用位图 …… 160
9.4 使用图形 …… 161
9.4.1 用XML资源定义图形 …… 161

Download at http://www.pin5i.com/

9.4.2 使用程序定义图形 …… 162
9.4.3 绘制各种图形 …… 162
9.5 使用动画 …… 167
9.5.1 使用逐帧动画 …… 167
9.5.2 使用补间动画 …… 168
小结 …… 172

第四部分 使用常用Android API

第10章 使用Android数据和存储API …… 174
10.1 使用应用程序首选项 …… 174
10.1.1 创建私有和共享首选项 …… 174
10.1.2 搜索和读取首选项 …… 175
10.1.3 新增、修改和删除首选项 …… 175
10.1.4 使用Android文件系统查找首选项数据 …… 176
10.2 使用文件和目录 …… 177
10.2.1 探索Android应用程序目录 …… 177
10.2.2 使用Android文件系统操作其他目录和文件 …… 179
10.3 使用SQLite数据库存储结构化数据 …… 180
10.3.1 创建SQLite数据库 …… 181
10.3.2 创建、修改和删除数据库记录 …… 183
10.3.3 在SQLite数据库中执行查询 …… 185
10.3.4 关闭和删除SQLite数据库 …… 189
10.3.5 设计持久数据库 …… 189
10.3.6 向应用程序界面绑定数据 …… 191
小结 …… 195
参考资料和更多信息 …… 195
第11章 使用内容提供器在应用程序间共享数据 …… 196
11.1 探索Android内容提供器 …… 196
11.1.1 使用MediaStore内容提供器 …… 196
11.1.2 使用CallLog内容提供器 …… 198
11.1.3 使用浏览器（Browser）内容提供器 …… 199
11.1.4 使用联系簿（Contacts）内容提供器 …… 200
11.1.5 使用UserDictionary（用户词典）内容提供器 …… 202
11.1.6 使用Settings内容提供器 …… 202
11.2 修改内容提供器的数据 …… 202
11.2.1 添加记录 …… 203
11.2.2 修改记录 …… 203
11.2.3 删除记录 …… 204
11.3 使用内容提供器增强你的应用程序 …… 204
11.4 实现内容提供器 …… 208
11.4.1 实现内容提供器接口 …… 209
11.4.2 定义数据URI …… 209
11.4.3 定义数据列 …… 210
11.4.4 实现重要的内容提供器方法 …… 210
11.4.5 更新Manifest文件 …… 214
11.5 使用动态文件夹（Live Folder） …… 215
小结 …… 217
参考资料和更多信息 …… 217
第12章 使用Android网络API …… 218
12.1 移动网络基础 …… 218
12.2 访问因特网（HTTP） …… 218
12.2.1 从Web读取数据 …… 219
12.2.2 使用HttpURLConnection …… 219
12.2.3 解析从网络获取的XML …… 220
12.2.4 异步操作 …… 222
12.2.5 使用AsyncTask …… 222
12.2.6 使用线程进行网络访问 …… 223
12.2.7 显示从网络资源获取的图像 …… 224
12.2.8 获取Android网络状态 …… 226
小结 …… 227
参考资料和更多信息 …… 227

Download at http://www.pin5i.com/

第13章 使用Android Web API ····· 228
13.1 使用WebView浏览Web ····· 228
13.1.1 设计带有WebView控件的布局 ····· 229
13.1.2 向WebView控件载入内容 ····· 229
13.1.3 为WebView控件添加功能 ····· 230
13.2 使用WebKit延伸Web应用 ····· 232
13.2.1 浏览WebKit API ····· 232
13.2.2 扩展Android Web应用程序的功能 ····· 232
13.3 使用Flash ····· 236
13.3.1 使用Flash应用程序 ····· 236
13.3.2 构建AIR应用程序 ····· 237
小结 ····· 237
参考资料和更多信息 ····· 237

第14章 使用位置服务API ····· 238
14.1 使用全球定位服务（GPS） ····· 238
14.1.1 在你的应用程序中使用GPS功能 ····· 238
14.1.2 查找你的位置 ····· 239
14.1.3 定位你的模拟器 ····· 240
14.2 位置地理编码 ····· 241
14.3 在地图上标注位置 ····· 243
14.3.1 地图标注内容 ····· 243
14.3.2 地图标注视图 ····· 244
14.3.3 获得调试API Key ····· 245
14.3.4 拖曳地图视图 ····· 247
14.3.5 缩放地图视图 ····· 247
14.3.6 标记点 ····· 248
14.4 位置服务扩展应用 ····· 251
小结 ····· 252
参考资料和更多信息 ····· 252

第15章 使用Android多媒体API ····· 253
15.1 使用多媒体 ····· 253
15.2 使用静态图像 ····· 254
15.2.1 使用摄像头捕捉静态图像 ····· 254
15.2.2 配置摄像头模式 ····· 257
15.2.3 分享图像 ····· 258
15.2.4 将图像指定为壁纸 ····· 259
15.3 使用视频 ····· 259
15.3.1 录制视频 ····· 260
15.3.2 播放视频 ····· 261
15.4 使用音频 ····· 262
15.4.1 录制音频 ····· 262
15.4.2 播放音频 ····· 263
15.4.3 共享音频 ····· 264
15.4.4 搜索多媒体 ····· 264
15.4.5 使用铃音 ····· 265
小结 ····· 265
参考资料和更多信息 ····· 265

第16章 使用Android电话API ····· 266
16.1 使用电话功能 ····· 266
16.1.1 为获取手机状态信息取得许可权限 ····· 266
16.1.2 获取通话状态 ····· 267
16.1.3 请求获取服务信息 ····· 268
16.1.4 监控信号强度和数据连接速度 ····· 269
16.1.5 使用电话号码 ····· 269
16.2 使用短消息 ····· 270
16.2.1 获得发送和接收短消息的许可权限 ····· 270
16.2.2 发送短消息 ····· 270
16.2.3 接收短消息 ····· 272
16.3 拨打和接听电话 ····· 273
小结 ····· 275
参考资料和更多信息 ····· 275

第17章 通过OpenGL ES使用Android 3D图形 ····· 276
17.1 使用OpenGL ES ····· 276
17.1.1 在Android中使用OpenGL ES ····· 276
17.1.2 保证设备兼容性 ····· 277
17.2 在Android SDK中使用OpenGL ES API ····· 277
17.3 手工处理OpenGL ES任务 ····· 278

Download at http://www.pin5i.com/

17.3.1 创建SurfaceView ····· 278
17.3.2 启动OpenGL ES线程 ····· 279
17.3.3 初始化EGL ····· 281
17.3.4 初始化GL ····· 282
17.3.5 屏幕绘制 ····· 283
17.4 绘制3D物体 ····· 283
17.4.1 绘制顶点（Vertices） ····· 283
17.4.2 为顶点着色 ····· 284
17.4.3 绘制更多复杂物体 ····· 285
17.4.4 照亮你的场景 ····· 286
17.4.5 为对象赋予材质 ····· 288
17.5 同Android视图和事件进行交互 ····· 289
17.5.1 允许OpenGL线程向应用程序线程发送信息 ····· 289
17.5.2 允许应用程序线程向OpenGL线程发送信息 ····· 290
17.6 清理OpenGL ES ····· 291
17.7 使用GLSurfaceView ····· 292
17.8 使用OpenGL ES 2.0 ····· 294
17.8.1 为OpenGL ES 2.0配置应用程序 ····· 295
17.8.2 获取OpenGL ES 2.0平面 ····· 295
小结 ····· 298
参考资料和更多信息 ····· 298

第18章 使用Android NDK ····· 299
18.1 何时使用Android NDK ····· 299
18.2 安装Android NDK ····· 300
18.3 探索Android NDK ····· 300
18.4 创建自己的NDK工程 ····· 300
18.4.1 使用Java调用原生代码 ····· 301
18.4.2 处理参数和返回值 ····· 302
18.4.3 在原生代码中使用异常 ····· 303
18.5 提升图形处理性能 ····· 304
小结 ····· 305
参考资料和更多信息 ····· 305

第19章 使用Android可选硬件API ····· 306
19.1 与设备硬件进行交互 ····· 306
19.2 使用设备传感器 ····· 307
19.2.1 使用不同的传感器 ····· 307
19.2.2 获取对传感器的访问 ····· 307
19.2.3 读取传感器数据 ····· 308
19.2.4 校正传感器 ····· 309
19.2.5 确定设备的方向 ····· 309
19.2.6 指北 ····· 309
19.3 使用Wi-Fi ····· 309
19.4 使用蓝牙 ····· 311
19.4.1 检查可用的蓝牙硬件 ····· 312
19.4.2 开启蓝牙 ····· 312
19.4.3 查询配对设备 ····· 313
19.4.4 搜索设备 ····· 313
19.4.5 建立设备间连接 ····· 313
19.5 监视电池 ····· 314
小结 ····· 316
参考资料和更多信息 ····· 316

第五部分 更多Android应用程序设计原则

第20章 使用通知 ····· 318
20.1 向用户发出通知 ····· 318
20.2 使用状态栏发布通知 ····· 319
20.2.1 使用NotificationManager服务 ····· 319
20.2.2 创建一个带有图标的简单文本通知 ····· 319
20.2.3 使用通知队列 ····· 320
20.2.4 更新通知 ····· 321
20.2.5 清除通知 ····· 321
20.3 让手机震动 ····· 322
20.4 灯光闪烁 ····· 323
20.5 发出声响 ····· 324
20.6 自定义通知 ····· 324
20.7 设计有用的通知 ····· 326
小结 ····· 326
参考资料和更多信息 ····· 326

第21章 使用服务 ····· 327
21.1 何时使用服务 ····· 327

Download at http://www.pin5i.com/

21.2 理解服务的生命周期 328
21.3 创建服务 328
21.4 控制服务 332
21.5 实现远程接口 333
21.6 实现可包装（Parcelable）类 335
小结 337
参考资料和更多信息 337

第22章 扩展Android应用程序 338

22.1 增强你的应用程序 338
22.2 使用应用程序小插件 339
22.2.1 创建应用程序小插件 339
22.2.2 安装应用程序小插件 345
22.2.3 成为小插件“宿主（Host）” 345
22.3 使用动态壁纸（Live Wallpaper） 345
22.3.1 创建动态壁纸 346
22.3.2 安装动态壁纸 348
22.4 成为内容类型处理器（Content Type Handler） 349
22.5 确定Intent动作和MIME类型 350
22.5.1 实现Activity处理Intent 350
22.5.2 注册Intent过滤器 351
22.6 让应用程序的内容可搜索 352
22.6.1 在应用程序内启用搜索 353
22.6.2 启用全局搜索 358
22.7 使用动态文件夹 360
22.7.1 创建动态文件夹 360
22.7.2 安装动态文件夹 364
小结 365
参考资料和更多信息 365

第23章 用户账户管理与数据同步 366

23.1 使用账户管理器管理账户 366
23.2 使用备份服务 367
23.2.1 选择远程备份服务 368
23.2.2 实现备份代理（Agent） 368
23.2.3 备份和恢复应用程序数据 371
小结 372
参考资料和更多信息 373

第24章 处理高级用户输入 374

24.1 使用文本输入法 374
24.1.1 使用软键盘 374
24.1.2 使用文本预测和用户词典 376
24.2 探索辅助功能（Accessibility）框架 376
24.2.1 使用语音识别服务 377
24.2.2 使用语音合成（TTS）服务 378
24.3 使用手势 380
24.4 处理普通单点触控手势 381
24.4.1 处理普通多点触控手势 385
24.4.2 让手势更自然 388
24.5 使用轨迹球 388
24.6 处理屏幕方向改变 388
小结 390
参考资料和更多信息 390

第25章 面向不同设备配置和语言 391

25.1 最大化应用程序兼容性 391
25.2 为兼容性设计用户界面 392
25.2.1 支持特定屏幕类型 393
25.2.2 使用九格拉伸图像 393
25.2.3 使用Working Square原理 395
25.3 提供应用程序替代资源 396
25.3.1 使用替代资源标识符 396
25.3.2 提供用于不同屏幕方向的资源 399
25.3.3 在程序中使用替代资源 400
25.3.4 高效地组织应用程序资源 400
25.4 应用程序国际化 401
25.4.1 使用替代资源实现国际化 401
25.4.2 在程序中实现本地化支持 404
25.5 面向不同设备配置 405
25.5.1 支持硬件配置 405
25.5.2 面向不同Android SDK版本 405
小结 407
参考资料和更多信息 408

Download at http://www.pin5i.com/

第六部分 对外发布你的Android应用程序

第26章 移动软件开发过程 …… 410

26.1 移动开发过程概览 …… 410
26.2 选择一种软件开发方法 …… 410
26.2.1 理解“瀑布模型”的危险 …… 411
26.2.2 认识迭代式开发的价值 …… 411
26.3 整合应用程序需求 …… 411
26.3.1 确定工程需求 …… 411
26.3.2 为移动应用程序开发用例 …… 413
26.3.3 吸纳第三方需求 …… 413
26.3.4 管理设备数据库 …… 413
26.4 项目风险评估 …… 415
26.4.1 确定目标设备 …… 415
26.4.2 获取目标设备 …… 416
26.4.3 确定应用程序需求的可行性 …… 417
26.4.4 理解质量保障的风险 …… 417
26.5 编写基本项目文档 …… 418
26.6 使用配置管理系统 …… 419
26.7 设计移动应用程序 …… 420
26.7.1 认识移动设备的局限 …… 420
26.7.2 移动应用程序体系结构探索 …… 420
26.7.3 可扩展和维护设计 …… 421
26.7.4 应用程序交互设计 …… 422
26.8 开发移动应用程序 …… 422
26.9 测试移动应用程序 …… 422
26.10 发布移动应用程序 …… 423
26.11 支持和维护移动应用程序 …… 423
小结 …… 424
参考资料和更多信息 …… 424

第27章 设计和开发“防弹”Android应用程序 …… 425

27.1 “防弹”移动应用程序设计最佳实践 …… 425
27.1.1 满足移动用户的需求 …… 425
27.1.2 为移动设备设计用户界面 …… 426
27.1.3 设计稳定、灵敏的移动应用程序 …… 426
27.1.4 设计安全的移动应用程序 …… 427
27.1.5 在设计移动应用程序时考虑利益的最大化 …… 428
27.1.6 在Android应用程序设计时使用第三方标准 …… 428
27.1.7 设计便于维护和升级的移动应用程序 …… 429
27.1.8 在应用程序设计中使用Andriod工具 …… 430
27.1.9 避免设计时的愚蠢错误 …… 430
27.2 “防弹”移动应用程序开发最佳实践 …… 431
27.2.1 设计有效的移动开发过程 …… 431
27.2.2 提早和经常地测试应用程序可行性 …… 431
27.2.3 使用代码标准、代码检查和单元测试以提高代码质量 …… 431
27.2.4 处理某单一设备的缺陷 …… 433
27.2.5 在应用程序开发中使用Android工具 …… 434
27.2.6 避免开发时的愚蠢错误 …… 434
小结 …… 434
参考资料和更多信息 …… 434

第28章 测试Android应用程序 …… 435

28.1 设计一个应用程序漏洞跟踪系统 …… 435
28.2 管理测试环境 …… 436
28.3 尽量全面地进行测试 …… 438
28.4 在应用程序测试中使用Android工具 …… 442
28.5 避免测试时的愚蠢错误 …… 443
28.6 外包测试的责任 …… 443
小结 …… 443
参考资料和更多信息 …… 443

第29章 销售你的Android应用程序 …… 444

29.1 选择合适的发布模式 …… 444
29.2 为发布而打包你的应用程序 …… 444

Download at http://www.pin5i.com/

29.2.1 准备封装你的代码 …… 445
29.2.2 为你的应用程序打包并签名 … 446
29.2.3 测试软件包的发布版本 …… 448
29.2.4 认证你的Android应用程序 … 448
29.3 发布你的应用程序 …… 448
29.3.1 在Android Market上销售你的应用程序 …… 448
29.3.2 在自己的服务器上销售你的应用程序 …… 453
29.3.3 销售应用程序的其他途径 …… 453
29.3.4 保护你的知识产权 …… 454
29.3.5 让用户付费 …… 454
小结 …… 455
参考资料和更多信息 …… 455

第七部分 附 录

附录A Android模拟器快速入门 …… 458
A.1 模拟器的目的：虚拟现实 …… 458
A.2 使用Android虚拟设备（AVD） …… 459
A.2.1 使用Android SDK和AVD管理器 …… 460
A.2.2 创建AVD …… 460
A.3 使用指定AVD启动模拟器 …… 463
A.3.1 配置模拟器启动项 …… 463
A.3.2 启动模拟器运行应用程序 …… 463
A.3.3 从Android SDK和AVD管理器中启动模拟器 …… 464
A.4 配置模拟器的GPS位置信息 …… 465
A.5 在两个模拟器实例之间拨打电话 …… 466
A.6 在两个模拟器实例之间发送消息 …… 466
A.7 通过控制台与模拟器交互 …… 467
A.7.1 使用控制台模拟来电 …… 467
A.7.2 使用控制台模拟短消息 …… 468
A.7.3 使用控制台发送GPS坐标 …… 468
A.7.4 使用控制台监视网络状态 …… 469
A.7.5 使用控制台操作电源设置 …… 469
A.7.6 使用其他控制台命令 …… 470
A.8 享受模拟器 …… 470
A.9 认识模拟器的局限 …… 470

附录B Android DDMS快速入门 …… 471
B.1 独立或结合Eclipse使用DDMS …… 471
B.2 熟悉DDMS的关键特性 …… 472
B.3 使用进程 …… 472
B.3.1 向Android应用程序关联调试器 …… 473
B.3.2 监视Android应用程序的线程活动 …… 473
B.3.3 触发垃圾回收（GC） …… 474
B.3.4 监视堆活动 …… 474
B.3.5 监视内存分配 …… 474
B.3.6 终止Android进程 …… 475
B.4 使用文件浏览器 …… 475
B.4.1 浏览模拟器或设备上的文件系统 …… 475
B.4.2 从模拟器或设备上拷贝文件 … 476
B.4.3 向模拟器或设备拷贝文件 …… 476
B.4.4 删除模拟器或设备上的文件 … 476
B.5 使用模拟器控制 …… 476
B.5.1 模拟语音来电 …… 476
B.5.2 模拟短消息接收 …… 477
B.5.3 发送位置坐标 …… 477
B.6 使用应用程序日志 …… 477
B.7 在模拟器或目标设备上截屏 …… 478

附录C Android调试桥快速入门 …… 479
C.1 列表显示连接的设备和模拟器 …… 479
C.2 针对特定设备使用ADB命令 …… 479
C.3 启动和停止ADB服务进程 …… 480
C.3.1 停止ADB服务进程 …… 480
C.3.2 启动和检测ADB服务进程 …… 480
C.4 发布Shell命令 …… 480
C.4.1 发布单条Shell命令 …… 480
C.4.2 使用Shell会话 …… 481
C.4.3 使用Shell启动和终止模拟器 …… 481
C.5 拷贝文件 …… 481

Download at http://www.pin5i.com/

C.5.1 向设备或模拟器发送文件…… 481
C.5.2 从设备提取文件…… 481
C.6 安装和卸载Android应用程序…… 482
C.6.1 安装应用程序 …… 482
C.6.2 重装应用程序 …… 482
C.6.3 卸载应用程序 …… 482
C.7 使用LogCat记录日志…… 483
C.7.1 显示所有的日志信息…… 483
C.7.2 在日志数据中包含日期和时间信息 …… 483
C.7.3 过滤日志信息 …… 483
C.7.4 清除日志 …… 484
C.7.5 将日志输出重定向至一个文件 …… 484
C.7.6 访问次级日志 …… 485
C.8 控制备份服务…… 485
C.8.1 强制备份操作 …… 485
C.8.2 强制恢复操作 …… 486
C.8.3 将存档数据清空…… 486
C.9 生成Bug报告…… 486
C.10 使用Shell查看SQLite数据库 …… 486
C.10.1 在ADB Shell中进行应用程序压力测试 …… 486
C.10.2 通过Shell安装和使用自定义二进制文件 …… 489
C.11 其他ADB命令探索…… 490

附录D Eclipse技巧和诀窍 …… 491
D.1 构建你的Eclipse工作空间 …… 491
D.1.1 集成源代码管理服务 …… 491
D.1.2 调整视图中的标签面板 …… 491
D.1.3 最大化窗口 …… 492
D.1.4 最小化窗口 …… 492
D.1.5 显示并排窗口 …… 492
D.1.6 显示同一文件的两个部分 …… 492
D.1.7 关掉不需要的标签面板 …… 492
D.1.8 保持对窗口的控制 …… 492
D.1.9 定制日志过滤器 …… 493
D.2 编写Java代码 ……493
D.2.1 善用自动补全（Auto-Complete）……493
D.2.2 格式化代码……493
D.2.3 创建类……494
D.2.4 创建方法存根（stub）……494
D.2.5 组织导入……494
D.2.6 重命名……494
D.2.7 重构代码……494
D.2.8 整理代码……496
D.2.9 提供Javadoc风格的文件……496
D.2.10 解决诡异的构建错误……496

附录E SQLite快速入门 ……497
E.1 探索SQLite的常见任务 ……497
E.2 使用sqlite3命令行接口 ……497
E.2.1 运行ADB Shell……498
E.2.2 连接数据库 ……498
E.2.3 探索你的数据库 ……498
E.2.4 导入和导出数据库 ……499
E.2.5 在命令行中执行SQL命令……501
E.2.6 使用其他的sqlite3命令……502
E.2.7 认识SQLite的局限……502
E.3 通过实例学习：学生成绩数据库……503
E.3.1 设计学生成绩数据库的模式…503
E.3.2 结合AUTOINCREMENT创建简单表 ……503
E.3.3 向表中插入数据 ……504
E.3.4 使用SELECT查询表数据……504
E.3.5 创建具有外键和复合主键的表 ……505
E.3.6 修改和更新表数据 ……506
E.3.7 使用JOIN进行多表查询……506
E.3.8 使用计算列 ……507
E.3.9 对计算列使用子查询 ……508
E.3.10 删除表 ……508

Download at http://www.pin5i.com/

第一部分

Android 概览

第 1 章 Android 简介

第 2 章 搭建你的 Android 开发环境

第 3 章 编写第一个 Android 应用程序

Download at http://www.pin5i.com/

第 1 章

Android 简介

移动开发领域正处在一个转折点：移动用户的需求日益增长，他们需要具有更多功能、更为个性化的手机；移动运营商意图为用户提供易于管理、获利更多的增值服务；开发人员需要获取更多自由，以开发出受用户欢迎的、功能强大的应用程序，同时避免在成功的道路上遭遇层层障碍；而手机制造商则呼唤一个稳定、安全和廉价的平台，以充分发挥硬件的性能。到目前为止，仅有一个移动平台可以满足所有的需求，那就是 Android，它在移动开发领域异军突起。具有划时代意义的开放移动平台 Android 横空出世，顺应潮流，定位准确而恰当，能够很好地满足市场的各方面需求。

本章将告诉读者 Android 是什么，为什么开发 Android，它是如何成长起来的，以及它在移动领域市场中所处的位置。

1.1 移动软件开发简史

要弄清楚为什么 Android 如此令人着迷，我们需要从移动通信的进化历程讲起，并逐步了解 Android 同其他与其竞争的平台的不同之处。

1.1.1 遥想当年

还记得电话仅仅是电话的年代么？那时我们所能依赖的仅仅是一条固定的电话线。当电话铃一响，我们就得匆匆奔去，而非简单地从口袋里掏出什么来；当我们在喧闹的人群中和同伴挤散时，只能踮起脚尖四处寻找，结果往往是苦苦等上好几个小时；当我们忘记了购物清单（见图 1.1）上的物品时，只能寻找公用电话或者无奈地开车回家。

这些日子早已一去不复返了。现在，这种基本问题的解决方式已经变成“单键拨号”或是一条类似于“哪？”、“20？”或“牛奶和？”的简单的短信。

手机在让我们感到安全的同时保持与他人的联系。当我们四处闲逛时，手机已不仅仅是用来联系朋友、家人和同事的工具，它还能够告诉我

图 1.1 手机已经成为了不可或缺的购物工具

Download at http://www.pin5i.com/

们到哪儿去、做什么以及怎么做。如今，即使再家常的事情也可能需要手机来解决。

让我们看看下面这则略带夸张但绝非虚构的故事：

一次，一个温暖的夏日之夜，正当我在 New Hampshire 的乡村新房子里专心致志地准备商务晚餐时，一只蝙蝠突然飞过头顶，一头扎了下来，把我吓得半死。

我本能地低头躲开，一边掏出手机给我在外面的丈夫发短信："房里有蝙蝠!"，我丈夫并没有立即回复（那时我气得想和他离婚），于是我打电话给我父亲求助，问他怎么把蝙蝠赶走。

他居然大声地笑了出来，什么都没有说。

我有些恼怒了，用手机拍了张那只蝙蝠的照片发给我丈夫，让他有负罪感，同时也发到了我的博客上，以告知全世界我在这荒郊野外的可怕遭遇。

最后，我用 Google 搜索了"驱赶蝙蝠"，然后找到了一篇写给类似我这样遭遇的人的"说明书"，我照着一步步做了。不仅如此，我还了解到 8 月下旬往往是小蝙蝠第一次离开栖息地学习飞翔的时候。这时我才意识到逮在手里的是一只蝙蝠宝宝，然后我冷静下来，用一只扫帚把它送出了我的房子。

问题总算解决了，完全是在我那部值得信赖的老 LG VX9800 手机的帮助下。

我想说明什么？手机几乎可以解决所有的问题，我们现在任何事情都依赖于它。

你有没有注意到在这个故事的整个过程中，我使用了五六个不同的移动应用程序。每一个应用程序由不同的公司开发，拥有不同的用户界面。有些界面设计得相当美观而适用，而有些则稍微逊色一点。有些应用程序是我花钱买的，而有些则原本就在我的手机里。

作为一个用户，这段经历的确有价值，但也不是特别具有启发意义。但作为一个移动开发人员，我却希望能够开发一个更强大的应用程序，以更为准确地处理我所遭遇的问题，甚至其他更多我还没有遇见的。如果愿意的话，我想制作一个更好的捕捉蝙蝠的装置。

在 Android 诞生之前，移动开发人员在编写应用程序时往往面临许多困境。要轻车熟路地开发一个唯一的、具有竞争力的、集各种功能为一体的更为强大的应用程序，甚至可以兼顾着处理类似通话和短消息这样的日常事务，通常是不现实的。

你也许不相信，不过，当了解了移动软件的发展历程后你就明白其中的道理了。

1.1.2 "砖头"

摩托罗拉 DynaTAC 8000X 是市面上销售的第一款手机。1983 年上市时，它是一个 13 英寸长、1.75 英寸宽、3.5 英寸厚、重达 2.5 磅的大块头，但却仅仅能够支撑半小时多一点的通话。当时它的零售价高达 3995 美元。

我们称它为"砖头"，这一绰号不是它的专利，而一并被赋予了早期许多让我们又爱又恨的移动电话。具有砖头大的尺寸，电池却连一次谈话都维持不了，这些早期的手机大多只出现在一些特殊人群的手里，比如旅行中的商业头领、安保人员，抑或富豪。第一代手机实在太贵了，光基本的服务费就足以让普通工薪阶层破产，更别说拿着它到处漫游。

早期的移动电话并不具有什么突出的功能（虽然如此，但即使是图 1.2 所示的摩托罗拉 DynaTAC 这样的手机，也拥有我们所熟悉的 Send、End 和 Clr 等按键）。这些早期手机的功

Download at http://www.pin5i.com/

能仅仅局限于拨打电话。如果你运气好的话，也许会有一个类似于“通信簿”这样的应用程序可供使用。

第一代移动电话由不同的制造商设计和生产，竞争相当激烈，他们都严密地保护着各自的商业机密。制造商们不愿意透露手机的内部工作细节，以至于都闭门造车，开发各自的手机软件。而作为一个开发人员，如果你被这个小圈子排除在外，那么就丝毫没有机会为这些手机编写应用软件。

图 1.2 市场销售的第一部移动电话：摩托罗拉 DynaTAC

在这一时期，手机制造商们彼此之间正进行着浪费时间的“冷战”。诺基亚在其某些黑白屏手机上加入了 20 世纪 70 年代著名的“贪食蛇”游戏，并因此而大受欢迎。其他制造商也纷纷效仿，在手机中加入了诸如乒乓球（Pong）、井字游戏（Tic-Tac-Toe）和俄罗斯方块这样的游戏。

虽然这些早期的手机存在不少缺陷，但它们的确完成了一件重要的事情，它们改变了人们传统观念中对于通信的认识。随着移动电话价格的下降，电池续航能力的增强，信号覆盖区域的拓宽，越来越多的人们开始拥有这些“掌中宝”。很快地，移动电话就不仅仅是新奇的小玩意儿了。

消费者们开始追求更多的功能和更多的游戏。但是有一个问题，手机制造商们却没有足够的动机与资源将用户需要的所有应用程序都放到手机里。他们需要某种途径来建立一个门户，以供用户娱乐和获得信息服务，而不是由手机设备本身提供。

如此，有什么途径能比因特网更能提供这些服务呢？

1.1.3 无线应用协议（WAP）

事实上，使用手机直接访问因特网并没有收到很好的效果。

在那个时候，专业的网站色彩斑斓，填充着各种文字、图片和多媒体资源。这些网站依赖于 JavaScript、Flash 和其他技术，往往具有 800 像素×600 像素甚至更高的分辨率，以增强用户体验。

1996 年，第一部翻盖手机摩托罗拉 StarTAC 发布，它仅仅具有一行只能显示 10 个数码的分行显示屏（后来的型号才具有了点阵显示）。同时，诺基亚发布了一款最早的滑盖手机 8110——被戏称为“黑客帝国手机”，因为它在电影里被大量地使用。8110 也仅仅能够显示 4 行文本、每行 13 个字符。图 1.3 显示了手机常见的一些样式。

由于这些手机只具有邮票般大的低分辨率屏幕，而且存储和数据处理能力均有限，所以它们无法进行密集的数据处理，而这往往是传统网络浏览器所必需的。除此之外，对于用户而言，满足数据传输所需的带宽也显得过于昂贵。

无线应用协议（Wireless Application Protocol，WAP）应运而生，以解决这一困境。简单地说，WAP 仅仅是因特网核心协议之一的 HTTP 缩略版。不同于传统的网络浏览器，WAP 浏览器被设计为可以在手机内存和带宽的限制下运行。第三方 WAP 网站编写了基于一种称为无线标记语言（Wireless Markup Language，WML）的网页，这些网页随之被显示在手机的 WAP 浏览器中。用户像浏览正常网页一样进行导航与跳转，只不过这些页面被设计得非常简单。

Download at http://www.pin5i.com/

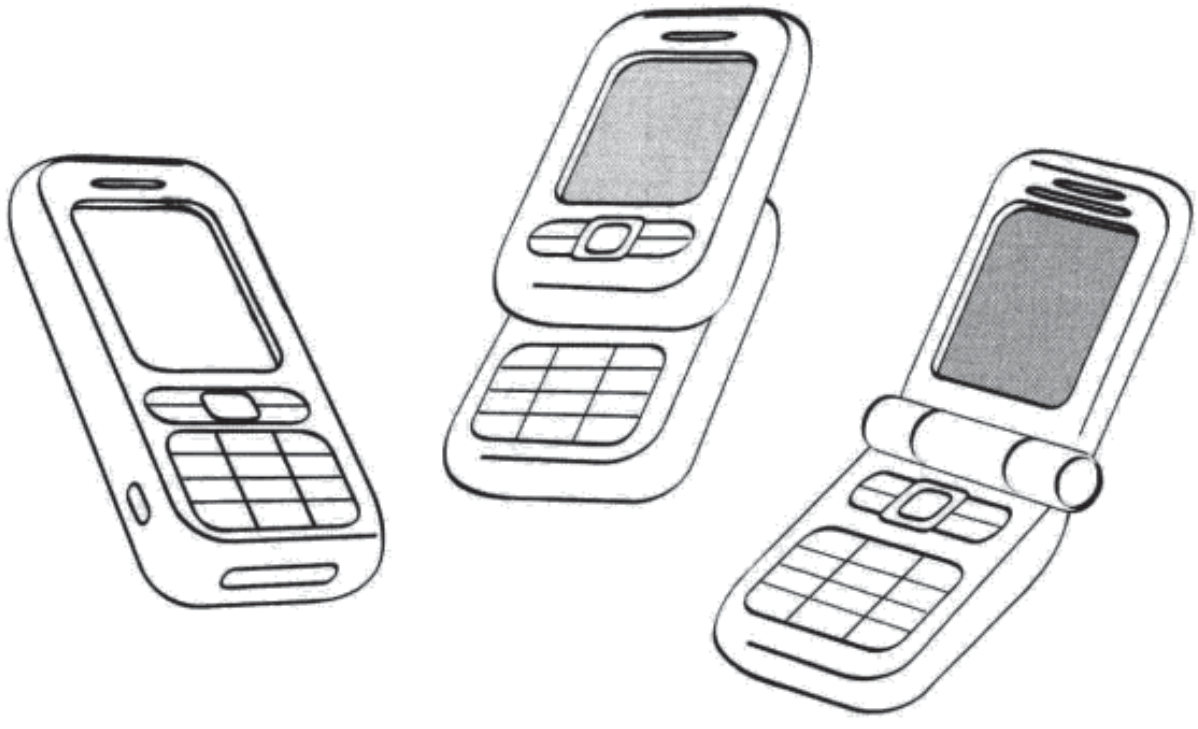


图 1.3 各种不同的手机样式：直板、滑盖和翻盖

WAP 解决方案给手机制造商带来了福音。他们的压力瞬间消失了，他们只需要编写一个 WAP 浏览器，然后把它移植到手机上，剩下的工作就依赖于其他开发人员为用户提供所需的内容了。

WAP 解决方案也大受移动运营商的欢迎。他们只需要提供一个定制的 WAP 入口，然后把用户引导到所需的内容处，最后就可以收取由数据流量产生的费用了，而这一费用往往非常可观。

开发人员和内容提供商也没有放弃这个机会。开发人员第一次有了针对用户所需内容进行开发的机会，有些人进行了尝试，但所取得的成功却非常有限。

大部分早期的 WAP 站点都是一些有名气的网站的延伸，诸如 CNN.com 和 ESPN.com 等，它们在寻找新的道路以谋求读者群的扩展。一夜之间手机用户都开始使用手机来浏览新闻、阅读股票分析以及了解比赛得分。

但利用 WAP 获取利润还是一件很困难的事情，也没有任何一种内建的付费机制。一些广受欢迎的用于盈利的简单 WAP 应用程序崭露头角，其中之一可以提供一个简单的壁纸和铃音列表，并第一次让用户可以对其手机进行个性化的设置。例如，用户访问这一站点以获取所需的资源，他们需要在一个简单的表单中填入自己的手机号码和型号，然后由内容提供商向用户发送一个兼容的图片或声音文件。付款和验证则必须通过一些需要额外付费的方式，如短信、增强消息业务（Enhanced Messaging Service，EMS）、彩信业务（Multimedia Messaging Service，MMS）和 WAP Push 等。

WAP 浏览器（特别是在早些时候）运行缓慢而且不稳定，程序容易崩溃。使用数字键盘输入冗长的 URL 是一件繁重的工作；在 WAP 页面中导航也不是一件轻松的事情。大多数 WAP 站点对不同手机均使用同样的代码，而且没有针对某一特定型号手机的任何使用说明。网站不会因为用户使用的是一个大彩屏手机，或是一个只有邮票大屏幕的黑白屏手机而有所变化；程序开发人员也无法调整程序以获得更好的用户体验，结果则差强人意，没有办法引人入胜。

内容提供商却往往不干预 WAP 网站，仅专注于在电视或杂志上公布自己的短信代码。在这样的情况下，用户使用付费短信索取壁纸或手机铃音，而内容提供商则热衷于把它们

Download at http://www.pin5i.com/

发回给用户。移动运营商渐渐爱上了这样的方式，因为他们可以从中获取很大一部分通信费用。

WAP 陷入了一种缺乏商业前景的境地。在某些市场，如日本，它日渐繁荣；而在另外一些市场，比如美国，却难以有所作为。对于网络冲浪而言，手机屏幕的尺寸显得过小。一次只能阅读一小段语句，然后等待数秒以下载下一段，这极大地影响了用户体验，尤其是每一秒钟的下载都会向用户收取费用。于是批评家们开始称 WAP 为“边等边付钱（Wait and Pay）”。

终于，提供 WAP 门户（启动 WAP 浏览器时载入的默认主页）的移动运营商开始对可访问的 WAP 站点进行限制。这一门户使得运营商可以限制用户所能浏览网站的数量，而把用户都引向运营商所推荐的内容提供商那里，同时把一些竞争者拒之门外。这一道围墙大大挫伤了第三方开发人员编写应用程序的积极性，何况他们本就面临着盈利的困境。

1.1.4 专用移动平台

用户需求的逐渐增大是毫无疑问的，他们永远都期待着更多。

在 WAP 下编写诸如电脑游戏这样包含密集图像的强大应用程序几乎是不可能的。通常 18～25 岁的人群最有可能通过改变壁纸和铃音来个性化自己的手机，他们一边看着他们的掌上游戏机一边想着能有一个兼具游戏机和手机或者音乐播放器和手机功能的东西。他们坚持认为，像任天堂 Game Boy 这样只具有 5 个按钮而可以让他们玩上好几个小时的东西，为什么不能放到手机上呢？另外一些人则盯上了他们的数码相机、Palm 或黑莓手机、iPod 甚至笔记本电脑，提出了同样的问题。市场似乎正站在设备融合的边缘摇摇晃晃。

如今内存变得越来越便宜，电池性能越来越好，PDA 和其他嵌入式设备开始能够运行各种版本的流行操作系统，如 Linux 和 Windows。传统的桌面应用程序开发人员突然进入了嵌入式设备的市场领域，尤其是从事他们所熟悉的智能手机技术，如 Windows Mobile。

同时手机制造商意识到如果他们想要继续销售传统的手机设备，就需要改变他们与手机设计有关的贸易保护主义政策，而且公开他们的内部架构，至少，在一定程度上公开。

于是各种类型的专用平台应运而生，程序开发人员依然积极地为他们开发应用程序。一些智能手机运行在 Palm OS（现在为 Garnet OS）或 RIM Blackberry OS 下；Sun Microsystems 公司推出了广受欢迎的 Java 平台和 J2ME（也就是 Java Micro Edition）；芯片制造商高通研发并授权了他的 Binary Runtime Environment for Wireless（简称 BREW）。其他平台，如 Symbian OS，则由一些手机制造商开发，如诺基亚、索尼爱立信、摩托罗拉和三星等；苹果 iPhone OS（OS X iPhone）在 2008 年也加入了这一行列。图 1.4 展示了几款手机，它们各自拥有不同的开发平台。

许多这些平台都有与之关联的开发程序。这些程序形成了某种“门槛”，使得开发人员这一群体维持着比较小的规模，并且共同遵守着什么能做和什么不能做的约定。而且这些程序往往是必需的，开发人员必须为此而承担一定的费用。

每一个平台都有一定的优势，也存在一些缺点。当然，开发人员一直热衷于争论到底哪个平台是最好的（通常来讲一般是自己正从事开发的平台）。

事实上没有一个平台获得绝对的胜利。一些平台适用于商业化的应用，赚取利润（如果你的公司拥有品牌支持）。其他一些应用程序则更加开放而适于发烧友或开发垂直市场应用程序（Vertical Market Application）。没有一个平台能够适用于所有的应用，结果是，移动电话

Download at http://www.pin5i.com/

像一个馅饼一样逐渐被分割成好几块，每一个平台占有着自己的那一份。

图 1.4 具有不同移动设备平台的手机

对于制造商和移动运营商而言，手机设备生产线变得复杂而快速。在不同地区、不同人群中，手机平台占据着不同的市场。不同于只有一种平台可选，制造商和运营商不得不销售基于各种不同平台的手机，以适应市场竞争。我们甚至可以发现一些手机支持多种平台，例如，Symbian 手机通常也支持 J2ME。

移动开发人员这一群体也随着市场的分化而被细分。我们现在已几乎不可能描绘出市场变化每一丝轨迹。程序开发这一专业已形成了自己的定位。平台开发的需求也极大程度地分化，移动软件开发人员同具有明显区别的不同开发环境、不同开发工具和不同程序语言打交道。在不同平台间进行移植的成本也变得很高，而且道路曲折而艰难。记录手机配置和测试需求、为程序签名和授权、运营商关系和应用程序市场成为了随之而来的附属产业。

对于需要一个移动应用程序的 ACME 公司来讲，这无疑是一场噩梦。他们开发的应用程序应该基于 J2ME、BREW、iPhone 还是 Windows Mobile？每个人都拥有不同的手机，ACME 不得不从中选择一个，或无奈而痛苦地全部选择。某些平台允许使用免费的应用程序，而其他的则不然。这使得开发垂直市场应用程序的机会有限而且代价高昂。

最终结果是，许多极好的应用程序没有被预期的用户所使用，甚至许多绝佳的点子根本就未能得以实现。

1.2 开放手机联盟（OHA）

下面我们把目光聚焦于搜索引擎广告巨头：Google。作为一个家喻户晓的名字，Google 对扩大其品牌知名度同时在无线市场领域发布其产品显示出了浓厚的兴趣。Google 公司的商业模式使得其在因特网市场上取得了令人瞩目的成绩，并出乎意料地获得了极大的成功，何况，从技术上讲，无线领域同因特网并没有太大的区别。

Download at http://www.pin5i.com/

1.2.1 无线 Google

Google 公司突然进军移动领域使其面临了所有你能想象到的问题。因特网用户所能享受到的自由权利在移动用户那里却得不到分享。因特网用户可以自主地选择电脑品牌、操作系统、因特网服务提供商和网络浏览器。

几乎所有的 Google 服务都是免费的，它们依靠广告支撑起来。Google Labs 套装里提供的许多应用程序都可以直接与现存手机上的应用软件相媲美。这些应用程序小到简单的日历或计算器，大到可以使用 Google Map 进行地图导航或从 News Alerts 上下载定制的新闻的程序，更别说同其他组织或公司合作的产物，如 Bloger、YouTube 等。

当这种“进军”没有满足用户需求时，Google 决定换一个角度进攻，即翻新整个建立在无线应用开发上的系统，旨在为用户和开发人员提供一个更为开放的环境——因特网模式。因特网模式允许用户在免费软件、共享软件和付费软件中自由选择，而这恰恰孕育了服务提供商的自由市场竞争。

1.2.2 成立开放手机联盟

在这种“用户至上，设计自主”宗旨的引导下，Google 引领了一场将现有保守的无线市场转变为“自由世界”的运动，在这一“自由世界”中，手机用户能够自主选择载体，并拥有对应用和服务不受限制的访问权。应用自身已获得的大量资源，Google 往前迈进了一大步：从关注美国联邦通信委员会（FCC）针对无线通信的一系列政策，到分析手机制造商、应用程序开发人员和手机用户的不同需求，Google 开始仔细审视无线领域的根基。

下一步，Google 联合了“无线社区”中其他和他持有相同意见的成员，并抛出了以下问题：如何才能打造一个更好的手机？

为了解答这个问题，开放手机联盟（Open Handest Alliance，OHA）于 2007 年 11 月成立了。开放手机联盟是一个由世界上最大、最成功的几家从事移动产业的公司组成的商业联盟。它的成员包括芯片制造商、手机制造商、软件开发人员和服务提供商。整个手机市场供应链都被涵盖其中。

Andy Rubin 被誉为 Android 平台之父。他的公司——Android 公司（Android Inc.）于 2005 年被 Google 收购。包括 Google 在内的 OHA 成员们鼎力协作，在 Android 公司技术的基础上，开始着手开发一个非专用的、开放标准的平台，旨在缓解前面提到的若干问题对移动社区发展的阻碍。这就是 Android 项目。直到今天，大多数的 Android 平台开发都仍然由 Rubin 所带领的团队在 Google 完成，他担任 Google 的技术副总裁（VP of Engineering）并负责绘制 Android 平台的发展蓝图。

Google 对 Android 项目投入巨大，以至于到底是谁对 Android 平台负责（OHA 还是 Google）这一问题已经模糊不清了。Google 主持着 Android 开源项目并且提供在线 Android 文档、工具、论坛，以及为开发人员提供的软件开发包（SDK）。每一个有关 Android 的重大新闻都源于 Google。Google 还举办了若干次会议和 Android 开发人员挑战赛（ADC）。这个挑战赛激励开发人员编写“杀手”级的 Android 应用程序，其奖金总额高达 1000 万美元。挑战赛的赢家和他们的应用程序作品都可以在 Android 网站上找到。

Download at http://www.pin5i.com/

1.2.3 制造商：设计 Android 手机

超过一半的 OHA 成员是手机制造商，如三星、摩托罗拉、HTC 和 LG 等，还有许多半导体公司，如英特尔、德州仪器、NVIDIA 和高通等。这些公司帮助设计了第一代 Android 手机。

最早推出的 Android 手机 T-Mobil G1 由手机制造商 HTC 设计开发，并由 T-Mobile 公司提供配套服务，于 2008 年 10 月正式发布。其他许多 Android 手机于 2009 年底至 2010 年初问世。该平台较快地显现出了强劲的势头。每一个新的 Android 设备都比之前一年的更为强大而令人激动。在接下来的 18 个月中，60 种不同的 Android 手机（由 21 个不同的厂家生产）应运而生，它们由 59 个运营商在遍布世界的 48 个国家中推广。2010 年 6 月，在一款备受期待的 Android 手机的发布会上，Google 宣布每天有超过 16 万（也就是一年将近 6 千万）部 Android 手机被激活。制造商和运营商的广泛支持所带来的优势似乎在这一点上得到了集中的体现。

截至目前，Android 平台被公认为是成功的。它撼动了整个移动市场，面对包括苹果 iPhone、RIM 黑莓和 Windows Mobile 在内的竞争对手，它稳健地成长，逐步赢得了属于自己的市场地盘。截至 2010 年夏季的数据显示，处于领先地位的黑莓在整个智能手机的市场份额中占有 31%，处于下滑趋势。紧跟其后的是苹果 iPhone，占有 28%。Android 虽然仅占有 19%，但其发展势头却异常迅猛，据某些消息来源显示，它是市面上最为畅销的智能手机平台。而微软 Windows Mobile 的市场份额已经下降至几个百分点，落后于 Android。

1.2.4 移动运营商：传播 Android 体验

在你拥有了一部手机之后，当然会希望用它来和外界联系。南美、中美、北美、欧洲、亚洲、印度、澳大利亚、非洲和中东的许多运营商都加入了 OHA，以保证 Android 应用程序能够顺畅地在世界范围内流通。拥有数亿用户的电信业巨头中国移动通信（China Mobile）是 OHA 的始创成员。

Android 的成功在很大程度上也是因为这样的事实：许多 Android 手机不具有传统智能手机的“价格标签”，不少都由运营商提供免费激活。其竞争对手，比如苹果 iPhone 就不可能拥有这样的“待遇”。任何一个人都能够买得起一个具有完整功能的手机，这在以前是不太可能的。我已经记不清到底有多少次一名服务生、一位酒店夜班经理或者是一位杂货店的结账员告诉我，他们新买了一部 Android 手机，并且这部手机改变了他们的生活。这一现象足以反映处于劣势的 Android 为何具有如此强劲的上升势头。

在美国，Android 平台受到了来自运营商的许多帮助，例如 Verizon 公司，它为第一部 Droid 手机启动了一个 1 亿美元的项目。许多其他类似 Droid 这样的手机也由其他一些运营商所推动。Sprint 公司发布了 Evo 4G（美国第一部 4G 手机），并且记录了一天的销售情况（http://j.mp/cNhb4b）。

1.2.5 内容提供商：开发 Android 应用程序

当用户拥有一部 Android 手机，毫无疑问，他们将会很需要一些“杀手”级的应用程序。

在这方面，Google 领先群雄，它开发了许多 Android 应用程序，比如邮件客户端、Web 浏览器，都是 Android 平台上的核心功能。一些 OHA 成员，如 eBay，也在着手将 Android

Download at http://www.pin5i.com/

应用程序和它的网上拍卖结合起来。

第一届 Android 开发者挑战赛（ADC）收到了 1788 件提交上来的作品，而第二届 ADC 则由具有专业眼光的 26000 名 Android 用户投票选出最终的 200 个应用程序，所有的这些应用程序都是新开发的游戏、小工具和大量的定位服务（Location-Based Services，LBS）软件。我们还看到了人道主义、社交网络以及一些资源聚合类（mash-up）应用程序。其中许多应用程序都在 Android Market（Google 为 Android 打造的软件销售和传播中心）上发布了。现在，虽然挑战赛已经告一段落了，但比赛成果仍令人印象深刻。

对于那些一开始就使用 Android 平台的人而言，新手机设备的发布速度始终太慢。T-Mobile G1 是市面上第一款商业的 Android 设备，但它有一款为开发人员而预先发布的手机。而之后的 Android 手机则具有了更多的令人印象深刻的硬件设备，它们使得开发人员能够潜心钻研，开发更为优秀的应用程序。

截至 2010 年 10 月，有超过 80000 个的应用程序可以在 Android Market 中获得，而这一数字还在急剧增长。而这仅仅是 Android Market 一家的数据，还没有将个人应用程序销售和在其他市场上的销售情况纳入统计。另外，还没有考虑，从 Android 2.2 开始，Flash 应用程序也可以运行在 Android 手机之上。这为 Android 用户提供了更为丰富的应用程序选择，也给 Android 开发人员带来的更多的机会。

目前，有超过 18 万的 Android 开发人员正致力于编写有趣而令人激动的应用程序。等到你读完这本书之后，这一数字将因为你而加 1。

1.2.6 充分发挥 Android 的优势

开放的 Android 平台已经被大量的移动开发人员群体所接受，其规模远远超过 OHA 成员。

随着 Android 手机和应用程序变得越来越容易获得，尤其考虑到和现有专用移动平台相比，Android 的成本较低，许多其他手机运营商以及手机设备制造商也抓住这一机会加入到了销售 Android 手机的行列中来。Android 平台的开放标准减少了运营商在获取许可证和版权上的成本，而如今我们正看到一些现有的平台，如 RIM、Windows Mobile 和苹果 iPhone 从专用平台转变为开放式手机。市场之门已经大大敞开，新的用户群体正在加入智能手机用户这一庞大的行列。而 Android 已“全副武装”，最大限度地满足他们的需求。

1.3 Android 平台的不同之处

Android 被誉为“第一个完整、开放而免费的移动平台”。

- **完整**：设计人员在开发 Android 平台时综合了各种考虑。他们从一个安全的操作系统出发，在上层构建了一个健壮的应用程序架构，从而提供了丰富的应用程序开发机会。
- **开放**：Android 平台建立在开放源代码的基础上。开发人员在开发应用程序时获得了前所未有的访问手机设备的权限。
- **免费**：开发 Android 应用程序是免费的。在这个平台上开发应用程序不需要任何许可证，不需要承担任何版权费用，也无需缴纳会员费、测试费，以及为程序签名或认证的费用。Android 应用程序可以通过各种不同的途径发布和赚取回报。

Download at http://www.pin5i.com/

1.3.1 Android：下一代平台

虽然 Android 所具有的许多新颖的特性在目前存在的移动平台中还不具有，但它的设计者们正努力促使许多被事实所证明有效的方法在无线通信领域中得以应用。诚然，其中的许多特性已在现有的专用平台中有所体现，但 Android 免费而开放地将它们融合了起来，虽然某些在这些平台上应用时所发现的缺陷正在处理与解决中。

Android 的吉祥物为一个绿色的小机器人，如图 1.5 所示。你会发现这个小男孩（或者女孩？）经常出现在一些与 Android 相关的材料中。

图 1.5 Android 吉祥物

Android 是新一代移动平台中的开拓者，使得这一平台上的开发人员同其竞争者有着明显的区别。Android 的设计人员仔细研究了现有平台各自的优缺点，并集合了这些平台最成功的特性。同时，Android 的设计人员避免了其他平台在过去所犯的错误。

自从 Android 1.0 SDK 发布以来，Android 平台的开发就以飞快的速度进行着。在相当一段时间里，每两个月就会有一款新的 Android SDK 推出！而每一个 Android SDK 都拥有自己的项目名称，这在技术部门是司空见惯的。对于 Android 而言，其 SDK 的命名依照的是各种甜点打头字母的顺序（如图 1.6 所示）。最新版 Android 的代号为姜饼（Gingerbread）。

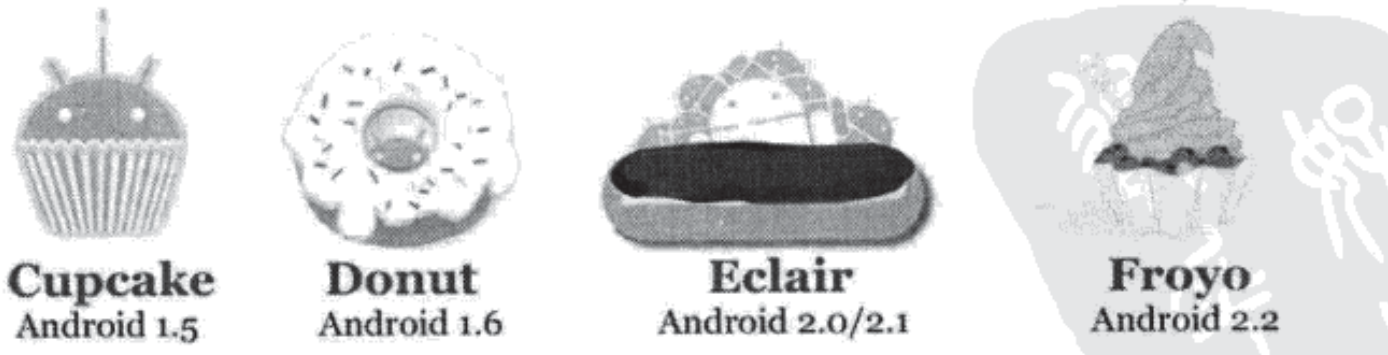


图 1.6 一些 Android SDK 和它们的代号

1.3.2 免费开放源代码

Android 是一个完全开放源代码的平台。无论是程序开发人员还是手机制造商都不需要为在这一平台上进行开发而承担版权或许可证费用。

Download at http://www.pin5i.com/

Android 所依赖的基础操作系统在 GNU General Public License Version 2（GPLv2）下获得了授权，这是一个强有力的“免费版权”的授权，以至于任何第三方的改进也必须继续遵守开放源代码的协议约定。Android 框架在 Apache Software License（ASL/Apache2）下发布，这一协议允许同时发布开源和非开源的软件版本。商业开发人员或机构（特别是手机制造商）可以选择在非开源的情况下对 Android 平台进行改进增强。而程序开发人员可以编写诸如针对某一特定手机设备的增强程序而获取利益，同时也可以在各种他们希望的协议下重新发布应用程序。

Android 应用程序开发人员可以在任何他们希望的协议模式下发布他们的应用程序。开发人员可以编写开源的免费软件，也可以开发需要授权许可的传统软件以获取一定的报酬，或者开发介乎两者之间的应用程序。

1.3.3 熟悉和廉价的开发工具

不同于某些专用平台需要开发人员支付注册费用、评估论证（vetting）和昂贵的编译器，开发 Android 应用程序无需任何先期支出。

（1）可免费获取的软件开发包（Software Development Kit，SDK）

Android SDK 和其他工具都可以免费获得。在接受和同意 Android 软件开发包协议条款后，程序开发人员可以从 Android 网站上下载 Android SDK。

（2）熟悉的语言、熟悉的开发环境

当考虑集成开发环境（Integrated Development Environments，IDE）时，开发人员往往拥有若干选择。许多开发人员选择流行而且可免费获取的 Eclipse IDE 来设计和开发 Android 应用程序。Eclipse 是用于 Android 开发的最受欢迎的 IDE，除此之外，使用 Eclipse 下的 Android 插件可以进一步地减轻 Android 开发负担。

Android 应用程序可以在以下操作系统上进行开发：

① Windows XP（32 位）或 Windows Vista（32 位或 64 位）；

② Mac OS X 10.5.8 或更高版本（仅限 x86）；

③ Linux（在 Linux Ubuntu 8.04 LTS，Hardy Heron 下测试通过）。

1.3.4 合理的开发人员学习曲线

Android 应用程序使用广受赞誉的 Java 程序设计语言编写。

Android 应用程序架构包含传统的程序设计概念（constructs），如在移动应用程序中普遍使用的进程与线程以及为对象封装而特别设计的数据结构。程序开发人员可以使用熟悉的类库，如 `java.net` 和 `java.text`。另外，完成诸如图形和数据库管理等任务的专业库也由明确而清晰的开放标准加以实现，例如 OpenGL Embedded Systems（OpenGL ES）、SQLite。

1.3.5 强大的应用程序开发支持

在过去，手机制造商往往与他们信赖的第三方软件开发机构建立某种特殊的关系（OEM/ODM 关系）。这些软件开发人员中的精英们编写诸如消息收发程序、Web 浏览器等本地（native）应用程序，之后将其移植到手机上作为一部分核心功能。为了设计这些应用程序，生产厂商必须授予开发人员足够的权限，以访问手机内部资源，同时也需要给予他们有关手

Download at http://www.pin5i.com/

机设备内部软件架构和固件的足够资料。

在 Android 平台下，本地应用程序同第三方应用程序之间没有区别，这样使得应用程序开发人员之间保持着良性的竞争。所有的 Android 应用程序均使用同样的库。Android 应用程序前所未有地拥有对底层硬件的访问权限，使得开发人员可以编写功能更为强大的应用程序。应用程序可以被扩展或完全替换掉。例如，Android 开发人员现在可以自由地设计开发针对某种 E-mail 服务器（如 Microsoft Exchange、Lotus Notes）的客户端应用程序。

1.3.6 丰富、安全的应用集成

如果回顾前面我与你分享的那个有关蝙蝠的故事，你会发现在整个过程中我使用了许多种手机应用程序：文字消息、拨号、照相、E-mail、图片消息和网络浏览器。每一个都是运行在手机上的独立应用程序，某些是内置的，而某些则是通过付费获得的。每一个程序都有其自身唯一的用户界面，但没有一个应用程序是真正集各种功能于一体的。

Android 则不会如此。Android 平台最具革命性也最令人着迷的特性之一则在于设计精妙的应用程序整合。如果你愿意的话，Android 可以提供所有供你制作一个更好的“蝙蝠捕捉器”所需的工具，它允许开发人员编写能够无缝整合诸如 Web 浏览器、地图软件、联系簿和消息应用程序核心功能的应用软件。应用程序也可以充当内容提供者以安全地在应用程序之间进行数据共享。

某些平台，如 Symbian，始终遭受着恶意软件的困扰。Android 强有力的应用程序安全模型则可以帮助用户和系统避免恶意软件的侵袭。

1.3.7 无障碍发布

Android 应用程序没有昂贵而耗时的测试和认证过程，而这一过程往往是其他平台所需要的，如 BREW、Symbian。

1.3.8 为应用程序提供的“自由市场”

Android 程序开发人员可以自由地选择任何一种他们希望的盈利模式。他们可以开发免费软件、共享软件，或者试用软件、靠广告盈利的软件以及付费软件。Android 在设计之初就致力于从根本上改变无线应用程序开发的现有规则。在过去，程序开发人员面临着许多与应用程序功能和特性无关的限制：

① 某一类型竞争软件数量的限制；

② 价格、盈利模式和版权费用的限制；

③ 运营商不愿意为小客户群提供应用程序。

使用 Android，开发人员可以编写并成功发布任何类型的应用程序。他们可以调整软件规模以适应小部分人群，而不必像移动运营商所期待的那样赚取大把的钞票。这样，则可以有针对性地为特定用户开发垂直市场应用程序。

由于开发人员对于应用程序的发布方式有着灵活自主的选择权，所以他们可以从中选择最有利的发布方式，而不必遵守别人的规则。Android 开发人员可以将他们的应用程序以多种形式发布到使用者手里。

图 1.7 Android Market

Google 开发了 Android Market（如图 1.7 所示），它是一个被普遍使用的以收入分享（revenue-sharing）模式运作的 Android 应

Download at http://www.pin5i.com/

用程序市场。

Handango.com 将 Android 应用程序纳入自己已建立的目录中，以使用其付费机制和收入分享模式。

开发人员可以使用自己的交付和支付机制。移动运营商也可以自由地开发自己的应用程序商店，同时建立自己的规则，但这已经不是开发人员发布应用程序的唯一方式了。

1.3.9 新生并逐渐成长的平台

Android 也许能成为下一代移动平台，但这一技术仍处于早期阶段。早期的 Android 开发人员不得不面对接触一个新平台时所要面对的典型困难：经常修改 SDK、缺乏文档资料和市场的不确定性。

另一方面，此时即开始钻研 Android 开发的开发人员能够因为率先进入市场而具有相当的竞争力，这在其他平台如 BREW、Symbian 上已经有所体现。早期的开发人员能够对 Android 平台给予一定的反馈，这将在一定程度上影响下一版本 SDK 所具有的功能，并且对其长远的规划与设计也有着积极的意义。最后，Android 拥有非常活跃而友好的论坛、社区。一些奖励性的项目，如 Android 开发人员挑战赛（Android Developer Challenge），已经吸引了众多新人对 Android 这一平台进行深入的挖掘。

Android SDK 的每一个新版本都对平台进行了实质的提升，在最新的修订中，Android 系统接受了相当有必要的“抛光打磨”，这使得无论在视觉效果还是在性能方面都得到了提升。尽管大多数的升级和改良都非常必要，而且受到了普遍的欢迎，但新的 SDK 版本常常在 Android 开发人员群体中激起不小的波澜。一批已经发布的应用程序需要重新进行测试，并提交到 Android 市场（Android Marketplace），以验证其符合新版 SDK 的要求。这迅速导致了所有 Android 手机固件上的升级，致使一部分旧的应用程序被淘汰。

一些较为老旧的 Android 设备无法运行新版本的 Android 平台。这意味着 Android 开发人员常常需要面向几个不同的 SDK 进行开发，以满足所有用户的需要。幸运的是，Android 开发工具使得这一过程前所未有地简单。

1.4 Android 平台

Android 本身是一个操作系统，也是供其上的应用程序进行开发的软件平台。同时，一些日常任务，如 Web 浏览、收发 E-mail 等，被作为核心应用程序加入到了 Android 手机中。

作为开放手机联盟一个强大而开源的无线应用开发环境，Android 是一个新兴的移动开发平台。设计这一平台的唯一目的是促成一个受移动用户和软件开发人员欢迎的、自由而开放的市场。

1.4.1 Android 的基本体系结构

Android 平台的设计使得其比之前的平台具有更强的容错能力。Android 手机设备运行在 Linux 操作系统下，这使得其上运行的 Android 应用程序具有很强的安全性。每一个 Andriod 应用程序均在其自身的虚拟机（virtual machine）中运行（如图 1.8 所示）。Android 应用程序为托管代码（managed code），所以，由应用程序而导致系统崩溃的可能性很小，这也降低了

Download at http://www.pin5i.com/

设备崩溃（device corruption）的可能性。

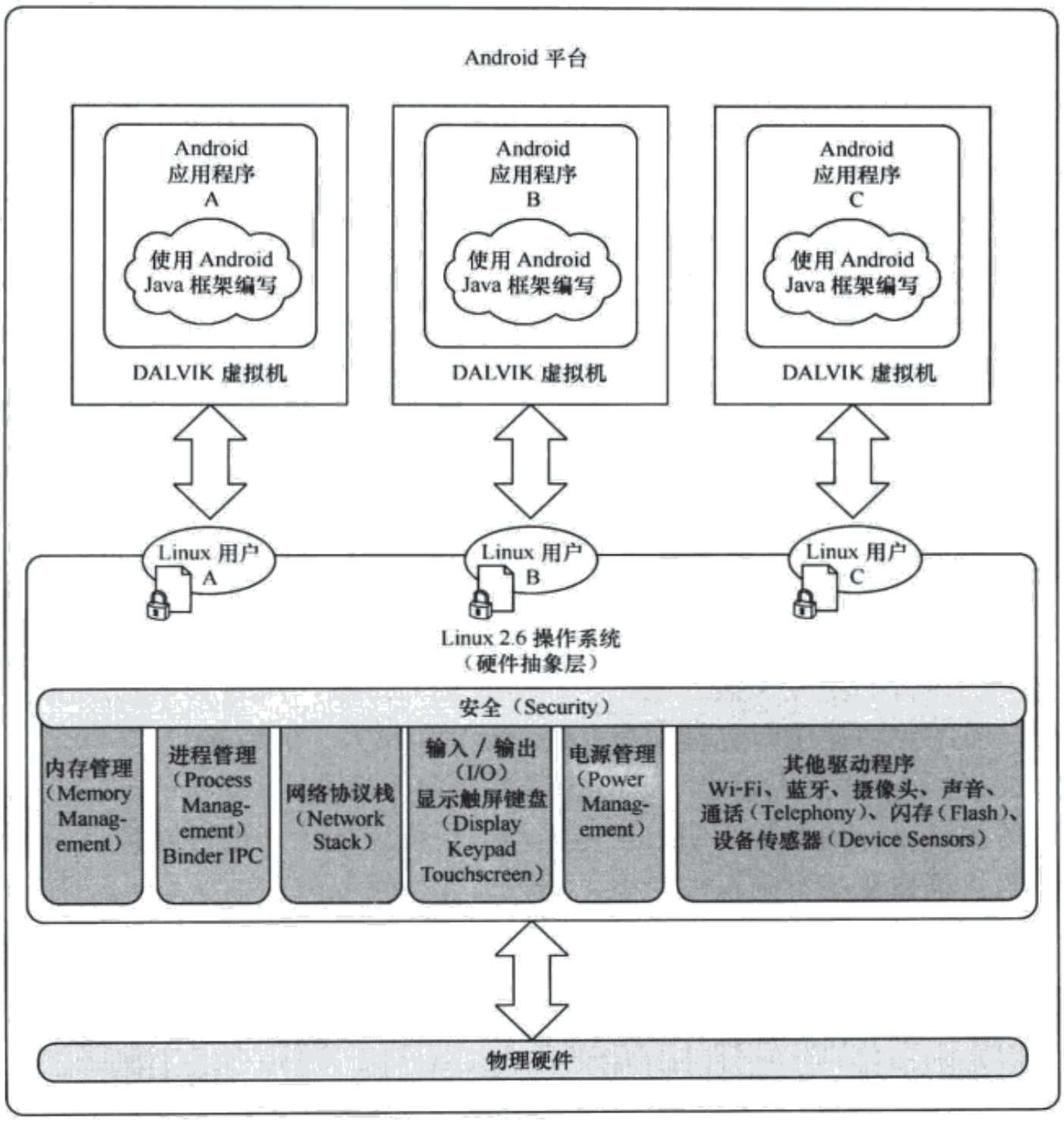


图 1.8 Android 平台体系结构图

1. Linux 操作系统

Linux 2.6 内核作为硬件抽象层（Hardware Abstraction Layer，HAL）在手机物理硬件与 Android 软件栈之间处理核心系统服务。

内核负责处理一些核心功能，包括：

① 应用程序强制执行的权限和安全；

② 低级（low-level）的内存管理；

③ 进程管理和线程操作；

④ 网络协议栈；

⑤ 显示、键盘输入、摄像头、Wi-Fi、闪存（Flash memory）、声音、Binder（IPC）（译注：IPC，Inter-Process Communication，进程间通信）驱动程序访问。

2. Android 应用程序运行时环境

每一个 Android 应用程序均运行在不同的进程中，这一进程对应一个 Dalvik 虚拟机

Download at http://www.pin5i.com/

（Virtual Machine，VM）的实例（instance）。基于 Java 虚拟机，Dalvik 为移动设备进行了优化设计。Dalvik 虚拟机具有较小的内存占用，而且多个 Dalvik 虚拟机的实例在手机设备上可以并发地运行。

1.4.2 安全与许可权限

若干种安全措施保障了 Android 平台的完整性。这些措施有助于保障用户数据的安全，并且避免硬件设备为恶意软件所利用。

（1）作为操作系统用户的应用程序

在安装了一个应用程序后，操作系统为该应用程序创建一个与之关联的新的用户配置文件。每一个应用程序都作为不同的用户运行，它在文件系统中拥有自己的私有文件、用户 ID 和一个安全的操作环境。

应用程序在操作系统上使用自己的用户 ID、自己的 Dalvik 虚拟机实例运行在自己的进程内。

（2）显式定义的应用程序权限

要访问系统上的共享资源，Android 应用程序需要注册所需的权限。有些权限可以允许应用程序使用手机设备的功能来拨打电话，而有些则允许接入网络、控制摄像头和其他硬件传感器。另外，应用程序在访问包括私有（private）和私人（personal）信息在内的共享数据（如用户首选项、位置、联系人信息等）时也需要一定的权限。

应用程序可以将自己的权限声明为可供其他应用程序使用，也可以为了更加精细地控制应用程序而声明任意数量的不同权限类型，如只读、读写权限。

（3）有限的特设（ad-hoc）授权

作为内容提供的应用程序也可能需要为其他应用程序提供即时的权限，以共享特定的信息。这可以通过使用统一资源标识符（Uniform Resource Identifier，URI）来进行临时的授权与撤销。

URI 建立对系统特定数据资源，如图像、文本等的索引。下面是一个 URI 的例子，它提供所有联系人的电话号码：

```
content://contacts/phones
```

为了了解这种权限的工作方式，让我们来看一个例子。

比方说，有一个应用程序，它用于记录用户的生日愿望，其中某些是可以公开的，而其他是用户不愿透露的。如果这一应用程序需要同其他应用程序共享数据，那么它就可以对可公开的“愿望”列表授予 URI 权限，以允许其他程序访问，而无需指明具体的列表项。

（4）应用程序通过签名建立用户信任

所有的 Android 应用程序包均使用证书进行了签名，这样用户可以了解应用程序的可靠性。证书的私钥（private key）为开发人员所有，这有助于在开发人员与用户之间建立一种信任关系，也使得开发人员可以控制系统上的哪些程序拥有访问权。不过，证书的颁发机构是没有存在的必要的，因为自签名的证书也可以被接受。

（5）市场开发人员注册

要在 Android Market 上发布应用程序，开发人员需要创建一个账号。Android Market 管理非常严密，不允许出现任何恶意软件。

Download at http://www.pin5i.com/

1.4.3 开发 Android 应用程序

Android SDK 提供了一整套广泛的应用程序接口（Application Programming Interface，API），它们既强大又能跟上时代前进的步伐。Android 手机设备的核心系统服务是向应用程序敞开的，所有应用程序均可以访问它们。在获得了适当的授权后，Android 应用程序之间可以进行数据共享，也可以安全地访问系统上的共享资源。

1. Android 编程语言选择

Android 应用程序使用 Java 编写（如图 1.9 所示）。目前，Java 语言是程序员在 Android 平台上进行开发的唯一选择。

图 1.9 Duke——Java 吉祥物

也有人推测，其他一些编程语言，如 C++，可能会在今后的 Android 版本中被引入。如果你的应用程序必须依赖于其他语言（如 C 或 C++）的本地代码，那么你可以考虑使用 Android 本地化开发包（Android Native Development Kit，NDK）将它们融合在一起。我们将在第 18 章“使用 Android NDK”中讨论这一话题。

2. 本地（native）与第三方应用程序间没有区别

不同于其他的手机开发平台，在 Android 平台上没有本地应用程序与开发人员创建的第三方应用程序的区别。在为应用程序提供适当授权后，所有的应用程序都拥有对核心库和底层硬件接口相同的访问权。

Android 手机设备默认移植了一系列的本地应用程序，如 Web 浏览器、联系人管理簿等。第三方应用程序与这些核心应用程序结合在一起，甚至对它们进行扩展，以丰富用户体验，甚至完全替代一些可选的应用程序。

3. 常用包（package）

使用 Android，移动开发人员再也不用从头做起，进行一些重复而无谓的开发。开发人员可以使用熟悉的 Android Java 包中提供的类库来完成一些常用的操作，如图形、数据库访问、网络接入、安全通信以及其他功能包（utility）（如 XML 解析器）。

Android 包包含对以下内容的支持：

① 常用用户界面 Widget（按钮、调节控件（Spin Controls）、文本输入框）；

② 用户界面布局；

③ 网络安全和 Web 浏览功能（SSL、WebKit）；

④ 结构化存储和关系型数据库（SQLite）；

⑤ 强大的 2D 和 3D 图形（包括 SGL 和 OpenGL ES）；

⑥ 音频和影音媒体格式（MPEG-4、MP3、静态图片）；

⑦ 对可选硬件设备的访问，如定位服务（Location-Based Services，LBS）、Wi-Fi、蓝牙和其他硬件传感器。

4. Android 应用程序架构

Android 应用程序架构提供开发一般应用程序的全部所需。Android 应用程序的生存周期

Download at http://www.pin5i.com/

内包括以下关键组件：

① Activity 是应用程序所执行的一系列函数；

② 一组视图（view）定义了应用程序的布局；

③ Intent 告知系统应用程序的计划（plan）；

④ 服务（Service）使得程序可以进行后台处理而无需与用户进行交互；

⑤ 通知（Notification）用于告知用户某些事件的发生。

Android 应用程序可以使用一系列的管理器（manager）同操作系统和底层硬件进行交互。每一个管理器负责保存部分底层系统服务的状态。例如，LocationManager 可以简化与手机定位服务之间的交互，ViewManager 和 WindowManager 管理用户界面基础（fundamental）。

应用程序可以同其他应用程序进行交互，或者作为 ContentProvider。诸如联系簿这样的内建应用程序即为“内容提供”程序，它可以允许第三方应用程序访问联系人数据，并且可以毫无限制地使用它们。

小　结

移动软件的开发随着时间推移不断向前发展。Android 成为了一种新的移动开发平台，它建立在之前的成功经验之上，同时避免了其他平台过去的失败。Android 的设计大大增强了开发人员的能力，以开发出具有创新特色的应用程序。这一平台完全开源，没有前期费用，使得开发人员可以充分享受到比在其他平台上进行开发更多的乐趣。

现在，是时候进行深入学习了，在开始编写 Android 代码后，你将会对 Android 到底能实现什么功能建立起理性的认识。

参考资料和更多信息

Android Development. http://developer.android.com.

Open Handset Alliance. http://www.openhandsetalliance.com.

Download at http://www.pin5i.com/

第2章

搭建你的 Android 开发环境

Android 开发人员在自己的计算机上编写和测试应用程序，然后将其部署到实际的设备上以供进一步测试。

在这一章中，你将学习如何安装与配置用于 Android 应用程序开发所需的全部工具。除此之外，我们还将探索 Android 软件开发工具包（Software Development Kit，SDK），看看它都提供了哪些有用的功能。

2.1 配置你的开发环境

要编写 Android 应用程序，你必须对编程环境进行一定的配置，以适应 Java 开发。所需的软件均可以免费下载。Android 应用程序可以在 Windows、Macintosh 和 Linux 系统下进行开发。

要进行 Android 应用程序开发，你需要在计算机上安装以下软件。

① Java Development Kit（JDK，Java 开发包）Version 5 或 6，可从如下地址下载：http://java.sun. com/javase/downloads/index.jsp。

② 兼容的 Java IDE，如 Eclipse，需要带有 JDT 插件，可以从如下地址下载：http://www.eclipse.org/downloads/。

③ Android SDK，包括工具和文档，下载地址为 http://developer. android.com/sdk/ index.html。

④ Eclipse 的 Android Development Tools（ADT，Android 开发工具）插件，可以通过 Eclipse 的软件更新机制下载。要获得有关插件安装的信息，可以参考 http://developer.android.com/sdk/eclipse-adt.html。虽然该工具对于开发而言是可选的，但我们强烈建议你安装它，并且在本书中，我们将会经常使用到它的某些特性。

完整的 Android 开发系统需求列表可以从以下网址获得：

http://developer.android.com/sdk/requirements.html。

安装说明位于如下网址：

http://developer.android.com/sdk/installing.html。

Tip

大多数开发人员使用流行的 Eclipse 集成开发环境（Integrated Development Enviroment，IDE）进行 Android 开发。Android 开发团队已经把 Android 开发工具直接整合到了 Eclipse IDE 中。当然，开发人员并不仅限于使用 Eclipse，他们也可以使用其他 IDE。如果想要获取更多关于使用其他开发环境的信息，可以访问以下链接：http://developer.android.com/guide/developing/other-ide.html。

Download at http://www.pin5i.com/

2.1.1 配置操作系统以适应设备调试

在 Android 手机设备上安装 Android 编译程序时，你需要将计算机操作系统调整至能够通过 USB 接口接受手机连接（如图 2.1 所示）的状态。在某些操作系统下，例如 Mac OS，无需任何配置即可直接使用，但在 Windows 下，你需要安装合适的 USB 驱动。以下链接中包含有 Windows 下可用的 USB 驱动程序：http://developer.android.com/sdk/win-usb.html。

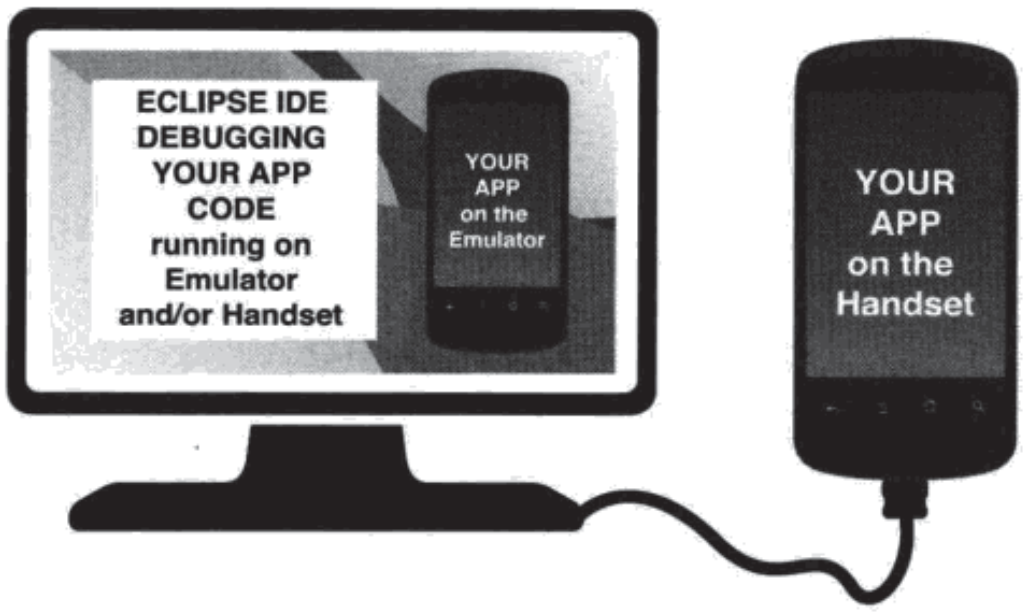


图 2.1 使用模拟器和 Android 手机进行 Android 应用程序的调试

2.1.2 配置 Android 硬件以供调试

Android 设备在默认情况不支持调试功能，所以，为了在设备上开发应用程序并且运行它们，你必须通过 USB 连接开启调试支持。

首先，你需要让你的设备允许安装来源于 Android Market 以外的 Android 应用程序，这可以通过进入 Home→Menu→Settings→Applications 来进行设置，你可以找到一个名为 Unknown Sources 的选项，然后开启它。

更多重要的开发设置可以进入 Home→Menu→Settings→Applications→Development（如图 2.2 所示）来进行设置。你需要开启如下选项。

- USB 调试（USB Debugging）：该设置允许你通过 USB 连接调试应用程序。
- Stay Awake：在设备连接后，该选项将避免设备在你进行开发工作时进入休眠状态。
- Allow Mock Locations：出于开发目的，该设置允许你向手机发送虚拟的位置信息，这对于使用位置服务（LBS）的应用程序将非常方便。

图 2.2 Android 调试设置

2.1.3 升级 Android SDK

Android SDK 有时会进行更新。通过使用 Android SDK 和 AVD 管理器，你可以非常方

Download at http://www.pin5i.com/

便地在 Eclipse 内部对 Android SDK 和工具进行更新，AVD 管理器是 Eclipse 下 ADT 插件的一部分。

Android SDK 的变动可能包含功能的增添、升级和移除，包名称的改变，以及工具的升级。对于每一个新版本的 SDK，Google 都提供以下有用的文档。

- 变化概览：对于 SDK 主要变动的简要描述。
- API 差异报告（Diff Report）：SDK 中具体变化的详细、完整列表。
- 版本注释：SDK 的已知问题列表。

你可以在如下网址找到有关增添和升级 SDK 组件的更多信息：http://developer.android.com/sdk/adding-components.html。

2.1.4 Android 软件开发包的问题

由于 Android SDK 始终都处于不断开发完善的状态，所以你可能会发现 SDK 的一些问题。如果你认为你发现了一个问题，那么在 Android 项目“问题跟踪（Issue Tracker）”网站上你将找到一个开放问题及其状态的列表。另外，你也可以提交一个新的问题报告，以供他人审阅。

Android 开源项目的“问题跟踪”网站的网址为：http://code.google.com/p/android/issues/list。通过下面的网址，你可以获得更多有关提交 bug 或漏洞问题的信息，提交的这些问题将会由 Android 平台开发团队进行处理：http://source.android.com/source/report-bugs.html。

> **Tip**
>
> 你提交的 bug 需要多长时间才能得到解决呢？你可以对 Android bug 的解决过程进行一些了解，网址如下：http://source.android.com/source/life-of-a-bug.html。

2.2 探索 Android SDK

Android SDK 主要由以下 5 个部分组成：Android SDK 许可协议、Android 文档、应用程序框架、工具和实例应用程序。

2.2.1 理解 Android SDK 许可协议

在下载 Android SDK 之前，你需要阅读并同意 Android SDK 许可协议，该协议是用户（开发者）和 Google（Android SDK 的版权持有人）之间的契约。

即使你的某个同事或朋友已经代表你同意了此协议，但作为一个开发者，你还是应该对以下几个重要条款有所了解：

（1）授权

Google（作为 Android 的版权持有人）授予用户一项有限的、全世界范围内的、免交使用费的、不可转让及非专有的许可，允许用户使用 SDK 进行 Android 平台应用程序开发。Google（以及第三方贡献者）授予用户许可，但是仍保留全部材料的版权及知识产权。使用 Android SDK 并不意味着用户被授权使用 Google 的其他品牌标志，包括但不限于 Google 品牌、徽标和商标等。你将不能删除其中任何版权声明。你的应用程序和其他第三方开发的程序之间的交互不属于本协议限定范围，接受独立条款的约束。

Download at http://www.pin5i.com/

（2）SDK 的用法

SDK 仅可被用于开发 Android 应用程序。任何衍生用法都是被禁止的，包括但不限于在其他任何设备上非法发布 SDK 或将部分 SDK 随其他软件发布。

（3）SDK 的变化及后向兼容性

Google 可能会随时更新 SDK 而不做任何通知，且没有责任考虑其后向兼容性。虽然 Android API 的变化是预发布版本 SDK 的主要问题，但是最新的发布的版本在一定程序上是稳定的。也就是说，每一次 SDK 更新都会影响到少部分现有的应用程序，而迫使其更新。

（4）开发者的权利

你可以保留你用 SDK 开发的 Android 软件的所有权利，包括知识产权。同时，你也对自己的作品承担一切责任。

（5）Android 应用程序的隐私规定

你需要承诺你开发的应用程序能够保护其用户的隐私及其他合法权益。如果你的应用程序使用或接受了用户的个人信息或私密信息（如用户名、密码等），那么你的应用程序应该提供一个合适的隐私声明并将此数据安全地存放起来。需要注意的是，隐私法律和法规可能会因用户所在地区的不同而不同，你作为一位开发人员需要独立地对数据的管理承担责任。

（6）Android 应用程序关于恶意软件的要求

你需要对你所开发的全部应用程序承担相应的责任。你需要保证不编写具有破坏性的软件或恶意软件。对你开发的应用程序中传输的所有数据，你负有全部责任。

（7）关于 Google 特定 API 的附加条款

使用 Android 地图 API 需要接受进一步的服务条款（尤其在使用 `com.google.android.maps` 和 `com.android.location.Geocoder` 包时）。在使用这些特定 API 之前你必须同意特殊的条款，并需要在软件中加入“Google Maps 保留版权”的提示。使用 Google 数据 API（Google Apps，如 Gmail、Bloger、Google Calendar、Google Finance Portfolio Data、Picasa、YouTube 等）时，仅限于用户在安装程序中显式地授予你的应用程序相关的权限。

（8）开发风险自担

使用服务时产生的一切风险均由你自行承担，Google 无需向你承担责任。

2.2.2 阅读 Android SDK 文档

本地化的 Android 文档存储在磁盘的/docs 子目录下（如图 2.3 所示）。

文档主要由以下 7 个部分组成。

① Home 标签通常是 Android 文档的起始点。在这里你将找到开发人员注意事项和一些有关 Android 开发的最新热门话题的重要链接。

② SDK 标签提供有关不同 Android SDK 版本之前的差异信息，同时还有关于 Android 本地化开发包（Android Native Development Kit，NDK）的相关内容，以及 Android SDK 的版本注释。

③ Dev Guide 标签介绍 Android 平台，并且涵盖了 Android 应用程序设计和开发的最佳实践，以及有关发布应用程序的信息。

④ Reference 标签提供 Android API 中具体的类和接口的详细层级列表。

⑤ Resources 标签提供对于 Android 技术文章和指南的访问。在这里你还可以找到

Download at http://www.pin5i.com/

Android 在线社区（群组、邮件列表以及官方 Twitter），以及随 Android SDK 提供的示例应用程序。

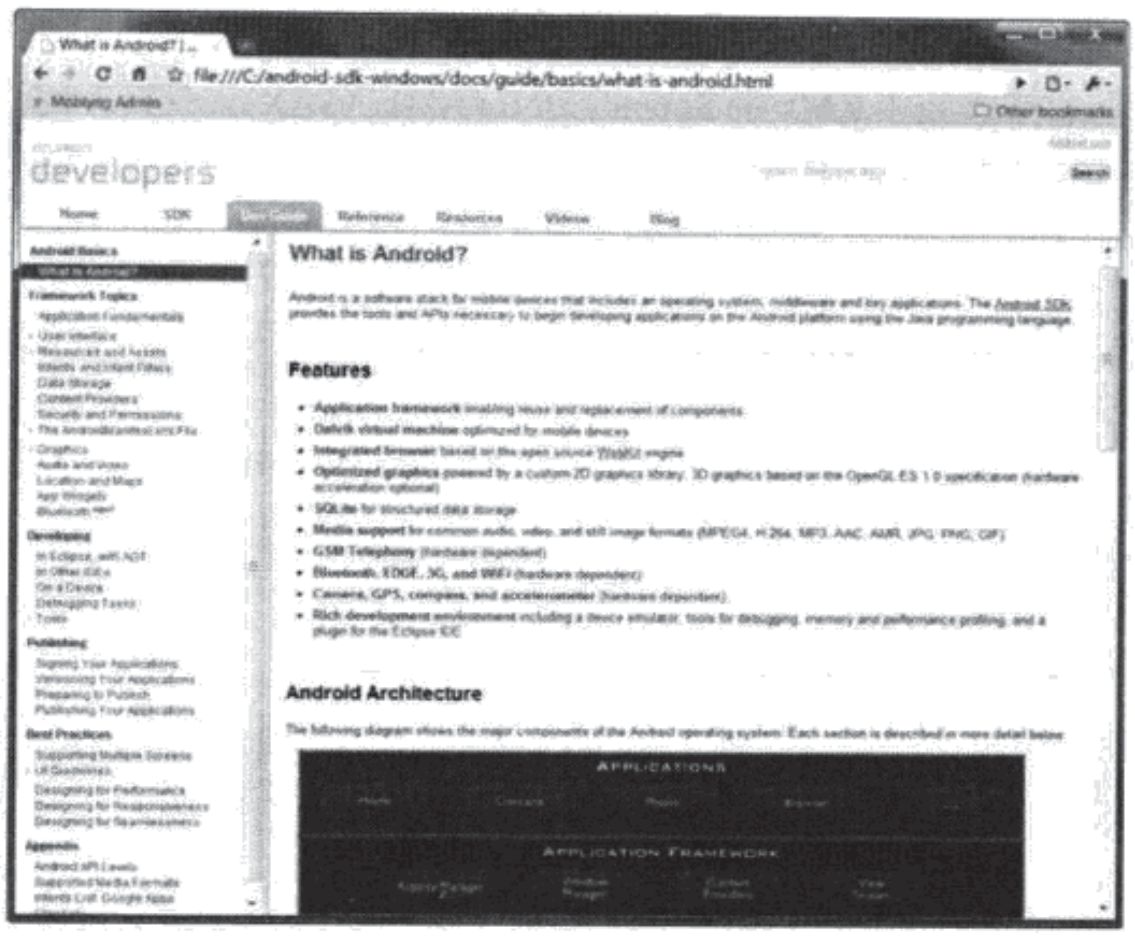


图 2.3 Android SDK 文档

⑥ Videos 标签提供对于有关 Android 开发的在线视频的访问，这些视频包括开发平台、开发人员 Tip，以及来源于每年一次的 Google I/O 大会的 Android 开发会议，还有开发人员“沙盒”访谈（sandbox interviews）。

⑦ Blog 标签提供对于 Android 开发团队发布的在线 Blog 的访问。在这里你将找到有关 SDK 版本的通告、有价值的开发 Tip 和即将到来的有关 Android 的事件的通知。

Android 文档在本地以 HTML 格式提供，在线可访问 http://developer.android.com。Android 文档中的某些网络资源（例如 Blog 和 Video 标签）只可以在线访问。

2.2.3 探索 Android 应用程序框架

`Android.jar` 文件提供了 Android 应用程序框架。Android SDK 由几个重要的包构成，如表 2.1 所示。

表 2.1 Android SDK 中重要的包文件

顶 级 包	用 途
`android.*`	Android 应用程序基础
`dalvik.*`	Dalvik 虚拟机支持类
`java.*`	核心类，提供熟悉的常用功能，如网络、安全、数学函数等
`javax.*`	Java 扩展类，包括加密支持、解析器、SQL 等
`junit.*`	单元测试支持
`org.apache.http.*`	超文本传输协议（Hypertext Transfer Protocol，HTTP）支持
`org.json`	JavaScript Object Notation（JSON）支持

Download at http://www.pin5i.com/

续表

顶 级 包	用 途
org.w3c.dom	用于文档对象模型（Document Object Model）核心（XML 和 HTML）的 W3C Java 绑定
org.xml.sax.*	用于 XML 的 Simple API for XML（SAX）支持
org.xmlpull.*	高性能的 XML 解析

另外还有一个可选的 Google API 附加组件，它作为 Android SDK 的扩展，帮助开发人员能够更好地使用 Google Maps 和其他 Google API 及服务。例如，如果你想要在你的应用程序中包含 MapView 控件，那么就需要安装和使用这一特性。该附加组件对应 com.google.* 包(包括 com.google.android.maps)以及 API Key 所需的附加服务协议条款。有关 Google API 附加组件的更多信息，可以参考 http: //code.google.com/android/add-ons/google-apis/。

2.2.4 了解 Android 工具

Android SDK 提供了许多用于设计、开发、调试和部署你的应用程序的工具。Eclipse 插件已将这些工具中的一部分无缝地集成到了你的开发环境当中，它们将提供用于创建和调试 Android 工程的各种不同的向导。

ADT 插件的设置选项可以在 Eclipse 的 Window→Preferences→Android 下找到。在其中，你可以设置安装 Android SDK 及相关工具的磁盘位置，以及其他许多构建（build）和调试设置。

ADT 插件向默认的 Eclipse IDE 中添加了不少有用的功能。在工具条上，你可以找到几个新增的按钮：

① 运行 Android SDK 和 AVD 管理器；

② 使用 Android 工程向导创建新的工程；

③ 使用 Android 工程向导创建新的测试工程；

④ 创建新的 Android XML 资源文件。

以上功能都可以通过 Eclipse 工具条中的按钮实现，如图 2.4 所示。

图 2.4 添加至 Eclipse 工具条的 Android 功能

另外，还有一个特殊的 Eclipse 视图，它用于 Android 应用程序的调试，被称作 DDMS（Dalvik Debug Monitor Server）。在 Eclipse 中，你可以通过选择 Window→Open Perspective→DDMS 来切换到该视图，也可以通过点击屏幕右上方的 DDMS 视图按钮来实现切换。我们将在本章的后面部分讨论更多有关 DDMS 的话题。在你设计完一个 Android 应用程序之后，还可以使用 ADT 插件来运行向导，为你的 Android 应用程序打包并签名，以供之后的发布。我们将在第 29 章“销售你的 Android 应用程序”中讨论更多有关这一话题的内容。

1. Android SDK 和 AVD 管理器

Android SDK 和 AVD 管理器如图 2.5 所示，是集成在 Eclipse 中的一个工具。它主要完成两个功能：管理开发机器上多个版本的 Android SDK 以及管理开发人员的 Android 虚拟设备（Android Virtual Device，AVD）配置。

Download at http: / / www.pin5i.com /

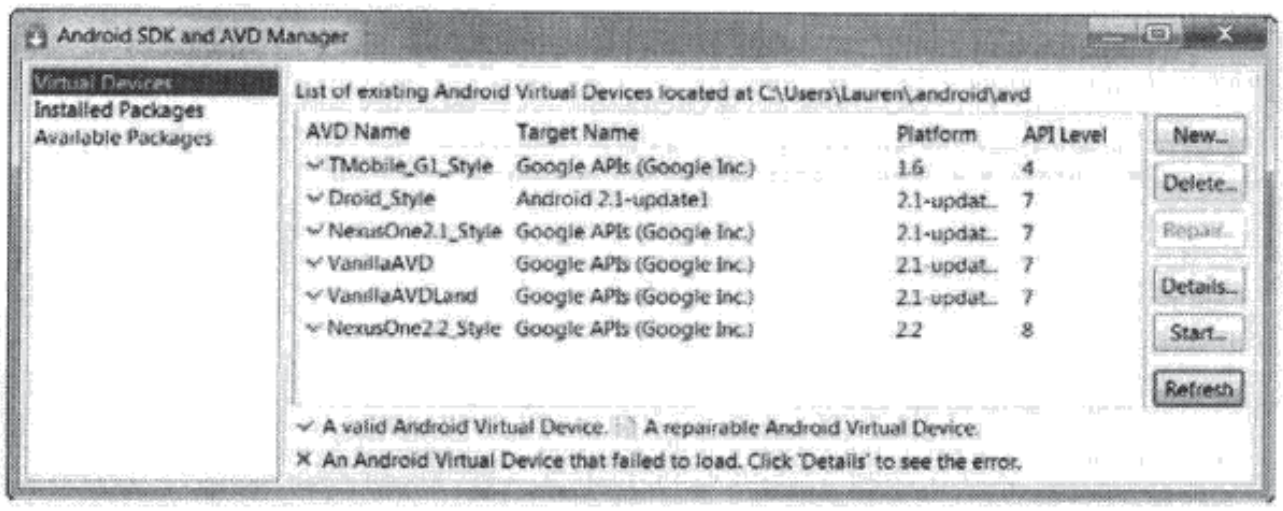


图 2.5 Android SDK 和 AVD 管理器

如同台式计算机，不同的 Android 设备运行在不同版本的 Android 操作系统下。开发人员所开发的应用程序需要适用于不同的 Android SDK。一些应用程序仅仅针对某个特定的 Android SDK，但另一些却会尝试同时提供对尽可能多的版本的支持。

Android SDK 和 AVD 管理器帮助 Android 开发人员开发出跨越多个平台版本的应用程序。当一个新的 Android SDK 发布以后，你可以使用这一工具下载和升级现有的开发工具，它们将继续保持向下兼容，并且使用老版本的 Android SDK。

该工具还管理 AVD 配置。要管理 Android 模拟器中的应用程序，你必须配置 AVD。AVD 配置文件描述模拟器需要模拟的设备类型，包括所支持的 Android 平台。你可以指定不同的屏幕尺寸和方向，还可以设定模拟器是否带有 SD 卡及其容量。

2. Android 模拟器

Android 模拟器如图 2.6 所示，是 Android SDK 提供的最重要的工具。在设计和开发 Android 应用程序时，你将会经常使用它。模拟器运行在你的电脑上，并且能够表现出和移动设备几乎一样的行为。你可以将 Android 应用程序载入模拟器进行测试和调试。

图 2.6 Android 模拟器

Download at http://www.pin5i.com/

模拟器是具有普适性的，并不特别针对某一款型号的手机。你需要通过 AVD 配置来描述模拟器需要模拟的硬件和软件细节。

Tip

你应该注意到了 Android 模拟器是真实的 Android 设备的替代品，并且是一个并不完美的替代品。模拟器是很有价值的测试工具，但并不能完全取代在真实的目标设备上的测试。

有关模拟器的更多信息，可以参考附录 A“Android 模拟器快速入门”。你也可以从 Android SDK 文档中获取有关 Android 模拟器的详尽信息：http://developer.android.com/guide/developing/tools/emulator.html。

3. Dalvik 调试监视服务（DDMS）

Dalvik 调试监视服务（The Dalvik Debug Monitor Service，DDMS）是一个通过 Android 开发工具插件集成在 Eclipse 中的命令行工具（如图 2.7 所示）。该工具为你提供连接设备的直接接口——不管是像模拟器这样的虚拟设备还是像手机这样的真实设备。你可以通过 DDMS 查看和管理运行在设备上的进程和线程、查看堆积数据、连接到进程进行调试，以及完成许多其他各种各样的任务。

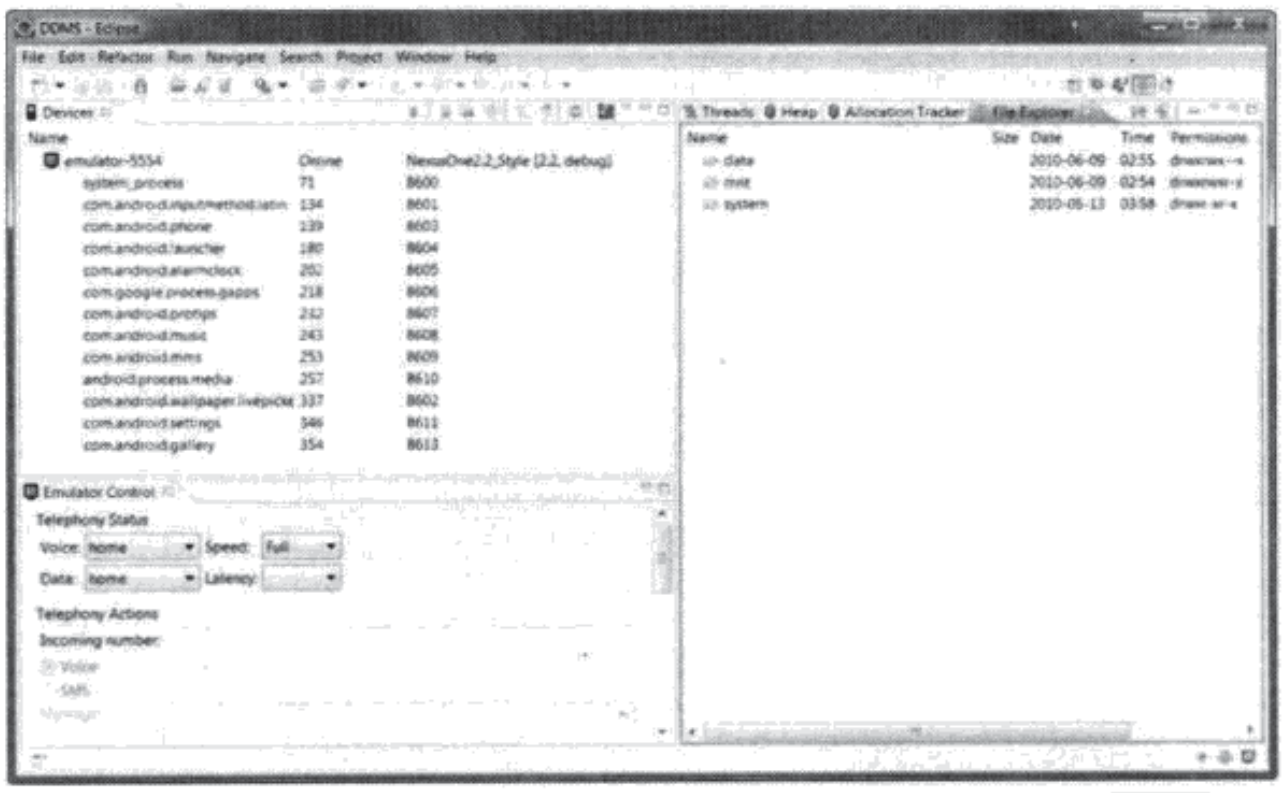


图 2.7 使用集成在 Eclipse 中的 DDMS 的界面

想获取关于 DDMS 的更多信息，请参考附录 B“Android DDMS 快速入门”。你也可以在 Android SDK 文档中找到关于 DDMS 的详细描述：http：//developer.android.com/guide/developing/tools/ddms.html。

4. Android 调试桥（ADB）

Android 调试桥（The Android Debug Bridge，ADB）是一个允许开发者通过标准 Java IDE，如 Eclipse，在模拟器和真实设备上调试 Android 代码的客户端/服务器工具。DDMS 和 Eclipse 下的 Android 开发插件都使用调试桥协调开发环境与模拟器（或设备）之间的工作。

开发者同样可以利用调试桥与设备文件系统进行交互，手动安装 Android 应用程序或是

Download at http://www.pin5i.com/

发布 Shell 命令。例如，sqlite3 命令，它允许你接入设备数据库；软件试验程序 Monkey（Application Exerciser Monkey，以下简称 Monkey），它能够随机生成用户输入，从而对你的软件进行压力测试。调试桥对于开发者而言最为重要的一个功能是日志系统（Logcat）。

想要获取更多关于调试桥的信息，请参考附录 C“Android 调试桥快速入门”。Android SDK 文档同样给出了详尽的说明：http：//developer.android.com/guide/developing/ tools/adb.html。

5. Android 层级阅览器

Android 层级阅览器（Hierarchy Viewer），如图 2.8 所示，是一个可视化工具，它能够显示出各组件的布局及其联系，从而帮助开发者设计及调试用户界面。开发者可以利用这个工具查看 View 的属性，开发出“像素完美”的布局。关于用户界面设计和 Android 层级阅览器的更多信息，我们将会在第 8 章“使用布局设计 Android 用户界面”中讲到。

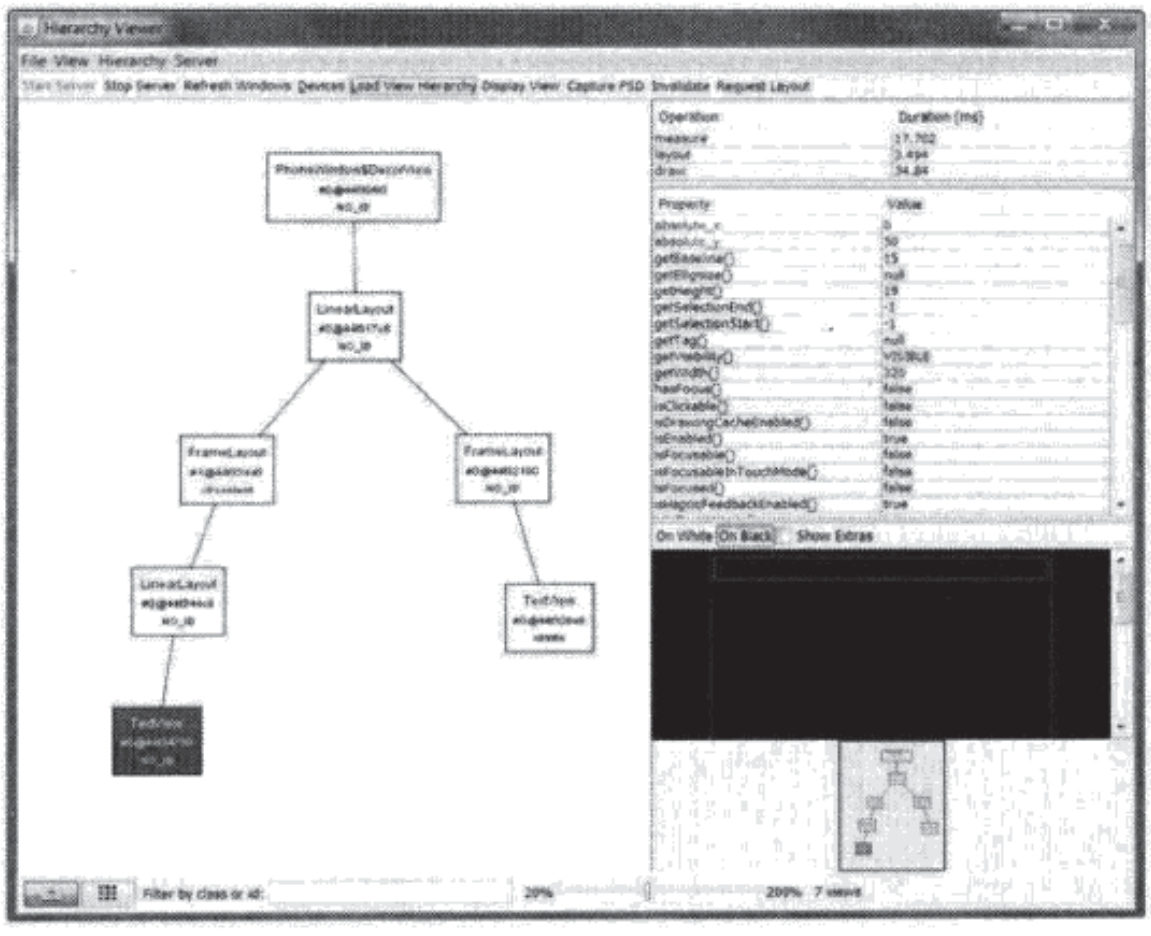


图 2.8　运行中的 Android 层级阅览器截屏

6. 其他工具

Android SDK 还有许多其他的工具。其中的许多提供底层功能的工具都通过 ADT 插件集成在 Eclipse 中了，如果你不用 Eclipse，这些工具可以在命令行中使用。

其他的工具有各自的特殊用途。例如，“Draw Nine-patch”工具能够帮助你设计一幅可以拉伸的 PNG 图片，使你的程序能够更好地适应不同大小的屏幕。还有，layoutopt 工具可以帮助开发人员优化他们的用户交互性能。接下来的章节中，我们在需要用到这些工具的时候将会对它们进行详细的说明。

你可以在以下网址找到全部关于 Android 工具的信息：http://developer.android.com/guide/developing/tools/index.html。

2.2.5　探索 Android 示例应用程序

Android SDK 提供了很多例子和 Demo 应用程序来帮助你学习 Android 程序开发。其中

Download at http://www.pin5i.com/

许多都是作为 Android SDK 的一部分提供的，位于 Android SDK 的/samples 子目录下。你可以在 Android 开发人员网站的 Resources 标签下找到更多的示例应用程序。

> **Tip**
>
> 对于某些 Android SDK 的安装程序，示例应用程序必须单独下载，这可以通过使用 Android SDK 和 AVD 管理器来升级 SDK 安装程序来实现。所有的示例应用程序均可以在 Android 开发人员网站上找到。

一些最简单、最直接的 Demo 应用程序如下所示。

- ApiDemos：一个菜单驱动的功能包，用于演示各种 Android API 的功能，涉及从用户界面 Widget 到应用程序生命周期中的各个组件，如服务、警告和通知。你可以在下面的网址找到一篇有关它的评论：http://developer.android.com/resources/samples/ApiDemos/。
- Snack：一个简单的小游戏，用于演示位图描绘和键盘操作。你可以在下面的网址找到一篇有关它的评论：http://developer.android.com/resources/samples/Snake/。
- NotePad：一个简单的列表程序，用于演示数据库访问和动态文件夹（Live Folder）的功能。你可以在下面的网址找到一篇有关它的评论：http://developer.android.com/resources/samples/NotePad/。
- LunarLander：一个简单的小游戏，用于演示绘图与动画。你可以在下面的网址找到一篇有关它的评论：http://developer.android.com/resources/samples/LunarLander/。

还有其他很多示例应用程序，只不过它们演示的都是某些具体的 Android 特性，这些特性将在本书后面部分予以介绍。

小　结

在这一章，你安装、配置并了解了编写一个 Android 应用程序所需的一切工具，包括合适的 JDK、Eclipse 开发环境和 Android SDK。你探索了许多随 Android SDK 一起提供的工具，并了解了它们的功能。最后，你还体验了 Android SDK 提供的示例程序。

现在你应该已经拥有了比较好的环境配置，可以开始编写第一个 Android 应用程序了。

参考资料和更多信息

Google 提供的 Android 开发人员向导.http://developer.android.com/guide/index.html.

Android SDK 下载站点.http://developer.android.com/sdk/.

Android SDK 许可协议.http://developer.android.com/sdk/terms.html.

Java 平台（标准版）.http://java.sun.com/javase.

Eclipse 项目.http://www.eclipse.org.

Download at http://www.pin5i.com/

第3章

编写第一个 Android 应用程序

到现在为止，在你的计算机上应该已经有了一个可以使用的 Android 开发环境了。如果理想的话，你应该也拥有了自己的 Android 硬件设备。现在，你可以开始编写 Android 应用程序了。

在本章中，你将学习如何在 Eclipse 中添加或创建一个工程；如何检验你的 Android 开发环境是否已经正确地进行了设置。你还将编写和调试你的第一个 Android 应用程序，它将运行在模拟器和真实设备上。

3.1 测试你的开发环境

测试你的开发环境是否已经正确配置的最好办法是导入一个现成的 Android 应用程序，然后运行它。通过 Android SDK 提供的示例程序，你可以非常容易地实现它，这些示例程序包含在/samples 子目录下。

在 Android SDK 提供的示例应用程序中，你能找到一个名为 Snake（贪食蛇）的游戏。要构建和运行 Snake 程序，你需要在 Eclipse 工作空间中创建一个新的 Android 工程以及一个适当的 Android 虚拟设备（AVD）配置文件，并且为该工程进行运行配置。一切准备就绪后，你就可以在 Android 模拟器或者 Android 设备上构建并运行该程序了。

3.1.1 在 Eclipse 工程中导入 Snake 应用程序

你要做的第一件事情是在 Eclipse 工作空间中导入 Snake 工程。要完成这一任务，可以参考以下步骤。

① 选择 File→New→Project。

② 选择 Android 节点下的 Android Project 向导（如图 3.1 所示）。

Tip

在使用了一次 Android Project 向导后，你可以继续执行 File→New→Android Project 菜单命令创建后续的工程。

③ 修改 Content 区域为“Create project from existing source”。

④ 定位至你的 Android 示例目录。

⑤ 选择 Snake 目录，而后所有的工程表单区域都将被自动填入，这一功能是通过 Manifest 文件实现的（如图 3.2 所示）。你可能还需要将 Build Target 设置为测试设备的特定 Android 版本。

Download at http://www.pin5i.com/

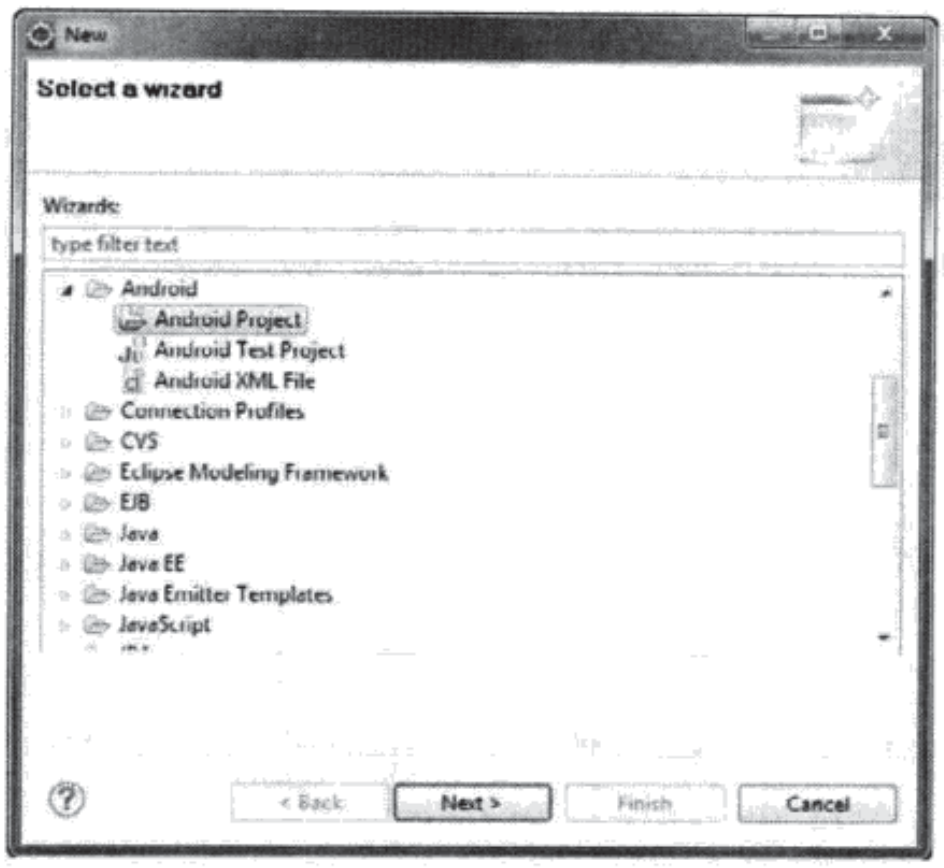


图 3.1 创建一个新的 Android 工程

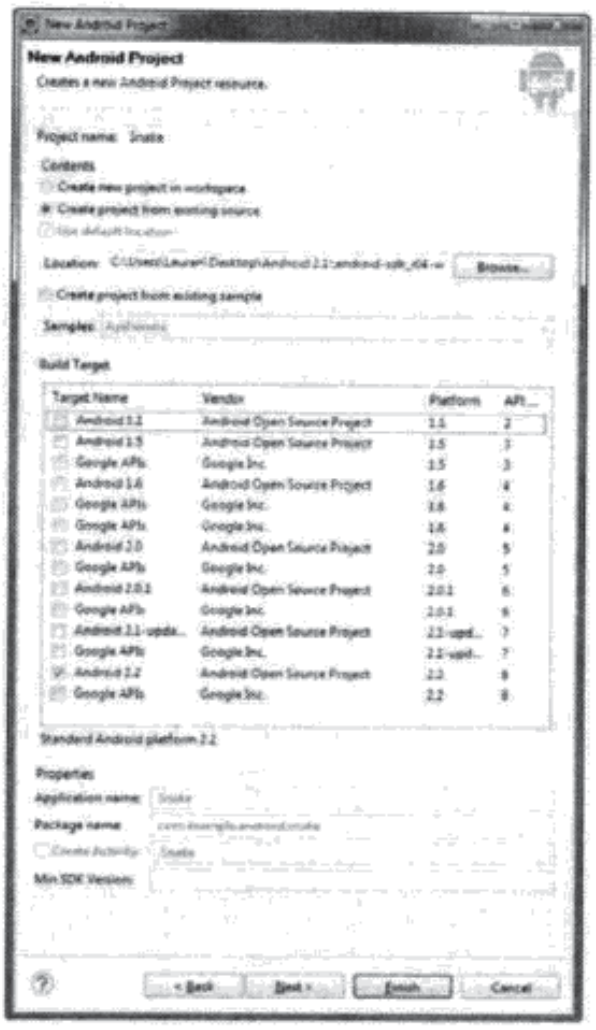


图 3.2 Snake 工程细节

⑥ 单击 Finish。现在你应该能够在工作区域（workspace）中看见 Snake 工程文件了（如图 3.3 所示）。

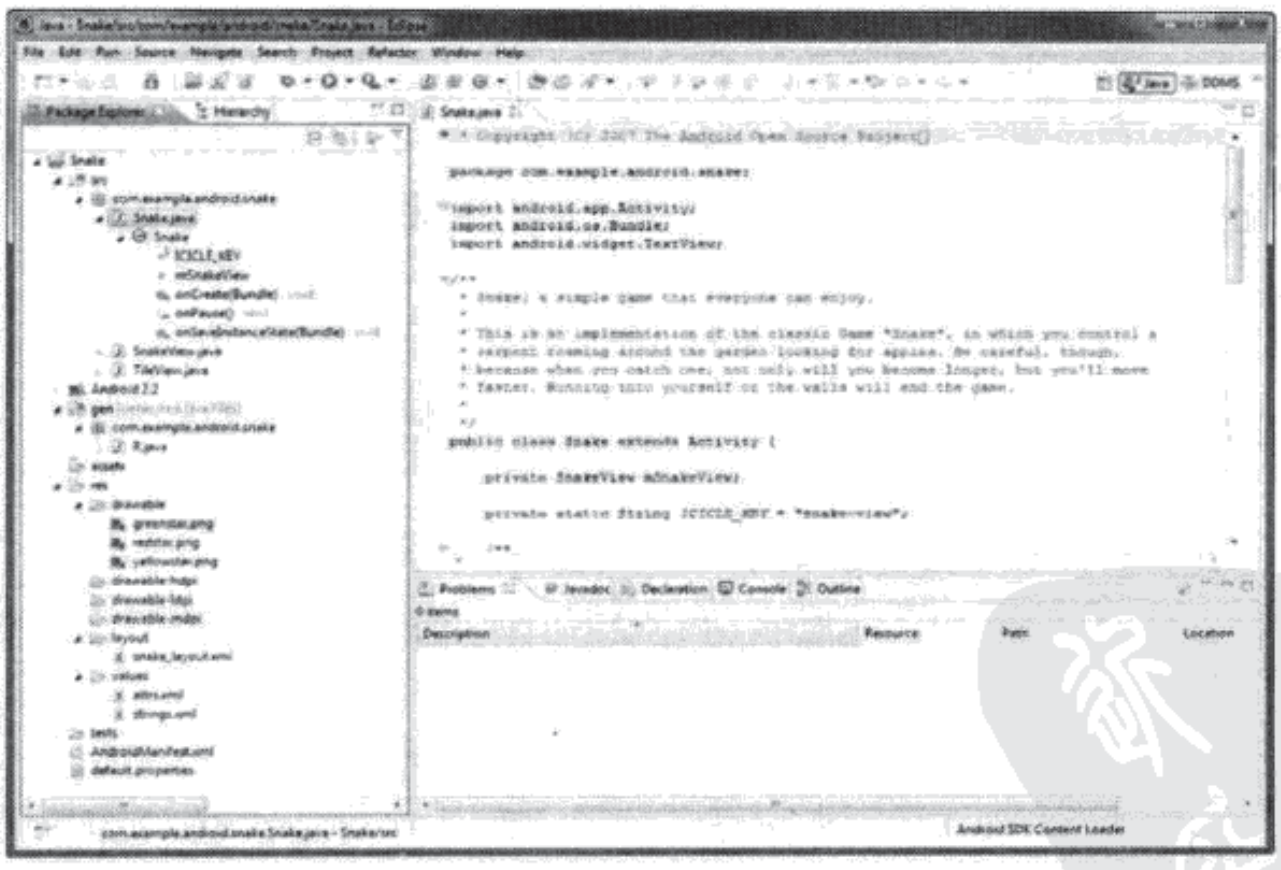


图 3.3 Snake 工程文件

Warning

某些时候，当你向工作空间中添加现存的工程时，Eclipse 可能会显示“Project ‘Snake’ is missing required source folder: ‘gen’”错误。如果这样，你只需要简单地将/gen 目录下的 R.java 文件删除即可。R.java 文件随后会自动生成，错误即可避免。

Download at http://www.pin5i.com/

3.1.2 为 Snake 工程创建 Android 虚拟设备（AVD）

下一步需要创建 Android 虚拟设备（Android Virtual Device，AVD），它描述用于运行 Snake 应用程序所要模拟的设备。这个 AVD 配置文件描述了你想用模拟器虚拟一个什么样的设备，包括它支持哪个版本的 Android 平台。你可以指定不同的屏幕尺寸和方向，可以指定模拟器中有没有 SD 卡，如果有，你甚至还可以指定存储卡的容量是多大。

在这个例子中，我们创建一个简单的 AVD，它能够满足一个默认 Android 2.2 设备的基本要求。下面我们给出创建一个基本的 AVD 的步骤。

① 点击 Eclipse 工具条中带有下箭头的 Android 图标，运行 Android 虚拟设备管理器（Android Virtual Device Manager）。如果找不到这个图标的话，也可以通过 Eclipse 的 Window 菜单来运行它。

② 在 Virtual Devices 菜单上，单击 New 按钮。

③ 为你的 AVD 取个名字。由于我们之后均使用默认参数，所以此处将 AVD 命名为 Android_Vanilla2.2。

④ 选择一个构建目标。我们需要一个基本的 Android 2.2 设备，所有在下拉菜单中选择 Android 2.2。

⑤ 选择 SD 卡的容量。单位可能为 KB 或 MB，它将使用硬盘上的空间。选择一个合理的值，比如 1GB（1024MB）。如果你的磁盘空间不足，或者确定应用程序不需要那么大的 SD 卡空间，那么可以选择一个更小的值，比如 64MB。

⑥ 选择一个皮肤。这个选项将控制模拟器的分辨率。这里我们使用 WVGA800 的屏幕，如图 3.4 所示。皮肤一般与相应的设备对应看，如 HTC Nexus One 和 Evo 4G，它们均是我们的开发机型。

图 3.4 在 Eclipse 中创建一个新的 AVD

Tip

虽然我们这里为 AVD 选择了一个相对较高的分辨率，但是具体选用怎样的分辨率需要由你的应用程序运行的目标设备所确定。

⑦ 单击 Create AVD 按钮，等待后台操作完成。

⑧ 单击 Finish 按钮。由于 AVD 管理器需要为 SD 卡镜像格式化内存，所以创建带有 SD 卡的 AVD 需要一些时间。

Tip

你也可以使用 Android 命令行工具来创建 AVD。

有关创建不同类型 AVD 的更多内容，可以参考附录 A“Android 模拟器快速入门”。

3.1.3 为 Snake 工程创建运行配置参数

下一步，你需要在 Eclipse 环境中创建运行配置参数，Snake 应用程序将在这一环境下构建及运行。运行配置参数用于配置模拟器的参数和应用程序的入口点。

Download at http://www.pin5i.com/

你需要使用不同的选项分别设置运行参数和调试参数。在 Eclipse 环境中，可以在 Run 菜单下创建这些参数（Run→Run Configuration→Run/Debug Configuration）。

下面的步骤演示如何为 Snake 应用程序创建基本的运行配置参数。

① 选择 Run→Run Configurations（或右键单击 Project→Run As）。

② 双击 Android Application。

③ 将运行配置命名为 SnakeRunConfiguration（如图 3.5 所示）。

图 3.5 Snake 工程运行配置，选择了 AVD 的 Target 标签页

④ 单击 Browse 按钮并选中 Snake 工程文件。

⑤ 切换至 Target 标签页，选择 Android_Vanilla2.2 AVD，这是之前从 AVD 列表中所创建的，如图 3.5 所示。

另外，你也可以在 Target 和 Common 标签页中设置其他的选项，但是此处我们使用默认设置。

3.1.4 在 Android 模拟器中运行 Snake 应用程序

现在你就可以根据接下来的步骤运行这个 Snake 应用程序了。

① 在工具栏下拉菜单中点击 Run As 图标（带三角的绿色圆圈）。

② 在出现的列表中选择你刚刚创建的 SnakeRunConfiguration。

③ Android 模拟器开始启动，这一过程可能需要持续一段时间。

Tip

确保此时你并没有使用 USB 连接 Android 设备，因为 Run Configurations 对话框的 Target 标签中 Device Target Selection 模式的默认设置为 Automatic，否则，Snake 应用程序将在目标设备（而非模拟器）中运行。

④ 如果需要的话，用手指从左至右划过屏幕解开屏幕锁，如图 3.6 所示。

⑤ 此时 Snake 应用程序启动了，如图 3.7 所示。

现在，你可以尝试着在模拟器上玩玩这个游戏。你也可以随时通过单击应用程序 drawer（抽屉）中 Snake 的图标来启动它。

Download at http://www.pin5i.com/

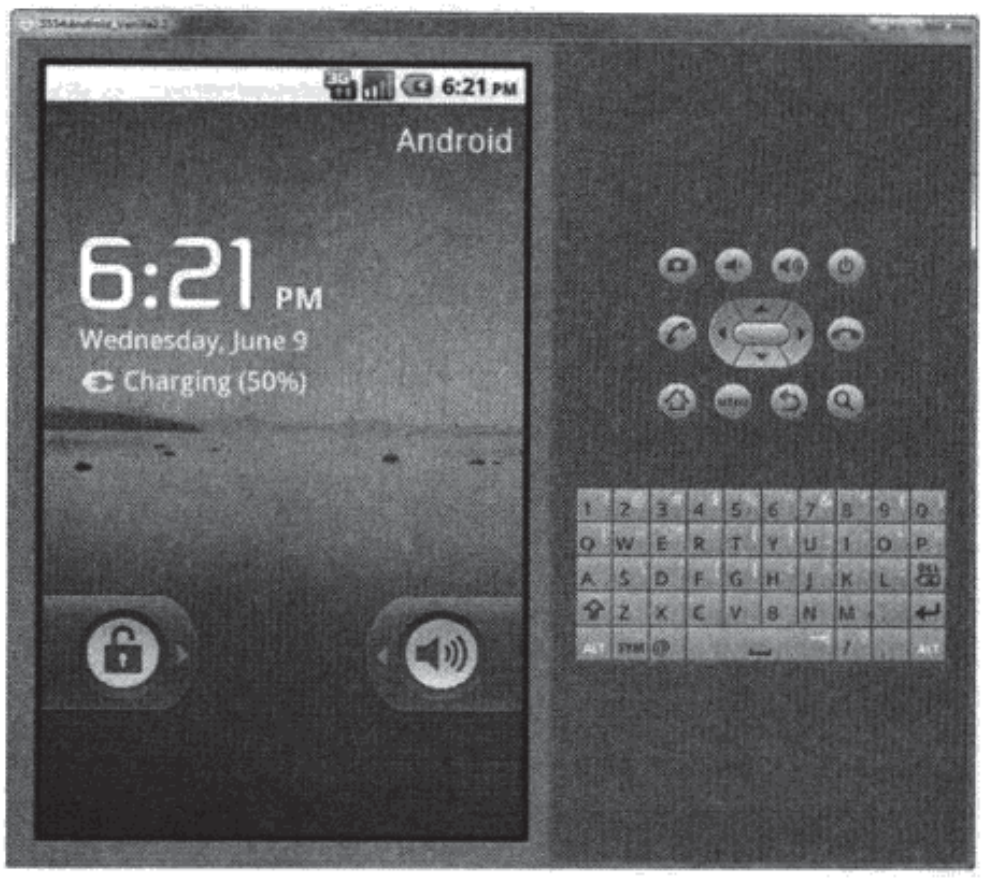


图 3.6 Android 模拟器运行界面（锁定状态）

图 3.7 Snake 游戏

3.2 构建你的第一个 Android 应用程序

现在是时候编写你的第一个 Android 应用程序了，你可以从一个简单的“Hello World”工程开始，并以此为基础，逐步扩展，掌握 Android 的某些特性。

Tip

本章提供的大部分代码均来源于 `MyFirstAndroidApp` 应用程序，本书的网站上提供了此程序的源代码下载。

3.2.1 创建和配置新的 Android 工程

创建一个新的 Android 工程在很大程度上与向 Eclipse 环境导入 Snake 应用程序是相同的。

Download at http://www.pin5i.com/

你需要做的第一件事情是在 Eclipse 环境中创建一个新工程。Android 工程向导为 Android 应用程序创建了所有需要的文件。在 Eclipse 中，你可以按照以下步骤创建一个新的工程。

① 选择 File→New→Android Project，或者点击 Eclipse 工具条中的创建图标。

② 为你的工程取一个名字。在这里，我们将新的工程命名为 `MyFirstAndroidApp`。

③ 为你的工程文件选取一个路径。由于这是一个新的工程，所以选择工作空间中的 Create New Project 单选按钮。勾选 Use Default Location 复选框，或者选择任意你想要存储源文件的路径。

④ 为你的应用程序选择一个构建目标。选择你与你手里拥有的设备兼容的目标，例如 Android 2.2。

⑤ 选择一个应用程序名称。这个名称将是应用程序的“用户友好”名称，它将显示在应用程序的启动目录中。这里，我们将 Application Name 指定为 My First Android App。

⑥ 选择一个包名称。这里你需要遵照传统的 Java 标准包命名规范。因为本书的所有示例代码都将定位在 `com.androidbook.*`命名空间下，所以我们将这个包命名为 `com.androidbook.myfirstandroidapp`，不过你也可以任意指定你想要的包名称。

⑦ 勾选 Create Activity 复选框。这将指示向导为你的应用程序创建一个默认的启动 Activity。将这个 `Activity` 类命名为 `MyFirstAndroidAppActivity`。

现在，你的工程设置应该大致如图 3.8 所示。

⑧ 最后，单击 Finish 按钮。

图 3.8 使用 Android 工程向导配置 My First Android App

3.2.2 Android 应用程序的核心文件和目录

每一个 Android 应用程序都拥有一系列核心文件，它们用于定义应用程序的功能（见表 3.1）。表 3.1 中的文件是每一个新的 Android 应用程序所默认创建的。

表 3.1 重要的 Android 工程文件及目录

Android 文件	描 述
`AndroidManifest.xml`	全局应用程序描述文件。它定义了应用程序的能力（capability）和权限，以及运行方式
`default.properties`	自动创建的工程文件。它定义了应用程序的构建（build）目标及其他所需的构建选项
`src` 文件夹	应用程序源代码所在的文件夹，该文件夹为必需的
`src/com.androidbook.myfirstandroidapp/MyFirstAndroidApp.java`	定义了 Android 应用程序入口的核心源文件
`gen` 文件夹	应用程序自动生成的资源文件所在的文件夹，该文件夹为必需的
`gen/com.androidbook.myfirstandroidapp/R.java`	应用程序资源管理源文件，它不能被编辑
`res` 文件夹	所有应用程序资源所在的文件夹，该文件夹为必需的。应用程序资源包括动画、可绘图像组件、布局文件、XML 文件、数据资源（如字符串）和原始（raw）文件

Download at http://www.pin5i.com/

续表

Android 文件	描　　述
res/drawable-*/icon.png	用于存储不同分辨率的应用程序图标的资源文件夹
res/layout/main.xml	主屏幕布局文件
res/values/strings.xml	应用程序字符串资源
assets 文件夹	用于存储应用程序组件（assest）的文件夹。应用程序组件为不以应用程序资源的方式管理的应用程序数据（文件、文件夹）

在 Eclipse 工作空间中还保存了其他许多文件，它们是 Eclipse 工程的组成部分。表 3.1 中所列出的这些文件都是在一个普通工程中将会用到的重要文件。

3.2.3 为新工程创建 Android 虚拟设备

下一步，我们需要做的是为应用程序的运行创建一个 AVD，这一 AVD 将描述你所期望的设备类型。在这个例子中，可以使用我们为 Snack 应用程序创建的 AVD。一个 AVD 描述的是一种设备，而不是一个应用程序，所以，你可以将同一个 AVD 应用于不同的应用程序中。同样，你也可以创建若干个配置相似，只有某些参数不同的 AVD（例如，其中安装了不同的应用程序，或是 SD 卡中有不同的内容）。

Note

获取更多关于创建不同种类 AVD 或是使用 Android 模拟器的信息，请参考附录 A “Android 模拟器快速入门”。

3.2.4 为工程配置运行参数

接下来，你需要在 Eclipse 中创建运行和调试参数，以配置 MyFirstAndroidApp 的构建和运行环境。运行参数用于设置所使用的模拟器选项和应用程序的入口点。

你可以使用不同的选项分别创建 Run Configuration（运行配置参数）和 Debug Configuration（调试配置参数）。我们先从为应用程序创建 Run Configuration 开始。

下面给出了为 MyFirstAndroidApp 创建一个基本的 Run Configuration 所需的步骤。

① 执行 Run→Run Configurations（或者右键单击 Project，选择 Run As）。

② 双击 Android Application。

③ 将 Run Configuration 命名为 MyFirstAndroidAppRunConfig。

④ 单击 Browse 按钮，选中 MyFirstAndroidApp 工程。

⑤ 切换至 Target 标签，将 Device Target Selection Mode 设置为 Manual。

Tip

当你在设置运行或调试配置参数时，如果将 Device Target Selection Mode 置为默认的 Automatic，那么你的应用程序将自动安装和运行在与计算机连接的设备上，否则，应用程序将使用指定的 AVD 在模拟器中运行。当选择 Manual 后，你可以在以下三个选项中作出选择：①使用已有的模拟器运行应用程序；②使用新的模拟器实例运行程序，并且允许重新设定 AVD；③在已联机的设备上运行程序（若设备已连接）。如果在模拟器运行时插入设备，即使当前模式设为 Automatic，你也可以在上面三个选项中做出选择。

Download at http://www.pin5i.com/

现在我们一起来为这个应用程序创建 Debug Configuration。这个过程和创建 Run Configuration 很相似。下面给出了为 MyFirstAndroidApp 创建一个基本的运行配置参数所需的步骤。

① 执行 Run→Debug Configurations（或者右键单击 Project，选择 Debug As）。

② 双击 Android Application。

③ 将 Debug Configuration 命名为 MyFirstAndroidAppDebugConfig。

④ 单击 Browse 按钮，选中 MyFirstAndroidApp 工程。

⑤ 切换至 Target 标签页，将 Device Target Selection Mode 设置为 Manual。

⑥ 单击 Apply 按钮，然后单击 Close 按钮。

现在，你已经为应用程序配置好了运行参数。

3.2.5 在模拟器中运行 Android 应用程序

现在你可以按照下面的步骤运行 MyFirstAndroidApp 应用程序。

① 点击工具栏中的 Run As 图标下拉菜单（带三角的绿色圆圈，类似于“播放”按钮）。

② 将下拉菜单展开，并且选择你创建的 Run Configuration。（如果它没有被列出，那么点击“Run Configurations...”项并且选择适当的配置。这样，在你下一次运行配置时，它将出现在下拉菜单中。）

③ 因为你选择了 Manual Target Selection 模式，现在你将被提示选择模拟器实例。修改选择项以启动一个新的实例，并且勾选你刚刚创建的 AVD，如图 3.9 所示。

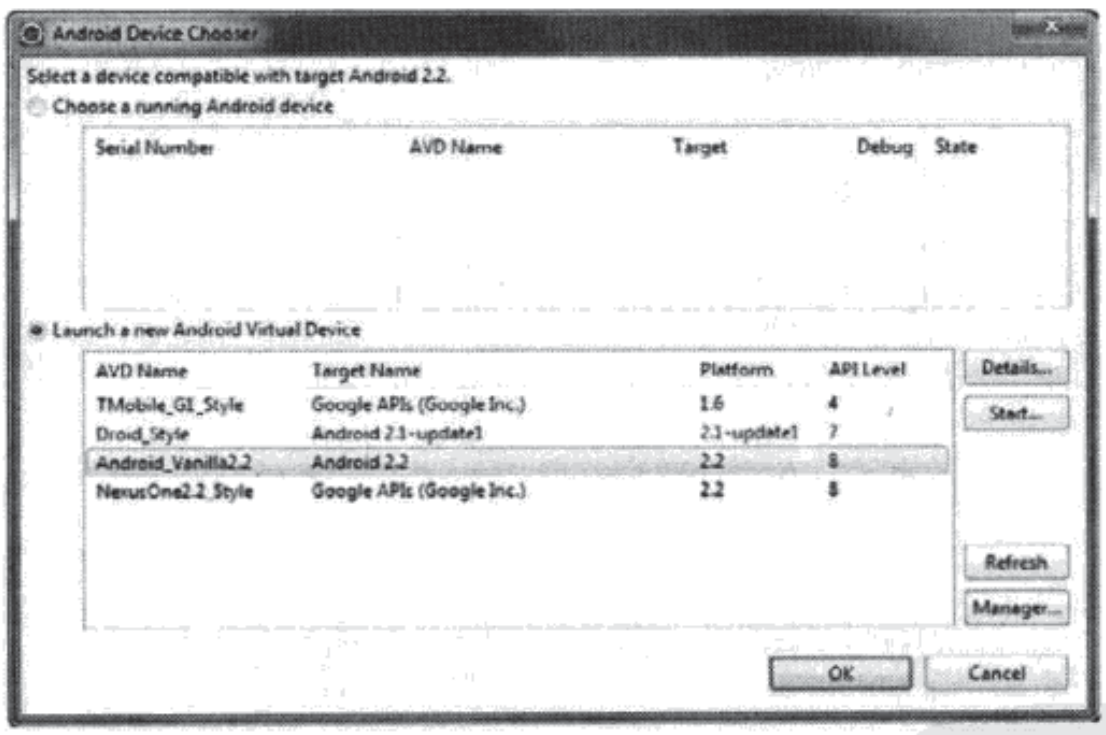


图 3.9 手动选择一个 Target Selection 模式

④ Android 模拟器开始启动，这一过程可能需要持续一段时间。

Tip

模拟器的启动可能需要耗费相当长的时间，所以你也许会考虑在工作时将它打开并放在一边，需要时再重新连接，以节省启动所需的时间。Eclipse 中的某些工具可以处理程序的重安装和重运行，所以你只需要简单地保持模拟器始终处于运行状态即可。

⑤ 点击 Menu 按钮解锁模拟器。

⑥ 应用程序启动，如图 3.10 所示。

⑦ 单击模拟器中的 Home 按钮终止应用程序。

Download at http://www.pin5i.com/

⑧ 将应用程序 Drawer 拉起来，这样可以看见所有已安装的应用程序。此时你的模拟器屏幕应该类似于图 3.11。

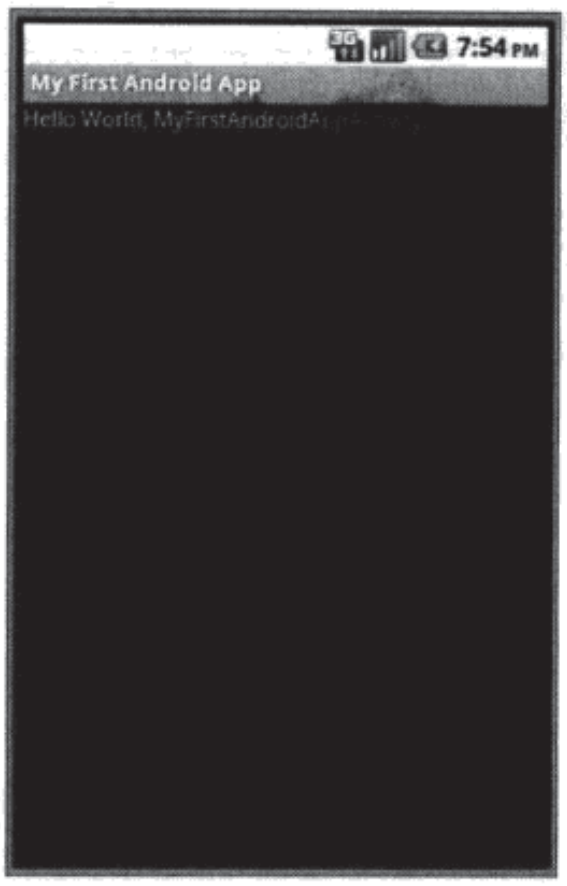


图 3.10 模拟器中运行的 My First Android App 应用程序

图 3.11 在 Drawer 中显示的 My First Android App 应用程序图标

⑨ 单击 My First Android Application 图标再次运行程序。

3.2.6 在模拟器中调试 Android 应用程序

在继续深入学习之前，你还需要熟悉如何在模拟器中调试程序。为了示范一些有用的调试工具的使用，我们首先在 My First Android Application 中制造一个错误。

在你的工程中，编辑文件 `MyFirstAndroidApp.java`，在你的类中创建一个名为 `forceError()` 的新方法，并且在类的 `onCreate()` 方法中调用它。`forceError()` 方法将在应用程序中产生一个新的未处理的错误。

`forceError()` 方法的代码应该大致如下所示：

```
public void forceError() {
    if(true) {
        throw new Error("Whoops");
    }
}
```

Tip

Eclipse 拥有针对代码编写和调试的不同视图（perspective）（由不同的特定面板组成）。你可以通过选择 Eclipse 环境右上角的适当名称来进行视图间的切换。Java 视图安排了合适的面板以进行代码编写和工程内部导航。调试（Debug）视图允许你设置断点，查看 LogCat 记录的信息，并且进行调试。Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）视图则允许你监视及操控模拟器和设备的状态。

此时运行应用程序并且观察将会发生什么事情也许对掌握调试技术是非常有用的。我们首先使用 Run Configuration。在模拟器中，你将会发现应用程序异常终止了。一个对话框将

Download at http://www.pin5i.com/

弹出并允许你强制关闭应用程序，如图 3.12 所示。

现在关闭应用程序和模拟器。我们可以参照以下步骤调试 MyFirstAndroidApp 应用程序。

① 单击工具条上的 Debug As 图标下拉菜单（带有下拉箭头的绿色小虫子）。

② 选择你创建的 Debug Configuration。（如果它没有被列出，那么点击“Debug Configurations...”项并且选择适当的配置。这样，在你下一次运行配置时，它将出现在下拉菜单中。）

③ 之后的步骤与 Run Configuration 一样，选择适当的 AVD，再次运行模拟器，必要时解锁屏幕。

之后将耗费一定的时间启动模拟器并且关联调试器（debugger）。如果这是你第一次调试 Android 应用程序，那么其间你需要在弹出的几个对话框中点选相应的按钮以继续调试的进行，其中之一如图 3.13 所示。

图 3.12 My First Android App 应用程序的“优雅崩溃”

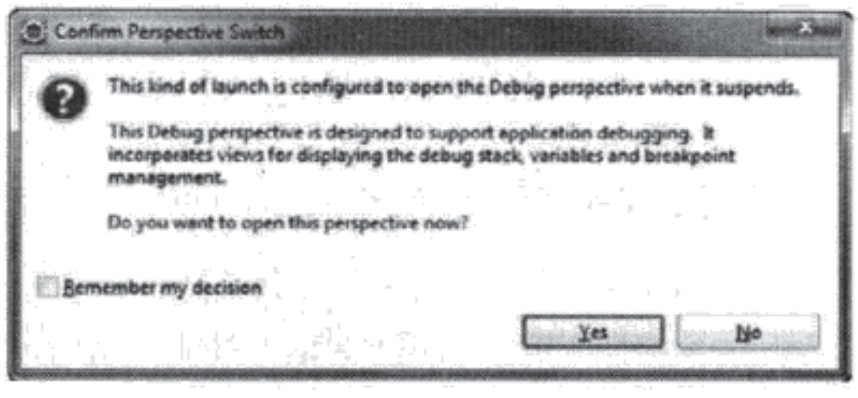


图 3.13 为使用 Android 模拟器调试而切换至调试视图

在 Eclipse 中，我们使用调试视图来设置程序断点、单步执行代码并且查看 LogCat 记录的应用程序的有关日志信息。这一次，当应用程序出错时，你可以使用调试器找到错误的原因。在 Eclipse 中，你可能需要通过几个对话框来完成调试的设置。如果你允许应用程序在抛出异常后仍然继续运行，那么你可以在 Eclipse 的 Debug 视图中查看结果。如果你查看 LogCat 日志面板，你将会发现你的应用程序由于一个未处理的异常而被强制终止（如图 3.14 所示）。

具体地讲，你将看见一个红色的 AndroidRuntime 错误：`java.lang.Error: Whoops`。

回到模拟器中，单击 Force Close 按钮。现在，在 `forceError()` 方法代码行左侧单击右键并选择 Toggle Breakpoint（或者按下 Ctrl+Shift+B 快捷键）以设置一个断点。

Tip

在 Eclipse 中，你可以使用 Step Into（F5）、Step Over（F6）、Step Return（F7）和 Resume（F8）来进行单步调试。

在 Mac OS X 平台下，你也许会发现 F8 键已被用于全局映射。如果你想要方便地使用快捷键命令，那么你可以修改 Eclipse 的快捷键映射：选择 Eclipse→Preferences→General→Keys，找到 Resume 项并且修改其快捷键。另外，你也可以修改 Mac OS X 的全局键映射：进入 System Preferences→Keyboard 和 Mouse→Keyboard Shortcuts，然后修改 F8 键映射为其他的按键。

Download at http://www.pin5i.com/

图 3.14 在 Eclipse 中调试 My Firstr Android App

在模拟器中，重新启动你的应用程序并且单步运行代码。你将会看见 MyFirstAndroidApp 抛出一个异常，并且该异常显示在调试视图的 Variable Browser 面板中。展开变量的细节将会看见这是一个“Whoops”错误。

这是一个很好的机会，你可以重复这一让程序崩溃的过程，以此来熟悉对应用程序调试选项的控制。然后，切换到 DDMS 视图，你将注意到模拟器拥有一个手机上运行的进程的列表，如 system_process 和 com.android.phone。如果你运行 MyFirstAndroidApp，那么你将看见 com.androidbook.myfirstandroidapp 作为一个进程显示在模拟器的列表中。当应用程序崩溃并被强制关闭时，你会发现这一项从进程列表中消失了。你可以使用 DDMS 终止进程、查看线程和堆，以及访问手机的文件系统。

3.2.7 为 Android 应用程序添加日志记录

在你深入了解 Android SDK 的各个特性之前，首先需要熟悉日志记录，它是一个用于调试和学习 Android 的重要资源。Android 的日志记录特性存在于 android.util 包的 Log 类中。

表 3.2 给出了 android.util.Log 类提供的一些实用方法。

表 3.2 android.util.Log 类的一些方法

方　法	说　明
Log.e()	记录错误
Log.w()	记录警告
Log.i()	记录信息类的消息
Log.d()	记录 Debug（调试）消息
Log.v()	记录 Verbose 消息

要为 MyFirstAndroidApp 应用程序添加日志支持，可以编辑 MyFirstAndroidApp.java 文件。首先，你需要为使用 Log 类添加适当的 import 表达式。

```
import android.util.Log;
```

Download at http://www.pin5i.com/

Tip

在 Eclipse 中，为了节省时间，你可以通过在类名上悬停并选择“Add Imported Class”项来导入并使用类。

你也可以使用“组织（Organize）”导入命令（Windows 系统：Ctrl+Shift+O；Mac 系统：Command+Shift+O）来让 Eclipse 自动组织你的导入。它将移除未使用的导入同时添加已使用但未导入的包。如果存在命名冲突，就像使用 Log 类时经常遇见的那样，你可以做出选择。

下一步，在 MyFirstAndroidApp 类中，声明一个字符串常量，它将用于表明这个类产生的所有日志消息。你可以使用 Eclipse 的 LogCat 特性来过滤出你的日志消息：

```
private static final String DEBUG_TAG= "MyFirstAppLogging";
```

现在，在 onCreate()方法中，你可以记录下一些消息：

```
Log.i(DEBUG_TAG, "Info about MyFirstAndroidApp");
```

Warning

此时，不要忘记将之前的 forceError()注释掉，这样程序才不会崩溃。

现在，你已经可以运行 MyFirstAndroidApp 了，保存工程，然后在模拟器中调试。你会注意到你输入的日志消息将会出现在 LogCat 的列表中，它们将带有标签字段 MyFirstAppLogging（如图 3.15 所示）。

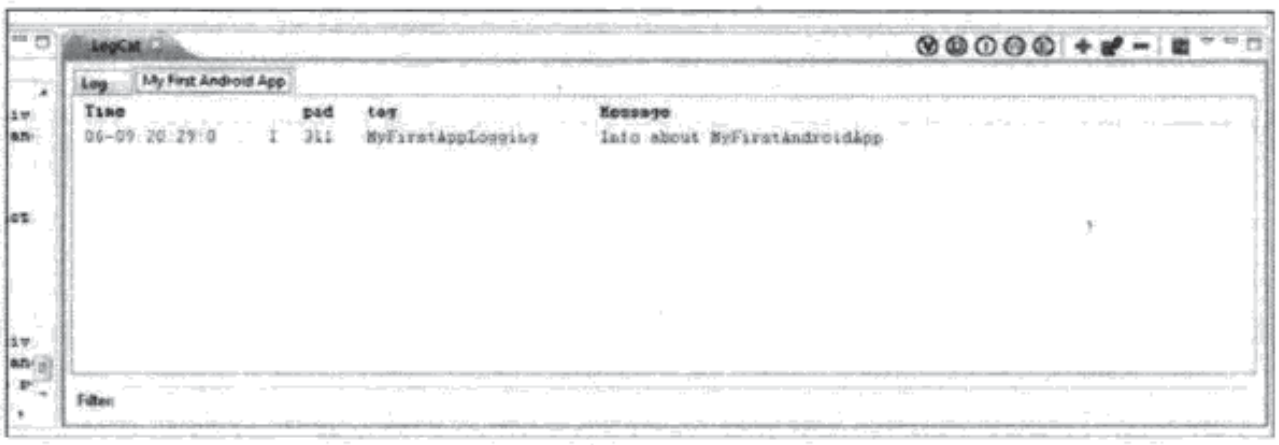


图 3.15 My First Android App 的一个已过滤的 LogCat 日志记录

Tip

也许你想要创建 LogCat 过滤器以保证仅显示含有特定标签的消息。要实现这一功能，在 Eclipse 的 LogCat 面板中单击绿色的加号按钮。将过滤器命名为 Just MyFirstApp 并且填充 Log Tag 为 MyFirstAppLogging。此时，你拥有第二个 LogCat 标签，其中仅含有你特定的日志信息。

3.2.8 为应用程序添加媒体支持

下一步，我们为 MyFirstAndroidApp 应用程序添加一些酷炫的功能：让它播放一个 MP3 音乐文件。Android 媒体播放器特性包含于 android.media 包的 MediaPlayer 类中。

你既可以从现存的应用程序资源中创建 MediaPlayer 对象，也可以使用统一资源标识符（Uniform Resource Identifier，URI）指定某个特定的目标文件。为了简便起见，我们从使用 android.net 包中的 Uri 类访问一个 MP3 开始。

Download at http://www.pin5i.com/

表 3.3 中列出了 android.media.MediaPlayer 和 android.net.Uri 类的某些方法。

表 3.3 重要的 MediaPlayer 和 URI 解析方法

方 法	说 明
MediaPlayer.create()	为播放指定的目标创建新的媒体播放器
MediaPlayer.start()	开始播放媒体
MediaPlayer.stop()	停止播放媒体
MediaPlayer.release()	释放媒体播放器对象资源
Uri.parse()	从适当的格式化 URI 地址中实例化一个 Uri 对象

要在 MyFirstAndroidApp 中加入 MP3 播放支持，需要编辑文件 MyFirstAndroidApp.java。首先，你需要为 MediaPlayer 类添加适当的 import 表达式。

```
import android.media.MediaPlayer;
import android.net.Uri;
```

下一步，在 MyFirstAndroidApp 类中，为你的 MediaPlayer 对象声明一个成员变量。

```
private MediaPlayer mp;
```

现在，在你的类中创建一个名为 playMusicFromWeb() 的函数，并且在 onCreate() 方法中调用这一函数。playMusicFromWeb() 方法将创建一个有效的 Uri 对象、一个 MediaPlayer 对象并且开始播放 MP3。如果这一操作由于某些原因而失败，方法将使用你自定义的日志标签记录一个自定义的错误日志。

playMusicFromWeb() 函数的代码应该大致如下所示：

```
public void playMusicFromWeb() {
    try {
        Uri file = Uri.parse("http://www.perlgurl.org/podcast/archives "
            + "/podcasts/PerlgurlPromo.mp3");
        mp = MediaPlayer.create(this, file);
        mp.start();
    }
    catch (Exception e) {
        Log.e(DEBUG_TAG, "Player failed", e);
    }
}
```

最后，你需要在应用程序运行结束后“干净地”退出。实现“干净”退出，你需要覆写 onStop() 方法，停止 MediaPlayer 对象的播放并且释放其资源。

Tip

在 Eclipse 中，你可以在类名上单击鼠标右键，选择 Source（或按下 Alt+Shift+S 快捷键）。执行 Override/Implement Methods 菜单项，然后勾选 onStop() 方法。

onStop() 方法的代码应该大致如下所示：

```
protected void onStop() {
    if (mp != null) {
        mp.stop();
        mp.release();
```

Download at http://www.pin5i.com/

```
        }
        super.onStop();
    }
```

现在，如果你在模拟器中运行 MyFirstAndroidApp（并且你拥有因特网连接以根据 URI 在线获取数据），你的应用程序将会播放一段 MP3。当关闭应用程序后，MediaPlayer 会停止播放并适当地释放资源。

3.2.9 为应用程序添加位置服务（LBS）

虽然你的应用程序已经能够“开口说话”了，但它却不知道自己“身处何方”。现在，我们来熟悉一些简单的获得 GPS 坐标的方法。

1. 使用 Google API 创建 AVD

如果想要在应用程序中将位置服务或地图集成进去，你将会用到 Android 手机上通常都会内置的一些 Google 应用程序，其中最常用的就是 Google Maps。所以，你需要创建一个新的 AVD，这个 AVD 的配置应当和 Android_Vanilla2.2 基本一致，只有一点不同：它的目标 API Level（在这里是 8）应该和 Google API 相同。你可以将这个 AVD 命名为 Android_with_GoogleAPIs_2.2。

2. 配置模拟器的位置

在你创建了一个有 Google API 支持的新 AVD 以后，你需要关闭你正在运行的模拟器，然后再次调试 My First Android App，这一次，选用新创建的 AVD。

模拟器上并没有位置感应器，所以你要做的第一件事情就是把模拟器和 GPS 坐标相绑定。在调试模式下使用一个支持 Google Maps 插件的 AVD 运行模拟器，然后按照以下步骤进行操作。

（1）在模拟器上

① 单击 Home 按钮返回主界面。

② 从应用程序 Drawer 中选择 Maps 应用程序。

③ 单击 Menu 按钮。

④ 执行 My Location 菜单项。

（2）在 Eclipse 中

① 单击 Eclipse 右上角的 DDMS 视图。

② 在屏幕左侧你将看见一个 Emulator Control 面板，下拉至 Location Control 项。

③ 手动输入你所在地的经纬度坐标。（注意它们的顺序是反的。）

④ 单击 Send 按钮。

Tip

要想获得某个特定位置的经纬度坐标信息，可以访问 http://maps.google.com。导航至你想要到达的地点，右键单击地图，选择“在此位置居中放置地图”。单击“分享链接”，并将 URL 复制下来。仔细观察这一 URL 地址，找出“ll”变量，它指示了这一位置的纬/经度。例如，“Yosemite Valley”链接的值为 ll = 37.746761,−119.588542，它表示该点的纬度为 37.746761，经度为−119.588542。

Download at http://www.pin5i.com/

回到模拟器，你将发现 Google Maps 现在显示的是你所绑定的位置，屏幕上应该像图 3.16 所示那样显示某个位置的地图，比如 Yosemite Valley。

现在，你的模拟器有了一个假想的坐标位置。

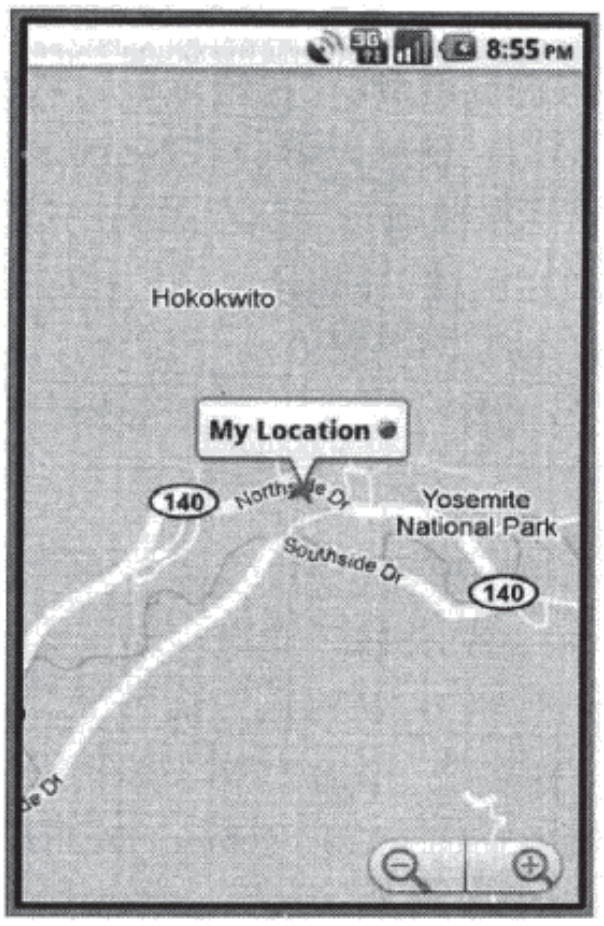


图 3.16 将模拟器位置设定至 Yosemite Valley

3. 查找上次记录位置

要将位置支持加入到 MyFirstAndroidApp 中，需要编辑文件 MyFirstAndroidApp.java。首先，你需要添加适当的 import 表达式。

```
import android.location.Location;
import android.location.LocationManager;
```

现在，在你的类中创建一个名为 getLocation()的新方法，然后在 onCreate()方法中调用它。getLocation()方法可以获取最后一次已知的手机位置，并将其以信息类消息的形式记录成日志文件。如果该操作由于某种原因而失败，那么该方法将会以日志的形式记录下一个错误。

getLocation()方法的代码应该大致如下所示：

```
public void getLocation() {
    try {
        LocationManager locMgr = (LocationManeger)
            getSystemService(LOCATION_SERVICE);
        Location recentLoc = locMgr.
            getLastKnownLocation(LocationManager.GPS_PROVIDER);
        Log.i(DEBUG_TAG, "loc: " + recentLoc.toString());
    }
    catch (Exception e) {
        Log.e(DEBUG_TAG, "Location failed", e);
    }
}
```

最后，你的应用程序需要一个特殊的许可权限以访问位置服务功能，为此，你需要在 AndroidManifest.xml 文件中注册这一许可。要将位置服务许可添加至你的应用程序中，按照以下步骤进行操作。

① 双击 AndroidManifest.xml 文件。

② 切换至 Permissions 标签页。

③ 单击 Add 按钮，然后选择 Uses Permission。

④ 在右侧的面板中，选择 android.permission.ACCESS_FINE_LOCATION。

⑤ 保存文件。

现在，如果你在模拟器上运行 MyFirstAndroidApp 应用程序，它将以信息类日志的方式记录下你提供给模拟器的 GPS 位置坐标，这些信息可以在 Eclipse 的 LogCat 面板中查看。

3.2.10 在硬件上调试应用程序

现在你已经掌握了在模拟器上运行应用程序的方法。接下来，让我们把应用程序安装到

Download at http://www.pin5i.com/

真正的硬件设备中。首先，你需要在 AndroidManifest.xml 文件中将你的应用程序注册为 Debuggable（可调试），按照以下步骤进行操作：

① 双击 AndroidManifest.xml 文件。

② 切换至 Application 标签页。

③ 将 Debuggable Appliation Attribute 属性设为 True。

④ 保存文件。

当然，你也可以通过直接修改 android:debuggable 属性来更改 AndroidManifest.xml 文件的 application 元素，如下所示：

```
<application ... android:debuggable="true">
```

如果你忘了将 debuggable 属性设定为 true，手机将显示一个对话框，以等待调试器建立同手机的连接，除非你选择了 Force Close 并且更新了 Manifest 文件。

现在，通过 USB 将你的 Android 设备同计算机连接起来，并且重新运行应用程序的 Run Configuration 或 Debug Configuration。由于你在配置中选择了 Manual（手动），此时你将看见在 Android Device Chooser 对话框（如图 3.17 所示）中，你的真实 Android 设备作为一个选项被列出来了。

选择列出的 Android 设备作为目标，然后你将看到 My First Android App 应用程序被载入到 Android 手机上，并且像之前一样被运行。由于你之前开启了手机的开发调试选项，所以此时你也可以在这里调试应用程序。你会发现手机使用 USB 调试连接的状态是开启的，因为在通知栏上有一个 Android 机器人小图标。图 3.18 是运行在真实手机上的应用程序的截屏。

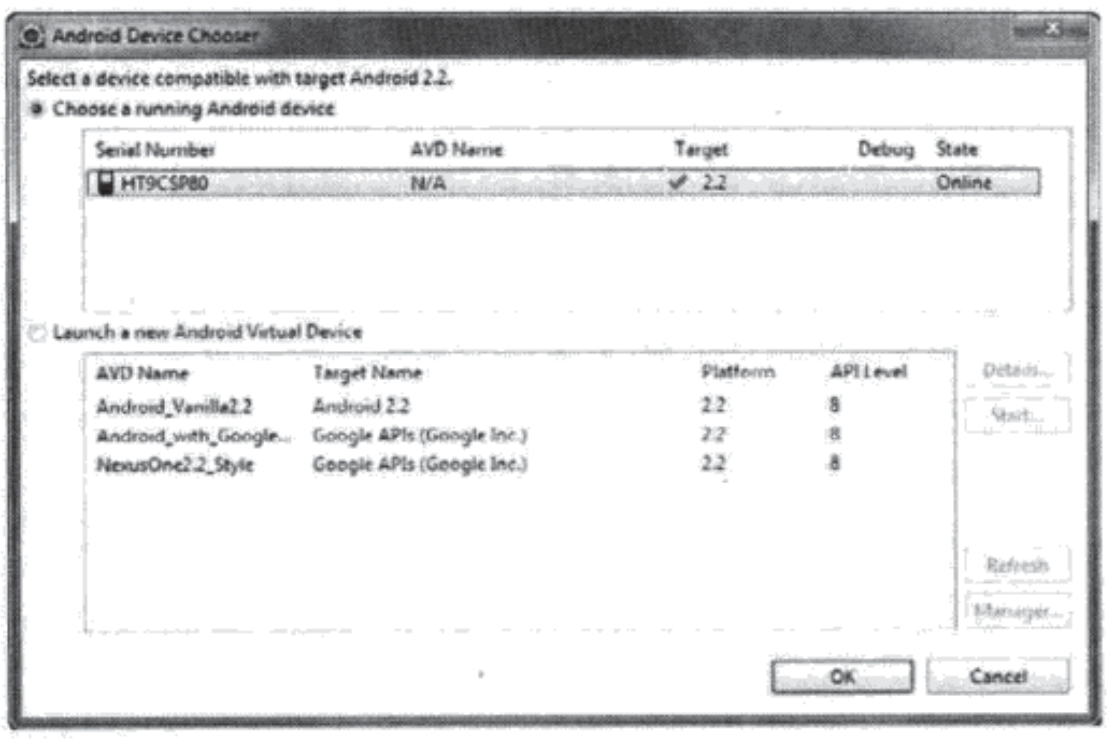


图 3.17 带有已通过 USB 连接的 Android 设备的 Android Device Chooser 对话框

图 3.18 运行在 Android 硬件设备上的 My First Android App

在真实设备和在模拟器上的调试过程大致相同，但也会有少许例外。比如，你无法使用模拟器空间来发送短消息或者获取当前所在位置信息，但却可以在真实设备上实现这些操作

Download at http://www.pin5i.com/

（真实的短消息和真实的位置信息）。

小　结

本章讲解了如何在 Eclipse 中添加、构建、运行并调试一个 Android 应用程序，从使用一个 Android SDK 中提供的示例程序测试开发环境出发，然后使用 Eclipse 创建了一个崭新的应用程序。同时，你还学习了如何快速地对程序进行改动，提前接触到了后续章节中将会学习的 Android 功能。

在后面的几章中，你将学习到更好地使用 Manifest 文件定义 Android 应用程序的方法，以及有关应用程序生存周期的运作方式等问题。你还将学习到如何组织应用程序资源以便更好地运用它们，这些资源包括图像、字符串等。

参考资料和更多信息

Windows 操作系统的 USB 驱动．http://developer.android.com/sdk/win-usb.html.

Android 开发指南（Dev Guide）．Developing on a Device http://developer.android.com/guide/developing/device.html.

Download at http://www.pin5i.com/

Download at http://www.pin5i.com/

第二部分

Android应用程序设计精髓

第4章 剖析 Android 应用程序

第5章 使用 Android Manifest 文件定义应用程序

第6章 管理应用程序资源

Download at http://www.pin5i.com/

第4章

剖析 Android 应用程序

传统的计算机科学领域通常根据功能和数据来定义一段程序，Android 应用程序也不例外。它们执行任务、在屏幕上显示信息，并且根据不同的资源对数据进行处理。

在资源受限的情况下为移动设备开发 Android 应用程序需要你对应用程序的生命周期有一个彻底的了解。Android 也使用自己专门的术语来描述这些组成应用程序的各种“零件”，例如 Context、Activity 和 Intent。本章将带领你熟悉 Android 应用程序最重要的组成部分。

4.1 掌握重要的 Android 术语

本章将向你介绍 Android 应用程序开发过程中所需使用的术语，并且为你构建起对 Android 应用程序功能和相互间交互过程的全面认识。本章所涉及的重要术语如下所示。

- **Context**：Context 是 Android 应用程序的中央控制中心（central command center）。所有应用程序特有的功能都可以通过 Context 进行访问。
- **Activity**：一个 Android 应用程序是由若干个任务（task）组成的，每一个任务被称为 Activity。应用程序中的任何一个 Activity 都有着自己唯一的任务或目的。
- **Intent（意图）**：Android 操作系统使用异步消息传送机制（asynchronous messaging mechanism）来使得任务请求与适当的 Activity 相匹配。每一个请求都被打包成一个 Intent。你可以把这种请求想象成是在陈述需要做某件事情的应用程序意图。
- **Service（服务）**：不需要用户交互的任务可以被封装成服务。如果某个操作比较冗长（占用时间过长），或者需要周期性地执行（例如查看服务器有无新邮件），那么使用服务是最合适的。

4.2 使用应用程序 Context

应用程序 Context 居于所有顶层(top-level)应用程序功能的中心位置(central location)。Context 类可以用来管理应用程序特定（application-specific）的配置细节，以及应用程序范围（application-wide）的操作和数据。使用应用程序 Context 来访问设置和资源可以在多个 Activity 实例中共享。

1. 获取应用程序 Context

你可以使用 getApplicationContext()方法从当前进程中获取 Context，如下所示：

Download at http://www.pin5i.com/

```
Context context = getApplicationContext();
```

2. 使用应用程序 Context

在获得了一个可用的应用程序 Context 后，你就可以使用它来访问应用程序范围的功能和服务了。

（1）获取应用程序资源

你可以使用应用程序 Context 的 getResources()方法来获取应用程序的资源。获取资源最直接的方式是使用资源标识符（ID），它是一个在 R.java 类中自动生成的数字。下面的示例通过标识符从应用程序资源中获得了一个 String 实例：

```
String greeting = getResources().getString(R.string.hello);
```

我们将在第 6 章“管理应用程序资源”中讨论更多有关应用程序资源的话题。

（2）访问应用程序首选项

你可以使用应用程序 Context 的 getSharedPreferences()方法访问共享的应用程序首选项（preferences）。SharedPreferences 类可以用来存储简单的应用程序数据，例如配置设置。

我们将在第 10 章“使用 Android 数据和存储 API”中讨论更多有关应用程序首选项的话题。

（3）使用 Context 访问其他应用程序的功能

应用程序 Context 提供对其他顶层（top-level）应用程序功能的访问。你还可以使用应用程序 Context 完成以下操作：

① 运行 Activity 实例；

② 获取应用程序所带的组件（asset）；

③ 获取一个系统服务（例如：位置服务）；

④ 管理私有的应用程序文件、目录和数据库；

⑤ 检查和使用应用程序权限。

上述列表的第一项（运行 Activity 实例）或许是你最常使用应用程序 Context 的理由。

Warning

由于 Activity 类继承自 Context 类，所以有时你也可以使用它来替代对应用程序 Context 的显式获取。然而，也不要在所有情况下都使用 Activity Context，因为这样可能会导致内存泄漏。你可以在如下网址找到一篇有关这一内容的非常精彩的文章：http://android-developers.blogspot.com/2009/01/avoiding-memory-leaks.html。

4.3 使用 Activity 执行应用程序任务

对任何一个 Android 应用程序来讲，Activity 类（android.app.Activity）都是其核心组成部分。许多时候，在应用程序中，你将会为每一个屏幕显示定义和实现一个 Activity。例如，一个简单的游戏应用程序可能会包含以下 5 个 Activity，如图 4.1 所示。

Download at http://www.pin5i.com/

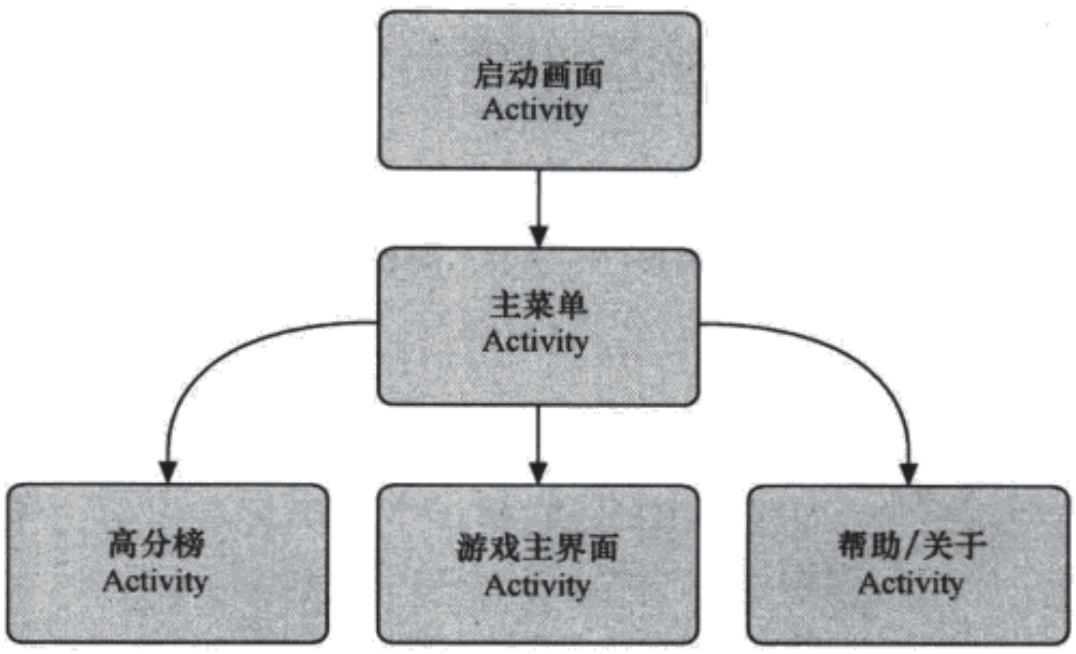


图 4.1 拥有 5 个 Activity 的简单游戏

- **启动画面**：这一 Activity 充当应用程序的主入口点。它显示应用程序的名称和版本信息，并且在一小段时间后跳转至主菜单界面。
- **主菜单界面**：用户在该 Activity 中可以实现向应用程序中的其他 Activity 跳转。此处，用户需要做出选择，告诉应用程序他们下一步想做什么。
- **游戏主界面**：游戏的核心部分在该 Activity 中进行。
- **高分榜**：游戏的得分情况或设置参数可以在该 Activity 中显示。
- **帮助/关于界面**：该 Activity 可以用于显示指导用户顺利操作游戏的信息。

4.3.1 Android Activity 的生命周期

Android 应用程序可以是多线程的，并且在内存和处理器性能支持的情况下，Android 操作系统允许同时运行多个应用程序。应用程序可以拥有后台进程，并且可以被来电等事件中断。然而，在同一个时间只能有一个活动的应用程序对用户可见，具体地讲，在任意给定时间，只能有一个应用程序的 Activity 处于前台。

Android 操作系统跟踪所有运行的 Activity 对象，它将这些对象放进一个 Activity 栈中（如图 4.2 所示）。当一个新的 Activity 启动时，处于栈顶的 Activity（即当前的前台 Activity）将会暂停，而这个新的 Activity 则被放入栈顶。当这一新的 Activity 完成时，它从栈中被移除，而先前的 Activity 则又回到栈顶。

图 4.2 Activity 栈

Download at http://www.pin5i.com/

Android 应用程序负责管理它们自身的状态、内存、资源和数据。它们必须能够平滑地暂停或恢复运行。理解 Activity 生命周期所包含的不同状态是设计和开发健壮的 Android 应用程序的先决条件。

1. 使用 Activity 回调管理应用程序状态和资源

Activity 在生命周期内重要的状态改变伴随着一系列重要的方法回调。这些回调方法如图 4.3 所示。

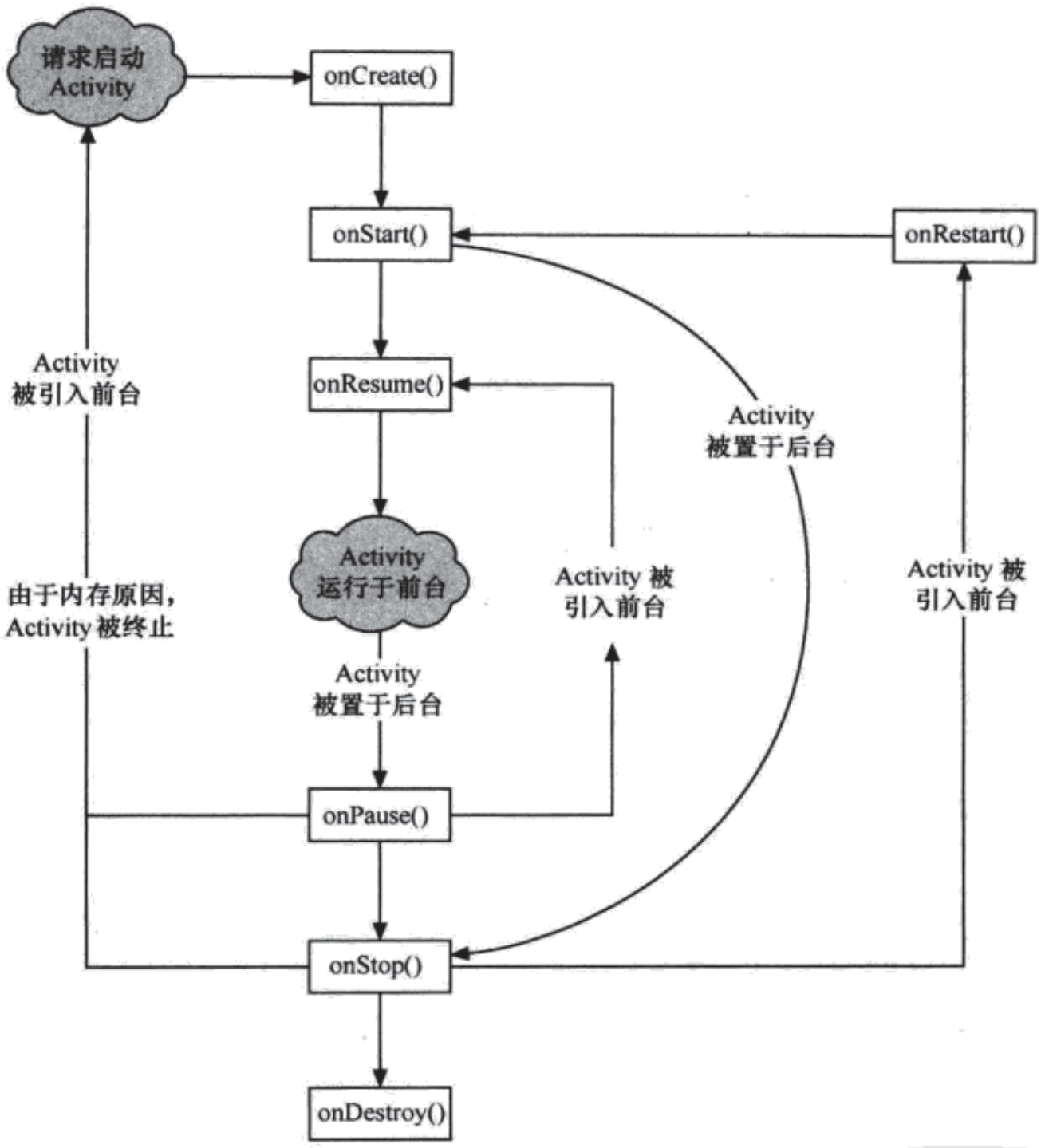


图 4.3 一个 Android Activity 的生命周期

下面是 Activity 类的一些重要的回调方法：

```
public class MyActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState);
    protected void onStart();
    protected void onRestart();
    protected void onResume();
    protected void onPause();
    protected void onStop();
    protected void onDestroy();
}
```

现在，我们逐一介绍这些回调方法：它们何时被调用，以及调用它们的目的是什么。

Download at http://www.pin5i.com/

（1）在 onCreate()中初始化静态 Activity 数据

当一个 Activity 首次运行时，onCreate()方法被调用。onCreate()方法仅拥有一个传入参数，即一个 Bundle 对象，但对于首次启动的 Activity 而言，这一参数为空。如果这一 Activity 先前因为内存的原因而被终止（killed），现在又需要重新启动，那么 Bundle 对象中将包含该 Activity 先前的状态信息，这样，它才能够被重新初始化。对于 onCreate()方法而言，在其中进行一些对于布局或数据绑定之类的设置是非常合适的，这将包含对 setContentView()方法的调用。

（2）在 onResume()中初始化和接收 Activity 数据

当这个 Activity 到达 Activity 栈的顶部，即成为前台进程时，onResume()方法被调用。虽然此时这一 Activity 可能还不为用户所见，但这里仍然是获取 Activity 运行所需资源（专有或其他）的最佳“场所”。通常地，这些都是进程密集型（process-intensive）资源，所以我们仅在 Activity 处于前台时保留它们。

Tip

onResume()方法非常适合用于启动音频、视频和动画。

（3）在 onPause()中停止、保存和释放 Activity 数据

当另一个 Activity 移至 Activity 栈的顶部时，当前的 Activity 将会接收到它已在栈中被下移的通知，同时 onPause()方法被调用。

在 onPause()方法中，Activity 应该终止在 onResume()方法中播放的所有音频、视频和动画。同时，在该方法中，你还必须解除（deactivate）某些资源，例如：在手动管理而非自动管理情况下的数据库 Cursor 对象。

Note

Android 提供了许多用于管理查询和 Cursor 对象的辅助功能。我们将在第 10 章“使用 Android 数据和存储 API”中深入探讨。

onPause()方法是 Activity 在后台最后一次能够有机会进行清理工作、释放无需资源的地方。你需要保存所有未提交的数据，以免由于应用程序终止后不再返回而丢失数据。

Tip

含有数据输入的 Android 应用程序没有必要遵循传统网页格式的模板（数据输入表单域加上“提交”和“取消”按钮）。相反，数据可以在用户输入各表单域的同时被保存，从而简化了用户界面和 onPause()方法。所以，我们只需要提供一个用于“取消”的按钮，而“保存”按钮则可省略。

Activity 也可以将状态信息保存至 Activity 特定的首选项，或应用程序域（applicationwide）的首选项中。我们将在第 9 章讨论更多有关首选项的话题。

Activity 的 onPause()方法必须快速执行完毕，因为新的前台 Activity 必须要等到 onPause()方法返回后才能启动。

Warning

一般而言，凡是在 onResume()方法中获取的资源和数据均需要在 onPause()方法中释放，否则，如果线程被终止，那么这些资源将有可能不会被彻底地释放掉。

Download at http://www.pin5i.com/

2. 避免 Activity 对象被销毁

在低内存资源的情况下，Android 操作系统会终止任何已暂停、停止或销毁的 Activity 的进程。这本质上意味着所有不处于前台的 Activity 都可能被关闭。

如果 Activity 在 onPause() 方法调用后被关闭，那么它的 onStop() 和 onDe stroy() 方法可能不会被调用。在 onPause() 方法中释放越多的资源，转入后台的 Activity 被终止的可能性就越小。

终止 Activity 的操作并不会将其从 Activity 栈中移除。相反，Activity 的状态会被存入一个 Bundle 对象中，前提是 Activity 实现了它并且使用了 onSaveInstance State() 方法保存自定义数据，不过某些 View 数据会自动进行保存。当用户稍后再次返回这一 Activity 时，它的 onCreate() 方法将被调用，不过此时会带有一个可用的 Bundle 对象作为传入参数。

Tip

为什么在应用程序返回时是否处于被终止的状态会显得如此重要呢？这主要是关系到响应性的问题。应用程序设计者必须在数据维护和快速返回所需资源消耗之间取得微妙的平衡，而不能在应用程序处于后台暂停状态时削弱 CPU 性能和其他系统资源。

（1）使用 onSaveInstanceState() 将 Activity 状态存储至 Bundle

如果某个 Activity 由于内存耗用的原因极易被 Android 操作系统所终止，那么它可以使用 onSaveInstanceState() 方法将状态信息保存至一个 Bundle 对象中。不过这一调用不能被保证在所有条件下都会发生，所以对于关键数据，最好在 onPause() 方法中进行提交与保存。

Tip

onSaveInstanceState() 方法也许可以很好地用来保存一些非关键信息，比如未提交的表单域数据或其他有助于提升应用程序用户体验的数据。

当之后重新返回这个 Activity 时，Bundle 对象将被传入 onCreate() 方法中，从而使得 Activity 又回到先前暂停之前的状态。你也可以使用 onRestoreInstanceState() 方法在 onStart() 调用之后读取 Bundle 对象的信息。

（2）在 onDestroy()中销毁静态 Activity 数据

当一个 Activity 被销毁时，onDestroy() 方法将被调用。onDestroy() 方法在下面两种情况下会被回调：Activity 自然地完成了它的生命周期，或者由于资源占用的原因被 Android 操作系统所终止。

Tip

如果 Activity 被 Android 操作系统所终止，那么 isFinishing() 方法将返回 false。这一方法也可以在 onPause() 方法中使用。尽管如此，这一 Activity 仍然会在之后的 onStop() 方法中被终止。

4.3.2 使用 Intent 管理 Activity 跳转

在 Android 应用程序的整个生命周期中，用户可能会在若干个不同的 Activity 之间进

Download at http://www.pin5i.com/

行跳转。有时，在 Activity 栈中，可能存在多个 Activity 的实例。开发人员需要注意在跳转过程中每一个 Activity 的生命周期问题。

某些 Activity 实例（例如应用程序启动画面）显示一次之后即永远被“主菜单”界面的 Activity 所替代。用户不可能再次跳转至显示启动画面的 Activity，除非重新启动应用程序。

Tip

在这种情况下，可以使用 startActivity()方法和适当的 finish()方法。

其他的 Activity 跳转可能是暂时的，例如，从显示对话框的子 Activity 跳转到初始的 Activity(先前在栈上被暂停后现在重新返回至此)。在这种情况下，父 Activity 运行子 Activity，并且等待返回一个结果。

Tip

在这种情况下，可以使用 startActivityForResult()和 onActivityResult()方法。

1. 使用 Intent 实现 Activity 之间的跳转

正如上文所提到的，Android 应用程序可以拥有若干个入口点，而没有 main()函数，这与 iPhone 等的开发是不一样的。相反，某个特定的 Activity 可以在 AndroidManifest.xml 文件中被指派为主 Activity，从而默认地被运行；有关该文件的更多内容，我们将在第 5 章“使用 Android Manifest 文件定义应用程序”中讨论。

其他 Activity 则可能被指派为在特定条件下运行。例如，一个音乐播放程序可能会指派两个 Activity：一个通用的 Activity 用于在应用程序菜单触发时默认运行；而另一个则被应用程序指派为可选的入口点 Activity，用于根据播放列表或艺术家播放指定的音乐。

2. 使用类名运行一个新的 Activity

你可以通过几种不同的方法启动 Activity。最简单的方法是使用应用程序上下文(Application Context)对象调用 startActivity()方法，同时传入为一个参数：一个 Intent 对象。

一个 Intent (android.content.Intent) 对象是 Android 操作系统用于满足任务需求以及向系统发送广播 Intent 事件所使用的异步消息机制，它同适当的 Activity 或 Service (如果需要的话则运行它) 协同工作。

目前，我们仍然聚焦于 Intent 对象本身，及其与 Activity 协同工作的方式。下面的代码使用了一个显式的 Intent 调用 startActivity()方法。该 Intent 请求名为 MyDrawActivity 的目标 Activity 由其自身类运行。该类在包中的其他地方实现。

```
startActivity(new Intent(getApplicationContext(),
    MyDrawActivity.class));
```

上面的代码对于某些应用程序来讲已经足够了，这些应用程序只需要简单地从一个 Activity 跳转至下一个。当然，你也可以通过一种更为稳健的方式使用 Intent 机制。例如，你可以使用 Intent 结构在 Activity 之间传送数据。

Download at http://www.pin5i.com/

3. 使用动作（Action）和数据创建 Intent

你已经了解了使用 Intent 根据类名来运行一个类的最简单的例子。Intent 无需显式地指定所要运行的组件或类。相反，你可以创建一个 Intent 过滤器（Filter）并且在 Android Manifest 文件中注册它。Android 操作系统会试图满足 Intent 的需求并且基于过滤器规则运行适当的 Activity。

Intent 对象的核心由两个主要部分构成：需要执行的动作（action）和需要处理的数据。你也可以通过 Intent Action 类型和 Uri 对象指定“动作/数据”对。正如你在上一章所看到的，一个 Uri 对象表示了一个字符串，该字符串标明了某个实体的位置和名称。所以，一个 Intent 的含义可以大致理解为对“谁”（即 Uri 所描述的资源）“做什么”（即需要执行的动作）。

最常用的动作类型在 Intent 类中被定义，包括 ACTION_MAIN（用于描述某个 Activity 的主入口点）和 ACTION_EDIT（用于将 Uri 和待编辑的数据联接起来）。你也可以在其他一些应用程序，如浏览器和电话拨号程序中，发现一些用于生成同 Activity 混合（integration）连接的 Action 类型。

4. 运行其他应用程序的 Activity

初始化时，应用程序可能只会启动在其自身包中定义的 Activity。然而，在获得了适当的授权许可后，应用程序也可能会运行包含于其他应用程序的外部 Activity。例如，一个客户关系管理（Customer Relationship Management，CRM）软件可能会运行“联系簿”应用程序以访问联系人数据库、选择某个特定的联系人并向 CRM 软件返回联系人的唯一标识符。

下面的代码展示了使用预定义的 Action（动作，此处为 ACTION_DIAL）创建一个简单 Intent 的实例，它将启动电话拨号程序来拨打某个以 Uri 对象表示的特定电话号码。

```
Uri number = Uri.parse(tel:5555551212);
Intent dial = new Intent(Intent.ACTION_DIAL, number);
startActivity(dial);
```

常用的 Google 应用程序的 Intent 列表可以通过以下网址获得：http://developer.android.com/guide/appendix/g-app-intents.html。同时，OpenIntents 网站提供由开发人员管理的 Intent 协议列表：www.openintents.org/en/intentstable，其中包含有不断更新的第三方应用程序提供的 Intent，当然也包含 Android SDK 中已有的。

5. 使用 Intent 传递附加信息

在一个 Intent 中，你也可以加入某些附加的数据。Intent 的 Extras 属性存储于 Bundle 对象中。Intent 类还具有一系列的辅助方法来获取或设置许多常用数据类型的“名称/值”数据对。

例如，下面的 Intent 实例包含两个额外的信息：一个字符串和一个布尔型量。

```
Intent intent = new Intent(this, MyActivity.class);
intent.putExtra(“SomeStringData”,”Foo”);
intent.putExtra(“SomeBooleanData”,false);
```

Download at http://www.pin5i.com/

Tip

用于标识 Intent 对象扩展信息的可以是任意字符串。不过，Android 约定对于附加信息的键名（key name）需要包含一个包前缀，例如：`com.androidbook.Multimedia.SomeStringData`。

6. 在你的应用程序中使用菜单组织 Activity 和 Intent

正如前面所讲到的，你的应用程序一般会拥有若干个屏幕显示，每一个均有自己的 Activity。菜单、Activity 和 Intent 之间的关系非常紧密。同一个菜单分别以不同方式用于 Activity 和 Intent 的情况是很常见的。

- **主菜单**：它如同一个“转换器”，在应用程序中，不同的菜单项用于运行不同的 Activity。例如，应用程序可能包含启动“游戏”Activity、“高分榜”Activity 和“帮助”Activity 的菜单项。
- **“钻取（Drill-Down）”菜单**：它类似于一个目录，每一个菜单项运行相同的 Activity，但它们均传入不同的数据作为 Intent 的一部分（例如，由所有数据库记录所构成的菜单）。运行某个菜单项可能会启动“编辑记录”Activity，同时传入该项所对应的唯一标识符。

4.4 使用服务

在学习 Android 开发之初就将自己陷入 Activity、Intent、Intent Filter（过滤器）等复杂的概念中是一件让人恐惧的事情。我们尽量将你开始编写 Android 应用程序之时即所需的一切信息抽取出来，呈现给你几个重要的 Activity 类，但是我们必须说明的是，这些类还有很多，它们中的大部分散布在整本书中，将同实例一起讲解。不过现在我们还是需要你稍微“热身”，因为很快在下一章讨论配置 Android Manifest 文件时就会涉及这些内容。

服务是应用程序的组成部分。Android Service 基本上等同于一个没有用户界面的 Activity。它可以如同后台进程一样的运行，也可以像 Web 服务器一样，处理第三方发送的请求。你可以通过 startService()、bindService()方法来使用 Intent 和 Activity 启动服务。任何允许应用程序访问的 Service 必须在 Android Manifest 文件中予以注册。

你可以出于不同目的而使用服务。通常，在不需要用户进行任何输入时，你可以使用服务。下面是一些实现（implement）或使用 Android 服务的例子。

① 天气、邮件或社交网络类应用程序可能需要实现一个服务来周期性地检查数据更新。（注意：有其他的方法也可以实现这种轮询，但这是服务的一种常用功能）。

② 照片或多媒体应用程序可能需要实现一个服务，这样在设备空闲时就可以将新增的内容打包并上传，以与在线数据同步。

③ 视频编辑应用程序可能需要将一些繁冗的操作添加到服务的队列中，这样就可以避免非关键任务对整体系统性能的影响。

④ 新闻应用程序可能需要实现一个服务在用户启动应用程序时进行新闻内容的“预装载”，这样可以提升程序的性能。

Download at http://www.pin5i.com/

有一个关于服务的很好的经验规则：如果某个任务需要使用一个辅助线程（worker thread），并且可能会影响应用程序的响应性和性能，那么可以考虑在应用程序的主生命周期外实现一个服务来处理这一任务。

我们将在第 21 章“使用服务”中讨论更多有关服务的内容。

4.5 接收和广播 Intent

Intent 服务于另一个目的。你可以向 Android 系统广播一个 Intent 对象（通过对 broadcastIntent() 的调用），继而其他有关的应用程序可以接收到这一广播（称为 BroadcastReceiver，即“广播接收器”）。你所编写的应用程序或许会同时实现发送和监听类似这样的 Intent 对象的动作。这种类型的 Intent 对象一般用于通知更大的系统某件事情的发生，它们使用特殊的 Intent Action 类型。

例如，在电池电量低的情况下，Intent 动作 ACTION_BATTERY_LOW 将广播一个警告。如果你的应用程序是某种类型的高耗能的 Service，那么你或许需要监听这一广播，以关闭 Service 直到恢复电能。你可以通过注册来监听广播的包含有 Intent 动作 ACTION_BATTERY_CHANGED 的 Intent 对象，从而获得电池电量级别的改变信息。当然，还有许多其他有趣的系统事件的广播 Intent 对象，例如 SD 卡状态改变、应用程序的安装或卸载，以及墙纸的改变等。

应用程序还可以通过广播机制来实现信息的共享。例如，一个电子邮件应用程序可以在它接收到一封新邮件时广播一个 Intent，这样，其他对此“感兴趣”的应用程序（如垃圾邮件过滤或反病毒应用程序）就能够做出相应的反应了。

Note

在第 19 章“使用 Android 可选硬件 API”中，我们将讲解有关硬件和电池的更多内容，届时，我们将给出使用 BroadcastReceiver 对象的实例。

小　结

我们尽量在提供完整参考和避免过度繁细之间取得一个平衡，因为许多知识在你开发一般的 Android 应用程序时是没有必要掌握的。但同时，我们会关注于那些对你继续深入学习 Android 应用程序开发有帮助的细节，以便你能够充分理解本书后续章节中所提供的示例。

Activity 和 View 类是所有 Android 应用程序的核心组成部分。在应用程序中，每一个 Activity 都承担自己特定的任务，通常它们都包含自身的由 View Widget 构成的用户界面。每一个 Activity 都通过一系列的生命周期回调方法管理自己的资源和数据。从一个 Activity 向另一个 Activity 的跳转是通过 Intent 机制而得以实现的。Intent 对象是由 Android 操作系统以运行适当的 Activity 或 Service 的方式处理和响应的异步消息。你也可以使用 Intent 对象向所有有关的正在监听的 BroadcastReceiver 广播系统域（systemwide）事件。

Download at http://www.pin5i.com/

参考资料和更多信息

Android SDK 参考中有关应用程序 Context 类的内容.http://developer.android.com/reference/android/content/Context.html.

Android SDK 参考中有关 Activity 类的内容.http://developer.android.com/reference/android/app/Activity.html.

Android 开发指南（Dev Guide）：Intents and Intent Filters. http://developer.android.com/guide/topics/intents/intents-filters.html.

Download at http://www.pin5i.com/

第5章

使用 Android Manifest 文件定义应用程序

Android 工程使用一个特殊的配置文件来定义应用程序的设置，被称为 Android Manifest 文件。它所定义的设置包括应用程序的名称、版本，应用程序所需的许可权限，以及组成应用程序的各个组件。在本章中，我们将深入 Android Manifest 文件，探索它的各个细节，并且学习如何通过它定义和描述不同类型的应用程序行为。

5.1 配置 Android Manifest 文件

Android 应用程序的 Manifest 文件为一种特定格式的 XML 文件，它必须伴随每一个 Android 应用程序。这一文件包含标识应用程序身份的重要信息，在其中定义了应用程序的名称、版本信息和应用程序所依赖的应用程序组件，以及应用程序运行所需的许可权限和其他的应用程序配置信息。

Android Manifest 文件名为 `AndroidManifest.xml`，它必须包含在 Android 工程的最顶层。Android 系统使用文件中的信息来实现以下功能。

① 安装和升级应用程序包。

② 向用户显示应用程序细节，如应用程序名称、描述和图表。

③ 指定应用程序的系统需求，包括所支持的 Android SDK，所需的硬件配置（例如，d-pad 导航），以及应用程序所依赖的平台特性（例如，使用多点触控）。

④ 运行应用程序的 Activity。

⑤ 管理应用程序的许可权限。

⑥ 配置其他的高级应用程序参数，包括作为服务、广播接收器或者内容提供器的配置细节。

⑦ 开启某些应用程序设置，例如是否允许调试，是否为应用程序测试使用配置工具等。

Tip

如果你使用的是带有 Android Eclipse 插件（Android Plug-In for Eclipse，ADT）的 Eclipse 环境，那么 Android 工程向导会为你创建初始的 `AndroidManifest.xml`。如果你没有使用 Eclipse，那么 `android` 命令行工具也同样可以为你创建 Android Manifest 文件。

Manifest 文件处于整个 Android 工程的最高层。你可以使用 Eclipse 的 Manifest 文件资源编辑器（Eclipse ADT 插件的一个特性）来编辑它，也可以手工进行编辑。

1. 使用 Eclipse 编辑 Manifest 文件

你可以使用 Eclipse 的 Manifest 文件资源编辑器来编辑 Manifest 文件。编辑器将 Manifest

Download at http://www.pin5i.com/

信息组织成如下几个组：

① Manifest 标签；

② Application 标签；

③ Permissions 标签；

④ Instrumentation 标签；

⑤ AndroidManifest.xml 标签。

让我们仔细研究 Android 的示例 Manifest 文件。下面的图片和示例来自于一个名为 Multimedia 的 Android 应用程序，我们将在第 15 章“使用 Android 多媒体 API”中构建它。我们之所以选择这个工程作为示例，是因为它显示了 Android Manifest 文件的若干个特点，而不是你在 MyFirstAndroidApp 工程中见到的简单的默认 Manifest 文件。

2. 使用 Manifest 标签页配置包范围（package-wide）的设置

Manifest 标签页(如图 5.1 所示)包含包范围的设置，包括包名、版本信息和支持的 Android SDK 信息。你还可以在这里设置任何需要的硬件或功能需求。

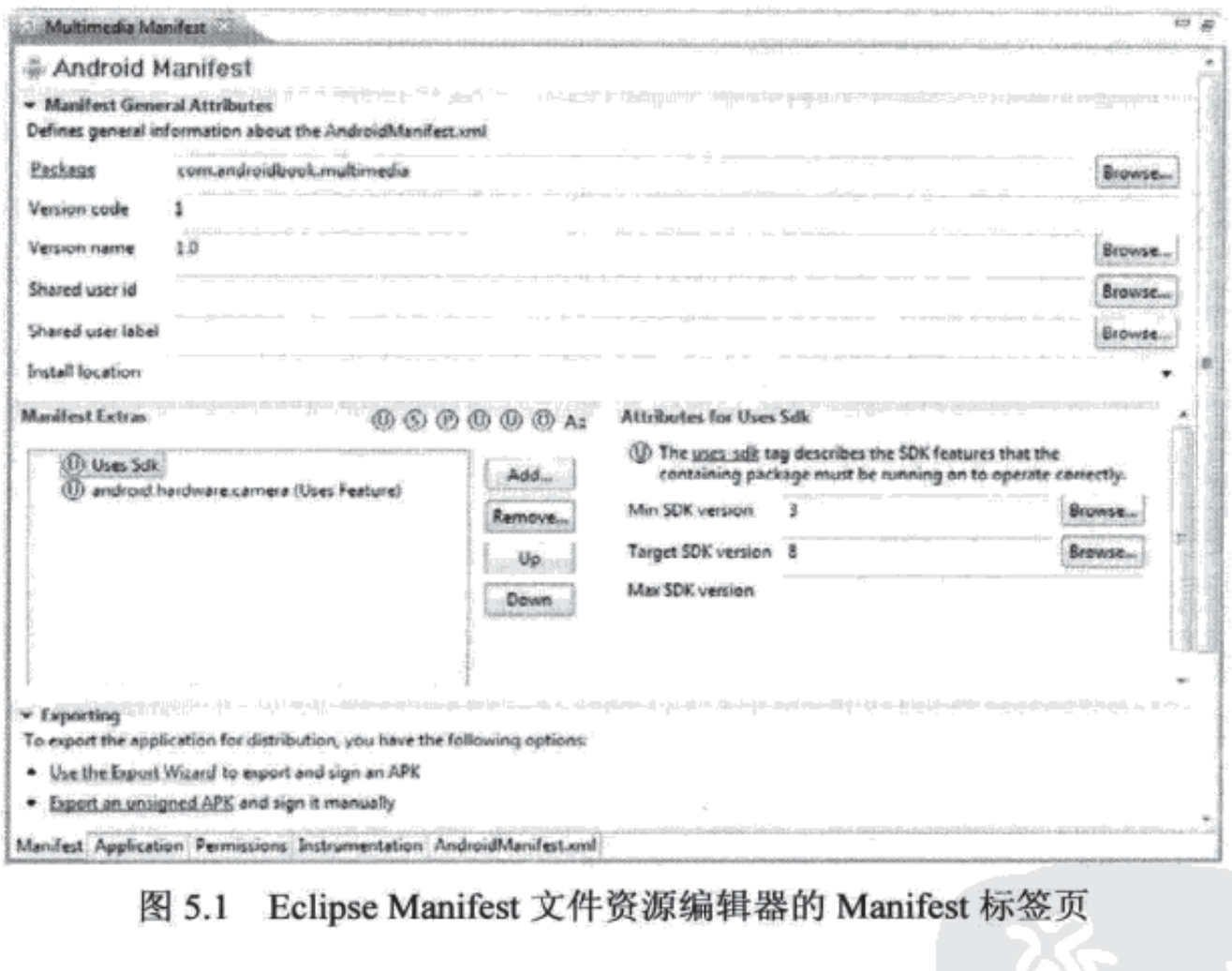


图 5.1 Eclipse Manifest 文件资源编辑器的 Manifest 标签页

3. 使用 Application 标签页管理应用程序和 Activity 设置

Application 标签页（如图 5.2 所示）包含应用程序范围的设置，包括应用程序标签（label）和图标，以及应用程序组件的信息，如 Activity、Intent 过滤器以及包括服务、Intent 过滤器、内容提供器在内的设置信息。

4. 使用 Permission 标签页强制指定（enforce）应用程序权限

Permission 标签页（如图 5.3 所示）包含应用程序所需的所有许可权限规则。它还可以用来指定为应用程序创建的自定义许可权限。

Download at http://www.pin5i.com/

Warning

不要将这里的 Permission 标签页与应用程序的 Permission 域（在 Application 标签页中的一个下拉列表）相混淆。此处的 Permission 标签页用来定义应用程序所需的许可权限。

图 5.2 Eclipse Manifest 文件资源编辑器的 Application 标签页

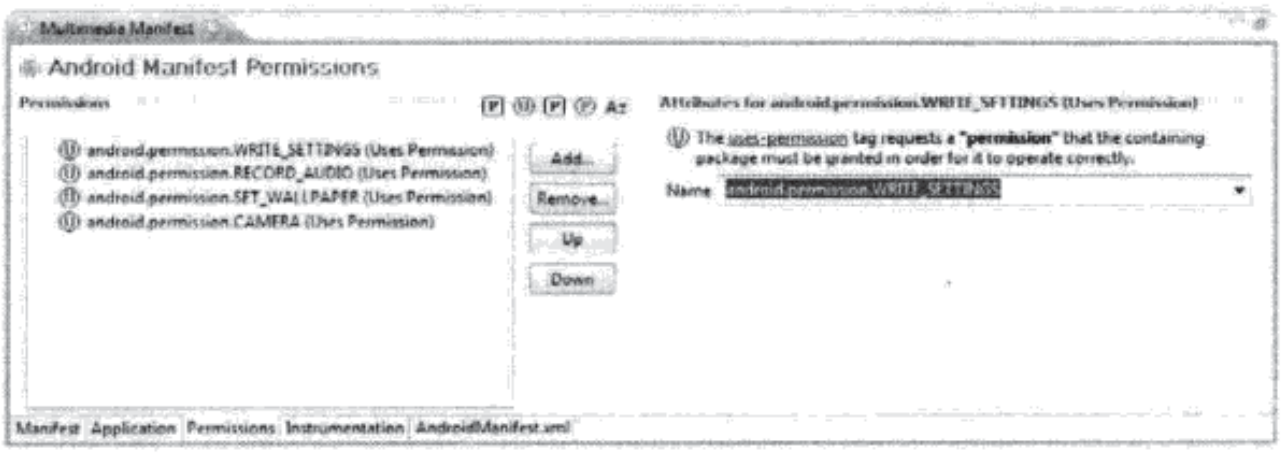


图 5.3 Eclipse Manifest 文件资源编辑器的 Permission 标签页

5. 使用 Instrumentation 标签页管理测试工具

Instrumentation 标签页允许开发人员声明用于监控应用程序的一切工具类。我们将在第 28 章“测试 Android 应用程序”中讨论更多有关测试及其工具的话题。

6. 手工编辑 Manifest 文件

Android 应用程序的 Manifest 文件为一种特定格式的 XML 文件。你可以通过点击 AndroidManifest.xml 标签页来手工编辑这个 XML 文件。

通常 Android Manifest 文件中包含一个<manifest>标记(tag)，其中包含一个<application>标记。下面是“Multimedia”应用程序的 AndroidManifest.xml 文件的示例：

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.androidbook.multimedia"
```

Download at http://www.pin5i.com/

```
        android:versionCode="1"
        android:versionName="1.0">
        <application android:icon="@drawable/icon"
            android:label="@string/app_name"
            android:debuggable="true">
            <activity android:name=".MultimediaMenuActivity"
                android:label="@string/app_name">
                <intent-filter>
                    <action
                        android:name="android.intent.action.MAIN" />
                    <category
                        android:name="android.intent.category.LAUNCHER" />
                </intent-filter>
            </activity>
            <activity android:name="AudioActivity"></activity>
            <activity android:name="StillImageActivity"></activity>
            <activity android:name="VideoPlayActivity"></activity>
            <activity android:name="VideoRecordActivity"></activity>
        </application>
        <uses-permission
            android:name="android.permission.WRITE_SETTINGS" />
        <uses-permission
            android:name="android.permission.RECORD_AUDIO" />
        <uses-permission
            android:name="android.permission.SET_WALLPAPER" />
        <uses-permission
            android:name="android.permission.CAMERA"></uses-permission>
        <uses-sdk
            android:minSdkVersion="3"
            android:targetSdkVersion="8">
        </uses-sdk>
        <uses-feature
            android:name="android.hardware.camera" />
    </manifest>
```

现在，我们来总结上面的文件都告诉了我们有关“Multimedia”应用程序的哪些信息。

① 应用程序使用包名 com.androidbook.multimedia。

② 应用程序的版本名称为 1.0。

③ 应用程序的版本号为 1。

④ 应用程序名称和标签（label）存储于包含在/res/values/strings.xml 资源文件中的名为@string/app_name 的字符串中。

⑤ 应用程序在 Android 设备上可调试。

⑥ 应用程序图标是一个名为 icon 的图形文件（格式可能是 PNG、JPG 或 GIF），存储在/res/drawable 目录下（实际上对于不同像素的图片，存在若干个不同的版本）。

⑦ 应用程序拥有 5 个 Activity（MultimediaMenuActivity、AudioActivity、StillImageActivity、VideoPlayActivity 和 VideoRecordActivity）。

⑧ MultimediaMenuActivity 是应用程序的主入口点。这个 Activity 是应用程序 Drawer 中的程序图标被选中时打开的 Activity。

⑨ 应用程序的运行需要以下许可权限：录制音频的能力、在手机设备上进行壁纸设定的能力、访问内置摄像头的能力和写入设置的能力。

Download at http://www.pin5i.com/

⑩ 应用程序可以运行在 API Level 处于 3～8 的任何情况下，换句话说，Android SDK 1.5 是其可支持的最低版本，应用程序编写的目标版本为 Android 2.2。

⑪ 最后，要正常地运行应用程序，需要一个摄像头。

现在，我们详细地展开讨论这些重要的配置。

5.2 管理你的应用程序标识

应用程序的 Android Manifest 文件定义了应用程序的属性。包名称必须在 Android Manifest 文件中的<manifest>标记内，使用 package 属性予以定义，如下所示：

```
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.androidbook.multimedia"
    android:versionCode="1"
    android:versionName="1.0">
```

5.2.1 为你的应用程序指定版本

为你的应用程序指定版本是让你的应用程序在这一开发领域内保持一定地位的重要原因。明智地为应用程序指定版本可以帮助减少版本混乱与疑惑，同时也能让产品支持与升级变得更容易。<manifest>标记内定义了两种版本属性：版本名称和版本号。

版本名称（android:versionName）是一个由开发人员定义的、用户易懂的版本属性。当用户管理他们设备上的应用程序，或从市场购买下载应用程序时，版本名称信息将呈现给用户。而开发人员则通过这一版本信息跟踪应用程序在开发领域内的版本。我们将在第 26 章“移动软件开发过程”中更为详细地讨论有关移动应用程序版本号的问题。

Warning

虽然你可以针对某些 Manifest 文件设置，如 android:versionName，使用诸如 @string 这样的资源引用，但需要注意的是某些发布系统可能会不支持。

Android 操作系统使用版本号（android:versionCode）来管理应用程序的升级，版本号是一个数值属性。我们将在第 29 章“销售你的 Android 应用程序”中讨论更多有关发布和升级支持的话题。

5.2.2 给你的应用程序加上名称和图标

Android Manifest 文件中的<application>标记可以用来设置应用程序的总体设置，如应用程序图标（android:icon）和友好名称（android:label）。这些设置均是<application>标记内的属性。

例如，下面我们将应用程序图标和标签分别设置为应用程序包里提供的可绘制（drawable）资源和字符串资源：

```
<application android:icon="@drawable/icon"
    android:label="@string/app_name">
```

另外，你也可以在<application>标记内设定其他可选的属性，例如应用程序描述（android:description），以及是否支持设备调试（android:debuggable="true"）等。

Download at http://www.pin5i.com/

5.3 强制指定应用程序的系统需求

除了配置应用程序标识以外，Android Manifest 文件还可用来指定应用程序正常运行所需的系统需求。例如，一个增强现实（Augmented Reality，AR）应用程序可能需要手机具有 GPS、指南针和摄像头。类似地，依赖于 Android SDK 中提供的蓝牙 API 的应用程序需要具有 SDK 达到 API Level 5 或更高（Android 2.0）的手机。类似这样的系统需求就可以在 Android Manifest 文件中予以定义和指定。之后，当应用程序出现在 Android Market 上时，可以通过这些信息进行筛选；当在系统上安装应用程序包时，Android 平台会检查这些需求，并且在必要时抛出错误。

开发人员可以在 Android Manifest 文件中配置的应用程序系统需求包括：

① 应用程序支持的 Android SDK；
② 应用程序使用的 Android 平台特性；
③ 应用程序需要的 Android 硬件配置；
④ 应用程序支持的屏幕尺寸和像素密度；
⑤ 应用程序链接的外部库。

5.3.1 针对特定 SDK 版本

Android 设备上运行着不同版本的 Android 平台。一般而言，较为老旧，功能较少，甚至较便宜的设备运行着较低版本的 Android 平台，相反，市面上更新、功能更强大的设备则更可能运行着最新的 Android 版本。

目前用户手里有着各种不同的 Android 设备。对于一个应用程序，开发人员必须决定它的受众群体。应用程序是否需要满足尽可能大的用户群体而支持尽可能多的平台版本？或者开发的是领先于潮流的游戏，需要最新的硬件设备支持？

开发人员可以在 Android Manifest 文件中使用<uses-sdk>标记来指定应用程序所支持的 Android 平台版本。这个标记由 3 个重要的属性组成。

- **minSdkVersion 属性**：该属性指定应用程序支持的最低 API Level。
- **targetSdkVersion 属性**：该属性指定应用程序支持的最佳 API Level。
- **maxSdkVersion 属性**：该属性指定应用程序支持的最高 API Level。

Tip

Android Market 使用应用程序 Manifest 文件中诸如<uses-sdk>这样的标记来筛选应用程序，以满足不同的用户。对于想要在 Android Market 中发布的应用程序而言，这是必需的标记。忽略它将会在构建环境中触发一个警告。

<uses-sdk>标记的每一个属性都是一个整数，它表示与某个 Android SDK 相关联的 API Level，并不与 SDK 版本直接对应，而与 SDK 直接关联的是 API Level 的修订（revision）。API Level 由 Android SDK 的开发人员设定，你可以在 SDK 文档中找到各个版本的 API Level。

Note

对于每一个新的 Android SDK 版本，API Level 会随之升高。SDK 的版本文档中包含这些信息。

表 5.1 给出了移植应用程序所需的 Android SDK 版本信息。

Download at http://www.pin5i.com/

表 5.1 Android SDK 版本及其 API Level

Android SDK 版本	API Level（整型数值）
Android 1.0 SDK	1
Android 1.1 SDK	2
Android 1.5 SDK (Cupcake)	3
Android 1.6 SDK (Donut)	4
Android 2.0 SDK (Éclair)	5
Android 2.0.1 SDK (Éclair)	6
Android 2.1 SDK (Éclair)	7
Android 2.2 SDK (FroYo)	8
Android SDK (Gingerbread)	9

1. 指定最低 SDK 版本

你始终应该为应用程序指定 minSdkVersion 属性，它代表你的应用程序所支持的最低 Android SDK 版本。

例如，如果你的应用程序需要 Android SDK 1.6 中的 API，那么你就需要查看 SDK 的文档，找到这一版本被定义为 API Level 4。所以，在 Android Manifest 文件中的<manifest>标记内插入下面的代码：

```
<uses-sdk android:minSdkVersion="4" />
```

就是这么简单。你需要使用可能的最低 SDK 版本，因为这一信息将在安装时予以验证，而且你应该会希望你的应用程序能够适应尽量多的手机设备。但是，你需要保证在这些非目标平台（所支持的 API Level 低于目标 SDK 的平台，将在后面的小节中讨论）上进行了充分的测试。

2. 指定目标 SDK 版本

你始终应该为应用程序指定 targetSdkVersion 属性，它代表你的应用程序所构建和测试的 Android SDK 版本。

例如，如果你的应用程序可以向下兼容到 Android 1.6（API Level 4），但以 Android 2.2 SDK（API Level 8）作为目标平台且在此基础上进行测试，那么你可以将 targetSdkVersion 属性指定为 8。这样，可以将下面的代码插入到 Android Manifest 文件的<manifest>标记块中：

```
<uses-sdk android:minSdkVersion="4" android:targetSdkVersion="8" />
```

为什么需要指定你使用的目标 SDK 的版本呢？这是因为 Android 平台具有内置的向下兼容功能。你可以这么想：某个 API 的某个特定方法从 API Level 1 时起就有了，但是该方法的内部（它的行为）可能已经随着 SDK 版本的更新发生了变化。通过为应用程序指定目标 SDK 版本，即使运行着更高版本的平台，Android 操作系统也会尝试将应用程序关联到指定版本的 SDK（也是你测试应用程序所使用的版本）上。这意味着应用程序需要抛弃新版 SDK 所提供的可能会导致错误发生的“高级”功能，而使用老旧的方式来运行。

Download at http://www.pin5i.com/

3. 指定最大 SDK 版本

通常很少指定应用程序的 maxSdkVersion 属性。这一属性表示应用程序所支持的最高 Android SDK 版本，用 API Level 来表示。它限定了应用程序的向上兼容性。

设置这一属性的目的之一是限制安装最新版本 SDK 的用户使用，例如，你有可能开发了某个应用程序，但仅有 Beta 版是免费的，而需要使用最新 SDK 的用户付费，这样，通过设置免费版应用程序 Manifest 文件的 maxSdkVersion 属性，可以实现对他们安装免费版的限制。这样做有什么缺点吗？有，如果用户对手机设备的 SDK 进行了在线升级，那么你的应用程序将停止工作（并且不会出现），这样会给用户带来不便，甚至可能会影响到他们在市场上对你应用程序的评价。

简而言之，仅在绝对有必要使用 maxSdkVersion 时才使用它，从而避免其带来的不必要的麻烦。

5.3.2 强制指定应用程序平台需求

Android 设备拥有不同的硬件和软件配置。某些设备有内置的键盘，而其他的可能仅仅依赖于软键盘。同理，某些 Android 设备支持最新的 3D 图形库，而其他的可能支持度有限甚至完全不支持。Android Manifest 文件使用几个标记来指定所支持或需要的系统特性和硬件配置。

1. 指定支持的输入方式

<uses-configuration>标记可以用来指定应用程序所支持的输入方式。对于五向导航（five-way navigation）、硬件键盘及其类型等均有不同的配置属性，还包括诸如方向杆（directional pad）、轨迹球和滚轮在内的导航设备以及触屏设置。

在配置属性内部没有“或”逻辑。如果应用程序支持多种输入配置，那么可以配置多个<uses-configuration>标记，每一个标记对应一个完整的配置。

例如，如果你的应用程序既需要支持物理键盘，又需要支持触屏（手指和笔输入均可），那么可以像下面这样在 Manifest 文件中分别定义两个独立的<uses-configuration>标记：

```
<uses-configuration android:reqHardKeyboard="true"
    android:reqTouchScreen="finger" />
<uses-configuration android:reqHardKeyboard="true"
    android:reqTouchScreen="stylus" />
```

要获得更多有关 Android Manifest 文件中<uses-configuration>标记的信息，可以查看 Android SDK 参考：http://developer.android.com/guide/topics/manifest/uses-configuration-element.html。

2. 指定需要的设备特性

不是所有的 Android 设备都支持所有的 Android 功能。换言之：Android 设备会选择性地包含某些 API（及其相关的硬件）。例如，不是所有的 Android 设备都支持多点触控或者带有摄像头闪光灯。

<uses-feature>标记可以用来指定应用程序正常运行所需的 Android 特性。这些设置是可选的，也就是说，Android 操作系统并不强制要求应用程序具有这些设置，但某些发布渠道，如 Android Market，则需要根据这些信息来筛选应用程序，从而满足特定用户的需求。

Download at http://www.pin5i.com/

如果你的应用程序需要多种特性的支持，那么你需要为每一种特性创建一个<uses-feature>标记。例如，某个需要光传感器（light sensor）和接近传感器（proximity sensor）支持的应用程序可以定义两个标记：

```
<uses-feature android:name="android.hardware.sensor.light" />
<uses-feature android:name="android.hardware.sensor.proximity" />
```

<uses-feature>标记常用来指定应用程序所支持的 OpenGL ES 版本。默认情况下，所有应用程序使用 OpenGL ES 1.0（这也是所有的 Android 设备支持的特性），但是，如果你的应用程序需要更新版本 OpenGL ES（如 2.0）的支持，则需要在 Android Manifest 文件中指定这一特性：<uses-feature>标记的 android:glEsVersion 属性可以用来进行这一设置。这里一般指定的是应用程序所需的最低 OpenGL ES 版本，如果你的应用程序可以运行在 1.0 版本和 2.0 版本下，那么就指定为最低的那个（这样更多的用户可以安装你的应用程序）。

有关 Android Manifest 文件中<uses-feature>标记的更多信息，可以查阅 Android SDK 参考。

3. 指定支持的屏幕尺寸

Android 设备具有各种不同的形状和尺寸，而目前市场上可见的 Android 设备也具有各种不同的屏幕尺寸和像素密度。Android 平台根据屏幕尺寸将屏幕分成了几个类型（小、普通和大），同时也可以由像素密度来划分（低、中和高）。这些指标很好地对 Android 平台的屏幕进行了分类。

应用程序可以针对特定的屏幕尺寸和像素密度提供自定义的资源（我们将在第 6 章“管理应用程序资源”中讨论这一话题）。<supports-screen>标记可以用来指定应用程序所支持的 Android 屏幕类型。

例如，如果应用程序支持 QVGA（小）和 HVGA 屏幕（普通）而不考虑像素密度，那么<supports-screen>标记可作如下配置：

```
<supports-screens android:smallScreens="true"
    android:normalScreens="true"
    android:largeScreens"false"
    android:anyDensity="true"/>
```

有关 Android Manifest 文件中<supports-screen>标记的更多信息，可以查阅 Android SDK 参考，也可以查看 Android 开发指南（Android Dev Guide）中的“Screen Support（屏幕支持）”部分。

5.3.3 使用外部库

你可以在 Android Manifest 文件中注册任何应用程序链接的共享库。默认地，每一个应用程序被链接到标准 Android 包（如 android.app）并且能够知道自己的包。但是，如果你的应用程序链接到了其他附加的包，那么它们必须在 Android Manifest 文件的<application>标记内使用<uses-library>标记进行注册。例如：

```
<uses-library android:name="com.sharedlibrary.sharedStuff" />
```

这一功能通常用来链接可选的 Google API。有关 Android Manifest 文件中<uses-library>标记的更多信息，可以查阅 Android SDK 参考。

Download at http://www.pin5i.com/

5.4 注册 Activity 和其他应用程序组件

应用程序的每一个 Activity 均需要在 Android Manifest 文件中使用<activity>标记予以定义。例如，下面摘录的 XML 片段定义了一个名为 AudioActivity 的 Activity 类：

```
<activity android:name="AudioActivity" />
```

这一 Activity 必须在 com.androidbook.multimedia 包中以类的方式予以定义，也就是在 Android Manifest 文件的<manifest>元素中指定的那个包。你还可以在 Activity 的名称中使用“.”作为前缀来指定 Activity 类的作用域：

```
<activity android:name=".AudioActivity" />
```

或者可以指定完整的类名：

```
<activity android:name="com.androidbook.multimedia.AudioActivity" />
```

Warning

你必须使用<activity>标记定义 Activity，否则它将不能运行。程序开发人员实现了一个 Activity 却不能正常运行，反复查找问题却终于发现只因为忘记了在 Android Manifest 文件中给予注册，这种情况是很常见的。而且更为麻烦的是，遇到这种情况的程序员一般会查看 LogCat 的输出，但是错误输出又仅仅类似于一个普通的崩溃，这让程序员更为恼火。

5.4.1 使用 Intent 过滤器为应用程序指派主入口点 Activity

Activity 可以被指派为主入口，方法是在应用程序的 AndroidManifest.xml 文件中使用 MAIN 动作类型和 LAUNCHER 族（category）配置 Intent Filter。

下面的 XML 标记将一个名为 MainMenuActivity 的 Activity 配置为应用程序的主入口点：

```
<activity android:name=".MultimediaMenuActivity"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

5.4.2 配置其他 Intent 过滤器

Android 操作系统使用 Intent 过滤器来解析隐式的 Intent，也就是那些没有指定 Activity 或待运行的组件的 Intent。Intent 过滤器可以应用于 Activity、Service 和 BroadcastReceiver。过滤器声明该组件为开放的，可以接收任何发送至 Android 操作系统的、匹配其规则的 Intent。

Intent 过滤器使用<intent-filter>标记定义，它必须包含至少一个<action>标记，也可以包含其他的信息，如<category>和<data>代码段。下面是一个示例 Intent 过滤器的代码段，它应该出现在<activity>区段中：

Download at http://www.pin5i.com/

```
<intent-filter>
    <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.BROWSABLE" />
        <category android:name="android.intent.category.DEFAULT" />
        <data android:scheme="geoname" />
</intent-filter>
```

上面的 Intent 过滤器使用预定义的称为 VIEW 的行为来定义，这一行为用于查看特定的内容。同时，它是 BROWSABLE（可导航）的，并且使用了“geoname”作为其规则，这样，如果某个 Uri 以“geoname://”开头，那么带有这一 Intent 过滤器的 Activity 将会运行。在第 14 章“使用位置服务 API”中，你将会读到有关这一特定 Intent 过滤器的更多内容。

Tip

你可以为你的应用程序自定义唯一的动作。如果这样，一定确保为这些动作编写相应的说明文档，以便第三方能够使用它们。

1. 注册服务和广播接收器（Broadcast Receiver）

所有的应用程序组件都需要在 Android Manifest 文件中定义。包括 Activity 在内，所有的服务（Service）和广播接收器都必须在 Android Manifest 文件中注册。其中，服务使用<service>标记注册，广播接收器使用<receiver>标记注册。它们均使用 Intent 过滤器。你将在后续章节中学到更多关于服务、广播接收器和 Intent 过滤器的知识。

2. 注册内容提供器（Content Provider）

如果你的应用程序充当了一个内容提供器（Content Provider），能够高效地为其他应用程序提供数据共享服务，那么它必须在 Android Manifest 文件中使用<provider>标记声明这一能力。配置一个内容提供器包括决定待分享的数据主体和访问它们所需的许可权限。

Note

我们将在第 11 章“使用内容提供器在应用程序间共享数据”中讨论更多有关内容提供器的内容。

5.5 使用许可权限

Android 操作系统处于锁定状态，所以应用程序仅有有限的“能力”在其线程之外对操作系统造成不利的影响。Android 应用程序运行在它自身的 Linux 用户账号和虚拟机“气泡”的包裹之中。

5.5.1 注册应用程序所需的许可权限

Android 应用程序默认没有任何许可权限。任何用于共享资源和授权访问的（无论是共享数据，如联系人数据库，还是访问底层硬件，如内置摄像头）都必须在 Android Manifest 文件内进行显式地注册。这些许可权限将在应用程序安装时被授予。

Download at http://www.pin5i.com/

Tip

当用户安装应用程序时，他们将会被告知应用程序运行所需的所有许可权限，用户必须同意授予这些权限。另外，应用程序只应该申请它所需要的许可权限。

下面的 XML 文件片段是从前面的 Android Manifest 文件中提取的，它使用<uses-permission>标记定义了使用内置摄像头的权限。

```
<uses-permission android:name="android.permission.CAMERA" />
```

完整的许可权限列表可以在 android.Manifest.permission 类中找到。应用程序的 Manifest 文件只应该包含它运行所需的许可权限。在安装时用户将会被告知每一个 Android 应用程序所需的许可权限。

Tip

也许你会发现，在某些时候，许可权限并不是强制性的（也就是说，你也可以在没有获得权限的情况下进行操作）。但是，我们必须慎重地采用这种方式，原因有二：（1）运行时用户会被告知应用程序正在进行敏感的操作；（2）在更新的 SDK 版本中，这种许可权限可能会带有强制性。

5.5.2 注册应用程序授予其他应用程序的许可权限

应用程序也可以使用<permission>标记定义属于自己的许可权限。许可权限必须使用 android.permission 属性进行描述，并且授予某个特定的应用程序组件，如 Activity。

Tip

应用程序权限使用 Java 风格的命名域（例如：com.androidbook.MultiMedia.ViewMatureMaterial）。

许可权限可以在以下若干个点被强制验证：

① 当启动一个 Activity 或 Service 时；

② 当访问由内容提供器提供的数据时；

③ 当处在函数调用层时；

④ 当发送和接收 Intent 广播时。

许可权限可以拥有 3 种主要的保护级别：normal（正常）、dangerous（危险）和 signature（签名）。normal 保护级别是一个不错的针对应用程序细节权限的默认级别。dangerous 保护级别用于高风险操作，它们将可能对设备造成不利的影响。最后，由于被控制的应用程序互通性（interoperability），signature 保护级别允许任何使用相同证书签名的应用程序使用其组件。你将在第 29 章中学习到有关应用程序签名的更多内容。

许可权限还可以被细分为若干类型，被称作 permission groups（许可权限组），它们描述或警告特定 Activity 需要许可权限的原因。例如，许可权限会应用于一些 Activity，它们可能需要暴露如位置和个人信息等的敏感用户数据（android.permission-group.LOCATION），访问底层硬件（android.permission-group.HARDWARE_CONTROLS），或进行会使用户付费的操作（android.permission-group.COST_MONEY）。完整的许可权限组列表可以在 Manifest.permission_group 类中找到。

你也可以使用<grant-uri-permission>标记引入处于 Uri 级别的细节权限。

Download at http://www.pin5i.com/

如果你想了解更多关于 Android 许可架构的知识，我们强烈推荐你阅读 Android 开发指南中关于许可和安全的章节。

5.6 探索 Manifest 文件的其他参数

现在我们已经讲解完了 Android Manifest 文件的基础内容，但是在 Android Manifest 文件内还有许多其他的可用配置，它们使用不同的标记区段，当然还有许多我们已经讨论过的包含于每一个标记中的属性。

其他一些可以在 Android Manifest 文件中进行配置的特性包括：

① 作为<application>标记的属性设置应用程序域（applicationwide）主题；

② 使用<instrumentation>标记配置检测（instrumentation）设置；

③ 使用<activity-alias>标记为 Activity 定义别称；

④ 使用<intent-filter>标记创建 Intent 过滤器；

⑤ 使用<receiver>标记创建广播接收器。

要获取 Android SDK 中每一个可用的标记及其属性（attribute）的详细描述，可以阅读 Android SDK 参考。

小　结

每一个 Android 应用程序都拥有一个特定格式的 XML 文件，它被称作 AndroidManifest.xml。这一文件非常详细地描述了应用程序的标识。你必须要在 Android Manifest 文件中定义的信息包括应用程序的名称、版本信息、程序包含哪些组件、要求何种设备配置，以及运行程序需要获取哪些许可。Android 操作系统在安装、升级和运行应用程序时都会使用 Android Manifest 文件。Android Manifest 文件中的某些细节信息还会为第三方所用，例如 Android Market 的发布频道。

参考资料和更多信息

Android 开发指南（Dev Guide）：The AndroidManifest.xml File. http://developer.android.com/guide/topics/ manifest/manifest-intro.html.

Android 开发指南（Dev Guide）：Android API Levels. http://developer.android.com/guide/appendix/api-levels.html.

Android 开发指南（Dev Guide）：Supporting Multiple Screens. http://developer.android.com/guide/practices/screens_support.html.

Android 开发指南（Dev Guide）：Security and Permissions. http://developer.android.com/guide/topics/security/security.html.

Download at http://www.pin5i.com/

第6章

管理应用程序资源

编写良好的代码往往在程序中调用它的资源，而不是直接将其写入代码中，这基于若干个原因。将应用程序资源存储于某个特定的位置更有利于管理它们，而且能使代码更容易阅读与管理。类似字符串这样的外部资源能使应用程序更容易根据不同的地区和语言来进行本地化。

在本章中，你将会学习到 Android 应用程序存储和访问类似字符串、图形等重要资源的方法。你还会学习如何在工程文件中管理 Android 资源，以实现本地化和针对不同设备进行配置。

6.1 什么是资源

所有的应用程序都由两部分构成：功能（代码指令）和数据（资源）。功能决定应用程序的行为，它包括让应用程序得以运行的所有算法。资源包括文本字符串、图像和图标、音频文件、视频和其他应用程序使用的组件。

> **Tip**
>
> 本章中的许多代码示例均来自于 SimpleResourceView、ResourceViewer、Resource Roundup 和 ParisView 应用程序。在本书的网站上可以下载到这些应用程序的源代码。

6.1.1 存储应用程序资源

在 Android 工程中，Android 资源文件是同 Java 类文件分开存储的。大多数常见的资源类型被存储于 XML 中。当然，你也可以将原始数据文件和图形作为资源来存储。

6.1.2 理解资源目录的层级结构

在 Android 工程内，资源是严格按照目录层级结构来组织的。所有的资源必须存储于以小写字母特别命名的子文件夹中，这些子文件夹存储于/res 工程目录中。

不同类型的资源存储于不同的目录下。当使用 Eclipse 插件时，资源子目录将自动在创建 Android 工程时建立，如表 6.1 所示。

表 6.1　　默认的 Android 资源目录

资源子目录	说　明
/res/drawable-*/	图形资源
/res/layout/	用户界面资源
/res/values/	简单数据，如字符串、颜色值等

Download at http://www.pin5i.com/

每一个资源类型对应着特定的资源子目录。例如，所有的图像均存储在/res/drawable文件夹中。资源还可以更进一步地使用带有特定名称的目录修饰符（qualifier）组织起来。例如，/res/drawable-hdpi 文件夹存储供高像素密度屏幕使用的图像，而/res/drawable-ldpi 文件夹则存储供低像素密度屏幕使用的图像，同理，/res/drawable-mdpi 文件夹则存储供中等像素密度屏幕使用的图像。反之，如果你的图像资源可供所有屏幕使用，那么可以简单地将它们存储在/res/drawable 文件夹中。我们将在本章的后面部分讨论有关资源目录的更多内容。

如果你使用的是包含 Android 开发工具插件（Android Development Tools Plug-In，ADT）的 Eclipse，那么你将会发现向工程中加入资源是一件很简单的事情。当你向工程资源目录/res 下的适当文件夹中添加新的资源时，插件将会探测到，然后在后台编译这些新资源，并生成 R.java 文件，以使你能够在程序中访问它。

如果你使用的是另外的开发环境，那么你需要在命令行模式下使用 aapt 工具，以编译资源和应用程序二进制文件以供在硬件设备和模拟器上发布。你可以在所有版本的 Android SDK/tools 目录下找到 aapt 工具。

Tip

可以使用 aapt 为你的应用程序自动创建组件的档案（archive）并且高效地编译它们。这一工具可以通过命令行参数进行配置，以达到例如仅为特定硬件或语言包含组件的目的。默认条件下所有目标的所有资源都将被包含。

6.1.3 资源的类型

Android 应用程序依赖于多种不同类型的资源实现用户界面的设计，如文本字符串、图像和配色方案。

这些资源存储在 Android 工程的/res 目录下，并且严格（理由充分时也可调整）存储于各自的子目录和文件中。所有资源文件的名称必须为小写且简单（仅由字母、数字和下画线构成）。

表 6.2 显示了 Android SDK 支持的资源类型以及它们在工程中的存储方式。

表 6.2 不同资源类型在 Android 工程资源目录中的存储方式

资源类型	所需的目录	文件名	XML 标记
字符串	/res/values/	strings.xml（推荐）	<string>
复数字符串	/res/values/	strings.xml（推荐）	<plurals>、<item>
字符串数组	/res/values/	strings.xml（推荐）	<string-array>、<item>
布尔型	/res/values/	bools.xml（推荐）	<bool>
颜色	/res/values/	colors.xml（推荐）	<color>
颜色状态表	/res/color/	例如:buttonstates.xml、indicators.xml	<selector>、<item>
尺寸	/res/values/	Dimens.xml（推荐）	<dimen>
整型	/res/values/	integers.xml（推荐）	<integer>

Download at http://www.pin5i.com/

续表

资源类型	所需的目录	文件名	XML 标记
整型数组	/res/values/	integers.xml（推荐）	<integer-array>、<item>
混合类型数组	/res/values/	Arrays.xml（推荐）	<array>、<item>
简单 Drawable 图形	/res/values/	drawables.xml（推荐）	<drawable>
图像	/res/drawable/	例如:icon.png、logo.jpg	支持的图想文件或 XML 文件定义的 Drawable 图形
补间动画	/res/anim/	例如:fadesequence.xml、spinsequence.xml	<set>、<alpha>、<scale>、<translate>、<rotate>
逐帧动画	/res/drawable/	例如：sequence1.xml、sequence2.xml	<animation-list> 、<item>
菜单	/res/menu/	例如： mainmenu.xml 、helpmenu.xml	<menu>
XML 文件	/res/xml/	例如:data.xml、data2.xml	由开发人员定义
原始文件	/res/raw/	例如： jingle.mp3、 some video.mp4、helptext.txt	由开发人员定义
布局	/res/layout/	例如:main.xml、help.xml	多种定义，但必须为布局控件
样式和主题	/res/values/	styles.xml、themes.xml（推荐）	<style>

Tip

某些资源文件，如动画文件和图形，是根据它们自己的文件名来进行引用的（与文件名后缀无关），所以你需要适当地为这些文件命名。

6.1.4 存储不同的资源类型

aapt 贯穿所有在/res 目录下正确格式化的文件，并且在源代码目录/src 中生成类文件 R.java，以访问所有变量。

在本章的稍后部分，我们将详细地讲解存储和使用不同类型资源的方法，但目前，你只需要理解不同类型的资源会以不同的方式进行存储。

1. 存储 String 等简单资源类型

简单的资源类型，例如字符串、颜色、尺寸等，以 XML 文件的方式存储于/res/values 工程目录下。每一个/res/values 目录下的资源文件必须以下面的 XML 头（header）作为文件的起始：

```
<?xml version="1.0" encoding="utf-8"?>
```

接着是根节点<resources>，然后紧跟着定义特定的资源元素，如<string>或<color>。每一个资源均由不同的元素名进行定义。

虽然 XML 文件的命名具有一定的随意性，但最好还是将不同类型的资源存储于不同的

Download at http://www.pin5i.com/

文件中，例如 strings.xml、colors.xml 等。当然，如果开发人员愿意的话，他们也可以为相同类型的资源创建多个文件来存储，例如两个分别称为 bright_colors.xml 和 muted_colors.xml 的 XML 文件。

2. 存储图形、动画、菜单和文件

不仅仅局限于在/res/values 目录下存储简单的资源类型，你也可以存储大量其他类型的资源，如动画、图像、任意 XML 文件和原始文件。但是这些类型的资源不存储在/res/values 目录下，而是根据它们的类型分别存储于不同的文件夹下。例如，你可以将动画序列存储于/res/anim 目录中。需要提醒读者的是，一定要确保你的这些资源文件被适当命名，因为资源名称继承自特定资源的文件名。例如，在资源目录/res/drawable 中的一个名为 flag.png 的文件会被命名为 R.drawable.flag。

3. 理解资源的解析（resolve）方式

几乎没有应用程序能够在任何环境下百分之百完美地运行。绝大部分应用程序需要处理一些特殊情况，这就需要“替代资源”。你可以根据十几个不同类型的标准来组织 Android 工程所使用的资源，这些标准包括语言和地区、屏幕特点、设备模式（夜间模式、停驻（docked）模式等）、输入方式以及其他许多设备间的差异。

把存储在资源目录顶层的资源用作默认资源，而把那些特定版本的资源当作替代资源，这样很有助于我们的理解。通常开发人员使用替代资源有两个目的，一是基于国际化和本地化的考虑，二是应用程序的设计要求它能够流畅地运行在不同的设备屏幕和屏幕方向下。

Android 平台有着一个健壮的机制，可供在运行时载入资源。举个例子，我们假设有一个简单的应用程序，它拥有必需的字符串、图像和布局资源。在这个应用程序中，这些资源存储在资源目录的顶层（例如，/res/values/strings.xml、/res/drawable/myLogo.png 和/res/layout/main.xml）。无论何种 Android 设备（小屏幕或者是大尺寸高分辨率屏幕，英语或汉语、纵向或横向屏幕等），你都可以在上面运行这个应用程序，它载入的资源都是相同的。

回到我们的简单示例应用程序中。我们可以创建一个中文的替代字符串资源，只需要向资源子目录中添加另外一个 strings.xml 文件，名为/res/values-zh/strings.xml（注意-zh 是一个修饰符）。我们还可以针对不同的屏幕像素密度提供不同的 Logo，那么我们需要 3 个 myLogo.png 文件：

- /res/drawable-ldpi/myLogo.png（低密度屏幕）；
- /res/drawable-mdpi/myLogo.png（中密度屏幕）；
- /res/drawable-hdpi/myLogo.png（高密度屏幕）。

最后，我们来看看如何让应用程序适应纵向和横向屏幕。为了有更好的用户体验，我们可以提供两个布局，移动一下控件的位置。

- /res/layout-port/main.xml（横向屏幕载入的布局）；
- /res/layout-land/main.xml（纵向屏幕载入的布局）。

拥有了上面这些替代资源之后，Android 平台会采取以下策略。

① 如果设备的语言设置为中文，那么将使用/res/values-zh/strings.xml，否则使用/res/values/strings.xml。

Download at http://www.pin5i.com/

② 如果设备的屏幕是低密度屏，那么将使用/res/drawable-ldpi/myLogo.png 资源，如果是中等密度，则使用 mdpi 的 drawable 资源，以此类推。

③ 如果设备使用了横屏模式，那么将载入/res/layout-land/main.xml 布局，如果使用的是纵向模式，那么/res/layout-port/main.xml 布局将被载入。

在创建替代资源时，需要注意以下 4 个重要的规则。

① Android 平台始终载入最有针对性并且最恰当的可用资源。如果替代资源不存在，那么使用默认资源。这样，你就需要熟悉目标设备，设计兼容性好的默认资源，并且审慎地添加替代资源。

② 替代资源的命名必须与默认资源完全一致。如果在/res/values/strings.xml 文件中有一个名为 strHelpText 的字符串，那么在/res/values-zh/strings.xml（中文）文件中，它也需要拥有相同的名称。其他类型的资源，如图像、布局等，也遵循相同的规则。

③ 好的应用程序设计应该遵从替代资源使用拥有对应的默认资源而无论使用何种设备这一原则，这样才能保证始终有资源可供载入。仅在你提供了所有类型的替代资源以后，才能够抛弃默认资源（例如，提供了 ldpi、mdpi 和 hdpi 的图像资源，从而理论上覆盖了所有的可能）。

④ 不要过于鲁莽地创建替代资源，因为它们会增加应用程序包的大小，并且可能会对性能造成影响，相反，应尽量将默认资源设计得具有很好的可扩展性和兼容性。例如，一个很好的布局设计可能同时支持横屏模式和纵屏模式，前提是你使用了合适的控件。

我们在这里花了很长的篇幅讲解替代资源，本章接下来的部分将首先讨论如何创建默认资源。在第 25 章“面向不同设备配置和语言”中，我们将探讨如何利用替代资源来使你的应用程序具有对不同设备配置更好的兼容性。

6.1.5 使用程序访问资源

程序开发人员使用 R.java 类文件及其子类（sub-class）访问特定的应用程序资源，它们都是在你向工程中添加资源时自动生成的（如果你使用的是 Eclipse）。你可以根据名字引用到任何一个资源标识符。例如，在/res/values/strings.xml 文件中定义的一个名为 strHello 的字符串资源可以在代码中作如下引用：

```
R.string.strHello
```

这个变量并不是同名为 hello 的字符串的实际数据相关联。相反地，你使用这一资源标识符来获取某个类型的资源（这里为字符串）。

例如，一个简单的获取字符串文本的方法如下所示：

```
String myString = getResources().getString(R.string.strHello);
```

首先，你通过应用程序 Context（android.content.Context）获取 Resources 实例，但由于 Activity 类继承自 Context 类，所以你可以使用上面那样的语句。接着，使用 Resources 实例来获得适当类型的资源。你将发现，Resources 类（android.content.res.Resources）拥有处理每种类型资源的辅助方法（helper method）。

在我们继续深入学习之前，通过一个简单的例子，在其中创建一些资源并进行研究会很有帮助。你不必因为你不能理解这个练习的某些方面内容而感到担心，在本章后面部分你将

Download at http://www.pin5i.com/

学习到每种不同类型资源的更多内容。

6.2 使用 Eclipse 设置简单资源

开发人员可以通过手工编写资源的 XML 文件，使用 aapt 编译它们并生成 R.java 文件来定义资源，当然也可以方便地使用内置于资源编辑器的 ADT 插件。

为了显示使用 Eclipse 插件来设置资源的方法，我们来看看一个例子。创建一个 Andriod 工程并在 Eclipse 中定位至/res/values/strings.xml 文件，双击打开并编辑它。strings.xml 资源文件将在右侧的面板中打开，并大致如图 6.1 所示。

图 6.1 Eclipse 资源编辑器（编辑器视图）中的字符串资源文件

在面板的底部有两个标签。Resources 标签提供插入简单资源类型的简易方法，如插入字符串、颜色和尺寸资源。strings.xml 标签显示了创建的原始 XML 资源文件。有时，直接手动地编辑 XML 文件可能更快速，特别是当需要一次性地添加多个新的资源时。单击 strings.xml 标签，之后面板将类似图 6.2 所示。

图 6.2 Eclipse 资源编辑器（XML 视图）中的字符串资源文件

现在，使用 Resources 标签页下的 Add 按钮添加几个资源。具体地说，添加下面的资源：

① 一个名为 prettyTextColor 的颜色资源，值为#ff0000；

② 一个名为 textPointSize 的尺寸资源，值为 14pt；

③ 一个名为 redDrawable 的 drawable 资源，值为#F00。

现在在你的 strings.xml 文件中已经有了不同类型的几个资源。如果你切换回 XML 视图，你将发现 Eclipse 资源编辑器已经将适当的 XML 元素加入到了文件中，应该如下所示：

Download at http://www.pin5i.com/

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">ResourceRoundup</string>
    <string
        name="hello">Hello World, ResourceRoundupActivity</string>
    <color name="prettyTextColor">#ff0000</color>
    <dimen name="textPointSize">14pt</dimen>
    <drawable name="redDrawable">#F00</drawable>
</resources>
```

保存 strings.xml 文件。Eclipse 插件将自动在你的工程中生成 R.java 文件，并含有适当的资源 ID，以供在编译之后使用程序访问这些资源。如果你定位至程序包中位于/src 目录下的 R.java 文件，它将如下所示：

```
package com.androidbook.resourceroundup;
public final class R {
    public static final class attr {
    }
    public static final class color {
        public static final int prettyTextColor=0x7f050000;
    }
    public static final class dimen {
        public static final int textPointSize=0x7f060000;
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
        public static final int redDrawable=0x7f020001;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int hello=0x7f040001;
    }
}
```

现在，你可以在代码中自由地使用这些资源了。如果你导航至 ResourceRoundup Activity.java 文件，那么你可以像下面这样加入几行代码来获取刚刚添加的资源：

```
import android.graphics.drawable.ColorDrawable;
...
String myString = getResources().getString(R.string.hello);
int myColor =
    getResources().getColor(R.color.prettyTextColor);
float myDimen =
    getResources().getDimension(R.dimen.textPointSize);
ColorDrawable myDraw = (ColorDrawable)getResources().
    getDrawable(R.drawable.redDrawable);
```

某些资源类型，如字符串数组，使用手工直接编辑 XML 来添加它们将会显得更加容易。例如，如果我们回到刚才的 strings.xml 文件，并且选择 strings.xml 标签页，那么我们可以通过插入下面的 XML 元素来向资源列表中加入字符串数组：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
```

Download at http://www.pin5i.com/

```
    <string name="app_name">Use Some Resources</string>
    <string
        name="hello">Hello World, UseSomeResources</string>
    <color name="prettyTextColor">#ff0000</color>
    <dimen name="textPointSize">14pt</dimen>
    <drawable name="redDrawable">#F00</drawable>
    <string-array name="flavors">
        <item>Vanilla</item>
        <item>Chocolate</item>
        <item>Strawberry</item>
    </string-array>
</resources>
```

保存了 strings.xml 文件之后，上面这个名为“flavors”的字符串数组就定义在了源文件 R.java 中，然后你就可以像下面语句所示的这样在 resourcesroundup.java 文件中通过程序使用这一资源了：

```
String[] aFlavors =
    getResources().getStringArray(R.array.flavors);
```

现在，你应该已经对如何使用 Eclipse 插件添加简单的资源有了大概的认识。其他许多不同类型的数据也都可以作为资源进行添加。将不同类型的资源存储在不同的文件中是一种惯例。例如，你可以把字符串存储在/res/values/strings.xml 文件中，但是需要把 prettyTextColor 颜色资源存储在/res/values/colors.xml 文件中，并且把 textPointSize 尺寸资源存储于/res/values/dimens.xml 文件中。重新组织资源层级目录下的资源文件并不会改变资源的名称，也不影响之前编写的代码。

现在我们来看看如何向你的工程中添加各种不同种类的资源。

6.3 使用资源

在这一节中，我们关注 Android 应用程序所能使用的各个具体的资源类型、它们在工程文件中的定义方式，以及如何使用程序访问这些资源数据。

对于每一种类型的资源，你将学习到哪些类型的值可以被存储，以及以怎样的格式存储。某些资源类型（如 Strings 和 Colors）可以在 Android 插件的资源编辑器中进行方便的编辑，同时另一些资源类型（如 Animation 序列）则通过直接编辑 XML 文件本身显得更为容易。

6.3.1 使用 String（字符串）资源

String 资源是开发人员可用的最简单的资源类型之一。String 资源可以用来在窗体中显示文本标签或帮助文字等。应用程序名称默认也作为 string 资源来存储。

String 资源在/res/values 工程目录下定义，并且在构建（build）时编译进应用程序包。所以包含撇号或单引号的字符串需要进行转义或使用双引号括起来。表 6.3 给出了一些格式正确的字符串。

表 6.3 String 资源格式示例

字符串资源值	字符串显示
Hello, World	Hello, World
"User's Full Name:"	User's Full Name:

Download at http://www.pin5i.com/

续表

字符串资源值	字符串显示
User\'s Full Name:	User's Full Name:
She said, \"Hi.\"	She said, "Hi."
She\'s busy but she did say, \"Hi.\"	She's busy but she did say, "Hi."

你可以通过单击 strings.xml 文件并选择 strings.xml 标签页来编辑它，也可以直接编辑 XML 代码。文件保存后，这些资源将自动被添加到 R.java 文件中。

字符串值使用其相对应的<string>标记，并且使用“名称—值”对的方式来定义。“名称”属性将是你在程序中引用某一特定字符串的依据，所以你需要比较讲究地为其命名。

下面是一个简单资源文件/res/values/strings.xml 的例子：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Resource Viewer</string>
    <string name="test_string">Testing 1,2,3</string>
    <string name="test_string2">Testing 4,5,6</string>
</resources>
```

对于字符串资源，你也可以为其添加三种具有 HTML 风格的属性，它们分别是加粗、斜体和下画线。你可以使用<b>、<i>和<u>标记来指定这些风格。例如：

```
<string
    name="txt"><b>Bold</b>,<i>Italic</i>,<u>Line</u></string>
```

6.3.2 使用格式化的字符串资源

你可以创建带有格式化的字符串，但是你需要将加粗、斜体和下画线标记进行转义。例如，下面的文字显示了一个得分和“win（赢）”或“lose（输）”字符串：

```
<string
    name="winLose">Score: %1$d of %2$d! You %3$s.</string>
```

如果你需要向上面的字符串添加加粗、斜体和下画线等格式化属性，那么你需要将标记进行转义。例如，如果你想要让最后的“win”和“lose”字符串变成斜体，那么你的资源定义应该大致如下所示：

```
<string name="winLoseStyled">
    Score: %1$d of %2$d! You&lt;i&gt;%3$s&lt;/i&gt;.</string>
```

如本章之前所述的一样，在程序中访问字符串资源是很直接的。你主要可以采用两种方式来访问字符串资源。

下面的程序将访问应用程序中名为“hello”的字符串资源，并且仅返回字符串本身。所有的 HTML 风格属性（加粗、斜体和下画线）均会从字符串中被剔除。

```
String myStrHello =
    getResources().getString(R.string.hello);
```

你也可以使用另外的方法访问字符串并保留其格式：

```
CharSequence myBoldStr =
    getResources().getText(R.string.boldhello);
```

Download at http://www.pin5i.com/

要想载入一个格式化的字符串，你需要确保所有的格式变量均已被转义。实现这一目的的一种方式是使用 TextUtils.htmlEncode() 方法：

```
import android.text.TextUtils;
...
String mySimpleWinString;
mySimpleWinString =
    getResources().getString(R.string.winLose);
String escapedWin = TextUtils.htmlEncode("Won");
String resultText =
    String.format(mySimpleWinString, 5, 5, escapedWin);
```

存储于 resultText 变量中的返回文本为：

```
Score: 5 of 5! You Won.
```

现在，像前面的 winLoseStyled 字符串资源那样，如果你需要使用风格化的字符串，那么你需要多采取几个步骤来处理转义的“斜体”标记。

```
import android.text.Html;
import android.text.TextUtils;
...
String myStyledWinString;
myStyledWinString =
    getResources().getString(R.string. winLoseStyled);
String escapedWin = TextUtils.htmlEncode(“Won”);
String resultText =
    String.format(myStyledWinString, 5, 5, escapedWin);
CharSequence styledResults = Html.fromHtml(resultText);
```

存储于 styledResults 变量中的返回文本为：

```
Score: 5 of 5! You <i>won</i>.
```

styledResults 这一变量可以用在类似于 TextView 对象的用户界面控件中，这些风格化的文本将在其中获得正确的显示。

Tip

有一种特殊的资源类型，称为<plurals>，它可以用来定义一些具有单复数形式的字符串。例如，你可以为字符串“You caught a goose!”定义一个 plural（复数形式）：“You caught %d geese!”。复数形式的字符串使用 Resource 类的 getQuantityString() 方法载入，而不是 getString() 方法。更多内容可以参考 Android SDK 文档中有关复数元素的部分。

6.3.3 使用字符串数组

你可以在资源文件中指定字符串列表。这是一种存储菜单选项和下拉列表项的好方式。String 数组在/res/values 工程目录的 XML 文件中定义，并且在构建（build）时编译进应用程序包。

String 数组使用其相对应的<string-array>标记和若干<item>子标记，每一个标记里面含有一个字符串。下面是一个简单数组资源文件/res/values/arrays.xml 的示例：

Download at http://www.pin5i.com/

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="flavors">
        <item>Vanilla Bean</item>
        <item>Chocolate Fudge Brownie</item>
        <item>Strawberry Cheesecake</item>
        <item>Coffee, Coffee, Buzz Buzz Buzz</item>
        <item>Americone Dream</item>
    </string-array>
    <string-array name="soups">
        <item>Vegetable minestrone</item>
        <item>New England clam chowder</item>
        <item>Organic chicken noodle</item>
    </string-array>
</resources>
```

如本章前面部分所讲的一样，访问字符串数组的方法非常简单。下面的代码获取了一个名为 flavors 的字符串数组：

```
String[] aFlavors =
    getResources().getStringArray(R.array.flavors);
```

6.3.4 使用 Boolean（布尔）资源

Android 资源层级还支持其他一些基础类型。布尔资源可以用来存储一些应用程序参数或者默认值，如游戏的设定等。布尔资源定义在/res/values 工程目录下的 XML 文件中，并且在构建应用程序时编译。

（1）在 XML 文件中定义布尔资源

布尔值需要使用<bool>标记以“名称—值”对的方式定义。它的名称即是在程序中访问特定布尔值的依据，所以需要起一个有意义的名字。

下面是/res/values/bools.xml 文件中的布尔资源示例：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <bool name="bOnePlusOneEqualsTwo">true</bool>
    <bool name="bAdvancedFeaturesEnabled">false</bool>
</resources>
```

（2）通过程序使用布尔资源

要使用布尔资源，需要通过 Resource 类进行载入。下面的代码访问了应用程序的一个名为 bAdvancedFeaturesEnabled 的布尔资源：

```
boolean bAdvancedMode =
    getResources().getBoolean(R.bool.bAdvancedFeaturesEnabled);
```

6.3.5 使用 Integer（整型）资源

除了字符串和布尔值以外，还可以将整数存储为资源。整型资源定义在/res/values 工程目录下的 XML 文件中，并且在构建时编译到应用程序包中。

（1）在 XML 文件中定义整型资源

整型值需要使用<integer>标记以“名称—值”对的方式定义。它的名称即是在程序中

Download at http://www.pin5i.com/

访问特定整型值的依据，所以需要起一个有意义的名字。

下面是/res/values/nums.xml 文件中的布尔资源示例：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <integer name="numTimesToRepeat">25</integer>
    <integer name="startingAgeOfCharacter">3</integer>
</resources>
```

（2）通过程序使用整型资源

要使用整型资源，你需要通过 Resource 类进行载入。下面的代码访问了应用程序的一个名为 numTimesToRepeat 的整型资源：

```
int repTimes = getResources().getInteger(R.integer.numTimesToRepeat);
```

Tip

与字符串数组类似，也可以使用带有子标记<item>的<integer-array>标记创建整型数组资源，在每一个<item>中定义一个整数。然后，在程序中可以使用 Resource 类的 getIntArray()方法载入。

6.3.6 使用颜色

Android 应用程序可以存储 RGB 颜色值，之后它可以在其他屏幕元素中使用。你可以使用这些值来设置文本和其他元素的颜色，如屏幕背景。颜色资源在/res/values 工程目录的 XML 中定义，并且在构建（build）时编译进应用程序包。

RGB 颜色值始终以井号（#）开头，另外，可以给出 alpha 值以控制透明度。下面给出了一些支持的颜色值格式示例：

① #RGB（例如：#F00 是 12 位颜色，表示红色）；

② #ARGB（例如：#8F00 是 12 位颜色，表示 alpha 为 50%的红色）；

③ #RRGGBB（例如：#FF00FF 是 24 位颜色，表示洋红）；

④ #AARRGGBB（例如：#80FF00FF 是 24 位颜色，表示 alpha 为 50%的洋红）。

颜色值使用其相对应的<color>标记，并且使用“名称—值”对的方式来定义。下面是一个简单的颜色资源文件/res/values/colors.xml 的示例：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="background_color">#006400</color>
    <color name="text_color">#FFE4C4</color>
</resources>
```

本章开头的实例访问了一个颜色资源。颜色资源事实上为一个简单的整数。下面的代码获取了一个称为 prettyTextColor 的颜色资源：

```
int myResourceColor =
    getResources().getColor(R.color.prettyTextColor);
```

6.3.7 使用尺寸

许多用户界面布局控件，如文本控件和按钮，都是按照特定的尺寸进行绘制的。这些尺

Download at http://www.pin5i.com/

寸可以作为资源进行存储。尺寸值始终以一个测量单位作为结束。

尺寸值使用其相对应的<dimen>标记，并且使用“名称—值”对的方式来定义。尺寸资源在/res/values 工程目录的 XML 中定义，并且在构建（build）时编译进应用程序包。

表 6.4 给出了支持的尺寸测量单位。

表 6.4 支持的尺寸测量单位

测量单位	说明	所需的资源标记	示例
像素	实际的屏幕像素	px	20px
英寸	物理测量单位	in	1in
毫米	物理测量单位	mm	1mm
点	普通字体测量单位	pt	14pt
屏幕密度独立像素（density-independent pixels）	相对于 160dpi 屏幕的像素(屏幕兼容的推荐单位)	dp	1dp
比例独立像素（scale-independent pixels）	对于字体显示的测量	sp	14sp

下面是一个简单尺寸资源文件/res/values/dimens.xml 的示例：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="FourteenPt">14pt</dimen>
    <dimen name="OneInch">1in</dimen>
    <dimen name="TenMillimeters">10mm</dimen>
    <dimen name="TenPixels">10px</dimen>
</resources>
```

尺寸资源事实上是简单的浮点值。下面的代码获取了一个称为 textPointSize 的尺寸资源：

```
float myDimension =
    getResources().getDimension(R.dimen.textPointSize);
```

Warning

在为应用程序选取尺寸单位时需要小心，如果你打算兼容具有不同屏幕尺寸和解析度的多种设备，那么你需要依赖于那些更具有普适性的尺寸单位，如 dp 和 sp，而非像素、点、英寸或者是毫米。

6.3.8 使用简单 Drawable 资源

你可以使用 drawable 资源类型指定一个简单的带颜色矩形，它可以在之后应用于其他的屏幕元素中。这些 drawable 类型使用特定的颜色定义，同 Color 资源的定义非常相似。

简单可绘制的 drawable 资源在/res/values 工程目录下的 XML 中定义，并且在构建（build）时编译进应用程序包。可绘制 drawable 资源使用其相对应的<drawable>标记，并且使用“名称—值”对的方式来定义。下面是一个简单 drawable 资源文件/res/values/drawable.xml 的示例：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
```

Download at http://www.pin5i.com/

```
    <drawable name="red_rect">#F00</drawable>
</resources>
```

虽然这将引发一点疑惑，但你也可以创建用于描述其他 Drawable 子类的 XML 文件，如 ShapeDrawable。Drawable 的 XML 定义文件同图像文件一起存储于工程的/res/drawable 目录下。这不同于使用<drawable>存储可绘制 drawable 资源。如前所述，PaintableDrawable 资源存储于/res/values 目录下。

下面是一个用于描述简单 ShapeDrawable 资源的文件/res/values/drawable.xml 的示例：

```
<?xml version="1.0" encoding="utf-8"?>
<shape
    xmlns:android=
        "http://schemas.android.com/apk/res/android"
    android:shape="oval">
    <solid android:color="#f00"/>
</shape>
```

我们将在第 9 章“Android 绘图与动画”中讨论有关图形和绘制形状（drawing shape）的更多内容。

使用<drawable>定义的 Drawable 资源是一个给定颜色的简单矩形，它由 Drawable 子类 ColorDrawable 表示。下面的代码获取了一个名为 redDrawable 的 ColorDrawable 资源：

```
import android.graphics.drawable.ColorDrawable;
...
ColorDrawable myDraw = (ColorDrawable)getResources().
    getDrawable(R.drawable.redDrawable);
```

Tip

还有许多其他的 Drawable 资源可以 XML 资源的方式定义。这些特殊的 Drawable 资源对应着特定的类，如 ClipDrawable 和 LevelListDrawable。有关特殊 Drawable 类型的内容可以参阅 Android SDK 文档。

6.3.9 使用图像

应用程序通常包含如图标和图形等的可视元素。Android 支持几种图像格式，它们可以作为资源直接包含进你的应用程序中。表 6.5 给出了这些图像格式。

表 6.5 Android 支持的图像格式

支持的图像格式	说　明	所需的扩展名
便携式网络图像格式（Portable Network Graphics，PNG）	推荐的格式（无损）	.png
九格拉伸图像（Nine-Patch Stretchable Images）	推荐的格式（无损）	.9.png
联合图像专家组格式（Joint Photographic Experts Group，JPG）	可接受的格式（有损）	.jpg、.jpeg
图形交换格式（Graphics Interchange Format）	不鼓励的格式	.gif

这些图像格式均能够得到主流图形编辑软件如 Adobe Photoshop、GIMP 和 Microsoft Paint 等的很好支持。九格拉伸图形可以使用 draw9patch 工具从 PNG 文件创建，它位于 Android

Download at http://www.pin5i.com/

SDK 的/tools 文件夹内。

Warning

所有的资源文件名必须为小写且简单（仅含有字母、数字和下画线）。这一规则适用于所有文件，包括图形。

向工程中加入图像资源非常简单。你只需要简单地将图像组件拖入/res/drawable 文件夹中，在构建（build）时它们将被自动包含进应用程序包中。

1. 使用九格拉伸（Nine-Patch Stretchable）图像

手机屏幕拥有各种不同的尺寸。想象一个可以根据不同的屏幕尺寸和方向或者不同长度的文字调整大小的图像，它将非常方便而适用。这将极大地节约开发人员为各种不同屏幕尺寸创建图像的时间。

Android 支持九格拉伸（Nine-Patch Stretchable）图像以达到这一目的。九格拉伸图像是拥有小块图像区域的简单 PNG 图像，这些图像区域被定义为可以进行适当的缩放，而非把整个图像作为一个整体进行缩放。通常中间区域为透明。

九格拉伸图像可以通过 Android SDK Tools 目录下的 draw9patch 工具由一个 PNG 文件创建。我们将在第 25 章中讨论更多有关九格拉伸图像的兼容性和使用方法的话题。

2. 通过程序使用图像资源

图像资源仅仅是另一种 Drawable，它称为 BitmapDrawable。大多数时候，你只需要图像的资源 ID 即可设定用户界面控件的属性。

例如，如果把图像文件 flag.png 拖入/res/drawable 目录中，并且向主布局中加入 ImageView 控件，那么我们可以下面的方法通过程序让图像在布局中得以显示：

```
import android,widget.ImageView;
...
ImageView flagImageView =
    (ImageView)findViewById(R.id.ImageView01);
flagImageView.setImageResource(R.drawable.flag);
```

如果你需要直接访问 BitmapDrawable 对象，那么你可以简单地直接获取这一资源，如下所示：

```
import android.graphics.drawable.BitmapDrawable;
...
BitmapDrawable bitmapFlag = (BitmapDrawable)
    getResources().getDrawable(R.drawable.flag);
int iBitmapHeightInPixels =
    bitmapFlag.getIntrinsicHeight();
int iBitmapWidthInPixels = bitmapFlag.getIntrinsicWidth();
```

最后，如果你使用九格图像，那么对 getDrawabel()的调用将返回一个 NinePatch Drawable 对象而不是一个 BitmapDrawable 对象。

```
import android.graphics.drawable.NinePatchDrawable;
...
NinePatchDrawable stretchy = (NinePatchDrawable)
    getResources().getDrawable(B.drawable.pyraimd);
```

Download at http://www.pin5i.com/

```
int iStretchyHeightInPixels =
    stretchy.getIntrinsicHeight();
int iStretchyWidthInPixels = stretchy.getIntrinisicWidth();
```

Tip

还有一个特殊的资源类型，名为<selector>，它可以用来定义某个控件在不同状态下所使用的不同颜色或 Drawable 资源。例如，你可以为 Button 控件定义一个颜色状态表：当按钮无效时为灰色，有效时为绿色，而按下时为黄色。类似地，你也可以为 ImageButton 在不同的状态下提供不同的 Drawable 资源。更多信息可以参考 Android SDK 文档中有关颜色和 Drawable 状态列表的内容。

6.3.10 使用动画

Android 支持逐帧动画和补间（tweening）动画。逐帧动画实际上是连续而快速地显示一个图像序列，而补间动画则对某一幅图像施用简单的变换，如旋转、淡入淡出等。

Android SDK 提供了一些辅助方法供载入和使用动画资源。这些方法包含在 android.view.animation.AnimationUtils 类中。

我们将在第 9 章中详细地讨论动画。而现在，我们只关注如何以资源的方式定义动画数据。

1. 定义和使用逐帧动画资源

当内容逐帧改变时，我们使用逐帧动画，它适用于一些复杂的帧转换动画——很像小孩子连环画。

要定义逐帧动画资源，你可以采取以下步骤。

① 将每个帧的图像存储为独立的 Drawable 资源。将这些图像以序列化的方式命名对以后的使用可能会有所帮助，例如，frame1.png、frame2.png 等。

② 在/res/drawable 资源目录的 XML 文件中定义动画集合资源。

③ 在程序中载入、启动并停止动画。

下面是逐帧动画资源文件/res/drawable/juggle.xml 的一个简单例子，它定义了一个耗时 1.5s 的简单的 3 帧动画。

```
<?xml version="1.0" encoding="utf-8" ?>
<animation-list
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:oneshot="false">
    <item
        android:drawable="@drawable/splash1"
        android:duration="50" />
    <item
        android:drawable="@drawable/splash2"
        android:duration="50" />
    <item
        android:drawable="@drawable/splash3"
        android:duration="50" />
</animation-list>
```

在<animation-list>中定义的逐帧动画资源由 Drawable 子类 AnimationDrawable

Download at http://www.pin5i.com/

来承载。下面的代码提取了一个名为 juggle 的动画资源：

```
import android.graphics.drawable.AnimationDrawable;
...
AnimationDrawable jugglerAnimation = (AnimationDrawable)getResources().
    getDrawable(R.drawable.juggle);
```

在有了一个可用的 AnimationDrawable 对象后，你可以将其关联到屏幕上的一个视图（View），并使用动画方法启动和停止动画。

2. 定义和使用补间（Tweened）动画资源

动画序列并不专属于某个具体的图形文件，也就是说，你可以编写一个序列而后将其应用于各种不同的图形中。例如，你可以让月亮、星星和钻石图像都跳动或摇晃起来，这只需要使用一个缩放动画序列或一个旋转动画序列就可以了。

图形动画序列存储于/res/anim 工程目录下的某个格式化的 XML 中，并且在构建（build）时编译进应用程序包。

下面是一个简单动画资源文件/res/anim/spin.xml 的示例，它定义了一段简单的旋转操作，将目标图像逆时针旋转 4 次，共 10s 完成：

```
<?xml version="1.0" encoding="utf-8" ?>
<set xmlns:android
    ="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <set>
        <rotate
            android:fromDegrees="0"
            android:toDegrees="-1440"
            android:pivotX="50%"
            android:pivotY="50%"
            android:duration="10000"/>
    </set>
</set>
```

让我们回到之前的 BitmapDrawable 示例中，现在我们可以为其加入一点动画元素了。只需要使用下面的代码载入动画资源文件 spin.xml 并启动动画即可。

```
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
...
ImageView flagImageView =
     (ImageView)findViewById(R.id.ImageView01);
flagImageView.setImageResource(R.drawable.flag);
...
Animation an =
    AnimationUtils.loadAnimation(this, R.anim.spin);
flagImageView.startAnimation(an);
```

现在你已经实现了图形旋转。注意到我们现在使用的是基类 Animation 对象来载入动画。你也可以使用某些子类，如 RotateAnimation、ScaleAnimation、TranslateAnimation 和 AlphaAnimation，来实现特定的动画。

在实现补间动画序列时有许多不同的“插补器（interpolator）”可供使用。

Download at http://www.pin5i.com/

6.3.11 使用菜单

你可以在工程文件中包含菜单资源。如同动画资源一样，菜单资源也不专属于某个特定的控件，而可以在任何菜单控件中被重用。

每一个菜单资源（一组独立的菜单项）均存储于/res/menu 工程目录下的某个格式化的 XML 中，并且在构建（build）时编译进应用程序包。

下面是一个简单菜单资源文件/res/menu/speed.xml 的示例，它定义了由按照某个顺序排列的 4 个菜单项组成的短菜单：

```
<menu xmlns:android
    ="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/start"
        android:title="Start!"
        android:orderInCategory="1"></item>
    <item
        android:id="@+id/stop"
        android:title="Stop!"
        android:orderInCategory="4"></item>
    <item
        android:id="@+id/accel"
        android:title="Vroom! Accelerate!"
        android:orderInCategory="2"></item>
    <item
        android:id="@+id/decel"
        android:title="Decelerate!"
        android:orderInCategory="3"></item>
</menu>
```

你可以使用 Eclipse 插件来创建菜单，在其中可以访问每个菜单项的各种配置属性。在上面的例子中，我们设定了每一个菜单项的标题（标签）和显示顺序。现在，你可以为这些标题使用字符串资源，而非把字符串本身敲入代码。例如：

```
<menu xmlns:android=
    "http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/start"
        android:title="@string/start"
        android:orderInCategory="1"></item>
    <item
        android:id="@+id/decel"
        android:title="@string/stop"
        android:orderInCategory="2"></item>
</menu>
```

要访问前面称为/res/menu/speed.xml 的菜单资源，你只需要将下面的代码覆写入应用程序的 onCreateOptionsMenu()方法中：

```
public boolean onCreateOptionMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.speed, menu);
    return ture;
}
```

就是这样。现在如果运行应用程序并且按下菜单按钮，你将会看见上面定义的菜单。还

Download at http://www.pin5i.com/

有其他许多 XML 属性可以关联到菜单项。完整的属性列表可以查看 Android SDK 参考中的菜单资源部分：http://d.android.com/guide/topics/resources/menu-resource.html。在第 7 章“探索用户界面屏幕元素”中我们将学习到有关菜单和菜单事件处理的更多内容。

6.3.12 使用 XML 文件

在工程中你可以包含任意 XML 资源文件。这些 XML 文件均存储于/res/xml 工程目录下，并且在构建（build）时编译进应用程序包。

Android SDK 提供各种支持 XML 操作的包和类。你将在第 10 章“使用 Android 数据和存储 API”、第 11 章“使用内容提供器在应用程序间共享数据”和第 12 章“使用 Android 网络 API”中学习到更多有关 XML 处理的知识。现在，我们创建一个 XML 资源文件并且通过代码访问它。

首先，将一个简单的 XML 文件放入/res/xml 目录下。这里，我们创建包含下面内容的 my_pets.xml 文件：

```
<?xml version="1.0" encoding="utf-8"?>
<pets>
    <pet name="Bit" type="Bunny" />
    <pet name="Nibble" type="Bunny" />
    <pet name="Stack" type="Bunny" />
    <pet name="Queue" type="Bunny" />
    <pet name="Heap" type="Bunny" />
    <pet name="Null" type="Bunny" />
    <pet name="Nigiri" type="Fish" />
    <pet name="Sashimi II" type="Fish" />
    <pet name="Kiwi" type="Lovebird" />
</pets>
```

现在，你可以使用下面的方式将 XML 文件作为一个资源通过程序进行访问：

```
XmlResourceParser myPets =
    getResources().getXml(R.xml.my_pets);
```

最后，要实现对 XML 的访问，可以采取下面的方式提取信息：

```
import org.xmlpull.v1.XmlPullParserException;
import android.content.res.XmlResourceParser;
...
int eventType = -1;
while (eventType != XmlResourceParser.END_DOCUMENT) {
    if(eventType == XmlResourceParser.START_DOCUMENT) {
        Log.d(DEBUG_TAG, "Document Start");
    } else if (eventType == XmlResourceParser.START_TAG) {

        String strName = myPets.getName();
        if(strName.equals("pet")) {
            Log.d(DEBUG_TAG, "Found a PET");
            Log.d(DEBUG_TAG,
                "Name: "+myPets.
                    getAttributeValue(null, "name"));
            Log.d(DEBUG_TAG,
                "Species: "+myPets.
                    getAttributeValue(null, "type"));
```

Download at http://www.pin5i.com/

```
            }
        }
        eventType = myPets.next();
    }
    Log.d(DEBUG_TAG, "Document End");
```

6.3.13 使用原始文件

应用程序也可以将原始文件作为资源进行引用。应用程序可能使用的原始文件包括音频文件、视频文件和其他不被 Android 资源打包工具 appt 支持的格式文件。

所有的原始资源文件均包含在/res/raw 目录下，并且在之后的操作中加入进包中。

Warning

所有的资源文件名必须为小写且简单（仅含有字母、数字和下画线）。这一规则同样适用于原始文件的文件名，即使 Android 工具并不处理这些文件，它们也还是需要被放入应用程序包中。

资源的文件名对于某一个路径而言必须是唯一的，而且应该具有一定的含义，因为文件名（不包含扩展名）将作为访问这些资源的名称。

你可以访问位于/res/drawable 目录下的原始文件资源和其他任何资源(如位图文件、其他任何未使用<resource> XML 方法定义的资源)。下面是打开一个名为 the_help.txt 的文件的一种方法：

```
import java.io.InputStream;
...
InputStream iFile =
    getResources().openRawResource(R.raw.the_help);
```

6.3.14 资源引用

你可以引用资源而非复制它们。例如，你的应用程序可能会需要在多个字符串数组中引用同一个字符串资源。

资源引用使用得最多的地方是在布局 XML 文件中，布局可以引用任意数量的资源来指定它的各种属性，如布局颜色、尺寸、字符串和图像。另一个使用资源引用较多的地方是样式（style）和主题资源。

资源按照下面的格式进行引用：

```
@[optional_package_name:]resource_type/variable_name
```

回顾我们之前使用过的汤名字符串数组。如果我们需要将汤名列表本地化，一种比较好的方式是先为每一种汤名创建单独的字符串资源，然后再将这些字符串资源的引用（而不是文本）存储于字符串数组中。

要达到这一目的，我们在/res/strings.xml 文件中定义字符串资源，如下所示：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Application name</string>
    <string name="chicken_soup"
        >Organic Chicken Noodle</string>
    <string name="minestrone_soup"
```

Download at http://www.pin5i.com/

```
        >Veggie Minstrone</string>
    <string name="chowder_soup"
        >New England Lobster Chowder</string>
</resources>
```

之后，我们就可以在/res/arrays.xml 文件中定义通过名称引用上面字符串的本地化字符串数组了，如下所示：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="soups">
        <item>@string/minstrone_soup</item>
        <item>@string/chowder_soup</item>
        <item>@string/chicken_soup</item>
    </string-array>
</resources>
```

Tip

在尝试保存 arrays.xml 文件之前就要保存 strings.xml 文件，这样，字符串资源（由 appt 识别并包含进 R.java 类中）才会先于引用它们的数组资源而定义，否则，你可能会收到如下错误提示：

Error: No resource found that matches the given name.（错误：没有找到同给定名称匹配的资源）

另外，你还可以为资源指定别名来进行引用。例如，你可以为系统资源“OK”字符串指定一个应用程序中使用的别名，在 strings.xml 资源文件中包含如下代码即可：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string id="app_ok">@android:string/ok</string>
</resources>
```

在本章后面部分你将学习到更多不同的可用系统资源。

Tip

与字符串和整型数组相似，你可以使用带有<item>子标记的<array>标记创建任何类型的资源数组，并且在每一个<item>中定义一个资源。然后你可以使用 Resource 类的 obtainTypedArray()方法载入这些资源的数组。这种资源数组通常用于组织一系列的 Drawable 资源，使得可以通过一次调用载入它们。更多信息可以参阅 Android SDK 文档中有关类型数组（typed array）资源的部分。

6.3.15 使用布局（Layout）

同网页开发人员使用 HTML 极为类似，Android 用户界面设计人员可以使用 XML 来定义 Android 应用程序的屏幕元素和布局。布局 XML 文件使得各个不同的资源能够聚合起来，构成 Android 应用程序的屏幕。布局资源文件均存储于/res/layout 工程目录下，并且在构建（build）时编译进应用程序包。布局文件可以包含多个用户界面控件，还可以定义整个屏幕的布局或描述其他布局中使用的自定义控件。

下面是一个布局文件（/res/layout/main.xml）的简单示例，它设置了屏幕的背景

Download at http://www.pin5i.com/

色并且在屏幕中间显示了一些文字（如图 6.3 所示）。

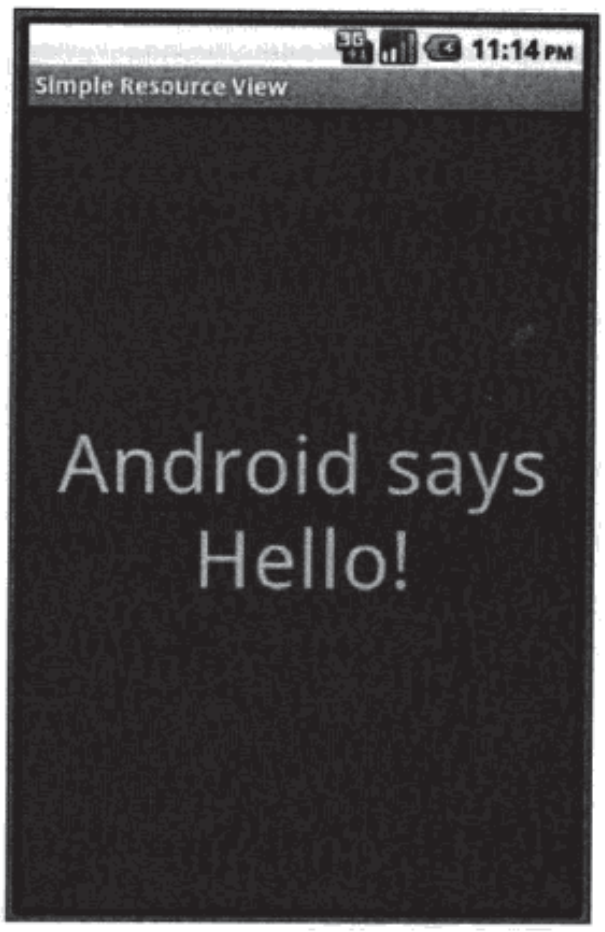


图 6.3 Main.xml 布局文件在模拟器上的显示方式

main.xml 布局文件引用了若干其他的资源，包括颜色、字符串和尺寸值，它们分别在 strings.xml、colors.xml 和 dimens.xml 文件中定义。下面给出了屏幕背景色的颜色资源定义以及 TextView 控件的颜色、字符串和文字尺寸定义：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/background_color">
    <TextView
        android:id="@+id/TextView01"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:text="@string/text_string"
        android:textColor="@color/text_color"
        android:gravity="center"
        android:textSize="@dimen/text_size"></TextView>
</LinearLayout>
```

上面的布局描述了屏幕上所有可见的元素。在这个示例中，一个 LinearLayout 控件被用来作为其他用户界面控件的容器，具体地讲，是一个显示了一行文字的简单 TextView 控件。

Tip

你可以在单独的 XML 文件中定义常用的布局，然后在其他的布局文件中使用<include>标记包含这些布局。例如，你可以在使用下面的<include>标记来包含另一个布局文件/res/layout/mygreenrect.xml:

```
<include layout="@layout/mygreenrect"/>
```

Download at http://www.pin5i.com/

1. 在 Eclipse 中设计布局

布局可以在 Eclipse 中使用 Android 插件提供的资源编辑器进行设计和预览（如图 6.4 所示）。每一个新的 Android 工程都提供/res/layout/main.xml 文件，如果你点击它，你将看见一个 Layout 标签页，其中显示了布局的预览，你还会看见一个 main.xml 标签页，它显示了布局文件的原始 XML 代码。

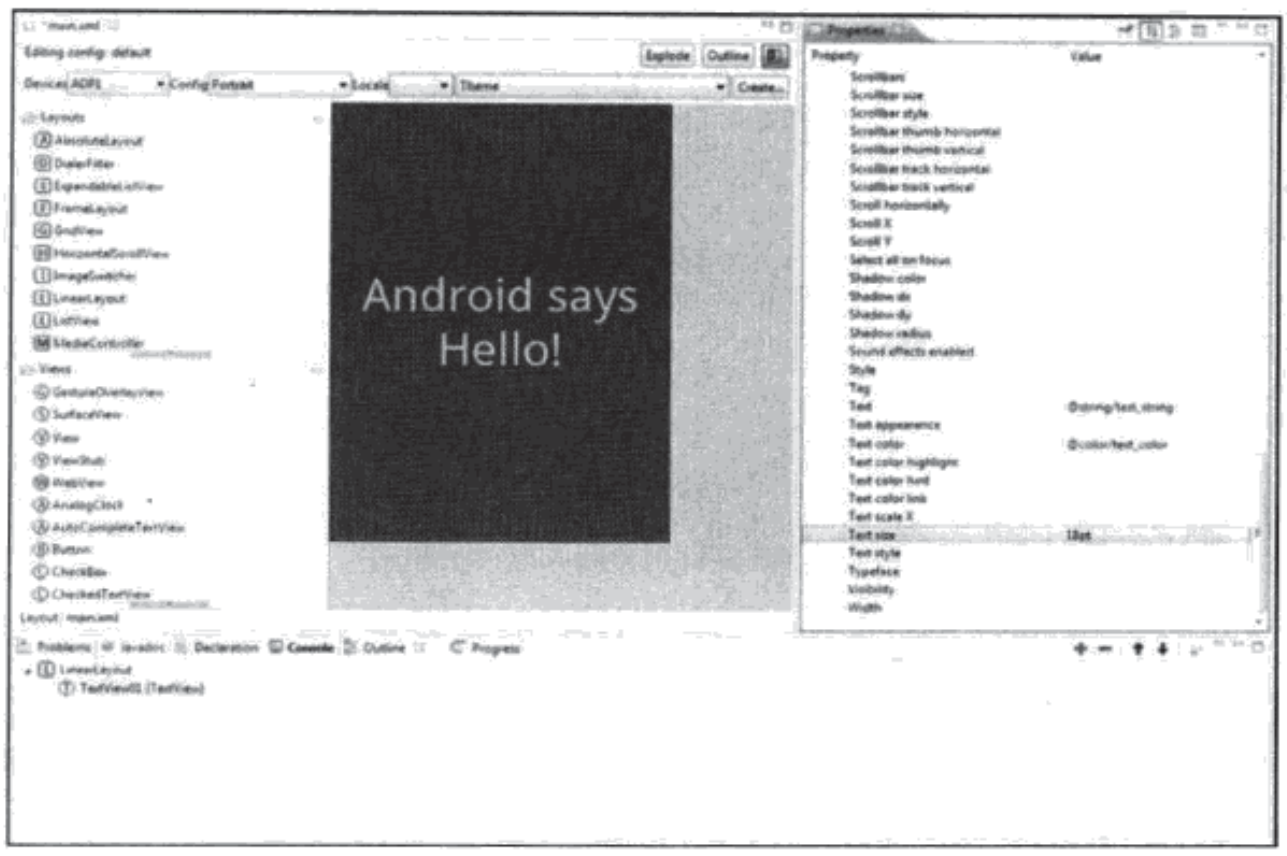


图 6.4 使用 Eclipse 设计布局文件

同大多数用户界面设计器一样，Android 插件能够很好地满足基本的布局需求，可以让你简单地创建用户界面控件，如 TextView 和 Button 控件等，还可以允许你在 Properties（属性）面板中方便地进行属性的设置。

Tip

可以将 Properties 面板移至 Eclipse 工作区域的最右端，这样可以在设计布局时更方便地定位属性并设置它们的值。

现在，我们来熟悉布局资源设计器。创建一个名为 ParisView 的新 Android 工程（它是本书提供的一个实例）。定位至/res/layout/main.xml 布局文件，双击以在资源编辑器中打开它。默认界面非常简单，仅仅拥有一个黑色（空白）的矩形和字符串文本。

在 Eclipse 视图 Resource 面板的底部，你会发现一个 Outline（提纲）标签。这一提纲显示了该布局文件的 XML 层级结构。默认地，你会看见一个 LinearLayout。如果你展开它，你会发现它包含一个 TextView 控件。单击 TextView 控件。Eclipse 视图的 Properties 面板显示出了该控件所有可用的属性。向下拉动滚动条至一个名为 text 的属性处，你将看见它被设置为一个字符串资源变量@string/hello。

Tip

在布局设计器中，你也可以通过单击其他某个布局控件来选择它们。当前选中的控件会被高亮为红色。我们推荐使用 Outline 视图，这样可以确定是否选择了正确的对象。

Download at http://www.pin5i.com/

你可以使用布局设计器来设置和预览布局 Widget 的属性。例如，你可以修改 TextView 的 text Size 属性，键入“18pt”（一种尺寸）。在预览区域中你将可以立即看到所修改属性的结果。

下面我们花一点时间在 main.xml 标签页上。切换至该标签页，你会发现你设置的属性现在显示为了 XML。如果现在你在模拟器中保存并运行工程，你将看见同刚才在设计器预览时看到的类似的结果。

现在回到 Outline 面板中，你将会看见一个绿色的加号按钮和一个红色的减号按钮，可以使用它们向你的布局文件中添加或从中移除控件。例如，在 Outline 视图中选中 LinearLayout，点击绿色的按钮向这一容器对象中加入一个新的控件。

选择 ImageView 对象。现在，在你的布局中就有了一个新的控件，但是你还不能实际地看见它，因为它还没有被完整地定义。

向/res/drawable 工程目录中拖入两个 PNG（或 JPG）图像文件，分别命名为 flag.png 和 background.png。现在，定位至 ImageView 控件的属性，单击资源浏览器中标示为“[...]”的按钮来设置 Src 属性。之后，你可以通过浏览工程中的 Drawable 资源，并选择刚刚添加的“flag”资源，也可以手动地设置这一属性，键入“@drawable/flag”即可。

现在，在预览中你已经可以看见显示的图像了。选中 LinearLayout 对象，设置其 Background 属性为你刚刚添加的“background”Drawable。

如果你保存布局文件并在模拟器或手机中运行应用程序（如图 6.5 所示），你会看见同在资源设计器预览面板中看见的极为相似的画面。

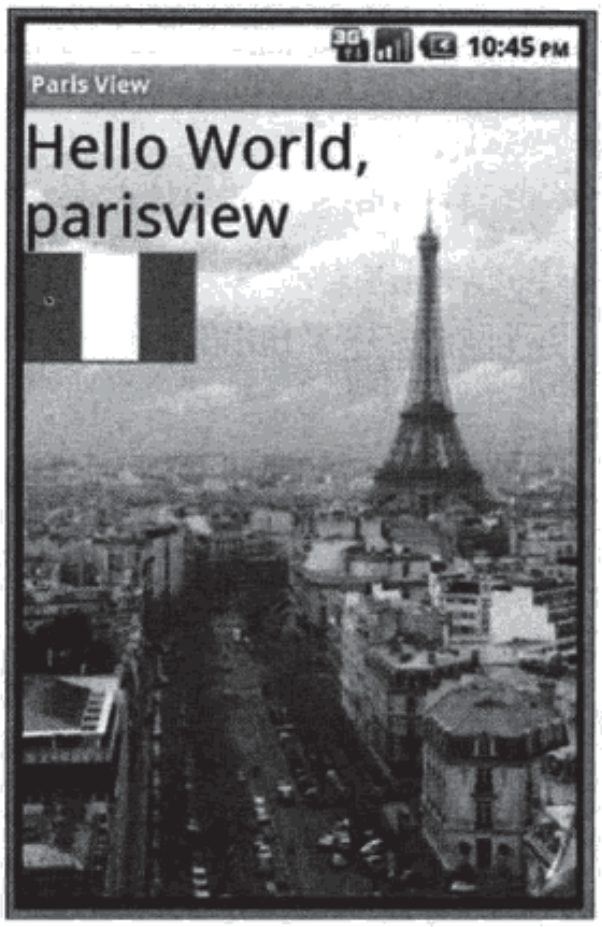


图 6.5 Android 模拟器显示的一个拥有 LinearLayout、TextView 和 ImageView 的布局

2. 通过程序使用布局资源

布局均继承自 View 类，无论它们是 Button 或 ImageViewWidget。下面的代码显示了获取一个名为 TextView01 的 TextView 对象的方法：

```
TextView txt = (TextView)findViewById(R.id.TextView01);
```

你也可以像访问任意 XML 文件一样从后台访问一个布局资源 XML 文件。下面的代码演示了为 XML 解析而获取 main.xml 布局文件的方法：

```
XmlResourceParser myMainXml =
    getResources().getLayout(R.layout.main);
```

开发人员还可以使用唯一的属性来自定义布局。我们将在第 8 章“使用布局设计 Android 用户界面”中探讨有关布局文件和 Android 用户界面设计的更多内容。

Download at http://www.pin5i.com/

Warning

在提供替代布局资源时需要特别小心。布局资源需要进行编译，并且其中包含的子控件通常根据其名称在代码中被引用。这样，如果你创建了一个替代布局资源，那么需要确保每一个重要的控件均存在于布局中，并且名称一致。例如，如果两个布局都拥有一个Button 控件，那么确定其标识符（android:id）在横屏和纵屏布局资源中是相同的。你也可以在布局中包含不同的控件，但那些比较重要的控件（程序中引用并进行交互的控件）应该在两个布局中相同。

6.3.16 使用样式（Style）

Android 用户界面设计人员可以将布局元素属性组合成样式（style）。布局控件均继承自View 基类，它具有许多有用的属性。独立的控件，如 Checkbox、Button 和 TextView，拥有同其行为相关联的特殊属性。

样式使用<style>标记，并且应该存储于/res/values/目录下。样式资源由 XML 定义，并且在构建（build）时编译进应用程序二进制中。

Tip

样式不能使用 Eclipse 资源设计器进行预览，但可以在模拟器中正确地显示。

下面是一个简单样式文件/res/values/styles.xml 的示例，它包含两个应用于TextView 和 EditView 对象的样式，其中之一供“必选”表单域使用，而另一个则提供给“可选”表单域。

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="mandatory_text_field_style">
        <item name="android:textColor">#000000</item>
        <item name="android:textSize">14pt</item>
        <item name="android:textStyle">bold</item>
    </style>
    <style name="optional_text_field_style">
        <item name="android:textColor">#0F0F0F</item>
        <item name="android:textSize">12pt</item>
        <item name="android:textStyle">italic</item>
    </style>
</resources>
```

颜色和尺寸是常见的样式属性。当然，使用资源的引用会更合适一些。下面同样是前面的styles.xml 文件，只不过这次文本颜色和尺寸属性使用的是 colors.xml 和 dimens.xml 资源文件中定义的值：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="mandatory_text_field_style">
        <item name="android:textColor"
            >@color/mand_text_color</item>
        <item name="android:textSize"
            >@dimen/important_text</item>
        <item name="android:textStyle">bold</item>
```

Download at http://www.pin5i.com/

```
    </style>
    <style name="optional_text_field_style">
        <item name="android:textColor"
            >@color/opt_text_color</item>
        <item name="android:textSize"
            >@dimen/unimportant_text</item>
        <item name="android:textStyle">italic</item>
    </style>
</resources>
```

现在，如果你可以分别使用两个 TextView 和 EditView 文本控件创建一个新的布局，那么你可以像如下代码般指定它们的样式属性：

```
style="@style/name_of_style"
```

下面的代码显示了名为/res/layout/form.xml 的表单布局，它可以实现上面提到的功能：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android=
        "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/background_color">
    <TextView
        android:id="@+id/TextView01"
        style="@style/mandatory_text_field_style"
        android:layout_height="wrap_content"
        android:text="@string/mand_label"
        android:layout_width="wrap_content" />
    <EditText
        android:id="@+id/EditText01">
        style="@style/mandatory_text_field_style"
        android:layout_height="wrap_content"
        android:text="@string/mand_default"
        android:layout_width="fill_parent"
        android:singleLine="true" />
    <TextView
        android:id="@+id/TextView02"
        style="@style/optional_text_field_style"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/opt_label"  />
<EditText
        android:id="@+id/EditText02">
        style="@style/optional_text_field_style"
        android:layout_height="wrap_content"
        android:text="@string/opt_default"
        android:singleLine="true"
        android:layout_width="fill_parent" />
<TextView
        android:id="@+id/TextView03"
        style="@style/optional_text_field_style"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

Download at http://www.pin5i.com/

```
            android:text="@string/opt_label"  />
        <EditText
            android:id="@+id/EditText03">
            style="@style/optional_text_field_style"
            android:layout_height="wrap_content"
            android:text="@string/opt_default"
            android:singleLine="true"
            android:layout_width="fill_parent"/>
    </LinearLayout>
```

最终的结果拥有 3 个表单域，每个表单域均由一个用作文本提示标签的 TextView 和供用户输入数据的 EditText 构成。“必选”风格应用在了“必选”标签和文本项的显示当中。另外两个域则使用“可选”风格。显示的结果应该大致如图 6.6 所示。

在第 7 章中我们将讨论有关样式的更多内容。

样式应用于某些特定的布局控件中，如 TextView 和 Button 对象中。通常，当你调用控件的构造函数时，你需要提供样式资源的 id。例如，名为 myAppIsStyling 的样式应该通过 R.style.myAppIsStyling 调用。

图 6.6 使用了两种风格的布局，分别供“必选”和“可选”域显示

6.3.17 使用主题（Theme）

主题（theme）同风格非常相似，但不同于样式一次只能应用于一个布局元素，主题可以应用于给定 Activity 的所有元素（浅显地说，即是整个屏幕）。

主题和样式使用相同的方式定义。主题使用<style>标记，而且应该被存储在/res/values 目录下。主题与样式之间唯一的区别在于主题不是将命名的风格应用于布局元素，而是在 AndroidManifest.xml 文件中定义 Activity 的 theme 属性。

我们将在第 7 章中讨论更多有关主题的内容。

6.4 引用系统资源

除了自己的资源外，应用程序还可以访问系统提供的资源。android 包容纳了各种类型的资源，你可以通过查看 android.R 子类来浏览它们。系统资源涉及以下方面：

① 淡入淡出的动画序列；
② E-mail/电话类型数组（家庭、工作等）；
③ 标准系统颜色；
④ 应用程序图标尺寸；
⑤ 多种常用的 drawable 和布局类型；
⑥ 错误字符串和标准按钮文字；
⑦ 系统风格和主题。

你可以同引用你自己定义的资源一样引用系统资源，将包名设置为 android。例如，要把背景色设置为系统颜色的深灰色，你可以将背景色属性值设为@android:color/darker_gray。你可以如同访问应用程序本身的资源一样访问系统资源，不同之处在于，访问系统资

Download at http://www.pin5i.com/

源需要在 android.R 类下引用 Android 资源包。

如果回顾前面的动画示例，则完全可以使用一段系统动画而不是由我们自己定义。下面是一段与前面示例几乎相同的动画代码，只不过它使用的是系统动画资源来实现淡入的效果：

```
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
...
ImageView flagImageView =
     (ImageView)findViewById(R.id.ImageView01);
flagImageView.setImageResource(R.drawable.flag);
...
Animation an = AnimationUtils.
    loadAnimation(this, android.R.anim.fade_in);
flagImageView.startAnimation(an);
```

Note

Android SDK 安装程序提供所有的默认 Android 资源，它们存放于/platform/<platform_version>/data/res 目录下。在该目录下你可以查看所有 drawable 资源、完整的 XML 布局文件和其他所有存在于 android.R.*包中的资源。

小　结

Android 应用程序依赖于各种资源，包括字符串、字符串数组、颜色、尺寸、drawable 对象、图形、动画序列、布局、样式和主题。另外，原始文件也可以成为资源，许多这类资源由 XML 定义，并且被组织到特别命名的工程目录下。默认资源和替代资源也都被定义在资源的层级目录中。

资源使用 R.java 类文件进行编译或访问，R.java 文件在编译资源时自动生成。开发人员在程序中使用这一特殊类访问应用程序和系统提供的资源。

参考资料和更多信息

Android Dev Guide（开发指南）：Application Resources.http://d.android.com/guide/topics/resources/index.html.

Android Dev Guide：Resource Types.http://d.android.com/guide/topics/resources/available-resources.html.

Download at http://www.pin5i.com/

Download at http://www.pin5i.com/

第三部分

Android 用户界面设计精髓

第 7 章 探索用户界面屏幕元素

第 8 章 使用布局设计用户界面

第 9 章 绘图与动画

Download at http://www.pin5i.com/

第7章

探索用户界面屏幕元素

大多数的 Android 应用程序不可避免地需要某种形式的用户界面。在本章中，我们将讨论 Android 软件开发包（Software Development Kit，SDK）中提供的用户界面元素。其中一些元素用于将信息显示给用户，而另外一些则用于从用户那里收集信息。

除此之外，你还将学习到运用各种不同的组件和控件来构建屏幕方法，以及如何让应用程序监听由用户操作产生的各种事件。最后，你还会学习如何应用控件样式以及怎样将主题应用于整个屏幕。

7.1 Android 视图和布局简介

在我们深入讲解之前，需要定义清楚几个概念。这将有利于在深入学习它们之前，构建起对 Android SDK 提供的特定性能更好的理解。首先，我们先来探讨 View 以及它在 Android SDK 中的地位。

1. Android 视图简介

Android SDK 拥有一个名为 android.view 的 Java 包。该包包含一些同屏幕绘制有关的接口和类。尽管如此，当我们提及 View 对象时，事实上仅仅指包中的一个类，即 android.view.View 类。

View 类是 Android 基本的用户界面构建组块，它代表屏幕的一个矩形区域。View 类是 Android SDK 中几乎所有用户界面控件和布局的基类。

2. Android 控件简介

Android SDK 包含一个名为 android.widget 的 Java 包。当我们提及控件时，通常指该包中的某个类。控件涵盖 Android SDK 中几乎所有可绘制到屏幕上的东西，包括 ImageView、FrameLayout、EditText 和 Button 对象。正如上文提到的，通常所有的控件都是从 View 类继承而来。

本章主要讲解用于显示数据和收集数据的控件。我们将详细地介绍其中许多基本的控件。

Note

不要将 android.widget 包中的用户界面控件与应用程序小插件相混淆。AppWidget 是一种应用程序扩展（extension），通常显示在 Android 的主页中。我们将在第 22 章“扩展 Android 应用程序”中讨论有关小插件的深入话题。

Download at http://www.pin5i.com/

3. Android 布局简介

布局（layout）是 android.widget 包中包含的一种特殊控件。布局控件仍然是 View 对象，但它并不实际地在屏幕上进行绘制。相反，它是其他控件（子控件）的父容器。布局控件依照一定的规则来决定子控件在屏幕上绘制的位置和方法。例如，LinearLayout View 类简单地以横向水平或纵向垂直的方式来绘制它的子控件，类似地，TableLayout 控件则以表格的形式显示其子控件。

在第 8 章“使用布局设计用户界面”中，我们将使用布局和其他容器来组织各种不同的控件。这些继承自 android.view.ViewGroup 类的特殊 View 控件，只有在你理解了容器可以容纳的各种显示控件后才有实用价值。出于需要，我们在本章中会使用到一部分布局 View 对象，以展示如何使用前面提到的那些控件。不过，在下一章之前，我们不会深入讨论 Android SDK 中提供这些布局类型的细节。

Note

本章中的许多代码示例均来自于 ViewSamples 应用程序。在本书的网站上可以下载到 ViewSamples 应用程序的源代码。

7.2 使用 TextView 显示文本

TextView 是 Android SDK 中最基本的用户界面元素或控件。你可以简单地使用它来在屏幕上显示文字。TextView 控件一般用于显示固定长度的文本字符串或标签（label）。

通常，TextView 控件为其他屏幕元素和控件的子控件。同大多数用户界面元素相同，它继承自 View，并且包含于 android.widget 包中。因为它是一个 View，所以一切标准属性，如宽度、高度、边距和可视性等，均适用于该类对象。不过，作为一个文本显示控件，TextView 拥有许多自己特有的属性，它们控制着控件的行为以及文本在不同情况下的显示方式。

首先，我们来看看如何快速地在屏幕上显示文本。用于在屏幕上显示文本的 XML 布局文件使用<TextView>作为其标记，可以在布局文件中将 TextView 的 android:text 属性设置为原始文本字符串，也可以设置为对字符串资源的引用。

下面是你可以用来设置一个 TextView 的 android:text 属性的两种方法的示例。第一种方法将该文本属性设置为原始字符串，而第二种则使用一个名为 sample_text 的字符串资源，该资源必须在 strings.xml 资源文件中予以定义。

```
<TextView
    android:id="@+id/TextView01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Some Sample text here" />
<TextView
    android:id="@+id/TextView02"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/sample_text" />
```

要将这一 TextView 显示在屏幕上，Activity 唯一需要做的事情就是将上面在 XML

Download at http://www.pin5i.com/

文件中定义的布局资源标识符作为参数，调用 setContentView()方法。另外，你也可以调用 TextView 对象的 setText()方法在程序中修改文本的显示，而获取文本则由 getText()方法来完成。

现在，我们来讨论 TextView 对象的更多常用属性。

7.2.1 配置布局和尺寸

TextView 对象也有一些用于控制文字显示和排列的专有属性。例如，你可以将 TextView 的高设置为单行，宽度设置为固定。这样，如果你将一个长字符串放到这个对象中，它将被截断。幸运的是，这个问题是可以通过设置另一些属性来解决的。

> **Tip**
>
> 当你深入研究 TextView 对象的属性时，你应该意识到 TextView 类具有所有仅供文本输入的功能。这意味着许多属性仅仅适用于输入域，而它一般在其子类 EditText 对象中使用。例如，用于帮助用户纠正拼写错误的 autoText 属性，就主要用来设置可编辑文本域（EditText），而在显示文字时则一般无需使用。

TextView 的宽度是由单位 em 控制的，而非像素。em 是使用在印刷业中用于定义字体的点尺寸而使用的术语（例如，在 12 点字体中 1em 等于 12 点）。这一测量方式使得开发人员能够更好地控制文本的大小，而不用关注字体的尺寸。你可以通过 em 属性来设置 TextView 的宽度。另外，你还可以使用 em 单位通过 maxEms 属性和 minEms 属性分别设置 TextView 的最大和最小宽度。

TextView 的高度可以通过指定文本的行数来进行设置，而非使用像素。同 em 单位相似，它也适用于控制文本的显示尺寸，而不用关心字体的尺寸。行数属性用于设置 TextView 可以显示的文本行数。另外，你也可以使用 maxLines 和 minLines 属性分别控制 TextView 显示的最大和最小高度。

下面是结合了以上两种尺寸属性的例子，示例中的 TextView 的高度被设置为 2 行，宽度设置为 12em。布局宽度和高度指定了 TextView 的尺寸，并且需要 XML 规则（schema）中的属性：

```
<TextView
    android:id="@+id/TextView04"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:lines="2"
    android:ems="12"
    android:text="@string/autolink_test" />
```

不同于前面例子中直接将文本进行截尾，我们可以通过设置 ellipsize 属性将文本的最后几个字符替换为省略号（...），这样用户可以知道存在剩余未被显示的文本。

7.2.2 在文本中创建上下文链接

如果你的文本包含 E-mail 地址、网址链接、电话号码或者地址，那么你也许会考虑使用 autoLink 属性（如图 7.1 所示）。autoLink 属性拥有 4 个值，它们可以结合起来使用。当它们可用时，这些 autoLink 的属性值将在应用程序中对这些类型的值创建标准的网页风格的链接。例如，将属性值设为 web 就可以自动地查找 URL 并将其链接到对应的网页。

Download at http://www.pin5i.com/

应用程序可以使用 autoLink 属性的以下几个值。

- **none**：使所有链接无效。
- **web**：使 URL 链接到网页。
- **email**：使 E-mail 地址链接到邮件客户端，并填充收信人地址。
- **phone**：使电话号码链接到拨号程序，自动输入号码，等待拨号。
- **map**：使街道地址与地图应用程序相链接以显示对应的位置。
- **all**：使所有链接有效。

图 7.1 3 个 TextView：简单，开启所有自动链接（不可单击），开启所有自动链接（可单击）

autoLink 特性的开启依赖于 Android SDK 内部对于各种类型的检测。在某些情况下，链接可能会不正确，也可能会被误导。

下面是一个开启了 E-mail 和网页链接功能的例子，而这两个功能是我们认为最可信赖的：

```
<TextView
    android:id="@+id/TextView02"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/autolink_test"
    android:autoLink="web|email" />
```

同样，这一属性也有两个辅助值。你可以将其设为 none 以确定没有数据需要链接，还可以设为 all 以开启所有已知类型的链接，如图 7.2 所示。TextView 默认不链接任何类型。如果你想要用户能够看见这些高亮的数据类型，而又不愿意他们点击，那么可以将 linksClickable 属性设为 false。

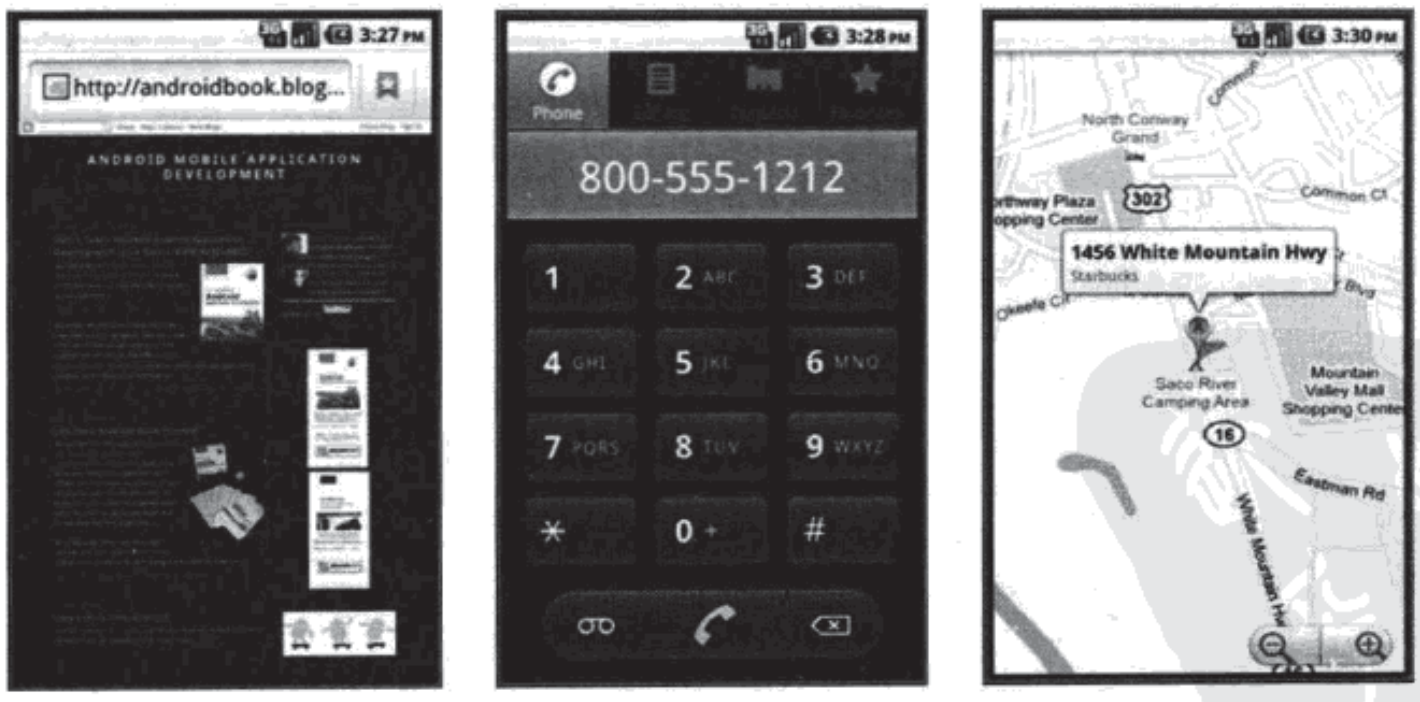


图 7.2 可单击的自动链接：URL 运行浏览器，电话号码启动拨号程序，街道地址链接到 Google Map

7.3 获取用户数据

Android SDK 提供了许多供获取用户输入数据所使用的控件。文本是应用程序最常从用

Download at http://www.pin5i.com/

户那里获取的数据，而 EditText 控件和 Spinner 控件是实现这一功能的最常用控件。

7.3.1 使用 EditText 控件获取用户输入文本

Android SDK 为用户输入提供了一个方便的控件，名为 EditText。EditText 类继承自 TextView。事实上，其大部分的功能均包含于 TextView，只是在作为 EditText 创建时变得可用了。EditText 对象拥有一系列默认可用的特性，图 7.3 显示了其中的若干风格的控件。

首先，我们来看看怎样在 XML 布局文件中定义 EditText 控件：

```
<EditText
    android:id="@+id/EditText01"
    android:layout_height="wrap_content"
    android:hint="type here"
    android:lines="4"
    android:layout_width="fill_parent" />
```

这段布局代码显示了基本的 EditText 元素。另外，可以关注两个有趣的东西。其一，hint 属性会在用户输入文字之前，在编辑框中显示一些文字，以提示用户所需输入的信息。其二，lines 属性设定输入框的行高度。如果没有设置，输入框域会随着用户键入文字而扩大。但是，在设置了一个固定的尺寸之后，用户可以使用滚动条来滚动文字以进行编辑。这一方法也适用于设置输入框的宽度。

默认地，用户可以通过长时单击来触发一个上下文菜单。这可以让用户能够使用一些类似复制、剪切和粘贴的功能，还可以切换输入法，并且将词语添加到用户常用词典中（如图 7.4 所示）。你并不需要为此功能添加任何附加的代码。同样，你也可以使用代码高亮文本的某个部分。调用 setSelection() 方法可以完成这一功能，而调用 selectAll() 则会高亮输入域中的所有文字。

图 7.3 各种风格的 EditText 控件、Spinner 控件和 Button 控件

图 7.4 在 EditText 控件中长时单击通常会触发提供“选择”、“剪切”和“粘贴”功能的上下文菜单

EditText 对象本质上是一个可编辑的 TextView。这意味着其中的文本可以像在 TextView

Download at http://www.pin5i.com/

中一样进行读取，即使用 getText() 方法读取。你也可以使用 setText() 方法在文本域中设置初始文字。当用户编辑一个已经存在数据的表单时，它将非常有用。最后，你可以将 editable 属性设置为 false，这样，用户就不能对文本域中的文字进行编辑，但仍然可以使用长时单击来复制文本。

1. 使用自动补全辅助用户

除了使用 EditText 控件提供基本的文本编辑功能以外，Android SDK 还为方便用户进行常规表单录入提供了一种途径。这一功能是通过自动补全特性来实现的。

自动补全有两种形式。一种是根据用户键入的信息自动将表格补全，这是比较标准的方式。如果用户开始键入某个在开发人员预设的列表中存在的字符串，那么在用户做出了选择的情况下，它将被自动补全。这是通过 AutoCompleteTextView 控件实现的，如图 7.5（左）所示。另一种方法允许用户键入一个项目（item）列表，其中的每一项均包含自动补全功能，如图 7.5（右）所示。这些列表项必须以某种方式被分隔开来，可以通过向处理该方法的 MultiAutoCompleteTextView 对象提供一个 Tokenizer 来实现。一般的 Tokenizer 实现是提供一个由逗号分隔的列表，它被特定的 MultiAutoCompleteTextView.CommaTokenizer 对象所使用。这种方式对于类似指定多项列表这样的功能有很大作用。

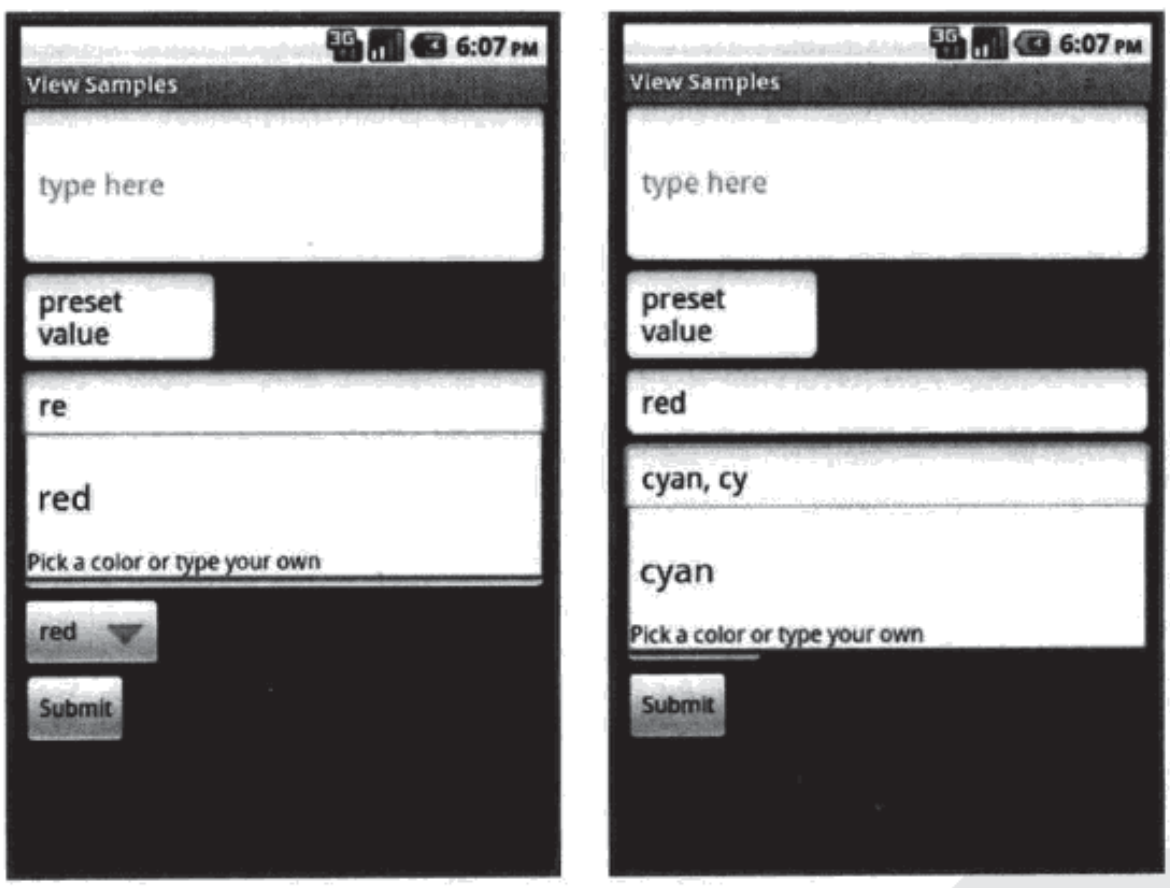


图 7.5 使用 AutoCompleteTextView（左）和 MultiAutoCompleteTextView（右）

两种自动补全文本编辑器均使用一个适配器（adapter）来获得提供给用户进行自动补全的文本列表。下面的例子显示了如何为用户提供一个可以帮助他们从某个数组中选择常用颜色功能的 AutoCompleteTextView：

```
final String[] COLORS = {
    "red","green","orange","blue","purple",
    "black","yellow","cyan","magenta" };
ArrayAdapter<String> adapter =
    new ArrayAdapter<String>(this,
        android.R.layout.simple_dropdown_item_1line,
        COLORS);
```

Download at http://www.pin5i.com/

```
AutoCompleteTextView test =     (AutoCompleteTextView)
    findViewById(R.id.AutoCompleteTextView01);
text.setAdapter(adapter);
```

在这个例子中，当用户在该表单域中键入时，如果他们开始键入 COLORS 数组中的任何一项，那么将会出现一个下拉列表，其中为用户提供了所有可能的自动补全项。需要注意的是这并不会限制用户的键入。用户仍然可以自由地输入任何他们想输入的文字（如“puce”）。适配器将控制下拉列表的外观。在这个示例中，我们使用内建的布局来实现。下面是提供给 AutoCompleteTextView 控件的布局资源定义：

```
<AutoCompleteTextView
    android:id="@+id/AutoCompleteTextView01"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:completionHint="Pick a color or type your own"
    android:completionThreshold="1" />
```

还有两件事情需要注意。第一，你可以通过设置 completionThreshold 属性来选择何时弹出下拉列表。在这个例子中，我们将其设置为单个字符，这样一旦有一个字符匹配时下拉列表将立即弹出。该属性的默认值为两个字符，也就是说当键入两个匹配的字符时，下拉列表才会弹出。第二，你可以给 completionHint 属性设置一些提示性的文字，它将显示在下拉列表的底部以提示用户的输入。最后，下拉列表的尺寸将被限定在 TextView 内。这意味着 TextView 应该足够大，以容纳下拉列表项和提示信息的显示。

MultiAutoCompleteTextView 本质上和普通的自动补全是相同的，除了你需要为其关联一个 Tokenizer 以告诉控件每一个自动补全项的起始位置。下面的例子使用和前面相同的适配器，不同的是为用户颜色列表增加了一个 Tokenizer，两个列表项之间用一个逗号分隔开：

```
MultiAutoCompleteTextView mtext =
    (MultiAutoCompleteTextView)
    findViewById(R.id.MultiAutoCompleteTextView01);
mtext.setAdapter(adapter);
mtext.setTokenizer(new MultiAutoCompleteTextView.CommaTokenizer());
```

正如你所看见的，唯一的改变在于设置了 Tokenizer。这里我们使用的是 Android SDK 提供的内建逗号 Tokenizer。在本例中，无论何时用户从列表中选择一种颜色，这种颜色的名称都将被自动补全，同时会添加一个逗号，这样用户可以迅速开始键入下一个颜色。和之前的一样，它也不会限制用户的输入。如果用户键入了“maroon”并且在其后输入了逗号，那么在用户继续键入下一个颜色名称时自动补全功能又会重新启动，而不会因为“maroon”未被自动补全而造成影响。你可以通过实现 MultiAutoCompleteTextView.Tokenizer 接口创建自定义的 Tokenizer。这样，可以实现由分号或其他更复杂的符号分隔的列表。

2. 使用输入过滤器约束用户输入

在某些情况下，你可能需要对用户的输入进行一些限制。在用户输入完成后进行验证是实现的一种途径。但是，避免浪费用户事件的更好方法是对输入进行过滤。可以为 EditText 控件设置一个 InputFilter 以实现这一功能。

Android SDK 提供了一些可用的 InputFilter 对象。某些 InputFilter 对象的规则是只允许输入大写字母或者对输入文字的长度进行限制。你可以通过实现 InputFilter 接

Download at http://www.pin5i.com/

口来创建自定义的过滤器，而过滤器只包含一个名为 filter() 的方法。下面是一个 EditText 控件的例子，它使用了内建的过滤器，实现了只允许输入两个大写字符的功能：

```
final EditText text_filtered =
    (EditText) findViewById(R.id.input_filtered);
text_filtered.setFilters(new InputFilter[] {
    new InputFilter.AllCaps(),
    new InputFilter.LengthFilter(2)
});
```

setFilters() 方法的调用需要一个 InputFilter 对象数组。这非常适用于像上面所示的那样联合若干个过滤器。在本例中，我们将所有的输入转换成大写。另外，我们还将最大允许的文本长度设置为了两个字符长。EditText 控件看起来和其他的没有任何区别，但当你键入小写字符时，它将被自动转换成大写，而且字符长度也被限定在两个字符以内。这并不意味着所有正确的输入都是有效的，但它的确帮了用户不少忙，至少他们不用担心输入过长，也不用顾及输入字符的大小写。同时，这样的限定虽然并没有使用户只能输入字母，但至少让应用程序能够确保用户的输入不会超过两个字符。输入过滤器也可以在 XML 中予以定义。

7.3.2 使用 Spinner 控件为用户提供可选输入

有些时候你可能需要将用户的输入限制在一个可选择的范围内。例如，如果用户需要输入一个国名，那么我们可以将其限制在一个可选范围内，因为它存在于一个固定的集合中。虽然你可以让用户随意输入，然后“卡”掉那些不正确的，但更好的选择是通过 Spinner 控件提供类似的功能。和自动补全方法类似，单选框（spinner）的所有可能选项也可以来源于某个 Adapter。你还可以在布局定义中使用一个数组资源（确切地说是一个字符串数组的引用，如@array/state-list）设置条目属性，以设定可选范围。单选框控件事实上并不是一个 EditText，虽然它们风格上很相似。下面是一个用于选择颜色的 Spinner 控件的 XML 布局定义：

```
<Spinner
    android:id="@+id/Spinner01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:entries="@array/colors"
    android:prompt="@string/spin_prompt" />
```

上面的代码将在屏幕上放置一个 spinner 控件（如图 7.6 所示）。当用户选中它时，将会弹出一个窗口，其中显示了提示文本，后接可用的选项列表。一次只允许从列表中选择一项，当其中一项被选中后，弹窗将消失。

另外，还有两件事情需要注意。第一，entries 属性需要通过关联一个数组资源来设置显示的选项，这里为@array/colors。

图 7.6 使用 spinner 控件限定选择范围

Note

如何创建数组资源的有关内容可以参考第 6 章“管理应用程序资源”。

Download at http://www.pin5i.com/

第二，prompt 属性定义为一个字符串资源。不同于其他的字符串属性，该属性需要的是一个字符串资源。提示信息将在选项弹窗显示时出现，它可以告诉用户选项的类型是什么。

由于 Spinner 控件并不是一个 TextView，而是一个 TextView 对象的列表，所以你并不能直接向其获取所选项的文本。相反，你需要获取选中的 View，并且直接提取选项文本：

```
final Spinner spin = (Spinner) findViewById(R.id.Spinner01);
TextView text_sel = (TextView)spin.getSelectedView();
String selected_text = text_sel.getText();
```

正如上面代码所显示的那样，你可以获得当前所选中的 View 对象，在这个例子中为一个 TextView。它允许我们提取文本并且直接使用它。另外，我们也可以调用 getSelectedItem() 或者 getSelectedItemId() 方法来处理其他的选择形式。

7.4 使用按钮、多选框和单选组

按钮是另一种常用的用户界面元素。在这一节中，你将学习到 Android SDK 提供的各种不同类型的按钮。它们包括基本的 Button、ToggleButton、CheckBox 和 RadioButton。在图 7.7 中，你可以找到任意一种按钮。

图 7.7 不同种类的按钮控件

基本的 Button 一般用于实现某些类型的操作，如提交表单或确认选择。它们可以包含文本或图像标签。

CheckBox 是一个包含两种状态（选择或不选择）的按钮。CheckBox 控件通常用于开启或关闭某个特性，或者在列表中进行多项选择。

ToggleButton 同 CheckBox 类似，但用于可视化地显示状态。Toggle（开关）的默认行为如同一个电源开关按钮。

RadioButton 提供单项选择的功能。将若干 RadioButton 控件组合在一个称为 RadioGroup 的容器中，可以使开发人员实现单项选择的功能。

7.4.1 使用基本按钮

Android SDK 中的 android.widget.Button 类提供了基本按钮的实现。在 XML 布局资源中，按钮使用 Button 元素来描述。基本按钮的主要属性为文本域。它是显示在按钮中部的标签。你会经常性地使用带有“OK”、“Cancel”或“Submit”字样的基本 Button 控件。

Tip

在 Android 系统资源字符串中，你可以找到许多常用的应用程序字符串值，它们包含在 android.R.string 中。其中有许多常用的按钮文字，如“yes”、“no”、“ok”、“cancel”、“copy”。第 6 章中包含有关系统资源的更多内容。

下面的 XML 布局资源文件显示了一个典型的 Button 控件定义。

```
<Button
    android:id="@+id/basic_button"
```

Download at http://www.pin5i.com/

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Basic Button" />
```

不含有处理单击事件代码的按钮本身并不能做任何事情，除了一点动画。下面是一个基本按钮的单击事件处理代码，它将在屏幕上显示一个 Toast 消息：

```
setContentView(R.layout.buttons);
final Button basic_button = (Button) findViewById(R.id.basic_button);
basic_button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Toast.makeText(Buttons.this,
            "Button clicked", Toast.LENGTH_SHORT).show();
    }
});
```

Tip

Toast（android.widget.Toast）是一个类似对话框的简单消息框，它显示一两秒钟即消失了。Toast 消息通常用于向用户提供不太重要的确认信息，在调试使用它也非常方便。

要想在某个按钮按下时处理单击事件，我们首先需要通过资源标识符获得它的引用。下一步，调用 setOnClickListener()方法。这需要一个有效的 View.OnClickListener 类的实例。提供该实例的一种简单途径是在方法调用中定义一个。这需要实现 onClick()方法。在 onClick()方法中，你可以任意地实现需要的动作。这里，我们只简单地向用户显示一条信息，以告知用户按钮的确被单击了。

主标签为一个图像的按钮为 ImageButton。ImageButton 在大多数情况下与基本按钮相同。单击事件采用相同的方法进行处理。它们之间的主要区别在于你可以将 ImageButton 的 src 属性设置为一幅图片。下面是在 XML 布局资源文件中定义 ImageButton 的例子：

```
<ImageButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/image_button"
    android:src="@drawable/droid" />
```

在这个例子中，引用了一个很小的 drawable 资源。你可以在图 7.7 中看到这个 Android 按钮（它位于 Basic Button 的右侧）。

7.4.2 使用多选框和开关（Toggle）按钮

多选框按钮通常用在可以进行多项选择的列表中。Android 多选框包含一个文本属性，它将显示在勾选框的旁边。它的使用同基本按钮的标签相似。事实上，它就是一个位于按钮旁边的 TextView。

下面是一个 CheckBox 控件的 XML 布局资源定义：

```
<CheckBox
    android:id="@+id/checkbox"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Check me?" />
```

在图 7.7 中（中间位置）你可以看见这个 CheckBox 是如何显示的。

Download at http://www.pin5i.com/

下面的示例演示了如何在程序中检测选框按钮的状态，并且修改文本标签以反映这种变化：

```
final CheckBox check_button = (CheckBox) findViewById(R.id.checkbox);
check_button.setOnClickListener(new View.OnClickListener() {
    public void onClick (View v) {
        TextView tv = (TextView)findViewById(R.id.checkbox);
            tv.setText(check_button.isChecked() ?
                "This option is checked" :
                "This option is not checked");
    }
});
```

这同基本按钮是相似的。多选框会根据其可用或不可用的状态而自动地进行显示。这使得我们可以在应用程序中处理其行为，而不必过多地担心按钮本身的行为。由布局定义我们可以发现文本起初显示一定的内容，而当用户按下按钮之后，它变成显示另外两个字符串，具体的内容由其状态确定。正如代码所示，CheckBox 也同样是一个 TextView。

Toggle 按钮在行为上同多选框相似，但通常用来显示或修改某些东西的开/关状态。如同 CheckBox，它也包含一个状态（选择或未选择）。同样与 CheckBox 一致的是我们仍然不必处理按钮本身显示的改变。但与 CheckBox 不同的是，在它旁边并不显示任何文字。相反，它有两个文本域：第一个是 textOn，表示“开启”状态下按钮上显示的文字；另一个属性是 textOff，表示“关闭”状态下按钮上所显示的文字。这两个属性的默认文本分别为“ON”和“OFF”。

下面的布局代码显示了一个开关按钮的定义，它能够根据其自身状态决定显示“Enabled”或“Disabled”：

```
<ToggleButton
    android:id="@+id/toggle_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Toggle"
    android:textOff="Disabled"
    android:textOn="Enabled" />
```

这一类型的按钮并不实际地显示其 text 属性的值，即使它被正确地进行了设置。从某种意义上说，它的唯一作用在于强调其不显示。你可以在图 7.7 中找到 ToggleButton，并查看它的外观。

7.4.3 使用 RadioGroup 和 RadioButton

单选按钮（Radio Button）通常用于只允许用户从某个小选项组中选择一个选项的情况。例如，对于询问性别的问题可能会有三个选项：男、女和未指定。一次只能够允许选择其中的一项。RadioButton 对象同 CheckBox 对象相似。RadioButton 拥有一个显示在旁边的文本标签，这一文本标签可以通过 text 属性进行设置，同时，它还有自己的状态（选择或未选择）。但是，RadioButton 对象可以被分组到 RadioGroup 中，它将限制它们的组合状态，以满足只允许其中之一被选择的要求。如果用户选中了一个已经处于选择状态的 RadioButton，它并不会变为未选择状态。不过，你可以为用户提供一种途径，以清除整个 RadioGroup 的状态，这样，所有的按钮都将处于未选择状态。

下面的 XML 布局文件定义了一个包含 4 个 RadioButton 对象（如图 7.7 所示）的

Download at http://www.pin5i.com/

RadioGroup。这些 RadioButton 对象拥有自己的文本域，如“Option 1”等。下面是这个 XML 布局资源定义的代码：

```
<RadioGroup
    android:id="@+id/RadioGroup01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    <RadioButton
        android:id="@+id/RadioButton01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Option1"></RadioButton>
    <RadioButton
        android:id="@+id/RadioButton02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Option2"></RadioButton>
    <RadioButton
        android:id="@+id/RadioButton03"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Option3"></RadioButton>
    <RadioButton
        android:id="@+id/RadioButton04"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Option4"></RadioButton>
</RadioGroup>
```

RadioButton 对象的动作是通过 RadioGroup 对象来实现的。下面的示例演示了处理在 RadioGroup 内的 RadioButton 对象的单击事件：

```
final RadioGroup group = (RadioGroup)findViewById(R.id.RadioGroup01);
final TextView tv = (TextView)
    findViewById(R.id.TextView01);

group.setOnCheckedChangeListener(new
    RadioGroup.OnCheckedChangeListener() {
        public void onCheckedChanged(
            RadioGroup group, int checkedId) {
            if (checkedId != -1) {
                RadioButton rb = (RadioButton)
                    findViewById(checkedId);
                if (rb != null) {
                    tv.setText("You chose: " + rb.getText());
                }
            } else {
                tv.setText("Choose 1");
            }
        }
});
```

如同布局示例所呈现的，正确地构造 RadioGroup 及其内部的 RadioButton 并不需要特别的操作。上面的代码也演示了如何进行注册（register）以便能够在 RadioButton 选项改变时接收到通知。

Download at http://www.pin5i.com/

从上面的代码可以看出，通知包含了被选中的某个特定 RadioButton 的资源标识符，它定义在布局文件中。要让其发挥作用，你需要提供与该资源标识符（或文本标签）相关联的东西。在这个例子中，我们查询到了被选择的按钮，获得了它的文本，并且将其关联至屏幕上的另一个 TextView 控件。

正如上文所提及的，整个 RadioGroup 可以被清理，这样所有 RadioButton 均处于未选择状态。下面的示例演示了如何通过单击 RadioGroup 外部的按钮来实现这一功能：

```
final Button clear_choice = (Button) findViewById(R.id.Button01);
clear_choice.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        RadioGroup group = (RadioGroup)
            findViewById(R.id.RadioGroup01);
        if (group != null) {
            group.clearCheck();
        }
    }
```

对 clearCheck()方法的调用将触发 onCheckedChangedListener()回调方法的调用。这就是为什么我们需要确定所接收到的资源标识符是有效的。一旦对 clearCheck()方法的调用结束，它将不再是一个有效的资源标识符，相反它将被设为-1 以表示没有 RadioButton 当前处于选择状态。

7.5 获取用户输入的日期和时间

Android SDK 提供了两种获取用户设置的日期和事件的控件。第一个是 DatePicker 控件（如图 7.8 上部所示）。它可以用来从用户处获取年、月、日。

图 7.8 日期和时间控件

下面是一段基本的 DatePicker 的 XML 布局资源定义：

```
<DatePicker
    android:id="@+id/DatePicker01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

从这个示例中可以看到，我们并没有为 DatePicker 控件指定任何属性。同许多其他的控件一样，你可以在代码中注册以获得在日期改变时的方法调用。这可以通过实现 onDateChanged()方法来完成。不过，它并不是以普通的方式来实现的：

```
    final DatePicker date = (DatePicker)findViewById(R.
id.DatePicker01);
    date.init(date.getYear(), date.getMonth(), date.
getDayOfMonth(),
        new DatePicker.OnDateChangedListener() {
            public void onDateChanged(DataPicker view, int year,
                int monthOfYear, int dayOfMonth) {
                    Date dt = new Date(year-1900,
                        monthOfYear, dayOfMonth, time.getCurrentHour(),
                        time.getCurrentMinute());
```

Download at http://www.pin5i.com/

```
                text.setText(dt.toString());
            }
        });
```

上面的代码通过调用 DatePicker.init() 方法设置了 DatePicker.OnDateChangedListener。DatePicker 控件使用当前日期进行初始化。在用户向 DatePicker 控件输入日期值时，TextView 将被设置为该值。年份参数将被减去 1900 以使其同 java.util.Date 类兼容。

TimePicker 控件（如图 7.8 所示）同 DatePicker 控件相似。它也同样不含有任何属性。不过，要注册当其值改变时调用的方法，你可以通过调用比较传统的 TimePicker.setOnTimeChangedListener() 方法来实现。

```
time.setOnTimeChangedListener(new TimePicker.OnTimeChangedListener() {
    public void onTimeChanged(TimePicker view,
        int hourOfDay, int minute) {
            Date dt = new Date(date.getYear()-1900, date.getMonth(),
                date.getDayOfMonth(), hourOfDay, minute);
                text.setText(dt.toString());
    }
});
```

同上一个示例一样，这段代码也设置了一个 TextView 来显示用户输入的时间值。当你把 DatePicker 和 TimePicker 联合起来使用时，用户可以设置完整的日期和时间。

7.6 使用指示器向用户显示数据

Android SDK 提供了若干控件，它们可以用来可视化地向用户显示信息。这些指示性的控件包括进度条、时钟和其他类似控件。

应用程序经常会执行一些需要消耗一定时间的任务。在此期间，如果能够使用某种进度指示器则会收到很好的效果，它能够告知用户应用程序正在后台做某件事情。另外，应用程序还可以显示用户进行某项操作时所处的位置，例如播放一曲音乐或是观看一个视频。Android SDK 提供几种类型的进度条。

图 7.9 不同种类的进度和评分指示器

标准的进度条是一个仅完成动画的圆圈指示器。它并不指示某项任务的量化完成情况。不过，它确实可以用来表示某件事情正在进行。这在用于指示时长模糊（indeterminate）的任务时非常有用。这种类型的进度指示器有三种不同的尺寸（如图 7.9 所示）。

第二种是水平进度条，它可以显示某项操作的完成情况。例如，你可以查看某个文件已下载了多少。这种水平进度条还可以在其上显示一个次级进度指示器。例如，这可以用来在播放某个媒体文件时显示其下载进度。

下面是一段基本模糊（indeterminate）进度条的 XML 布局资源定义代码：

Download at http://www.pin5i.com/

```
<ProgressBar
    android:id="@+id/progress_bar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

默认的风格是一个中等尺寸的圆圈进度指示器，根本不是一个“条”。另外两种风格的模糊进度条为 progressBarStyleLarge 和 progressBarStyleSmall。这一风格能够自动地进行动画。下面的示例显示了一个水平进度指示器的布局定义：

```
<ProgressBar
    android:id="@+id/progress_bar"
    style="?android:attr/progressBarStyleHorizontal"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:max="100" />
```

我们同时还在这个示例中将 max 属性设置为了 100。这可以用来模拟一个百分比进度条。也就是说，设置进度为 75 表示已完成 75%。

我们可以像下面的代码一样在程序中设置指示器的进度：

```
mProgress = (ProgressBar) findViewById(R.id.progress_bar);
mProgress.setProgress(75);
```

你还可以将这些进度条放进应用程序的标题栏中（如图 7.9 所示）。这可以节省应用程序占用的空间区域，还能够使不改变屏幕外观地开启或关闭某个模糊进度指示器变得更加容易。模糊进度指示器常常用来显示在待载入数据已完成绘制的页面上，这种情况经常发生在网络浏览器的屏幕上。下面的代码演示了如何在你的 Activity 屏幕中放置一个这种类型的模糊进度指示器：

```
requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
requestWindowFeature(Window.FEATURE_PROGRESS);
setContentView(R.layout.indicators);
setProgressBarIndeterminateVisibility(true);
setProgressBarVisibility(true);
setProgress(5000);
```

要在你的 Activity 对象的标题栏中使用这种模拟指示器，你需要像上面所示的这样请求 Window.FEATURE_INDETERMINATE_PROGRESS 特性。这将在标题栏的右侧显示一个很小的圆圈指示器。对于在标题栏背景中显示的水平进度条，你需要请求 Window.FEATURE_PROGRESS 特性。该特性必须在应用程序调用 setContentView()方法前被设为可用，正如上面的例子所显示的。

另外，你需要了解两个重要的默认行为。第一，指示器默认是可见的。像上面的例子那样调用可见性方法可以完成可见和不可见的设置。第二，水平进度条的默认最大进度值为 10000。在前面的例子中，我们将其进度值设置为了 5000，即相当于 50%。当进度值达到最大值时，指示器将淡出消失，这同时适用于两种类型的指示器。

7.7 使用 SeekBar 调整进度

你已经掌握了如何向用户显示进度。那么，如果你想要让用户能够移动指示器，例如在播放媒体文件时移动当前光标位置或者调整音量设置，该怎么办呢？这可以通过 Android

Download at http://www.pin5i.com/

SDK 提供的 SeekBar 控件来实现。它类似于普通的水平进度条，但包含一个图标，或称为调节器（selector），它可以被用户拖动。系统提供带有默认图标的调节器，但任何 drawable 项目均可以作为图标使用。在图 7.9（中部）中，我们将默认的图标替换成了一个小 Android 图形。

下面是一个简单 SeekBar 的 XML 布局资源定义示例：

```
<SeekBar
    android:id="@+id/seekbar1"
    android:layout_height="wrap_content"
    android:layout_width="240px"
    android:max="500" />
```

使用这一示例 SeekBar，用户可以在值为 0 和 500 之间随意拖动图标。虽然它仅仅是一种可视化的实现，但它也可以精确地指示用户所选择的值。要实现这一功能，你可以提供一种 onProgressChanged() 方法的实现，如下所示：

```
SeekBar seek = (SeekBar) findViewById(R.id.seekbar1);
seek.setOnSeekBarChangeListener(
    new SeekBar.OnSeekBarChangeListener() {
        public void onProgressChanged(
            SeekBar seekBar, int progress,boolean fromTouch) {
            ((TextView)findViewById(R.id.seek_text))
                .setText("Value: "+progress);
            seekBar.setSecondaryProgress(
                (progress+seekBar.getMax())/2);
        }
});
```

在上面的示例中，有两点有趣的事情需要注意。第一，fromTouch 参数可以告诉代码值的改变是来源于用户输入，还是像我们在普通 ProgressBar 控件中演示的那样来源于程序。第二，SeekBar 同样允许设置次级进度值。在这个例子中，我们将次级进度值设置为进度条用户选择值和最大值之间的中间值。你可以使用这一特性来指示视频播放和缓冲流的进度。

7.7.1 使用 RatingBar 显示评分数据

虽然 SeekBar 允许用户进行数值设定，如音量，但是 RatingBar 却拥有更特殊的作用：显示评分或从用户那里获得评分。默认地，这种进度条显示五个五星图案。用户可以在其上进行水平拖动来实现评分的设置。程序代码也可以进行值的设定。但是，次级指示符不能使用，因为它为这一特定的控件内部所使用。

下面是一个拥有四颗星的 RatingBar 的 XML 布局资源定义：

```
<RatingBar
    android:id="@+id/ratebar1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:numStars="4"
    android:stepSize="0.25" />
```

这种对于 RatingBar 的布局定义既设置了五星的数量，又指定了每个评分之间的增量值。这里，用户可以在 0 和 4.0 之间选择任何评分值，但其步长，也就是 stepSize，只能为 0.25。例如，用户可以设置值为 2.25。这对用户是默认可见的，表现为五星被部分地填充。图 7.9（中部）显示了 RatingBar 的行为。

Download at http://www.pin5i.com/

虽然值对于用户来讲是可视化地显示出来的，但你也许仍然想要给用户显示一个具体的数值。这可以通过实现 RatingBar.OnRatingBarChangeListener 类的 onRatingChanged() 方法来完成。

```
RatingBar rate = (RatingBar) findViewById(R.id.ratebar1);
rate.setOnRatingBarChangeListenger(new
    RatingBar.OnRatingBarChangeListener() {
        public void onRatingChanged(RatingBar ratingBar,
            float rating, boolean fromTouch) {
                ((TextView)findViewById(R.id.rating_text))
                    .setText("Rating: "+ rating);
    }
});
```

上面的示例显示了如何注册监听器。当用户使用这一控件进行评分设定时，某个 TextView 的值被设定为用户输入的数值评分。需要注意的一点是，不同于 SeekBar，实现的 onRatingChanged() 方法是在修改完成之后被调用的，通常是在用户抬起手指时。也就是说，当用户在五星上拖动设置评分时，这一方法并不会被调用，而是在用户停止触碰控件时获得调用。

7.7.2 使用 Chronometer 显示时间推移

有时候你可能想要显示时间的推移而不是进程的增加。在这种情况下，你可以使用 Chronometer 控件作为一个计时器（如图 7.9 底部所示）。当需要记录用户进行某项操作所消耗的时间，或者在游戏中需要限制时间时，它将会非常实用。Chronometer 控件可以使用文本进行格式化，如下面的 XML 布局资源定义所示：

```
<Chronometer
    android:id="@+id/Chronometer01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:format="Timer: %s" />
```

Chronometer 对象的 format 属性可以用来设置一些显示在时间周围的文字。Chronometer 只有在它的 start() 方法被调用之后才会显示时间及附加的文字。要停止计时，可以调用 stop() 方法。最后，你还可以修改计时器的计时起点。也就是说，你可以将开始计时的时间设为过去的某个时刻，而不是依照计时器本身的设定。setBase() 方法可以实现这样的操作。

Tip

Chronometer 使用 elapsedRealtime() 方法提供基准时间（time base）。向 setBase() 方法传入 android.os.SystemClock.elapsedRealtime() 可以在 0 点处启动 Chronometer。

在接下来的例子中，我们通过计时器的资源标识符将其从 View 中提取出来。然后检测它的基值（计时起点）并将其设为 0。最后，我们启动计时器，它将从这一起点开始计时。

```
final Chronometer timer =
    (Chronometer)findViewById(R.id.Chronometer01);
long base = timer.getBase();
Log.d(ViewsMenu.debugTag, "base = "+base);
timer.setBase(0);
timer.start();
```

Download at http://www.pin5i.com/

Tip

你可以通过实现Chronometer.OnChronometerTickListener接口来监听计时器的改变事件。

7.7.3 显示时间

在应用程序中显示时间往往是不太有必要的，因为 Android 设备会有一个状态条显示当前时间。不过，仍然有两个时钟控件可以用来显示这类信息：DigitalClock 和 AnalogClock 控件。

1. 使用 DigitalClock

DigitalClock 控件（如图 7.9 下部所示）可以根据用户的设置采取标准的数字格式并以紧凑文本的方式显示当前时间。它是一个 TextView，所以所有能够在 TextView 中使用的功能均能够在其上应用，除了修改它的文本。例如，你可以修改它的文字颜色和风格。

默认地，DigitalClock 控件会显示到秒位，并且能够随着时间自动更新。下面是一个 DigitalClock 控件的 XML 布局资源定义：

```
<DigitalClock
    android:id="@+id/DigitalClock01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

2. 使用 AnalogClock

AnalogClock 控件（如图 7.9 下部所示）是一种两个指针的钟盘显示时钟，其中一个指针用于显示“分”，另一个显示“时”。它仍然会根据时间的推移每分钟进行更新。钟盘的显示图像会依据包含它的 View 的尺寸进行缩放。

下面是一个 AnalogClock 控件的 XML 布局资源定义：

```
<AnalogClock
    android:id="@+id/AnalogClock01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

AnalogClock 控件的钟面非常简单。不过你可以通过设定分钟和小时数来改变钟面的指针显示，而且，如果你想它更具有风格的话，还可以指定特定的 drawable 资源来修改钟面。这两种时钟 Widget 均不接受不同于系统时间的显示，也不能显示静态时间。它们只能根据设备所在的时区显示当前的系统时间，所以它们并不显得特别实用。

7.8 为用户提供选项和上下文菜单

在 Android 应用程序中，你需要注意两种特殊的应用程序菜单：选项菜单（options menu）和上下文菜单（context menu）。

7.8.1 使用选项菜单

Android SDK 提供一种方法使得用户可以在应用程序中通过按下菜单按钮来调出一个菜

Download at http://www.pin5i.com/

单（如图 7.10 所示）。你可以在应用程序中使用选项菜单来提供帮助、导航或其他扩展控制，以及配置选项。OptionsMenu 控件可以包含图标、子菜单和键盘快捷方式。

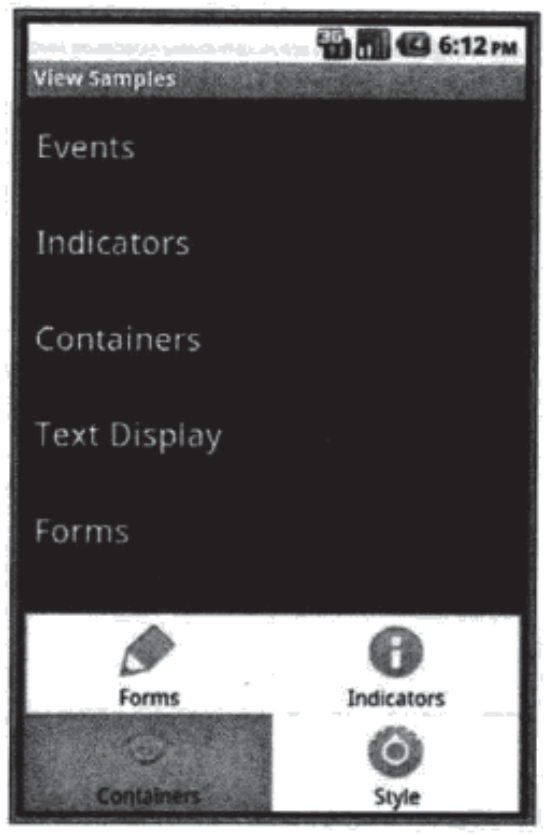


图 7.10 选项菜单

要实现在用户按下 Menu 按钮时弹出选项菜单，你需要覆写 Activity 中 onCreateOptionMenu()方法的实现。下面是一个向用户提供含有三个选项的菜单的例子：

```
    public boolean onCreateOptionsMenu( android.view.
Menu menu) {
        super.onCreateOptionsMenu(menu);
        menu.add("Forms")
            .setIcon(android.R.drawable.ic_menu_edit)
            .setIntent(new Intent(this, FormsActivity.
class));
        menu.add("Indicators")
            .setIntent(new  Intent(this,  Indicators
Activity.class))
            .setIcon(android.R.drawable.ic_menu_info_
details);
        menu.add("Containers")
            .setIcon(android.R.drawable.ic_menu_view)
            .setIntent(new Intent(this, ContainersActivity.class));
        return true;
    }
```

我们为其中的每一个选项设置了一个内建的图标资源，并且关联了一个 Intent。为了清晰起见，我们使用常规的字符串来指定菜单项的标题。当然，使用资源标识符也是可以的。在这个例子中，并不需要其他用户处理菜单动作的代码。当选择其中的任何一个选项时，由其 Intent 所描述的 Activity 将被启动。

这种类型的菜单可以用来导航至应用程序的任何重要部分，例如从任何一个部分跳转至帮助页面。选项菜单的另一个重要用途是针对给定屏幕进行选项设定。这些选项可以以可选（checkable）菜单选项的形式进行配置。用户按下菜单按钮调出的初始菜单并不支持可选菜单项。相反，这些菜单项必须放置在 SubMenu（子菜单）控件中，它是一种在菜单中进行配置的 Menu 控件。SubMenu 对象支持可选菜单项但并不支持图标和其他 SubMenu 菜单项。在上面例子的基础上，下面的代码演示了在程序中向之前的 Menu 添加 SubMenu 控件的方法：

```
SubMenu style_choice = menu.addSubMenu("Style")
    .setIcon(android.R.drawable.ic_menu_preferences);
style_choice.add(style_group, light_id, 1, "Light")
    .setChecked(isLight);
style_choice.add(style_group, dark_id, 2, "Dark")
    .setChecked(!isLight);
style_choice.setGroupCheckable(style_group, true, true);
```

上面的这段代码必须添加在 onCreateOptionsMenu()方法实现的返回表达式之前。它向之前的菜单中添加了一个带有图标的菜单项，称为“Style”。当选中这个“Style”选项时，将显示一个带有两个 SubMenu 控件的弹出式菜单。这两个菜单项被分组在了一起，除此，默认状态下的可选框图标（checkable icon）和单选按钮图标类似，并且其是否处于选择状态是在创建时被设定的。

Download at http://www.pin5i.com/

要处理菜单选项的选择事件，我们仍然实现 onOptionsItemSelected()方法，如下所示：

```
public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == light_id) {
        item.setChecked(true);
        isLight = true;
        return true;
    } else if (item.getItemId() == dark_id) {
        item.setChecked(true);
        isLight = false;
        return true;
    }

    return super.onOptionsItemSelected(item);
}
```

这一方法必须调用父类的 onOptionsItemSelected()方法以进行一些基本的操作。这需要传入一个实际的 MenuItem 对象，我们可以使用它来获取之前赋予的标识符，以确定对应的被选中的菜单项，从而进行相应的操作。这里，我们只改变了值然后返回。默认地，当某一个菜单项被选择后，Menu 控件连同其可选菜单项都会一并消失。这意味着它适用于进行快速设定，而不适用于一次性对多个选项进行设置的扩展设定。

当你向选项菜单中加入了太多菜单项时，具体地说多于 5 个，将会出现一个显示为“More”的菜单项。如果用户选择了它，整个菜单将会被显示出来。完整的菜单不会显示其图标，而且，虽然可选菜单项也会得到显示，但并不能在这里进行设定。另外，菜单项的完整标题（full title）也不会显示。起初的菜单，也就是图标菜单，仅显示菜单标题的一部分。你可以为每个菜单项设置 condensedTitle 属性，这样，它就会显示这一属性的值作为标题，而不是截取之后的原菜单标题。例如，对于标题“Instant Message”，你可以设定它的 condensedTitle 属性为“IM”。

7.8.2 使用 ContextMenu

ContextMenu 是 Menu 的子类型，你可以对其进行配置，这样当用户在 View 上长时单击时它将显示出来。顾名思义，ContextMenu 提供给用户一个针对所选项的上下文关联菜单。

ContextMenu 对象比 OptionsMenu 对象稍微复杂一些。要显示它，你需要实现 Activity 的 onCreateContextMenu()方法。但是，在调用它之前，你必须先调用 registerForContextMenu()方法，并且传入需要在其中显示上下文菜单的 View 对象。这意味着屏幕上的每一个 View 均可以拥有一个上下文菜单，这样可以使菜单与需要进行的操作之间高度关联。

下面我们给出了一个 Chronometer 计时器的例子，它能够对用户长时单击做出反应，并弹出一个上下文菜单：

```
registerForContextMenu(timer);
```

在调用了 registerForContextMenu()方法并执行之后，用户则可以在 View 上进行长时单击，从而打开上下文菜单。每一次进行这样的操作时，Activity 的 onCreateContextMenu()方法都会被调用，其中的代码也都会被执行。

下面是上面所使用的 Chronometer 控件所关联的上下文菜单的例子：

Download at http://www.pin5i.com/

```
public void onCreateContextMenu(
    ContextMenu menu, View v, ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);

    if (v.getId() == R.id.Chronometer01) {
        getMenuInflater().inflate(R.menu.timer_context, menu);
        menu.setHeaderIcon(android.R.drawable.ic_media_play)
            .setHeaderTitle("Timer controls");
    }
}
```

前面我们提到过，当用户进行长时单击时，任何 View 控件都可以注册并触发对 onCreateContextMenu()方法的调用。这意味着我们需要检测用户想调出哪个 View 控件的上下文菜单。下一步，我们根据使用 XML 定义的菜单资源对前面的菜单进行扩充。由于我们不能在菜单资源文件中定义头信息，所以我们为其设置一个 Android SDK 中储备的资源，并且添加一个标题。下面是扩展的菜单资源：

```
<menu
    xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/start_timer"
        android:title="Start" />
    <item
        android:id="@+id/stop_timer"
        android:title="Stop" />
    <item
        android:id="@+id/reset_timer"
        android:title="Reset" />
</menu>
```

上面的代码定义了 3 个菜单项。如果它不是一个上下文菜单，那么我们可以为其设置图标。不过，上下文菜单并不支持图标，也不支持子菜单或者快捷键。第 6 章讲解了更多有关设置菜单资源的内容。

现在，我们需要实现 Activity 的 onContextItemSelected()方法以处理 ContextMenu 的单击事件了。下面给出了示例：

```
public boolean onContextItemSelected(MenuItem item) {
super.onContextItemSelected(item);
boolean result = false;
Chronometer timer = (Chronometer)findViewById(R.id.Chronometer01);
switch (item.getItemId()) {
    case R.id.stop_timer:
        timer.stop();
        result = true;
        break;
    case R.id.start_timer:
        timer.start();
        result = true;
        break;
    case R.id.reset_timer:
        timer.setBase(SystemClock.elapsedRealtime());
        result = true;
```

Download at http://www.pin5i.com/

```
                break;
        }
        return result;
    }
```

因为在这个例子中，我们只有一个上下文菜单，所以在这个方法中，我们找到了Chronometer 视图。无论哪个上下文菜单项被选择，该方法都会被调用，所以，你需要注意资源标识符的唯一性，或者记录下到底是哪一个菜单被显示出来了。由于每次需要显示上下文菜单时它都会被创建，所以这是可以实现的。

7.9 处理用户事件

在前面的一个控件的例子中，你已经接触到了基本事件的处理。例如，你知道当用户单击一个按钮时该如何进行处理。用户可能进行的各种操作将会产生其他的许多事件。这一节将向你简单地介绍其中的一些事件。首先，我们需要讨论 Android 的输入状态。

7.9.1 监听触摸模式状态改变

Android 的屏幕可以处于两种状态中的一种。它的状态决定了如何处理 View 控件上的焦点。当触屏模式开启时，通常只有诸如 EditText 这样的对象才只能在选中时获得焦点。而其他对象，由于它们可以在用户触摸屏幕时被直接选中，所以不必获得焦点就可以触发其行为。当触屏模式关闭时，用户则可以在更多的对象之间切换焦点。它们包括按钮和其他只需要通过单击来触发行为的视图。在这种情况下，用户需要使用方向键、轨迹球或滚轮来在它们之间进行导航，并且通过“Enter”或选择键来选择它们。

当你需要处理某种事件时，获得屏幕当前所处的模式是很有用的。例如，如果你的应用程序依赖于焦点，或者某个特定的控件没有获得焦点时，那么应用程序就需要知道手机设备当前的触屏模式，因为不同模式下的焦点行为是不同的。

应用程序可以使用 android.view.ViewTreeObserver 类中的 addOnTouchModeChangeListener() 方法来注册并监听触屏模式的改变。应用程序需要实现 ViewTreeObserver.OnTouchModeChangeListener 类以监听这些事件的发生。下面是实现它的一个例子：

```
View all = findViewById(R.id.events_screen);
ViewTreeObserver vto = all.getViewTreeObserver();
vto.addOnTouchModeChangeListener(
    new ViewTreeObserver.OnTouchModeChangeListener() {
        public void onTouchModeChanged(
            boolean isInTouchMode) {
            events.setText("Touch mode: " + isInTouchMode);
        }
});
```

在这个例子中，我们获取了位于布局中最顶层的 View。ViewTreeObserver 监听着一个 View 对象以及它所有的子 View 对象。使用布局中高层的 View 意味着 ViewTreeObserver 监听整个布局内的事件。onTouchModeChanged() 方法为 ViewTreeObserver 提供了一个当触屏模式改变时的调用方法。它仅仅需要传入当前 View 所处的模式。

Download at http://www.pin5i.com/

运行上面的示例代码，我们可以发现，一旦用户触及屏幕，触屏模式就会立即改变为 True。相反，当用户选择使用诸如按键、轨迹球和滚轮等其他的输入方法时，一旦用户开始输入，应用程序将会立即报告触屏模式改变为 False。

7.9.2 监听全屏幕上的事件

在前面一个小节中，你已经掌握了如何通过 ViewTreeObserver 来使应用程序能够监听屏幕的触屏模式状态的改变。ViewTreeObserver 还提供对以下三种事件的监听，这三种事件可以发生在全屏幕上，也可以发生在整个 View 以及它的子对象中。

- **PreDraw：** 在 View 及其子对象绘制前收到通知。
- **GlobalLayout：** 当 View 及其子对象的布局可能发生改变时收到通知，这一改变包括可见性的改变。
- **GlobalFocusChange：** 当焦点在 View 及其子对象内发生改变时收到通知。

应用程序可能需要在屏幕进行绘制前做一些工作。在这种情况下，你可以通过 ViewTreeObserver.OnPreDrawListener 类接口的实现，调用 addOnPreDrawListener() 方法来完成这一操作。

类似地，应用程序可以感知到 View 的布局或可见性在什么时候改变了。当你的应用程序需要动态地改变 View 的显示内容，并且需要检测这个 View 是否还很好地匹配屏幕时，它将会非常有用。这时，你的应用程序需要为 ViewTreeObserver 对象的 addGlobalLayoutListener() 方法提供一个 ViewTreeObserver.OnGlobalLayoutListener 类接口的实现。

最后，你的应用程序可以注册以感知在 View 控件及其任何子 View 控件之间焦点的改变。在应用程序需要监视用户在屏幕上的移动时，则可能需要实现它。在处于触屏模式下可能会比不在触屏模式下有较少的焦点改变。在这种情况下，应用程序需要为 addGlobalFocusChangeListener() 方法提供 ViewTreeObserver.OnGlobalFocusChangeListener 类接口的实现。下面是这一实现的一个示例：

```
vto.addOnGlobalFocusChangeListener(new
    ViewTreeObserver.OnGlobalFocusChangeListener() {
        public void onGlobalFocusChanged(
            View oldFocus, View newFocus) {
                if (oldFocus != null && newFocus !=null) {
                    events.setText("Focus \nfrom: " +
                    oldFocus.toString() + " \nto: " +
                    newFocus.toString());
                }
            }
    });
```

这个例子使用了和上一个例子相同的 ViewTreeObserver、vto 和 TextView 事件。它显示了当前获得焦点的 View 和之前获得焦点的 View 均传递给了监听器。从这里开始，你的应用程序就可以执行需要的操作了。

如果你的应用程序仅仅想要在用户修改某个特定的 View 时进行值检测，那么你可能只需要注册以监听该 View 上的焦点改变即可。这将在本章后面部分讨论。

7.9.3 监听长时单击

在前面讲解 ContextMenu 控件的小节中，你已经学习到了如何为一个 View 添加上下

Download at http://www.pin5i.com/

文菜单，这种菜单会在用户在某个 View 上长时单击时被激活。典型的长时单击发生在用户单击触屏并保持接触状态一段时间后。不过，长时单击事件也可以在用户使用如键盘或轨迹球这样的非触摸方法导航至某个位置并按下“Enter”或“Select”键一段时间后触发。这种操作通常也被称为“按住不放（press-and-hold）”操作。

虽然上下文菜单的弹出是长时单击事件的典型应用，但你也可以通过监听长时单击来实现任何你需要的操作。虽然如此，它仍然是和触发上下文菜单相同的事件。所以，如果已经为某个 View 添加了上下文菜单，那么你也许就不能再监听长时单击事件了，否则，执行其他的动作可能会对用户造成混乱，也可能会导致上下文菜单不能正常显示。出于良好的用户界面设计考虑，要始终坚持保障其可用性。

Tip

通常长时单击是对于标准单击的可选操作。也就是说，如果在电脑上单击鼠标左键是标准单击的话，长时单击则相当于单击了鼠标右键。

你的应用程序可以在任何 View 上监听长时单击。下面的示例演示了如何在一个 Button 控件上监听长时单击。

```
Button long_press = (Button)findViewById(R.id.long_press);
long_press.setOnLongClickListener(new View.OnLongClickListener() {
    public boolean onLongClick(View v) {
        events.setText("Long click: " + v.toString());
        return true;
    }
});
```

首先，Button 对象是通过提供其标识符来获取的。之后，通过我们实现的 View.OnLongClickListener 类接口，调用了 setOnLongClickListener()方法。这样，用户长时单击的 View 将会传递到 onLongClick()事件处理器中。这里我们又一次使用了和前面相同的 TextView，以显示长时单击事件发生的文本提示。

7.9.4 监听焦点改变

我们已经讨论过在整个屏幕上对于焦点改变的监听方法。所有的 View 对象都可以在其焦点状态改变时触发一个对其监听器的调用。这是通过向 setOnFocusChangeListener()方法提供一个 View.OnFocusChangeListener 类的实现来完成的。下面的示例演示了如何监听一个 EditText 控件上的焦点改变事件：

```
TextView focus = (TextView)findViewById(R.id.text_focus_change);
focus.setOnFocusChangeListener(new View.OnFocusChangeListener() {
    public void onFocusChange(View v, boolean hasFocus) {
        if (hasFocus) {
            if (mSaveText != null) {
                ((TextView)v).setText(mSaveText);
            }
        } else {
            mSaveText = ((TextView)v).getText().toString();
            ((TextView)v).setText("");
        }
    }
```

Download at http://www.pin5i.com/

在上面的实现中，我们仍然使用了一个 String 类型的私有成员变量 mSaveText。在作为一个 TextView 获取了这一 EditText 之后，我们需要完成两件事情中的一件。如果用户将焦点从这个 EditText 上移开，那么我们将用户刚刚键入的文本转存至 mSaveText 中，同时清除 EditText 中的文本。如果用户将焦点移至它，那么我们恢复其上的数据。对于一个需要填写许多冗长项目的表单，而你又希望能够有一种很好的方法让用户明确他们正在填写的项目，那么这种方法将可以起到很好的作用。它还可以应用在许多场合，例如当用户移开焦点时对输入文本进行验证，或者初次获得焦点时显示一些文本。

7.10 使用对话框

Activity 可以使用对话框来组织信息，并且对用户驱动的事件做出响应。例如，某个 Activity 可以显示一个对话框来告知用户某些信息，或者询问用户是否确定诸如删除数据记录这样的操作。针对简单任务使用对话框可以控制应用程序 Activity 的数量，从而便于管理。

> **Tip**
>
> 本节中的许多代码示例均来自于 SimpleDialogs 应用程序。在本书的网站上可以下载到 SimpleDialogs 应用程序的源代码。

7.10.1 探索不同类型的对话框

Android SDK 提供了若干种不同类型的对话框，每一种都有自己独特的功能。不同的对话框类型包括以下几种。

- **Dialog：** 所有 Dialog 类型的基类。图 7.11 的左上方显示了一个基本的 Dialog。
- **AlertDialog：** 带有一个、两个或三个 Button 控件的 Dialog，如图 7.11 上方中间所示。
- **CharacterPickerDialog：** 用于显示和选择与基本字符相关联的声调字符的 Dialog。图 7.11 的右上方显示了一个 CharacterPickerDialog。
- **DatePickerDialog：** 带有一个 DatePicker 控件的 Dialog，如图 7.11 左下方所示。
- **ProgressDialog：** 带有一个模糊或非模糊 ProgressBar 控件的 Dialog。图 7.11 下方中间显示了一个模糊 ProgressDialog。
- **TimePickerDialog：** 带有一个 TimePicker 控件的 Dialog，如图 7.11 右下方所示。

图 7.11 Android 中可用的不同类型的对话框

Download at http://www.pin5i.com/

如果上面的这些对话框都不能满足你的需求，那么你还可以创建一个自定义的 Dialog，这需要你为其提供特定的布局。

7.10.2 跟踪对话框的生命周期

每一个 Dialog 都必须在包含它的 Activity 中予以定义。一个 Dialog 可能会启动一次，而后重复使用。理解 Activity 是如何管理 Dialog 的对于正确地实现 Dialog 是非常重要的。我们来看看 Activity 用于管理 Dialog 所使用的几个关键方法。

① showDialog()方法用来显示一个 Dialog。

② dismissDialog()方法用来停止一个 Dialog 的显示，但这个 Dialog 仍然驻留在 Activity 的 Dialog 池中。如果再次使用 showDialog()方法显示这个 Dialog，那么处于缓存中的对话框将再次显示出来。

③ removeDialog()方法用于将 Dialog 从 Avtivity 的 Dialog 对象池中移除，这意味着这个 Dialog 之后不再存在了。而如果你再次调用 showDialog()，那么它将被重新创建。

向一个 Activity 中加入一个 Dialog 需要以下几个步骤。

① 在 Activity 内定义一个唯一的 Dialog 标识符。

② 实现 Activity 的 onCreateDialog()方法，一旦提供了唯一标识符，那么该方法将返回适当类型的 Dialog。

③ 实现 Activity 的 onPrepareDialog()方法，它将根据情况初始化 Dialog。

④ 通过唯一标识符，使用 showDialog()方法启动 Dialog。

1. 定义对话框

Activity 所使用的 Dialog 必须提前定义。每一个 Dialog 都拥有一个标识符（一个整数）。在调用 showDialog()方法时需要传入这一标识符。此时，onCreateDialog()方法将被调用，它将返回一个适当类型的 Dialog。

Activity 的 onCreateDialog()方法的实现取决于开发人员，它将为给定的标识符返回适当的 Dialog。如果某个 Activity 拥有多个 Dialog 窗口，那么在 onCreateDialog()方法的内部往往会具有一个 switch 表达式来判断输入的参数，也就是 Dialog 标识符。

2. 初始化对话框

由于 Activity 通常都将 Dialog 保留在自己的 Dialog 池中，所以可能会有必要每次在显示时都重新初始化 Dialog，而非仅仅在创建时初始化一次。为了达到这个目的，你可以覆写 Activity 的 onPrepareDialog()方法。

虽然 onCreateDialog()方法仅仅在最初创建 Dialog 时被调用，但是 onPrepare Dialog()方法却是每次在调用 showDialog()方法时都被调用，这样我们就有机会在每次即将显示 Dialog 时对它进行一些修改了。

3. 启动对话框

你可以通过调用 showDialog()方法并且传递一个有效的 Dialog 标识符（它可以被 onCreateDialog()方法所识别）来显示任何一个定义在 Activity 中的 Dialog。

Download at http://www.pin5i.com/

4. 解散对话框

大多数的对话框都会自行解散（dismiss），但是，如果你想强制解散某个 Dialog，可以简单地调用 dismissDialog() 方法并传入 Dialog 标识符。

5. 移除对话框

解散一个 Dialog 并不会销毁它。如果这个 Dialog 再次被显示，那么缓存中的内容将被显示。如果你想强制 Activity 从它的池中移除某个 Dialog 不再使用，那么可以调用 removeDialog() 方法并且传入一个有效的 Dialog 标识符。

7.10.3 使用自定义对话框

当默认的对话框类型不能满足你的需求时，可以创建一个自定义的 Dialog。创建自定义 Dialog 的一个简单办法是从一个 AlertDialog 开始，使用 AlertDialog.Builder 类覆盖掉默认的布局。为了用这种方法创建一个自定义 Dialog，你需要实现以下几个步骤。

① 设计一个可供在 AlertDialog 中显示的自定义布局资源。

② 在 Activity 中定义一个自定义 Dialog 的标识符。

③ 改写 Activity 的 onCreateDialog() 方法以构建和返回正确的自定义 AlertDialog。你应该使用 LayoutInflator 来延展你的自定义布局以使其适应 Dialog。

④ 使用 showDialog() 方法启动 Dialog。

7.11 使用样式

样式是一组 View 的属性值。这一样式随后可以应用于单独的 View 控件中。样式可以包括诸如字体、文字颜色等的设置。具体的属性依赖于所绘制的 View，虽然本质上来讲，样式属性确实能够改变所绘制对象的面貌。

在本章之前的例子中，你已经学习到了 XML 布局资源文件是如何包含各种属性的引用的，它们将用于控制 TextView 对象的外观。你可以一次性地使用样式来定义应用程序的标准 TextView 属性，然后在 XML 布局文件或 Java 程序中引用这一样式。正如我们在第 6 章中所讲解的，样式通常定义在资源文件 res/values/styles.xml 中。这一 XML 文件包含一个由任意数量的 style 标记构成的 resources 标记，style 标记内的每一个 item 标记指定一项属性及其应用于样式的值。

下面的示例定义了两种不同的样式：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="padded_small">
        <item name="android:padding">2px</item>
        <item name="android:textSize">8px</item>
    </style>
    <style name="padded_large">
        <item name="android:padding">4px</item>
        <item name="android:textSize">16px</item>
    </style>
</resources>
```

Download at http://www.pin5i.com/

当应用了上面的样式后，它将缩进设置为两个像素，并且将 textSize 设置为 8 个像素。下面是该样式在布局资源文件中应用于一个 TextView 的示例：

```
<TextView
    style="@style/padded_small"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Small Padded" />
```

样式支持继承；所以，样式可以引用另外的样式作为其父样式。在这种情况下，它们沿用父样式的属性。下面是使用样式继承的一个示例：

```
<style name="red_padded">
    <item name="android:textColor">#F00</item>
    <item name="android:padding">3px</item>
</style>

<style name="padded_normal" parent="red_padded">
    <item name="android:textSize">12px</item>
</style>

<style name="padded_italics" parent="red_padded">
    <item name="android:textSize">14px</item>
    <item name="android:textStyle">italic</item>
</style>
```

在上面的示例中，你可以发现我们在一个样式中定义了两个常用的属性，并且在另外两个拥有自身属性的样式中引用了它们。任何样式都可以作为父样式被引用；但是，只有一个样式可以被设置为 View 的样式属性。应用上面定义好的 padded_italics 样式可以将文本设置为 14 像素尺寸、斜体、红色并且拥有缩进。下面是应用该样式的一个例子：

```
<TextView
    style="@style/padded_italics"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Italic w/parent color" />
```

在这个例子中你可以发现，应用一个拥有父样式的样式与应用一个普通样式之间没有区别。事实上，一个普通的样式既可以直接应用于 View，也可以作为父样式应用于其他样式。

```
<style name="padded_xlarge">
    <item name="android:padding">10px</item>
    <item name="android:textSize">100px</item>
</style>
<style name="green_glow" parent="padded_xlarge">
    <item name="android:shadowColor">#0F0</item>
    <item name="android:shadowDx">0</item>
    <item name="android:shadowDy">0</item>
    <item name="android:shadowRadius">10</item>
</style>
```

这里，padded_xlarge 样式被设定为 green_glow 样式的父样式。所有的 6 个属性随后将应用于该样式下的任何视图。

Download at http://www.pin5i.com/

7.12 使用主题

主题是一个或多个样式的集合（正如在资源中所定义的那样），但不同于将样式应用于特定的控件，主题则是应用于特定 Activity 内的所有 View 对象。一次性地对一组 View 对象应用一个主题，可以简化统一用户界面风格的过程，同时也提供了一条很好的用于定义配色方案（color scheme）和其他常用控件属性设置的途径。

Android 主题本质上是一种应用于整个屏幕的样式。你可以调用 Activity 的 setTheme()方法在程序中指定主题，而 setTheme()方法需要使用样式资源标识符。这样，如果可用的话，样式中指定的每一个属性将会应用于 Activity 中的每一个 View。不过，在布局文件中显式定义的样式和属性将会覆盖主题中所定义的这些属性。

例如，考虑下面的样式：

```
<style name="right">
    <item name="android:gravity">right</item>
</style>
```

它可以作为主题被应用于整个屏幕。这将使得 Activity 内的任何视图都拥有向右对齐的重力（gravity）属性。应用该主题如同在 Activity 中调用 setTheme()方法一样简单，如下所示：

```
setTheme(R.style.right);
```

另外，你还可以在 AndroidManifest.xml 文件的<activity>元素中，直接将主题作为 Activity 实例的属性指定给它，如下所示：

```
<activity android:name=".myactivityname"
    android:label="@string/app_name"
    android:theme="@style/myAppIsStyling">
```

不同于在 XML 布局文件中应用一种样式，多个主题可以应用于同一个屏幕。这需要容许灵活地定义样式属性，并且需要考虑一种情况，即根据在屏幕上所显示的内容而应用不同的配置。这体现在下面的代码中：

```
setTheme(R.style.right);
setTheme(R.style.green_glow);
setContentView(R.layout.style_samples);
```

在上面的例子中，right 样式和 green_glow 样式均作为主题应用在了整个屏幕上。在屏幕的各种 TextView 上，你既可以看到绿色荧光的文字，也能发现向右对齐的重力效果，如图 7.12 所示。最后，我们将布局应用于 Activity。这必须在设置了主题之后进行。也就是说，所有的主题必须在调用 setContentView()方法或 inflate()方法之前被应用，否则，这些主题所定义的属性将不起作用。

图 7.12 将荧光文字、缩进和对齐融合进一个主题中

经过深思熟虑并且设计良好的主题和样式结合起

Download at http://www.pin5i.com/

来可以让你的应用程序拥有统一的风格，并且易于维护。Android 含有一定数量的内建主题，它们可以成为学习的起点。这些主题包括 Theme_Black、Theme_Light 和 Theme_NoTitleBar_Fullscreen 等，它们均是应用程序所使用的系统主题 Theme 的变体。

小　结

Android SDK 提供了大量用户界面组件，使用它们，开发人员可以创造出更具魅力并且方便易用的应用程序。本章向你介绍了许多非常有用的控件，并分别讨论了它们的行为、风格以及如何处理由用户触发的各种事件。

在本章中，你学习了怎样把控件结合起来形成一个供用户输入的表单。这些用于表单的重要 Widget 包括 EditText、Button、RadioButton、CheckBox 和 Spinner。你还学习到了有关进度指示和时间指示的控件。你还掌握了 Android 应用程序可以使用的各种类型的用户界面构架元素，包括上下文菜单和选项菜单，以及各类对话框。除了在屏幕上绘制控件以外，你还掌握了如何探测用户的行为，例如单击和焦点改变，以及怎样处理这些事件。最后，你还学习了为单独的控件指定样式，以及将主题应用于整个屏幕（或者更具体地说，一个单独的 Activity）的方法，这样，你的应用程序可以始终保持风格上的完全统一。

在本章中，我们探讨了许多有关常用用户界面控件的内容；不过，还有许多我们没有涉及。在第 9 章“Android 绘图与动画”和第 15 章“使用 Android 多媒体 API”中，我们将使用诸如 ImageView 和 VideoView 这样的图像控件来显示可绘制的图形和视频。在下一章中，你将会学习如何简便而精准地使用各种不同的布局和容器控件来组织屏幕上不同种类的控件。

Download at http://www.pin5i.com/

第8章

使用布局设计用户界面

在本章中，我们将讨论如何为 Android 应用程序设计用户界面。这里，我们把焦点集中到不同种类的布局控件上，你可以使用它们通过不同的方法来组织屏幕元素。另外，我们还将谈及一些更复杂的 View 对象，我们称其为“容器视图（container view）”。它们是一些可以包含、容纳其他 View 对象和控件的 View 对象。

8.1 创建 Android 用户界面

应用程序的用户界面可以简单也可以复杂，甚至可以包含许多不同的屏幕。布局和用户界面控件可以定义为应用程序资源，或者在运行时由程序创建。

8.1.1 使用 XML 资源创建布局

正如前面的章节所讨论的，Android 提供了一个简单的方法来创建位于/res/layout 工程目录下的布局文件，它以 XML 的形式被定义为应用程序的资源。这是构建 Android 用户界面，特别是定义预先知道属性的静态屏幕元素和控件最常用、最方便的途径，也适用于设置默认属性，而后在运行时再由程序进行修改。

Warning

Eclipse 布局资源设计器是设计和预览布局资源的有用工具。不过，预览并不能精确地显现最终用户所能在手机设备上看到的外观。对于这一点，你必须在模拟器甚至目标设备上测试你的应用程序。

几乎任何一个 ViewGroup 或者 View（或者 View 的子类）的属性均可以使用 XML 布局资源文件进行配置。将一些静态的创建和用户界面控件的布局以及控件属性的基本定义放到 XML 中，而不是编写杂乱的代码，将大大简化用户界面设计的过程。当然，开发人员也可以根据需要使用程序来控制布局，但他们也能够在 XML 模板中设置所有的默认值。

你一定能够认出下面的代码是包含一个 LinearLayout 和 TextView 控件的简单示例。这是 Eclipse 中任何一个新的 Android 工程所提供的默认布局文件，即/res/layout/main.xml：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
```

Download at http://www.pin5i.com/

```
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello" />
</LinearLayout>
```

这段 XML 显示了包含一个单独的 TextView 的基本布局。第 1 行代码，作为大多数 XML 文件都包含的内容，是必不可少的。由于在所有的文件中都可以看到它，所以在其他的例子中我们都将其省略了。

接下来，我们定义了 LinearLayout 元素。LinearLayout 是一种将子 View 显示在单列或单行中的 ViewGroup。当应用于整个屏幕时，如果方向垂直，则它仅仅表示把每一个 View 绘制在前一个 View 的下方；而如果方向水平，那么它表示每一个 View 均绘制在前一个 View 的右侧。

最后，代码定义了一个单独的子 View，这里是一个 TextView。TextView 是一种控件，同时它也是一种 View。TextView 用于在屏幕上绘制文本。在这个例子中，它将绘制定义在“@string/hello”字符串资源中的文字。

虽然仅仅创建一个 XML 文件并不会实际地在屏幕上绘制任何东西，但某个特定的布局通常会与一个特定的 Activity 相关联。在你的默认 Android 工程中，仅有一个 Activity，它默认使用 main.xml 布局。使用 main.xml 布局的标识符调用 setContentView() 可以将 Activity 与 main.xml 相关联。不含扩展名的布局 ID 同对应的 XML 文件相匹配。在这种情况下，前面的例子来源于 main.xml，所以这一布局的标识符为简单的 main。

```
setContentView(R.layout.main);
```

Tip

虽然有点令人疑惑，但术语“布局”在 Android 开发中有两个不同（但相关）的意义。

在与资源相关的意思中，/res/layout 目录所包含的 XML 资源定义通常称为布局资源文件。这些 XML 文件提供用于绘制屏幕的模板；布局资源文件可以包含任意数量的视图。我们在第 6 章“管理应用程序资源”中讨论过布局资源的相关话题。

术语“布局”还用于指代一系列的 ViewGroup 类，如 LinearLayout、FrameLayout、TableLayout 和 RelativeLayout。这些布局类用来组织 View 控件。我们将在本章的稍后部分中讲解这些类。

所以可以这么说：你可以在布局资源文件（如/res/layout/myScreen.xml）中定义一个或多个布局（如带有两个子控件（TextView 和 ImageView）的 LinearLayout）。

8.1.2 使用代码创建布局

你还可以在运行时通过程序创建用户界面组件，比如布局，但出于组织原则和维护性上考虑，这最好用于一些特殊的情况，而非普遍使用。最大的原因在于使用程序创建布局非常困难，并且难于维护，相反使用 XML 资源的方法就显得直观而更加组织化，而且不具备 Java 技能的设计人员也可以完成。

Download at http://www.pin5i.com/

Tip

这一部分中的许多代码示例都来自于 SameLayout 应用程序。在本书的网站上可以下载 SameLayout 应用程序的源代码。

下面的例子演示了如何使用程序让 Activity 实例化一个 LinearLayout 视图，并且在其中放置两个 TextView 对象。示例并没有使用任何资源，所有的行为都在运行时进行。

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    TextView text1 = new TextView(this);
    text1.setText("Hi there!");

    TextView text2 = new TextView(this);
    text2.setText("I'm seconed. I need to wrap.");
    text2.setTextSize((float) 60);

    LinearLayout ll = new LinearLayout(this);
    ll.setOrientation(LinearLayout.VERTICAL);
    ll.addView(text1);
    ll.addView(text2);

    setContentView(ll);
}
```

当 Activity 被创建时，其 onCreate()方法将被调用。该方法所做的第一件事只是一些常规的“家务事”，它调用了基类的构造函数。

接下来，代码实例化了两个 TextView 控件。每个 TextView 的 Text 属性均使用 setText()方法进行了设定。所有的 TextView 属性，例如 TextSize，均通过 TextView 对象相应的方法调用进行了设置。这些行为所实现的功能与之前在 Eclipse 布局资源设计器中设置 Text 和 TextSize 属性是一样的，只是它们是在运行时进行的，而非定义在布局文件并编译至应用程序包以外。

Tip

XML 的属性名称通常和程序中所使用的获取或设置控件属性值时的方法名相似。例如，android:visibility 对应 setVisibility()方法和 getVisibility()方法。在前面定义 TextView 的示例中，用于获取和设置 TextSize 属性的方法分别为 getTextSize()和 setTextSize()。

要适当地显示 TextView 对象，我们需要将其包裹在某种类型（一个布局）的容器中。在这个例子中，我们使用了一个 LinearLayout，并且将屏幕方向设定为了 VERTICAL，这样，第二个 TextView 将绘制在第一个的底部，每一个均与屏幕的左方对齐。两个 TextView 控件按照我们所希望的顺序被添加进了 LinearLayout 中。

最后，我们调用了 Activity 类的 setContentView()方法在屏幕上绘制 LinearLayout 及其包含的内容。

正如你所看到的，上面的代码可以随着 View 控件以及每个 View 所需属性的增加而迅速扩展。下面是相同的布局，只是现在被表示成 XML 布局文件的形式：

Download at http://www.pin5i.com/

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:id="@+id/TextView1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Hi There!"
    />
<TextView
    android:id="@+id/TextView2"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textSize="60px"
    android:text="I'm seconed. I need to wrap."
    />
</LinearLayout>
```

图 8.1 两种不同方法创建的屏幕拥有相同的效果

你应该会发现上面的代码并不是前面部分代码的字面翻译，尽管它们的输出是一致的，如图 8.1 所示。

首先，在 XML 布局文件中，layout_width 和 layout_height 是必需的属性。其次，每一个 TextView 对象均关联了自己的唯一 id 属性，这样它可以在程序中进行访问。最后，还需要指定 textSize 属性所使用的单位。XML 属性使用 dimension 类型（如第 6 章中所描述的一样），而非一个浮点数。

最后的结果与使用程序编写的方法仅有微小的差异。但是，却远比其容易阅读与维护。要显示这一布局视图只需要简单的一行代码即可。如果布局资源存储在/res/layout/resource_based_layout.xml 文件中，只需使用以下代码：

```
setContentView(R.layout.resource_based_layout);
```

8.2 组织用户界面

我们在第 7 章“探索用户界面屏幕元素”中讨论了在 Android 中，View 类是如何为用户界面构建一个区域的。我们知道，所有的用户界面控件，例如 Button、Spinner 和 EditText，均继承自 View 类。

现在，我们来探讨一种特殊的 View，它被称为 ViewGroup。继承于 ViewGroup 的类允许开发人员用来在屏幕上显示 View 对象（包括所有的你在第 7 章中学习到的用户界面控件），以实现某种组织化的风格。

如同其他的 View 对象（包括第 7 章中谈及的控件），ViewGroup 对象控制着屏幕空间上的一个矩形区域。ViewGroup 与普通控件的区别在于 ViewGroup 对象能够容纳其他

Download at http://www.pin5i.com/

View 对象。一个包含其他 View 对象的 View 被称作父视图（view），而被父 View 包含的 View 对象则被称作子视图，或“孩子”。

子 View 对象在程序中通过 addView() 方法被添加到 ViewGroup 中。在 XML 中，View 控件子对象通过定义子节点的方式被添加到 ViewGroup 中（在父 XML 元素中，如同我们前面多次看到的使用 LinearLayout ViewGroup 那样）。

ViewGroup 的子类被划分为两种类型：

- 布局类；
- 视图容器控件。

Android SDK 还提供层级阅览（Hierarchy Viewer）工具来帮助开发人员直观地看见所设计的布局，这将在本章稍后部分讲解。

1. 为布局使用 ViewGroup 子类

许多用于屏幕设计的重要 ViewGroup 子类的名称往往以“Layout”结尾，例如，LinearLayout、RelativeLayout、TableLayout 和 FrameLayout。其中的每一个类均按照一定的方式在屏幕上定位 View 对象（控件）。例如，我们已经使用过 LinearLayout 来将各种 TextView 和 EditText 以单列的形式排列在屏幕上。我们也可以使用 AbsoluteLayout 来指定每一个控件在屏幕上的绝对 *x/y* 坐标位置，但这种方法并不能在不同的屏幕解析度下简单地进行移植。用户一般情况下并不和 Layout 对象直接打交道，而是与其中包含的 View 对象相交互。

2. 将 ViewGroup 子类作为视图容器

第二种 ViewGroup 的子类为其间接子类。这些特殊的 View 对象既能够如同 Layout 对象一样充当 View 的容器，同时也能够像普通控件那样提供一些功能，以允许用户与其交互。不幸的是，这些类都不具有一个容易记忆的名字，它们以其所提供的功能类型而命名。

这一类型的类包括 Gallery、GridView、ImageSwitcher、ScrollView、TabHost 和 ListView。把这些对象看做不同类型的 View 浏览器可以帮助你理解它们。ListView 将每一个 View 作为一个列表项来显示，用户可以通过垂直滚动的方式浏览这些独立的 View 对象。Gallery 是一个水平滚动 View 对象列表；用户可以通过左右滚动来浏览其中的 View 对象。TabHost 是一个更为复杂的 View 容器，其中每一个 Tab 都可以包含一个 View，用户可以通过名称选择某个 Tab 来查看其中的 View 内容。

3. 使用层级阅览工具（Hierarchy Viewer Tool）

除了 Android 插件提供的 Eclipse 布局资源设计器之外，Android 软件开发包（Software Development Kit，SDK）还提供了一个称为层级阅览器（Hierarchy Viewer）的用户界面工具。在 Android SDK 的子目录/tools 下可以找到这个层级阅览器。

层级阅览器是一种可以允许你检视 Android 应用程序的 View 对象及其父子关系的可视化工具。你可以在运行时深入到某个特定的 View 对象，并且查看其各个 View 属性。你甚至可以保存运行在模拟器或设备上的应用程序状态截图，尽管这一特性出于某种原因不是特别稳定。

要运行同模拟器中的应用程序相关联的层级阅览器，可以采取以下步骤。

① 在模拟器中运行你的 Android 应用程序。

② 导航至 Android SDK 的/tools 目录，运行层级阅览器。

Download at http://www.pin5i.com/

③ 从设备列表中选择你的模拟器实例。

④ 从可用的窗口中选择你想要查看的应用程序。例如，要载入本书中提供的应用程序，选择类似第 6 章中的 ParisView 这样的工程。

⑤ 单击菜单栏上的 Load View Hierarchy 按钮。

默认地，层级阅览器载入应用程序的布局视图。这包含了以树状方式显示的父子视图关系。除此，当选中“树”中的某一个视图节点时，属性面板中将显示其各种属性。屏幕上将会出现 View 对象的线框模型，并且一个红色的方块将高亮当前所选中的视图，它与屏幕上相同位置的视图相对应。

Tip

如果你为应用程序的视图对象提供了一个便于记忆而有意义的 id 属性，而非使用默认自动生成的 id 标记，那么在使用层级阅览工具时，将会更有效地在不同 View 对象间进行切换。例如，一个名为 SubmitButton 的 Button 控件看起来好过 Button1。

图 8.2 显示了载入了第 6 章中的 ParisView 工程的层级阅览器，这是一个单一屏幕的应用程序，它拥有一个单独的 LinearLayout，其中包含一个 TextView 和其 ImageView 子控件，它们都被包含在了一个 ScrollView 控件中（以提供滚动的功能）。右侧的子树显示了应用程序的主体部分，它以标识符为 ParisViewLayout 的 LinearLayout 作为起点。另外的子树是应用程序的标题栏。简单地在每个子节点上进行双击可以在它自己的窗口中打开这一单独的 View 对象。

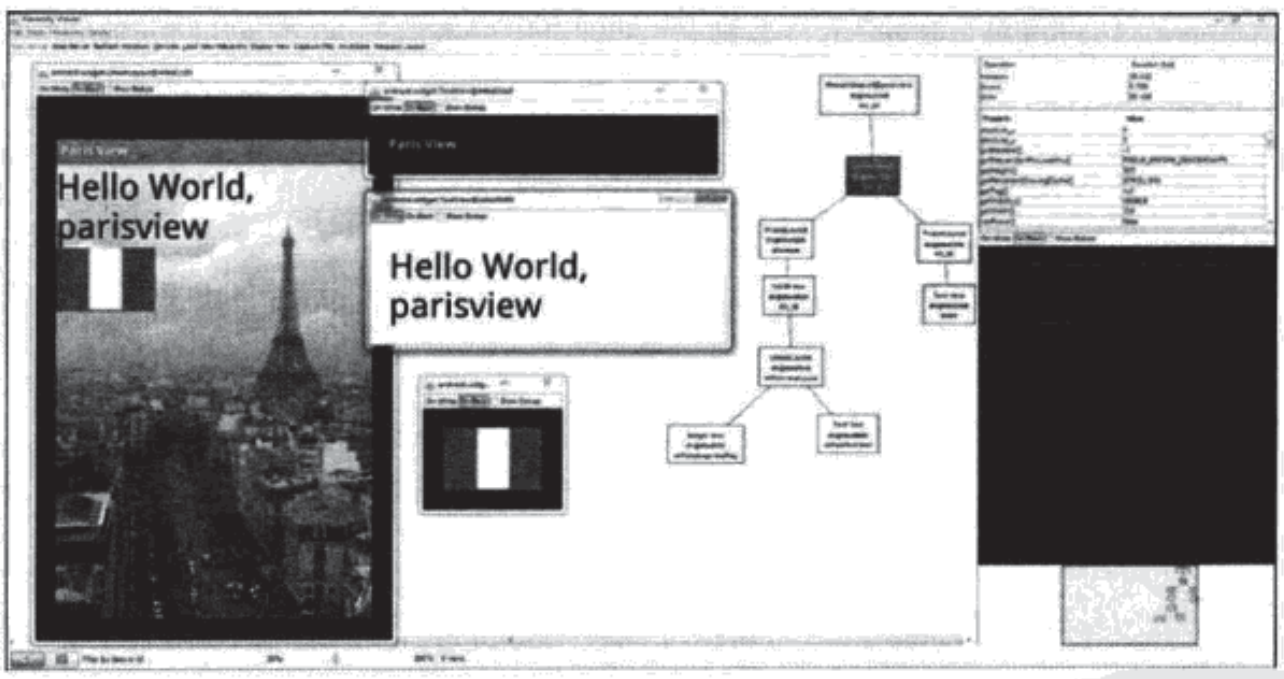


图 8.2 显示在层级阅览工具（布局视图）中的 ParisView 应用程序

在“树”中选中适当的 View，并且单击菜单栏中的 Display View 按钮可以在单独的窗口中打开这一 View。在图 8.2 中，你可以看见 Display View 按钮对于每一个子节点均可用：包含国旗的 ImageView、包含文本的 TextView、LinearLayout 父节点（它包含它的子对象）以及最终的应用程序标题栏。

你可以使用“精细像素（Pixel Perfect）”视图通过一个放大镜来仔细查看应用程序（如图 8.3 所示）。你也可以载入一个 PNG 模型文件（mockup file）覆盖你的用户界面并且调整应用程序的外观。你可以通过单击层级阅览器左下方的“九像素”按钮来访问“精细像素”视图，而单击“三像素”按钮则会返回到布局视图。

层级阅览工具对于同 View 控件相关的绘制任务的调制将会非常有价值。如果你怀疑一

Download at http://www.pin5i.com/

个 View 是否可用或者疑惑为什么它没有被绘制出来，那么你可以尝试着运行层级阅览器并且检查出问题的 View 对象的属性。

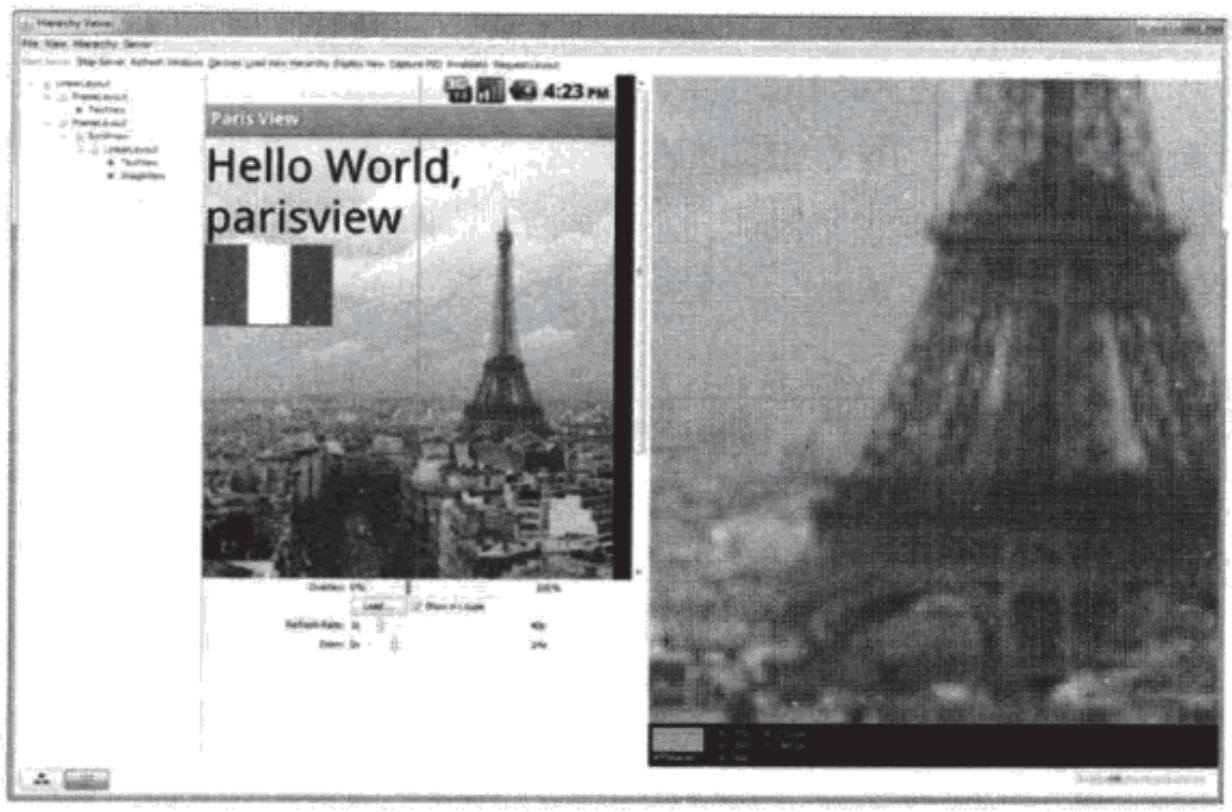


图 8.3 显示在层级阅览工具（“精细像素”视图）中的 ParisView 应用程序

你可以使用层级阅览工具来调试你应用程序的用户界面，或者同其进行交互。具体地讲，开发人员可以使用菜单栏上的 Invalidate 和 Request Layout 按钮，它们分别对应 UI 线程的 View.Invalidate() 和 View.requestLayout() 函数。这两个函数初始化 View 对象并且根据事件绘制或重绘它们。

最后，你还可以使用层级阅览器来解构其他应用程序（特别是示例程序）是如何处理它们的布局和显示的。当你想要创建一个同另外的应用程序相似的布局时，它将会非常有用，特别是当所参照的应用程序使用的是储备（stock）View 类型时。不过，你也可能发现一些没有在 SDK 中提供的 View 类型，并且你需要自己实现这些自定义的类。

8.3 使用内建的布局类

我们谈论了相当多有关 LinearLayout 布局的话题，但是还有其他几种类型的布局我们没有涉及。每一种布局都拥有自己不同的方法和顺序来在屏幕上显示其子 View 控件。布局均继承自 android.view.ViewGroup。

Android SDK 框架中内建的布局类型包括：

- FrameLayout；
- LinearLayout；
- RelativeLayout；
- TableLayout。

Tip

这一部分中的许多代码示例都来自于 SimpleLayout 应用程序。在本书的网站上可以下载 SimpleLayout 应用程序的源代码。

所有的布局，无论类型，都拥有布局属性。这些布局属性将应用于包含于布局的任何子

Download at http://www.pin5i.com/

View。布局属性可以在运行时通过程序设定，但更为理想的是使用下面的语法在 XML 布局文件中进行设置：

```
android:layout_attribute_name="value"
```

有几种布局属性是为所有的 ViewGroup 对象所共享的，包括尺寸属性和边距属性。你可以在 ViewGroup.LayoutParams 类中找到这些基本的布局属性。边距属性使得布局中包含的每一个子 View 在各个方向的边沿处均进行缩进。可以在 ViewGroup.MarginLayoutParams 类中找到这些属性。还有许多的 ViewGroup 属性用于处理子 View 的绘制边界和动画设置。

表 8.1 给出了所有 ViewGroup 子类型共享的一些重要属性。

表 8.1　　重要的 ViewGroup 属性

属性名	应用于	描述	值
android:layout_height	父视图、子视图	布局中子视图控件所需的高度属性	特定的尺寸值或 fill_parent、wrap_content，match_parent 选项需要 API Level 8 以上
android:layout_width	父视图、子视图	布局中子视图控件所需的宽度属性	尺寸值或 fill_parent、wrap_content，match_parent 选项需要 API Level 8 以上
android:layout_margin	子视图	视图四边的额外空间	特定尺寸值

下面是一个设置为屏幕尺寸的 LinearLayout 的 XML 布局资源示例，它包含了一个设置为 LinearLayout 完整高度和宽度（也就是屏幕尺寸）的 TextView。

```
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:id="@+id/TextView01"
        android:layout_height="fill_parent"
        android:layout_width="fill_parent" />
</LinearLayout>
```

下面是一个 Button 对象的示例，该示例代码出现在布局资源文件中，它通过 XML 为这一 Button 对象设置了一定的边距：

```
<Button
    android:id="@+id/Button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Press Me"
    android:layout_marginRight="20px"
    android:layout_marginTop="60px" />
```

记住布局元素可以覆盖屏幕上的任何矩形区域，而非一定是整个屏幕。由于布局是 View 对象，所以它们可以相互容纳。这给开发人员组织屏幕元素带来了极大的便利与灵活性。一种比较常见的布局设计方式是把 FrameLayout 或 LinearLayout（在第 7 章的许多例子中都可以看到）作为父布局填充整个屏幕，然后再在其中使用最合适的布局组织单独的屏幕元素。

现在，我们来分别讨论每一种常见的布局类型，以及它们彼此之间的区别。

Download at http://www.pin5i.com/

8.3.1 使用 FrameLayout

FrameLayout 视图被设计用来显示一堆子 View 项。我们可以向布局内加入若干个视图，但每一个 View 均以布局的左上角作为参照点进行绘制。它可以用来在同一片区域显示多个图像，如图 8.4 所示，此时布局的尺寸将匹配其中最大的子 View。

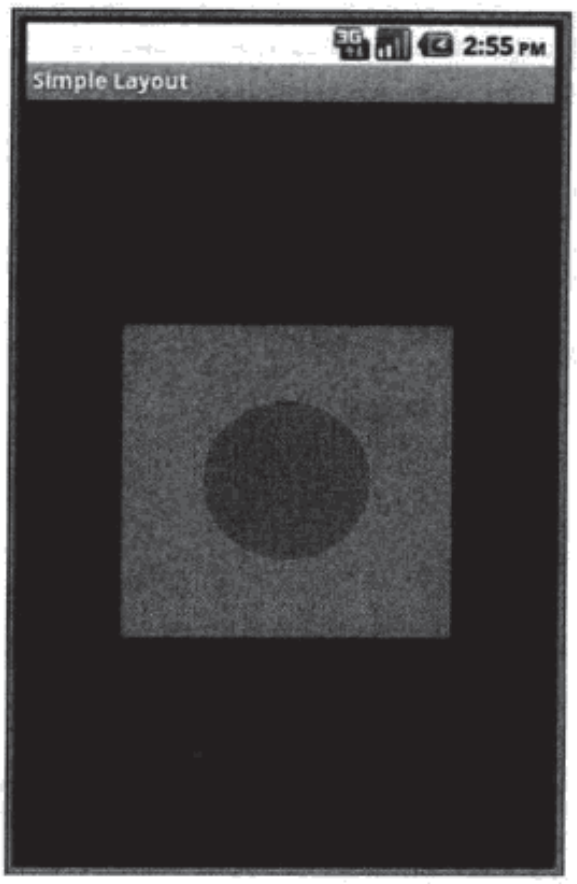


图 8.4 FrameLayout 使用示例

我们可以在 android.control.FrameLayout.LayoutParams 中找到针对 FrameLayout 子 View 对象可用布局属性。表 8.2 列出了其中比较重要的 FrameLayout 属性。

表 8.2 重要的 FrameLayout 视图属性

属 性 名	应用于	描 述	值
android:foreground	父视图	覆盖原内容绘制 Drawable	Drawable（可绘制）资源
android:foregroundGravity	父视图	前景 Drawable 的重力属性	一个或多个常量，多个常量由“|”分隔开。常量有：top、bottom、left、right、center_vertical、fill_vertical、center_horizontal、fill_horizontal、center 和 fill
android:measureAllChildren	父视图	将布局尺寸限制为所有的子对象或设置为 VISIBLE 的子对象（而非设置为 INVISIBLE 的子对象）	true 或 false
android:layout_gravity	子视图	重力常量，描述子 View 如何放置在父布局中	一个或多个常量，多个常量由“|”分隔开。常量有：top、bottom、left、right、center_vertical、fill_vertical、center_horizontal、fill_horizontal、center 和 fill

下面是一个包含两个子 View 对象的 FrameLayout 的 XML 布局资源示例，这两个子 View 均为 ImageView 对象。示例首先绘制一个绿色的矩形，然后在其表面绘制一个红色

Download at http://www.pin5i.com/

的椭圆。绿色的矩形稍大一些，所以 FrameLayout 的边界以它为准：

```
<FrameLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:id="@+id/FrameLayout01"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:layout_gravity="center">
    <ImageView
        android:id="@+id/ImageView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/green_rect"
        android:minHeight="200px"
        android:minWidth="200px" />
    <ImageView
        android:id="@+id/ImageView02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/red_oval"
        android:minHeight="100px"
        android:minWidth="100px"
        android:layout_gravity="center" />
</FrameLayout>
```

8.3.2 使用 LinearLayout

LinearLayout 视图将它的子 View 对象排列在一行（如图 8.5 所示）或一列中，这取决于它的方向属性设置为水平还是垂直。对于创建表单而言，这是一种非常方便实用的布局方法。

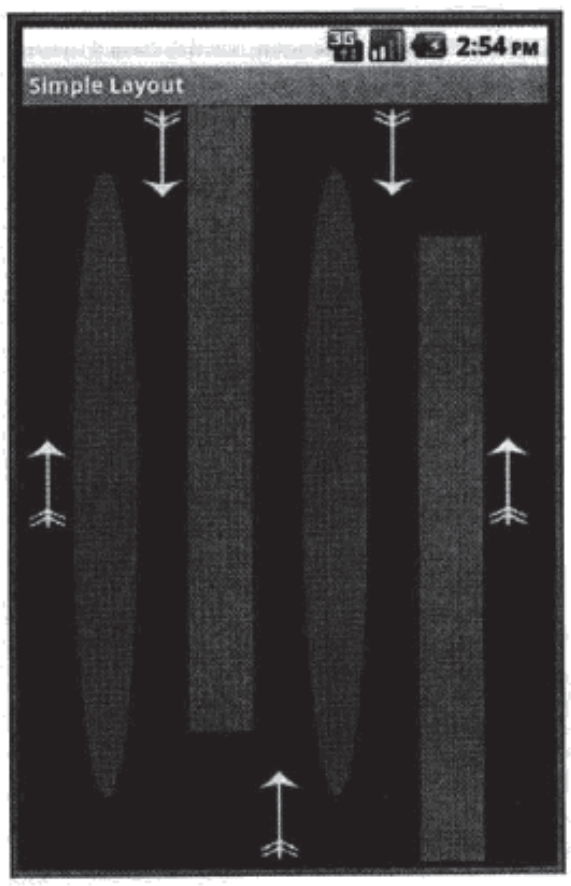


图 8.5 LinearLayout 示例（水平方向）

你可以在 android.control.LinearLayout.LayoutParams 中找到针对 LinearLayout 子 View 对象的可用布局属性。表 8.3 列出了其中一些重要的 LinearLayout 视图属性。

Download at http://www.pin5i.com/

表 8.3 重要的 LinearLayout 视图属性

属 性 名	应用于	描 述	值
android:orientation	父视图	布局为行（水平）模式或列（垂直）模式	horizontal（水平）或 vertical（垂直）
android:gravity	父视图	布局中子视图的重力属性	一个或多个常量，多个常量由“|”分隔开。常量有：top、bottom、left、right、center_vertical、fill_vertical、center_horizontal、fill_horizontal、center 和 fill
android:layout_gravity	子视图	特定子视图的重力属性。用来定位视图	一个或多个常量，多个常量由“|”分隔开。常量有：top、bottom、left、right、center_vertical、fill_vertical、center_horizontal、fill_horizontal、center 和 fill
android:layout_weight	子视图	特定子视图的权重。用来确定使用父控件的屏幕控件比例	一个父视图内所有子视图的值之和必须为 1。例如，一个子控件的值为 0.3，那么另一个则为 0.7

8.3.3 使用 RelativeLayout

RelativeLayout 视图允许你通过子 View 控件的相对关系来指定其位置。例如，你可以通过唯一标识符设定某个子 View 位于另一个 View 的“上方”、“下方”、“左侧”或“右侧”。你还可以将子对象设置为相互对齐，或者倚靠父布局的边界。将 RelativeLayout 属性很好地结合起来可以简化创建有趣的用户界面的过程，达到预期的效果，而无需求助于多重布局的复杂组合。图 8.6 显示了每一个按钮控件之间的相互位置关系。

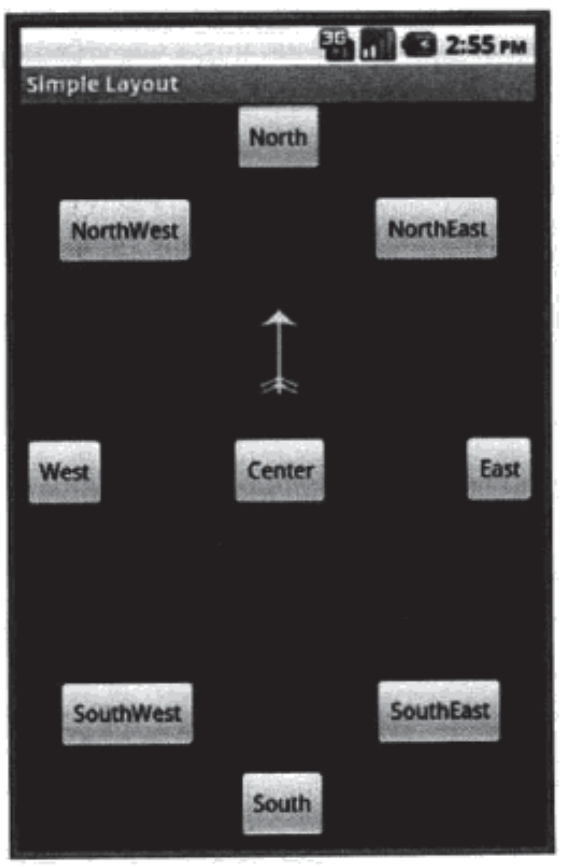


图 8.6 RelativeLayout 使用示例

你可以在 android.control.RelativeLayout.LayoutParams 中找到针对 RelativeLayout 子 View 对象可用的布局属性。表 8.4 列出了其中比较重要的 RelativeLayout 视图属性。

Download at http://www.pin5i.com/

表 8.4 重要的 RelativeLayout 视图属性

属 性 名	应用于	描 述	值
android:gravity	父视图	布局中子视图的重力属性	一个或多个常量，多个常量由“|”分隔开。常量有：top、bottom、left、right、center_vertical、fill_vertical、center_horizontal、fill_horizontal、center 和 fill
android:layout_centerInParent	子视图	在父视图中水平、垂直均居中	true 或 false
android:layout_centerHorizontal	子视图	在父视图中水平居中	true 或 false
android:layout_centerVertical	子视图	在父视图中垂直居中	true 或 false
android:layout_alignParentTop	子视图	与父视图上边沿对齐	true 或 false
android:layout_alignParentBottom	子视图	与父视图下边沿对齐	true 或 false
android:layout_alignParentLeft	子视图	与父视图左边沿对齐	true 或 false
android:layout_alignParentRight	子视图	与父视图右边沿对齐	true 或 false
android:layout_alignRight	子视图	与指定视图（ID）的右边沿对齐	视图 ID；例如：@id/Button1
android:layout_alignLeft	子视图	与指定视图（ID）的左边沿对齐	视图 ID；例如：@id/Button1
android:layout_alignTop	子视图	与指定视图（ID）的上边沿对齐	视图 ID；例如：@id/Button1
android:layout_alignBottom	子视图	与指定视图（ID）的下边沿对齐	视图 ID；例如：@id/Button1
android:layout_above	子视图	将该视图的下边沿放置于指定视图（ID）之上	视图 ID；例如：@id/Button1
android:layout_below	子视图	将该视图的上边沿放置于指定视图（ID）之下	视图 ID；例如：@id/Button1
android:layout_toLeftOf	子视图	将该视图的右边沿放置在指定视图（ID）左侧	视图 ID；例如：@id/Button1
android:layout_toRightOf	子视图	将该视图的左边沿放置在指定视图（ID）右侧	视图 ID；例如：@id/Button1

下面是一个拥有两个子 View 对象的 RelativeLayout 的 XML 布局资源示例，两个子对象分别为一个 Button 对象和一个 ImageView 对象，其中，Button 对象相对于其父视图对齐，而 ImageView 则相对 Button（和父视图）定位：

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=
```

Download at http://www.pin5i.com/

```
"http://schemas.android.com/apk/res/android"
android:id="@+id/RelativeLayout01"
android:layout_height="fill_parent"
android:layout_width="fill_parent">
<Button
    android:id="@+id/ButtonCenter"
    android:text="Center"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true" />

<ImageView
    android:id="@+id/ImageView01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@id/ButtonCenter"
    android:layout_centerHorizontal="true"
    android:src="@drawable/arrow" />
</RelativeLayout>
```

Warning

AbsoluteLayout 类已经被废止了。AbsoluteLayout 使用特定的 x 和 y 坐标来定位子视图。当需要采用“精细像素”布局时，这种方式显得比较有用，但是，这种精确到像素的布局方式并不能很好地与具有不同屏幕尺寸的其他设备相兼容。在大多数情况下，其他常用的布局类型，如 FrameLayout 和 RelativeLayout，足以替代 AbsoluteLayout，所以我们鼓励你尽可能地使用它们。

8.3.4 使用 TableLayout

TableLayout 将其子视图排列成若干行，如图 8.7 所示。每一行内单独的 View 对象需要使用 TableRow 布局 View（基本上等同于一个水平布局的 LinearLayout）来添加到表格中。TableRow 的每一列能够容纳一个 View（或者是包含其他子 View 对象的布局）。添加到 TableRow 中的 View 项按照其加入的先后顺序逐列放置。如果需要的话，你也可以通过指定列序号（以 0 为基点）来跳过一些无需放置对象的列（如图 8.7 中最底部一行所示）；否则，View 将会被放置在右侧的下一个列中。列的尺寸会随着该列中最大的 View 而进行放缩。另外，如果你想要某个 View 占据整行的话，也可以添加普通的 View 对象替代 TableRow 元素。

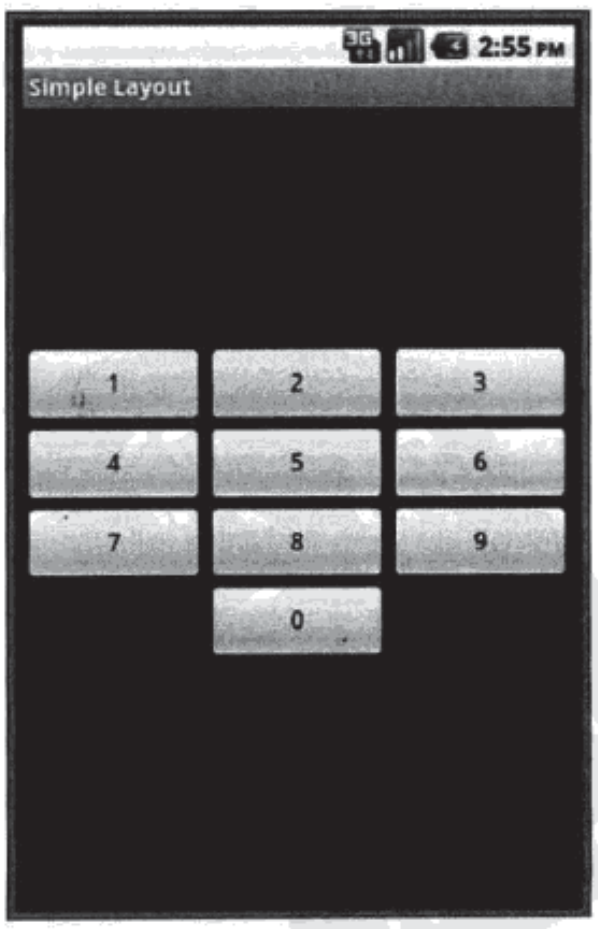


图 8.7 TableLayout 使用示例

你可以在 android.control.TableLayout.LayoutParams 中找到针对 TableLayout 子 View 对象的可用布局属性。在 android.control.TableRow.LayoutParams 中，你还可以找到针对 TableRow 子 View 对象的可用布局属性。表 8.5 列出了比较重要的 TableLayout View 对象属性。

Download at http://www.pin5i.com/

表 8.5 重要的 TableLayout 和 TableRow 视图属性

属 性 名	应 用 于	描 述	值
android:collapseColumns	TableLayout	使用逗号分隔的废弃（collapse）列序号表（以 0 为基点）	字符串或字符串资源；例如："0,1,3,5"
android:shrinkColums	TableLayout	使用逗号分隔的收缩（shrink）列序号表（以 0 为基点）	字符串或字符串资源；"*"代表所有列。例如："0,1,3,5"
android:stretchColums	TableLayout	使用逗号分隔的伸展（stretch）列序号表（以 0 为基点）	字符串或字符串资源；"*"代表所有列。例如："0,1,3,5"
android:layout_colum	TableRow 子视图	该视图所在的列序号（以 0 为基点）	整数或整型资源；例如：1
android:layout_span	TableRow 子视图	该视图所跨列的列数量	不小于 1 的整数或整型资源；例如：3

下面是一个含有两行（两个 TableRow 子对象）的 TableLayout 的 XML 布局资源示例。TableLayout 被设置为了将列的尺寸伸展至屏幕宽度。第一个 TableRow 拥有 3 列；每一格中均有一个 Button 对象。第二个 TableRow 仅显式地在其第二格中放置了一个 Button 视图。

```
<TableLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:id="@+id/TableLayout01"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="*">
    <TableRow
        android:id="@+id/TableRow01">
        <Button
            android:id="@+id/ButtonLeft"
            android:text="Left Door" />
        <Button
            android:id="@+id/ButtonMiddle"
            android:text="Middle Door" />
        <Button
            android:id="@+id/ButtonRight"
            android:text="Right Door" />
    </TableRow>
    <TableRow
        android:id="@+id/TableRow02">
        <Button
            android:id="@+id/ButtonBack"
            android:text="Go Back"
            android:layout_column="1" />
    </TableRow>
</TableLayout>
```

8.3.5 在屏幕上使用多布局

将不同的布局方法联系起来可以在一个单独的屏幕上创建丰富的布局。记住：由于布局能够包含 View 对象，而且它本身也是一种 View，所以布局能够包含其他布局。图 8.8 展示了通过布局视图的结合创建出的复杂而有趣的屏幕。

Download at http://www.pin5i.com/

图 8.8 联合使用多个布局的示例

Warning

记住：移动应用程序的每一个屏幕页面都应该保持整洁与相对简单。这不仅仅是基于积极的用户体验考虑，让复杂（且层级繁多）的视图堆满屏幕也会导致性能的下降。除了使用层级阅览器仔细检查应用程序的布局之外，你还可以使用 layoutopt 命令行工具来辅助优化你的布局、发现不需要的组件。这一工具使用布局优化技术来辅助发现一些诱因，如<merge>和<include>标记。

8.4 使用内建的 View 容器类

布局并非唯一能够包含其他 View 对象的控件。虽然使用布局可以很方便地将其他 View 对象放置在屏幕上，但它们毕竟不能互动。现在，我们使用另外一种 ViewGroup：容器。这种 View 对象通常可以容纳其他简单的 View 类型，并且能够允许以某种标准的方式同其交互，以在其子 View 对象之间进行导航。与布局类似，这些控件也都有着各自设计周到的作用。

Android SDK 框架中内建的 ViewGroup 容器类型包括：

① 列表、网格和“画廊（gallery）”；

② 使用 ViewFlipper、ImageSwitcher、TextSwitcher 所表示的“开关选项（switcher）”；

③ 使用 TabHost 和 TabControl 表示的标签（Tab）；

④ 使用 ScrollView 和 HorizontalScrollView 进行滚动；

⑤ 使用 SlidingDrawer 隐藏或显示内容。

Tip

本章中的许多代码示例都来自于 AdvancedLayout 应用程序。在本书的网站上可以下载 AdvancedLayout 应用程序的源代码。

Download at http://www.pin5i.com/

8.4.1 使用数据驱动容器

某些 View 容器控件被设计用来以某种方式显示重复的 View 对象。这类 View 容器控件的代表有 ListView、GridView 和 GalleryView。

- ListView：包含一个垂直滚动、水平填充的 View 对象列表，一般来讲，其中的每一项都包含一行数据，用户可以选择其中的一项执行某些操作。
- GridView：包含具有一定列数的网格所容纳的 View 对象，该类容器常常用来显示图像或图标，用户可以选择其中的一项执行某些操作。
- GalleryView：包含一个水平滚动的 View 对象列表，通常也用于图像、图标的显示，用户可以选择其中的一项执行某些操作。

这些容器都属于 AdapterView 控件类型。AdapterView 控件包含一系列子 View 控件以显示某个数据源的数据。Adapter 根据数据源生成这些子 View 控件。对于所有这些容器控件，这一部分非常重要，所以我们首先谈论 Adapter 对象。

在这一小节中，你将学习到如何使用 Adapter 对象将数据绑定到 View 对象上。在 Android SDK 中，Adapter 从数据源读取数据，并且按照某种规则将数据提供给 View 对象，具体的规则依赖于所使用的 Adapter 类型。这一 View 对象随后作为某个 AdapterView 的子 View 对象填充其中。

最常见的 Adapter（适配器）类是 CursorAdapter 和 ArrayAdapter。CursorAdapter 从 Cursor 获得数据，而 ArrayAdapter 则从一个数组中获取数据。当需要使用数据库中的数据时，CursorAdapter 是一个很好的选择。ArrayAdapter 则适用于读取单列数据，或者从数组资源中获得数据。

对于 Adapter 对象，我们还有几点需要掌握。当创建一个 Adapter 时，你需要提供一个布局标识符。它将作为模板用来填充每一行的数据。你所创建的模板包含 Adapter 所需关联数据的控件的标识符。一个足够简单的布局可能只包含单一的 TextView 控件。当创建 Adapter 时，既需要引用布局资源，也还要引用 TextView 控件的标识符。Android SDK 提供了可供应用程序使用的常用布局资源。

1. 使用 ArrayAdapter

ArrayAdapter 将数组的每一个元素同布局资源中单独的一个 View 相绑定。下面是一个创建 ArrayAdapter 的示例：

```
private String[] items = {
    "Item 1", "Item 2", "Item 3" };
ArrayAdapter adapt =
    new ArrayAdapter<String>
        (this, R.layout.textview, items);
```

在这个例子中，出现了一个名为 items 的 String 数组。它被 ArrayAdapter 用来作为源数据。我们还使用了一个布局资源，它将作为填充每项数据的 View 而多次重复，定义如下：

```
<TextView xmlns:android=
   "http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
```

Download at http://www.pin5i.com/

```
    android:textSize="20px" />
```

上面的布局资源文件虽然仅包含一个单独的 TextView，但是，我们可以依然将其资源标识符传入构造函数，而在更为复杂的布局中构造它。AdapterView 中的每一个子 View 均通过上面的 Adapter 获取一个填充了 String 数组中某一项字符串的 TextView。

如果你定义了数组资源，那么你也可以直接将 AdapterView 的“数据条目（entries）”属性设为该数组资源的标识符，如此，将会自动地提供 ArrayAdapter。

2. 使用 CursorAdapter

CursorAdapter 将一列或多列数据同布局资源中提供的 View 对象相绑定。这最好通过一个示例来说明。下面的例子示范了查询“联系人（Contacts）”Content Provider（内容提供者）数据，并且创建 CursorAdapter 的方法。CursorAdapter 需要使用一个 Cursor。

Note

要获得更多有关Cursor对象的信息，请参阅第10章“使用Android数据和存储API”。

```
Cursor names = managedQuery(
    Contacts.Phones.CONTENT_URI, null, null, null, null);
startManagingCursor(names);
ListAdapter adapter = new SimpleCursorAdapter(
    this, R.layout.two_text,
    names, new String[] {
        Contacts.Phones.NAME,
        Contacts.Phones.NUMBER
    }, new int[] {
        R.id.scratch_text1,
        R.id.scratch_text2
    });
```

在这个例子中出现了两个新的概念。第一，你需要了解 Cursor 必须包含一个名为_id 的字段。在本例中，我们可以确定“联系人”内容提供器拥有这一字段。它将在随后处理用户选择项时发挥作用。

Note

虽然 Contact 类已经废弃不用了，但它却是访问联系人信息的唯一方法，无论在老版本还是新版本的 Android 下。我们将在第 11 章“使用内容提供器在应用程序间共享数据”中更深入地讨论有关联系人和内容提供器的话题。

我们通过调用 manageQuery() 方法获得了 Cursor。之后，我们实例化了一个 SimpleCursorAdapter，并将其作为 ListAdapter。布局 R.layout.two_text 拥有两个 TextView 对象，它们将作为最后一个参数使用。SimpleCursorAdapter 允许我们将数据库中的列同布局中指定的控件相匹配。对于每一个从查询中返回的数据行，我们都会在 AdapterView 中得到一个布局的实例。

3. 向 AdapterView 绑定数据

现在你已基本掌握了 Adapter 对象，可以将其应用于任意一个 AdapterView 控件了。它们都能够很好地工作。虽然 Gallery 技术上需要一个 SpinnerAdapter，但

Download at http://www.pin5i.com/

SimpleCursorAdapter 也能够返回 SpinnerAdapter。下面是一个通过 ListView 实现的示例，它接续了前面的例子：

```
((ListView)findViewById(R.id.list)).setAdapter(adapter);
```

对于 AdapterView 的 setAdapter()方法的调用，这里是通过 ListView 实现的，应该在调用 setContentView()之后。这就是向你的 AdapterView 绑定数据所需进行的全部操作。图 8.9 分别显示了相同的数据呈现在 ListView、Gallery 和 GridView 中的情况。

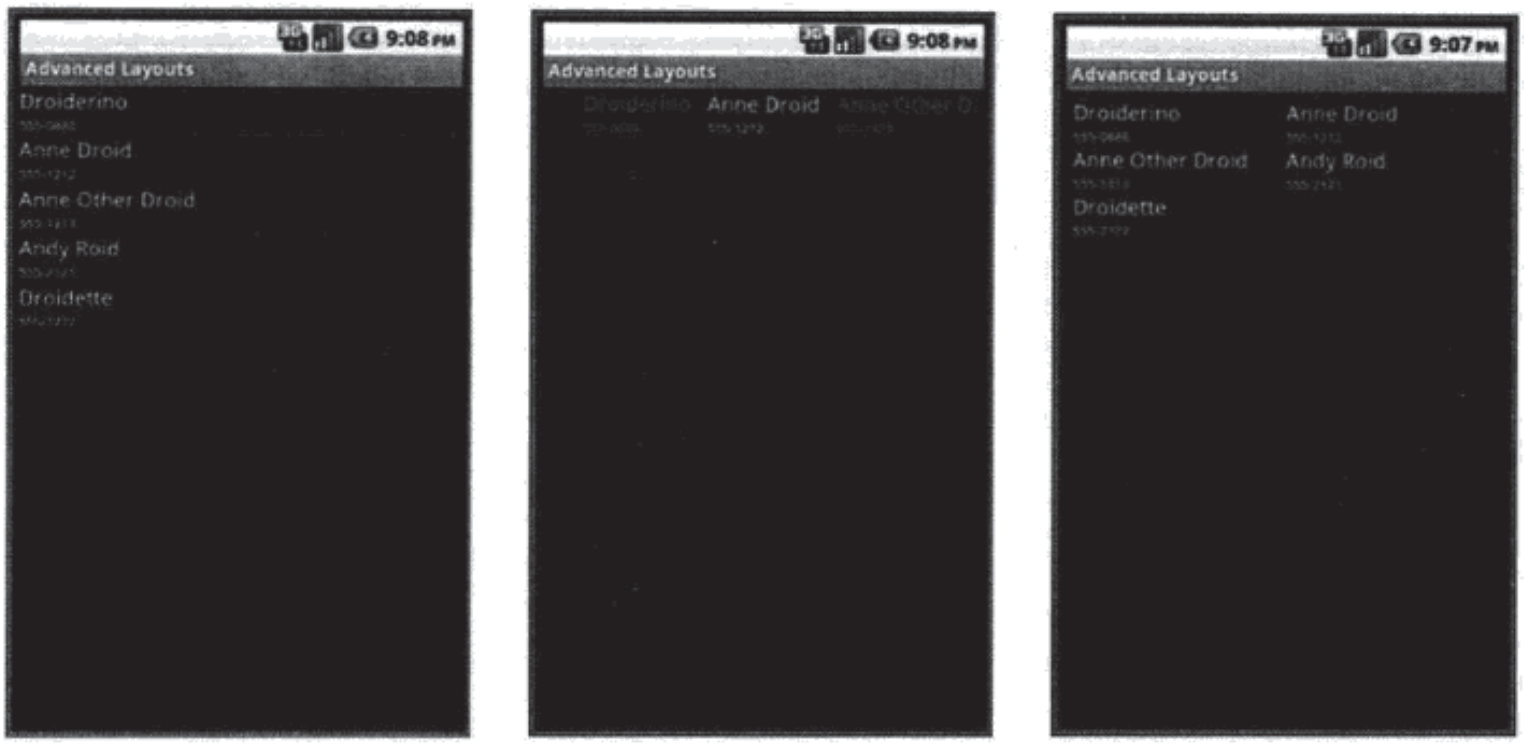


图 8.9 ListView、Gallery 和 GridView：相同的数据、相同的列表项、不同的布局视图

4. 处理选择事件

AdapterView 控件通常用来向用户呈现一些数据，而用户可以在其中进行选择。前面谈及的三种控件（ListView、GridView 和 Gallery）均能够允许应用程序以相同的方式监听单击事件。对此，你需要调用 AdapterView 的 setOnItemClickListener()方法，并传入一个 AdapterView.OnItemClickListener 类的实例作为参数。下面是该类的一种实现：

```
av.setOnItemClickListener(
    new AdapterView.OnItemClickListener() {
    public void onItemClick(
        AdapterView<?> parent, View view,
        int position, long id) {
        Toast.makeText(Scratch.this, "Clicked _id=" +id,
            Toast.LENGTH_SHORT).show();
    }
});
```

在前面的例子中，av 是我们实例化的 AdapterView。所有有趣的事情都发生在 onItemClick()方法内。parent 参数为发生单击事件的 AdapterView，当屏幕上有不止一个 AdapterView 时，这一参数将会起到很大的作用。view 参数则为被单击项所在的 View。position 是以 0 作为起点的，它表示用户所单击的列表项在列表中所处的位置。最后，id 参数表示所选列表项所对应的_id 列。当需要查询所选项所在的行数据的更多信息时，这个参数将发挥其作用。

除此之外，你的应用程序还可以在某个特定项上监听长时单击事件，而且对于已选中的项亦如此。虽然各个参数都是一样的，但应用程序将在高亮的列表项改变时收到事件调用。

Download at http://www.pin5i.com/

这可以对用户使用方向键滚动条目做出反应，而且并不会选中该项。

5. 使用 ListActivity

ListView 控件通常用于显示全屏菜单或列表供用户进行选择。基于此，你也许会考虑使用 ListActivity 作为基类。使用 ListActivity 可以简化这一类型的屏幕。

首先，要处理列表项事件，你现在需要在你的 ListActivity 中提供一些实现。例如，和 onListItemClickListener 相等价的是在你的 ListActivity 中实现 onListItemClick()方法。

第二，要关联一个 Adapter，需要调用 setListAdapter()方法，它可以在调用 setContentView()方法之后完成。不过，这暗示了使用 ListActivity 的一些局限。

要使用 ListActivity，使用 setContentView()方法设置的布局必须包含一个 ListView，该 ListView 的标识符必须设置为 android:list，这是不能更改的。你也可以使用一个标识为 android:empty 的 View，以显示当没有数据从 Adapter 返回时的情况。这种方法仅仅适用于 ListView 控件，所以它拥有一定的局限性。不过，如果它确实适合你的应用程序，那么将为你节省不少的代码量。

Tip

你可以使用 ListView.FixedViewInfo 通过 ListView 的 addHeaderView()和 addFooterView()方法来创建 ListView 的“头（header）”和“脚（footer）”。

8.4.2 使用标签组织视图

Android SDK 提供了一个灵活性很强的方法来实现标签（tab）界面。与 ListView 和 ListActivity 相似，有两种方法可以在 Android 平台上创建标签。你既可以使用 TabActivity，它把屏幕简化成若干个标签，也可以从头开始创建你自己的标签页面。两种方法都依赖于 TabHost 控件。

Warning

Eclipse 布局资源编辑器在设计模式下不会正确地显示 TabHost 控件。为了设计这种类型的布局，你应该使用 XML 布局模式，然后在 Android 模拟器或者 Android 设备上查看标签。

1. 使用 TabActivity

拥有标签的屏幕布局由一个 TabActivity 和一个 TabHost 构成。TabHost 由 TabSpec 构成，TabSpec 是 TabHost 的嵌套类，它包含诸如标签标题和内容这样的标签信息。标签的内容既可以是一个预定义的 View，而后 Activity 通过 Intent 对象来运行它，也可以是一个通过实现 TabContentFactory 来创建的 View。

标签并不是你所想象的那么复杂。每一个标签都是一个高效的 View 容器。这个 View 可以是任何一个已经准备好待显示的 View，例如 XML 布局文件；也可以通过运行一个 Activity 来获得。下面的例子演示了上面的两种方法，它使用了在本章前面的示例中创建的 View 对象和 Activity 对象：

Download at http://www.pin5i.com/

```
public class TabLayout
    extends TabActivity
    implements android.widget.TabHost.TabContentFactory {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        TabHost tabHost = getTabHost();
        LayoutInflacter.from(this).inflate(
            R.layout.example_layout,
            tabHost.getTabContentView(), true);
        tabHost.addTab(tabHost.newTabSpec("tab1")
            .setIndicator("Grid").setContent(
                new Intent(this, GridLayout.class)));
        tabHost.addTab(tabHost.newTabSpec("tab2")
            .setIndicator("List").setContent(
                new Intent(this, List.class)));
        tabHost.addTab(tabHost.newTabSpec("tab3")
            .setIndicator("Basic").setContent(
                R.id.two_texts));
        tabHost.addTab(tabHost.newTabSpec("tab4")
            .setIndicator("Factory").setContent(
                this));
    }
    pubilc View createTabContent(String tag) {
        if (tag.compareTo("tab4") == 0) {
            TextView tv = new TextView(this);
            Date now = new Date();
            tv.setText("I'm from a factory. Created: "
                + now.toString());
            tv.setTextSize((float) 24);
            return (tv);
        } else {
            return null;
}}}
```

上面的示例创建了拥有 4 个标签的标签布局，如图 8.10 所示。第一个标签来源于最近的 GridView 示例；第二个标签取自之前的 ListView 示例；第三个标签为一个拥有两个 TextView 对象的基本布局，这两个 TextView 对象均在前面演示过的 XML 布局文件中进行了定义；而第四个标签则由“工厂（factory）”创建。

图 8.10 显示的 4 个标签

代码首先获得了 TabHost 实例。这一对象可以允许我们为进行屏幕绘制而添加 Intent 对象和 View 标识符。而 TabActivity 则提供了一种方法来获取当前的 TabHost 对象。

之后的操作则与标签视图没有太大的关系。LayoutInflater 用于将一个 View 的 XML 定义转换为实际的 View 对象。这一过程通常会在调用 setContentView() 时发生，但我们并没有这样做。对于通过标识符引用 View 对象而言，使用 LayoutInflater 是必要的，这在创建第三个标签时体现出来了。

最后，代码按照需要显示的顺序依次向 TabHost 添加

Download at http://www.pin5i.com/

了4个标签。这一操作是通过多次调用 TabHost 的 addTab() 方法实现的。前两次调用本质上是相同的。每一次均创建一个 Intent，它需要使用在标签中运行的 Activity 对象的名称。它们都是之前使用过的用于全屏显示的相同 Activity 类。而且即使 Activity 没有设计为全屏显示，它也可以毫无障碍地工作。

接下来，在创建第三个标签时，我们使用了一个 View 的标识符。在之前对 LayoutInflater 的调用中所使用的布局文件里，也包含一个处于 LinearLayout 定义最顶层的标识符，它与这里所使用的相匹配，也与之前在基本 LinearLayout 定义中使用过的相同。这里仍然不需要对这个视图进行任何修改，尽管它需要显示在一个标签中。

下一步添加了一个 TabActivity 类的标签。这是完全有可能的，因为该类本身也继承于 TabHost.TabContentFactory，而它需要实现 createTabContent() 方法。这一视图将在用户第一次选择该标签时被创建，所以这里也不需要其他更多的信息。用于创建这个标签的标记必须被记录下来，因为这是 TabHost 用来识别标签的方式。

最后，代码实现了 createTabContent() 方法，以使用第四个标签。该方法首先检验了传入的标记是否为记录下来的第四个标签的标记。如果确信无疑，将创建一个关联了某个文本字符串的 TextView 对象实例，这一字符串包含了当前的时间。文本的尺寸被设置为了24像素。字符串中所记录的时间戳正好可以显示视图创建的时间，而且也能够证明简单地切换标签并不会重新创建这一视图。

Android 针对标签功能所提供的灵活性使得它可以很好地用在那些定义了一堆视图的应用程序中。对于现存的 View 和 Activity 对象，几乎可以不做修改地直接在 TabHost 中进行工作。

2. 不使用 TabActivity 实现分页标签

不使用 TabActivity 类也可以设计带有标签的布局。不过，这需要掌握一定的 TabHost 和 TabWidget 控件底层知识。在不使用 TabActivity 的前提下，要在 XML 布局文件中定义标签，那么需要从 TabHost（例如 TabHost1）着手。在 TabHost 内部，包含一个纵向的 LinearLayout，其中包含一个 TabWidget（id 为@android:id/tabs）和一个 FrameLayout（id 为@android:id/tabcontent）。然后，将每个标签的内容定义在 FrameLayout 里。

在 XML 中适当地定义了 TabHost 之后，你需要在 Activity 的 onCreate() 方法中使用 TabHost 的 setup() 方法载入并初始化它。首先，你需要为每一个标签创建一个 TabSpec，使用 setIndicator() 方法设置标签指示器（tab indicator），使用 setContent() 方法设置标签内容。然后，使用 TabHost 的 addTab() 方法添加每一个标签。最后，你需要使用诸如 setCurrentTabByTag() 方法为 TabHost 设定默认标签。

8.4.3 添加滚动支持

提供屏幕纵向滚动的一个简单方法是使用 ScrollView（垂直滚动）和 HorizontalScrollView（水平滚动）控件。这两种控件都可以用来作为包裹容器，其中的所有子 View 控件都拥有连续的滚动条。ScrollView 和 HorizontalScrollView 控件虽然只能有一个子控件，不过这个子控件可以是一个布局，如 LinearLayout，这样就可以包含所有“真正”用于滚动的子控件了。图 8.11 展示了带有和不带有 ScrollView 的控件。

Download at http://www.pin5i.com/

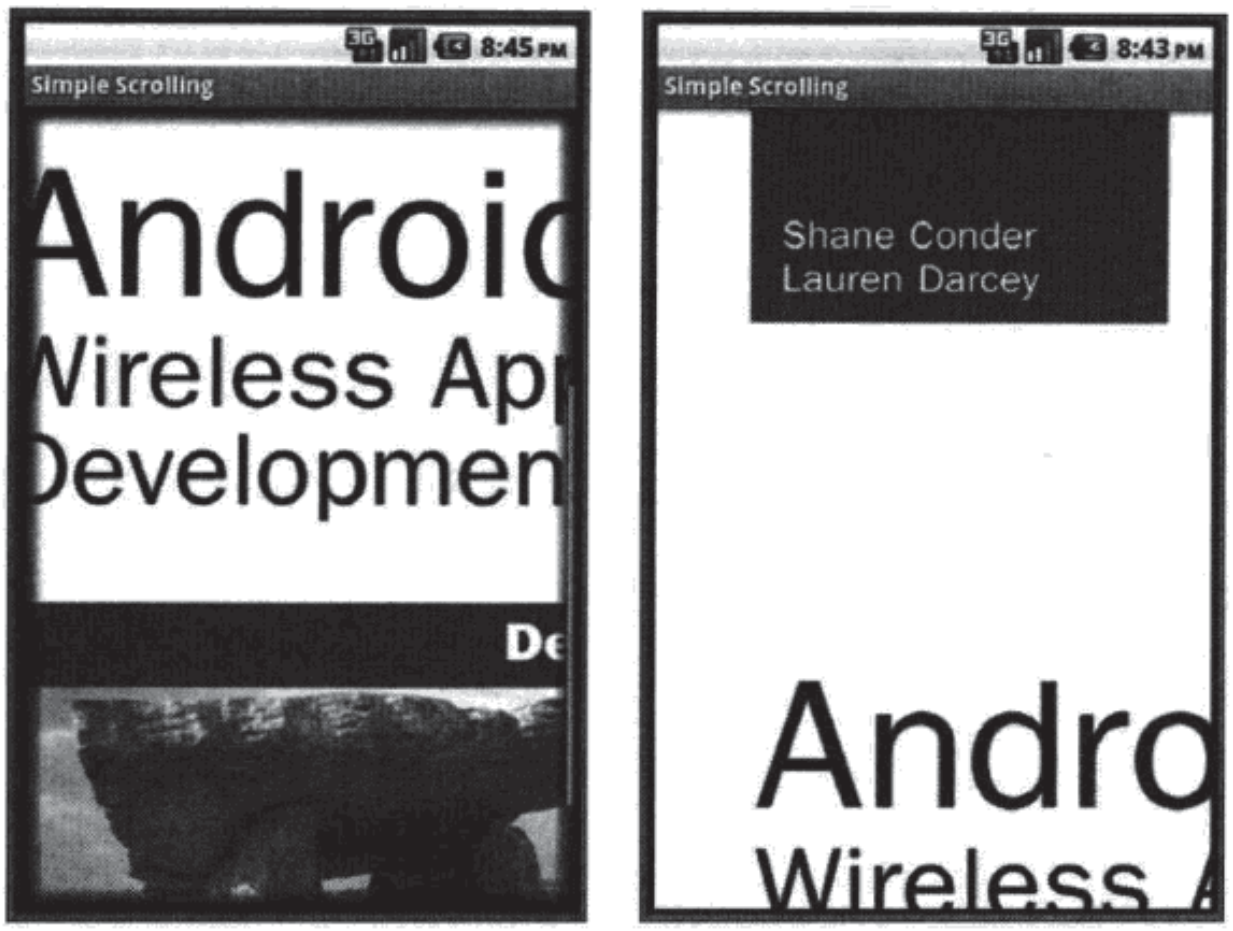


图 8.11 不带有（左）和带有（右）ScrollView 的控件

Tip

一些有关滚动的代码示例都来自于 SimpleScrolling 应用程序。在本书的网站上可以下载 SimpleScrolling 应用程序的源代码。

8.4.4 探索其他视图容器

Android SDK 还提供了其他许多显示对象和控件。下面列出了其中的一部分。

（1）开关（Switcher）

ViewSwitcher 控件只包含两个子 View 对象，并且一次只会显示其中的一个。它在这两个对象当中进行切换，动画亦是如此。通常使用的是 ImageSwitcher（如图 8.12 所示）和 TextSwitcher 对象。它们都提供设置新的子 View 的方法，这一新的 View 可以是一个 Drawable 资源，也可以是一个文本字符串，并且随后从当前显示内容以动画的方式切换到新内容。

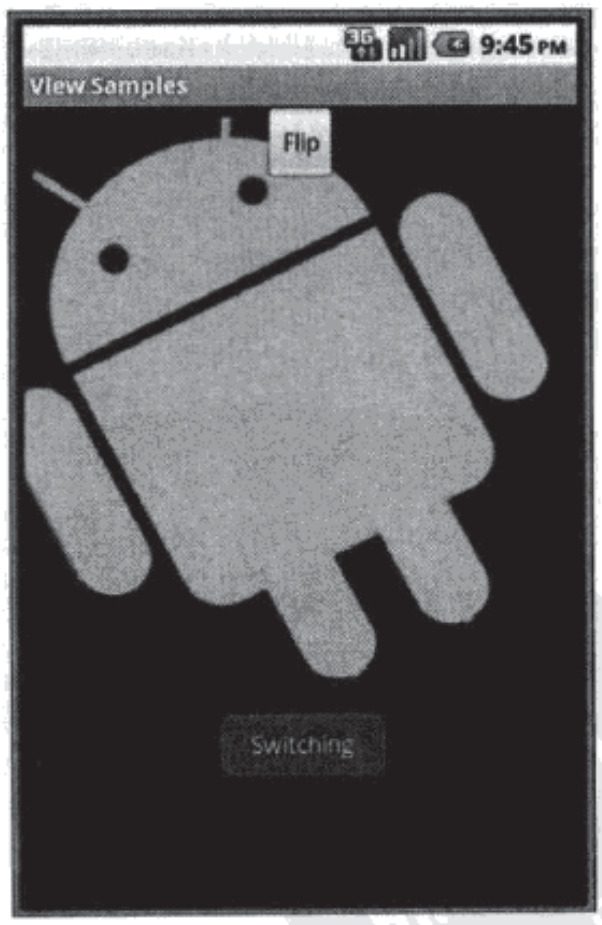


图 8.12 正在切换两个 Drawable 资源的 ImageSwitcher

（2）滑动抽屉（Sliding Drawer）

另一种 View 容器是 SlidingDrawer 控件。这个控件包含两个部分：一个手柄（handle）和一个容器视图。用户可以拖动手柄来显示内部的内容，然后可以再拖动手柄关闭它。SlidingDrawer 既可以垂直显示也可以水平显示，并且始终包含于一个占有较大屏幕控件的布局内。这使得 SlidingDrawer 特别适用于进行应用程序配置，如游戏控制。用户可以拉开抽屉，暂停游戏，修改一些特性，然后关闭 SlidingDrawer 返回游戏。图 8.13 显示了当一个 SlidingDrawer 拉开时的样子。

Download at http://www.pin5i.com/

图8.13 打开SlidingDrawer以显示内容

小 结

Android SDK提供了很多用于设计美观、实用的屏幕界面的强大方法。本章向你介绍了其中的许多方法。你首先学习到的是许多Android布局控件，它们可以控制其他Widget在屏幕上的显示。在很多情况下，它们允许你在某个单一屏幕下进行设计，并且能够在大多数不同尺寸和高宽比的屏幕下工作。

之后你学习了其他一些能够包含视图并且以特定的方式将其组织和排布在屏幕上的容器对象。这些对象包括诸如标签在内的显示框架（paradigm），标签的使用类似于我们日常生活中所接触到的文件夹标签。此外，我们还学习了其他各种不同的控件，它们可以在屏幕上以某种易于阅读与浏览的方式显示数据。现在，你已经掌握了开发实用而美观的应用程序用户界面所需的全部工具。

Download at http://www.pin5i.com/

第9章

绘图与动画

本章讨论 Android 内建的绘图与动画特性，包括创建自定义 View 类，以及使用 Canvas 和 Paint 在屏幕上进行图形和文本的绘制等。我们还将讨论在屏幕上显示动画的各种方法。

9.1 屏幕绘图

在第 7 章“探索用户界面屏幕元素”和第 8 章“使用布局设计用户界面”中，我们已经讲解了有关布局和各种可用的 Android View 类的知识，它可以使屏幕设计变得简洁而高效。现在，我们需要考虑一些稍微底层的问题，探讨如何在屏幕上绘制对象。不过现在我们将在稍微底层的环境下工作，讨论屏幕绘制有关的话题。使用 Android，我们可以在屏幕上绘制各种诸如 PNG 和 JPG 这样的图像，以及文本和基本图形。我们可以使用各种颜色、样式和渐变来绘制它们，还可以使用标准的图像变换方法来对它们进行修改。我们甚至还可以让这些可绘制的对象动起来，以制造生动的效果。

Tip

本章中的许多代码示例都来自于 SimpleDrawing 应用程序。在本书的网站上可以下载到 SimpleDrawing 应用程序的源代码。

要在屏幕上进行绘制，你需要一个可用的 Canvas（画布）对象。通常，我们按照自己的目的，通过继承 View 类，并且实现 onDraw() 方法，来获得一个 Canvas。

例如，下面是一个简单的名为 ViewWithRedDot 的 View 子类。我们覆写了 onDraw() 方法，以控制 View 的外观，在本例中，它在黑色的背景下绘制了一个红色的圆。

```
private static class ViewWithRedDot extends View {
    public ViewWithRedDot(Context context) {
        super(context);
    }

    @Override
    protected void onDraw(Canvas canvas) {
        canvas.drawColor(Color.BLACK);
        Paint circlePaint = new Paint();
        circlePaint.setColor(Color.RED);
        canvas.drawCircle(canvas.getWidth()/2,
            canvas.getHeight()/2,
            canvas.getWidth()/3, circlePaint);
    }
}
```

Download at http://www.pin5i.com/

之后我们可以像使用其他布局一样地使用这个 View。例如，我们可以使用下面的代码覆写 Activity 的 onCreate() 方法：

```
setContentView(new ViewWithRedDot(this));
```

最终运行的结果应该类似于图 9.1。

图 9.1 ViewWithRedDot 视图在黑色的画布背景上绘制了一个红色的圆

1. 理解画布

Canvas（android.graphics.Canvas）对象使用一片矩形区域，并且按照顺序接受绘制方法的调用。可用的绘制方法包括图像、文本、图形的绘制，以及对裁剪区域（clipping region）的支持。

Canvas 的尺寸由容器视图划定。你可以通过 getHeight() 和 getWidth() 方法来获得 Canvas 的尺寸。

2. 理解 Paint

在 Android 中，Paint（android.graphics.Paint）对象不止用来存储颜色。Paint 类封装了样式、复杂的颜色和渲染信息，它们可以用在诸如图形、图片和给定 Typeface 的文本绘制当中。

（1）使用 Paint 色彩

你可以使用 setColor() 方法来设置 Paint 的颜色。标准颜色在 android.graphics.Color 类中已经进行了预定义。例如，下面的代码将 Paint 的颜色设置为了红色：

```
Paint redPaint = new Paint();
redPaint.setColor(Color.RED);
```

（2）使用 Paint 抗锯齿

抗锯齿功能使得许多图像（无论是图形还是字体（Typeface））都能够在屏幕上更平滑地显示出来。这一属性是在某个对象的 Paint 中进行设定的。

例如，下面的代码实例化了一个带有抗锯齿功能的 Paint 对象：

```
Paint aliasedPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
```

（3）使用 Paint 样式（Style）

Paint 样式（style）控制一个对象的颜色填充方式。例如，下面的代码实例化了一个 Paint 对象，并且将其样式属性设置为了 STROKE，这意味着对象应该以线框的方式被描绘出来，而不进行填充（默认地）：

```
Paint linePaint = new Paint();
linePaint.setStyle(Paint.Style.STROKE);
```

（4）使用 Paint 渐变

你可以使用渐变子类中的一个来创建色彩渐变。不同的渐变类（如图 9.2 所示），包括

Download at http://www.pin5i.com/

LinearGradient、RadialGradient 和 SweepGradient，均继承自父类 android.graphics.Shader。

所有的渐变都至少需要两种颜色：起始色和终止色，但也可以包含由数组定义的任意数量的颜色。不同类型渐变之间的区别在于渐变所“流动”的方向。另外，根据所需，渐变还可以被设置为“镜像”或“重复”。

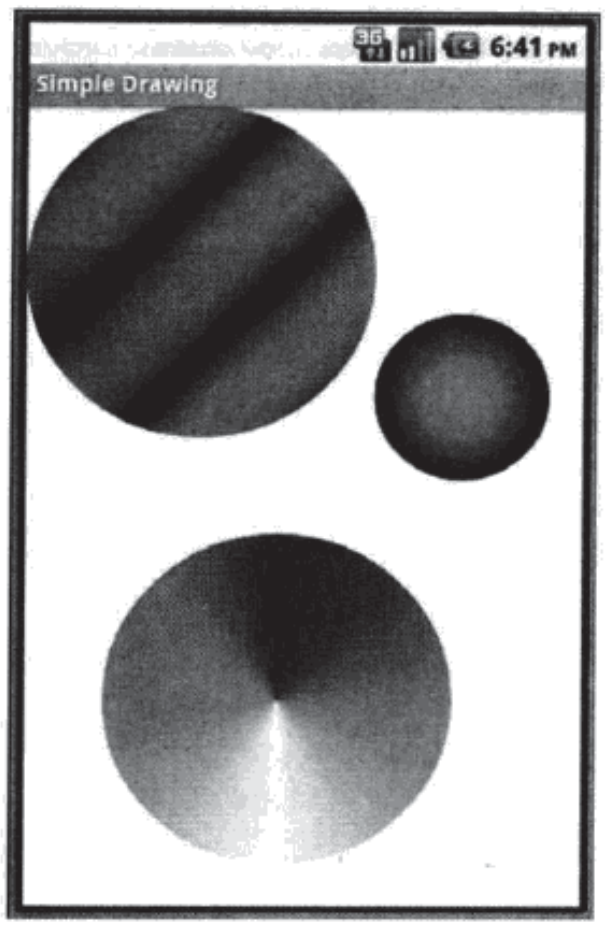


图 9.2 LinearGradient（上）、RadialGradient（右）和 SweepGradient（下）的示例

你可以使用 setShader()方法来设置 Paint 的渐变。

（5）使用线形渐变

线形渐变是一种沿着某条直线方向改变颜色的渐变类型。图 9.2 左上方的圆形使用了镜像方式的“黑—红”渐变。

这可以通过在绘制 Canvas 之前创建一个 LinearGradient 并且调用 Paint 的 setShader()方法来实现，如下所示：

```
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.LinearGradient;
import android.graphics.Paint;
import android.graphics.Shader;
...
Paint circlePaint = new Paint(Paint.ANTI_ALIAS_FLAG);
LinearGradient linGrad = new LinearGradient(0, 0, 25, 25,
    Color.RED, Color.BLACK,
    Shader.TileMode.MIRROR);
circlePaint.setShader(linGrad);
canvas.drawCircle(100, 100, 100, circlePaint);
```

（6）使用辐射形渐变

辐射渐变是一种以某一点作为起点，并以圆周的方式向外辐射来改变颜色的渐变类型。图 9.2 右侧的小圆使用了“绿—黑”辐射渐变。

这可以通过在绘制 Canvas 之前创建一个 RadialGradient 并且调用 Paint 的 setShader()方法来实现，如下所示：

```
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.LinearGradient;
import android.graphics.Paint;
import android.graphics.Shader;
...
Paint circlePaint = new Paint(Paint.ANTI_ALIAS_FLAG);
RadialGradient radGrad = new RadialGradient(250,
    175, 50, Color.GREEN, Color.BLACK,
    Shader.TileMode.MIRROR);
circlePaint.setShader(radGrad);
canvas.drawCircle(250, 175, 50, circlePaint);
```

Download at http://www.pin5i.com/

（7）使用扫描形渐变

扫描形（sweep）渐变是一种以“切饼”的方式来改变颜色的渐变类型。这种类型的渐变多用来作为“色彩选择器（color chooser）”。图 9.2 底部的圆使用了“红—黄—绿—蓝—品红”扫描形渐变。

这可以通过在绘制 Canvas 之前创建一个 SweepGradient 并且调用 Paint 的 setShader()方法来实现，如下所示：

```
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.SweepGradient;
import android.graphics.Paint;
import android.graphics.Shader;
...
Paint circlePaint = new Paint(Paint.ANTI_ALIAS_FLAG);
SweepGradient sweepGrad = new
    SweepGradient(canvas.getWidth()-175,
    canvas.getHeight()-175,
    new int[] { Color.RED, Color.YELLOW, Color.GREEN,
    Color.BLUE, Color.MAGENTA }, null);

circlePaint.setShader(sweepGrad);
canvas.drawCircle(canvas.getWidth()-175,
    canvas.getHeight()-175, 100,
    circlePaint);
```

（8）使用 Paint 功能（Utility）绘制文本

Paint 类还包含许多功能（Utility）和特性，它们用于在屏幕上渲染不同字体和样式的文本。现在，我们可以开始在屏幕上绘制一些文字了。

9.2 使用文字

Android 提供几种默认的字体和样式。应用程序也可以使用自定义的字体，这需要将字体文件作为其一个组件（asset）包含进应用程序，并且像使用资源那样使用 AssetManager 载入字体。

9.2.1 使用默认字体

默认地，Android 使用 Sans Serif 字体，但也可以使用等宽（Monospace）字体和 Serif 字体。下面的代码片段在 Canvas 上绘制了一段带有抗锯齿功能的文本，它使用的是默认字体（Sans Serif）：

```
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Typeface;
...
Paint mPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
Typeface mType;

mPaint.setTextSize(16);
mPaint.setTypeface(null);

canvas.drawText("Default Typeface", 20, 20, mPaint);
```

Download at http://www.pin5i.com/

你也可以载入另外的字体，例如“等宽”字体：

```
Typeface mType = Typeface.create(Typeface.MONOSPACE,
    Typeface.NORMAL);
```

也许你还会用到“斜体”文字，在这种情况下，你只需要简单地设置字体族（font family）和样式即可：

```
Typeface mType = Typeface.create(Typeface.SERIF,
    Typeface.ITALIC);
```

Warning

不是所有的字体族都支持所有的字体样式。你需要检测以确保设备支持你所期望的字体样式。

另外，你还可以设置字体的其他属性，如抗锯齿、下画线和删除线等，它们可以通过 `Paint` 对象的 `setFlags()` 方法来实现。

```
mPaint.setFlags(Paint.UNDERLINE_TEXT_FLAG);
```

图 9.3 显示了 Android 默认支持的 `Typeface` 族和样式。

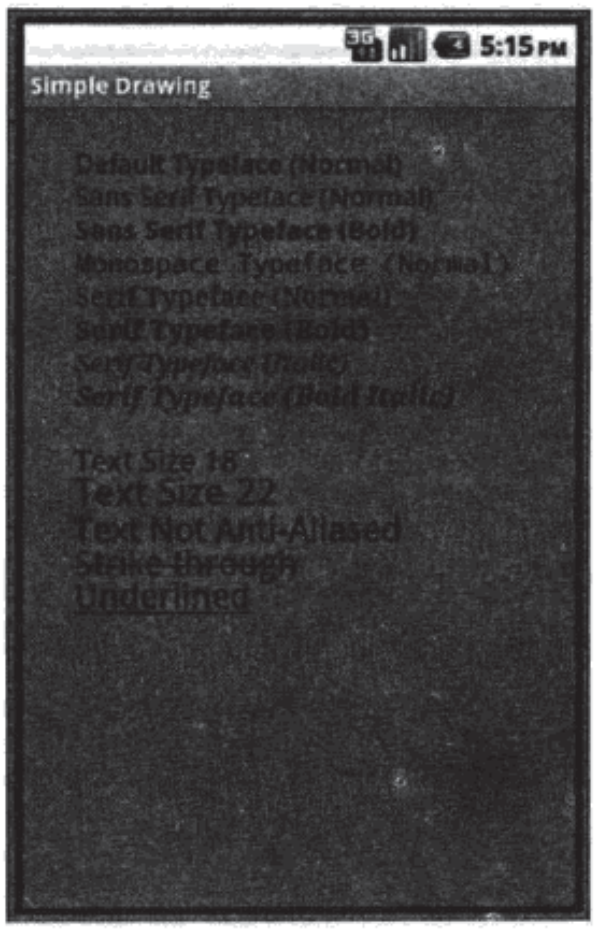


图 9.3 一些 Android 支持的字体和字体样式

9.2.2 使用自定义字体

你可以很容易地在应用程序中使用自定义字体，只需要将字体文件以应用程序组件（asset）的方式包含进来，并且在需要的时候载入即可。自定义的字体可以用于需要制造个性化的外观与感受（look-and-feel）时，也可以用来实现一些原本不支持的语言符号或自定义符号。

例如，我们也许希望使用一种简易的“象棋”字体来实现一个简单、可缩放的国际象棋游戏。“象棋”字体包含了所有用来实现棋盘和棋子的符号元素。Hans Bodlaender 友好地提供了一种免费的“象棋”字体——*Chess Utrecht*。使用这种 *Chess Utrecht* 字体，字母“Q”代表一个白方格上的黑王后，而字母“q”则绘制一个白方格上的白王后，如此等等。这种俏

Download at http://www.pin5i.com/

皮的字体可在网站 www.chessvariants.com/d.font/utrecht.html 上下载，它是一个名为“Chess1.ttf”的字体文件。

要使用诸如 *Chess Utrecht* 这样的自定义字体，你只需要简单地从网站上下载它，并且将 chess1.ttf 文件复制到磁盘的工程目录：/assets/fonts/chess1.ttf。

现在，你可以像载入任何资源那样使用程序载入字体了：

```
import android.graphics.Typeface;
import android.graphics.Color;
import android.graphics.Paint;
...
Paint  mPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
Typeface mType =
     Typeface.createFromAsset(getContext().getA
ssets(),
     "fonts/chess1.ttf");
```

之后你就可以使用这个 *Chess Utrecht* 字体在屏幕上“绘制”一个棋局了（如图 9.4 所示），当然你还需要遵照一定的字符序列。

图 9.4 使用 *Chess Utrecht* 字体绘制的棋局

9.2.3 测量文字所需的屏幕尺寸

你可以使用 measureText() 和 getTextBounds() 方法来测量一个给定 Paint 的文字尺寸，以确定这些文字需要占用多大的矩形面积。

9.3 使用位图

你会发现在 android.graphics 包中使用诸如位图（包括 NinePatch）这样的图像有很多好处。位图核心类为 android.graphics.Bitmap。

1. 在画布上绘制位图

你可以将位图绘制在一个 Canvas 上，比如使用某一个 drawBitmap() 方法在 View 的 onDraw() 方法中实现。例如，下面的代码将载入一个 Bitmap 并将其绘制到画布上：

```
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
...
Bitmap pic = BitmapFactory.decodeResource(getResources(),
     R.drawable.bluejay);
canvas.drawBitmap(pic, 0, 0, null);
```

2. 缩放位图

也许你需要将图像缩小到一个更小的尺寸。在这种情况下，你可以使用 createScaled Bitmap() 方法，如下所示：

```
Bitmap sm = Bitmap.createScaledBitmap(pic, 50, 75, false);
```

Download at http://www.pin5i.com/

另外，你还可以通过 getWidth() 和 getHeight() 方法所返回的值来维持 Bitmap 的高宽比。

3. 使用 Matrix 变换位图

你可以使用非常有用的 Matrix 类来实现 Bitmap 图像的变形操作（如图 9.5 所示）。Matrix 类可以用来执行诸如对图像镜像或变形的操作，当然它还有许多其他的功能。

下面的代码使用 createBitmap() 方法生成了一个新的 Bitmap，它是一个现存的名为 pic 的 Bitmap 的镜像：

```
import android.graphics.Bitmap;
import android.graphics.Matrix;
...
Matrix mirrorMatrix = new Matrix();
mirrorMatrix.preScale(-1, 1);

Bitmap mirrorPic = Bitmap.createBitmap(pic, 0, 0,
   pic.getWidth(), pic.getHeight(),mirrorMatrix,
flase);
```

除了镜像之外，你还可以使用 Matrix 对图像进行旋转 30° 的操作：

```
Matrix mirrorAndTilt30 = new Matrix();
mirrorAndTilt30.preRotate(30);
mirrorAndTilt30.preScale(-1, 1);
```

图 9.5 使用 Android 位图类对同一幅图像进行缩放、倾斜和镜像

在图 9.5 中，你可以看到结合了不同的倾斜、镜像等 Matrix 变形之后的效果。当你不再使用 Bitmap 时，可以使用 recycle() 方法释放内存：

```
pic.recycle();
```

Android SDK 中还提供了其他各种类型的 Bitmap 特效和功能，不过其数量众多而且已经超过了本书的范围，在此我们不做进一步讲解。你可以查看 android.graphics 包以获取更多的信息。

9.4 使用图形

你可以使用 ShapeDrawable 类并结合其他不同类型专门的 Shape 类，来绘制一些常用的基本图形，例如矩形和椭圆。你可以通过 XML 资源文件定义 Paintable 可绘制（drawable）资源，但更常见的做法，特别是对于一些复杂的图形，往往是通过程序来实现绘制。

Tip

这一部分中的许多代码示例都来自于 SimpleShapes 应用程序。在本书的网站上可以下载 SimpleShapes 应用程序的源代码。

9.4.1 用 XML 资源定义图形

在第 6 章“管理应用程序资源”中，我们向你展示了如何使用位于/res/drawable 资

Download at http://www.pin5i.com/

源目录下的 XML 文件类定义诸如矩形这样的基本图形。

以下名为/res/drawable/green_rect.xml 的资源文件描述了一个简单的绿色矩形：

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid android:color="#0f0" />
</shape>
```

随后你可以载入这一图形资源，并将其设置为 Drawable，如下所示：

```
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageResource(R.drawable.green_rect);
```

你应该注意到许多 Paint 属性都可以通过 XML 作为 Shape 定义的一部分来进行设定。例如，下面的 Oval（椭圆）图形就使用了线形渐变（红—白）和边框（stroke）样式来进行定义：

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="oval">
    <solid android:color="#f00" />
    <gradient android:startColor="#f00"
        android:endColor="#fff"
        android:angle="180" />
    <stroke android:width="3dp" android:color="#00f"
        android:drawWidth="5dp" android:dashGap="3dp" />
</shape>
```

9.4.2 使用程序定义图形

你还可以使用程序来定义这些 ShapeDrawable 实例。不同的图形均在 android.graphics.drawable.shapes 包中以不同类的方式提供。例如，前面提到的“绿色矩形”则可以通过程序进行如下定义：

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.RectShape;
...
ShapeDrawable rect = new ShapeDrawable(new
RectShape());
rect.getPaint().setColor(Color.GREEN);
```

之后，你可以将这个 Drawable 直接设置给 ImageView：

```
ImageView iView = (ImageView)findViewById(R.id.
ImageView1);
iView.setImageDrawable(rect);
```

图 9.6 显示了上面实现的这个绿色矩形。

图 9.6 一个绿色矩形

9.4.3 绘制各种图形

android.graphics.drawable.shapes 包中囊括的部分图形如下：

① 矩形（和正方形）；

② 圆角矩形；

Download at http://www.pin5i.com/

③ 椭圆（和圆）；
④ 弧和线；
⑤ 以路径（Path）方式定义的其他图形。

这些图形可以以 Drawable 资源的方式直接在 ImageView 视图中使用，或者你也可以找到相应的方法将这些基本图形创建到 Canvas 中。

1. 绘制矩形

绘制矩形和正方形（即长、宽相等的矩形）差不多相当于从一个 RectShape 类中创建一个 ShapeDrawable。RectShape 对象不含有任何尺寸，但被限定在其容器对象内，在这里是 ShapeDrawable。你可以为 ShapeDrawable 设置一些基本的属性，例如 Paint 颜色和默认的尺寸。

下面，我们尝试创建一个品红色的矩形，其长为 100 像素，宽为 2 像素，这使得它看起来像一条水平直线。随后我们将其设置为可绘制（drawable）对象，这样，图形就可以显示在 ImageView 中了：

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.RectShape;
...
ShapeDrawable rect = new ShapeDrawable(new RectShape());
rect.setIntrinsicHeight(2);
rect.setIntrinsicWidth(100);
rect.getPaint().setColor(Color.MAGENTA);

ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageDrawable(rect);
```

2. 绘制圆角矩形

你可以创建圆角矩形，它可以很好地用来制作自定义按钮。只需要从 RoundRectShape 对象创建一个 ShapeDrawable 即可。RoundRectShape 需要一个带有 8 个浮点值的数组，它们指定了圆角的半径。例如，下面的代码创建了一个简单的青色圆角矩形：

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.RoundRectShape;
...
ShapeDrawable rndrect = new ShapeDrawable(
    new RoundRectShape( new float[] { 5, 5, 5, 5, 5, 5, 5, 5 },
    null, null));
rndrect.setIntrinsicHeight(50);
rndrect.setIntrinsicWidth(100);
andrect.getPaint().setColor(Color.CYAN);
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageDrawable(rndrect);
```

图 9.7 显示了上面代码的运行结果，即一个圆角矩形。

如果你愿意的话，还可以创建一个包含在外矩形之中的内圆角矩形。下面的代码创建了一个这样的矩形，其外部是一个圆角的白色矩形，而内部也是一个圆角矩形：

```
import android.graphics.drawable.ShapeDrawable;
```

Download at http://www.pin5i.com/

```
import android.graphics.drawable.shapes.RoundRectShape;
...
float[] outerRadii = new float[]{ 6, 6, 6, 6, 6, 6, 6, 6 };
RectF insetRectangle = new RectF(8, 8, 8, 8);
float[] innerRadii = new float[]{ 6, 6, 6, 6, 6, 6, 6, 6 };

ShapeDrawable rndrect = new ShapeDrawable(
    new RoundRectShape(
    outerRadii,insetRectangle , innerRadii));

rndrect.setIntrinsicHeight(50);
rndrect.setIntrinsicWidth(100);
rndrect.getPaint().setColor(Color.WHITE);
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageDrawable(rndrect);
```

图 9.8 显示了这个内外圆角矩形的效果。

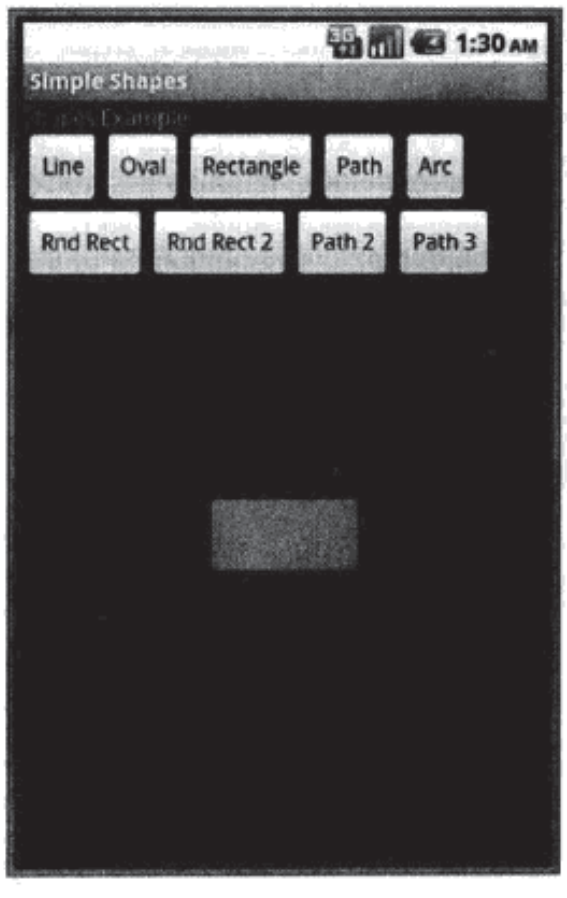


图 9.7 青色的圆角矩形

图 9.8 一个内嵌圆角矩形的白色圆角矩形

3. 绘制椭圆和圆

你可以通过使用 OvalShape 对象创建一个 ShapeDrawable 来生成椭圆或圆（即高、宽相等的椭圆）。OvalShape 对象不具有尺寸，但限定在其容器对象的区域内，在这里即为 ShapeDrawable。你可以设置 ShapeDrawable 的一些基本属性，比如 Paint 颜色和默认尺寸。例如，下面的代码创建了一个高为 40 像素、宽为 100 像素的椭圆，它看起来像一个飞盘：

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.OvalShape;
...
ShapeDrawable oval = new ShapeDrawable(new OvalShape());
oval.setIntrinsicHeight(40);
oval.setIntrinsicWidth(100);
oval.getPaint().setColor(Color.RED);
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageDrawable(oval);
```

Download at http://www.pin5i.com/

图 9.9 显示了这个红色的椭圆。

4. 绘制扇形（Arc）

你可以绘制一个扇形（Arc），根据你所指定的旋转角度，它可能看起来像一个饼图，或者一个“吃豆游戏”的小人（Pac-Man）。你可以通过使用一个 ArcShape 对象创建一个 ShapeDrawable 来生成扇形。ArcShape 对象需要两个参数：startAngle（起始角）和 sweepAngle（旋转角）。startAngle 以三点钟方向作为起点。正的 sweepAngle 值以顺时针方向旋转，而负值则按逆时针方向旋转。你可以使用“0、360”这两个值来绘制一个圆。

下面的代码创建了一个扇形，它看起来像一个品红的“吃豆小人”。

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.ArcShape;
...
ShapeDrawable pacMan =
    new ShapeDrawable(new ArcShape(0, 345));
pacMan.setIntrinsicHeight(100);
pacMan.setIntrinsicWidth(100);
pacMan.getPaint().setColor(Color.MAGENTA);
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageDrawable(pacMan);
```

图 9.10 显示了上面代码的运行结果。

图 9.9 一个红色椭圆

图 9.10 345° 的品红色扇形（代表“吃豆小人”）

5. 绘制路径

你可以通过指定一系列的路径点来生成任何你想要的图形。android.graphics.Path 类封装了一系列画直线和曲线的方法，使用这些方法可以绘制出更大的图形。

例如，下面的 Path 定义了形状不太规则的五星图案：

Download at http://www.pin5i.com/

```
import android.graphics.Path;
...
Path p = new Path();
p.moveTo(50,0);
p.lineTo(25,100);
p.lineTo(100,50);
p.lineTo(0,50);
p.lineTo(75,100);
p.lineTo(50,0);
```

之后，你可以将这个五星 Path 封装进一个 PathShape 中，创建一个 ShapeDrawable，并且将其涂成黄色。

```
import android.graphics.drawable.ShapeDrawable;
import android.graphics.drawable.shapes.PathShape;
...
ShapeDrawable star =
    new ShapeDrawable(new PathShape(p, 100, 100));
star.setIntrinsicHeight(100);
star.setIntrinsicWidth(100);
star.getPaint().setColor(Color.YELLOW);
```

默认地，上面的代码将生成一个使用 Paint 颜色（即黄色）填充的五星图形（如图 9.11 所示）。或者，你也可以设置 Paint 样式为 Stroke，以绘制线形的五星。

```
star.getPaint().setStyle(Paint.Style.STROKE);
```

绘制的结果应该和图 9.12 所示的相似。

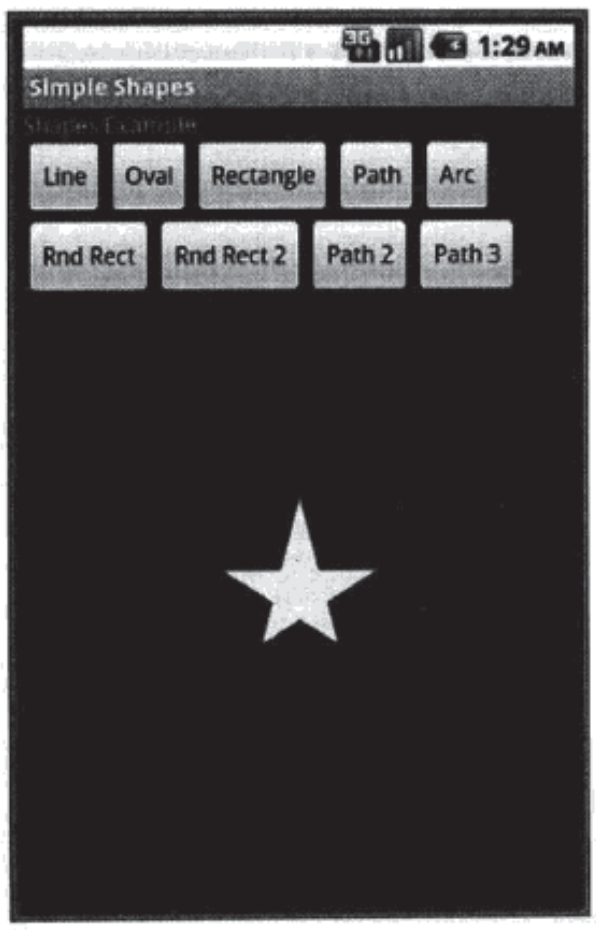


图 9.11 一个黄色的五星

图 9.12 使用 Paint 线框样式绘制的黄色五星

Tip

对于 Android SDK 中提供的图形支持技术的讨论将贯穿本书始终。当你已经熟悉了基本的绘图知识后，我们强烈地建议你查看 Android 软件开发包（Software Development Kit，SDK）中提供的“APIDemo”示例应用程序。

Download at http://www.pin5i.com/

9.5 使用动画

Android 平台支持以下三种类型的图形动画：

① 动态 GIF 图像；

② 逐帧动画；

③ 补间（Tweened）动画。

动态 GIF 将动画帧存储在图像中，你只需要像任何其他图像可绘制资源一样简单地将这些 GIF 包含进去。对于逐帧动画，开发人员必须提供动画的所有图像帧。不过，对于补间（Tweened）动画而言，只需要使用一幅图像就可以了，当然这依赖于可以使用程序实现的变换。

Tip

本章中的许多代码示例都来自于 ShapeShifter 应用程序。在本书的网站上可以下载 ShapeShifter 应用程序的源代码。

9.5.1 使用逐帧动画

你可以把逐帧动画看做是一本数字连环画册（flipbook），其中的每一幅图像均同上一幅有微小差别。将这些图像快速地按顺序显示在屏幕上则会给人以动态的效果。这种技术被称作逐帧动画，它经常使用以动态 GIF 图像的形式出现在网络上。

逐帧动画比较适合用于呈现复杂的图像变换，这些变换一般很难通过程序实现。

例如，我们可以使用一个含有三幅图片的序列来创建一段“精灵杂耍（genie juggling）”动画，如图 9.13 所示。

图 9.13 “精灵杂耍”动画的三幅帧图像

在每一帧中，“精灵”保持固定，但“礼物”的位置却有微小的变化。动画的流畅性来源于足够数量的图像帧和适当的切换速度。

下面的代码演示了如何载入三幅 Bitmap（即上面的三幅“精灵”图像帧）并且创建一个 AnimationDrawable。之后我们将这个 AnimationDrawable 设置为 ImageView 的背景资源并且启动了动画。

```
ImageView img = (ImageView)findViewById(R.id.ImageView1);

BitmapDrawable frame1 = (BitmapDrawable)getResources().
    getDrawable(R.drawable.f1);
BitmapDrawable frame2 = (BitmapDrawable)getResources().
    getDrawable(R.drawable.f2);
BitmapDrawable frame3 = (BitmapDrawable)getResources().
    getDrawable(R.drawable.f3);

int reasonableDuration = 250;
AnimationDrawable mAnimation = new AnimationDrawable();

mAnimation.addFrame(frame1, reasonableDuration);
mAnimation.addFrame(frame2, reasonableDuration);
mAnimation.addFrame(frame3, reasonableDuration);

img.setBackgroundDrawable(mAnimation);
```

Download at http://www.pin5i.com/

要将动画设置为重复循环播放，可以调用setOneShot()方法：

```
mAnimation.setOneShot(false);
```

要启动动画的播放，则可以调用start()方法：

```
mAnimation.start();
```

我们还可以通过stop()方法来停止动画的播放：

```
mAnimation.stop();
```

虽然在这个例子中我们只使用了ImageView的背景，但是你也可以使用各种类型的View Widget来播放动画。例如，你可以使用ImageSwitcher视图并且通过一个计时器来修改Drawable资源的显示。这种类型的操作最好能够在另外的线程中进行。上面的动画应该看起来和图9.14类似，你可以想象一下它动起来的情景。

图9.14 Android模拟器中的“精灵”动画

9.5.2 使用补间动画

使用补间动画，你需要提供一个单独的Drawable资源，它可以是一幅Bitmap图像（如图9.15左侧所示）、一个ShapeDrawable、TextView（如图9.15右侧所示）或者任何其他类型的View对象，动画中间的帧则由系统生成。Android提供对于几种常见图像变换的补间支持，包括Alpha、旋转、缩放和移动（translate）。你可以将补间动画应用于任何View，无论其为一个包含Bitmap的ImageView，还是Drawable图形，或者诸如TableLayout这样的布局。

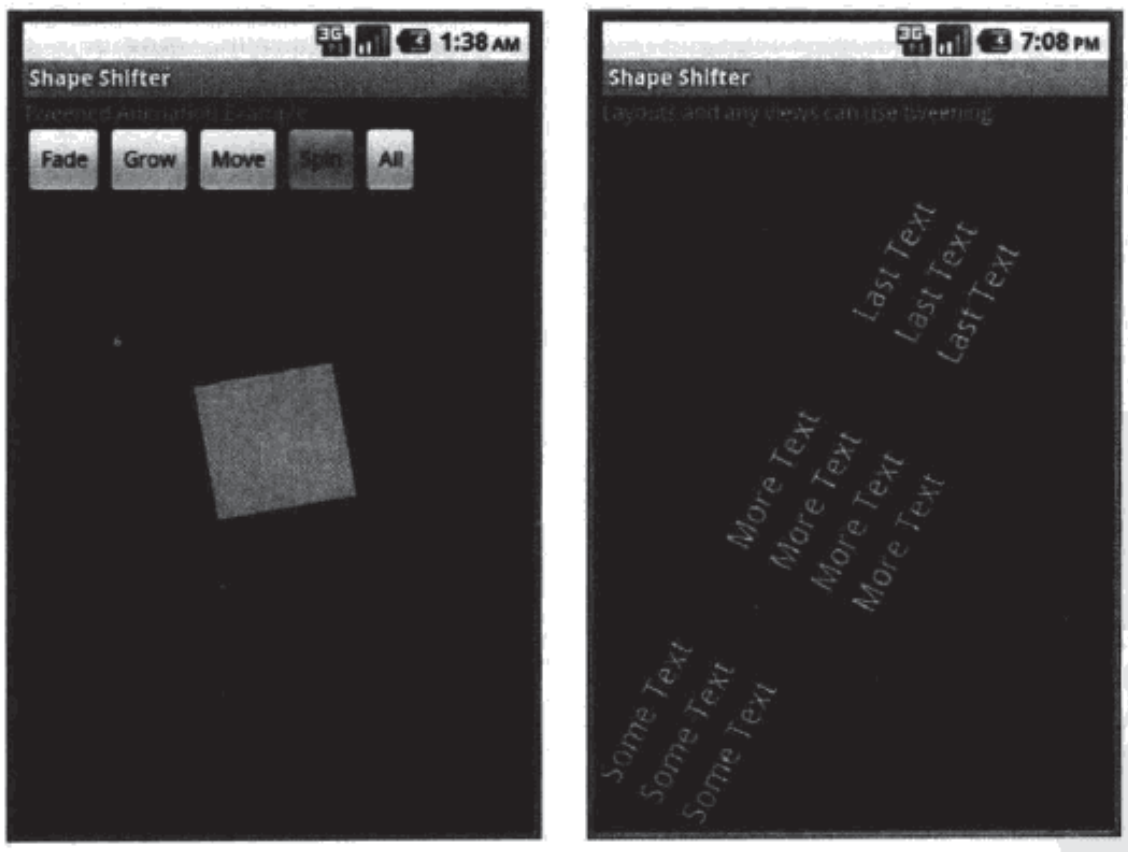


图9.15 旋转的绿色矩形Drawable（左）和TableLayout（右）

1. 定义补间变换

你可以将补间动画定义为XML资源文件，或者由程序生成。所有的补间动画都具有一

Download at http://www.pin5i.com/

些共同的属性，包括何时开始、持续多久以及在动画完成时是否回到开头等。

（1）以 XML 资源的方式定义补间动画

在第 6 章中，我们向你展示了如何将动画序列作为一个包含特定格式的 XML 文件进行存储，该文件位于/res/anim/资源文件夹中。例如，下面这个名为/res/anim/spin.xml 的资源文件描述了一个简单的持续 5s 的旋转动画：

```
<?xml version="1.0" encoding="utf-8" ?>
<set xmlns:android
    = "http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <rotate
        android:fromDegrees="0"
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="5000" />
</set>
```

（2）使用程序定义补间动画

你也可以使用程序来定义补间动画。不同的动画类型使用包含在 android.view.animation 包中的不同类。例如，前面实现的旋转动画可以像下面这样定义：

```
import android.view.animation.RotateAnimation;
...
RotateAnimation rotate = new RotateAnimation(
    0, 360, RotateAnimation.RELATIVE_TO_SELF, 0.5f,
    RotateAnimation.RELATIVE_TO_SELF, 0.5f);

rotate.setDuration(5000);
```

（3）定义同步和序列补间动画

不同的动画变换可以同时进行，也可以序列化地进行，这可以通过 startOffset 和 duration 两个属性进行设定，它们分别控制动画变换的起点和持续时间。动画变换可以被包含进<tag>标记（在程序中使用 AnimationSet）中以共享属性。

下面名为/res/anim/grow.xml 的动画资源文件包含两种缩放动画变换的设定：首先，我们使用 2.5s 来使对象尺寸翻倍，然后，消耗 1s 的时间让对象恢复到原来的大小。

```
<?xml version="1.0" encoding="utf-8" ?>
<set xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <scale
        android:pivotX="50%"
        android:pivotY="50%"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:toXScale="2.0"
        android:toYScale="2.0" />
    <scale
        android:starOffset="2500"
        android:duration="2500"
        android:pivotX="50%"
        android:pivotY="50%"
        android:fromXScale="1.0"
```

Download at http://www.pin5i.com/

```
            android:fromYScale="1.0"
            android:toXScale="0.5"
            android:toYScale="0.5" />
    </set>
```

（4）载入动画

AnimationUtils 辅助类使得载入动画变得非常简单。下面的代码载入了一个名为/res/anim/grow.xml 的 XML 资源文件并且将其应用到了一个 ImageView 中，它的源资源是一个绿色矩形的图形 Drawable：

```
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
...
ImageView iView = (ImageView)findViewById(R.id.ImageView1);
iView.setImageResource(R.drawable.green_rect);
Animation an =
    AnimationUtils.loadAnimation(this, R.anim.grow);
iView.startAnimation(an);
```

我们可以监听 Animation 事件，包括动画开始、结束以及重复事件，这可以通过实现一个 AnimationListener 类来完成。下面我们实现了一个自定义的 MyListener 类：

```
class MyListener implements Animation.AnimationListener {

    public void onAnimationEnd(Animation animation) {
        // Do at end of animation
    }

    public void onAnimationRepeat(Animation animation) {
        // Do each time the animation loops
    }

    pubilc void onAnimationStart(Animation animation) {
        // Do at start of animation
    }
}
```

之后，你可以注册你的 AnimationListener，如下所示：

```
an.setAnimationListener(new MyListener());
```

2. 探索四种不同的补间变换

现在，我们分别介绍四种类型的补间变换，它们是：

① 透明化变换（Alpha）；

② 旋转（Rotate）；

③ 缩放（Scale）；

④ 移动（Translate）。

（1）使用 Alpha 透明变换

透明化是通过 Alpha 变换来实现的。Alpha 变换可以用来实现一个对象的淡入和淡出，或者在屏幕上将不同对象进行分层。

Alpha 值从 0.0（完全透明，即不可见）到 1.0（完全不透明，即可见）。Alpha 动画涉及一个起始透明度（fromAlpha）值和一个终止透明度（toAlpha）值。

Download at http://www.pin5i.com/

下面的 XML 资源文件片段定义了一段透明度改变的动画，它占用 5s 的时间从完全透明变换为完全可见：

```
<alpha
    android:fromAlpha="0.0"
    android:toAlpha="1.0"
    android:duration="5000">
</alpha>
```

在程序中，你可以使用 android.view.animation 包中的 AlphaAnimation 类来实现同上面代码相同的功能。

（2）使用旋转变换

旋转操作可以使对象在其边界内绕着某个支点沿顺时针或逆时针方向旋转。

旋转操作使用角度进行定义。例如，你可能需要让某个对象沿顺时针方向一次旋转一周。要实现这个效果，你需要设定其 fromDegrees（起始角）属性为 0，并将其 toDegrees（终止角）属性设为 360。如果要按逆时针方向旋转对象，只需要将 toDegrees 属性设为-360 即可。

默认地，对象以(0, 0)坐标点为支点转动，也就是对象的左上角。这对于旋转类似时钟指针这样的对象显得非常适用，但很多时候，你会希望对象绕着其中点旋转，这可以简单地通过修改支点来实现，支点可以是一个固定的数值，也可以是一个百分比。

下面的 XML 资源文件片段定义了一段旋转动画，它花 5s 的时间以顺时针方向旋转一周，其支点定义为对象的中点：

```
<rotate
    android:fromDegrees="0"
    android:toDegrees="360"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="5000" />
```

在程序中，你可以使用 android.view.animation 包中的 RotateAnimation 类来创建相同的动画。

（3）使用缩放变换

你可以使用缩放操作来水平或垂直地拉伸对象。缩放操作被定义为相对比例，也就是说比例值 1.0 相当于 100%，即完整尺寸。要将对象尺寸减半，也就是变为 50%，可以将其目标比例值设为 0.5。

你可以通过设置不同的比例来实现水平和垂直方向的缩放，也可以将其设为相同的值以进行等比例缩放。对于特定的缩放操作，你需要设定 4 个值：起始比例（fromXScale 和 fromYScale）和目标比例（toXScale 和 toYScale）。和之前相似，你仍然可以使用支点，这样任何缩放操作都以给定的(x, y)坐标为固定点进行。

下面的 XML 资源文件片段定义了一段缩放动画，它占用 5s 的时间使对象放大一倍，其支点定义在对象中心：

```
<scale
    android:pivotX="50%"
    android:pivotY="50%"
    android:fromXScale="1.0"
    android:fromYScale="1.0"
    android:toXScale="2.0"
```

Download at http://www.pin5i.com/

```
    android:toYScale="2.0"
    android:duration="5000" />
```

在程序中，你可以使用 android.view.animation 包中的 ScaleAnimation 类来创建相同的动画。

（4）使用移动变换

你可以使用平移（translate）操作来实现对象的移动。平移操作将对象从(*x*, *y*)坐标位置移动到另一个坐标。

要实现平移操作，你需要指定对象坐标值的变化量，或称作“增量（delta）”，它有 4 个值：起始位置（fromXDelta 和 fromYDelta）和相对目标位置（toXDelta 和 toYDelta）。

下面的 XML 资源文件片段定义了一段平移动画，它持续 5s，通过改变 *y* 轴的值将一个对象向上（负值）移动了 100 个单位。我们同时将 fillAfter 属性设为了 true，这样对象就不会在动画完成时突然“跳”回原来的地方了：

```
<translate android:toYDelta="-100"
    android:fillAfter="true"
    android:duration="2500" />
```

在程序中，你可以使用 android.view.animation 包中的 TranslateAnimation 类来创建相同的动画。

（5）使用不同的插补器（Interpolator）

动画插补器（Interpolator）决定变换发生的频率。Android SDK 提供了若干种不同的插补器。

- AccelerateDecelerateInterpolator：动画开始时很慢，然后加速，最后减速停止。
- AccelerateInterpolator：动画开始时很慢，然后加速。
- AnticipateInterpolator：动画先倒着播放，然后正向播放。
- AnticipateOvershootInterpolator：动画先倒着播放，然后正向播放，冲过终点后，再在终点处停止。
- BounceInterpolator：动画“弹”到终点。
- CycleInterpolator：动画循环播放一定次数后进入下一个循环。
- DecelerateInterpolator：动画开始时很快，然后减速。
- LinearInterpolator：动画保持匀速。
- OvershootInterpolator：动画冲过终点，然后在终点处停止。

你既可以在程序中使用 setInterpolator() 方法指定动画所使用的插补器，也可以在动画的 XML 资源中使用 android:interpolator 属性予以定义。

小　结

Android SDK 提供 android.graphics 包，它包含用于在屏幕上绘制图像和文本的各种不同的类。图像库（graphics library）包括 Bitmap 图像功能（utilities）、Typeface 和字体支持、Paint 颜色和样式、不同种类的渐变，以及各种类型的基本或非基本图像等，它们都可以绘制在屏幕上，甚至使用补间或逐帧机制转换为动画。

在第 17 章“通过 OpenGL ES 使用 Android 3D 图形”中，我们将更加深入地探讨有关使用 OpenGL ES 库的 2D 和 3D 渲染的话题。

Download at http://www.pin5i.com/

第四部分

使用常用 Android API

第 10 章 使用 Android 数据和存储 API

第 11 章 使用内容提供器在应用程序间共享数据

第 12 章 使用 Android 网络 API

第 13 章 使用 Android Web API

第 14 章 使用位置服务 API

第 15 章 使用 Android 多媒体 API

第 16 章 使用 Android 电话 API

第 17 章 通过 OpenGL ES 使用 Android 3D 图形

第 18 章 使用 Android NDK

第 19 章 使用 Android 可选硬件 API

Download at http://www.pin5i.com/

第 10 章

使用 Android 数据和存储 API

应用程序的核心是算法（functionality）和数据。在本章中，我们将讨论在 Android 中存储、管理和共享应用程序数据的各种方法。应用程序可以使用不同的方式存储和管理数据。例如，应用程序可以结合应用程序首选项（preferences）、文件系统和内建的 SQLite 数据库支持来进行信息的本地存储。你所选择的方法依赖于你的应用程序需求。在本章中，你将学会各种存储数据和进行数据交互的机制。

10.1 使用应用程序首选项

许多应用程序需要一个称为“共享首选项（shared preferences）”的轻量级的数据存储机制，以存储应用程序状态、简单的用户信息、配置参数和其他类似的信息。

Tip

这一部分中的许多代码示例都来自于 SimplePreferences 应用程序。在本书的网站上可以下载 SimplePreferences 应用程序的源代码。

Android 提供一种简单的首选项系统来存储基本的应用程序数据，它位于 `Activity` 级别，并且可以为应用程序的所有 `Activity` 所共享。你不能在应用程序包之外共享首选项。首选项以一组“键/值”对的方式进行存储。下面列出了首选项设置所支持的一些数据类型：

① 布尔值；
② 浮点值；
③ 整型值；
④ 长整型值；
⑤ 字符串值。

首选项功能可以在 `android.content` 包的 `SharedPreferences` 接口中找到。要在你的应用程序中使用首选项支持，你必须采取以下步骤。

① 获取一个 `SharedPreferences` 对象的实例。
② 创建一个 `SharedPreferences.Editor` 来修改首选项内容。
③ 使用 Editor 修改首选项。
④ 提交你的修改。

10.1.1 创建私有和共享首选项

单独的 Activity 可以拥有它们自己的私有首选项。这些首选项仅供某个特定的 `Activity`

Download at http://www.pin5i.com/

所使用，而不能与应用程序内的其他 Activity 共享。Activity 只拥有一组私有首选项。

下面的代码获取了 Activity 的私有首选项：

```
import android.content.SharedPreferences;
...
SharedPreferences settingActivity =getPreferences(MODE_PRIVATE);
```

创建共享的首选项与此类似。但还是有两点区别：第一，我们必须为这个首选项设置命名；第二，需要调用不同的方法来获得首选项实例：

```
import android.content.SharedPreferences;
...
SharedPreferences settings =
    getSharedPreferences("MyCustomSharedPreferences", 0);
```

共享首选项在应用程序中可以通过其名称被任何 Activity 访问。对于你能够创建的不同共享首选项的数量并没有限制。比如，应用程序可以拥有一个名为 UserNetworkPreferences 的共享首选项，还可以拥有另一个名为 AppDisplayPreferences 的共享首选项。如何组织这些共享首选项完全取决于开发人员，不过你仍然需要将你的首选项名称声明为一个变量（在基类或头部中），这样可以方便你在其他多个 Activity 中使用它。例如：

```
public static final String PREFERENCE_FILENAME = "AppPrefs";
```

10.1.2 搜索和读取首选项

对于首选项的读取是一个很直接的过程，只需要获取你想要读取的 SharedPreferences 实例即可。你可以通过名称查看首选项，获取具有严格类型的首选项并且可以注册以监听首选项的改变。表 10.1 列出了在 SharedPreferences 接口中的一些有用的方法。

表 10.1　　重要的 android.content.SharedPreferences 方法

方　法	说　明
SharedPreferences.contains()	通过名称查看某个首选项是否存在
SharedPreferences.edit()	获取编辑器（Editor）以修改首选项
SharedPreferences.getAll()	获取所有首选项“键/值”的映射
SharedPreferences.getBoolean()	通过名称获取布尔型首选项
SharedPreferences.getFloat()	通过名称获取浮点型首选项
SharedPreferences.getInt()	通过名称获取整型首选项
SharedPreferences.getLong()	通过名称获取长整型首选项
SharedPreferences.getString()	通过名称获取字符串型首选项

10.1.3 新增、修改和删除首选项

要修改首选项，你需要打开首选项的 Editor 进行修改，并且提交数据。表 10.2 列出了在 SharedPreferences.Editor 接口中的一些有用的方法。

Download at http://www.pin5i.com/

表 10.2 重要的 android.content.SharedPreferences.Editor 方法

方 法	说 明
SharedPreferences.Editor.clear()	移除所有的首选项。该操作将首先执行，而无论何时在编辑会话中调用它。之后所有的修改才能进行与提交
SharedPreferences.Editor.remove()	通过名称移除某个特定的首选项。该操作将首先执行，而无论何时在编辑会话中调用它。之后所有的修改才能进行与提交
SharedPreferences.Editor.putBoolean()	通过名称设置布尔型首选项
SharedPreferences.Editor.putFloat()	通过名称设置浮点型首选项
SharedPreferences.Editor.putInt()	通过名称设置整型首选项
SharedPreferences.Editor.putLong()	通过名称设置长整型首选项
SharedPreferences.Editor.putString()	通过名称设置字符串型首选项
SharedPreferences.Editor.commit()	提交当次编辑会话的所有更改

下面的代码片段获取了 Activity 的私有首选项，打开首选项编辑器，加入了一个名为 SomeLong 的长整型首选项，最后保存了更改：

```
import android.content.SharedPreferences;
...
SharedPreferences settingActivity = getPreferences(MODE_PRIVATE);
SharedPreferences.Editor prefEditor = settingsActivity.edit();
prefEditor.putLong("SomeLong", java.lang.Long.MIN_VALUE);
prefEditor.commit();
```

10.1.4 使用 Android 文件系统查找首选项数据

应用程序首选项在内部被存为 XML 文件。你可以通过文件浏览器（File Explorer）使用 DDMS 来访问这些首选项文件。你可以在下面的 Android 文件系统目录中找到这些文件：

/data/data/<package name>/shared_prefs/<preferences filename>.xml:

对于私有首选项，其文件名即是 Activity 的类名，而对于共享首选项，其名称可以由你给定。下面是一个简单的首选项文件的示例，文件包含了所有类型的首选项：

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<map>
    <string name="String_Pref">Test String</string>
    <int name="Int_Pref" value="-1247483648" />
    <float name="Float_Pref" value="-Infinity" />
    <long name="Long_Pref" vlue="92233720368547758O7" />
    <boolean name="Boolean_Pref" value="false" />
</map>
```

理解应用程序首选项文件的格式对于调试应用程序是很有帮助的。你可以使用 Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS），它可以将首选项文件复制到设备中，也可以从设备上将其复制出来。

Note

关于使用 DDMS 和文件浏览器的更多内容，可以参考附录 B “Android DDMS 快速入门”。

Download at http://www.pin5i.com/

10.2 使用文件和目录

记得我们在第 1 章“Android 简介”中提到过，每一个 Android 应用程序都是以其自身的用户运行在底层的 Linux 操作系统中。它拥有自己的私有应用程序目录和文件。在 Android SDK 中，你还可以找到各种类型的 Java 文件功能类（如 `java.io`），它们可以处理不同类型的文件，例如文本文件、二进制文件和 XML 文件。

在第 6 章“管理应用程序资源”中，你还了解到 Android 应用程序可以将静态的原始文件和 XML 文件作为自己的资源。虽然提取文件同访问资源有一点区别，但访问文件的方式都是类似的。

Android 应用程序文件存储在 Android 文件系统的标准目录层级下。你可以使用 DDMS 文件浏览器在应用程序目录结构中导航。

Tip

这一部分中的许多代码示例都来自于 SimplePreferences 和 FileStreamOfConsciousness 应用程序。SimpleFiles 应用程序演示了基本的文件和目录操作，但该工程没有用户界面（而是需要查看 LogCat 的输出结果）。FileStreamOfConsciousness 应用程序演示了如何将一个字符串写入到文件中；它是一个多线程的应用程序。在本书的网站上可以下载这两个应用程序的源代码。

10.2.1 探索 Android 应用程序目录

Android 应用程序数据存储在 Android 文件系统的如下根目录中：

```
/data/data/<package name>/
```

另外，几个子文件夹被创建来存储数据库、首选项和所需的文件。你也可以根据具体需要创建自定义的目录。所有的文件操作均从与应用程序 `Context`（上下文）对象打交道开始。表 10.3 列出了一些应用程序文件管理所需的重要方法。你可以使用所有标准的 `java.io` 包所提供的功能，比如 `FileStream` 对象等。

表 10.3 重要的 `android.content.Context` 文件和目录管理方法

方　法	说　明
`Context.openFileInput()`	打开应用程序文件以供读取。这些文件位于/files 子目录中
`Context.openFileOutput()`	创建或打开应用程序文件以供写入。这些文件位于/files 子目录中
`Context.deleteFile()`	通过名称删除应用程序文件。这些文件必须位于/files 子目录中
`Context.fileList()`	获得所有位于/files 子目录下的文件列表
`Context.getFilesDir()`	获取应用程序/files 子目录对象
`Context.getCacheDir()`	获取应用程序/cache 子目录对象
`Context.getDir()`	根据名称创建或获取一个应用程序子目录

1. 在默认应用程序目录中创建和写入文件

Android 应用程序依赖于一个名为 `openFileOutput()` 的方法来实现临时文件（occasional file）。使用该方法将在应用程序数据目录下的默认位置创建文件：

Download at http://www.pin5i.com/

/data/data/<package name>/files/

例如，下面的代码创建并打开了一个名为 Filename.txt 的文件。我们向其写入一行，并随即关闭：

```
import java.io.FileOutputStream;
...
FileOutputStream fos;
String strFileContents = "Some text to write to the file.";
fos = openFileOutput("Filename.txt", MODE_PRIVATE);
fos.write(strFileContents.getBytes());
fos.close();
```

我们可以使用 MODE_APPEND 打开这个文件并向其添加数据：

```
import java.io.FileOutputStream;
...
FileOutputStream fos;
String strFileContents = "More text to write to the file.";
fos = openFileOutput("Filename.txt", MODE_APPEND);
fos.write(strFileContents.getBytes());
fos.close();
```

我们创建的文件在 Android 文件系统中的路径是：

/data/data/<package name>/files/Filename.txt

2. 在默认应用程序目录中读取文件内容

与上面类似，我们有一种读取默认/files 子目录下文件的简便方法。下面的代码片段打开了一个名为 Filename.txt 的文件，并进行读取。

```
import java.io.FileInputStream;
...
String strFileName = "Filename.txt";
FileInputStream fis = openFileInput(strFileName);
```

3. 逐字节读取原始文件

文件的读写操作是通过标准 Java 方法实现的。以字节方式读取不同类型基本文件的操作由 java.io.InputStream 的子类实现。例如，DataInputStream 用来进行按行读取操作。

下面是读取一个文本文件的简单示例，它一行一行地进行读取，并且将其存储于一个 StringBuffer 中：

```
FileInputStream fis = openFileInput(filename);
StringBuffer sBuffer = new StringBuffer();
DataInputStream dataIO = new DataInputStream(fis);
String strLine = null;

while ((strLine = dataIO.readLine()) != null) {
    sBuffer.append(strLine + "\n");
}

dataIO.close();
fis.close();
```

Download at http://www.pin5i.com/

4. 读取 XML 文件

Android SDK 包含几个用于处理 XML 文件的功能包，包括 SAX、一个 XML 拉解析器（Pull Parser）和有限的 DOM Level 2 Core 支持。表 10.4 列出了 Android 平台下用于 XML 解析的包。

表 10.4 重要的 XML 功能包

包	描　述
android.sax.*	编写标准 SAX 处理器的框架
android.util.Xml.*	XML 功能包，包括 XMLPullParser
Org.xml.sax.*	核心 SAX 功能。项目：www.saxproject.org/
javax.xml.*	SAX 和有限的 DOM Level 2 Core 支持
Org.w3c.dom	DOM Level 2 Core 接口
Org.xmlpull.*	XmlPullParser 和 XMLSerializer 接口。项目：www.xmlpull.org/

Tip

第 6 章中的 ResourceRoundup 工程给出了一个解析静态 XML 文件的示例，它使用一个称为 XmlResourceParser 的 XmlPullParser 将其解析为了应用程序资源并包含进了应用程序中。

10.2.2 使用 Android 文件系统操作其他目录和文件

如果你的文件不多，且它们都存储在/files 子目录下的话，使用 Context.openFileOutput()和 Context.openFileInput()方法就已经很好了，但如果你有较为复杂的文件管理需求的话，就需要设置自己的目录结构。要实现这一目标，你必须使用标准 java.io.File 类方法同 Android 文件系统打交道。

下面的代码获取了一个/files 应用程序子目录下的 File 对象，同时取得了该目录下所有文件名的列表：

```
import java.io.File;
...
File pathForAppFiles = getFilesDir();
String[] fileList = pathForAppFiles.list();
```

下面是一个在文件系统中创建文件的更为一般的方法。该方法适用于 Android 文件系统中任何你拥有访问权限的地方，而非仅限于/files 目录：

```
import java.io.File;
import java.io.FileOutputStream;
...
File fileDir = getFilesDir();
String strNewFileName = "myFile.dat";
String strFileContents = "Some data for our file";
File newFile = new File(fileDir, strNewFileName);
newFile.createNewFile();

FileOutputStream fo =
    new FileOutputStream(newFile.getAbsolutePath());
fo.write(strFileContents.getBytes());
fo.close();
```

Download at http://www.pin5i.com/

你可以使用 File 对象来管理特定目录下的文件，或者创建该目录下的子目录。例如，你可以在“album（专辑）”目录下存储“track（音轨）”文件，或者在该目录下创建文件，而非在默认目录。

Tip

应用程序应该将较大的数据存储在外部设备（使用 SD 卡）中，而不是在有限的内部存储器上。

假设你想要缓存一些数据来提升应用程序的性能并且提高网络访问频率。在这种情况下，你可以创建缓存文件。对于缓存文件，也有一个默认用于存储它们的应用程序目录。在 Android 文件系统中，该目录为：

```
/data/data/<package name>/cache/
```

Warning

应用程序负责管理自己的缓存目录，并且将其保持在一个合理的尺寸内（通常推荐 1MB）。当内部存储空间不足或用户卸载了应用程序后，Android 文件系统将删除缓存文件。

下面的代码获得了一个/cache 应用程序子目录下的 File 对象，它在该目录下创建了一个新文件，向其中写入了一些数据，然后关闭了文件并删除了它：

```
File pathCacheDir = getCacheDir();
String strCacheFileName = "myCacheFile.cache";
String strFileContents = "Some data for our file";

File newCacheFile = new File(pathCacheDir, strCacheFileName);
newCacheFile.createNewFile();

FileOutputStream foCache =
    new FileOutputStream(newCacheFile.getAbsolutePath());
foCache.write(strFileContents.getBytes());
foCache.close();

newCacheFile.delete();
```

10.3 使用 SQLite 数据库存储结构化数据

当你的应用程序需要一个更为健壮的数据存储机制时，Android 文件系统所包含的针对应用程序的关系型数据库可以满足你的要求，它就是 SQLite。SQLite 是一个基于文件的轻量级数据库，这使得它特别适合嵌入式设备。

Tip

这一部分中的许多代码示例都来自于 SimpleDatabase 应用程序。在本书的网站上可以下载 SimpleDatabase 应用程序的源代码。

这些数据库和其中的数据是应用程序所私有的。如果要将其同其他应用程序共享，那么你必须把你的应用程序变为一个内容提供器（本章稍后部分将讲解）以提供这些数据。

Android SDK 包含了若干有用的 SQLite 数据库管理类。它们中的许多都存在于 android.database.sqlite 包中。其中，你可以发现很多功能包类，比如管理数据库的

Download at http://www.pin5i.com/

创建和版本信息、数据库管理以及查询生成辅助类（query builder helper class）等，它可以帮助你生成正确的 SQL 表达式和查询。你可以使用特定的 Cursor 对象来重复查询。另外，你还会发现与 SQLite 相关联的一些特殊情况。

这里我们将焦点集中于在 Android 应用程序中创建数据库。为此，我们使用内建的 SQLite 支持通过程序创建并使用 SQLite 数据库，存储应用程序的信息。不过，如果你的应用程序使用的是另一种类型的数据库，那么你也可以利用更为普遍的数据库类（存在于 android.database 包中）来帮助你使用来自其他内容提供器的数据。

除了通过程序代码来创建和使用 SQLite 数据库外，开发人员还可以直接使用 sqlite3 命令行工具与应用程序数据库进行交互，该命令行工具可以通过 ADB Shell 接口进行访问。对于需要为测试目的而管理数据库状态（和内容）的 QA 人员来讲，它是一个非常有用的调试工具。

Note

要获得有关通过命令行工具设计和应用 SQLite 数据库的更多内容，可以参见附录 E "SQLite 快速入门"。这一附录分为了两个部分：前半部分是对 sqlite3 命令行界面中最常用的功能的简介以及 SQLite 较其他主流 SQL 数据库的局限性；附录的后半部分则提供了一个完整的指南，它将教会你如何从零开始构造 SQLite 数据库并且使用它。如果你初次接触 SQL（或 SQLite）或者其语法对你来讲已经生疏了，那么附录中的这个指南特别适合你。

10.3.1 创建 SQLite 数据库

有几种方法可以为你的应用程序创建 SQLite 数据库。为了演示如何创建并使用一个简单的 SQLite 数据库，我们首先新建一个名为 SimpleDatabase 的 Android 工程。

1. 使用应用程序 Context 创建 SQLite 数据库实例

为你的应用程序创建一个新的 SQLiteDatabase 实例的最简单方法是使用应用程序 Context 的 openOrCreateDatabase() 方法，如下所示：

```
import android.database.sqlite.SQLiteDatabase;
...
SQLiteDatabase mDatabase;
mDatabase = openOrCreateDatabase(
    "my_sqlite_database.db",
    SQLiteDatabase.CREATE_IF_NECESSARY,
    null);
```

2. 在设备文件系统中定位应用程序数据库文件

Android 应用程序将其数据库（SQLite 或其他）存储于特定的应用程序目录下：

/data/data/<application package name>/databases/<databasename>

所以，在这个例子中，数据库的路径为：

/data/data/com.androidbook.SimpleDatabase/databases/my_sqlite_database.db

你可以通过这个路径使用 sqlite3 命令行界面访问数据库。

3. 配置 SQLite 数据库属性

现在你已经拥有了一个有效的 SQLiteDatabase 实例，下面将对它进行一定的配置。重要的数据库配置选项包括版本、本地化（locale）和线程安全锁（thread-safe locking）特性。

Download at http://www.pin5i.com/

```
import java.util.Locale;
...
mDatabase.setLocale(Locale.getDefault());
mDatabase.setLockingEnabled(true);
mDatabase.setVersion(1);
```

4. 创建表和其他 SQLite 模式（Schema）对象

创建表和其他的 SQLite 模式（schema）对象同编写适当的 SQLite 表达式并执行它们一样简单。下面的代码是一个有效的 CREATE TABLE SQL 表达式。表达式创建了一个名为 tbl_authors 的数据库表。它包含三个字段：能够自动增长的唯一 id 号（它作为表的主键）、firstname 和 lastname 文本字段。

```
CREATE TABLE tbl_authors (
id INTEGER PRIMARY KEY AUTOINCREMENT,
firstname TEXT,
lastname TEXT);
```

上面的 CREATE TABLE SQL 表达式可以被封装进一个 static final 的字符串变量（名为 CREATE_AUTHOR_TABLE）中，并且可以使用 execSQL() 方法来执行它：

```
mDatabase.execSQL(CREATE_AUTHOR_TABLE);
```

execSQL() 方法用于是非查询（nonquery）。你可以使用它来执行任何有效的 SQLite SQL 表达式。例如，你可以使用它来创建、修改（update）、删除数据库表、视图（view）和其他常用的 SQL 对象。在这个应用程序中，我们添加了另一个名为 tbl_books 的表。tbl_books 的模式（schema）如下所示：

```
CREATE TABLE tbl_books (
id INTEGER PRIMARY KEY AUTOINCREMENT,
title TEXT,
dateadded DATE,
authorid INTEGER NOT NULL CONSTRAINT authorid REFERENCES tbl_authors(id) ON
DELETE
CASCADE);
```

不幸的是，SQLite 并不使用强制的外键约束（foreign key constraint）。相反，我们必须使用自定义的 SQL 触发器（trigger）来约束它们。所以，我们自己创建触发器，例如下面的这一个触发器强调了“图书”必须拥有一个有效的“作者”：

```
private static final String CREATE_TRIGGER_ADD =
"CREATE TRIGGER fk_insert_book BEFORE INSERT ON tbl_books
FOR EACH ROW
BEGIN
SELECT RAISE(ROLLBACK, 'insert on table \"tbl_books\" violates foreign key
constraint \"fk_authorid\"') WHERE  (SELECT id FROM tbl_authors WHERE id =
NEW.authorid) IS NULL;
END;";
```

之后，我们就可以简单地通过执行 CREATE TRIGGER SQL 表达式来创建这个触发器了：

```
mDatabase.execSQL(CREATE_TRIGGER_ADD);
```

除此，我们还需要引入更多的触发器来帮助我们强调“作者”和“图书”表之间的联

Download at http://www.pin5i.com/

系，一个用在修改 tbl_books 表时，而另一个用在删除 tbl_authors 表的记录时。

10.3.2 创建、修改和删除数据库记录

我们已经建立起了这个数据库，现在需要为其创建数据了。SQLiteDatabase 类包含三个易用的方法来实现数据操作。正如你所预见的，它们分别是 insert()、update()和 delete()。

1. 插入记录

我们使用 insert()方法向表中加入新的数据，并且使用 ContentValues 对象来将我们待插入数据的字段名和值进行配对。例如，下面我们向 tbl_authors 表中加入“J.K Rowling”。

```
import android.content.ContentValues;
...
ContentValues values = new ContentValues();
values.put("firstname", "J.K.");
values.put("lastname", "Rowling");
long newAuthorID = mDatabase.insert("tbl_authors", null, values);
```

insert()方法返回新创建记录的 id。我们使用这一作者 id 来创建该作者所著书的记录。

Tip

还有另外一个名为 insertOrThrow()的方法，它完成与 insert()方法相同的工作，但它会在失败时抛出一个 SQLException（SQL 异常），它对于你查看错误的原因将会非常有帮助。

当你需要在程序中使用数据记录时，可以创建一个简单的类（即 Author 类和 Book 类）来封装这些数据。

2. 更新记录

你可以使用 update()方法修改数据库的记录。update()方法需要 4 个参数：

① 需要修改记录的表；

② 包含待修改字段数据的 ContentValues 对象；

③ 可选的 WHERE 子句，其中的“?”代表 WHERE 子句参数；

④ WHERE 子句参数数组，其中的每一个元素将会替代第三个参数中对应的“?”。

向 WHERE 子句中传入 null 表示修改表中的所有记录。这适用于对数据库记录的全面修改。

大多数时候，我们根据其唯一标识符来修改某个单一的数据记录。下面的函数使用两个参数：修改后的图书标题和 bookId。我们使用 id 在 tbl_books 表中找到对应的记录，然后修改其图书的标题。和之前相同，我们仍然使用 ContentValues 对象来绑定字段名称和数据值：

```
public void updateBookTile(Integer bookId, String newtitle) {
    ContentValues values = new ContentValues();
    values.put("title", newtitle);
    mDatabase.update("tbl_books",
        values, "id=?", new String[] { bookId.toString() });
}
```

由于我们并不需要修改其他的字段，所以没有必要在 ContentValues 对象中包含它们。我们只包含了标题字段，因为它是唯一需要修改的。

Download at http://www.pin5i.com/

3. 删除记录

你可以使用 remove()方法将数据记录从数据库中移除。remove()方法需要 3 个参数：

① 需要删除记录的表；

② 可选的 WHERE 子句，其中的“?”代表 WHERE 子句参数；

③ WHERE 子句参数数组，其中的每一个元素将会替代第二个参数中对应的“?”。

向 WHERE 子句中传入 null 表示删除表中的所有记录。例如，下面的函数调用将删除 tbl_authors 表中的所有记录：

```
mDatabase.delete("tbl_authors", null, null);
```

大多数时候，我们根据其唯一标识符来删除某个单一的数据记录。下面的函数使用了一个参数，即 bookId，并且使用它来删除 tbl_books 表中唯一的 id（主键）和与之相对应的数据记录。

```
pubilc void deleteBook(Integer bookId) {
    mDatabase.delete("tbl_books", "id=?",
        new String[] { bookId.toString() });
}
```

并不是一定需要主键（id）来删除记录。WHERE 子句可以由你自己编写。例如，下面的函数将根据作者的唯一 id 来删除 tbl_books 表中其所有的著作。

```
public void deleteBookByAuthor(Integer authorID) {
    int numBooksDeleted = mDatabase.delete("tbl_books", "authorid=?",
        new String[] { authorID.toString() });
}
```

4. 使用事务（Transaction）

经常性地，你可能需要多个数据库操作要么一起执行，要么都不执行。这时，你可以使用 SQL 事务（Transaction）将这些操作组合起来；一旦某个操作失败，你可以处理错误，并且恢复或回滚所有的操作。如果所有的操作都成功了，那么你可以提交它们。下面是一个事务的基本结构：

```
mDatabase.beginTransaction();
try {
    // Insert some records, updated others, delete a few
    // Do whatever you need to do as a unit, then commit it

    mDatabase.setTransactionSuccessful();
} catch (Exception e) {
    // Transaction failed. Failed! Do something here.
    // It's up to you.
} finally {
    mDatabase.endTransaction();
}
```

现在我们稍微详细地讨论事务。一项事务总是从调用 beginTransaction()方法开始的，并且包含在 try/catch 语句块中。如果操作成功，你可以通过调用 setTransaction Successful()方法来提交更改。而如果你没有调用这一方法，那么所有的操作都将会回滚，而不会被提交。最后，你需要通过调用 endTransaction()方法来终止事务，就是这么简单。

Download at http://www.pin5i.com/

在某些情况下，你可能需要从一个异常（exception）中恢复，并且继续事务的处理。例如，如果你在访问一个只读数据库时出现了异常，那么你可以打开该数据库，并重试你的操作。

最后，需要注意的是事务是可以嵌套的，外部事务可以提交或回滚内部事务。

10.3.3 在 SQLite 数据库中执行查询

数据库可以提供任意种可能的数据存储方式，但能够让你如愿地获取所需的数据才是其最强大之处。这基本上与设计适当的数据库模式（schema）息息相关，也得益于具有一定技巧的 SQL 查询，当然大多数的查询都是通过 SELECT 表达式实现的。

Android 提供了许多查询应用程序数据库的方法。你可以运行原始的 SQL 查询表达式（字符串），或者使用 SQL 表达式生成辅助类来生成适当的查询表达式，也可以将后端（backend）的数据库直接同某些特定的用户界面控件相绑定，例如容器视图。

1. 使用游标（Cursor）

当从一个 SQL 查询中返回结果时，通常使用一个 Cursor（android.database.Cursor）来访问这些结果。Cursor 对象很像文件指针，它允许对查询结果进行随机访问。

你可以把查询结果看作一个表，其中每一行对应一个返回的记录。Cursor 对象包含了一些有用的方法来确定查询返回结果的数量以及每一条返回记录的列名称（字段）。返回结果中的列由查询本身进行定义，而非由数据库的列来确定。这将会关系到计算列（calculated colums）、列别名和混合列。

Cursor 对象一般只维持短暂的时间。如果你只需要进行一些简单的操作（比如返回记录的条数或者在当你知道只有一条返回记录的时候所进行的操作），那么你可以快速地执行查询并且获得你所需的数据；在使用完后，不要忘记关闭 Cursor，如下所示：

```
//简单的查询: select * from tbl_books
Cursor c = mDatabase.query("tbl_books",null,null,null,null,null,null);
//这里使用 Cursor 进行一些简单的操作……
c.close();
```

（1）在应用程序周期内管理 Cursor

当 Cursor 返回了多个记录，或者你想要进行某些复杂的操作时，你可以考虑将这一操作运行在一个与 UI 线程不同的线程上。你仍然需要管理 Cursor。

Cursor 对象必须在应用程序周期内进行管理。当应用程序暂停或关闭时，Cursor 必须通过调用 deactivate() 方法被激活（deactivated）；而当应用程序重新开始时，Cursor 应该使用 requery() 方法刷新其数据。当不再需要 Cursor 时，必须调用 close() 方法来释放其资源。

作为开发人员，你可以借助应用程序生存周期内的各种回调方法，如 onPause()、onResume() 和 onDestroy() 等，来实现管理 Cursor 的调用。

如果你不想麻烦，就像我们一样，不愿意处理这些生存周期中的事件，那么你也可以把管理 Cursor 对象的责任推给其父 Activity，这是通过调用 Activity 的 startManagingCursor() 方法实现的。Activity 将会处理剩下的问题，比如在需要的时候去激活和重激活 Cursor，以及在 Activity 被销毁时销毁 Cursor。之后，你还可以随时重新接手 Cursor 对象的管理工作，只需要简单地调用 stopManagingCursor() 方法即可。

下面我们进行与上面相同的查询，然后将 Cursor 的管理事务交给其父 Activity：

Download at http://www.pin5i.com/

```
//简单的查询: select * from tbl_books
Cursor c = mDatabase.query("tbl_books",null,null,null,null,null,null);
startManagingCursor(c);
```

需要注意的是，被管理的 Cursor 对象通常都是类的成员变量。

（2）迭代（Iterating）查询结果行返回特定数据

你可以使用 Cursor 来对返回结果进行迭代（iterate），一次针对一行数据，可以使用各种导航方法，例如 moveToFirst()、moveToNext()和 isAfterLast()。

对于返回结果的某个特定行，你可以使用 Cursor 来获取给定列的数据。因为 SQLite 对类型要求并不严格，所以你可以使用 getString()方法始终把字段数据作为字符串来提取，当然，你也可以使用针对特定类型的函数严格地按照类型来获取值，以满足应用程序的安全需求。

例如，下面的方法使用了一个有效的 Cursor，打印了返回结果的数量，并且显示了一些列信息（名称和列数量）。之后，它对查询结果进行迭代，并打印了每一条记录。

```
public void logCursorInfo(Cursor c) {
    Log.i(DEBUG_TAG), "*** Cursor Begin *** " + " Results:" +
        c.getCount() + "Columns: " + c.getColumnCount());

    // 打印列名称
    String rowHeaders = "|| ";
    for (int i = 0; i < c.getColumnCount(); i++) {
        rowHeaders = rowHeaders.concat(c.getColumnName(i) = " || ");
    }

    Log.i(DEBUG_TAG, "COLUMNS " + rowHeaders);

    // 打印所有记录
    c.moveToFirst();
    while (c.isAfterLast() == false) {

        String rowResults = "|| ";
        for (int i = 0; i < c.getColumnCount(); i++) {
            rowResults = rowResults.concat(c.getString(i) + " || ");
        }

        Log.i(DEBUG_TAG,
            "Row " + c.getPosition() + ": " + rowResults);

        c.moveToNext();
    }
    Log.i(DEBUG_TAG, "*** Cursor End ***");
}
```

上面的函数在 LogCat 中的输出应该大致如图 10.1 所示。

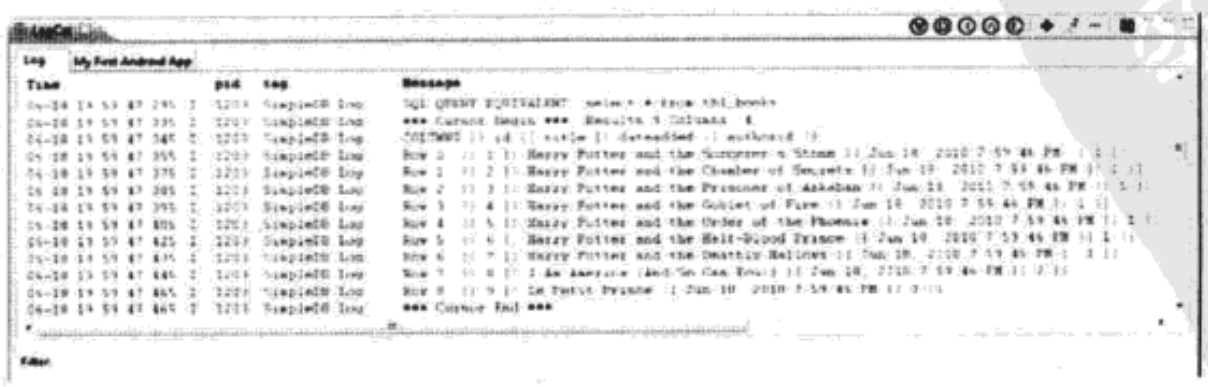


图 10.1 logCursorInfo()示例方法的输出

Download at http://www.pin5i.com/

2. 执行简单的查询

学习数据库查询的第一步应该是掌握 SQLiteDatabase 类中的 query() 方法。该方法查询数据库并以 Cursor 对象的形式返回结果。

一般使用的 query() 方法需要下面的参数。

- [String]：执行查询的表名称。
- [String Array]：返回的列名称的列表（可以都使用 null）。
- [String] WHERE 子句：可以都使用 null；可以含有可选的参数，如“?”。
- [String Array]：可选的参数值，以替代之前参数中的“?”。
- [String] GROUP BY 子句：null 表示无分组。
- [String] HAVING 子句：null，除非 GOURP BY 子句需要它。
- [String] ORDER BY 子句：如果为 null，则使用默认顺序。
- [String] LIMIT 子句：如果为 null，则无限制。

在本章前面的部分，我们只使用了一个参数（即表名称）来调用 query() 方法。

```
Cursor c = mDatabase.query("tbl_books",null,null,null,null,null,null);
```

这等同于下面的 SQL 查询：

```
SELECT * FROM tbl_books;
```

Tip

充当子句的参数（WHERE、GROUP BY、HAVING、ORDER BY、LIMIT）均是字符串，但你并不需要在其中包含关键字，如 WHERE。相反，你只需要使用关键字后的部分子句即可。

下面，我们向查询中加入 WHERE 子句，这样可以实现一次只提取一个记录。

```
Cursor c = mDatabase.query("tbl_books,null,
"id=?", new String[]{"9"}, null, null, null);
```

这等价于下面的 SQL 查询：

```
SELECT * tbl_books WHERE id=9;
```

对于微型的数据库而言，一次选择所有数据是可以的，但它并不高效。你应该始终精简你的 SQL 查询以实现仅返回所需数据，而没有其他无关的信息。要尽量使用强大的 SQL 语句来减轻查询的负担，而不是自己通过程序再对结果进行处理。例如，如果你只需要图书表中书目的标题，那么你可以使用如下的 call() 方法调用：

```
String asColumnsToReturn[] = { "title", "id" };
String strSortOrder = "title ASC";
Cursor c = mDatabase.query("tbl_books", asColumnsToReturn,
    null, null, null, null, strSortOrder);
```

这同下面的 SQL 语句是一样的：

```
SELECT title, id FROM tbl_books ORDER BY title ASC;
```

3. 使用 SQLiteQueryBuilder 执行复杂查询

随着你的查询变得越来越复杂，尤其是当牵涉多个数据库表时，你可能需要使用

Download at http://www.pin5i.com/

SQLiteQueryBuilder 类，它可以帮助你生成复杂的查询。

当不只一个表牵涉某个查询中时，你要确保使用完整的列名称。例如，tbl_books 表中的标题列的完整名称应该为 tbl_books.title。下面我们使用 SQLiteQueryBuilder 来生成和执行一个简单的带有 INNER JOIN 的查询，它关联了两个表，从而获得所有书目及其作者的列表：

```
import android.database.sqlite.SQLiteQueryBuilder;
...
SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();

queryBuilder.setTables("tbl_books, tbl_authors");
queryBuilder.appendWhere("tbl_books.authorid=tbl_authors.id");

String asColumnsToReturn[] = {
    "tbl_books.title",
    "tbl_books.id",
    "tbl_authors.firstname",
    "tbl_authors.lastname",
    "tbl_books.authorid" };
String strSortOrder = "title ASC";

Cursor c = queryBuilder.query(mDatabase, asColumnsToReturn,
    null, null, null, null, strSortOrder);
```

首先，我们实例化一个新的 SQLiteQueryBuilder 对象。然后，我们可以将涉及的表设置到 JOIN 中，并且使用 WHERE 从句限定 JOIN 的发生。接下来，调用 SQLiteQuery Builder 的 query()方法，它与我们之前使用过的 query()方法类似，唯一的不同之处在于我们需要传入一个 SQLiteQueryBuilder 实例而不是表名称。上面使用 SQLiteQueryBuilder 构建的查询等价于下面的 SQL 表达式：

```
SELEST tbl_books.title,
tbl_books.id,
tbl_authors.firstname,
tbl_authors.lastname,
tbl_books.authorid
FROM tbl_books
INNER JOIN tbl_authors on tbl_books.authorid=tbl_authors.id
ORDER BY title ASC;
```

4. 不使用 Builder 和列映射执行原始查询

上面讲解的 Android 查询辅助功能在需要执行非典型或非常复杂的查询时可能会显得比较繁杂。在这种情况下，你可能需要使用 rawQuery()方法。rawQuery()方法只需要一个 SQL 表达式（如果引入了“?”则可能带有可选参数）并且以 Cursor 的形式返回结果。如果你已经能够明确你的 SQL 查询，并且不想关心那些 SQL 查询生成功能的输入、输出，那么你可以使用这种方法。

例如，假设我们有一个带有 UNION 的查询。这种查询当然也可以借助 QueryBuilder 来生成，但当我们需要考虑列别名等事情时，就会显得比较麻烦。

假如我们想要执行下面的 SQL UNION 查询，它能够返回所有书名和作者名中含有“ow”（比如“Hallows”、“Rowling”）的书目及对应作者，如下所示：

Download at http://www.pin5i.com/

```
SELECT title AS Name,
'tbl_books' As OriginalTable
FROM tbl_books
WHERE Name LIKE '%ow%'
UNION
SELECT (firstname||' '|| lastname) AS Name,
'tbl_authors' AS OriginalTable
FROM tbl_authors
WHERE Name LIKE '%ow%'
ORDER BY Name ASC;
```

我们可以很简单地生成一个类似上面查询的字符串，然后使用 rawQuery() 方法执行它。

```
String sqlUnionExample = "SELECT title AS Name, 'tbl_books' AS
    OringinalTable from tbl_books WHERE Name LIKE ? UNION SELECT
    (firstname||' '||lastname) AS Name, 'tbl_authors' AS OriginalTable
    from tbl_authors WHERE Name LIKE ? ORDER BY Name ASC;";
Cursor c = mDatabase.rawQuery(sqlUnionExample,
    new String[]{ "%ow%", "%ow%"});
```

我们将子串（"ow"）放入了可选参数中，这样就可以使用相同的代码来进行其他子串的查询。

10.3.4 关闭和删除 SQLite 数据库

当你已经不再使用数据库时，就应该关闭它。当然，有些时候你也可能想要修改或删除表，或者是删除数据库。

1. 删除表和其他 SQLite 对象

删除表和其他 SQLite 对象同创建它们是完全一样的：构造适当的 SQLite 表达式并且执行它们。例如，要删除表和触发器，我们可以执行三个 SQL 表达式：

```
mDatabase.execSQL("DROP TABLE tbl_books;");
mDatabase.execSQL("DROP TABLE tbl_authors;");
mDatabase.execSQL("DROP TRIGGER IF EXISTS fk_insert_book;");
```

2. 关闭 SQLite 数据库

当你不再使用数据库时，可以使用 SQLiteDatabase 实例的 close() 方法将其关闭，如下所示：

```
mDatabase.close();
```

3. 使用应用程序 Context 删除 SQLite 数据库实例

删除 SQLiteDatabase 最简单的方法是使用应用程序 Context 的 deleteDatabase() 方法。数据库的删除是根据其名称进行的，而且这种删除将是永久性的，所有的数据和模式都会丢失。

```
deleteDatabase("my_sqlite_database.db");
```

10.3.5 设计持久数据库

一般而言，应用程序创建一个数据库是在其剩下的生存周期中使用的，也就是说，使用

Download at http://www.pin5i.com/

到应用程序被卸载时为止。到目前，我们已经讨论了创建、使用和删除数据库的基本知识。

实际上，移动应用程序并非在运行中创建一个数据库，然后使用它，最后将其删除。相反，它们在需要使用时第一次创建一个数据库，并且使用它。Android SDK 提供了一个名为 SQLiteOpenHelper 的辅助类来帮助你管理应用程序的数据库。

要使用 SQLiteOpenHelper 为 Android 应用程序创建一个数据库，你需要继承该类，并且作为应用程序的一个成员变量来实例化它。为了演示如何实现这一过程，我们创建一个新的 Android 工程，名为 PetTracker。

Tip

这一部分中的许多代码示例都来自于 PerTracker 应用程序。在本书的网站上可以下载 PerTracker 应用程序的源代码。我们将基于这个工程来展开本章后面部分的讲解。

1. 跟踪数据库字段名

现在，我们可以开始学习如何使用程序来组织数据库字段，以避免 SQL 查询中的错误了。实现这一目标的简单方式之一是创建一个类（比如 PetDatabase）来封装数据库的模式，如下所示：

```
import android.provider.BaseColumns;

public final class PetDatabase {

    private PetDatabase() {}

    public static final class Pets implements BaseColumns {
        private Pets() {}
        public static final String PETS_TABLE_NAME="table_pets";
        public static final String PET_NAME="pet_name";
        public static final String PET_TYPE_ID="pet_type_id";
        public static final String DEFAULT_SORT_ORDER="pet_name ASC";
    }

    public static final class PetType implements BaseColumns {
        private PetType () {}
        public static final String PETTYPE_TABLE_NAME="table_pettypes";
        public static final String PET_TYPE_NAME="pet_type";
        public static final String DEFAULT_SORT_ORDER="pet_type ASC";
    }
}
```

通过实现 BaseColumns 接口，今后我们可以使用它来设置数据库友好应用程序界面 Widget 的基础，这通常需要一个特殊的列，它名为“_id”。我们把这一列作为数据库表的主键来使用。

2. 扩展 SQLiteOpenHelper 类

要继承 SQLiteOpenHelper 类，我们必须实现几个重要的方法，它们可以帮助管理数据库的版本。这些需要覆写的方法是 onCreate()、onUpgrade()和 onOpen()。我们使用刚刚定义的 PetDatabase 类来生成适当的 SQL 表达式，如下所示：

```
import android.content.Context;
improt android.database.sqlite.SQLiteDatabase;
import android,database.sqlite.SQLiteOpenHelper;
```

Download at http://www.pin5i.com/

```
improt com.androidbook.PetTracker.PetDatabase.PetType;
import com.androidbook.PetTracker.PetDatabase.Pets;

class PetTrackerDatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "pet_tracker.db";
    private static final int DATABASE_VERSION = 1;

    PetTrackerDatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE " +PetType.PETTYPE_TABLE_NAME+" ("
            + PetType._ID + " INTEGER PRIMARY KEY AUTOINCREMENT ,"
            + PetType.PET_TYPE_NAME + "TEXT"
            + ");");
        db.execSQL("CREATE TABLE " + Pets.PETS_TABLE_NAME + " ("
            + Pets._ID + " INTEGER PRIMARY KEY AUTOINCREMENT ,"
            + Pets.PET_NAME + " TEXT,"
            + Pets.PET_TYPE_ID + " INTEGER" // FK to pet type table
            + ");");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion,
        int newVersion) {
        // 实现如何在升级模式版本时移动应用程序数据
        // 根据所需，移动或删除数据，这由你决定
    }

    @Override
    public void onOpen(SQLiteDatabase db) {
        super.onOpen(db);
    }
}
```

接下来我们就可以为数据库创建一个成员变量了，如下所示：

```
PetTrackerDatabaseHelper mDatabase = new
PetTrackerDatabaseHelper(this.getApplicationContext());
```

现在，无论应用程序何时需要同数据库进行交互，我们都可以请求得到一个有效的数据库对象。我们可以获得一个只读数据库，或者是可读写数据库，还可以关闭它。例如，在下面的代码中我们获得了一个可以进行写入的数据库：

```
SQLiteDatabase db = mDatabase.getWritableDatabase();
```

10.3.6 向应用程序界面绑定数据

在使用应用程序数据库的许多情况下，你都需要将用户界面同数据库中的数据联系起来。比如使用从数据库表中获得的值来填充下拉列表、表单，或者只显示特定的查询结果。有各

Download at http://www.pin5i.com/

种不同的方法可以将数据库数据同用户界面进行绑定。作为开发人员，你可以决定使用由特定用户界面控件提供的内建数据绑定功能，或者一步步地自己构建用户界面。

1. 像使用其他数据一样使用数据库数据

如果你仔细查看本书 CD 提供的“PetTracker”应用程序，你将会发现其功能并不包含什么神奇的数据绑定特性，应用程序只不过非常清晰地使用了数据库，并将其作为用户界面的一部分而已。

具体来说，数据库在以下情况下将会使用。

① 当你填写“Pet Type（宠物类型）”字段，AutoComplete 功能将查看 table_pettypes 表中的宠物类型列表，如图 10.2（左）所示。

图 10.2 PetTracker 应用程序：主界面（左、中）和宠物列表界面（右）

② 当你使用“Pet Entry Form（宠物登记表）”（如图 10.2（中）所示）保存新的记录时。

③ 当你显示“Pet List（宠物列表）”页面时，我们将查询所有的宠物，并且使用一个 Cursor 来动态地构建一个 TableLayout，如图 10.2（右）所示。

这对于小规模的数据是可行的，但是，它确实具有各种各样的缺陷。例如，所有工具都在主线程中完成，这样，你添加的记录越多，你应用程序的反应速度就会越慢。第二，在将数据库结果映射到单个用户界面组件时，引入了相当多的自定义代码。假如你决定在不同的控件中显示数据，那么你将重新做很多工作。第三，我们可能需要时常重新查询数据库以刷新结果，或者是出于其他的目的。

> **Note**
>
> 这里我们的确是以数据结构和计算机术语来命名我们的宠物的，这确实比较“变态”。比如说，“Null”是一只黑色的疯狂小兔子。Shane 总是指着它叫“Null Pointer（空指针）”。

2. 使用数据适配器（Data Adapter）向控件绑定数据

一般而言，你应该比较喜欢将数据绑定到用户界面控件上，然后让它们负责数据的显示。例如，我们可以使用一个比较“高档”的 ListView 来显示宠物列表，而无需非得“从零开始”打造 TableLayout。我们可以使用 Cursor，并且手工地生成 ListView 的子项目，

Download at http://www.pin5i.com/

甚至更好的情况是，我们可以简单地创建一个数据适配器（data adapter）将 Cursor 的结果映射到 ListView 中的每一个子 TextView 上。

在本书网站上包含的一个名为 PetTracker2 的工程实现了上面的过程。它在行为上同 PetTracker 示例程序很像，只不过它针对 ListView 使用了 SimpleCursorAdapter，并且通过一个 ArrayAdapter 来处理 AutoCompleteTextView 特性。

Tip

本书网站上还会提供对 PetTracker 应用程序升级版（比如 PetTracker2、PetTracker3 等）源代码的下载。

（1）使用 SimpleCursorAdapter 绑定数据

现在我们来看看如何创建一个数据适配器（data adapter）来模拟“Pet Listing”界面，其中列出了每一个宠物的名字和种类。同时，我们还将继续保持列表项的删除功能。

不知你是否记得第 8 章“使用布局设计用户界面”中讲过的，ListView 容器 Widget 可以包含诸如 TextView 这样的子对象。在这种情况下，我们想要显示每一个宠物的名字和种类。所以，我们创建一个名为 pet_item.xml 的布局文件，其为 ListView 列表项的模板：

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/RelativeLayoutHeader"
    android:layout_height="wrap_content"
    android:layout_width="fill_parent">
    <TextView
        android:id="@+id/TextView_PetName"
        android:layout_width="wrap_content"
        android:layout_height="?android:attr/listPreferredItemHeight"
        android:layout_alignParentLeft="true" />
    <TextView
        android:id="@+id/TextView_PetType"
        android:layout_width="wrap_content"
        android:layout_height="?android:attr/listPreferredItemHeight"
        android:layout_alignParentRight="true" />
</RelativeLayout>
```

同时，在 Pet List 的主布局文件中，我们将 ListView 放置在屏幕的适当位置。布局文件的 ListView 部分大致应该如下所示：

```
<ListView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/petList" android:divider="#000" />
```

现在我们通过程序来填充 ListView，必须采取以下步骤。

① 执行查询并且返回一个有效的 Cursor（一个成员变量）。

② 创建一个数据适配器，它将 Cursor 的列映射到 pet_item.xml 布局模板中的 TextView 控件中。

③ 将适配器与 ListView 关联起来。

在下面的代码中，我们执行了上面的过程：

Download at http://www.pin5i.com/

```
SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();
queryBuilder.setTables(Pets.PETS_TABLE_NAME +", "+
    PetType.PETTYPE_TABLE_NAME);

queryBuilder.appendWhere(Pets.PETS_TABLE_NAME + "." +
    Pets.PET_TYPE_ID + "=" + PetType.PETTYPE_TABLE_NAME + "." +
    PetType._ID);

String asColumnsToReturn[] = { Pets.PETS_TABLE_NAME + "." +
    Pets.PET_NAME, Pets.PETS_TABLE_NAME +
    "." + Pets._ID, PetType.PETTYPE_TABLE_NAME + "." +
    PetType.PET_TYPE_NAME };

mCursor = queryBuilder.query(mDB, asColumnsToReturn, null, null,
    null, null, Pets.DEFAULT_SORT_ORDER);

startManagingCursor(mCursor);

ListAdapter adapter = new SimpleCursorAdapter(this,
    R.layout.pet_item, mCursor,
    new String[]{Pets.PET_NAME, PetType.PET_TYPE_NAME},
    new int[]{R.id.TextView_PetName, R.id.TextView_PetType});

ListView av = (ListView)findViewById(R.id.petList);
av.setAdapter(adapter);
```

注意到_id 列连同名字和类型列都出现在了查询中。这是让适配器与 ListView 正常工作所必需的。

使用 ListView（如图 10.3（左）所示）而非自定义的用户界面允许我们方便地使用 ListView 控件的内建特性，比如当列表很长时可以进行滚动，在需要时可以提供上下文菜单等。_id 列用来作为每一个 ListView 子节点的唯一标识符。如果我们选择列表中的某一项，那么就会使用这一标识符来执行某些操作，比如将其删除。

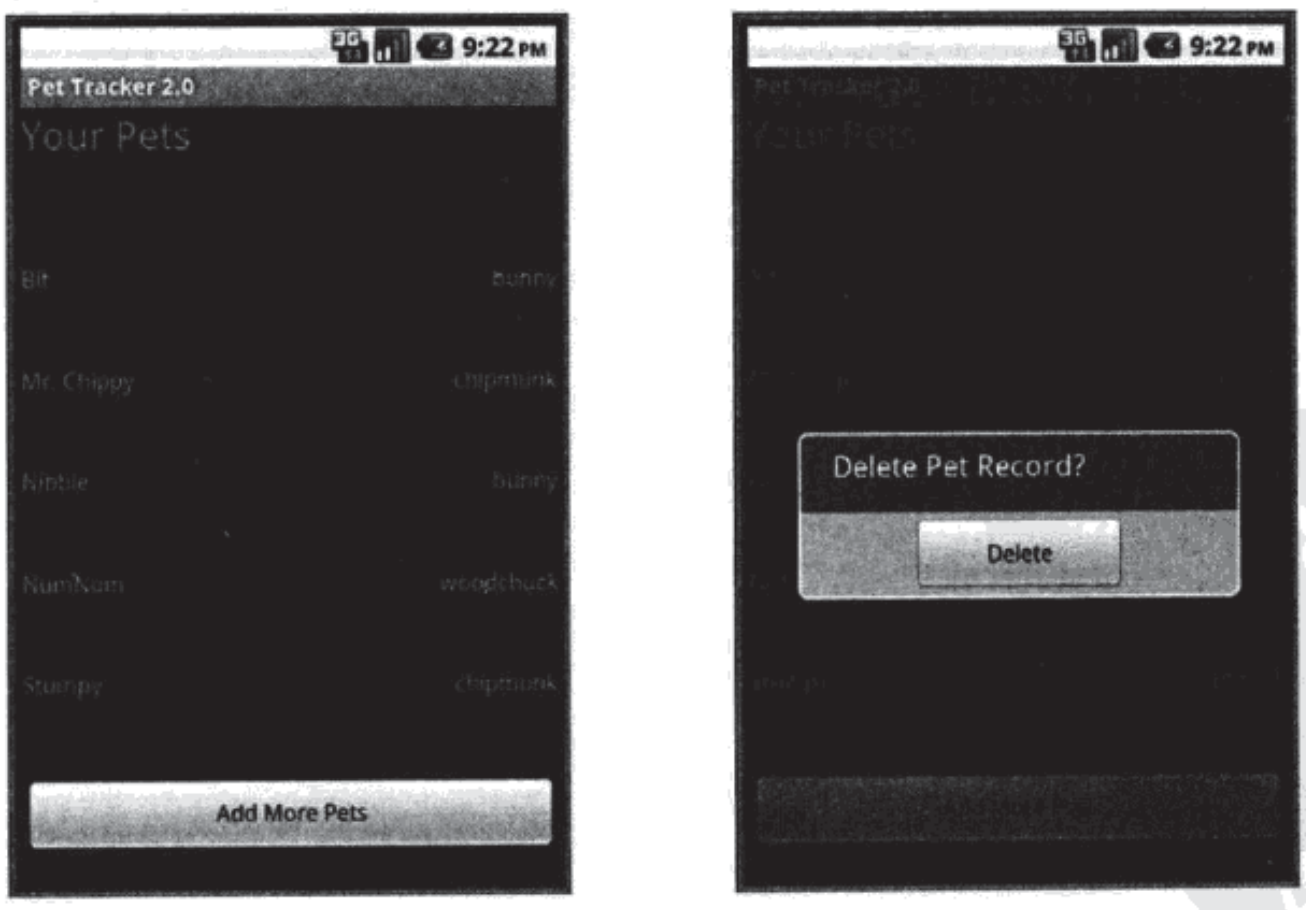


图 10.3 PetTracker2 应用程序：由 ListView（左）实现的带有“删除”功能（右）的 Pet List 界面

现在，我们通过监听 onItemClick()事件来实现“删除”功能，并且提供一个“删除确认”的对话框（如图 10.3（右）所示）。

Download at http://www.pin5i.com/

```
av.setOnItemClickListener (new AdapterView.OnItemClickListener() {
    public void onItemClick ( AdapterView<?> parent, View view,
        int position, long id) {

        final long deletePetId = id;

        new AlertDialog.Builder(SuperPetList.this).setMessage(
            "Delete Pet Record?").setPositiveButton(
            "Delete", new DialogInterface.OnClickListener() {

                @Override
                public void onClick(DialogInterface dialog,int which) {
                    deletePet(deletePetId);
                    mCursor.requery();
        }}).show();
    }
});
```

在图 10.3 中你可以看到上面代码的运行效果。

注意在 PetTracker2 示例应用程序中，我们也使用了一个 ArrayAdapter 来将 pet_types 表中的数据与 Pet Entry 界面中的 AutoCompleteTextView 相绑定。虽然我们的下一个示例就将讲解如何采用更可取的方式实现这一功能，但是我们仍然将这段代码放在 PetTracker 示例中，以向你说明你始终可以对 Cursor 提供的数据进行处理。在这个例子中，我们为 AutoText 选项手工地创建了一个 String 数组，使用称为 android.R.layout.simple_dropdown_item_1line 的内建 Android 布局资源来定义 AutoText 列表中每一项的外观。你可以在 Android SDK 的资源子目录中找到这个内建的布局资源。

（2）在数据库中存储非基本类型（如图像）

由于 SQLite 是一个单独的文件，所以尝试在数据库中存储二进制数据是几乎没有意义的。相反，可以将地理位置数据存储为数据库中的一个文件路径或者 URI，并且在使用时通过它们进行访问。我们将在下一章中通过实例来演示如何在数据库中存储图像 URI。

小　结

在 Android 平台下有各种不同的方法可以实现应用程序数据的存储和管理。方法的选择依赖于需要进行存储的数据类型。使用这些技巧可以让你更好地利用 Android 提供的强大特性。

应用程序可以使用下列数据存储机制之一：

① 轻量的应用程序首选项（Activity 级别和应用程序范围）；

② 带有 XML 文件格式的 Android 文件系统的文件和目录支持；

③ 能够进行结构化存储的应用程序特有 SQLite 数据库。

在本章中，你学习了如何在 Android 应用程序中设计持久型数据访问机制，理解了如何将来自不同数据源的数据同用户界面控件相绑定，例如同 ListView 对象绑定。

参考资料和更多信息

SQLite 网站. www.sqlite.org/index.html.

SQLzoo.net. http://sqlzoo.net/.

Download at http://www.pin5i.com/

第 11 章

使用内容提供器在应用程序间共享数据

在 Android 中，应用程序可以通过内容提供器（Content Provider）来访问其他应用程序的数据，自己也可以成为内容提供器以向其他应用程序提供内部数据。

首先，我们将看看在 Android 平台上都有哪些可用的内容提供器，以及如何使用它们。然后，你将看到如何使用内容提供器来增强我们在上一章中使用的示例应用程序。最后，我们将讲解如何让应用程序成为一个内容提供器以便共享信息，例如，与 LiveFolders 共享信息。

11.1 探索 Android 内容提供器

Android 设备移植了一些内建的应用程序，其中的许多均作为内容提供器向其他应用程序共享其数据。你的应用程序可以从各种渠道访问这些内容提供器的数据。Android 中包含的内容提供器可以在 android.provider 包中找到。表 11.1 列出了其中比较有用的一些内容提供器。

表 11.1 有用的内容提供器

提 供 器	说 明
MediaStore	手机上的视听资料和外部存储
CallLog	拨打和接听的通话
Browser	浏览历史和书签
ContactsContract	手机的联系人数据库或者电话簿
Settings	系统范围的设备设置和首选项
UserDictionary	用户定义的词典，可以用于文本输入的预测

现在我们来详细地讲解每一种内容提供器。

Tip

这一部分中的许多代码示例都来自于 SimpleContentProvider 应用程序。在本书的网站上可以下载 SimpleContentProvider 应用程序的源代码。

11.1.1 使用 MediaStore 内容提供器

你可以使用 MediaStore 内容提供器来访问手机和外部存储中的媒体。这些媒体的主要类型包括音频、图像和视频。这些不同类型的媒体可以分别使用 android.provider.

Download at http://www.pin5i.com/

MediaStore 下的内容提供器类进行访问。

大多数的 MediaStore 类允许同数据进行完全的交互。你既可以从设备中提取和删除媒体文件，也可以向其中增添媒体文件。另外还有一系列的辅助类，它们定义了若干最常用的数据列供应用程序请求。

表 11.2 列出了 android.provider.MediaStore 中最常使用的一些类。

表 11.2 常用的 MediaStore 类

类	说　明
Video.Media	管理设备上的视频文件
Images.Media	管理设备上的图像文件
Images.ThumbNails	管理图像中的图标
Audio.Media	管理设备上的音频文件
Audio.Albums	管理由专辑（album）组织起来的音频文件
Audio.Artists	通过所属艺术家（artist）来管理音频文件
Audio.Genres	通过所属流派（genre）来管理音频文件
Audio.Playlists	通过所属的播放列表来管理音频文件

下面的代码演示了如何从一个内容提供器那里请求所需的数据。示例通过 MediaStore 查询了手机 SD 卡上存储的所有音频文件的标题和时长（duration）。执行代码之前需要向模拟器的虚拟 SD 卡中添加一些音频文件。

```
String[] requestedColumns = {
    MediaStore.Audio.Media.TITLE,
    MediaStore.Audio.Media.DURATION
};

Cursor cur = managedQuery(
    MediaStore.Audio.Media.EXTERNAL_CONTENT_URI,
    requestedColumns, null, null, null);

Log.d(debugTag, "Audio files: " + cur.getCount());
Log.d(debugTag, "Columns: " + cur.getColumnCount());

String[] columns = cur.getColumnNames();

int name = cur.getColumnIndex(MediaStore.Audio.Media.TITLE);
int size = cur.getColumnIndex(MediaStore.Audio.Media.DURATION);

cur.moveToFirst();
while (!cur.isAfterLast()) {
    Log.d(debugTag, "Title" + cur.getString(name));
    Log.d(debugTag, "Length: " +
        cur.getInt(size) / 1000 + " seconds");
    cur.moveToNext();
}
```

MediaStore.Audio.Media 类对每一个内容提供器给出的数据字段（即列）提供了预定义的字符串。你可以通过定义所需数据列名称的字符串数据来限制查询的音频文件数据字段。在上面的例子中，我们将结果限制为音轨（track）标题和每个音频文件的时长。

Download at http://www.pin5i.com/

之后我们使用了 managedQuery()方法调用。其中，第一个参数是预定义的待查询内容提供器（大多数情况下，主要的外部存储即为 SD 卡）的 URI。第二个参数是返回列的列表（音频文件标题和时长）。第三、四个参数为选择过滤参数。第五个参数为结果提供一个排序方法。我们将后三个参数置为 null，因为我们要查询的是当前位置下所有的音频文件。通过使用 managedQuery()方法，我们获得了一个 Cursor 作为返回结果。之后，就可以通过查看这一 Cursor 来获得结果了。

11.1.2 使用 CallLog 内容提供器

Android 提供了一个内容提供器来访问手机的通话记录，这是通过 android.provider.CallLog 类实现的。CallLog 初看起来似乎并不能为开发人员带来什么，但仔细思考会发现它有许多妙用。你可以使用 CallLog 来过滤最近拨出、接听和未接的电话。每一次通话的日期和时长都可以记录下来，并返回给“联系簿”应用程序以供用户进行通话人身份识别。

CallLog 是一个应用于客户关系管理（Customer Relationship Management，CRM）软件中的重要内容提供器。用户甚至可以使用特殊的标签来标示“联系簿”应用程序中的号码。

为了演示 CallLog 内容提供器是如何工作的，我们假设一种情况，即需要生成一份同标记了自定义的 HourlyClient123 的号码之间进行通话的记录报表。Android 允许针对某一号码使用自定义标签，在本例中使用了这一点：

```
String[] requestedColumns = {
    CallLog.Calls.CACHED_NUMBER_LABEL,
    CallLog.Calls.DURATION
};

Cursor calls = managedQuery(
    CallLog.Calls.CONTENT_URI, requestedColumns,
    CallLog.Calls.CACHED_NUMBER_LABEL
    + " = ?", new String[] { "HourlyClient123" } , null);

Log.d(DEBUG_TAG, "Call count: " + calls.getCount());

int durIdx = calls.getColumnIndex(CallLog.Calls.DURATION);
int totalDuration = 0;

calls.moveToFirst();
while (!calls.isAfterLast()) {
    Log.d(DEBUG_TAG, "Duration: " + calls.getInt(durIdx));
    totalDuration += calls.getInt(durIdx);
    calls.moveToNext();
}

Log.d(DEBUG_TAG, "HourlyClient123 Total Call Duration: " + totalDuration);
```

这段代码同之前 MediaStore 音频文件代码相似。我们再一次从列出所请求的数据列开始：通话标签和通话时长。但这一次有所不同的是，我们并不想获得记录中的所有通话数据，而只针对带有 HourlyClient123 标签的那些通话提取信息。要将查询结果过滤到只含有特定标签，就需要指定 managedQuery()调用中的第三个和第四个参数了。它们结合起来等价于数据库中的 WHERE 子句。第三个参数通过带有可选参数（用“?”表示）的列名称指定 WHERE 子句的格式。第四个参数为一个 String 数组，其中按顺序定义了替代前面可选

Download at http://www.pin5i.com/

参数（“?”）的字符串，这同之前使用简单 SQLite 数据库查询时是一样的。

同之前一样，Activity 管理 Cursor 对象的生存周期。我们使用相同的方法来迭代 Cursor 的每条记录，并且累加总的通话时长。

对于访问 CallLog 内容提供器提供的数据，你的应用程序需要一定的特殊权限。你可以通过“Eclipse 向导（Wizard）”来声明 uses-permission 标记，也可以在 AndroidManifest.xml 文件中加入如下代码：

```
<uses-permission
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:name="android.permission.READ_CONTACTS">
</uses-permission>
```

比较令人疑惑的是，并不存在 CallLog 这个许可权限。相反，访问 CallLog 的应用程序所使用的都是 READ_CONTACTS 权限。虽然这些值存储于内容提供器中，但数据却与“Contacts”提供器提供的相似。

Tip

在 android.Manifest.permission 类中可以找到所有可用的许可权限。

11.1.3 使用浏览器（Browser）内容提供器

另一个非常有用的内建内容提供器是 Browser。Browser 内容提供器提供用户浏览站点的历史记录和 Web 站点的书签信息。这一内容提供器可以通过 android.provider.Browser 类来进行访问。如同 CallLog 类，由 Browser 内容提供器提供的信息可以生成一些数据统计，供某些跨应用程序的功能使用。你可以使用 Browser 内容提供器将你应用程序所支持 Web 的站点添加到书签中。

在这个例子里，我们查询 Browser 内容提供器以找到添加了书签的最常访问的五个站点。

```
String[] requestedColumns = {
    Browser.BookmarkColumns.TITLE,
    Browser.BookmarkColumns.VISITS,
    Browser.BookmarkColumns.BOOKMARK
};

Cursor faves = managedQuery(Browser.BOOKMARKS_URI, requestedColumns,
    Browser.BookmarkColumns.BOOKMARK + "=1", null,
    Browser.BookmarkColumns.VISITS + " DESC limit 5");

Log.d(DEBUG_TAG, "Bookmarks count: " + faves.getCount());

int titleIdx = faves.getColumnIndex(Browser.BookmarkColumns.TITLE);
int visitsIdx = faves.getColumnIndex(Browser.BookmarkColumns.VISITS);
int bmIdx = faves.getColumnIndex(Browser.BookmarkColumns.BOOKMARK);

faves.moveToFirst();

while (!faves.isAfterLast()) {
    Log.d("SimpleBookmarks", faves.getString(titleIdx) + " visited "
          + faves.getInt(visitsIdx) + " times : "
          + (faves.getInt(bmIdx) != 0 ? "true" : "false"));
    faves.moveToNext();
}
```

Download at http://www.pin5i.com/

同之前一样，这里定义了请求的列，进行了查询，并且 Cursor 对其结果进行了迭代。

注意：managedQuery()调用从本质上变得更复杂了。让我们更为仔细地看看它的参数。第一个参数 Browser.BOOKMARKS_URI 是针对所有浏览历史记录的 URI，而非仅仅是添加了书签的。第二个参数仍然是查询结果的请求列。第三个参数指定了“书签”属性必须为真，这是一个用来过滤查询的列。第四个参数为可选参数，仅用在使用替代值的地方，所以在这里并没有使用，其值被设置为了 null。最后，第五个参数指定了结果的排序方式（降序表示按最常访问排序）。获取浏览器访问历史记录需要设置 READ_HISTORY_BOOKMARKS 权限。

Tip

注意我们在 managedQuery()的第五个参数中同时还使用了 LIMIT 表达式。虽然没有在上面的正文中特别说明，但使用这种方法限制查询结果能够收到很好的效果，同时还能够提高应用程序的性能，特别是在返回结果很长的情况下。

11.1.4 使用联系簿（Contacts）内容提供器

“联系簿”数据库是移动电话上使用得最多的应用程序。人们总是希望能够方便、快捷地拨打朋友、家人、同事和客户的电话号码。另外，大多数手机使用“联系簿”应用程序来显示通话人的身份，包括昵称、照片或图标。

Android 提供一个内置的“联系簿”应用程序，并且将联系人数据通过内容提供器接口同其他 Android 应用程序进行分享。作为一个应用程序开发人员，这意味着你可以在你的应用程序中使用用户的联系人数据来提升用户体验。

Note

访问用户联系簿的内容提供器原先称为 Contacts。Android 2.0 引入了一个增强的联系簿管理内容提供器类来管理用户的联系簿。这个类名为 ContactsContract，它包含一个名为 ContactsContract.Contacts 的子类。对于 API Level 5，它是被推荐使用的类。不过，鉴于联系簿是各个 Android 平台版本下常用的功能，所以为了向下兼容，我们使用原始的联系簿内容提供器方法。新的 ContactsContract 方法可以参考本书网站上的相关文章（http://androidbook.blogspot.com/2010/09/contacts-contact.html）。

1. 访问私有联系人数据

你的应用程序需要特定的许可权限来访问由 Contacts 内容提供器提供的私有用户信息。你必须使用 READ_CONTACTS 许可权限来声明 uses-permission 标记以读取这些信息。

开始从“联系簿”应用程序中读取联系人数据的代码应该大致相同。

```
Cursor oneContact = managedQuery( People.CONTENT_URI, null, null, null,
    "name desc LIMIT 1");

Log.d(debugTag, "Count: " + oneContact.getCount());
```

这一短小的示例简单地演示了查询某条联系人信息的方法。我们使用了 LIMIT 来获取其中的一条记录。如果你仔细查看返回的数据列，将会发现其中不仅仅包括联系人的名字和索引（index）。数据字段并非显式地返回。相反，返回的结果包含一些用于构建对应数据的 URI 的值。我们需要使用这些索引值来请求联系人的数据。

Download at http://www.pin5i.com/

具体地，我们提取联系人的主要 E-mail 和主要电话号码。

```
int nameIdx = oneContact.getColumnIndex(Contacts.People.NAME);
int emailIDIdx = oneContact
    .getColumnIndex(Contacts.People.PRIMARY_EMAIL_ID);

int phoneIDIdx = oneContact
    .getColumnIndex(Contacts.People.PRIMARY_PHONE_ID);

oneContact.moveToFirst();
int emailID = oneContact.getInt(emailIDIdx);
int phoneID = oneContact.getInt(phoneIDIdx);
```

既然我们已经获得了联系人名字、主要 E-mail 地址和主要电话号码数据列的索引值，就可以构建同这些信息相关联的 Uri 对象，并查询其主要 E-mail 和主要电话号码了。

```
Uri emailUri = ContentUris.withAppendedId(
    Contacts.ContactMethods.CONTENT_URI,
    emailID);

Uri phoneUri = ContentUris.withAppendedId(
    Contacts.Phones.CONTENT_URI, phoneID);

Cursor primaryEmail = managedQuery(emailUri,
    new String[] {
        Contacts.ContactMethods.DATA
    },
    null, null, null);

Cursor primaryNumber = managedQuery(phoneUri,
    new String[] {
        Contacts.Phones.NUMBER
    },
    null, null, null);
```

在获取了联系人的特定 E-mail 和电话号码的适当列索引后，我们调用了 ContentUris.withAppendedId()方法以利用拥有的标识符从现有的 Uri 创建新的 Uri 对象。当已知数据表中某条记录的索引值后，就可以直接选取这一特定的数据行了，这也可以通过一个可选参数来实现。最后，我们使用两个新的 Uri 对象分别进行了 managedQuery()的两次调用。

现在我们得到了请求列 String 数组的快捷方式，因为每条记录中仅包含一列：

```
String name = oneContact.getString(nameIdx);
primaryEmail.moveToFirst();
String email = PrimaryEmail.getString(0);
primaryNumber.moveToFirst();
String number = PrimaryNumber.getString(0);
```

如果 E-mail 或电话号码并不存在，那么将会抛出一个 android.database.CursorIndexOutOfBoundsException 异常。这可以被捕捉到，或者你可以在之前就检查 Cursor 中是否已经返回了实际的值来避免异常的抛出。

2. 查询特定联系人信息

如果上面的代码对于获取一个电话号码来讲显得过于繁杂，你也不必担心。对于快速获

Download at http://www.pin5i.com/

取一条数据，确实有一个很好的解决途径。下面的代码段演示了如何获取某个联系人的主要电话号码和他的名字。联系人的主要电话号码在手机上的“联系簿”管理程序中被指定为其默认号码。如果在查询中没有获得任何结果，那么可以使用这个主要电话号码字段。

```
String[] requestedColumns = {
    Contacts.Phones.NAME,
    Contacts.Phones.NUMBER,
};

Cursor contacts = managedQuery(
    Contacts.Phones.CONTENT_URI,
    requestedColumns,
    Contacts.Phones.ISPRIMARY + "<>0",
    null, "name desc limit 1");

Log.d(debugTag, "Contacts count: "
    + contacts.getCount());

int nameIdx = contacts
     .getColumnIndex(Contacts.Phones.NAME);
int phoneIdx = contacts
     .getCoulmnIndex(Contacts.Phones.NUMBER);

contacts.moveToFirst();
Log.d(debugTag, "Name: " + contacts.getString(nameIdx));
Log.d(debugTag, "Phone: " + contacts.getString(phoneIdx));
```

这段代码看起来应该很眼熟，它是通过联系人名字查询其电话号码的简短而直接的方法。Contacts.Phones.CONTENT_URI 包含电话号码，但同时也包含联系人的名字。这与 CallLog 内容提供器相似。

11.1.5 使用 UserDictionary（用户词典）内容提供器

另一个有用的内容提供器为 UserDictionary 提供器。它可以用来在文本域中预测文本的输入，或者用在其他的用户输入机制中。存储在词典中的每一个词语通过其语种（Locale）进行组织，并且通过频率决定其权值。你可以使用 UserDictionary.Words 类的 addWord()方法来向用户自定义词典中添加词语。

11.1.6 使用 Settings 内容提供器

另一个有用的内容提供器为 Settings 提供器。你可以用它来访问设备相关设置和用户首选项。这些设置的组织方式与 Settings 应用程序中一样——通过类别进行组织。你可以在 android.provider.Settings 类中找到更多有关 Settings 内容提供器的信息。

11.2 修改内容提供器的数据

内容提供器并非只是静态的数据资源。它们同时可以用来增添、修改和删除数据，当然前提是内容提供器应用程序实现了这些功能。你的应用程序必须拥有适当的许可权限（即 WRITE_CONTACTS 和与其相对的 READ_CONTACTS）来执行这些操作。

Download at http://www.pin5i.com/

11.2.1 添加记录

在使用 Contacts 内容提供器时，我们可以使用程序向联系人数据库中添加新的记录。

```
ContentValues values = new ContentValues();

values.put(Contacts.People.NAME, "Sample User");

Uri uri = getContentResolver().insert(
    Contacts.People.CONTENT_URI, values);

Uri phoneUri = Uri.withAppendedPath(uri,
    Contacts.People.Phones.CONTENT_DIRECTORY);

values.clear();
values.put(Contacts.Phones.NUMBER, "2125551212");
values.put(Contacts.Phones.TYPE, Contacts.Phones.TYPE_WORK);

getContentResolver().insert(phoneUri, values);

values.clear();

values.put(Contacts.Phones.NUMBER, "3135551212");
values.put(Contacts.Phones.TYPE, Contacts.Phones.TYPE_MOBILE);

getContentResolver().insert(phoneUri, values);
```

如同我们使用 ContentValues 类向应用程序的 SQLite 数据库中插入记录一样，我们也可以在这里使用它。创建联系人的第一步是为 Contacts.People.Name 列指定联系人的名字。在关联其他信息（如电话号码）之前，我们需要为新联系人指定名字。你可以把它想象为在一个数据库表中创建一行新的数据，该表提供指向电话号码表的“一对多”对应。

下一步，我们向 Contacts.People.CONTENT_URI 路径下的数据库添加数据。我们使用 getContentResolver()方法来获取与 Activity 相关联的 ContentResolver。其返回值是新联系人的 Uri。我们需要使用它来为新联系人添加电话号码。之后，我们清除 ContentValues 实例并加入 Contacts.Phones.NUMBER 和 Contacts.Phones.TYPE。通过 ContentResolver，我们向新创建的 Uri 中加入了这一数据。

Tip

此时，你或许在疑惑这些数据的结构到底是怎样的。揭开谜底的最好办法是彻底地查看应用程序所使用的内容提供器的文档资料。

11.2.2 修改记录

插入数据并不是你能做的唯一变动。一行或多行数据可以被更新、修改。下面的代码段显示了如何更新内容提供器中的数据。在这个例子中，我们使用其唯一标识符更新了特定联系人的“说明（note）”字段。

Download at http://www.pin5i.com/

```
ContentValues values = new ContentValues();
values.put(People.NOTES, "This is my boss");
Uri updateUri = ContentUris.withAppendedId(People.CONTENT_URI, rowId);
int rows = getContentResolver().update(updateUri, values, null, null);
Log.d(debugTag, "Rows updated: " + rows);
```

我们又一次使用了 ContentValues 对象来建立数据字段同想要更新的数据值之间的映射（在本例中为“说明”字段）。这一过程将替换掉联系人任何 NOTES 字段中现存的记录。之后，我们创建了将要更新的联系人的 Uri。ContentResolver 类的 update() 方法简单地完成了对数据的更改。最后我们可以确认只有一条记录获得了更新。

Tip

在更新数据行时，可以使用过滤器（Filter）值。这允许同时对多行数据进行修改。虽然内容提供器必须支持这一功能，但我们会发现 Contacts Provider 在联系人 URI 上阻止了这一功能，因为它可以避免开发人员对联系人进行大面积的灾难性修改。

11.2.3 删除记录

现在你已经学习了如何使用示例用户数据来添加和修改“联系簿”，下面我们来讨论如何删除它。删除数据相对要直接一些。

1. 删除所有记录

下面的代码删除给定 URI 上所有的数据行，你也许需要比较小心地执行这段代码：

```
int rows = getContentResolver().delete(People.CONTENT_URI, null, null);
Log.d(debugTag, "Rows: "+ rows);
```

delete() 方法根据可选参数的限定删除给定 URI 上的所有符合条件的数据行。但在上面的例子中，它包含了所有 People.CONTENT_URI 下的数据行，也就是所有的联系人记录。

2. 删除特定记录

经常地，你会想要选中特定的数据行并且删除它，这可以通过将唯一标识符附加到 URI 末尾来实现，也可以通过匹配适当的模式来确定所删除的数据行。

例如，下面的代码将删除所有名字为“Sample User”的联系人记录，该条记录为之前我们所创建的。

```
int rows = getContentResolver().delete(People.CONTENT_URI,
    People.NAME + "=?",
    new String[] {"Sample User"});
log.d(debugTag, "Rows: "+ rows);
```

11.3 使用内容提供器增强你的应用程序

内容提供器的概念显得有些复杂，而通过一个实例是理解和掌握它的捷径。PetTracker 系列应用程序很好很全面，但它毕竟缺乏一些令人振奋的东西。假如说应用程序能够显示宠物的照片，那岂不更好？我们开始吧！实现这一目标只有一个难点：我们需要访问 Android

Download at http://www.pin5i.com/

系统中的另一个应用程序——Media Store 应用程序所提供的图片。

Tip

这一部分中的许多代码示例都来自于 PetTracker3 应用程序。在本书的网站上可以下载 PetTracker3 应用程序的源代码。

在图 11.1 中，你可以看到使用了 Media Store 内容提供器的 Pet Tracking 工程的运行结果。

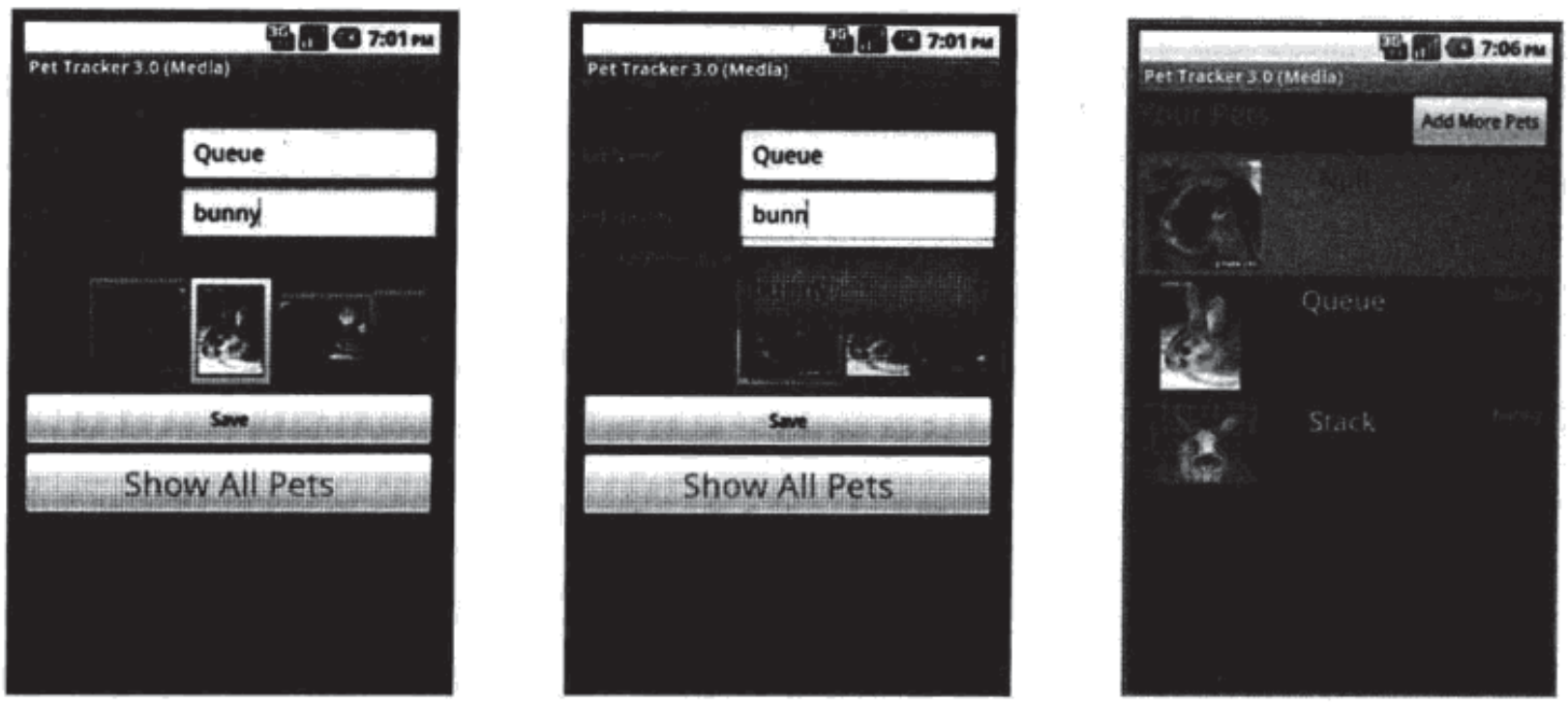


图 11.1 Pet Tracker 应用程序：Entry 界面（左、中）和 Pet Listing 界面（右）

既然现在你能够看到带有照片显示功能的应用程序，那么我们先停止实现它的步伐。PetTracker3 应用程序拥有同之前的 Pet Tracker 工程相同的基础架构，但有以下几个关键之处不同。

① 在 Pet Entry 界面中，你可以从 `Gallery` Widget 中选择一幅照片。`Gallery` 中显示了所有 SD 卡（或模拟器上的模拟 SD 卡）中的图像，这是通过访问 `MediaStore` 内容提供器实现的（如图 11.1（左）所示）。

② 在 Pet Listing 界面中，每一幅图片都显示在 `ListView` 中（如图 11.1（右）所示），这里再次使用 `MediaStore` 内容提供器来访问特定的图像。

③ 在 Pet Listing 界面中，`ListView` 的每一个列表项（如图 11.1（右）所示）均使用了自定义的布局。新的 PetTracker3 示例应用程序显示了实现它的两种方法：一种是通过在现有的自定义 XML 文件的基础上进行扩展，另一种则是通过程序生成布局。

④ 在内部，我们两次继承了 `BaseAdapter`，从而成功地将数据与 `ListView` 和 `Gallery` 分别进行绑定，从而满足了我们个性化的需求。

⑤ 最后，我们提供了对于 `SimpleCursorAdapter.CursorToStringConverter` 类和 `FilterQueryProvider` 类方法的自定义实现，它使得 `AntoCompleteTextView` 与内部的 SQLite 数据库表 `pet_types` 直接绑定（如图 11.1（中）所示），并且改变了 `AutoCompleteTextView` 的行为以匹配字符串的任意位置，而非仅仅是开头。尽管我们在之后的内容中并不会详细地讲解这一内容的细节，但是最好能够查看示例代码，以获得有关这一特殊实现的更多信息。

首先，我们需要确定从哪里获得照片。我们可以使用内置的照相机拍照以便能够访问它们，但是在模拟器（只能拍“假照片”）上使用时，出于简便的考虑，还是下载一些又可爱又有趣的宠物照片到 SD 卡里比较好。

Download at http://www.pin5i.com/

Tip

要运行 PetTracker3 示例应用程序，你需要对模拟器进行一些配置以便能够使用虚拟的 SD 卡。为了使代码简洁可读，我们并不对诸如没有图像这样的错误进行处理，也不对媒体类型进行检查。

当你可以在模拟器中运行浏览器浏览某个网站，并且下载一些图片时，说明你已经进行了正确的设置。你可以在 Gallery 应用程序中浏览这些图片。

要通过浏览器下载一幅图像，可以选中它并选择“保存图片”选项。你可以下载一些你喜欢的宠物（或小孩等）到 SD 卡中，你也可以借用我们的小兔图片，示例应用程序中使用了它们，你也可以从下面的网址获取：http://tinyurl.com/geekybuns。

1. 使用 URI 在 Android 系统上定位内容

大多数对内容提供器的访问是以查询的形式呈现的：联系人列表、书签列表、通话列表以及音频文件列表。应用程序使得对这些数据的请求类似于对数据库的访问，并且可以获得相同结构的返回结果。查询的结果通常需要使用一个 Cursor 进行迭代。不过，为了避免麻烦的查询，我们使用 URI。

你可以把 URI 看作是内存存在的“地址”。URI 地址是层级化的。大多数的内容提供器，比如联系簿和 MediaStore，定义了一些预定义的 URI。例如，要访问外部的媒体设备（也就是 SD 卡）中的图像，我们可以使用在 MediaStore.Images.Media 类中定义的如下 URI：

```
Uri mMedia = Media.EXTERNAL_CONTENT_URI;
```

2. 使用 managedQuery()获取内容提供器数据

我们可以像查询数据库一样地使用 URI 查询“Media Store”内容提供器。现在我们使用 managedQuery()方法返回一个包含了 SD 卡上所有媒体文件的 Cursor。

```
String[] projection = new String[] { Media._ID, Media.TITLE };

Uri mMedia = Media.EXTERNAL_CONTENT_URI;

Cursor mCursorImages = managedQuery(mMedia, projection, null, null,
    Media.DATE_TAKEN + " ASC"); // ORDER BY 子句
```

现在我们获得了 SD 卡上存储的每一个可用的媒体文件。

现在我们拥有了这一 Cursor，但仍然有一些杂务需要完成，以便能够使用 Gallery 控件显示这些单独的图片。

3. 向 Gallery 控件绑定数据

我们需要继承 BaseAdapter 类来生成一个新的名为 ImageUriAdapter 的数据适配器，将获取的 URI 数据与 Gallery Widget 绑定。我们自定义的 ImageUriAdapter 将 Cursor 的结果映射到 GalleryRecord 对象数组中，它与 Gallery Widget 中的子项一一对应。ImageUriAdapter 的代码太长，这里仅对必须实现的方法进行讲解。

① ImageUriAdapter()构造函数负责将 Cursor 映射到 GalleryRecord 对象，它封装了 URI 的主干（base）部分和可以附加在主干 URI 之后的单独的图像 id，从而得到图

Download at http://www.pin5i.com/

像的完整 URI。

② getItem()和 getItemId()方法返回特定图像的唯一标识符。当用户在 Gallery 中单击某个图片时，唯一标识符的作用就体现出来了。我们将这些信息保存到数据库中，这样就可以知道哪幅图片对应哪只宠物了。

③ getView()方法返回自定义的 View 控件，它与 Gallery 中的每一个子 View 相对应。在这种情况下，我们返回包含对应图像的 ImageView。我们将每一个视图的 Tag 属性设置为其关联的 GalleryRecord 对象，它包含了 Cursor 的所有映射信息。这是一个很有意思的 Widget 存储扩展信息的方法，存储的信息可以在之后使用。

在所有的这些方法都实现了之后，我们可以将这个新实现的自定义适配器连同 Cursor 一起赋予 Gallery。

```
ImageUriAdapter iAdapter = new ImageUriAdapter(this,
    mCursorImages, mMedia);
final Gallery pictureGal = (Gallery) findViewById(R.id.GalleryOfPics);
pictureGal.setAdapter(iAdapter);
```

4. 获取选中的 Gallery 图像并将其存入数据库中

注意我们向 SQLite 数据库中添加了两个新的列：图像的 URI 的主干（base）部分和独立的图像 id，它是附加到主干 URI 的唯一标识符。我们并非将图像存储在数据库中，而是仅仅使用 URI 信息来获取它们。

当用户在 Pet Entry 界面按下 Save 按钮时，我们将检查选中的 Gallery 项，然后从所选 View 的 Tag 属性中提取出所需的信息，如下所示：

```
final Gallery gall = (Gallery) findViewById(R.id.GalleryOfPics);

ImageView selectedImageView = (ImageView) gall.getSelectedView();
GalleryRecord galleryItem;

if (selectedImageView != null) {
   galleryItem = (GalleryRecord)selectedImageView.getTag();
   long imageId = galleryItem.getImageId();
   String strImageUriPathString = galleryItem.getImageUriPath();
}
```

之后，我们就可以像之前一样保存宠物信息了。

5. 使用 URI 显示 SD 卡图像

既然我们已经能够保存宠物信息了，那么接下来让我们把目光转移到 Pet Listing 界面上。我们的 ListView 正变得越来越复杂：每一项都包含一个 ImageView 和两个 TextView 控件，分别显示宠物的名字和种类。我们从为每一个 ListView 列表项定义一个名为 pet_item.xml 的布局模板开始。这应该比较熟悉，它包含一个 ImageView 和两个 TextView 对象。

我们想要确保这一实现是可扩展的，以免在今后向单独的 ListView 列表项中添加新功能时出现问题。所以我们不使用标准的适配器和内建的 Android 布局模板，而是使用另一个名为 PetListAdapter 的自定义适配器。

Download at http://www.pin5i.com/

PetListAdapter 同我们之前为 Gallery 控件所实现的 ImageUriAdapter 类似。只不过这次不是针对 Gallery 子项，而是 ListView 的子记录，它与每一个宠物相对应。和之前一样，构造函数建立 Cursor 数据同 PetRecord 对象数组的映射。

PetListAdapter 的 getView() 方法是整个过程的关键。这里我们使用了一个 LayoutInflater 来扩展适用于每一个 ListView 列表项的名为 pet_item.xml 的自定义布局文件。我们再一次使用了视图的 Tag 属性来存储之后可能会用到的信息。正是在这个地方，我们使用了存储在数据库中的 URI 信息，借助 Uri.parse() 和 ContentUris.withAppendedId() 方法重建了完整的图像 URI，同时使用 setImageURI() 方法将 URI 和对应的 ImageView Widget 关联起来。

既然我们已经准备就绪，那么就可以将 PetListAdapter 关联到 ListView 了：

```
String asColumnsToReturn[] = {
    Pets.PETS_TABLE_NAME + "." + Pets.PET_NAME,
    Pets.PETS_TABLE_NAME + "." + Pets._ID,
    Pets.PETS_TABLE_NAME + "." + Pets.PET_IMAGE_ID,
    PetType.PETTYPE_TABLE_NAME + "." + PetType.PET_TYPE_NAME };

mCursor = queryBuilder.query(mDB, asColumnsToReturn, null, null,
    null, null, Pets.DEFAULT_SORT_ORDER);
startManagingCursor(mCursor);

SetListAdapter adapter = new PetListAdapter(this, mCursor);
ListView av = (ListView) findViewById(R.id.petList);
av.setAdapter(adapter);
```

这就是对于上面讲解内容的实现。需要注意的是你仍然可以通过程序来创建 ListView 列表项的布局（参考 PetListItemView 类和 PetListAdapter.getView() 方法的注释以获取更多信息）。

现在你已经知道了如何借助内容提供器来提升应用程序的健壮性，但这个例子仅仅只是管中窥豹地揭示了内容提供器的部分特性。

11.4 实现内容提供器

你的应用程序中拥有数据吗？其他应用程序能够使用它们扩展出有趣的应用吗？要与其他的应用程序共享你应用程序中的数据，你需要首先通过为其他应用程序提供标准化的内容提供器接口来成为一个内容提供器；之后，你必须在 Android Manifest 文件中将应用程序注册为内容提供器。成为内容提供器最直接的方法就是将需要共享的信息存储在 SQLite 数据库中。

其中的一个例子就是 GPS 轨迹点（track point）内容提供器。它允许用户查询和保存坐标点。每一个点的数据包括时间戳、经纬度和海拔高度。

Tip

这一部分中的许多代码示例都来自于 Tracks 应用程序。在本书的网站上可以下载 Tracks 应用程序的源代码。为了使用这个应用程序，你需要获得你自己的 Google API Key: http://code.google.com/android/maps-api-signup.html。

Download at http://www.pin5i.com/

11.4.1 实现内容提供器接口

实现一个内容提供器接口相对来说比较容易。下面的代码显示了应用程序成为一个内容提供器所需实现的基本接口，具体来说需要实现 5 个重要的方法：

Tip

可以使用 Eclipse 来方便地创建一个新的类，同时包含所需的基本覆写。方法为：右键单击需要添加类的包，选择 New（新建）→Class（类），在 Name 域中输入内容提供器的名称，选择 android.content.ContentProvider 作为其父类，并且勾选 Inherited abstract methods（继承抽象方法）复选框。

```
public class TrackPointProvider extends ContentProvider {

    public int delete(Uri uri,
        String selection, String[] selectionArgs) {
        return 0;
    }

    public String getType(Uri uri) {
        return null;
    }

    public Uri insert(Uri uri, ContentValues values) {
        return null;
    }

    public boolean onCreate() {
        return false;
    }

    public Cursor query(Uri uri, String[] projection,
        String selection, String[] selectionArgs, String sortOrder) {
        return null;
    }

    public int update(Uri uri, ContentValues values,
        String selection, String[] selectionArgs) {
            return 0;
        }
    }
```

11.4.2 定义数据 URI

提供器应用程序需要定义一个“基底（base）”URI 以供其他应用程序访问这一内容提供器。这一 URI 的定义形式必须为 public static final Uri，名称必须为 CONTENT_URI，并且必须以“content://”开头。URI 必须是唯一的，这样最好使用内容提供器的完整有效类名称来命名它。下面的代码为我们的 GPS 轨迹点 Provider 创建了一个 URI 名称：

```
public static final Uri CONTENT_URI =
    Uri.parse("content://com.androidbook.TrackPointProvider");
```

Download at http://www.pin5i.com/

11.4.3 定义数据列

使用内容提供器的用户必须知道这个内容提供器都提供哪些数据列。在这个例子中，使用的列有时间戳、经纬度和海拔高度，另外还包含记录编号的列，名为_id。

```
public final static String _ID = "_id";
public final static String TIMESTAMP = "timestamp";        //时间戳
public final static String LATITUDE = "latitude";          //纬度
public final static String LONGITUDE = "longitude";        //经度
public final static String ELEVATION = "elevation";        //海拔高度
```

内容提供器的用户使用和上面相同的字符串。像上面这样的内容提供器通常将数据存储在SQLite 数据库中。如果确实是这样，那么使列名称同数据库字段名称相匹配能够简化代码。

11.4.4 实现重要的内容提供器方法

这一小节介绍上面提到的 5 个方法的具体实现，它们在另外的应用程序使用内容提供器时将为系统所调用。这里的“系统”即指 ContentResolver 接口，它在前面的内容中已经被我们用来直接地访问内建内容提供器。

其中的部分方法可以使用由 Android SDK 提供的一种辅助类，即 UriMatcher，它用来以某种模式匹配到达的 Uri 值，从而加速开发过程。UriMatcher 的描述和使用将在下面几个方法的实现中讲解。

1. 实现 query()方法

我们从一个简单示例查询的实现开始。任何查询的实现都需要返回一个 Cursor 对象。获得Cursor对象的一种便捷方式为返回许多内容提供器所使用的来自底层SQLite数据库的Cursor。事实上，ContentProvider.query()与 SQLiteQueryBuild.query()调用相兼容。下面的例子使用它来快速地构建查询并返回了一个 Cursor 对象。

```
public Cursor query(Uri uri, String[] projection,
    String selection, String[] selectionArgs,
    String sortOrder) {

    SQLiteQueryBuilder qBuilder = new SQLiteQueryBuilder();

    qBuilder.setTables(TrackPointDatabase.TRACKPOINTS_TABLE);

    if ((sURIMatcher.match(uri)) == TRACKPOINT_ID) {
        qBuilder.appendWhere("_id=" + uri.getLastPathSegment());
    }

    Cursor resultCursor = qBuilder.query(mDB
        .getReadableDatabase(), projection,
        selection, selectionArgs, null, null,
        sortOrder, null);

    resultCursor.setNotificationUri(getContext()
         .getContentResolver(), uri);
    return resultCursor;
}
```

首先，代码获得了 SQLiteQueryBuilder 对象的实例，它使用一些方法调用来建立查

Download at http://www.pin5i.com/

询。之后，setTables()方法指定需要使用的数据库表。UriMatcher 类检查所请求的行，这将在稍后详细讨论。

然后，实际的查询被调用。内容提供器的查询相对 SQLite 查询来讲要简单一些，所以部分参数被传入，而另一部分则略去了。SQLite 数据库的实例为只读的。由于这仅仅是对数据的查询，所以是可以接受的。

最后，Cursor 需要知道是否有源数据被修改了。这是通过调用 setNotificationUri() 方法实现的，它将被告知观察哪一个 URI 下的数据修改。对应用程序 query() 方法的调用可以来自多个线程，这样，在 Cursor 返回后发生数据更改是有可能的。所以，通过这种方法能够保持数据的同步。

2. 探索 UriMatcher 类

UriMatcher 类是用于模式匹配的辅助类，它针对传入内容提供器的 URI。在实现必须实现的内容提供器时经常用到。下面是用在这些示例当中的 UriMatcher：

```
public static final String AUTHORITY =
    "com.androidbook.TrackPointProvider"

private static final int TRACKPOINTS = 1;
private static final int TRACKPOINT_ID = 10;

private static final UriMatcher sURIMatcher =
    new UriMatcher(UriMatcher.NO_MATCH);
static {
    sURIMatcher.addURI(AUTHORITY, "points", TRACKPOINTS);
    sURIMatcher.addURI(AUTHORITY, "points/#", TRACKEPOINT_ID);
}
```

首先，对于每一个不同的模式，随意指定一个数值以标识它们。之后，创建了一个 UriMatcher 的静态实例。构造函数所需的参数仅仅是在没有匹配时的返回值。在 UriMatcher 类之中提供了这样的一个值。

然后，URI 值连同其对应的标识符被添加进了匹配器（Matcher）中。URI 被打散成授权部分（authority portion）和路径部分（path portion），前者定义在 AUTHORITY 中，后者以原始字符串的形式传入。路径也可以包含模式，比如“#”符号代表一个数字，而“*”符号是一个通配符代表任何文字。

3. 实现 insert()方法

insert()方法用于向内容提供器中添加数据。下面是 insert()方法的一种示例实现：

```
public Uri insert(Uri uri, ContentValues values) {

    int match = sURIMatcher.match(uri);
    if (match != TRACKPOINTS) {
        throw new IllegalArgumentException(
                "Unknown or Invalid URI " + uri);
    }

    SQliteDatabase sqlDB = mDB.getWritableDatabase();
```

Download at http://www.pin5i.com/

```
        long newID = sqlDB.
            insert(TrackPointDatabase.TRACKPOINTS_TABLE, null, values);

        if (newID > 0) {
            Uri newUri = ContentUris.withAppendedId(uri, newID);
            getContext()
                .getContentResolver().notifyChange(newUri, null);
            return newUri;
        }

        throw new SQLException("Failed to insert row into " + uri);
    }
```

代码首先对 Uri 进行了验证以确保这正是所需进行输入插入的地方，比如，一个指向特定行的 Uri 就不符合要求。下一步，代码获取了一个可写入的数据库对象实例。通过它，可以根据传入的 Uri 所指定的数据库表以及传入的值完成数据的插入。在这里，我们没有对传入值进行任何错误检查。相反，底层的数据库实现能够抛出异常，这一异常可以被内容提供器的用户所处理。

如果插入操作成功完成，将会创建一个 Uri 以通知系统底层数据被修改了，这是通过 ContentResolver 的 notifyChange()方法实现的，否则，将会抛出一个异常。

4. 实现 update()方法

update()方法用来修改一行已存的数据。它拥有与 insert()和 query()方法相似的内容。数据的更新应用于由传入的 Uri 对象所指定的数据。

```
    public int update(Uri uri, ContentValues values,
        String selection, String[] selectionArgs) {

        SQLiteDatabase sqlDB = mDB.getWritableDatabase();
        int match = sURIMatcher.match(uri);
        int rowAffected;

        switch (match) {
            case TRACKPOINTS:
                rowAffected = sqlDB.update(
                    TrackPointDatabase.TRACKPOINTS_TABLE,
                    values, selection, selectionArgs);
                break;

            case TRACKPOINT_ID:
                String id = uri.getLastPathSegment();
                if (TextUtils.isEmpty(selection)) {
                    rowsAffected = sqlDB.update(
                        TrackPointDatabase.TRACKPOINTS_TABLE,
                        values, _ID + "=" + id, null);
                } else {
                    rowsAffected = sqlDB.update(
                        TrackPointDatabase.TRACKPOINTS_TABLE,
                        values, selection + " and " + _ID + "="
                        +id, selectionArgs);
                }
                break;
            default:
```

Download at http://www.pin5i.com/

```
                throw new IllegalArgumentException(
                    "Unknown or Invalid URI " + uri);
        }

        getContext().getContentResolver().notifyChange(uri, null);
        return rowsAffected;
    }
```

在上面这段代码中，我们获取了一个可写入的 SQLiteDatabase 实例，并且通过调用 UriMatcher 的 match()方法对用户传入的 Uri 类型进行了验证。但是这里并没有进行任何值和参数的检验。不过，要终止对于特定 Uri（如影响多行数据的 Uri 或者 TRACKPOINT_ID 的匹配）的更新，可以抛出 java.lang.UnsupportedOperationException 异常来进行指明。在这个例子中，内容提供器充分信任其用户，所以没有进行类似的处理。

在调用适当的 update()方法后，通过对 notifyChange()方法的调用通知系统对应 URI 的数据进行了更改。这将通知任何该 URI 数据的监听者数据发生了更改。最后，返回了被改变的行数目，这也是 update()方法一般需要返回的值。

5. 实现 delete()方法

现在我们来学习清除数据库。下面是实现 delete()方法的示例。你会发现，它并没有检测用户是否删除了过多的数据，而且你还会发现，它与前面的 update()方法非常类似。

```
public int delete(Uri uri, String selection, String[] selectionArgs) {
    int match = sURIMatcher.match(uri);

    SQLiteDatabase sqlDB = mDB.getWritableDatabase();
    int rowsAffected = 0;
    switch (match) {

        case TRACKPOINTS:
            rowsAffected = sqlDB.delete(
                TrackPointDatabase.TRACKPOINTS_TABLE,
                selection, selectionArgs);
            break;

        case TRACKPOINT_ID:
            String id = uri.getLastPathSegment();
            if (TextUtils.isEmpty(selection)) {
            rowsAffected =
                sqlDB.delete(TrackPointDatabase.TRACKPOINTS_TABLE,
                _ID+"="+id, null);
            } else {
                rowsAffected =
                    sqlDB.delete(TrackPointDatabase.TRACKPOINTS_TABLE,
                    selection + " and " +_ID+"="+id, selectionArgs);
            }
            break;
        default:
            throw new IllegalArgumentExpection(
                "Unknown or Invalid URI " + uri);
    }
    getContext().getContentResolver().notifyChange(uri, null);

    return rowsAffected;
}
```

Download at http://www.pin5i.com/

同之前一样，代码获取了一个可写入的数据库实例，并且通过调用 UriMatcher 的“匹配”方法对用户传入的 Uri 类型进行了验证。如果结果是一个目录 Uri，那么将使用用户传入的可选参数执行删除。但如果结果是某个特定的数据行，那么行索引值将用来限制之后的删除行为，无论是否考虑可选参数。允许不使用可选参数而进行针对特定标识符的删除操作使得其不用关心这一操作来自哪里。

与之前一样，通过对 ContentResolver 的 notifyChange()方法的调用通知了系统数据的更改。也与前面一样，返回了被修改的行数目，我们将其暂存下来作为 delete()方法的返回值。

6. 实现 getType()方法

最后需要实现的方法是 getType()方法。该方法的作用是为传入的特定 Uri 返回其 MIME 类型。它不需要返回针对数据列的 MIME 类型。

```
public static final String CONTENT_ITEM_TYPE =
    ContentResovler.CURSOR_ITEM_BASE_TYPE +
    "/track-points";

public static final String CONTENT_TYPE =
    ContentResolver.CURSOR_DIR_BASE_TYPE +
    "/track-points";

public String getType(Uri uri) {
    int matchType = sURIMatcher.match(uri);
    switch (matchType) {

        case TRACKPOINTS:
            return CONTENT_TYPE;

        case TRACKPOINT_ID:
            return CONTENT_ITEM_TYPE;

        default:
            throw new
                IllegalArgumentException("Unknown or Invalid URI "
                + uri);
    }
}
```

首先，代码定义了两个 MIME 类型。Android SDK 为单独的项目或目录提供了“指导值（guideline value）”，它们被用在了这里。对应的字符串分别为 vnd.android.cursor.item 和 vnd.android.cursor.dir。最后，match()方法检测了所提供 Uri 的类型，这样才能返回正确的 MIME 类型。

11.4.5 更新 Manifest 文件

最后，你需要更新应用程序的 AndroidManifest.xml 文件以向系统中的其他应用程序提供这一内容提供器接口。在这里需要设定类名和授权（authority），或者 content://URI 所指定的域名。例如，content://com.androidbook.TrackPointProvider 为上面的内容提供器所使用的“基底”URI，这意味着其授权为 com.androidbook.TrackPointProvider。

Download at http://www.pin5i.com/

下面的 XML 显示了这个例子：

```
<provider
    android:authorities="com.mamlambo.gpx.TrackPointProvider"
    android:multiprocess="true"
    android:name="com.mamlambo.gpx.TrackPointProvider"
</provider>
```

multiprocess 的值被设置为了真，这是因为这些数据不需要在同时运行的多个内容提供器实例中进行同步。两个或更多的应用程序同时访问这个内容提供器是完全有可能的，所以适当的“同步性”设置显得很有必要。

Note

我们经常性地引用发送给“观察者”的通知。在第 20 章“使用通知”中，你将学习发送至设备的通知的有关内容。

11.5 使用动态文件夹（Live Folder）

LiveFolder（android.provider.LiveFolder）是一个强大的功能，它补充了内容提供器的接口。LiveFolder 是一个包含由内容提供器生成的内容的特殊文件夹。例如，用户可以创建一个 LiveFolder，它包含其最喜欢的联系人（Fave Five）、在一个自定义 E-mail 应用程序中最常打开的 E-mail，或者在任务管理应用程序中具有高权限的任务。

当用户选择创建一个 LiveFolder 时，Android 系统将提供一个所有能够响应 ACTION_CREATE_LIVE_FOLDER Intent 的 Activity 列表。如果用户选择了你的 Activity，那么这个 Activity 将创建 LiveFolder 并且使用 setResult()方法将其传回给系统。

LiveFolder 由下面几个组件构成：

① 文件夹名称；

② 文件夹图标；

③ 显示模式（网格或列表）；

④ 为文件夹提供内容的内容提供器 URI。

让内容提供器为 LiveFolder 提供数据的首要任务是为需要处理可用 LiveFolder 的 Activity 提供一个<intent-filter>。这通过 AndroidManifest.xml 文件实现，如下所示：

```
<intent-filter>
    <action android:name=
        "android.intent.action.CREATE_LIVE_FOLDER" />
    <category
        android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

下一步，需要在已经被定义的 Activity 的 onCreate()方法中处理这一行为。结合前面的 Provider 示例，下面的代码可以用来处理这一操作：

```
super.onCreate(savedInstanceState);

final Intent intent = getIntent();
final String action = intent.getAction();
if (LiveFolders.ACTION_CREATE_LIVE_FOLDER.equals(action)) {
```

Download at http://www.pin5i.com/

```
        final Intent resultIntent = new Intent();

        resultIntent.setData(TrackpointProvider.LIVE_URI);
        resultIntent.putExtra(
            LiveFolders.EXTRA_LIVE_FOLDER_NAME, "GPX Sample");
        resultIntent.putExtra(LiveFolders.EXTRA_LIVE_FOLDER_ICON,
            Intent.ShortcutIconResource.fromContext(
            this, R.drawable.icon));
        resultIntent.putExtra(LiveFolders.EXTRA_LIVE_FOLDER_DISPLAY_MODE,
            LiveFolders.DISPLAY_MODE_LIST);

        setResult(RESULT_OK, resultIntent);
    } // ... onCreate()方法的剩余代码
```

上面的代码定义了 LiveFolder 的核心组件：名称、图标、显示模式和 Uri。Uri 并不与现存的完全相同，因为它还需要某些特定的字段。这直接指向了下一个任务：修改内容提供器以准备提供给 LiveFolder 的数据。

首先，我们定义了一个新的 Uri。在这个例子中，“/live”被加入到了现存 CONTENT_URI 的结尾。例如：

```
public static final Uri LIVE_URI = Uri.parse("content://"
    + AUTHORITY + "/" + TrackPointDatabase.TRACKPOINTS_TABLE
    + "/live");
```

这一新的 Uri 模式被添加到了 UriMatcher 中。接下来，修改了 query() 的实现以能够识别出新的 Uri，我们为其添加了一个“方案（projection）”，定义如下：

```
switch (sURIMatcher.match(uri)) {
case TRACKPOINT_ID:
    qBuilder.appendWhere("_id=" + uri.getLastPathSegment());
    break;
case TRACKPOINTS_LIVE:
    qBuilder.setProjectionMap(
        TRACKPOINTS_LIVE_FOLDER_PROJECTION_MAP);
    break;
// ... 其他选择支
}
Cursor c = qBuilder.query( // ...
```

“方案”对于使用中的 LiveFolder Provider 而言非常重要。在返回的 Cursor 中有两个字段是必需的：LiveFolder._ID 和 LiveFolder.NAME。另外，除了这两个之外，其他的字段（例如 LiveFolder.DESCRIPTION）能够修改视图的外观和行为。在这个例子中，TIMESTAMP 用于名称，如下面的“方案”实现所示：

```
private static final Hashmap<String,String>
    TRACKPOINTS_LIVE_FOLDER_PROJECTION_MAP;
static {
    TRACKPOINTS_LIVE_FOLDER_PROJECTION_MAP =
        new HashMap<String,String>();
    TRACKPOINTS_LIVE_FOLDER_PROJECTION_MAP.put(
        LiveFolders._ID, _ID + " as " + LiveFolders._ID);
    TRACKPOINTS_LIVE_FOLDER_PROJECTION_MAP.put(
        LiveFolders.NAME, TIMESTAMP + " as " + LiveFolders.NAME);
}
```

Download at http://www.pin5i.com/

在这之后，LiveFolder 应该能够“活动”了。在这个例子中，它会显示一个日期列表，如图 11.2 所示。

图 11.2　显示日期的示例 LiveFolder 列表

小　结

你的应用程序可以使用并修改其他充当内容提供器的 Android 应用程序所提供的数据。Android 提供的 MediaStore、Browser、CallLog 和 Contacts 内容提供器均可被 Android 应用程序所访问，从而增强应用程序的功能。应用程序还可以成为内容提供器以向其他应用程序提供数据。成为内容提供器包含对几个方法的实现，它们用于管理向其他应用程序提供数据的内容和方式，甚至可以借助 LiveFolder 直接将数据提供给主页。

参考资料和更多信息

Android Dev Guide（开发指南）.: Content Provider. http:// developer. android.com/guide/ topics/ providers/content-providers.html.

Download at http://www.pin5i.com/

第 12 章

使用 Android 网络 API

带有网络组件的应用程序远比其他应用程序更动态、更灵活，内容也丰富得多。网络对于应用程序而言有非常多的作用：可以提供更新和升级的内容，可以拥有某些社区网络的特性而非仅仅是一个单机运行的程序，可以将繁重的处理工作交给服务器来完成，还可以将数据存储在超越用户自身手机范围的地方。

熟悉 Java 网络的读者应该会很熟悉 `java.net` 包。除此，Android 还提供了许多功能（utility）类，它们用于处理各种网络操作和协议。本章将聚焦于超文本传输协议（Hypertext Transfer Protocol，HTTP），它是移动网络应用程序使用得最多的协议。

12.1 移动网络基础

Android 平台下的网络连接具有一个标准化的处理，它结合了熟悉而强大的库和技术，例如 `java.net`。尽管在应用程序中实现网络连接一般来讲是一种很直接的方式，但移动应用程序开发人员依然需要考虑网络连接不稳定的情况，例如当用户在家里或办公室时的网络状态（网络的连接性）依赖于用户所处的位置和所使用的设备，而用户需要的是稳定、流畅的应用程序，这意味着你需要在设计网络应用程序时给予更多的付出。幸运的是，Android SDK 提供了一系列的工具和类来保障这一点。

Warning

作为 Android 软件开发包许可协议（License Agreement）的一部分，开发人员需要同意一系列有关网络应用程序的最佳方案（best practice）。如果你打算在应用程序中使用网络支持，那么最好能够重温协议中的这些条款，以保证开发出的应用程序符合协议。

12.2 访问因特网（HTTP）

最常见的向/从网络传输数据的方式是使用 HTTP。你可以使用 HTTP 来封装几乎所有类型的数据，并且可以使用加密套接字协议层（Secure Sockets Layer，SSL）来保障数据的安全性，这在传输具有一定私密性的数据时将会发挥非常重要的作用。除此，HTTP 所使用的大多数端口通常也可以从手机网络打开。

Tip

这一部分中的许多代码示例都来自于 SimpleNetworking 应用程序。在本书的网站上可以下载 SimpleNetworking 应用程序的源代码。

Download at http://www.pin5i.com/

12.2.1 从 Web 读取数据

从 Web 读取数据是一个极为简单的过程。例如，如果你仅仅需要从一个 Web 站点读取一些数据，而且你已经有了这个站点的地址，那么你就可以通过一个 URL 类（作为 java.net 包的一部分）来读取 Web 服务器上某个文件的定长部分，如下所示：

```
import java.io.InputStream;
import java.net.URL;

// ...

URL text = new URL(
    "http://api.flickr.com/services/feeds/photos_public.gne" +
    "?id=26648248@N04&lang=en-us&format=atom");

InputStream isText = text.openStream();
byte[] bText = new byte[250];
int readSize = isText.read(bText);
Log.i("Net", "readSize = " + readSize);
Log.i("Net", "bText = "+ new String(bText));
isText.close();
```

首先，代码创建了一个新的 URL 对象，这需要传入我们想要读取的数据所在的 URL。然后，为这个 URL 资源打开了一个流（stream）。从这里开始，我们读取数据并关闭 InputStream。从服务器读取数据就是这么简单。

> **Note**
>
> 正如我们在本书引言中所说的，考虑到可读性，示例代码中的异常处理语句被省略了。尽管如此，在设计与网络有关的代码时，你会经常需要为示例代码添加一些异常处理的机制以便进行调试。当然，你也可以查看本书网站上提供的示例代码，其中包含正确的异常处理机制的实现。

不过，一定要记住，由于我们是在同网络资源打交道，所以错误随时都有可能发生。我们的手机可能没有网络覆盖；服务器可能因为维护的原因而关闭，甚至完全消失了；URL 可能无效；也可能由于网络用户操作时间过长而发生超时错误。

上面讲解的方法可能适用于某些情况，例如当应用程序只需要一个轻量级的、不太关键的网络特性时。在许多情况下，从一个 URL 读取数据之前可能需要了解一些更多的信息，比如，需要读取的数据到底有多大。

最后，为了 Android 应用程序能够正常地使用网络特性，许可权限是必不可少的。你的应用程序需要在 AndroidManifest.xml 文件中包含下面的表达式：

```
<uses-permission
    android:name="android.permission.INTERNET" />
```

12.2.2 使用 HttpURLConnection

我们可以使用 HttpURLConnection 对象来对 URL 进行一些侦查，以避免错误地传输过多的数据。HttpURLConnection 获取一些有关 URL 对象所引用资源的信息，包括 HTTP 状态（status）和头（header）信息。

Download at http://www.pin5i.com/

你还可以从 HttpURLConnection 对象中获取内容的长度、类型和日期时间等信息，这样你可以检查从上次访问了某个 URL 后其数据有否变动。

下面是一个 HttpURLConnection 的简短示例，它查询了上面我们使用过的那个 URL 地址：

```
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;

//...

URL text = new URL(
    "http://api.flickr.com/services/feeds/photos_public.gne
➥?id=26648248@N04&lang=en-us&format=atom");
HttpURLConnection http =
    (HttpURLConnection)text.openConnection();
Log.i("Net", "length = " + http.getContentLength());
log.i("Net", "respCode = " + http.getResponseCode());
Log.i("Net", "contentType = "+ http.getContentType());
Log.i("Net", "content = "+http.getContent());
```

上面代码中的“Log”行演示了 HttpURLConnection 类的几个重要方法的使用。如果 URL 的内容被验证是正确的，那么之后你可以调用 http.getInputStream()方法来获得一个与之前相同的 InputStream 对象。从这里开始，读取网络资源的过程就变得和之前一样了，但我们获得了有关这一资源的更多信息。

12.2.3 解析从网络获取的 XML

很大一部分网络资源的传输存储在一种结构化的形式中，它称为“可扩展标记语言（Extensible Markup Language，XML）”。举例来讲，RSS 种子就是以标准 XML 的格式提供的，它用来为许多网站提供其数据。

Android SDK 提供了各种类型的 XML 功能包。在第 6 章“管理应用程序资源”中，我们初步认识了 XML Pull Parser。在第 10 章“使用 Android 数据和存储 API”中，我们还讲解了各种可用的 SAX 和 DOM 支持。

从网络解析 XML 类似于解析 XML 资源文件或者文件系统中的原始文件。Android 提供了一种快速而高效的 XML Pull Parser，它是网络应用程序解析器的首选。

下面的代码演示了如何使用 XML Pull Parser 从 flickr.com 读取一个 XML 文件，并且从中提取出一定的信息。其中，一个名为 status 的 TextView 在这段代码之前已经进行了关联，它可以显示解析操作的状态。

```
import java.net.URL;

import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserFactory;

//...

URL text = new URL(
    "http://api.flickr.com/services/feeds/photos_public.gne
➥?id=26648248@N04&lang=en-us&format=atom");

XmlPullParserFactory parserCreator =
```

Download at http://www.pin5i.com/

```
    XmlPullParserFactory.newInstance();
XmlPullParser parser = parserCreator.newPullParser();

parser.setInput(text.openStream(), null);

status.setText("Parsing...");
int parserEvent != parser.getEventType();
while (parserEvent != XmlPullParser.END_DOCUMENT) {
    switch(parserEvent) {
        case XmlPullParser.START_TAG:
            String tag = parser.getName();

            if (tag.compareTo("link") == 0) {
                String relType =
                    parser.getAttributeValue(null, "rel");

                if (relType.compareTo("enclosure") == 0 ) {
                    String encType =
                        parser.getAttributeValue(null, "type");

                    if (encType.startsWith("image/")) {
                        String imageSrc =
                            parser.getAttributeValue(null, "href");
                        Log.i("Net",
                            "image source = " + imageSrc);
                    }
                }
            }
            break;
        }
    parserEvent = parser.next();
}
status.setText("Done...");
```

在创建了 URL 后，下一步则是从 XmlPullParserFactory 中获取一个 XmlPullParser 实例。Pull Parser 拥有一个主方法，它可以返回下一个事件。Pull Parser 返回的事件同 SAX 解析器处理类实现中使用的方法相似。代码以迭代的方式进行处理，但这一方法在移动应用上更为高效。

在这个例子中，我们唯一检测的事件为 START_TAG 事件，它指定了一个 XML 标记的开始。属性值可以被查询与比较。这个例子专注于查找 flickr 种子 XML 中的图像 URL。一旦找到，将会生成一条日志记录。

你可以侦测下列 XML Pull Parser 的事件。

- **START_TAG**：当找到一个新的标记时返回（即<tag>）。
- **TEXT**：当找到文本时返回（即在“<tag>text</tag>”中找到“text”）。
- **END_TAG**：当找到标记的结束时返回（即</tag>）。
- **END_DOCUMENT**：当到达 XML 文件的末尾时返回。

另外，解析器还可以用来验证输入。通常，无验证的解析是在内存环境受到约束的情况下发生的，例如在移动环境中。相容的（compliant）非验证的解析是这种 XML Pull Parser 使用的默认方式。

Download at http://www.pin5i.com/

12.2.4 异步操作

用户需要一个能够流畅运行的应用程序，所以诸如网络连接这样耗时的操作不应该阻塞主要的 UI 线程。目前使用的网络操作方式会造成 UI 线程的阻塞，直到网络操作完成为止。对于小型的任务，这或许可以接受，但是，当加入了超时机制或附加处理（例如解析 XML）之后，你就需要将这些耗时的操作从 UI 线程中移开。

Android SDK 提供两条用于将某些操作从主 UI 线程上分离开来的途径：AsyncTask 类和标准的 Thread 类。

AsyncTask 类是 Android 开发中所使用的一个特别的类，它封装了后台操作，同时在后台操作的整个生命周期内，帮助其与主 UI 线程的通信。另外，开发人员还可以使用标准的 Java 方法和类构建自己的线程解决方案，但这也就需要完全自主地管理整个线程生命周期。

12.2.5 使用 AsyncTask

AsyncTask 是一个抽象的辅助类，它可以用来管理后台操作，并最终返回到 UI 线程。它为异步操作创建了一个简单的接口，从而不必手工地创建 Java 线程类。

开发人员可以创建一个 AsyncTask 的子类并实现相应的事件方法，这与为后台处理创建线程并使用消息和消息处理机制更新 UI 不同。onPreExecute() 方法在后台操作开始前运行在 UI 线程上。doInBackground() 方法处理后台操作，同时 publishProgress() 方法周期性地通知 UI 线程有关后台操作的进度。当后台操作完成后，运行在 UI 线程上的 onPostExecute() 方法给出最终的更新。

下面的代码给出了 AsyncTask 的使用示例，它同之前使用 Thread 所实现的功能是一样的：

```
private class ImageLoader extends
    AsyncTask<URL, String, String> {

@Override
protected String doInBackground(
    URL... params)  {
    // 只有一个参数
    try {
        URL text = params[0];

        //... 解析代码 {

        publishProgress(
            "imgCount = " + curImageCount);

        // ... 解析代码结束 }

    }
    catch(Exception e) {
        Log.e("Net",
            "Failed in parsing XML", e);
        return "Finished with failure.";
    }

    return "Done...";
}
```

Download at http://www.pin5i.com/

```
protected void onCancelled() {
    Log.e("Net", "Async task Cancelled");
}

protected void onPostExecute(String result) {
    mStatus.setText(result);
}

protected void onPreExecute() {
    mStatus.setText("About to load URL");
}

protected void onProgressUpdate(
    String... values) {
    // 只使用了一个值
    mStatus.setText(value[0]);
    super.onProgressUpdate(values);
}}
```

当 AsyncTask.execute()方法运行后，doInBackground()方法在后台线程中运行，同时其他的方法运行在 UI 线程中。这样，就无需管理 Handler 或者提交 Runnable 对象了，从而简化了代码的编写和调试。

12.2.6 使用线程进行网络访问

下面的代码演示了如何运行一个新线程，它将建立起同远程服务器的连接，获取并解析 XML，并且向 UI 线程提交一个反馈来改变 TextView：

```
impot java.net.URL;

import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserFactory;

//...

new Thread() {
    public void run() {
        try {
            URL text = new URL(
            "http://api.flickr.com/services/feeds/photos_public.gne?
➥id=26648248@N04&lang=en-us&format=atom");

            XmlPullParserFactory parserCreator =
                XmlPullParserFactory.newInstance();
            XmlPullParser parser =
                parserCreator.newPullParser();

            parser.setInput(text.openStream(), null);

            mHandler.post(new Runnable() {
                public void run() {
                    status.setText("Parsing...");
                }
            });
```

Download at http://www.pin5i.com/

```
            int parserEvent = parser.getEventType();
            while (parserEvent !=
                XmlPullParser.END_DOCUMENT) {

                // 在此插入解析代码...

                parserEvent = parser.next();
            }

            mHandler.post(new Runnable() {
                public void run() {
                    status.setText("Done...");
                }
            });

        } catch (Exception e) {
            Log.e("Net", "Error in network call", e);
        }
    }
}.start();
```

在这个例子中，我们使用了一个匿名的 Thread 对象来完成任务。我们创建并立即调用了它的 start() 方法。然而，既然这段代码运行在另一个独立的线程中，那么有关用户界面更新的内容必须提交回主线程。这是通过主线程上的一个 Handler 对象和创建的 Runnable 对象实现的，Runnable 对象执行了对名为 status 的 TextView 控件的 setText() 方法的调用。

代码的其余部分与之前的例子相同。在单独的线程中执行解析代码和网络连接代码可以使用户界面以可响应的方式继续工作，让网络和解析操作在后台进行，这样可以带来流畅而友好的用户体验。同时，这也使得用户可以采取一些临时性的措施，比如中断数据传输。你可以实现 Thread 来监听某个特定的事件并且检查特定的标签。

12.2.7 显示从网络资源获取的图像

前面我们讲解了如何在单独的线程中解析 XML，下面我们稍微深入一点，通过一个例子来学习使用一些非原始的数据类型。

继续前面有关从 flickr 种子中解析图像地址的例子，我们来显示一些从种子中获取的图像。下面的示例读取图像信息并且将其在屏幕上显示出来，它演示了使用网络资源的另一种方式：

```
import java.io.InputStream;
import java.net.URL;

import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserFactory;
import android.os.Handler;

//...

final String imageSrc =
    parser.getAttributeValue(null, "href");
```

Download at http://www.pin5i.com/

```
final String currentTitle = new String(title);
imageThread.queueEvent(new Runnable() {
    public void run() {
        InputStream bmis;
        try {
            bmis = new URL(imageSrc).openStream();
            final Drawable image = new BitmapDrawable(
                BitmapFactory.decodeStream(bmis));
            mHandler.post(new Runnable() {
                public void run () {
                    imageSwitcher.setImageDrawable(image);
                    info.setText(currentTitle);
                }
            });
        } catch (Exception e) {
            Log.e("Net", "Failed to grab image", e);
        }
    }
});
```

你可以在之前描述过的解析器线程中找到上面的这段代码。在确定了图像源和标题之后，一个新的 Runnable 对象加入了队列，以在单独的线程中执行图像的处理。该线程仅仅是一个简单的队列，它接收匿名的 Runnable 对象，然后在上一个运行完成至少 10s 后执行它，这样实现了将种子中的图片进行幻灯片放映的效果。

Warning

虽然上面的代码对于本地资源和 URL 都可适用，但对于一些通过低速连接访问的资源来讲，或许不能正常工作。这在 Android SDK 中是一个已知的问题，在低速网络连接中载入较大尺寸的位图时会引起这样的缓冲问题。本章的代码中提供了一种相对直接的办法来解决上述问题。

如同第一个网络应用示例一样，它们都需要创建一个新的 URL 对象，并且从中提取出 InputStream。ImageSwitcher 需要关联一个 Drawable 对象。之后，我们调用了 BitmapFactory.decodeStream()方法，它需要一个 InputStream 作为参数。

最后，从这个运行在其他线程的 Runnable 对象开始，一定时间间隔以后，又一个匿名的 Runnable 对象被传回主线程，从而使用新的图像实际地修改 ImageSwitcher。图 12.1 显示了包含解码状态和当前图像的屏幕外观。

虽然当从 flickr.com 网站获得的种子被解码后，所有的这些都会持续地显示下去，但某些操作还是会显得比其他的缓慢。例如，当进行图片解码或屏幕绘制时，在解码的过程中还是能够感觉到明显的时延。当下的移动设备上应该不会出现这样的问题，因为之前的大多数设备对于应用程序都只提供一个可用的线程。想要向用户提供适度的平滑和即时的响应体验，这需要开发人员的精心设计。

图 12.1 显示了 flickr 网站图片和种子解码状态的屏幕

Download at http://www.pin5i.com/

12.2.8 获取 Android 网络状态

Android SDK 提供收集有关当前网络状态信息的功能。在使用一个网络资源之前对其是否可用进行测试是很有必要的。ConnectivityManager 类提供了许多实现该操作的方法。下面的代码检测了移动（蜂窝）网络是否可用并且进行了连接。除此，它同时还检测了 Wi-Fi 网络的连接情况：

```
import android.net.ConnectivityManager;
import android.net.NetworkInfo;

// ...

ConnectivityManager cm = (ConnectivityManager)
    getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo ni =
    cm.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
boolean isWifiAvail = ni.isAvailable();
boolean isWifiConn = ni.isConnected();
ni = cm.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
boolean isMobileAvail = ni.isAvailable();
boolean isMobilConn = ni.isConnected();

status.setText("Wifi\nAvail = " + isWifiAvail +
    "\nConn = " + isWifiConn +
    "\nMobile\nAvail = " + isMobileAvail +
    "\nConn = " + isMobileConn);
```

首先，对 getSystemService()方法的调用获取了 ConnectivityManager 对象实例，它由应用程序的 Context 提供，是其一部分。之后，该实例分别针对 TYPE_WIFI 和 TYPE_MOBILE（蜂窝网络）获取了两个 NetworkInfo 对象。可以通过这些对象的方法查询其连接的可用性，也可以查询到更细节的状态信息，以便知晓网络连接（或断开）所处的状态。图 12.2 显示了在模拟器上模拟了移动网络但没有 Wi-Fi 支持时的运行结果。

图 12.2 Android SDK 模拟器模拟的典型网络状态

Download at http://www.pin5i.com/

即使网络可用，也并不意味着网络资源所在的服务器同样可用。不过，ConnectivityManager 的 requestRouteToHost()方法可以检测服务器的可用性。这样，当网络出现问题时，应用程序可以给用户提供更多的反馈信息。

如果你的应用程序需要读取网络状态，需要显式地声明许可权限。在AndroidManifest.xml文件中可以加入下面的表达式：

```
<uses-permission
    android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Tip

可以使用模拟器的网络设置来模拟各种类型的蜂窝网络从 GSM 到 HSDPA 的数据传输速率（data rates）。除此之外，你还可以自行调整网络的延迟（latency）以尽量与蜂窝网络相似。虽然这种方法可以测试应用程序在所选网络类型下网络状态较好时的性能，但却不能模拟出真实的网络环境，例如当用户正在乘坐电梯，或者是火车行进中丢失了信号而正在获取，这时网络都没有很好地覆盖。只有在实际的物理设备上进行这方面的测试才能反映出真实的结果。

小　结

许多应用程序使用网络来增强和提升应用程序的功能和质量。然而，用户的网络连接状况却不是时刻能够保证的。应用程序开发人员需要仔细地进行设计和实现，以便能够为用户提供一个稳定而流畅的应用程序。在应用程序中集成网络功能需要在设计时就加以考虑。决定应用程序是否支持网络连接、多大程度上支持也是应用程序设计过程中的重要环节，我们将在第 26 章“移动软件开发过程”中讲解更多的有关内容。

参考资料和更多信息

Java.net package：http://developer.android.com/reference/java/net/package-summary.html.

Android.net package：http://developer.android.com/reference/android/net/package-summary.html.

XML Pull Parsing：http://www.xmlpull.org/.

Android XML Pull Parser:http://developer.android.com/reference/org/xmlpull/v1/XmlPullParser.html.

Download at http://www.pin5i.com/

第 13 章

使用 Android Web API

移动开发人员经常依赖 Web 技术来增强他们的应用程序：提供最新的内容、集成诸如社交网络这样的流行 Web 服务等。Android 应用程序可以通过各种途径来利用因特网资源，比如在应用程序中包含一个特殊的 WebView 控件，从而提供浏览器的功能，还可以使用标准的 WebKit 库来扩展基于 Web 的各种功能。稍微新一点的 Android 设备还提供运行 Flash 应用程序的能力。在本章中，我们将讨论 Android 平台所提供的 Web 技术。

13.1 使用 WebView 浏览 Web

从 Web 获取数据的应用程序通常最终都会将这些数据显示在屏幕上。Android 应用程序可以使用 WebView 控件来完成这项工作，而不必使用各种不同的控件来实现自定义的各种界面。你可以把 WebView 控件看作是一个类似于浏览器的视图。

WebView 控件利用 WebKit 渲染（rendering）引擎将 HTML 数据绘制在屏幕上。它所使用的数据既可以是 Web 上的 HTML 页，也可以来源于本地。WebKit 是一个开源的浏览器引擎，你可以在其官方网站上阅读到更多的信息：http://webkit.org。

Tip

这一部分中的许多代码示例都来自于 SimpleWeb 应用程序。在本书的网站上可以下载 SimpleWeb 应用程序的源代码。

使用 WebView 控件需要 android.permission.INTERNET 许可权限。你可以在应用程序的 Android Manifest 文件中加入以下代码：

```
<uses-permission android:name="android.permission.INTERNET" />
```

在考虑 WebView 控件是否是你应用程序的最佳选择时，你需要意识到在你的应用程序中，可以随时通过 Intent 运行“浏览器（Browser）”应用程序。如果你想为用户访问网络提供完整支持（比如书签）的话，那么最好还是运行浏览器应用程序，导航到特定的站点，然后让用户完成浏览，最后返回到你的应用程序中，如下所示：

```
Uri uriUrl = Uri.parse("http://androidbook.blogspot.com/");
Intent launchBrowser = new Intent(Intent.ACTION_VIEW, uriUrl);
startActivity(launchBrowser);
```

通过 Intent 运行浏览器并不需要任何特别的许可权限。也就是说，你的应用程序不需要拥有 android.permission.INTERNET 权限。除此之外，由于用户从你应用程序的当前

Download at http://www.pin5i.com/

Activity 转移到浏览器应用程序的 Activity，再使用“后退（Back）”键返回的整个过程非常流畅，所以这与在你自己的应用程序中实现一个带有 WebView 的 Activity 类没有太大的区别。

13.1.1 设计带有 WebView 控件的布局

WebView 控件可以像其他视图一样被嵌入到布局资源文件中，它既可以占据整个屏幕，也可以只占一部分。下面是在布局资源中定义 WebView 的典型示例：

```
<WebView
    android:id="@+id/web_holder"
    android:layout_height="wrap_content"
    android:layout_width="fill_parent"
/>
```

通常而言，你应该给 WebView 控件留出足够的空间来显示文字和图像，在设计带有 WebView 控件的布局时始终记住这一点。

Warning

Eclipse 布局资源编辑器并不能正确地显示 WebView 控件。你需要通过 Android 模拟器或者真实设备来检查布局显示的正确性。

13.1.2 向 WebView 控件载入内容

你可以使用许多方法载入 WebView 控件的内容。比如，WebView 控件可以载入某个站点，或者渲染原始的 HTML 代码。Web 页面既可以存储在远程 Web 服务器上，也可以存储在本地设备中。

下面是使用 WebView 控件载入特定 Web 站点的示例：

```
final WebView wv = (WebView) findViewById(R.id.web_holder);
wv.loadUrl("http://www.perlgurl.org/");
```

你不需要再编写任何其他的代码来实现相关页面的显示。类似地，你可以载入存储在应用程序组件目录下的名为 webby.html 的 HTML 文件，如下所示：

```
wv.loadUrl("file:///android_asset/webby.html");
```

如果你想要渲染原始 HTML 的话，可以使用 loadData() 方法：

```
String strPageTitle = "The Last Words of Oscar Wilde";
String strPageContent = "<h1>" + strPageTitle +
    ": </h1>\"Either that wallpaper goes, or I do.\"";
String myHTML = "<html><title>" + strPageTitle
    +"</title><body>"+  strPageContent  +"</body>
</html>";
wv.loadData(myHTML, "text/html", "utf-8");
```

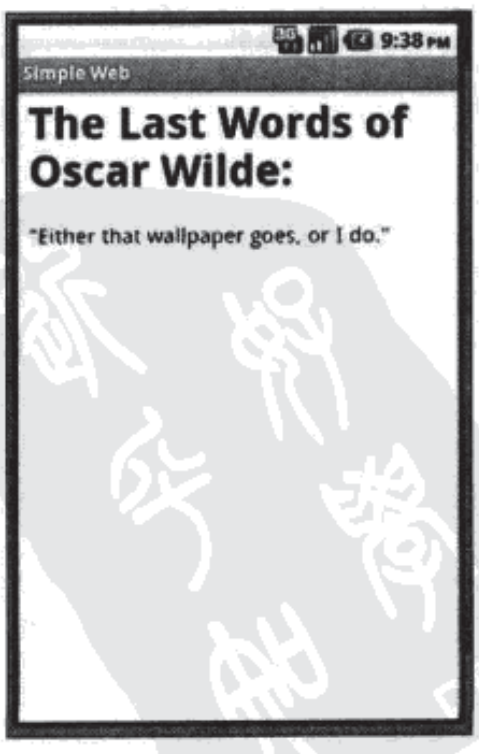


图 13.1 用来显示 HTML 的 WebView 控件

图 13.1 显示了 WebView 控件的运行结果。

不幸的是，并不是所有的网站都是为移动设备而设计的。不过，你可以很方便地调整 Web 内容的尺寸比例，以便能够适应 WebView 控件的大小。这可以通过设置控件的初始缩放比例（scale）来实现，如下所示：

Download at http://www.pin5i.com/

```
wv.setInitialScale(30);
```

对 setInitialScale()方法的调用将页面尺寸缩小到了原始的 30%。对于那些使用绝对尺寸定义的页面，需要缩小它们以能够在屏幕上看见网页的全貌。但这样一些文本可能会变得太小以至于不能阅读，所以必要的测试或者适当修改页面的设计（如果你可以控制的话）可以增强用户体验。

Tip

如果你想全屏显示 WebView 控件，那么可以通过代码创建一个 WebView，然后将其传入 setContentView()方法中即可，setContentView()方法会在 Activity 的 onCreate()方法中调用。

13.1.3 为 WebView 控件添加功能

你也许注意到了 WebView 控件并不具有浏览器的全部功能。比如，它并不显示网页的标题，也不提供重新载入页面的按钮。事实上，如果用户在 WebView 控件中显示的页面上单击链接后，并不会向视图内载入新的页面。相反，它会触发浏览器应用程序。

默认情况下，所有的 WebView 控件均使用其内部渲染引擎 WebKit 来显示开发人员提供的 Web 内容，但是有许多途径可以用来增强 WebView 控件的功能。你可以使用以下 3 个类来辅助修改控件的行为：WebSettings 类、WebViewClient 类和 WebChromeClient 类。

1. 使用 WebSettings 修改 WebView 的设置

WebView 控件有许多默认的设置：不带有缩放控件、不允许 JavaScript、默认的字体大小、User-agent 字符串等。你可以使用 getSettings()方法来改变 WebView 控件的设置。getSettings()方法将返回一个 WebSettings 对象，它可以用来对 WebView 进行配置。下面是一些有用的设置。

① 使用setSupportZoom()和setBuiltInZoomControls()方法开启或关闭缩放控件。
② 使用 setJavaScriptEnabled()方法开启或关闭 JavaScript 支持。
③ 使用 setLightTouchEnabled()方法开启或关闭鼠标悬停（mouseover）。
④ 配置字体、文本尺寸和其他的显示特性。

你还可以使用 WebSettings 类来配置 WebView 的插件，以及设置多窗口特性等。

2. 使用 WebViewClient 处理 WebView 事件

WebViewClient 类允许应用程序监听特定的 WebView 事件，如页面载入、表单提交和载入新的 URL 等。你还可以使用 WebViewClient 类来处理页面载入过程中产生的错误。你可以使用 setWebViewClient()方法向 WebView 绑定一个有效的 WebViewClient 对象。

下面是一个处理 onPageFinished()方法的示例，它可以在页面载入完毕后将页面标题显示在屏幕上：

```
WebViewClient webClient = new WebViewClient() {

    public void onPageFinished(WebView view, String url) {
        super.onPageFinished(view, url);
        String title = wv.getTitle();
```

Download at http://www.pin5i.com/

```
            pageTitle.setText(title);
        }};

    wv.setWebViewClient(webClient);
```

当页面载入完成后，onPageFinished()方法会被调用，随后它调用了 WebView 的 getTitle()方法以获取网页的标题。代码运行的结果如图 13.2 所示。

3. 使用 WebChromeClient 添加 Browser Chrome

你可以使用 WebChromeClient 来修改浏览器的行为，方法与使用 WebViewClient 类似。不过，WebChromeClient 是用于那些绘制在页面内容之外的对象，一般被称作“browser chrome”。它也包含一些针对特定 JavaScript 调用的回调，例如 onJsBeforeUnload()，以确认从该页面跳出的导航。你可以使用 setWebChrome Client()方法将一个有效的 WebChromeClient 对象与 WebView 相绑定。

下面的代码演示了如何使用 WebView 的特性来同用户进行交互。我们在 WebView 的下方添加了一个 EditText 和 Button 控件，Button 的事件处理实现如下所示：

```
Button go = (Button) findViewById(R.id.go_button);
go.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        wv.loadUrl(et.getText().toString());
    }
});
```

如代码所示，仅仅需要再次调用 loadUrl()方法就可以使 WebView 控件下载显示另一个 HTML 页面，如图 13.3 所示。这样，一个普通的 Web 浏览器就实现了，它可以嵌入到任何应用程序中，不过它一般需要受到某种限制，以使用户仅仅可以访问必要的 HTML 文档。

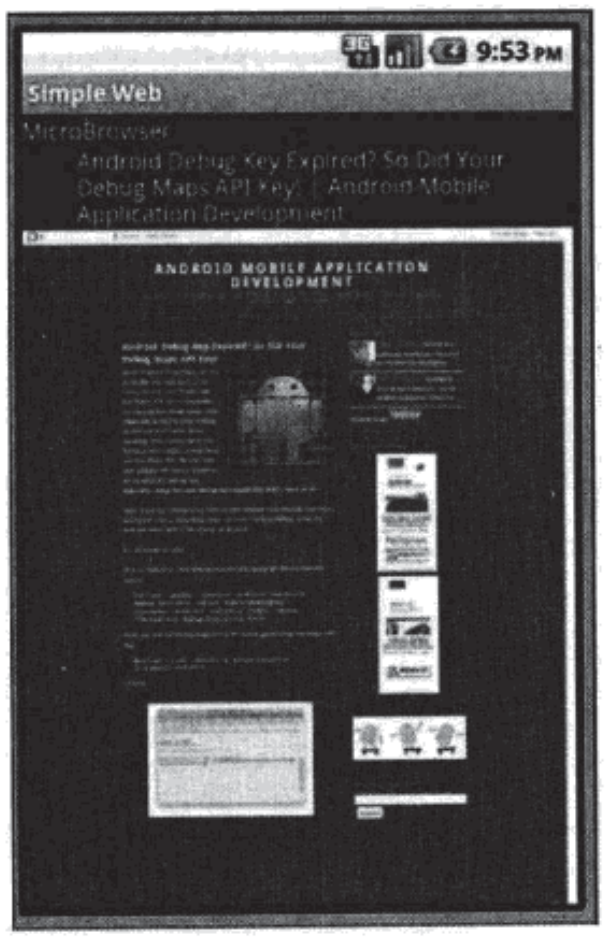


图 13.2 带有标题显示功能的 WebView 控件

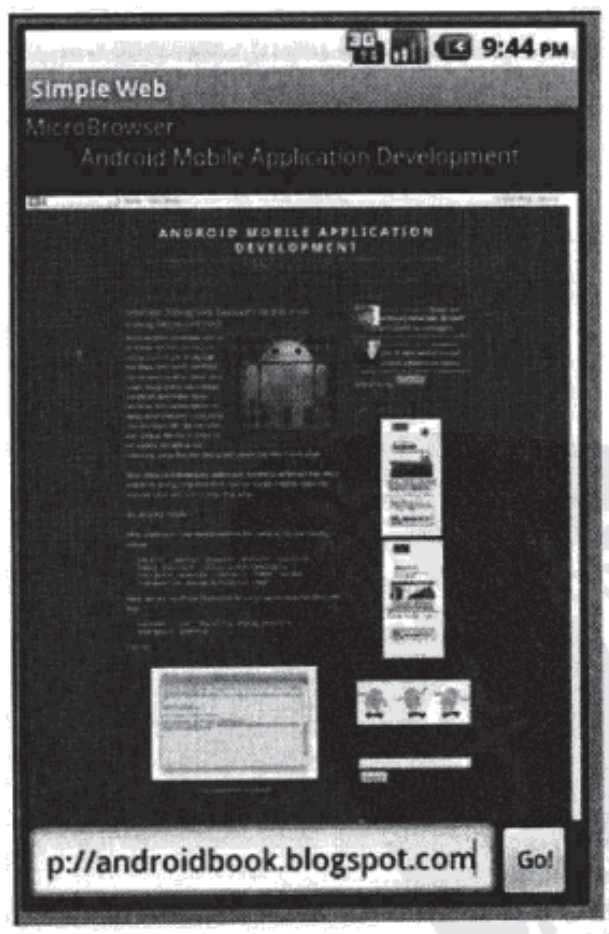


图 13.3 带有 EditText 的 WebView，它允许用户输入任意 URL

使用 WebChromeClient 可以在屏幕上添加一些典型的 Chrome。例如，你可以用它来

Download at http://www.pin5i.com/

监听页面标题的改变、各种 JavaScript 请求甚至面向开发人员的控制台消息等。

```
WebChromeClient webChrome = new WebChromeClient() {
    @Override
    public void onReceivedTitle
        (WebView view, String title) {
        Log.v(DEBUG_TAG, "Got new title");
        super.onReceivedTitle(view, title);
        pageTitle.setText(title);
    }
};
wv.setWebChromeClient(webChrome);
```

在上面的代码中，我们覆写了 WebChromeClient，使得它可以监听到页面标题的改变。随后，这一标题被设置给了屏幕上一个可见的 TextView。

无论你是使用 WebView 来显示应用程序的主用户界面，还是仅仅用于帮助页的显示，在很多情况下，同完全自己设计屏幕相比，它都可以帮助你节省编写代码的时间。使用开源引擎（WebKit、WebView）的力量，可以为你的应用程序提供标准而强大的 HTML 浏览器。WebKit 的应用非常广泛，它被用在了许多桌面浏览器中，包括 Apple Safari 和 Google Chrome，各种移动操作系统也使用它作为浏览器，如 Apple iOS、Nokia、Palm WebOS 和 BlackBerry 手机，还包括其他的一些平台，如 Adobe AIR。

13.2 使用 WebKit 延伸 Web 应用

Android 平台上的所有 HTML 渲染均使用 WebKit 渲染引擎。android.webkit 包提供了一系列使用强大的 WebView 控件浏览因特网的 API。你应该熟悉 WebKit 的各种类和接口，以便能够在应用程序中增强 WebView 的用户体验。

Android 自带的“浏览器”应用程序不提供任何类或接口（虽然你可以使用内容提供器访问浏览器应用程序的数据）。相反，在应用程序中，你必须使用一些类和接口去控制 WebView 控件的浏览行为。

13.2.1 浏览 WebKit API

以下是 android.webkit 包提供的一些非常有用的类。

① CacheManager 类用于控制 WebView 的缓存（cache）项。

② ConsoleMessage 类可以用来获取 WebView 输出的 JavaScript 控制台。

③ CookieManager 类用于设定和获取 WebView 的用户 Cookie。

④ URLUtil 类提供验证各类 Web 地址的有效性的功能。

⑤ WebBackForwardList 和 WebHistoryItem 类可以用来查询 WebView 的浏览历史。

下面我们就来看看如何使用上面这些类来增强 WebView。

13.2.2 扩展 Android Web 应用程序的功能

我们使用一些目前已经讨论过的 WebKit 功能，并且通过一个示例来讲解后面的内容。对于一个移动开发人员而言，把应用程序设计成一个 Web 应用程序是很普遍而可以理解的，因为这样可以获得使用不同平台的用户，同时最大程度地避免了开发和管理一些平台相关的

Download at http://www.pin5i.com/

代码。不过，不足之处在于，Web 应用程序不能访问本地平台提供的代码，也不能使用一些本地的应用程序（比如那些使用 Java 开发的 Android 应用程序）所提供的功能，如访问内置摄像头或者访问 Android 底层特性。

开发人员可以通过设计一个轻量级的 Java Shell 应用程序，并使用 WebView 控件作为访问 Web 应用程序内容的入口，来增强 Web 应用程序的功能。Web 应用程序与本地 Java 应用程序之间的通信可以使用诸如 JavaScript 这样的脚本语言的实现。

Tip

这一部分中的许多代码示例都来自于 SimpleWebExtension 应用程序。在本书的网站上可以下载该应用程序的源代码。

让我们来创建一个简单的 Android 应用程序，它将展示 Web 内容和本地 Android 代码之间的通信。这个例子需要你了解一定的 JavaScript 知识。

你可以按照下面的步骤来创建这个应用程序。

① 创建一个新的 Android 应用程序。

② 创建一个带有名为 html_viewer 的 WebView 控件和一个名为 call_js 的 Button 控件的布局。将 Button 控件的 onClick 属性设定为一个名为 setHTMLText 的方法。

③ 在应用程序 Activity 的 onCreate() 方法中，使用 findViewById() 方法获取 WebView 控件。

④ 获取 WebSettings 并且调用 setJavaScriptEnabled() 方法来开启 WebView 的 JavaScript 功能。

⑤ 创建一个 WebChromeClient 对象，并且实现它的 onConsoleMessage() 方法，以便能够监视 JavaScript 控制台消息。

⑥ 使用 setWebChromeClient() 方法将 WebChromeClient 对象添加到 WebView 中。

⑦ 通过调用 WebView 控件的 addJavascriptInterface() 方法以允许 JavaScript 接口控制你的应用程序。你需要定义 JavaScript 接口具体能够控制的功能，以及调用可用的命名空间。这里，我们允许 JavaScript 初始化 Toast 消息。

⑧ 使用某个标准方法（如 loadUrl() 方法）向 WebView 控件载入内容。在这里，我们载入一个在应用程序包中定义好的 HTML 组件。

如果你完成了上面的步骤，那么 Activity 的 onCreate() 方法应该大致如下所示：

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    final WebView wv = (WebView) findViewById(R.id.html_viewer);
    WebSettings settings = wv.getSettings();
    settings.setJavaScriptEnabled(true);
    WebChromeClient webChrome = new WebChromeClient() {
        @Override
        public boolean onConsoleMessage(ConsoleMessage consoleMessage) {
            Log.v(DEBUG_TAG, consoleMessage.lineNumber()
                + ": " + consoleMessage.message());
            return true;
        }
    };
```

Download at http://www.pin5i.com/

```
        wv.setWebChromeClient(webChrome);
        wv.addJavascriptInterface(new JavaScriptExtensions(), "jse");
        wv.loadUrl("file:///android_asset/sample.html");
    }
```

这样，一个 WebChromeClient 类就设置好了，像往常一样使用调试标记就可以方便地跟踪应用程序 Log 的输出了，所有的 JavaScript console.log 消息都会显示在 LogCat 的输出中。上面的代码为 JavaScript 接口定义了一个名为 jse 的命名空间（当然，命名空间的名字你可以随意取），这样，所有通过 JavaScript 对这个 Java 类的调用都需要使用 jse.命名空间作为开头，后面接上适当的公有方法，例如：jse.javaMethod()。

接下来你可以在 Activity 内定义 JavaScriptExtensions 子类，它只有一个方法，这个方法会触发 Android 的 Toast 消息：

```
class JavaScriptExtensions {
    public static final int TOAST_LONG = Toast.LENGTH_LONG;
    public static final int TOAST_SHORT = Toast.LENGTH_SHORT;
    public void toast(String message, int length) {
        Toast.makeText(SimpleWebExtension.this, message, length).show();
    }
}
```

JavaScript 代码可以访问 JavaScriptExtensions 类中的所有东西，包括成员变量和方法，当然，返回值也一样有效。

现在，我们把目光转移到向 WebView 控件载入 Web 页面上来。在这个例子中，只需要在应用程序的/assets 目录下创建一个名为 sample.html 的文件就可以了。下面是 sample.html 文件的内容：

```
<html>
<head>
<script type="text/javascript">

function doToast() {
    jse.toast("'"+document.getElementById('form_text').value +
        "' -From Java!", jse.TOAST_LONG);
}

function doConsoleLog() {
    console.log("Console logging.");
}

function doAlert() {
    alert("This is an alert.");
}

function doSetFormText(update) {
    document.getElementById('form_text').value = update;
}

</script>
</head>
<body>
<h2>This is a test.</h2>
<input type="text" id="form_text" value="Enter something here..." />
```

Download at http://www.pin5i.com/

```
<input type="button" value="Toast" onclick="doToast();" /><br />
<input type="button" value="Log" onclick="doConsoleLog();" /><br />
<input type="button" value="Alert" onclick="doAlert();" />
</body>
</html>
```

上面的示例文件定义了 4 个 JavaScript 函数，并且在 WebView 中显示一个表单。

① doToast()函数使用之前在 addJavaScriptInterface()方法中定义的 jse 对象访问 Android 应用程序。addJavaScriptInterface()方法可以看作是以 JavaScript 编写的 JavaScriptExtensions 类。如果 doToast()函数有返回值，那么我们可以将其指派给一个变量。

② doConsoleLog()函数写入 JavaScript 控制台日志，控制台由 WebChromeClient 的 onConsoleMessage()回调获得。

③ doAlert()函数显示了如何在 WebView 控件内部通过运行一个对话框显示 Alert 消息。如果你想要替换 Alert 的外观，那么可以覆写 WebChromeClient.onJSAlert()方法。

④ doSetFormText()函数展示了本地 Java 代码如何同 JavaScript 接口进行通信，并且向 Web 应用程序提供数据。

最后，为了展示 Java 调用 JavaScript，你需要为 Activity 类中的 Button 控件定义一个单击事件处理。这里，名为 setHTMLText()的 onClick 处理器将通过调用 JavaScript 函数 doSetFormText()在当前载入的页面中执行 JavaScript 代码，doSetFormText()在之前的 Web 页面代码中已经定义过了。下面是 setHTMLText()方法的实现：

```
public void setHTMLText(View view) {
    WebView wv = (WebView) findViewById(R.id.html_viewer);
    wv.loadUrl("javascript:doSetFormText('Java->JS call');");
}
```

上面在当前载入页面上对 JavaScript 的调用不允许返回值。但是，有几种途径可以实现对返回值的访问，这需要修改一定的架构设计，通常实现为异步调用，并且通过实现其他方法来获得响应。

Warning

需要注意的是使用 addJavascriptInterface()方法实现 Android 应用程序对 JavaScript 的控制的安全性。确保 WebView 只会载入你所指定的内容，而非互联网上的其他内容。另外，JavaScript 接口并不运行在 UI 线程中，所以你需要引入简单的进程间通信技术，如使用 handler 向其他线程反馈消息等。

图 13.4 显示了该应用程序在 Android 设备上运行的情况。

这种风格的开发被广泛地应用在了开源项目 PhoneGap 中，它致力于向各种平台的本地代码提供一系列的标准 JavaScript 接口，包括 iOS、Android、BlackBerry、Symbian 和 Palm。访问下面的网址可以获得更多有关 PhoneGap 的信息：http://phonegap.com。

图 13.4 使用 JavaScript 接口的简单 Android 应用程序

Download at http://www.pin5i.com/

13.3 使用 Flash

对于一些需要在移动应用中使用 Flash 应用程序的开发人员来讲，Android 是目前唯一支持桌面 Flash 10.1（相较于 Flash Lite 而言，它的功能较为有限）的智能手机操作系统。不过，在 Android 平台中引入 Flash 技术是一把双刃剑。我们来看看一些事实观点。

① Flash 或许不是“未来”，但它确实是时下很流行的 Web 应用。在互联网上有千百万 Flash 应用程序和网站，它们都可以使用 Android 设备进行访问。这很大程度上满足了用户的需求。

② 本地的 Android 应用程序始终会运行得更好，而且占用更小的资源（比如更小的电池消耗），同时提供了更紧密的平台交互性和兼容性，并且比 Flash 应用程序支持更多的 Android 设备。

③ 是否决定在 Android 平台下使用 Flash 应用程序而非本地 Java 应用程序是在设计时需要重点考量的。因为运行效率、安全性和设备兼容性（包括向下兼容）都是使用 Flash 时需要权衡的。

④ 你不能期望所有的 Flash 应用程序都可以被完美地载入然后运行，因为它们不一定完全遵守移动应用程序的一些约束和 UI 规范，比如小屏幕上的触屏支持、较慢的处理器以及各种中断事件（如电话拨入）等。

尽管如此，仍然有千百万可供我们使用的 Flash 应用程序。接下来，我们来看看如何在 Android 平台下使用 Flash 应用程序。

13.3.1 使用 Flash 应用程序

Android 2.2 及以上版本的 Android 设备可以运行 Flash 应用程序（目前是 Flash 10.1）。为了运行 Flash，Android 设备上必须安装 Adobe 的 Flash Player for Android。

用户可以从 Android Market 上下载 Adobe 的 Flash Player for Android 应用程序。当然，Android 设备也有可能预装了 Adobe 的应用程序。不过需要注意，只有一些性能强大的 Android 设备可以流畅地运行 Flash，从而提供满意的用户体验。Flash Player 安装后，它类似一个典型的浏览器插件（如图 13.5 所示）。用户可以开启或关闭它，你也可以使用 `WebView` 控件改

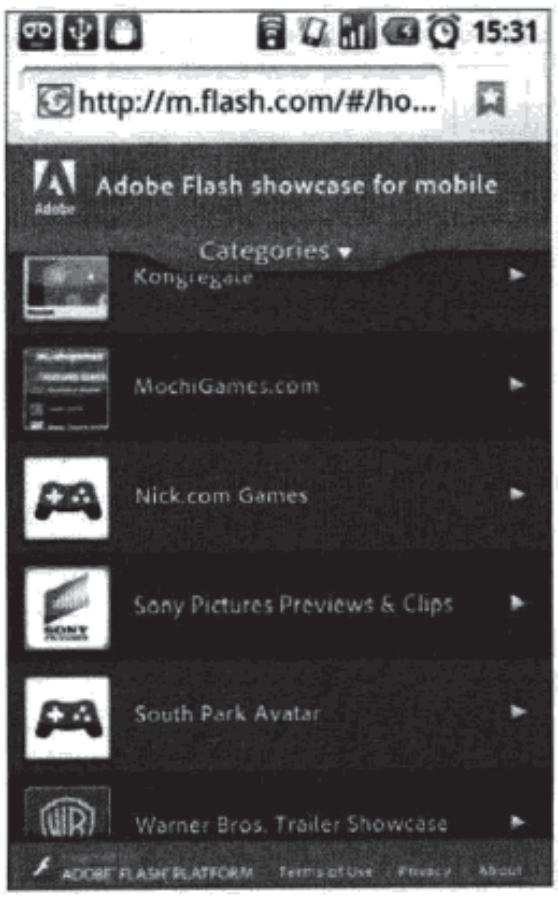


图 13.5 Nexus One 下运行的显示了许多移动 Flash 应用程序的 Flash 应用程序

Download at http://www.pin5i.com/

变 Flash 插件的可用性。

13.3.2 构建 AIR 应用程序

Adobe 使用运行 ActionScript 3 脚本的 AIR 工具包创建了许多用于开发跨平台应用程序的工具，ActionScript 3 是 Adobe 的 Web 脚本语言，它用于 Web 和 Flash 应用程序中。Adobe 公司最近发布了 Adobe AIR for Android，它使得开发人员可以创建 AIR 应用程序，然后可以编译成本地的 Android APK 文件，之后 APK 文件可以像其他 Android 应用程序一样进行发布。开发人员可以使用带有 AIR 应用程序开发扩展包的 Adobe Flash Professional CS5 工具进行开发，而后编译成 Android 包文件，像本地 Android 应用程序一样进行部署。

Note

在本书成稿之时，Adobe AIR for Android 进行了预发布。开发人员可以访问以下网址：http://labs.adobe.com/technologies/air2/android/。

小　结

Android 开发人员可以使用强大的 `WebView` 控件为应用程序增添浏览器的功能。开发人员还可以使用强大的 WebKit 等技术进一步增强应用程序的 Web 功能。在 Android 2.2 中，Flash 以 Adobe 应用程序的形式被引入到了 Android 平台中。Adobe 还开发了能够将 ActionScript 应用程序编译成 Android APK 文件的工具套件，从而使得它们可以像本地 Android 应用程序一样进行部署。

参考资料和更多信息

WebKit Open Source Project（WebKit 开源项目）：http://www.webkit.org.

W3School's JavaScript Tutorial（W3School JavaScript 指南）：http://www.w3schools.com/js/js_intro.asp.

Adobe AIR Tool Suite（Adobe AIR 工具套件）：http://www.adobe.com/products/air/tools/.

Download at http://www.pin5i.com/

第 14 章

使用位置服务 API

无论是出于安全性考虑还是仅仅因为方便，手机普遍引入了位置服务（Location-Based Services，LBS）。就其本身来看，将位置信息、导航支持和地图服务加入到你的应用程序中将会大大扩展其功能应用。

在本章中，你将学习如何使用 Android SDK 提供的位置服务，如何利用特定的硬件设备来判断手机所处的位置，这种设备包括内置的全球定位系统（Global Positioning System，GPS）等。你还将学习如何将原始的左边点信息转换成具有描述性的位置名称，也包括如何反过来转换。最后，我们将探索使用地图进行位置标记的几个不同方法。

14.1 使用全球定位服务（GPS）

AndroidS DK 提供通过内置 GPS 设备访问位置信息的几条途径，当然前提是这种设备存在且可用。一般来说，Android 手机都会配备 LBS 功能，移动电话的位置信息可以用于协助紧急救援。不过，不是所有的 Android 设备都是手机，也不是所有手机都带有用户可用的 LBS。如果 GPS 功能被禁用了，或是 Android 设备未配有 LBS 硬件，Android SDK 提供了另外的 API 用以决定使用替代的位置服务提供商。这些途径既具有自己的一些优点，但同时也有一定的缺点，表现在耗电量、速度和数据精度上。

Tip

本章提供的大部分代码都来自于应用程序 SimpleLocation。该程序的源代码可以在本书的网站上下载。

14.1.1 在你的应用程序中使用 GPS 功能

LBS 及其相关硬件（如内置的精确 GPS）都是 Android 设备的可选功能。除了适当的许可权限以外，你还可以在 Android Manifest 文件中指定应用程序所需的具体可选功能。可以使用 Android Manifest 文件中的<uses-feature>标记来声明应用程序所需的或所使用的具体 LBS。虽然这个标记并不是 Android 操作系统强制要求的，但它可以为诸如 Android Market 这样的应用程序商店提供筛选服务，以便用户能够精确地进行选择。如果你的应用程序只有在具有位置识别功能的设备上才能运行的话，那么你可以在应用程序的 Manifest 文件中添加如下<uses-feature>标记：

```
<uses-feature android:name="android.hardware.location" />
```

Download at http://www.pin5i.com/

而如果你的应用程序需要精确的定位功能（也就是说设备上具有可用的 GPS 硬件，而不仅仅是基于基站信号的定位），那么你可以使用下面的<uses-feature>标记：

```
<uses-feature android:name="android.hardware.location.gps" />
```

Note

<uses-feature>标记的新参数已经添加到了最新发布的 Android SDK 中。例如我们上面提到的参数 android.hardware.location 就添加到了 Android 2.2（API Level 8）中。

14.1.2 查找你的位置

要确定设备所在的位置，你需要执行几步操作并且做出一些选择。下面列出了这一过程中的重要步骤。

① 使用 LOCATION_SERVICE 参数调用 getSystemService()方法，获取一个 LocationManager 实例。

② 在 AndroidManifest.xml 文件中加入适当的许可权限，这取决于应用程序所需获取的位置信息类型。

③ 使用 getAllProviders()方法或 getBestProvider()方法选择一个服务提供方（provider）。

④ 实现 LocationListener 类。

⑤ 使用所选的服务提供方和 LocationListener 对象，通过调用 requestLocationUpdates()方法启动位置信息的接收。

在获取 LocationManager 对象时并不需要特殊的许可权限，许可权限用于选择可用的服务提供方。下面的代码获取了一个 LocationManager 对象的实例：

```
import android.location.*;
...
LocationManager location =
    (LocationManager)getSystemService(Context.LOCATION_SERVICE);
```

在加入到 AndroidManifest.xml 许可文件后，下面的 XML 代码段分别向应用程序提供粗略和精确的位置服务许可：

```
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

现在我们的应用程序已经拥有了使用位置信息的许可权限，而且拥有了可用的 LocationManager，下面就需要确定所使用的位置信息服务提供方了。下面的代码配置了一个 Criteria 对象，并且基于其中的信息请求了一个服务提供方：

```
Criteria criteria = new Criteria();
criteria.setAccuracy(Criteria.NO_REQUIREMENT);
criteria.setPowerRequirement(Criteria.NO_REQUIREMENT);

String.setProvider = location.getBestProvider(criteria, true);
```

setAccuracy()方法可以传入 ACCURACY_COARSE 和 ACCURACY_FINE 常量值作为参

Download at http://www.pin5i.com/

数，它们可以用来（结合适当的许可权限）请求一个应用程序有权访问的服务提供方。你可以用 setPowerRequirment()方法来寻找一个符合特定电源消耗要求的提供方，例如 POWER_HIGH 或 POWER_LOW。Criteria 对象还能够允许我们指定所选的提供方是否向用户收取费用，是否需要海拔高度信息，以及其他的细节。如果应用程序拥有某些特定的需求，那么可以在这里进行设定。但是，进行这些 Criteria 的设置并不意味着提供方能够满足这些条件。所以可能需要一些更为灵活的设置以扩大用户的设备选择范围。传入 getBestProvider()方法的一个 Boolean 参数使得应用程序只请求可用的服务提供方。

通过由 getBestProvider()方法返回的服务提供方，应用程序可以请求位置信息。在此之前，应用程序需要提供 LocationListener 类的实现。LocationListener 对象包含四个方法：有两个用来告诉应用程序服务是否可用；一个提供服务提供方的状态（例如 GPS 接收器能够发现的可用卫星数量）；最后一个用来为应用程序提供位置信息。下面实现最后一个方法的示例，即 onLocationChanged()方法：

```
public void onLocationChanged(Location location) {
    String locInfo = String.
        format("Current loc = (%f, %f) @ (%f meters up)",
        location.getLatitude(), location.getLongtitude(),
        location.getAltitude() );
    if (lastLocation != null) {
        float distance = location.distanceTo(lastLocation);
        locInfo += String.
            format("\n Distance from last = %f meters", distance);
    }
    lastLocation = location;
    status.setText(locInfo);
}
```

onLocationChanged()方法接收一个 Location 对象，它是从所选服务提供方获取的最新位置信息。在这个例子中，应用程序仅仅将获得的位置信息打印了出来，包括海拔高度，这一信息不是所有的提供方都会提供。之后，代码使用的 Location 对象的功能函数 distanceTo()计算出了从上次调用 onLocationChanged()方法后所移动的距离。

如何使用所获取的位置信息取决于应用程序自身。应用程序可能想要将这些位置信息转换成一个地址，并且显示在某个嵌入的地图上，或者运行内置的地图应用程序以此为中心点显示周边地图。

Tip

在使用某些 LBS 服务之前，你可以使用 AVD 将目标设备的 Android SDK 和 Google API 都配置出来，使用 Google API 将一些应用程序（例如 Maps）载入模拟器中。但一般来说，LBS 应用的设计和实现应该最好在真实设备上完成。

14.1.3 定位你的模拟器

Android 模拟器可以模拟位置服务，但是和你想的一样，它并不会通过某个“底层硬件”来获得可用的卫星支持。Android SDK 提供一种模拟位置数据的方法，这是通过使用 GPX 或 KML 文件提供一个单一位置点来实现的。这仅仅适用于模拟器，而非物理设备，但它可以用

Download at http://www.pin5i.com/

来对你的位置服务应用程序进行测试。

对于更多有关模拟器的内容，可以参考附录 A“Android 模拟器快速入门”。

14.2 位置地理编码

确定经纬度对于精确定位、跟踪和测量都非常有用，但对于用户而言缺乏直观性。Android SDK 提供了一些辅助类来将原始的位置数据转换为直观的地理名称，这一过程也可以反过来进行，即将地理名称或地址转换成原始的位置坐标。

Warning

根据 Android 文档，针对 Google API 的 AVD 允许用户使用“Google Experience”在模拟器上进行测试。Google API 提供了使用 Google Maps 的功能，同时提供了位置解码（geocoder）服务。虽然没有写在具体的文档中，但是我们需要注意不是所有的 AVD 的 API Level 都支持这些位置解码服务。例如，针对 API Level 6 具有 Google API 的 AVD 提供了对位置解码的服务，但 API Level 7 和 Level 8 下的 AVD 就不提供这一服务（在本书成稿时）。当你使用不具有后台（back-end）位置解码服务的 AVD 时，会得到一个不具有后台服务的异常。本章中的代码最好在针对 API Level 6 的 AVD 下运行，当然还需要 Google API 的支持，或者也可以在一个具有位置解码后台服务的真实设备上运行。

Geocoder 对象可以在没有任何许可权限的条件下使用。下面这段代码演示了使用 Geocoder 对象来获得位置名称，它使用了前面传入 LocationListener 的 onLocationChanged() 方法中的 Location 对象：

```
Geocoder coder = new Geocoder(this);
try {
    Iterator<Address> addresses = coder
        .getFromLocation(location.getLatitude(),
        location.getLongtitude(), 3).iterator();
    if (addresses != null) {
        while (addresses.hasNext()) {
            Address nameLoc = addresses.next();
            String placeName = namedLoc.getLocality();
            String featureName = namedLoc.getFeatureName();
            String country = namedLoc.getCountryName();
            String road = namedLoc.getThoroughfare();
            locInfo += String.format("\n[%s][%s][%s][%s]",
                placeName, featureName, road, country);
            int addIdx = namedLoc.getMaxAddressLineIndex();
            while (addIdx >= 0 ) {
                String addLine = namedLoc.getAddressLine(addIdx);
                locInfo += String.
                    format("\nLine %d: %s", addIdx, addLine);
                addIdx-;
            }
        }
    }
} catch (IOException e) {
    Log.e("GPS", "Failed to get address", e);
}
```

Download at http://www.pin5i.com/

有两种方法可以从 getFromLocation() 的调用中获取信息，它们都在上面的代码中体现出来了。需要注意的是同一个位置可能有多个 Address 结果，它们以 List<Address> 对象的形式表示。一般而言，第一个 Address 是最详细的，后面的 Address 对象可能包含较少的细节，而且描述一个更大的范围。

第一种方法是查询特定的信息，例如使用 getFeatureName() 方法或 getLocality() 方法。这两个方法并不确保能够返回所有位置的有用信息。不过当你确定只需要一个大体的信息（比如国家）时，它们还是很有用的。

第二种方法是通过“地址行（address line）”查询信息。这通常用来向用户提供地址信息、方向指示或者所需的街道地址。也就是说，返回的地址可能不是完整的。你只需要简单地调用 getAddressLine() 方法并且以 getMaxAddressLineIndex() 所返回的次数进行迭代即可。图 14.1 显示了包含三种地址结果的示例。

Geocoder 对象还支持通过命名的位置或地址行来获取其对应的经纬度信息。它的输入可以比较随意，在大多数情况下能够得到返回结果。例如，下面所有的地址都能够得到有效而正确的结果：“Eiffel Tower（埃菲尔铁塔）”、“London，UK（英国伦敦）”、“Iceland（冰岛）”、“BOS（波士顿）”、“Yellowstone（黄石）”和“1600 Pennsylvania Ave，DC（美国华盛顿哥伦比亚特区宾夕法尼亚大道 1600 号）”。

下面的代码给出了一个基于用户输入信息进行位置计算的示例，它通过一个按钮实现同用户的交互：

```
public void onClick(View v) {
    String placeName = name.getText().toString();

    try {
        List<Address>geocodeResults =
            coder.getFromLocationName(placeName, 3);

        Iterator<Address> locations = geocodeResults.iterator();
        String locInfo = "Results:\n";

        while (locations.hasNext()) {
            Address loc = locations.next();
            locInfo += String.format("Location: %f, %f",
                       loc.getLatitude(), loc.getLongtitude());
        }

        results.setText(locInfo);
    } catch (IOException e) {
        Log.e("GeoAddress", "Failed to get location info", e);
    }
}
```

访问 getFromLocationName() 方法得到的是一个 Address 对象的 List，这同之前的示例很相似。图 14.2 显示了输入“Eiffel Tower”之后的结果。

一定要做好可能会返回多个结果的准备。向用户提供选择器，让用户从结果中选择最佳位置是个好办法。另一种让用户确认输入位置正确的方式是提供一个地图，并在其上标注出结果。接下来我们就将讨论两种不同的使用 Google Maps 标注地理位置的方法。

Download at http://www.pin5i.com/

图 14.1 三个“地址”的位置信息显示

图 14.2 对 Eiffel Tower 进行位置编/解码之后的结果

Warning

地理编码操作通常需要连接至网络，因此，不应该在主 UI 线程中运行这一操作，而应该为其单独建立一个线程任务，以免导致应用程序的相应速度变慢。

14.3 在地图上标注位置

Android SDK 提供两种不同的方法来显示地理位置，它们均使用 Google Maps。第一种方法是使用某个特定位置的 `Uri` 来运行一个内建的 Google Maps 应用程序。第二种方法是在你的应用程序中嵌入 `MapView` 来显示地图。

14.3.1 地图标注内容

在前面的章节中，我们讲解了如何确定一个位置的经纬度。现在我们使用内置的地图应用程序来将其标示在地图上。下面这段代码演示了如何进行操作：

```
String geoURI = String.format("geo:%f, %f", lat, lon);
Uri geo = Uri.parse(geoURI);
Intent geoMap = new Intent(Intent.ACTION_VIEW, geo);
startActivity(geoMap);
```

代码的第一个任务是创建一个符合地图应用程序所处理的 URI 格式的 `String`，也就是以“`geo:`”开头，后接纬度和经度。这一 URI 之后用于创建一个新的 `Uri` 对象，进而创建一个新的 `ACTION_VIEW Intent`。最后，我们调用 `startActivity()` 方法。如果经纬度数据有效，例如“雅典卫城（Acropolis）”这一位置，其结果应大致如图 14.3 所示。

这种方法的地图显示使用了内置的地图应用程序，在这里为 Google Maps。如果应用程序并不需要整个地图应用程序的细节，或者不需要提供其他进一步的地图控制，那么这是最快捷而方便的实现方法，而且用户也应该比较习惯于其手机设备上自带的地图应用程序。

Download at http://www.pin5i.com/

图 14.3 “雅典卫城（Acropolis）”的地图显示结果

14.3.2 地图标注视图

有时，我们需要将一个地图嵌入应用程序中以提供更为流畅的用户体验。下面，我们向地理信息编/解码示例中添加一个简单的地图，它能够在用户输入地名后迅速地在地图上显示这一位置。

下面的这段 XML 代码给出了需要在布局文件中所做的修改，它引入了一个名为 MapView 的 Widget：

```
<com.google.android.maps.MapView
    android:id="@+id/map"
    android:apiKey="yourMapKey"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
```

你可能已经注意到了，MapView 的 XML 有些特别之处。首先，标记的名称是完整的；其次，它需要一个 apiKey 属性，稍后我们将获取它。

另外，AndroidManifest.xml 文件也需要为使用包含 Google Maps 的 MapView 做些更改。下面给出了两个需要修改的地方：

```
<application
...
    <uses-library
        android:name="com.google.android.maps" />
</application>
<uses-permission
    android:name="android.permission.INTERNET" />
```

上面给出的两个许可权限都是必需的。MapView 对象需要显式地声明 INTERNET 许可权限，并且其库也需要进行显式地引用，否则，将会发生错误。

Download at http://www.pin5i.com/

最后需要说明的是，MapView 只能在 MapActivity 中使用。从 MapActivity 的外部访问 MapView 将会导致一个错误。MapActivity 同一个普通的 Activity 类似，但它需要实现 isRouteDisplayed() 方法。如果需要显示路线的话，那么该方法必须返回 true，不然则返回 false。下面是无需显示路线时的默认实现：

```
@Override
protected boolean isRouteDisplayed() {
    // 我们并不显示路线
    return false;
}
```

现在，应用程序就可以使用 MapView 来向用户显示地图了。下面的代码段演示了一个 MapController 对象的获取，它用于控制 MapView 所显示的位置：

```
MapView map = (MapView) findViewById (R.id.map);
map.setSatellite(true);
final MapController mapControl = map.getController();
mapControl.setZoom(17);
```

上面的代码通过设置 MapView 来显示卫星视图，它看起来更吸引人。之后，MapController 对象设置了地图的缩放级别。值越大，地图的缩放等级越高；反之，等级越小；1 为其最小值。代码中所使用的 17 通常会显示几个城市街区，但对于某些区域，这个值可能会过大。稍后我们将讨论如何让用户能够轻松地控制地图的缩放等级。

在上一个示例的基础上，向按钮的处理函数中加入下面的代码，以处理对于地名的解析：

```
GeoPoint newPoint = new
    GeoPoint((int)(lat * 1E6), (int)(lon * 1E6));
mapControl.animateTo(newPoint);
```

在上面的例子中，我们创建了一个新的 GeoPoint 对象，它将用在 animateTo() 方法中。GeoPoint 对象使用百分之一度（microdegree）作为其单位，所以我们需要将位置数据的结果乘以 1E6（1000000 或 100 万）。animateTo() 方法可以使 MapView 所显示的内容平滑地过渡到新的位置。其过渡显示的时长取决于因特网的连接速度以及 MapView 所处的模式。setCenter() 方法可以用来设置地图的中心。

最后，我们基本上可以对示例进行测试了。不过，还有最后一件事情需要注意。你需要从 Google 获取一个 Google Maps Key 以使用其提供的地图服务。

14.3.3 获得调试 API Key

要使用应用程序中的 MapView，你必须拥有一个从 Google 获取的 Google Maps API Key。这个 Key 使用 MD5 特征对你的应用程序进行认证。

对于产品发布，你需要按照以下步骤以将认证替换成发布版本。你可以阅读第 29 章“销售你的 Android 应用程序”以获得更多信息。对于软件测试而言，你可以使用由 Android SDK 创建的调试认证。

你需要执行下面的步骤来生成一个适当的 API Key。

① 为调试认证生成一个 MD5 特征（fingerprint）。

② 使用 Google 账号登录 http://code.google.com/android/maps-api-signup.html。

③ 接受服务条款。

Download at http://www.pin5i.com/

④ 将第①步中得到的 MD5 特征粘贴进去。

⑤ 保存在下一个页面中显示的 Android Maps API Key。

第①步是在你用于进行开发的机器上完成的，你需要定位到 Android SDK 所使用的调试认证。在所有的平台下，默认的文件名为 debug.keystore。如果你使用 Eclipse，那么文件的位置将在 Android 构建首选项（build preferences）中列出。之后你可以运行下面的代码，它需要指定这个文件（确保 Java 工具存在于系统的 PATH 中）：

```
keytool -list -keystore /path/to/debug.keystore -storepass android
```

上面命令的执行结果为你需要在上面的第④步中粘贴的特征值。在继续执行后面的步骤之前请务必仔细阅读条款协议。虽然协议允许多种类型的应用程序，但你仍然需要确保你所开发的应用程序包含在这个范畴中，并且你预先使用的这项服务是被 Google 所许可的。

Tip

Android SDK 用于调试的默认 Key 库只能够支持一年的时间，而且对开发者的计算机具有唯一性。我们强烈推荐你使用持续时间较长而且能够在开发团队中共享的调试 Key，这样你的 Google Maps API Key 可以持续更长时间。除此之外，在安装使用别人的调试 Key 生成的应用程序时，你不需要先进行卸载。我们可以通过下面的命令来生成一个上面提及的 Key:

```
keytool -genkey -keypass android -keystore debug.keystore
➥-alias androiddebugkey -storepass android
➥-validity 10000
➥-dname "CN=Android Debug,O=Android,C=US"
```

它将生成一个能够持续 10000 天，并且能够在团队中共享的调试 Key。如果这个 Key 不在默认位置的话，你需要确保在 Eclipse 中对它进行了引用。

当你成功地完成上面的步骤并获取了 Key 之后，就可以在所使用的 MapView 的 Layout 文件定义中引用它了。现在，当你运行应用程序后，屏幕显示应该大致如图 14.4 所示。

图 14.4 对于地名“Sydney Opera House（悉尼歌剧院）”的 MapView 显示结果

Download at http://www.pin5i.com/

Tip

如果你工作在多用户开发的机器上，或者作为团队的一名成员，那么你需要一个适用于所有调试认证的 API Key。调试认证可以从一台设备复制到多台设备，这能使对 Android Maps API Key 的签名和验证获得成功。这样可以节约许多时间，你就可以不必为团队中的每个人分别修改代码和布局文件了。

14.3.4 拖曳地图视图

有些时候返回的位置可能没有精确地显示出用户所需的正确位置，或者用户希望通过查看周围的情况来确定所在的位置。一种满足这种需求的方法是拖曳（pan）地图。幸运地，实现这种操作同在布局文件中开启单击功能一样简单：

```
<com.google.android.maps.MapView
android:id="@+id/map"
android:clickable="true"
android:apikey="mapApiKey"
android:layout_width="fill_parent"
andorid:layout_height="wrap_content" />
```

现在，如果用户打算查找 Great Pyramids（大金字塔），那么他可以通过拖曳地图，使其向东移动一点，这样就可以看到著名的金字塔了，如图 14.5 所示。

图 14.5 查找 Great Pyramids 的结果（左）以及向东移动地图后的显示（右）

14.3.5 缩放地图视图

某些时候，拖曳地图查看周边的操作可能并不能解决用户的问题。他们可能需要在相同位置进行放大或缩小。我们的应用程序并不需要重新实现缩放控制，仅通过一个函数的调用，我们就可以简单地使用这一功能了：

```
map.setBuiltInZoomControls(true);
```

当用户在地图上点击时，缩放控制将淡入显示在视图内，并且可以提供缩放功能，如

Download at http://www.pin5i.com/

图 14.6 所示。

图 14.6 左图显示了 Wilmington 镇的鸟瞰图，右图显示了放大后的 Long Man of Wilmington

14.3.6 标记点

现在用户已经可以拖曳和缩放地图了，但他们可能会丢失自己的位置。当然，他们可以重新进行搜索以找回原先的位置，但有没有更好的办法能够帮助他们“记住”某一位置呢？Android SDK 提供了实现这一目标的几种方法。一种方法是将 MapView 作为任意 View 对象的容器，这个 View 对象可以使用 GeoPoint 进行关联，而不是一个普通的屏幕或 View 坐标。另一种方法是使用 ItemizedOverlay，它专门用于标记一个或多个位置。最后，你还可以通过 Overlay 和 onDraw() 方法的实现手工地在地图上进行描绘。

对于前面的地名查找示例，我们使用第一种方法。假如你现在已经有了一个合适的地图标记（marker），它是一个 Drawable 资源，下面的代码演示了如何实现地图的标注：

```
GeoPoint newPoint = new GeoPoint((int)(lat * 1E6),(int)(lon*1E6));
// 在该点上加入一个视图
MapView.LayoutParams mapMarkerParams = new
    MapView.LayoutParams(LayoutParams.WRAP_CONTENT,
    LayoutParams.WRAP_CONTENT,
    newPoint, MapView.LayoutParams.TOP_LEFT );

ImageView mapMarker = new ImageView(getApplicationContext());
mapMarker.setImageResource(R.drawable.paw);
map.addView(mapMarker, mapMarkerParams);
```

MapView 布局参数允许你设置一个 GeoPoint。这样可以使添加的 View 保持在地图上固定的地理位置，并且能够随着地图的拖曳而移动，如图 14.7 所示。

记住添加的 View 将一直“粘”在地图上，除非 MapView 自身消失。如果应用程序需要向用户提供多位置标记的功能，那么有一种简单的实现方法，即使用 ItemizedOverlay 对象。

Download at http://www.pin5i.com/

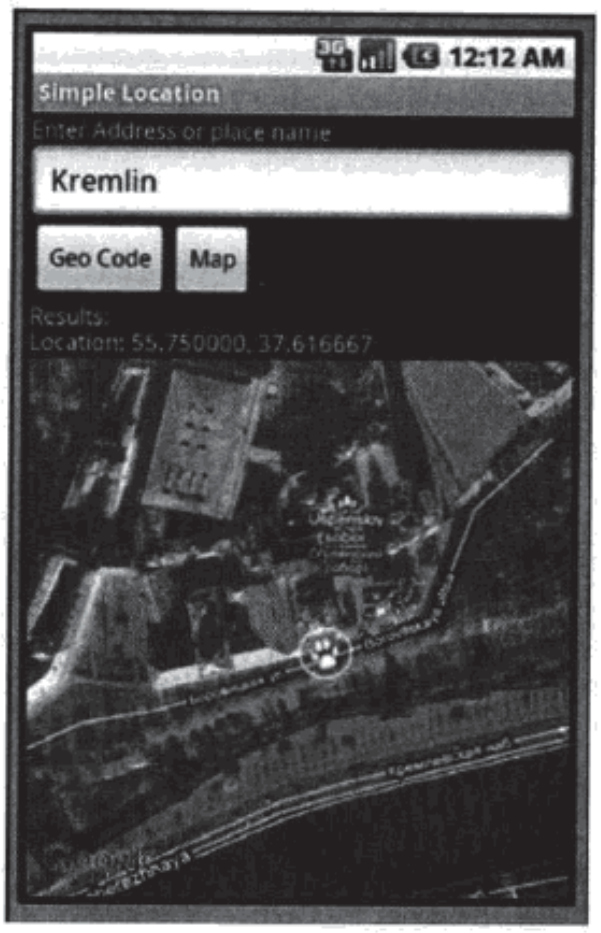


图 14.7 “Kremlin（克里姆林宫）”位于标记（一个带有圆圈的爪印）的左上角

在这个例子中，我们将创建一个静态的 ItemizedOverlay，它将标示出一个背包旅行者沿着阿帕拉契山脉小径（Appalachian Trail）在怀特山（White Mountain）行走的足迹：

```
private class HutsItemizedOverlay
    extends ItemizedOverlay<OverlayItem> {

    public HutsItemizedOverlay(Drawable defaultMarker) {}

    protected OverlayItem createItem(int i) {}
    public int size () {}
}
```

要实现上面的类，我们需要提供每一个 ItemizedOverlay<OverlayItem>方法的实现。首先，我们定义构造函数：

```
public HutsItemizedOverlay(Drawable defaultMarker) {
    super(defaultMarker);

    boundCenterBottom(defaultMarker);

    populate()
}
```

传入的 Drawable 为我们稍后将在 MapActivity 的 onCreate()方法中定义的对象。系统并不提供默认的构造。调用 boundCenterBottom()方法可以使地图坐标处于屏幕正中下部，并去除标记底部的阴影，这能够让它看起来显得更自然。默认的阴影来自于图片的上方。如果我们想要完全去除阴影，那么你可以按照如下方式覆写 draw()方法：

```
@Override
public void draw(Canvas canvas, MapView mapView, boolean shadow) {
    super.draw(canvas, mapView, false);
}
```

最后，在构造函数中，我们调用了 populate()方法。一旦位置数据可用，就应该完成

Download at http://www.pin5i.com/

这一操作。由于我们将其静态地编译到了应用程序中，所以我们在返回之前调用它。populate()方法将会调用我们实现的 createItem()方法，调用的次数取决于 size()方法的实现。下面是我们实现的 createItem()方法，其中包含一个不大的数组，它定义了旅行者的“足迹”，数组元素并没有严格的顺序：

```
public GeoPoint hutPoints[] = new GeoPoint[] {
    // Lakes of the Clouds
    new GeoPoint(44258793, -71318940),
    // Zealand Falls
    new GeoPoint(44195798, -71494402),
    // Greanleaf
    new GeoPoint(44160372, -71660385),
    // Galehead
    new GeoPoint(44187866, -71568734),
    // Carter Notch
    new GeoPoint(44259224, -71195633),
    // Mizpah Spring
    new GeoPoint(44219362, -71369473),
    // Lonesome Lake
    new GeoPoint(44138452, -71703064),
    // Madison Spring
    new GeoPoint(44327751, -71283283),
};

@Override
protected OverlayItem createItem(int i) {

    OverlayItem item = new OverlayItem(hutPoints[i], null, null);
    return item;
}
```

在数组中，我们将所有的位置值都乘以了 100 万，这样正好满足了 GeoPoint 对象所需的“百万分之一度”的单位。在 createitem()方法中，位置数组基于其传入的顺序进行索引。剩下的两个文本参数（Title 和 Snippet）在这里均不使用，所以其值均为 null。数组的最大索引值由 size()方法确定，它仅仅返回数组的大小。

```
@Override
public int size() {
    return hutPoints.length;
}
```

这样，我们实现了必需的 ItemizedOverlay<OverlayItem>类。接下来，应用程序需要将其“告知”MapView。下面的代码演示了如何在 MapActivity 的 onCreate()方法中进行这一操作：

```
@Override
protected void onCreate(Bundle data) {
    super.onCreate(data);
    setContentView(R.layout.huts);

    Drawable marker = getResources().getDrawable(R.drawable.paw);

    HutsItemizedOverlay huts = new HutsItemizedOverlay(marker);
    MapView map = (MapView)findViewById(R.id.map);
    map.setSatellite(true);
```

Download at http://www.pin5i.com/

```
        List<Overlay> overlays = map.getOverlays();
        overlays.add(huts);

        FrameLayout zoomFrame = (FrameLayout)
        findViewById(R.id.map_zoom_holder);
        zoomFrame.addView(map.getZoomControls());
    }
```

首先，代码从资源中获取了 Drawable，接下来，我们实例化了 HutsItemizedOverlay 对象。其中的 OverlayItems 需要加入到已经存在于 MapView 的对象中。MapView 的 getOverlays() 方法返回当前 Overlay 对象的列表。调用这一列表的 add() 方法可以插入每一个“足迹”。最后，缩放控制被加入到了 MapView 中，这样，用户可以对地图进行缩放操作。运行应用程序并且放大“New Hampshire（新罕布什尔州）”后，用户应该能够看见如图 14.8 所示的画面。

图 14.8 带有“足迹”标记的地图，足迹散落在新罕布什尔州的阿帕拉契山脉

在用户需要查看某个位置时，强制其地图进行拖曳和缩放显得不太友好。ItemizedOverlay<OverlayItem>类的两个方法可以提供综合了多个点之后的地图显示参数。将这个功能与覆写后的 getCenter() 方法结合起来，可以使地图能够“照顾”到所有的“足迹”点进行绘制，它会自动地移动到相应位置并采用适当的缩放级别，而 getCenter() 方法默认只返回第一个位置点的信息。下面这段代码可以加入到 onCreate() 方法中以实现这一功能：

```
    MapController mapControl = map.get
Controller();

    mapControl.setCenter(huts.getCenter());
    mapControl.zoomToSpan(
        huts.getLatSpanE6(), huts.getLon
SpanE6());
```

getCenter() 方法计算出所有给定位置点的平均经纬度。可以提供一个中央点，也可以提供中央点附近的第一个位置点，这样我们就无需覆写 get Center() 函数了。

14.4 位置服务扩展应用

你已经学习了 Android 提供的几个不同的位置服务工具，不过，还有几个工具是你需要关注的。

LocationManager 支持“接近警报（Proximity Alerts）”，当手机设备距离目标位置某段距离时，它可以触发一个 PendingIntent。它可以用来提醒用户需要在某个路口转弯，也可以用来进行目标搜寻等。

你已经知道了如何使用 ItemizedOverlay。一般而言，你可以通过指定自己的

Download at http://www.pin5i.com/

Overlay 在 Canvas 上绘制自定义的对象和 View。这可以用来绘制针对某一位置的弹出式菜单，放置不随地图移动的 Logo，或者在地图上显示导航提示等。这一功能类似于在给定位置显示照片，这在 Google Maps 的著名景点处有所体现。

GpsStatus、GpsStatus.Listener 和 GpsSatellite 类提供了有关 GPS 引擎所使用的 GPS 卫星的更多信息。GpsStatus 和它的 Listener 子类监视 GPS 引擎并且获得一个使用中的卫星列表。GpsSatellite 类代表着 GPS 引擎正在使用的某个卫星的当前状态，如卫星的高度（elevation）以及它是否是最近用来进行 GPS 修正的卫星。

Tip

LBS 应用程序是 Android 应用程序中相当流行的类型。不过，LBS 服务与网络服务类似，它们在有些时候会变得不可依赖，或者根本无法使用。所以，当你在设计 LBS 应用程序时，需要注意它的响应性。这也就意味着你需要将 LBS 相关的任务通过线程机制或者是 AsyncTask 异步地完成，当然，你还可以把它实现成 Android 的服务（更多信息可以参考第 21 章）。

小　结

具有 Google Maps 支持和 Key 授权的 Android SDK 可以极大地丰富 Android 应用程序的位置服务功能。某些应用程序需要流畅而平滑的地图支持，但其他一些应用程序只需要提供一个内建的地图程序供用户进行简单的操作即可。开发人员可以通过各种类型的重载（overlay）操作来丰富并增强地图所提供的信息，以供用户使用。尽管如此，使用位置服务来增强 Android 应用程序才刚处于起步阶段，我们可以在这条道路上进行更多新的探索。

参考资料和更多信息

Android Dev Guide（开发指南）：Location and Maps http://developer.android.com/guide/topics/location/index.html.

Get Your Own Google Maps API Key（获取你自己的 Google Map API Key）：http://code.google.com/android/add-ons/google-apis/mapkey.html.

Download at http://www.pin5i.com/

第 15 章

使用 Android 多媒体 API

多媒体——无论是图像、视频还是声音——早已成为了移动设备广受欢迎的主要原因。如今的智能手机设备几乎都拥有一个摄像头，它们可以捕捉和显示静态图像、播放视频，同时提供卓越的音乐播放功能。

在本章中，你将学习到如何使用摄像头捕获静态图像，以及录制和回放音频和视频文件。

15.1 使用多媒体

Android SDK 提供了一系列的方法来处理音频和视频媒体，包括对于多种媒体类型和格式的支持。单独的 Android 设备和开发人员可以扩展其支持的媒体格式列表。但也不是所有的 Android 手机都拥有相同的多媒体功能，所以你需要在发布应用程序前确认目标设备的硬件特性。

Android 平台提供的多媒体特性主要表现在以下三个方面：

① 静态图像（通过摄像头拍摄）；

② 音频（通过麦克风录制，通过扬声器或音频输出口回放）；

③ 视频（通过摄像头和麦克风录制，通过扬声器或音频输出口回放）。

诸如内置摄像头、扬声器和音视频输出端口这样的多媒体硬件都是 Android 的可选设备。

为了获得适当的许可权限，你可以在 Android Manifest 文件中指定应用程序所需要的可选特性。可以使用 Android Manifest 文件的<uses-feature>标记来声明应用程序需要使用摄像头。记住，虽然<uses-feature>标记不是必需的，但是，诸如 Android Market 这样的应用程序商店会根据这些信息来供用户筛选应用程序。

任何申请 CAMERA 许可权限的应用程序都视为使用全部的摄像头功能。如果你的应用程序可以访问摄像头，也可以不访问，那么你可以设定<uses-feature>标记中的 android:required 域为 false。另外，如果你的应用程序需要麦克风和带有自动对焦功能的摄像头，但又不需要设备上提供的闪光灯，诸如这样的需求，你可以通过设置具体的摄像头参数来实现：

```
<uses-feature android:name="android.hardware.microphone" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
```

Tip

本章中的许多代码示例都来自于 SimpleMultimedia 应用程序。在本书的网站上可以下载该应用程序的源代码。

Download at http://www.pin5i.com/

15.2 使用静态图像

在第 7 章“探索用户界面屏幕元素”中我们讲解了如何使用 ImageView Widget 来显示静态的图像。如果用户的手机设备拥有内置的摄像头，那么用户还可以使用 Android SDK 的 Camera 对象捕获静态图像。除此，还可以通过 WallpaperManager 类将图像设为壁纸。

15.2.1 使用摄像头捕捉静态图像

Camera 对象（android.hardware.Camera）控制着设备上的摄像头，并且为其提供一些可用的支持。摄像头的预览功能依赖于适当类型的 SurfaceHolder 的关联。这使得应用程序可以控制摄像头预览画面的显示区域，包括显示位置和尺寸。

下面给出了无需绘制预览帧时向应用程序中加入摄像头捕捉功能的实现步骤（Camera SurfaceView 将显示摄像头视图）。

① 创建一个继承自 SurfaceView 的类，并且实现 SurfaceHolder.Callback。在接下来的例子中，我们将这个类命名为 CameraSurfaceView。

② 在 surfaceCreated() 方法中，获取一个 Camera 对象的实例。

③ 在 surfaceChanged() 方法中，配置并应用（apply）Camera.Parameters；之后，调用 startPreview() 方法。

④ 向 CameraSurfaceView 中加入一个用于捕获图像的方法。

⑤ 将 CameraSurfaceView 加入到适当的布局中。

⑥ 设置某种触发机制，比如按钮，来让用户触发图像的捕获。

⑦ 实现一个 PictureCallback 类来处理捕获图片的存储。

⑧ 在 AndroidManifest.xml 文件中加入 android.permission.CAMERA 许可权限。

⑨ 在 surfaceDestroyed() 方法中释放 Camera 对象。

我们首先从 CameraSurfaceView 类开始。

```
import android.hardware.Camera;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

private class CameraSurfaceView extends SurfaceView
implements SurfaceHolder.Callback {

private SurfaceHolder mHolder;
private Camera camera = null;

pubic CameraSurfaceView(Context context) {
    super(context);
    mHolder = getHolder();
    mHolder.addCallback(this);
    mHolder.setType(
        SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
}

public void surfaceChanged(SurfaceHolder holder,
    int format, int width, int height) {
}
```

Download at http://www.pin5i.com/

```
        public void surfaceCreated(SurfaceHolder holder) {
        }

        public void surfaceDestroyed(SurfaceHolder holder) {
        }

        public boolean capture(Camera.PictureCallback
            jpegHandler) {
        }
    }
```

CameraSurfaceView 类的构造函数配置了 SurfaceHolder，包括将 SurfaceHolder 的类型设置为 SURFACE_TYPE_PUSH_BUFFERS，这将在摄像头内部机制中使用。这一构造函数比较适合在 Activity 的 onCreate()函数中调用。当显示准备就绪后，系统会自动调用 surfaceCreated()方法。下面我们实例化 Camera 对象：

```
    public void surfaceCreated(SurfaceHolder holder) {
        camera = Camera.open();
        camera.setPreviewDisplay(mHolder);
    }
```

Camera 对象使用其拥有的一个静态方法来获取一个可用的实例。由于 Surface 已经可用了，所以经过配置的 Holder 就可以同其相关联了。现在可能还得不到有关 Surface 的信息，不过在下一个对于 surfaceChanged()方法的调用中，将会设置摄像头的相关参数，并且启动预览，如下所示：

```
    public void surfaceChanged(SurfaceHolder holder,
        int format, int width, int height) {
        List<Camera.Size> sizes = params.getSupportedPreviewSizes();

        Camera.Size pickedSize = getBestFit(sizes, width, height);
        if (pickedSize != null) {
            params.setPreviewSize(pickedSize.width, pickedSize.height);
            camera.setParameters(params);
        }
        camera.startPreview();
    }
```

surfaceChanged()方法向应用程序提供了摄像头预览所需的适当宽度和高度。在给这些属性赋予 Camera 对象后，预览将会启动。此时，用户将能够从屏幕上看见摄像头捕获的画面。但是，当你在模拟器中调试时，将只能看到带有一个动画方块和黑白相间格子图案的画面，如图 15.1 所示。这是对摄像头捕获画面的一个模拟，这样，从某种层面上讲，摄像头的测试才能够得以在模拟器中进行。

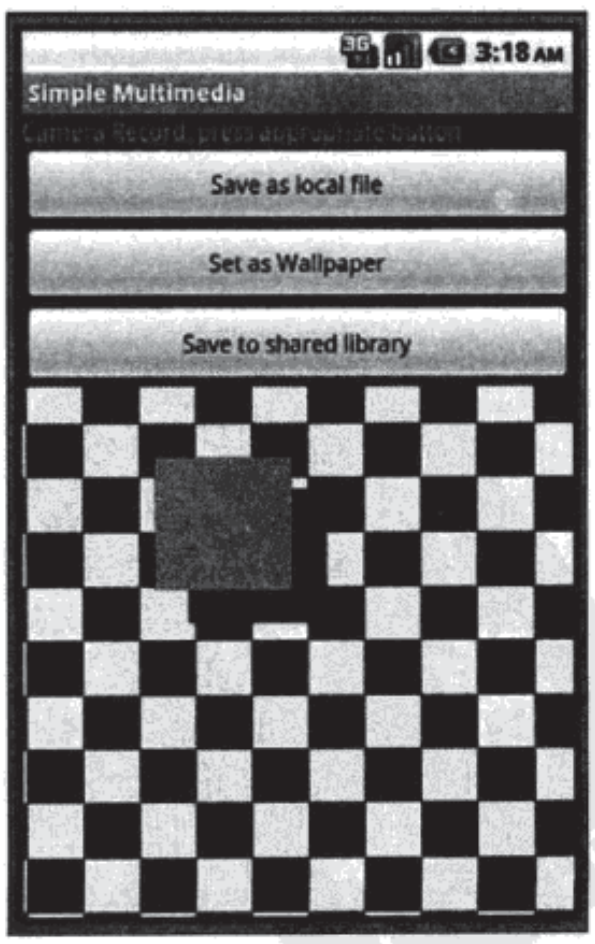


图 15.1 显示了模拟摄像头视图的模拟器屏幕画面

Note

传入 surfaceChanged()方法的格式参数与 Camera 对象的 setPreviewFormat()方法中的格式参数之间并没有关系。

当不再显示 Surface 时，surfaceDestroyed()方

Download at http://www.pin5i.com/

法将被调用。下面是适合这个例子的 surfaceDestroyed()方法的实现：

```
public void surfaceDestroyed(SurfaceHolder holder) {
    camera.stopPreview();
    camera.release();
    camera = null;
}
```

在 surfaceDestroyed()方法中，应用程序将终止预览并且释放 Camera 对象。当需要再次使用 CameraSurface View 时，将会再一次调用 surfaceCreated()方法，所以这里是进行该操作最合适的地方。

捕获一幅静态图像所需的最后一步是选择某种方式调用Camera对象的takePicture()方法。CameraSurface View 可以提供对 Camera 对象的公有访问，但在这个例子中，我们将提供一种在 CameraSurfaceView 类中实现图像捕获的方法：

```
public boolean capture(Camera.PictureCallback jpegHandler) {
    if (camera != null) {
        camera.takePicture(null, null, jpegHandler);
        return ture;
    } else {
        return false;
    }
}
```

你还可以使用 takePicture()方法来关联一个回调方法以播放快门声音，或者在图像被传感器捕获之前采取任何其他的操作。另外，PictureCallback 还可以用于获取摄像头的原始数据。

Note

摄像头原始数据的格式可能因设备的不同而不同。

现在，Activity 中的 CameraSurfaceView 对象已将准备就绪了。在这个例子中，我们使用了一个包含用于控制位置的 FrameLayout Widget 的 Activity。下面是将 cameraView 关联到布局的示例代码：

```
final CameraSurfaceView cameraView = new
    CameraSurfaceView(getApplicationContext());
FrameLayout frame = (FrameLayout) findViewById(R.id.frame);
frame.addView(cameraView);
```

接下来，按钮单击事件的处理方法将调用 CameraSurfaceView 对象的 capture()方法。下面给出了一种实现的示例：

```
public void onClick(View v) {
    cameraView.capture(new Camera.PictureCallback() {

        public void onPictureTaken(byte[] data,
            Camera camera) {
            FileOutputStream fos;

            try {
                String filename = "capture.jpg";
                fos = openFileOutput("capture.jpg",
                    MODE_WORLD_READABLE);
```

Download at http://www.pin5i.com/

```
                fos.write(data);
                fos.close();

            } catch (Exception e) {
                Log.e("Still", "Error writing file" , e);
            }
        }
    });
}
```

从回调中获得的数据可以直接以 JPEG 文件的形式写到应用程序的文件目录中。如果像上面的示例代码这样编写，那么捕获的图像就只能被创建它的应用程序自身所使用。在某些情况下，这也许能够满足需求。不过有时应用程序可能需要将图像同手机上的其他应用程序共享，例如，在使用 MediaStore 内容提供器的“Pictures”应用程序中显示它。这可以通过 ContentResolver 对象提供的媒体库入口来实现。

Warning

对于任何繁冗耗时的操作，比如向系统写入大文件，你都需要将它们放在一个单独的线程中运行，以保证应用程序的正常响应。

15.2.2 配置摄像头模式

你可以使用 Camera 类来配置具体的摄像头捕捉参数，许多这类参数都存储在 Camera.Parameters 类中，然后使用 setParameters() 方法进行设定。

1. 使用普通摄像头参数

让我们仔细看看 Camera.Parameter 类，以下是一些有用的摄像头参数：

① 闪光灯模式（当具备闪光灯硬件时）；
② 对焦模式（固定点、景深（depth of field）、无限远等）；
③ 白平衡设置（荧光灯、白炽灯模式等）；
④ 场景模式（雪天、海滩、烟火等）；
⑤ 特效（负片、老照片（sepia）等）；
⑥ 边缘锐化（anti-banding）设置（去除噪点）。

不同的设备支持不同的参数，所以你需要在开启参数前确认设备支持这些参数。另外，你还可以使用 Camera.Parameter 类来检验摄像头所支持的特性。例如，getSupported FlashModes()、getSupportedFocusModes() 等方法可以返回设备所支持的参数。除此之外，Camera.Parameters 类还包含一些用于访问摄像头底层设置的方法，如曝光补偿和 EXIF 信息等。

2. 摄像头变焦

Camera 类的 startSmoothZoom() 和 stopSmoothZoom() 方法可以用来控制摄像头的变焦。你也可以通过 Camera.Parameter 类来设定变焦参数，以下是它提供的一些有用的变焦方法。

Download at http://www.pin5i.com/

① isZoomSupported()方法可以确定是否支持变焦。
② isSmoothZoomSupported()方法可以确定是否支持平滑变焦。
③ getMaxZoom()方法可以得到最大变焦倍数。
④ getZoom()方法可以获取当前的变焦倍数。
⑤ setZoom()方法可以设定当前的变焦倍数。
⑥ getZoomRatios()方法可以用来计算变焦倍数（例如 1x、2x、10x）。

由于具体摄像头的不同，变焦可能是数码变焦，也可能是光学变焦，或者是二者的结合。

15.2.3 分享图像

如上面所演示的那样将图像存储在应用程序的本地目录中对于某些应用程序可能会比较适用；然而，对于其他的应用程序，可能将所捕获的图像存入设备的媒体库显得更合适。ContentResolver 可以协同 MediaStore 对象，将图像存入共享的图像库中。下面的例子演示了将由摄像头拍摄的静态图像以图片文件的形式存入 MediaStore 内容提供器的方法，它使用了相同的 Camera 回调：

```
public void onPictureTaken(byte[] data, Camera camera) {
    Log.v("Still", "Image data received from camera");
    try {
        Bitmap bm = BitmapFactory.decodeByteArray(
            data, 0, data.length);
        String fileUrl = MediaStore.Images.Media.
            insertImage(getContentResolver(), bm,
            "Camera Still Image",
            "Camera Pic Sample App Took");

        if (fileUrl == null) {
            Log.d("Still", "Image Insert failed");
            return;
        } else {
            Uri picUri = Uri.parse(fileUrl);
            sendBroadcast(new Intent(
                Intent.ACTION_MEDIA_SCANNER_SCAN_FILE,
                picUri));
        }
    } catch (Exception e) {
        Log.e("Still", "Error writing file", e);
    }
}
```

图像被转换成了 Bitmap 对象，它将被传入 insertImage()方法。该方法在共享图像库中创建一个条目。在图像被插入后，我们使用返回的 URL 来创建一个 Uri 对象，它表示了这个新图像的位置，并且通过广播特定 Intent 的方法来通知 Media Scanner（媒体扫描器）找到这个图像。要确定扫描是否成功完成，你可以调用静态方法 MediaScannerConnection.scanFile()，并且提供 MediaScannerConnection.OnScanCompletedListener 类的实现。

现在，所有使用 MediaStore 内容提供器的应用程序（例如“Pictures”应用程序）都可以访问到这幅图像。

Download at http://www.pin5i.com/

Warning

要通过模拟器使用 MediaStore，你必须拥有一个映射（mounted）的 SD 卡镜像。另外，虽然技术上讲并不需要强制 Media Scanner 扫描新增的图像，但是我们发现如果在访问某个图像之前没有进行扫描的话，模拟器和设备上的“Pictures”应用程序可能会崩溃。所以，事先发送一个 Intent 是很明智的选择。

15.2.4 将图像指定为壁纸

设置自定义壁纸是通过感兴趣的图片来个性化用户手机的一种很好的途径。Wallpaper Manager 类提供了访问壁纸的功能。你将在第 22 章“扩展 Android 应用程序”中学习到更多内容，届时你将学习创建 Live Wallpaper（Live 壁纸）。现在，我们只用它来设置静态图像的壁纸。

当前的壁纸可以通过调用 getDrawable() 或 peekDrawable() 方法来获取。getDesired MinimumHeight() 和 getDesiredMinimumWidth() 方法允许应用程序调整特定设备上的壁纸尺寸。最后，壁纸可以通过调用 setResource()、setBitmap() 和 setStream() 方法进行关联。

下面对 Camera 对象的调用设置了一个壁纸：

```
public void onPictureTaken(byte[] data, Camera camera) {
    Bitmap recordedImage =
        BitmapFactory.decodeByteArray(data, 0, data.length);
    try {
        WallpaperManager wpManager = WallpaperManager
            .getInstance(StillImageActivity.this);
        wpManager.setBitmap(recordedImage);
    } catch (Exception e) {
        Log.e("Still", "Setting wallpaper failed.", e);
    }
}
```

图片能够在本地被复制以用作壁纸，所以原始图片可以不保留，这在本例中比较恰当，因为它从未被写入到磁盘。可以通过调用 clear() 方法完全移除关联的壁纸。

最后需要说明的是，你的应用程序需要在 AndroidManifest.xml 文件中声明 android.permission.SET_WALLPAPER 许可权限。

Note

在 API Level 5（Android 2.0）以前，简单的壁纸操作是通过直接访问 Context 对象实现的。更多信息可以参考 Android SDK 文档中有关 Context.setWallpaper() 和 Context.getWallpaper() 方法的部分。

15.3 使用视频

最近几年，视频已逐渐成为手机设备上常见的媒体。市场上的大多数手机都支持拍摄和播放视频，Android 亦如此，虽然不同设备的视频特性会有所不同。

Download at http://www.pin5i.com/

15.3.1 录制视频

Android 应用程序可以使用 MediaRecorder 类来录制视频。使用 MediaRecorder 涉及以下步骤。

① 实例化一个新的 MediaRecorder 对象。
② 设置视频源。
③ 设置视频的输出格式。
④ 设置视频的录制尺寸（可选）。
⑤ 设置视频的帧率（可选）。
⑥ 设置视频的编码方式。
⑦ 设置用于存储录制视频的文件（扩展名必须与输入格式相匹配）。
⑧ 设置预览窗口（surface）。
⑨ 为录制准备对象。
⑩ 开始录制。
⑪ 当完成录制时停止并释放录制对象。

按照上面的步骤，使用标准按钮进行控制，可以创建一个 Activity 用来录制和播放视频。“录制”按钮的 onClick() 方法应该大致如下所示：

```
pubilc void onClick(View v) {
    if (videoRecorder == null) {
    videoRecorder = new MediaRecorder();
    }
    String pathForAppFiles =
        getFilesDir().getAbsolutePath();
    pathForAppFiles += RECORDED_FILE;

    videoRecorder.setVideoSource(
        MediaRecorder.VideoSource.CAMERA);

    videoRecorder.setOutputFormat(
        MediaRecorder.OutputFormat.MPEG4 );

    videoRecorder.setVideoSize(640, 480);
    videoRecorder.setVideoFrameRate(30);
    videoRecorder.setVideoEncoder(
        MediaRecorder.VideoEncoder.H264);

    videoRecorder.setOutputFile(pathForAppFiles);
    videoRecorder.setPreviewDisplay(surface);

    videoRecorder.prepare();
    videoRecorder.start();

    // 可在此插入按钮处理和其他行为的代码
}
```

在上面的代码中，我们实例化了一个 videoRecorder 对象，并且针对录制源设置了一些视频参数。对于每一种视频配置，均有不同的参数值可以用来设定，不过，这仍然要根据设备来确定。

“停止”按钮的 onClick() 事件处理代码应该大致如下所示：

```
public void onClick(View v) {
    if (videoRecorder != null) {
```

Download at http://www.pin5i.com/

```
            videoRecorder.stop();
            videoRecorder.release();
            videoRecorder = null;
        }
        // 可在此插入按钮处理和其他行为的代码
    }
```

最后，需要录制视频的应用程序需要在 AndroidManifest.xml 文件中显式地声明 android.permission.CAMERA 许可权限。

现在，我们来加入视频回放的功能，这样我们就可以观看刚刚拍摄的视频了。

15.3.2 播放视频

Android SDK 提供的最简单的视频回放方法是使用 VideoView Widget，它通过一个 MediaController Widget 来提供基本的视频控制功能。下面是 Activity 的 onCreate() 方法的实现，它演示了一种视频回放的解决方案：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.moving);

    VideoView vv = (VideoView) findViewById(R.id.video);
    MediaController mc = new MediaController(this);
    Uri video = Uri.parse(MOVIE_URL);

    vv.setMediaController(mc);
    vv.setVideoURI(video);
}
```

Note

Android 模拟器并不能在所有屏幕分辨率下都能完美地播放视频，所以，你需要将播放视频的代码放在真实设备上进行测试。

带有这些控件的简单布局文件应该大致如图 15.2 所示。MediaController 提供了一个漂亮的 ProgressBar，它显示了视频的下载进度和当前的播放进度。当使用了 VideoView 的 setMediaController() 方法后，将无需再使用 MediaController 的 setAnchorView() 方法，因为它已经被自动设置成了 VideoView。

调用 setVideoURI() 方法将会自动开始回放。你可以创建一个监听器来监听回放的完成事件，这可以通过 VideoView 的 setOnCompletionListener() 方法来实现。VideoView 对象还提供其他几种有用的方法，例如，getDuration() 方法以及直接对回放进行控制的 pause() 方法等。如果需要提供对媒体更好的控制，或者需要使用其他的方式来播放媒体，那么可以使用 MediaPlayer

图 15.2 含有各种默认控件的视频回放界面

Download at http://www.pin5i.com/

对象。使用这个对象与使用 Camera 类似，你需要一个 SurfaceHolder。

Warning

MediaController 不能通过 findViewById()方法从布局文件的 XML 定义中获取。它必须通过程序进行实例化，并且使用 Activity 的 Context，而不是 Application 的 Context。

15.4 使用音频

同视频非常相似，Android SDK 提供了多种方法以实现音频的播放和录制。音频文件可以是资源、本地文件，以及 Uri 对象所指定的共享或网络资源。音频录制通过设备内置的麦克风实现（通常作为一个电话来讲，麦克风设备应该是必需的）。

15.4.1 录制音频

Android SDK 中的 MediaRecorder 对象提供音频的录制功能，要使用它需要执行以下的几个步骤，它们基本上是你很熟悉的。

① 实例化一个新的 MediaRecorder 对象。
② 设置音频源。
③ 设置音频的录制格式。
④ 设置存储音频的文件格式。
⑤ 设置录制的目标文件。
⑥ 为录制准备对象。
⑦ 开始录制。
⑧ 当录制完成时停止并释放录制对象。

根据上面的步骤，创建两个简单的按钮和一个简单的 Activity 即可实现音频的录制和回放。“录制”按钮的 onClick()方法应该大致如下所示：

```
public void onClick(View v) {
    if (audioRecorder == null) {
        audioRecorder = new MediaRecorder();
    }
    String pathForAppFiles =
        getFilesDir().getAbsolutePath();
    pathForAppFiles += RECORDED_FILE;

    audioRecorder.setAudioSource(
        MediaRecorder.AudioSource.MIC);
    audioRecorder.setOutputFormat(
        MediaRecorder.OutputFormat.DEFAULT);
    audioRecorder.setAudioEncoder(
        MediaRecorder.AudioEncoder.DEFAULT);

    audioRecorder.setOutputFile(pathForAppFiles);

    audioRecorder.prepare();
    audioRecorder.start();

    // 可在此插入按钮处理和其他行为的代码
}
```

Download at http://www.pin5i.com/

上面的代码实例化了 audioRecorder 对象。对于我们的目的而言，默认的录制源和输出文件就可以满足要求了。

Warning

如果你发现录制并没有开始，检查一下所使用的文件扩展名。例如，当使用 MPEG-4 时，Android SDK 需要文件的扩展名为.mp4，否则，录制将不会开始。

“停止”按钮的 onClick()方法应该大致如下所示：

```
public void onClick(View v) {
    if (audioRecorder != null) {
        audioRecorder.stop();
        audioRecorder.release();
        audioRecorder = null;
    }
    // 可在此插入按钮处理和其他行为的代码
}
```

最后，要录制音频的应用程序需要在 AndroidManifest.xml 文件中显式地声明 android.permission.RECORD_AUDIO 许可权限。

现在，我们来加入音频回放的功能，这样我们就可以播放刚刚录制的音频了。

15.4.2 播放音频

MediaPlayer 对象可以用来播放音频。下面的步骤用于准备一个待播放的音频文件。

① 实例化一个 MediaPlayer 对象。

② 使用 setDataSource()方法设置文件的路径。

③ 调用 MediaPlayer 对象的 prepare()方法。

④ 调用 start()方法开始音频的播放。

⑤ 可以通过调用 stop()方法停止播放。

“播放”按钮的 onClick()方法应该大致如下所示，它将播放前面例子中录制的音频：

```
public void onClick(View v) {
    if (player == null) {
        player = new MediaPlayer ();
    }
    try {
        String audioFilePath =
            getFilesDir().getAbsolutePath();
        audioFilePath += RECORDED_FILE;

        player.setDataSource(audioFilePath);

        player.prepare();
        player.start();
    } catch (Exception e) {
        Log.e("Audio", "Playback failed.", e);
    }
}
```

音频数据源可以是一个本地文件路径、有效的文件对象，或者一个指向音频源的有效 Uri。在程序中，可以通过调用 stop()方法来停止音频的播放。另外，可以设置 MediaPlayer.

Download at http://www.pin5i.com/

OnCompletionListener 获得播放完成时的回调。当使用完 MediaPlayer 对象后，需要调用 release()方法以释放所有使用过的资源，这与释放 MediaRecorder 对象相似。

> **Tip**
>
> AudioManager（android.media.AudioManager）是一个系统服务。你可以通过调用 getSystemService(Context.AUDIO_SERVICE)方法来请求一个 AudioManager。AudioManager 可以用来查询、管理和修改设备级别的音频设置。在 Android 2.2 SDK 中，许多新的 API 被添加到了 AudioManager，用于管理音频焦点，也就是说，同时播放的不同声源将能够被协调起来。这一功能对于诸如播客（podcast）和音乐播放器这样的音频流应用程序将尤为重要。

15.4.3 共享音频

音频可以与系统上的其他应用程序共享。ContentResolver 可以向 MediaStore 内容提供器发送文件。下面的代码片段显示了如何在设备上的音频库中配置一个音频条目：

```
ContentValues values = new ContentValues(9);

values.put(MediaStore.MediaColumns.TITLE, "RecordedAudio");
values.put(MediaStore.Audio.Media.ALBUM,
    "Your Groundbreaking Album");
values.put(MediaStore.Audio.Media.ARTIST, "Your name");
values.put(MediaStore.Audio.Media.DIAPLAY_NAME,
    "The Audio File You Recorded In Media App");
values.put(MediaStore.Audio.Media.IS_RINGTONE, 1);
values.put(MediaStore.Audio.Media.IS_MUSIC, 1);
values.put(MediaStore.MediaColumns.DATE_ADDED,
    System.currentTimeMillis()/1000);
values.put(MediaStore.MediaColumns.MINR_TYPE,"audio/mp4");
values.put(MediaStore.Audio.Media.DATA, pathForAppFiles);

Uri audioUri = getContentResolver().insert(
    MediaStore.Audio.Media.EXTERNAL_CONTENT_URI, values);
if (audioUri == null) {
    Log.d("Audio", "Content resolver failed");
    return;
}
```

通过上面的设置，设备上的其他基于音频的应用程序就可以使用之前录制的音频了。例如，设置 IS_MUSIC 标签可以使其能够显示在不同类型的音乐播放器中，并且能够根据其专辑信息进行排序；设置 IS_RINGTONE 标签可以使其显示在设备的铃音列表中。

手机设备会周期性地扫描新的媒体文件。不过，如果想要加速这一进程，可以向系统发送一个 BroadcastIntent 来通知它有新的音频文件出现。下面的代码演示了向 Content 库中添加音频的方法：

```
sendBroadcast(new Intent(
    Intent.ACTION_MEDIA_SCANNER_SCAN_FILE, audioUri));
```

在处理了上面的 Intent 之后，音频文件会立即出现在相关的应用程序中。

15.4.4 搜索多媒体

你可以使用一个名为 android.intent.action.MEDIA_SEARCH 的 Intent 来搜索设备上的多

Download at http://www.pin5i.com/

媒体。例如，可以像下面这样执行针对特定艺术家和歌曲的搜索：

```
Intent searchMusic = new Intent(
    android.provider.MediaStore.INTENT_ACTION_MEDIA_SEARCH);
searchMusic.putExtra(android.provider.MediaStore.EXTRA_MEDIA_ARTIST,
     "Cyndi Lauper");
searchMusic.putExtra(android.provider.MediaStore.EXTRA_MEDIA_TITLE,
     "I Drove All Night");
searchMusic.putExtra(android.provider.MediaStore.EXTRA_MEDIA_FOCUS,
     "audio/*");
startActivity(searchMusic);
```

如果你的设备上有一堆音乐（比如 Cyndi Lauper 的“I Drove All Night”）并且运行这个 Intent，那么你将可以直接定位到你请求的音乐。注意如果你安装了多个音乐播放器，那么在第一次发送 Intent 时，你也许需要从中选择一个（如“Music”应用程序）。

15.4.5 使用铃音

同壁纸设置非常相似，设置自定义的铃音也是个性化手机设备的常用方式。Android SDK 提供了通过 RingtoneManager 对象来管理铃音的各种方法。前面示例中录制的音频可以通过下面的静态方法调用被指定为当前铃音：

```
RingtoneManager.setActualDefaultRingtoneUri(
    getApplicationContext(),
    RingtoneManager.TYPE_RINGTONE, audioUri);
```

其中，类型还可以设定为 TYPE_ALARM 或 TYPE_NOTIFICATION，这样可以为其他使用音频的系统事件配置声音。要成功地执行上面的操作，应用程序必须拥有在 AndroidManifest.xml 文件中声明的 android.permission.WRITE_SETTINGS 许可权限。使用 RingtoneManager.getActualDefaultRingtoneUri() 静态方法还可以获取当前所设置的铃音。你可以使用返回的 Uri 来播放这一铃音，这在那些需要对用户进行提醒的应用程序中可能使用较多。

小　结

在应用程序中使用多媒体可以显著地增强应用程序的吸引力、实用性以及功能性。Android SDK 中包含了一系列的 API，它们通过摄像头和麦克风等设备提供图像捕获、音/视频录制等功能；同时还提供音频、视频的播放和静态图像的显示功能。多媒体资源可以为某个特定的应用程序所私有，也可以通过 MediaStore Content Provider 在所有应用程序之间进行共享。

参考资料和更多信息

Android Dev Guide（开发指南）：Audio and Video.http://developer.android.com/guide/topics/media/index.html.

Android Supported Media Formats（Android 支持的媒体格式）.http://developer.android.com/guide/appendix/media-formats.html.

Download at http://www.pin5i.com/

第 16 章

使用 Android 电话 API

虽然 Android 平台被设计为能够运行在几乎所有类型的设备上，但目前市场上主要的 Android 设备还是手机。应用程序可以利用这一点，在其功能中融入手机、电话的特性。

本章将向你介绍 Android SDK 中提供的各种类型的 API，它们均与“电话”这一主题相关。

16.1 使用电话功能

Android SDK 提供了一系列有用的功能包，应用程序可以使用它们来整合设备上可用的电话功能与特性。一般而言，开发人员应该首先并且主要将 Android 设备看做电话。虽然这些设备可以运行应用程序，但与电话相关的操作具有很高的优先权。例如，你的应用程序不能够打断任何一次通话。要避免类似情况发生，你的应用程序就应该知晓用户当前正在进行的操作，这样它可以根据不同的情况判断自己的行为。例如，应用程序通过查询电话的状态，发现用户正在进行通话，那么可以将提醒改为震动而非响铃。

另一方面，应用程序可能需要提供拨打电话或者发送短消息的功能。手机通常支持短消息服务（Short Message Service，SMS），它在文字信息中使用得非常普遍。应用程序提供这种功能可以增强其受欢迎的程度，并且在桌面环境中不太容易被复制。由于许多 Android 设备本身是一部电话，所以应用程序可能会经常性地与电话号码和联系人数据库打交道；有些应用程序可能还需要访问拨号器，以便于拨号或者检查电话的状态信息。在应用程序中增添电话功能可以丰富用户体验，从而提升应用程序的整体价值。

Tip

本章中的许多代码示例都来自于 SimpleTelephony 应用程序。在本书的网站上可以下载该应用程序的源代码。

16.1.1 为获取手机状态信息取得许可权限

我们首先从获取设备通话状态开始学习，它包括有关于电话通话状态的请求，通信服务的相关信息，还有对于电话号码的验证与处理等。我们首先从学习 `TelephonyManager` 对象开始，它位于 `android.telephony` 包中。

这个部分中的许多方法调用都需要在 Android 应用程序 Manifest 文件中显式声明的许可权限。`READ_PHONE_STATE` 许可权限用于授权获取一些信息，比如通话状态、手机号码和设备标识符或序列号。`ACCESS_COARSE_LOCATION` 许可权限则提供有关蜂窝定位信息的授

Download at http://www.pin5i.com/

权。我们在第 14 章“使用位置服务 API”中已经比较详细地讨论过了有关位置服务（Location-Based Services）的内容。

下面的这段 XML 需要在 AndroidManifest.xml 文件中使用，以便能够访问基本的电话状态信息：

```
<uses-permission
    android:name="android.permission.READ_PHONE_STATE" />
```

16.1.2 获取通话状态

你可以使用 TelephonyManager 对象来获取电话的状态以及有关电话服务（如手机的电话号码等）的信息。

可以通过 getSystemService()方法获取一个 TelephonyManager 的实例：

```
TelephoneManager telManager = (TelephonyManager)
    getSystemService(Context.TELEPHONY_SERVICE);
```

使用一个有效的 TelephonyManager 实例，应用程序可以进行若干个查询。其中一个重要的方法是 getCallState()。该方法可以返回手机设备当前的语音通话状态。下面这段代码显示了查询通话状态的方法，并列出了所有可能的返回值：

```
int callStatus = telManager.getCallState();
String callState = null;

switch (callStatus) {
    case TelephonyManager.CALL_STATE_IDLE:
        callState = "Phone is idle.";
        break;
    case TelephonyManager.CALL_STATE_OFFHOOK:
        callState = "Phone is use.";
        break;
    case TelephonyManager.CALL_STATE_RINGING:
        callState = "Phone is ringing!";
        break;
    }
Log.i("telephony", callState);
```

上面三种通话状态在模拟器中可以通过 Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）工具来进行模拟，这将在附录 B“Android DDMS 快速入门”中详细讲解。

查询通话状态在某些情况下可能会非常有用。不过，对通话状态的改变进行监听则可以使应用程序能够对用户即将进行的行为做出反应。例如，当手机响铃时，游戏可能自动暂停并保存状态信息以便用户能够安全地接听电话。应用程序可以注册以监听通话状态的改变，这是通过 TelephonyManager 的 listen()方法实现的。

```
telManager.listen(new PhoneStateListener() {
    pubilc void onCallStateChanged(
        int state, String incomingNumber) {
            String newState = getCallStateString(state);
        if (state == TelephonyManager.CALL_STATE_RINGING) {
            Log.i("telephony", newState +
                " number = " + incomingNumber);
        } else {
```

Download at http://www.pin5i.com/

```
                Log.i("telephony", newState);
            }
        }
    }, PhoneStateListener.LISTEN_CALL_STATE);
```

在上面的代码中，当手机开始响铃、用户拨打电话、用户接听电话或者通话断开时，监听器对象（listener）都将被调用。另外，监听器对象还会在其关联后立即被调用以获得初始的状态。

另一个有用的手机状态是服务状态。这一信息可以告诉应用程序手机是否被信号所覆盖，是否仅仅能够拨打紧急电话，或者是否由于处于飞行模式等原因而关闭了手机的无线电通信。要实现这一目标，应用程序可以添加 PhoneStateListener.LISTEN_SERVICE_STATE 标记来监听上面的状态，并且实现 onServiceStateChanged() 方法，以接收一个 ServiceState 对象的实例。另外，应用程序还可以通过构造一个 ServiceState 对象并且直接对其进行查询来获得状态信息，如下所示：

```
int serviceStatus = serviceState.getState();
String serviceStateString = null;
switch (serviceStatus) {

    case ServiceState.STATE_EMERGRNCY_ONLY:
        serviceStateString = "Emergency calls only";
        break;

    case ServiceState.STATE_IN_SERVICE:
        serviceStateString = "Normal service";
        break;

    case ServiceState.STATE_OUT_OF_SERVICE:
        serviceStateString = "No service available";
        break;

    case ServiceState.STATE_POWER_OFF:
        serviceStateString = "Telephony radio is off";
        break;
    }
Log.i("telephony", serviceStateString);
```

除此之外，手机是否处于漫游（roaming）状态还可以通过调用 getRoaming() 方法来确定。一个友好而节约用户开销的应用程序可以提醒用户有关漫游的信息，以避免一些费用较高的漫游操作，比如数据传输等。

16.1.3 请求获取服务信息

除了通话和服务状态信息之外，还可以获取其他有关设备的信息。这些信息对于普通的应用程序而言可能不太实用，但却可以用来诊断故障或者提供基于特定运营商网络的特殊服务。下面的代码获取了几条服务信息。

```
String opName = telManager.getNetworkOperatorName();
Log.i("telephony", "operator name = " + opName);


String phoneNumber = telManager.getLine1Number();
Log.i("telephony", "phone number = " + phoneNumber);
```

Download at http://www.pin5i.com/

```
String providerName = telManager.getSimOperatorName();
Log.i("telephony", "provider name = " + providerName);
```

网络运营商名称为手机设备当前连接的运营商的描述性名称。这通常是当前的基站运营商（Tower Operator）。SIM 卡中的运营商名称一般就是用户同意使用其服务的运营商名称。在这个应用程序编程接口（API）中的电话号码则被定义为 MSISDN，一般为 GSM 手机的号码（即其他用户拨打以与其通话的号码）。

16.1.4 监控信号强度和数据连接速度

有时，应用程序可能会根据手机的信号强度或者运营商所提供的服务类型采取不同的行为。例如，一个需要高带宽的应用程序可能会根据连接速率（如 1xRTT、EDGE 这样的低速连接或者 EV-DO、HSDPA 这样的高速连接）决定其所获取的流信息的质量或者缓冲的大小。TelephonyManager 可以用来获取这些信息。

如果你的应用程序需要对手机状态的改变做出响应，那么可以使用 Telephony Manager 的 listen() 方法并且实现一个 PhoneStateListener 类，这样，我们就可以处理类似服务、数据连接、通话状态、信号强度以及其他手机状态信息的改变了。

16.1.5 使用电话号码

同电话甚至仅仅是“联系人”打交道的应用程序，通常需要处理电话号码的输入、验证和使用。Android SDK 包含了一系列的功能函数，它们简化了对于电话号码字符串的处理。应用程序可以使用基于本地化设置的格式化电话号码。例如，下面的代码使用了 formatNumber() 方法：

```
String formattedNumber =
    PhoneNumberUtils.formatNumber("9995551212");
Log.i("telephony", formattedNumber);
```

在我的本地化设置下，向日志输入的结果应该为“999-555-1212”。调用 PhoneNumberUtils.compare() 方法还可以比较两个电话号码。应用程序还可以通过调用 PhoneNumberUtils.isEmergencyNumber() 方法来确定给定的电话号码是否是紧急号码。这可以在用户拨打紧急号码之前提醒用户，还可以用在电话号码来源可疑的情况下。

Tip

有许多与电话号码格式相关的功能包，它们能够根据不同的本地信息对电话号码进行格式化。在编写应用程序时需要注意不同的国家所使用的电话号码格式是不相同的。例如，有一个名为 formatJapaneseNumber() 的方法可以为电话号码加上在日本使用的特殊前缀。

formatNumber() 方法还可以使用一个 Editable 作为其参数以在适当的位置格式化一个号码。这里，我们通过关联一个 PhoneNumberFormattingTextWatcher 对象来观察 TextView（或 EditText 以用于用户输入），并且在显示号码时对其进行格式化，以演示上面提到的这一有用的特性。下面的代码给出了配置一个能够格式化输入电话号码的 EditText 的方法：

```
EditText numberEntry = (EditText) findViewById (R.id.number_entry);
numberEntry.addTextChangedListener(
```

Download at http://www.pin5i.com/

```
        new PhoneNumberFotmattingTextWatcher());
```

当用户键入有效的电话号码时，它将以与当前本地化设置相匹配的格式显示出来。图 16.1 显示了仅在 EditText 中键入“19995551212”时的画面。

图 16.1 仅输入数字时的格式化显示结果

16.2 使用短消息

最近几年，短消息服务（SMS）已随处可见。综合了消息服务的应用程序，即使是将该服务输出到其他应用程序（outbound），也能够为用户提供社区类应用程序实现的沟通与交际功能。SMS 功能通过 android.telephony 包提供给应用程序。

16.2.1 获得发送和接收短消息的许可权限

SMS 功能需要两个不同的许可权限，分别授权给短消息的发送和接收。可以向 AndroidManifest.xml 文件中加入下面的 XML，它将声明这两个权限：

```
<uses-permission
    android:name="android.permission.SEND_SMS" />
<uses-permission
    android:name="android.permission.RECEIVE_SMS" />
```

16.2.2 发送短消息

要发送 SMS，应用程序需要首先获得一个 SmsManager 实例。不同于其他的系统服务，这是通过调用 SmsManager 的静态方法 getDefault() 实现的：

```
final SmsManager sms = SmsManager.getDefault();
```

现在应用程序已经拥有了 SmsManager，那么发送 SMS 就只需要简单地一行调用语句就可以了。

Download at http://www.pin5i.com/

```
Sms.sendTextMessage(
    "9995551212", null, "Hello!", null, null);
```

在提供了 PendingIntent 接收短消息发送信息的广播后，应用程序才能够知道 SMS 是否已经成功发送。下面的代码演示了对 PendingIntent 的配置，以监听 SMS 的状态：

```
Intent msgSent = new Intent("ACTION_MSG_SENT");

final PendingIntent pendingMsgSent =
    PendingIntent.getBroadcast(this, 0, msgSent, 0);
registerReceiver(new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        int result = getResultCode();
        if (result != Activity.RESULT_OK) {
            Log.e("telephony",
                "SMS send failed code = " + result);
            pendingMsgReceipt.cancel();
        } else {
            messageEntry.setText(" ");
        }
    }
}, new IntentFilter("ACTION_MSG_SENT"));
```

PendingIntent pendingMsgSent 可以在 sendTextMessage() 中使用。消息接收的反馈与上面的代码大致相似，但是是在发送设备接收到来自网络的关于目标设备收到短消息的反馈后才被调用。

如果我们将上面的代码连同一个带有格式化电话号码功能的 EditText 结合在一起，再加上一个用于输入消息正文条目的字段和一个按钮，就可以创建一个简单的用于发送 SMS 的表单了。“发送”按钮的处理代码应大致如下所示：

```
Button sendSMS = (Button) findViewById(R.id.send_sms);
sendSMS.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        String destination =
            numberEntry.getText().toString();

        String message =
            messageEntry.getText().toString();

        sms.sendTextMessage(destination, null, message,
            pendingMsgSent, pendingMsgReceipt);

        registerReceiver(...);
    }
}
```

在加入了上面的代码后，运行结果应该大致如图 16.2 所示。在这个应用程序中，我们使用了模拟器有关“电话号码”的窍门（即它的端口号）。这是不使用硬件设备且避免向手机运营商付费的情况下进行 SMS 发送测试的好办法。

改进上面程序的一种途径是设置发送接收器（sent receiver）在接收到消息成功发送的通知时修改屏幕上显示的某个图像。除此，还可以添加另一个图像，以在消息成功到达时显示。另外，还可以使用 ProgressBar Widget 来让用户跟踪消息发送的进度。

Download at http://www.pin5i.com/

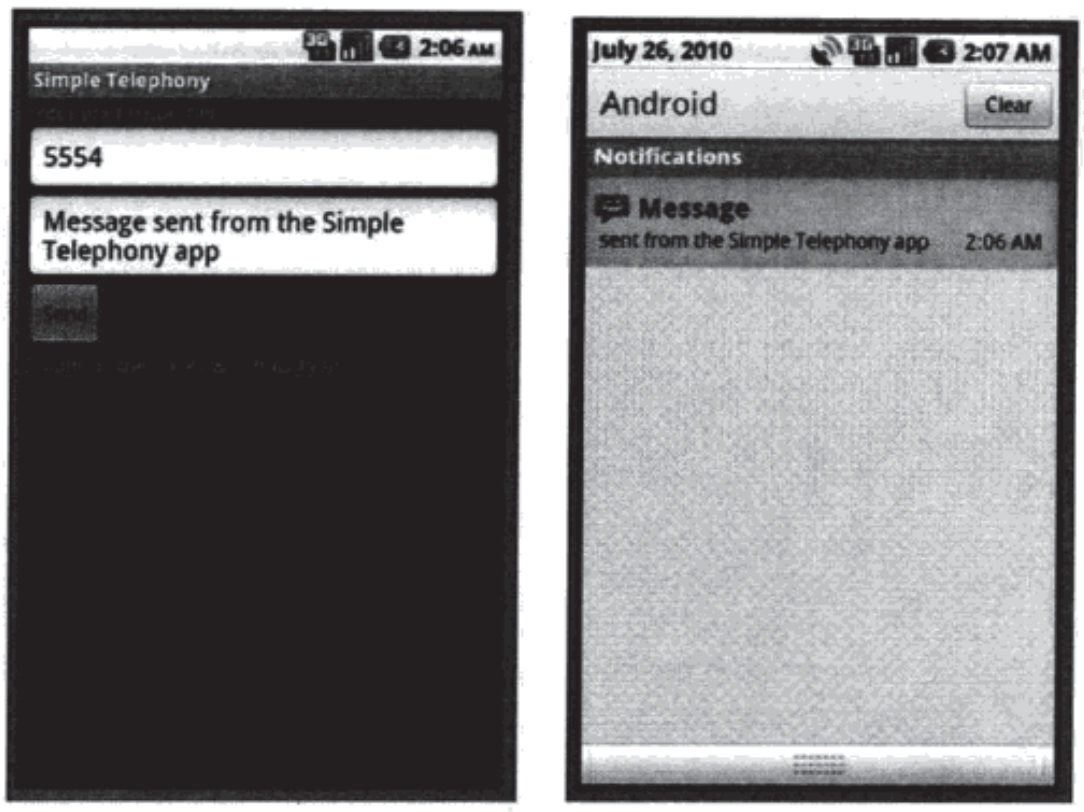


图 16.2 两个模拟器：一个正在从应用程序中发出 SMS，另一个则接收它

16.2.3 接收短消息

应用程序还可以接收 SMS。要实现这一功能，应用程序必须注册一个 BroadcastReceiver 来监听与接收 SMS 有关的 Intent。以这种方法监听 SMS 的应用程序并不能避免其他应用程序接收到相同消息。

我们在前面示例的基础上进行扩展，下面的代码演示了如何将一个到达的消息显示在屏幕上的 TextView 中。

```
final TextView receivedMessage = (TextView) findViewById(
    R.id.received_message);

rcvIncoming = new BroadcastReceiver() {

    public void onReceive(Context context, Intent intent) {
        Log.i("telephony", "SMS received");
        Bundle data = intent.getExtras();
        if (data != null) {
            object pdus[] =
                (Object[]) data.get("pdus");

            String message = "New message:\n";
String sender = null;

for (Object pdu : pdus) {
    SmsMessage part = SmsMessage.
        createFromPdu((byte[])pdu);

    message += part.
        getDisplayMessageBody();

    if (sender == null) {
        sender = part.
            getDisplayOriginatingAddress();
    }
}
receivedMessage.setText(
    message + "\nForm: "+sender);
numberEntry.setText(sender);
```

Download at http://www.pin5i.com/

```
            }
        }
    };

    registerReceiver(rcvIncoming, new IntentFilter(
        "android.provider.Telephony.SMS_RECEIVED"));
```

整段代码可以添加到 Activity 的 onCreate()方法中。首先，代码获取了消息的 Bundle。其中，一个 Object 数组包含了带有 PDU 数据的几个字节数组（PDU 数据被习惯性地用在无线消息协议中）。幸运的是，在 Android SDK 中可以通过调用静态方法 SmsMessage.createFromPdu()来解码这些数据。从这里开始，我们就可以通过调用 getDisplayMessageBody()来提取 SMS 的内容了。

发送消息的长度可能会超出 SMS 的限度。如果这样，它将会在发送端被分成若干个部分。要处理这种情况，我们可以循环以提取出每一个接收到的 Object 所对应的主体部分，并且仅在第一次获得其发送地址。

Tip

当处理多部分文本消息时，需要提醒用户每一部分都有可能按照整条消息的价格被收取费用，这可能会导致较高的花销。另外，还需要告知用户接收或发送文本消息的应用程序可能会产生由运营商收取的费用。

应用程序可以使用 SmsManager.divideMessage()方法来发送类似的多部分消息。这一方法将一个 String 分解成不大于 SMS 允许最大长度的若干部分，应用程序随后可以将 divideMessage()方法的结果传入 sendMultipartTextMessage()方法中来发送一个多部分消息。

接下来，代码更新了 TextView 中的文本字符串以显示用户接收到的消息。同时，发送人的地址也被一同更新，这样可以减少收件人在回复时的操作。最后，我们在系统中注册 BroadcastReceiver。这里使用的 IntentFilter：android.provider.Telephony.SMS_RECEIVED 相当有名但并未在文档中说明。这样，我们只能使用其原始的字符串定义了。

Warning

我们强烈地推荐你关注 Android SDK 有关这个功能的更新。今后版本的 SDK 可能会官方地加入这个字符串，也可能完全取消这一功能。

16.3 拨打和接听电话

如今对于年轻人来说，手机不只是用来接打电话的。任何应用程序都可以用来初始化一次拨号或者接听来电，但是这些功能需要谨慎的设计和实现，一定要避免干扰用户设备自带的电话功能。

Tip

在测试拨打另一台设备时，也可以使用两个模拟器的实例。如同在 SMS 发送的例子中一样，模拟器的端口号即为其可供拨打的电话号码。

Download at http://www.pin5i.com/

1. 拨打电话

之前你已经掌握了如何感知手机是否处于响铃状态。现在，你还将学习到如何让你的应用程序拨打电话。

在前面发送和接收 SMS 示例的基础上，我们现在通过在屏幕上添加按钮来实现拨号功能，这与前面的例子很相似，不同之处仅在于前面实现的是发送短消息的功能。

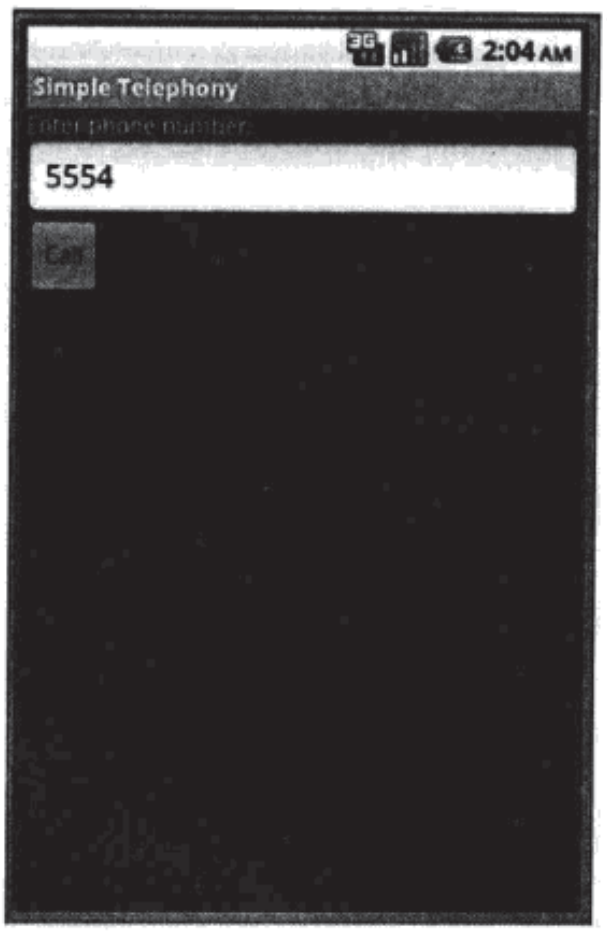


图 16.3 用户可以在 EditText 控件中输入一个号码，按下 Call（拨打）按钮后将在应用程序内初始化一次拨号

Android SDK 允许以两种方式将电话号码传入拨号器（dialer）。第一种方法是使用已键入的电话号码运行拨号器。用户需要随后按下“拨出（Send）”按钮来实际地进行拨号。这种方法不需要任何特殊的许可权限。第二种方法则是实际地拨打电话。它需要在 AndroidManifest.xml 文件中加入 android.permission.CALL_PHONE 许可权限。

让我们来看一个示例，它显示了如何通过输入框获得一个号码，并在用户按下按钮后启动拨号器，如图 16.3 所示。

我们提取了用户键入 EditText 的电话号码（或者是最近收到的 SMS 的地址）来完成这一操作。下面的代码演示了在用户按下“Call”按钮后如何运行拨号器。

```
Button call = (Button) findViewById(R.id.call_button);
call.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        Uri number = Uri.parse("tel:" +
            numberEntry.getText().toString());
        Intent dail = new Intent(
            Intent.ACTION_DAIL, number);
        startActivity(dail);
    }
});
```

首先，代码从 EditText 中提取了电话号码，并且为其添加了 tel:前缀，使其成为一个用于 Intent 的有效的 Uri。之后，代码使用 Intent.ACTION_DIAL 创建了一个新的 Intent，以使用现存的电话号码运行拨号器。另外，Intent.ACTION_VIEW 在这里也适用，且提供相同的功能。如果使用 Intent.ACTION_CALL 替代它，那么将实现直接拨号。一般而言我们并不推荐这么做，因为它可能是由误操作引起的。最后，我们调用了 startActivity()方法启动拨号器，如图 16.4 所示。

2. 接听电话

与应用程序接收和处理短消息类似，应用程序可以注册以接听来电，这需要实现一个广播接收器来处理带有 Intent.ACTION_ANSWER 的 Intent。

Download at http://www.pin5i.com/

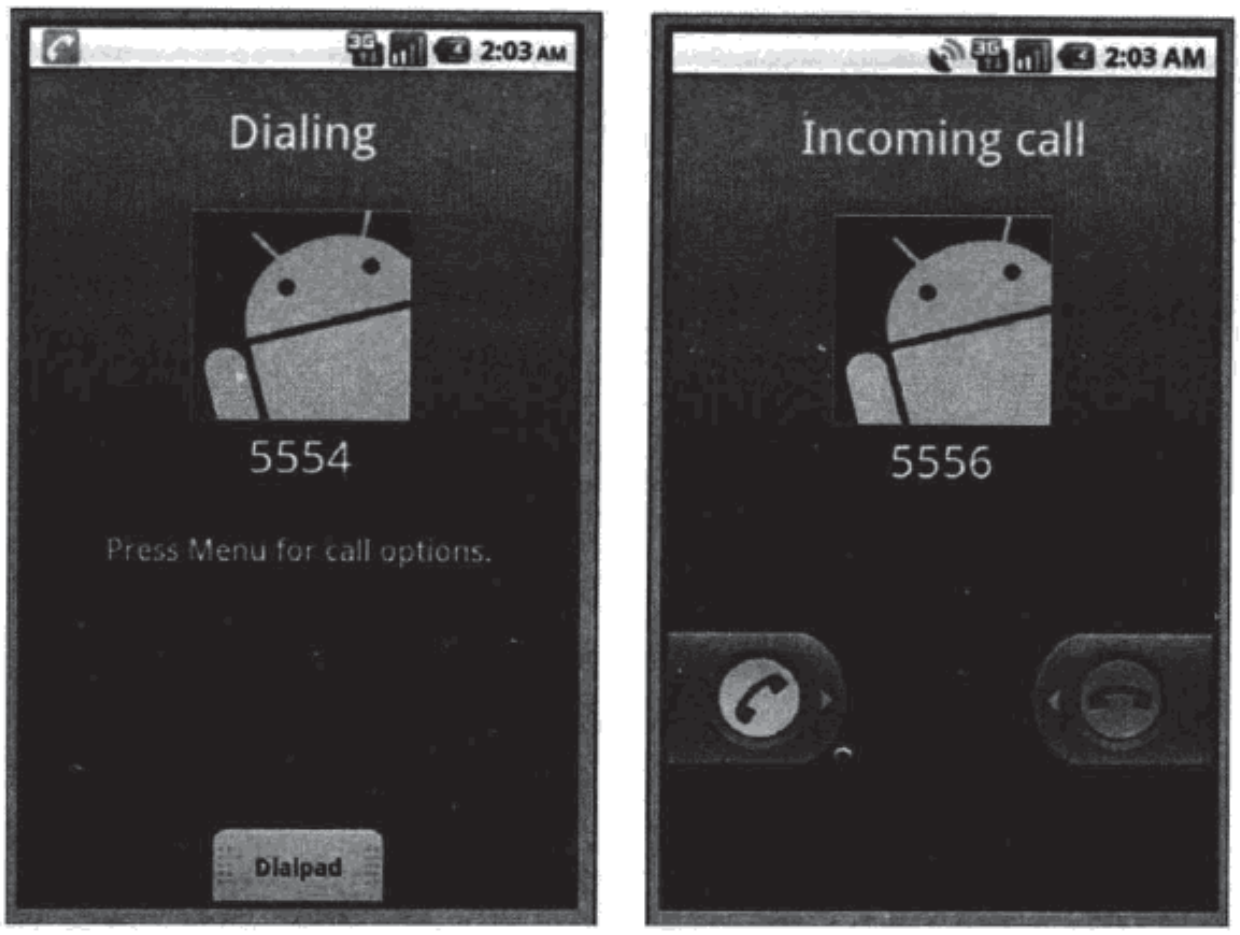


图 16.4 一个模拟器在用户按下应用程序中的“Call”按钮后拨打另一个模拟器

如果你并不关心通话本身，而只是想获取来电的一些信息，那么你可以考虑使用 `CallLog.Calls` 内容提供器（`android.provider.CallLog`）。`CallLog.calls` 类可以用来获取最近的通话信息，例如：

① 谁拨打的电话；
② 什么时候拨打的；
③ 拨入还是拨出；
④ 是否已经接听；
⑤ 通话的时长。

小　结

Android SDK 提供了许多有用的电话功能包来处理电话的拨打和接听以及 SMS 的发送和接收（在适当的许可权限下），除此，它还提供了格式化处理电话号码的功能。

这些电话功能包使得应用程序可以同设备核心的电话特性无缝连接。开发人员也可以将语音通话和消息特性综合到自己的应用程序中，从而获得令人惊喜的新特性。另外，消息服务正变得越来越普及，所以，将消息功能融入到应用程序中可以为其带来耳目一新的用户体验，从而更能够受到用户的欢迎。

参考资料和更多信息

3GPP Specifications (SMS)：http://www.3gpp.org/specifications.
Wikipedia’s Write-Up on SMS：http://en.wikipedia.org/wiki/SMS.

Download at http://www.pin5i.com/

第 17 章

通过 OpenGL ES 使用 Android 3D 图形

我们周围的世界并不是二维平面的，而是一个带有深度的三维空间。虽然手机将各种对象显示在一个平面上，但是为游戏和应用程序提供虚拟的景深，早已成为了增强其真实感的手段之一。为了实现这一目的，Android 软件开发包（Software Development Kit，SDK）提供了对于 OpenGL ES 的实现。

OpenGL ES 是一个在嵌入式系统中使用的图形应用程序编程接口（API），它基于 OpenGL 桌面标准，在各种无线平台上都很常见，且受到了所有主流移动平台的支持，包括 Windows Mobile、Symbian、MeeGo、BREW、Apple iOS、Palm WebOS 以及本书所讲解的 Android。根据平台版本的不同，Android 设备支持各种不同版本的 OpenGL ES。

本章将讨论如何使用 Android SDK 中提供的 OpenGL GS。熟悉 OpenGL 概念会对后面的学习很有帮助。本章并不全面而具体地介绍 OpenGL 本身，只讲解如何在 Android 设备上利用 OpenGL ES 来实现各种不同的效果。这包括配置 EGL（Embedded-System Graphics Library，嵌入式系统图形库）和 GL（Graphics Libraries，图形库）、绘制 3D 动画以及设置场景、灯光和材质等。

17.1 使用 OpenGL ES

在 1992 年之前，硅谷图形公司（Silicon Graphics，SGI）拥有一项商用的（proprietary）图形标准，称为“综合光栅图像系统图形库（Integrated Raster Imaging System Graphics Library，IRIS GL）”，一般简称为“GL”。1992 年，为了清理代码并且使 GL 更易维护和发展，SGI 开发了 OpenGL 并且设立了一个由若干公司组成的集团来负责维护 GL 的这一开放标准。而今，这一集团就是我们所熟知的“Khronos Group”，它是一个拥有 100 多位成员的公司。OpenGL ES 在 21 世纪初期被开发出来，使得这一开放库扩展到了嵌入式设备的领域，它是 OpenGL 的子集。此后不久开发出了 EGL，它为原始的平台图形提供公用接口层（common interface layer）。

在这个接口中，OpenGL 仅仅指 GL，在 OpenGL ES 中亦如此。在本章中，GL 通常指底层的对象和 OpenGL 中的接口，以便在代码中遵照统一的命名约定。OpenGL ES 通常指 Android 实现的 OpenGL 的子集 OpenGL ES。最后，OpenGL 用于指代库中一些更为普遍的概念。

17.1.1 在 Android 中使用 OpenGL ES

Android 开发人员可以通过以下两种方法实现 3D 图形。

① Android SDK 提供 OpenGL ES 功能，它涉及三个包：`android.opengl`、Khronos 的`javax.microedition.khronos.opengles`和`javax.microedition.khronos.egl`。

Download at http://www.pin5i.com/

② Android 本地开发包（Android Native Development Kit，NDK）可以使用 OpenGL ES 1.1 和 2.0 的本地库以提供最佳性能。

Note

在本章中，我们将焦点集中在使用 Android SDK 开发 OpenGL ES 应用程序上。我们将在第 18 章“使用 Android NDK”中讨论 Android 本地开发包的相关内容。

Android SDK 支持不同版本的 OpenGL ES，这依赖于 API Level 或者平台的版本。

① Android 1.0（API Level 1）及以上版本的设备都支持 OpenGL ES 1.0（`android.opengl`）。

② Android 1.6（API Level 4）及以上版本的设备都支持 OpenGL ES 1.1（`android.opengl.GLES11`）。

③ Android 2.2（API Level 8）及以上版本的设备都支持 OpenGL ES 2.0（`android.opengl.GLES20`）。

17.1.2 保证设备兼容性

需要使用 OpenGL 功能的应用程序应该在 Android Manifest 文件中使用`<uses-feature>`标记声明 `android:glEsVersion` 属性。这样像 Android Market 一类的应用程序商店可以根据应用程序所需的 OpenGL 版本对设备进行筛选过滤。`android:glEsVersion` 属性是一个 32 位的数字，其中高位字节代表主版本号，而低位则表示次版本号。

① 如果应用程序只需要 OpenGL ES 1.0，那么就不需要声明`<uses-feature>`标记，因为默认情况下每个应用程序都是支持 OpenGL ES 1.0 的。

② 如果应用程序需要 OpenGL ES 1.1，那么你需要像下面这样声明`<uses-feature>`标记：`<uses-feature android:glEsVersion="0x00010001" />`。

③ 如果应用程序需要 OpenGL ES 2.0，那么你需要像下面这样声明`<uses-feature>`标记：`<uses-feature android:glEsVersion="0x00020000" />`。

另外，在 Android Manifest 文件中，只能够列出一个版本的 OpenGL ES。对于一些在运行时检查支持的图形库，进而兼容多版本 OpenGL 的应用程序而言，需要指定一个最低的支持版本。不过，一般而言，支持高版本的 OpenGL ES（比如 2.0）的应用程序也能够兼容一些早期版本（如 1.1 和 1.0）。

17.2 在 Android SDK 中使用 OpenGL ES API

在 Android 下 OpenGL ES 的使用是 Android `View` 对象概念和常规 OpenGL ES 概念的综合。有很多的途径可以初始化并使用 Android SDK 中提供的 OpenGL ES 功能。

（1）开发人员可以实现自己的 OpenGL ES 解决方案，处理 EGL 和 GL 的初始化，管理 OpenGL ES 调用的各个辅助线程，以及在 `SurfaceView` 控件上进行绘制。

（2）对于 Android 1.5，开发人员可以利用 `GLSurfaceView` 和 `GLSurfaceView.Renderer` 类来帮助处理 EGL 的初始化以及线程管理。调用都在用户定义的 `Renderer` 类中进行。`Renderer` 类负责处理绘制和 GL 的初始化，并且它运行在 UI 线程之外。

在本章中，我们将给出上面两种方法的实例。虽然第二种方法（使用 `GLSurfaceView`）

Download at http://www.pin5i.com/

确实简单一些，但是跟着我们一步步按照第一种方法“手工”地实现，你会学习到更完整的 Android OpenGL ES 知识。另外，许多开发人员可能需要将代码移植到不同的平台下，所以，我们从“手工”方法开始，这样，我们可以比较详细地了解建立、绘制和销毁 OpenGL ES 的正确步骤。这两种方法所使用的概念和类是非常接近的，所以，即使你只使用 `GLSurfaceView` 方法搭建工程，了解其中的细节也是很有好处的。

Tip

本章中的许多代码示例都来自于 SimpleOpenGL 应用程序。在本书的网站上可以下载该应用程序的源代码。

17.3 手工处理 OpenGL ES 任务

对于那些需要在 Android 1.5 之前的版本上进行开发的用户，或者是需要对渲染管道（rendering pipeline）和初始化进行更严格控制的用户而言，我们将讲解不使用 `GLSurfaceView` 而实现 OpenGL 应用程序的方法。下面的步骤用于初始化 OpenGL ES，这样才能够通过 OpenGL 的接口实现屏幕绘制。

① 使用 `SURFACE_TYPE_GPU` 类型初始化 `SurfaceView`。
② 为 OpenGL 开启一个新的线程，所有的 OpenGL 调用都将在该线程上实施。
③ 初始化 EGL。
④ 初始化 GL。
⑤ 开始绘制。

Warning

当 OpenGL ES 在应用程序的一个线程中被初始化后，所有随后的调用都必须在这个线程上进行；否则，调用将不会成功。虽然你也可以使用应用程序的主线程，但额外的操作和循环会降低应用程序的响应性。这确实牵涉一些进程同步的问题，但我们必须处理它们，这将在本章稍后部分中讲解。

17.3.1 创建 SurfaceView

在屏幕上绘制 3D 图形的第一步是创建你自己的 `SurfaceView`。这包含对 `SurfaceView` 的继承和对 `SurfaceHolder.Callback` 回调的实现。下面是一段空的实现，我们将在稍后完善它：

```
private class BasicGLSurfaceView
    extends SurfaceView
    implements SurfaceHolder.Callback {

    SurfaceHolder mAndroidHolder;

    BasicGLSurfaceView (Context context) {
        super(context);
        mAndroidHolder = getHolder();
        mAndroidHolder.addCallback (this);
        mAndroidHolder.setType(
```

Download at http://www.pin5i.com/

```
            SurfaceHolder.SURFACE_TYPE_GPU);
    }

    public void surfaceChanged(SurfaceHolder holder,
        int format, int width, int height) {}

    public void surfaceCreated(SurfaceHolder holder) {}

    public void surfaceDestroyed(SurfaceHolder holder) {}
}
```

首先，在构造函数中，代码调用了 getHolder()方法，获取并存储 SurfaceHolder 对象。由于 SurfaceView 实现了 SurfaceHolder.Callback 接口，那么这个 SurfaceView 将被指定为接收事件回调的对象。最后，“平面（surface）”类型必须被设置为 SURFACE_TYPE_GPU 以供 OpenGL ES 的相关调用能够顺利进行。这个类在 Activity 中初始化并且被设置为 Content View，如下所示：

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate (savedInstanceState);
    mAndroidSurface = new BasicGLSurfaceView(this);
    setContentView(mAndroidSurface);
}
```

虽然将 SurfaceView 设置为整个 Content View 能够很好地工作，但是这样做毕竟不太灵活，特别是你想在屏幕上除了 3D 区域的地方实现其他操作的时候。有一种方法既能够让 SurfaceView 放置在屏幕上，同时又能享受到 XML 布局文件提供的便利，那就是使用一个容器 Widget（比如 FrameLayout），并在其中添加这个 View。例如，考虑下面的 FrameLayout 定义，它可以出现在任何布局中：

```
<FrameLayout
    android:id="@+id/gl_container"
    android:layout_height = "100px"
    android:lauout_width = "100px" />
```

上面的代码在屏幕上的某个区域放置了一个 100 像素×100 像素的方形，放置的具体位置依赖于布局其余的部分。现在，下面的代码使用这一 FrameLayout 的标识符将一个子 SurfaceView 放入其中。

```
mAndroidSurface = new TextureGLSurfaceView(this);
setContentView(R.layout.constrained);
FrameLayout v = (FrameLayout) findViewById(R.id.gl_container);
v.addView(mAndroidSurface);
```

在这个例子中，R.layout.constrained 是我们的布局资源，它包含了带有特定标识符的 FrameLayout，并且我们在程序中使用了它。随着我们继续 Android OpenGL ES 的初始化，你将明白为什么无论我们在 OpenGL“平面”中绘制什么，上面的代码始终能够有效运行。

17.3.2 启动 OpenGL ES 线程

在 Android 中，对屏幕的更新只能在应用程序的主线程中进行，有时这一主线程也被称为 UI 线程。但是，被我们用来在第二线程中分担某些图形处理工作的 SurfaceView Widget 却可以更新其屏幕区域。这就是 OpenGL 线程。与从 UI 线程中更新屏幕一样，所有的 OpenGL

Download at http://www.pin5i.com/

调用必须包含在相同的线程内。

前面提到过 SurfaceView 提供 SurfaceHolder.Callback 接口的实现。SurfaceView底层的平面只有在调用了 surfaceCreated()方法之后且在调用 surfaceDestroyed()方法之前才可以被访问。只有处在这两个调用之间的时间段内，我们才拥有一个可供 OpenGL 实例进行绘制的有效平面。

所以，在 surfaceCreated()方法被调用之前，我们不会创建 OpenGL 线程。下面的代码是对 surfaceCreated()方法的实现，它将开启一个 OpenGL 线程：

```
public void surfaceCreated(SurfaceHolder holder) {
    mGLThread = new BasicGLThread(this);
    mGLThread.start();
}
```

正如之前所提到的，这里比简单地运行线程要稍微复杂一些。SurfaceView 对象被传入了线程。这是因为 OpenGL 的调用需要知道具体在哪个 SurfaceView 上进行绘制。

BasicGLThread 类是 Thread 的一种实现，它包含了我们用于运行 OpenGL 线程的代码。下面的代码段显示了 BasicGLThread 的若干函数，另外，BasicGLThread 是 Activity 类的私有成员。

```
private class BaiscGLThread extends Thread {
    SurfaceView sv;
    BasicGLThread(SurfaceView view) {
        sv = view;
    }

    private boolean mDone = false;
    public void run() {
        initEGL();
        initGL();
        while (!mDone) {
            // 绘制代码
        }
    }

    public void requestStop() {
        mDone = true;
        try {
            join();
        } catch (InterruptedException e) {
            Log.e("GL", "failed to stop gl thread", e);
        }
        cleanupGL();
    }

    public void cleanupGL() { }
    public void initGL() { }
    public void initEGL() { }

    // 主 OpenGL 变量
}
```

在创建过程中，SurfaceView 被保存下来以供之后使用。在 run()方法中，对 EGL 和 GL 进行了初始化，我们将在本章稍后部分介绍它们。然后执行与绘制相关的代码，可能

Download at http://www.pin5i.com/

只执行一次，也可能像代码中所示意的那样通过一个循环来执行。最后，线程可以被安全地从外部通过调用 requestStop()方法所停止。这同时还会清理 OpenGL 的资源。在本章后面的“清理 OpenGL ES”部分中将讲解更多的内容。

17.3.3 初始化 EGL

至此，应用程序已经拥有了一个带有有效 Surface 的 SurfaceView 和一个刚刚被运行的 OpenGL 线程。对于大部分 OpenGL 的实现来讲，其第一步都是初始化 EGL，即原始硬件。每一次的操作基本上都是相同的，所以我们可以一次性地编写这段代码，然后重复使用它。在 Android 上初始化 EGL 必须执行以下步骤。

① 获得 EGL 对象。
② 初始化 Display。
③ 获取配置。
④ 将 EGLSurface 链接到 Android SurfaceView。
⑤ 创建 EGL 上下文（context）。
⑥ 告诉 EGL 使用哪个 Display、平面和上下文。
⑦ 获得可供渲染使用的 GL 对象。

Android SDK 提供了一些使用 OpenGL ES 的功能包（utility）类。第一个就是 GLDebugHelper 类。OpenGL 调用并不直接返回错误，而是在内部设置一个错误值，它可以通过查询得到。你可以使用 GLDebugHelper 类包装所有的 EGL 和 GL 调用，并且使用包装器（wrapper）来检测错误值并且将其抛出。获得 EGL 对象的调用就使用了这个包装器，如下所示：

```
mEGL = (EGL10) GLDebugHelper.wrap(
    EGLContext.getEGL(),
    GLDebugHelper.CONFIG_CHECK_GL_ERROR |
    GLDebugHelper.CONFIG_CHECK_THREAD,
    null);
```

这里，我们获取并且包装了 EGL10 对象。打开 CONFIG_CHECK_GL_ERROR 标签将会检测所有的 GL 错误。除此之外，打开了 CONFIG_CHECK_THREAD 标签后，包装器将确保所有的 GL 和 EGL 调用均来自正确的线程。

现在，我们可以对 Display 进行初始化，如下所示：

```
mGLDisplay = mEGL.eglGetDisplay(EGL10.EGL_DEFAULT_DISPLAY);
```

默认的 Display EGL10.EGL_DEFAULT_DISPLAY，由 Android OpenGL ES 实现的内部进行配置。在有了 Display 之后，我们就可以初始化 EGL 并且获得这一实现的版本了：

```
int[] curGLVersion = new int[2];
mEGL.eglInitialize(mGLDisplay, curGLVersion);
```

不同设备下的 GL 版本可能会不同，所以初始化了 Display 之后，我们可以查询哪一个配置是与我们所需配置最接近的：

```
int[] mConfigSpec = {   EGL10.EGL_RED_SIZE, 5,
                        EGL10.EGL_GREEN_SIZE, 6,
                        EGL10.EGL_BLUE_SIZE, 5,
                        EGL10.EGL_DEPTH_SIZE, 16,
                        EGL10.EGL_NONE };
```

Download at http://www.pin5i.com/

```
EGLConfig[] configs = new EGLConfig[1];
int[] num_config = new int[1];
mEGL.eglChooseConfig(mGLDisplay, mConfigSpec,
                     configs, 1, num_config);
mGLConfig = configs[0];
```

前面的配置对于模拟器和当前硬件设备均适用。如果你对所选的配置能否在应用程序的目标平台上正常工作不太确定的话，那么检查配置返回的列表是一个很好的办法。

现在，我们可以基于这一配置创建 EGL 平面了：

```
mGLSurface = mEGL.eglCreateWindowSurface
    (mGLDisplay, mGLConfig, sv.getHolder(), null);
```

前面提到过，我们将 SurfaceView 保存下来可供之后使用。这里，我们使用它将原始的 Android 平面传递给 EGL，这样它们能够正确地进行链接。在完成所有工作之前还需要获得 EGL 上下文和 GL 对象的实例。

```
mGLContext = mEGL.eglCreateContext(
    mGLDisplay, mGLConfig, EGL10.EGL_NO_CONTEXT, null);
```

现在，我们已经拥有了 Display、平面和上下文，那么可以获取 GL 对象了。

```
mEGL.eglMakeCurrent(mGLDisplay, mGLSurface,
    mGLSurface, mGLContext);
mGL = (GL10) GLDebugHelper.wrap(
    mGLContext.getGL(),
    GLDebugHelper.CONFIG_CHECK_GL_ERROR |
    GLDebugHelper.CONFIG_CHECK_THREAD, null);
```

我们再一次使用 GLDebugHelper 来包装 GL 对象，这样它能够检测错误并且验证所使用的线程。上面的代码完成了 Android 上对 EGL 的初始化。接下来，我们可以初始化 GL，以搭建工程并且设置其他的渲染选项。

17.3.4 初始化 GL

振奋人心的时刻到了。我们已经完整地初始化了 EGL，并且拥有了有效的 GL 对象，所以，现在可以初始化绘图空间（drawing space）了。在这个例子中，我们不会绘制太过复杂的东西，所以保留大多数选项的默认值。

通常，初始化 GL 所使用的首要方法调用之一为设置其“视口（viewport）”。下面的代码给出了示例，它设置了与 SurfaceView 同等尺寸的视口：

```
int width = sv.getWidth();
int height = sv.getHeight();
mGL.glViewport(0, 0, width, height);
```

屏幕上平面的具体位置由 EGL 在内部确定。我们还将使用 SurfaceView 的宽度和高度来确定 GL 所渲染的纵横比。在下面的代码中，我们完成了对 GL 的基本配置：

```
mGL.glMatrixMode(GL10.GL_PROJECTION);
mGL.glLoadIdentity();
float aspect = (float) width/height;
GLU.gluPerspective(mGL, 45.0f, aspect, 1.0f, 30.0f);
mGL.glClearColor(0.5f, 0.5f, 0.5f, 1);
```

Android SDK 提供了若干类似于 GLUT（OpenGL Utility Toolkit）中的帮助类。这里，我们

Download at http://www.pin5i.com/

使用其中的一个来定义视图（perspective），包括垂直角度、纵横比以及远近剪切面（near and far clipping planes）。gluPerspective()方法对于配置投影矩阵（projection matrix）非常有用，投影矩阵可以将 3D 场景转换为一个平面（flat surface）。最后，我们使用灰色清除整个屏幕区域。

17.3.5 屏幕绘制

既然 EGL 和 GL 对象都已经初始化好了，那么我们可以开始在屏幕上绘制物体了。在这个例子中，为了证明所有的准备工作都已就绪，并且可以开始绘制，我们在屏幕上放置一个三顶点（three-vertex）平面（通俗地讲即“三角形”）。下面给出了示例代码：

```
mGL.glMatrixMode(GL10.GO_MODELVIEW);
mGL.glLoadIdentity();
GLU.gluLookAt(mGL, 0, 0, 10f, 0, 0, 0, 0, 1, 0f);
mGL.glColor4f(1f, 0f, 0f, 1f);
while (!mDone) {
        mGL.glClear(GL10.GL_COLOR_BUFFER_BIT |
            GL10.GL_DEPTH_BUFFER_BIT);
        mGL.glRotatef(1f, 0, 0, 1f);
        triangle.draw(mGL);
        mEGL.eglSwapBuffers(mGLDisplay,
mGLSurface);
    }
```

也许你已经注意到了，上面的代码似乎缺了什么。这段代码并没有显示绘制三角形的命令。不过，它的确使用了 Android SDK 的功能包方法 gluLookAt()来变换模型视图矩阵（model view matrix）。这里，它将观察点位置设为离原点 10 个单位，并且朝向原点方向，这个值通常设定为正的 *y* 轴方向。在循环中，注意到并没有指定特征矩阵（identity matrix）。这给 glRotatef()方法带来了复合的效果，使三角形能够以逆时针方向旋转。在下一节“绘制 3D 物体”中，我们将讨论在 Android 中使用 OpenGL ES 进行绘制的细节。

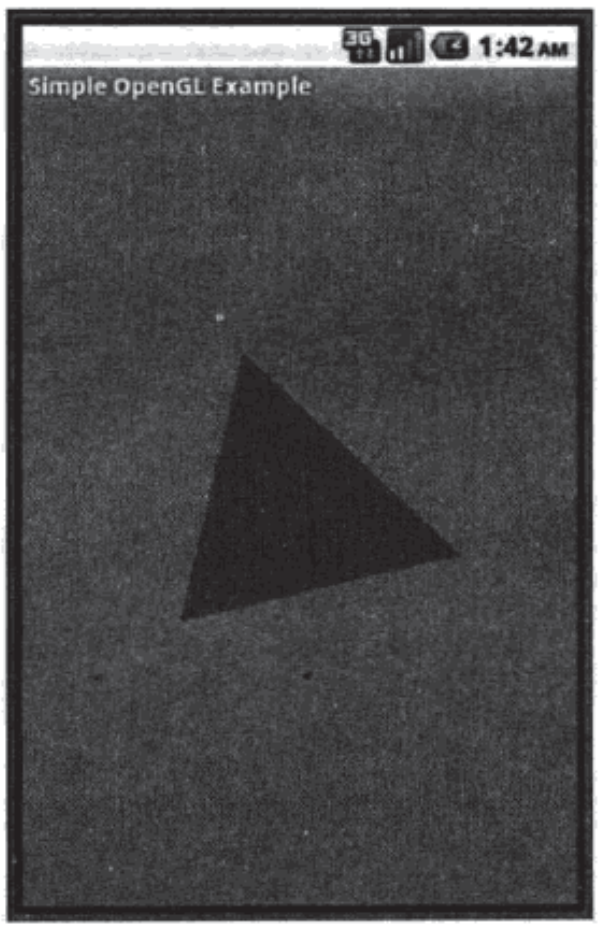


图 17.1 在 Android 模拟器中使用 OpenGL ES 渲染的红色三角形

运行上面的代码后，将看到类似图 17.1 所示屏幕界面。

现在，你已经搭建好了 Android SDK 中一个可用的 OpenGL ES 环境。我们将从这里开始，讲解更多有关在这个环境中进行绘制的话题。

17.4 绘制 3D 物体

既然你已经在 Android 中搭建好了一个能够顺利工作的 OpenGL ES 环境，那么是时候开始着手进行一些实际的绘制了。这一节将带领你浏览若干个示例，每一个均构建在前面的基础上，它们都引入了 OpenGL ES 中 Android 特有的新概念。

17.4.1 绘制顶点（Vertices）

OpenGL ES 支持两个基本的绘制调用：glDrawArrays()和 glDrawElements()。这两个方法都需要一个由 glVertextPointer()调用关联的顶点缓冲（vertex buffer）。由于 Android 运行在 Java 上，所以不能随意地传入一个数组，因为数组的内容可能会在内存中移

Download at http://www.pin5i.com/

动。这样，我们必须使用 ByteBuffer、FloatBuffer 或者 IntBuffer 以使数据保持在固定的内存位置。将各种类型的数组转换成缓冲是很普遍的，所以我们实现了一些辅助方法。下面给出了一个将浮点型数组转换为 FloatBuffer 的代码示例：

```
FloatBuffer getFloatBufferFromFloatArray(float array[]) {
    ByteBuffer tempBuffer =
        ByteBuffer.allocateDirect(array.length * 4);
    tempBuffer.order(ByteOrder.nativeOrder());
    FloatBuffer buffer = tempBuffer.asFloatBuffer();
    buffer.put(array);
    buffer.position(0);
    return buffer;
}
```

上面的代码创建了一个 32 比特的浮点型缓冲，其元素间间隙（stride）为 0。返回的 FloatBuffer 可以存储下来并传入 OpenGL 的调用中。下面的示例演示了这一操作，它使用了我们前面接触过的三角形：

```
float[] vertices = {
    -0.559016994f, 0, 0,
    0.25f, 0.5f, 0f,
    0.25f, -0.5f, 0f
};
mVertexBuffer = getFloatBufferFromFloatArray(vertices);
```

通过关联的缓冲，我们可以进行三角形的绘制了，如下所示：

```
void drawTriangle(GL10 gl) {
    gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
    gl.glVertexPointer(3, GL10.GL_FLOAT, 0, mVertexBuffer);
    gl.glDrawArrays(GL10.GL_TRIANGLES, 0, 3);
}
```

我们需要开启 GL_VERTEXT_ARRAY 状态，虽然这也可以在 GL 配置中完成，但这是使用 OpenGL ES 进行绘制所必需的。之后，我们将顶点缓冲传入了 glVertexPointer() 方法，并且告诉 GL 我们使用的是浮点值。另外，顶点值所对应的参数为 GL_FIXED，也可以使用它，这样某些 Android 实现可能会快一些。最后，glDrawArrays() 调用通过 3 个顶点绘制了一个三角形。运行结果如图 17.1 所示。

17.4.2 为顶点着色

在 OpenGL ES 中，你可以使用一个颜色数组为每个将要绘制的顶点分别赋予色彩。这可以通过调用 glColorPointer() 方法来实现，它使用“颜色缓冲（buffer of colors）”。下面的代码为三个顶点生成了一个颜色缓冲：

```
float[] colors = {
    1f, 0, 0, 1f,
    0, 1f, 0, 1f,
    0, 0, 1f, 1f,
    };
```

有了可用的缓冲后，我们就可以使用它来为三角形着色了，如下面的代码所示：

```
void drawColorful(GL10 gl) {
    gl.glEnableClientState(GL10.GL_COLOR_ARRAY);
```

Download at http://www.pin5i.com/

```
        gl.glColorPointer(4,GL10.GL_FLOAT, 0, mColorBuffer);
        draw(gl);
        gl.glDisableClientState(GL10.GL_COLOR_ARRAY);
    }
```

首先，代码开启了 GL_COLOR_ARRAY 客户状态（client state）。之后，调用 glColorPointer()方法关联了刚刚创建的颜色缓冲。调用 draw()方法在屏幕上绘制出了如图 17.2 所示的三角形。

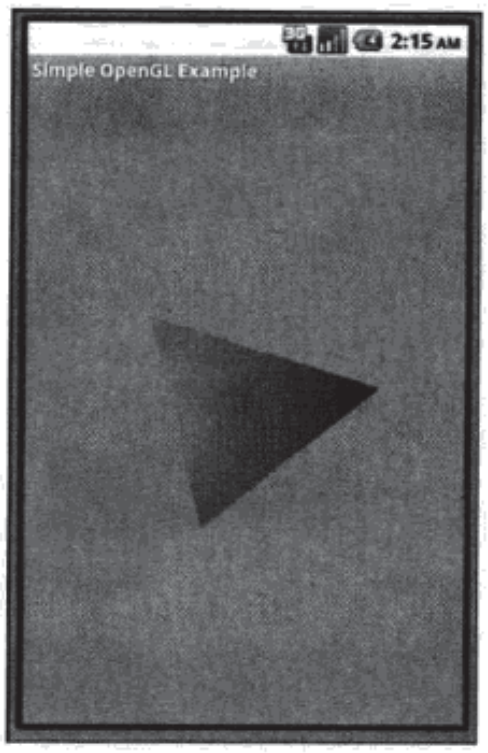


图 17.2 顶点分别为红、绿、蓝的平滑渐变三角形

17.4.3 绘制更多复杂物体

一个标准的立方体拥有 8 个顶点。但是在 OpenGL ES 中，6 个面中的每一个面都需要使用两个三角形来绘制。而每一个三角形需要 3 个顶点。所以，这加起来总共需要 36 个顶点来绘制一个仅仅只需要 8 个点表示的物体。一定有更好的办法来完成这个任务。

OpenGL ES 支持索引数组（index arrays）。一个索引数组是基于当前顶点数组的一个顶点索引列表。索引数组必须是一个缓冲，在这个例子中，我们使用一个 ByteBuffer，因为并没有太多的顶点需要指定。索引数组列出了使用 glDrawElements()方法绘制顶点的顺序。注意颜色数组（以及我们稍后将会看到的普通数组）仍然与顶点数组相关联，而不是索引数组。下面是仅使用 8 个定义好的顶点来绘制 OpenGL 立方体的代码：

```
float vertices[] = {
    -1,1,1, 1,1,1, 1,-1,1, -1,-1,1,
    1,1,-1, -1,1,-1, -1,-1,-1, 1,-1,-1
};
byte indices[] = {
    0,1,2, 2,3,0, 1,4,7, 7,2,1, 0,3,6, 6,5,0,
    3,2,7, 7,6,3, 0,1,4, 4,5,0, 5,6,7, 7,4,5
};
FloatBuffer vertexBuffer =
    getFloatBufferFromFloatArray(vertices);
ByteBuffer indexBuffer =
    getByteBufferFromByteArray(indices);
gl.glVertexPointer(3, GL10.GL_FLOAT, 0, vertexBuffer);
gl.glDrawElements(GL10.GL_TRIANGLES, indices.length,
    GL10.GL_UNSIGNED_BYTE, indexBuffer);
```

Download at http://www.pin5i.com/

顶点定义了立方体的大体形状。然后，我们使用索引数组定义了 36 个顶点的绘制顺序，这样它可以绘制出 12 个三角形并构成我们所需的立方体（前面提到过 OpenGL ES 并不支持四边形）。现在，屏幕上应该能够出现一个如图 17.3（左）所示那样的红色图形。不过它看起来并不特别像一个立方体，在没有阴影的情况下，它看起来更像一个随机生成的多边形。另外，在你将 `glDrawElements()` 中的参数修改为 `GL_LINE_LOOP` 而非 `GL_TRIANGLES` 之后，将会看到一个线框版本的图形，如图 17.3（右）所示。现在你能够发现它确实是一个立方体了。顶点缓冲可以被不同的索引缓冲所重用。这在某些情况下非常有用，比如你需要使用相同集合的顶点来定义多个图形，之后通过变形（transformation）将其绘制到自己所在的位置。

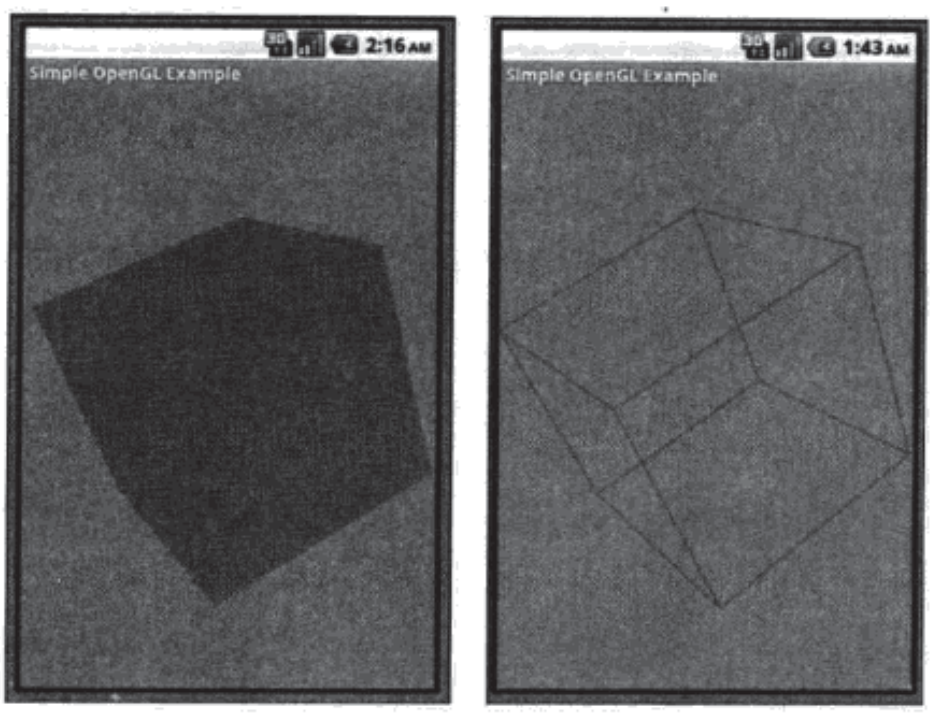


图 17.3 不带阴影的实体立方体（左）和用线框表示的相同立方体（右）

17.4.4 照亮你的场景

前面绘制的 3D 物体是一个立方体，但它看起来却像平面（2D 屏幕）上一个奇怪的多边形。每一个面可以通过在每次绘制面的调用之间应用颜色设置来绘制成不同颜色。但是，这样仍然会绘制出一个看起来非常“平板”的立方体。那么，为什么不能使用光源来点亮整个场景以使立方体看起来有一定的景深呢？

在为场景提供光源之前，每一个面上的每一个顶点都需要指定一个向量，以定义光线的反射和渲染方式。虽然我们可以为顶点指定任意向量，但通常还是与其所定义的面相垂直，这被称作面的法线方向。回顾我们前面立方体的例子，现在可以理解为什么一个立方体不能只由 8 个顶点来定义了，因为这样每个顶点只能拥有一个法线向量，所以我们必须使用 3 个顶点，这样每一个顶点可以分属不同的面，那么一个立方体事实上需要包含 24 个顶点（从技术上讲，你可以定义一组索引数组，然后在每次面绘制调用之间修改法线数组，不过更常用的做法是使用一个较大的列表来包含顶点，并且使用一个单独的法线向量数组）。

如同颜色数组一样，法线数组也按顺序应用于顶点数组中的每一个顶点。使用光源照亮场景是一个相对复杂的话题，如果你不太熟悉这些知识，那么可以查看本章末“参考资料及更多信息”部分中提供的材料链接，在那里你可以学习到更多的知识。对于现在而言，我们只给出一个在 Android 中使用 OpenGL ES 光源的简单示例。

下面的代码使用了简单的光源：

```
mGL.glEnable(GL10.GL_LIGHTING);
mGL.glEnable(GL10.GL_LIGHT0);
```

Download at http://www.pin5i.com/

```
mGL.glLightfv(GL10.GL_LIGHT0, GL10.GL_AMBIENT,
    new float[] {0.1f, 0.1f, 0.1f, 1f}, 0);
mGL.glLightfv(GL10.GL_LIGHT0, GL10.GL_DIFFUSE,
    new float[] {1f, 1f, 1f, 1f}, 0);
mGL.glLightfv(GL10.GL_LIGHT0, GL10.GL_POSITION,
    new float[] {10f, 0f, 10f, 1f}, 0);
mGL.glEnable(GL10.GL_COLOR_MATERIAL);
mGL.glShadeModel(GL10.GL_SMOOTH);
```

上面的代码使用了光照，开启了 GL_LIGHT0，之后设置了光源的颜色和亮度。最后，光源被放置在了 3D 场景中的某个位置。除此之外，我们还开启了 GL_COLOR_MATERIAL，这样在光照时将结合绘制物体时所使用的颜色。另外，代码还应用了平滑阴影模型（smooth shading model），它将会避免同一个面上的三角形之间出现视觉上的罅隙。材质颜色的定义会使光照效果看起来更别致，而且更具有真实感，不过这个话题已经超出了本书的范围。

下面是立方体的绘制代码，假设我们现在已经有了一个含有 24 个点的完整顶点数组和定义了绘制顺序的索引数组：

```
gl.glEnableClientState(GL10.GL_NORMAL_ARRAY);
gl.glVertexPointer(3, GL10.GL_FLOAT, 0, mNormalBuffer);
gl.glDraeElements(GL10.GL_FLOAT, 0, mNormalBuffer)
gl.glDrawElements(GL10.GL_TRIANGLES, indices.length,
GL10.GL_UNSIGNED_BYTE, mIndexBuffer);
```

注意上面的代码使用了法线数组和法线模式，否则，光照效果看起来将显得不正常。和其他的数组一样，法线数组也需要使用 Java 中的固定缓冲，如下面的代码所示：

```
float normals[] = {
    // 前
    0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,
    // 后
    0, 0, -1, 0, 0, -1, 0, 0, -1, 0, 0, -1,
    // 顶
    0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
    // 底
    0, -1, 0, 0, -1, 0, 0, -1, 0, 0, -1, 0,
    // 右
    1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
    // 左
    -1, 0, 0, -1, 0, 0, -1, 0, 0, -1, 0, 0 };
mNormalBuffer = getFloatBufferFromFloatArray(normals);
```

上面的代码使用了我们前面讨论过的一个辅助方法创建了一个 FloatBuffer。我们使用了浮点数组来定义法线。代码同时还显示了法线的定义方式以及它如何同每一个顶点相关联。（注意到我们现在已经为这个立方体设置了 24 个顶点。）另外，你可以将法线向量设置为稍微偏离平面的垂直方向，这可以制造出各种不同的光照效果，不过如果要达到更为精确的光照效果的话，就需要增加物体所包含的多边形数量，或者为其赋予材质。图 17.4 显示了一个实实在在的立方体，现在它具有了很好的景深效果。

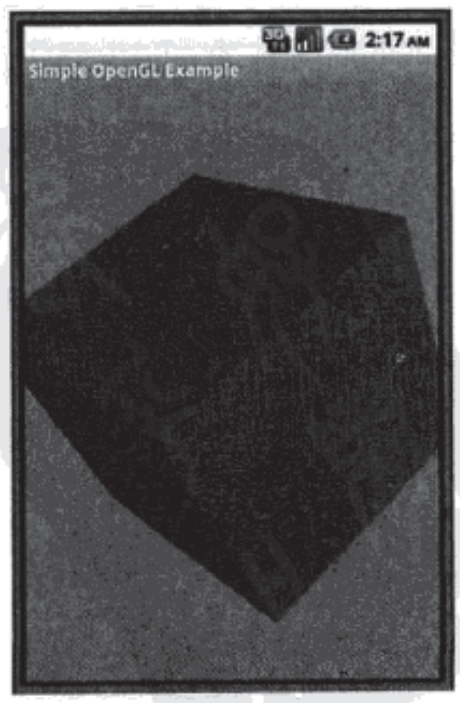


图 17.4 带有自右向左光源照射的立方体

Download at http://www.pin5i.com/

17.4.5 为对象赋予材质

为表面赋予材质，或者说在表面进行贴图，是一个非常复杂的话题，需要“长篇大论”。不过只学习在 Android 中使用材质就足够了，所以我们继续使用前面带有颜色和光照的立方体，并为它赋予材质。

首先，需要开启材质功能，如下面的代码所示：

```
mGL.glEnable(GL10.GL_TEXTURE_2D);
int[] textures = new int[1];
mGL.glGenTextures(1, textures, 0);
```

上面的代码开启了材质功能，并且为一个材质创建了一个内部命名的“空槽（slot）”。我们将在下一段代码中通过这个空槽来告诉 OpenGL 它需要操作的材质。

```
gl.glBindTexture(GL10.GL_TEXTURE_2D, textures[0]);
Bitmap bitmap = BitmapFactory.decodeResourse(
    c.getResources(), R.drawable.android);
Bitmap bitmap256 = Bitmap.createScaledBitmap(
    bitmap, 256, 256, false);
GLUtils.texImage2D(GL10.GL_TEXTURE_2D, 0, bitmap256, 0);
bitmap.recycle();
bitmap256.recycle();
```

也许你现在正在分析上面的 Android 代码，那么，我们从头开始。OpenGL ES 需要使用位图来充当材质。幸运的是，Android 自身带有一个 `Bitmap` 类，它可以读取几乎所有格式的图像，包括 PNG、GIF 和 JPG 文件等。你也可以直接从 `Drawable` 资源标识符中获取，上面的代码就使用了这种方式。OpenGL 还需要所使用的材质必须为正方形且边长为 2 的幂，比如 64×64 或 256×256。由于源图像可能不具有上面所要求的精确尺寸，我们使用一个简单的 Android 方法调用即可将其进行缩放。而如果源图像不是准确的正方形，那么它将失去原有纵横比，以满足缩放后为一个正方形。不过有些时候维持原有的纵横比进行缩放然后使用某个颜色填充边缘显得更容易，但这超出了这个示例的范畴。

最后，代码使用 `GLUtils.texImage2D()` 方法将一个 Android `Bitmap` 关联为 OpenGL 的材质。OpenGL 将在内部进行保存，所以我们可以调用 `recycle()` 方法清理掉使用过的 Bitmap 对象。

既然我们已经为 OpenGL ES 设置了材质，那么下一步就是告诉它需要将材质绘制在哪里。这可以使用材质坐标缓冲来实现，它与其他所有需要关联到固定的 Java 缓冲中的数组相似。下面是针对我们的立方体示例的代码：

```
float texCoords[] = {
    1,0, 1,1, 0,1, 0,0,
    1,0, 1,1, 0,1, 0,0,
    1,0, 1,1, 0,1, 0,0,
    1,0, 1,1, 0,1, 0,0,
    1,0, 1,1, 0,1, 0,0,
    1,0, 1,1, 0,1, 0,0,
    };
mCoordBuffer = getFloatBufferFromFloatArray(texCodes);
gl.glEnableClientState(GL10.GL_TEXTURE_COORD_ARRAY);
gl.glTexCoordPointer(2, GL10.GL_FLOAT, 0, mCoordBuffer);

draw(gl);
```

Download at http://www.pin5i.com/

正如之前所讲的，这段代码为材质坐标创建了一个固定的缓冲。我们为每一个面均设置相同的值，这样每一个顶点都拥有关联的坐标。((0, 0)点为材质的左下角，(1, 1)为右上角。）之后，我们打开 GL_TEXTURE_COORD_ARRAY 状态并且告诉 OpenGL 使用哪一个缓冲。最后，代码绘制了立方体，它将产生如图 17.5（左）所示的输出。可以看到，虽然有了材质，但物体本身的颜色设置仍然在发挥作用。如果去掉了颜色的应用，其输入应该像图 17.5（右）那样。

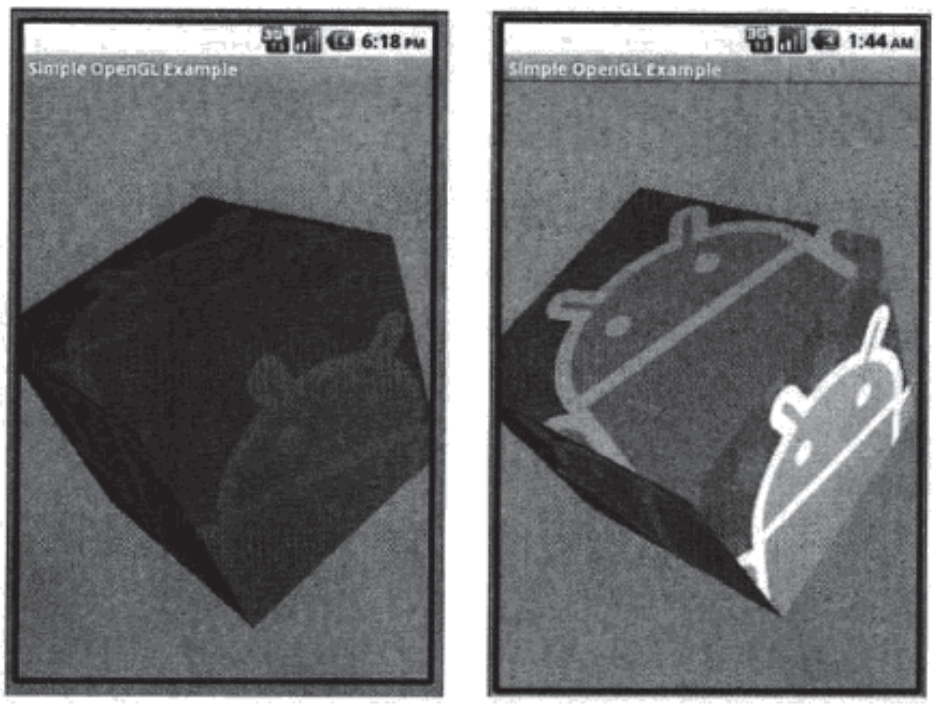


图 17.5 带有材质的红色立方体（左）和无颜色设置的相同立方体（右）

17.5 同 Android 视图和事件进行交互

现在你已经遍历了 Android OpenGL ES 的介绍，并且在屏幕上绘制了 3D 物体。事实上，这些 3D 物体都绘制在 SurfaceView 上面，它拥有所有 View Widget 的典型 Android 属性。现在我们使用这些属性同应用程序其余的部分进行交互。

首先，我们讲解如何从 OpenGL 线程向主线程回传信息以监视其行为。之后，我们给出一个示例，它演示了如何从主线程向 OpenGL 线程发送按键事件，以控制屏幕上的动画。

17.5.1 允许 OpenGL 线程向应用程序线程发送信息

Android SDK 为在另外的线程中运行代码提供了一个辅助类。Handler 类可以允许一段代码运行在目标线程上，即 Handler 初始化所在的线程。为了示例的目的，我们将其实现在一个 Activity 类中。

```
public final Handler mHandler = new Handler();
```

这使得 OpenGL 线程可以通过调用 Handler 的 post() 方法在 Activity 线程上执行代码。这允许我们操作屏幕上的其他 View 对象，而它们不能够被 Activity 线程之外的 OpenGL 线程所操作。对于这个例子，屏幕渲染的帧率将由 OpenGL 线程计算，并且回传给 Activity 线程。下面的方法实现了这一功能：

```
public void calculateDisplayFPS() {
    if (showFPS) {
        long thisTime = System.currentTimeMillis();
        if (thisTime - mLastTime < mSkipTime) {
            mFrames++;
        } else {
```

Download at http://www.pin5i.com/

```
                mFrames++;
                final long fps =
                    mFrames / ((thisTime-mLastTime)/1000);
                mFrames = 0;
                mLastTime = thisTime;
                mHandler.post(new Runnable() {
                    public void run() {
                        mFPSText.setText("FPS = " + fps);
                    }
                });
            }
        }
    }
```

calculateAndDisplayFPS()方法从 OpenGL 线程上的动画循环中被调用。其数学原理相对比较直接：总的帧数除以它们的总时值，以秒为单位。之后我们将这个数值回传给 Activity 进程中的 Handler，它创建一个新的 Runnable 对象，并将一个包含当前帧率信息的 String 传给 TextView。

无论怎样，每一次这样的循环迭代实质上都将导致性能的下降。计数器跟踪了已绘制的帧数，我们利用它进行了计算，并且在每次 mSkipTime 的时长过去之后显示了计算出的结果。实验表明，5000ms 是一个比较合适的值，它不会因为计算性能反而过多地影响性能。图 17.6 显示了包含帧率的结果。

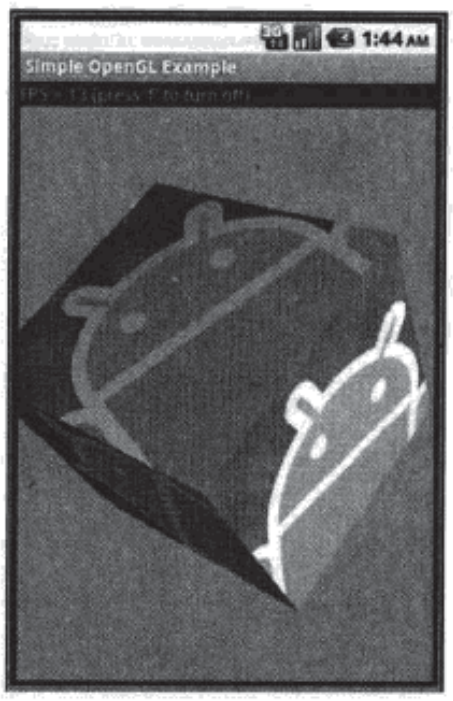


图 17.6 带有帧率显示、材质、光照和阴影的立方体

17.5.2 允许应用程序线程向 OpenGL 线程发送信息

现在我们来关注反过来的情况。我们需要应用程序主线程与 OpenGL 线程进行通信。这可以使用一个 Handler 向 OpenGL 线程传送需要运行的代码。不过，如果我们并不需要执行 OpenGL 代码，那么就不需要在 OpenGL 线程上下文中执行它。相反，我们可以为 SurfaceView 加入一个按键事件处理器（handler）来加速或中断 OpenGL 线程中的动画。

SurfaceView 在接收按键事件之前必须获得焦点。有两种方法对此进行设置：

```
setFocusable(true);
setFocusableInTouchMode(true);
```

为两种触摸模式均设置 Focusable 属性可以使其始终能够接收到按键事件，而无论其处于何种模式下。现在，我们在 SurfaceView 中实现这个按键事件处理器。首先，我们实现一个用于开关帧率显示的处理器。下面是一种对 onKeyDown()方法覆写的实现示例：

```
public boolean onKeyDown(int keyCode, KeyEvent event) {
    switch (keyCode) {
        case KeyEvent.KEYCODE_F;
            mGLThread.toggleFPSDisplay();
        return true;
    }
    return super.onKeyDown(KeyCode, event);
}
```

当用户按下“F”键，OpenGL ES 线程中的 toggleFPSDisplay()方法将被调用。它仅仅改变一个 boolean 标签的状态。如果用户一直按住不放，那么 onKeyDown()方法将

Download at http://www.pin5i.com/

会一直被调用，从而不断地开/关帧率的显示。有很多方式可以避免这种现象发生，比如只在onKeyUp()中执行这一操作，或者使用不同的按键来开启或关闭标签状态。

我们提供给用户的下一个控制方式是通过按下“P”键来暂停动画。可以在 onKeyDown()中加入下面的条件检测表达式：

```
case KeyEvent.KEYCODE_P:
    mGLThread.setAnim(false);
    return true;
```

这里，无论 onKeyDown()被调用了多少次，状态值都将被强制地设为 false。下一步，当用户释放按键时，在 onKeyUp()的实现中恢复了动画。

```
public boolean onKeyUp(int keyCode, KeyEvent event) {
    switch (keyCode) {
        case KeyEvent.KEYCODE_P:
            mGLThread.setAnim(true);
            return true;
    }
    return super.onKeyUp(keyCode, event);
}
```

又一次地，我们将状态值强制设为 true，这样，当用户放开按键时动画都会恢复，而无论当前的状态是什么。在这个例子中，对整个 while()动画循环的内部使用一个 if 表达式则可以暂停整个渲染。

在这几个示例中，都没有通过实际地在 OpenGL 线程中运行代码来改变其标签状态。使用这种方式是基于以下几条理由。

① 能够独占性地（exclusively）对值进行设定（没有并发性的问题）。

② 在循环中标签的具体状态并不重要。

③ 没有对 OpenGL 的调用。

前两条理由的含义在于我们无需为程序的正常和安全运行而执行线程同步操作。最后一条理由意味着我们无需在 OpenGL 线程上创建一个 Handler 来执行在适当线程上的 OpenGL 调用。当然，在很多情况下上面的条件是不存在的，不过，讨论线程同步的话题超出了本章的范畴，但是，标准的 Java 方法可以用来解决这一问题。

17.6 清理 OpenGL ES

在应用程序使用完 OpenGL 后有必要对其进行清理。这发生在应用程序需要退出或 Activity 因为某种原因而改变的时候。推荐的“优雅”关闭 OpenGL 的过程是重置并销毁你配置的“平面（Surface）”和上下文，继而终止 EGL 实例。下面的代码演示了这一过程：

```
private void cleanupGL() {
    mEGL.eglMakeCurrent(mGLDisplay, ELG10.EGL_NO_SURFACE,
        EGL10.EGL_NO_SURFACE, EGL10.EGL_NO_CONTEXT);
    mEGL.eglDestroySurface(mGLDisplay, mGLSurface);
    mEGL.eglDestroyContext(mGLDisplay, mGLContext);
    mEGL.eglTerminate(mGLDisplay);
}
```

首先，eglMakeCurrent()移除我们使用过的平面和上下文。接着，eglDestroy Surface()和 eglDestroyContext()释放 OpenGL 为平面和上下文保有的所有资源。最

Download at http://www.pin5i.com/

后，通过调用 eglTerminate() 终止了 OpenGL。如果 OpenGL 运行在另外的线程中，那么这个线程此时也可以被终止了。

适当地清理 OpenGL 完全依赖于应用程序自身。Android 没有像对 Cursor 对象等那样对 OpenGL 提供任何辅助类，以帮助其在 Android 生存周期中自动进行清理。

17.7 使用 GLSurfaceView

Android 1.5（API Level 3）中引入了几个针对 OpenGL ES 的辅助类，可以用来简化应用程序的 OpenGL ES 实现。GLSurfaceView 和 GLSurfaceView.Renderer 类能够简化代码的编写，这样你可以将更多的精力集中在实际的 GL 绘制过程，而非处理 OpenGL ES 调用的各种细节。从本质上讲，GLSurfaceView 类负责处理 EGL 初始化、线程操作和对用户定义的 Renderer 类的调用，而 Renderer 类则处理具体的绘制和 GL 的初始化。

Tip

这一部分中的许多代码示例都来自于 SimpleOpenGL 应用程序中的 ShowAndroidGLActivity.java 类。在本书的网站上可以下载它的源代码。

你可以通过继承或者直接实例化来使用这个类。无论哪种实现，都需要为其提供一个 GLSurfaceView.Renderer 实现，以包含用于绘制和 GL 初始化的回调方法。Activity 必须向 GLSurfaceView 传送 onPause() 和 onResume() 事件。EGL 的初始化由 GLSurfaceView 对象处理，同时与主线程分离的线程用来减轻其处理负荷。使用它可以有效地减轻代码的编写量，以使开发人员能够专注于实际的 GL 绘制部分。

下面给出了整个 Activity 的代码，它复制了我们在本章前面部分实现的彩色三角形（如图 17.2 所示）：

```
public class AndroidOpenGL extends Activity {
CustomSurfaceView mAndroidSurface = null;

protected void onPause() {
    super.onPause();
    mAndroidSurface.onPause();
}

protected void onResume() {
    super.onResume();
    mAndroidSurface.onResume();
}

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    mAndroidSurface = new CustomSurfaceView(this);
    setContentView(mAndroidSurface);
}

private class CustomSurfaceView extends GLSurfaceView {
    final CustomRenderer mRenderer = new CustomRenderer();

    public CostomSurfaceView(Context context) {
        super(context);
```

Download at http://www.pin5i.com/

```
        setFocusable(true);
        setFocusableInTouchMode(true);
        setRenderer(mRenderer);
    }

    public boolean onKeyDown(int keyCode, KeyEvent event) {
        switch (keyCode) {
        case KeyEvent.KEYCODE_P:
            queueEvent (newRunnable() {
                public void run() {
                    mRenderer.togglePause();
                }
            });
            return true;
        }
        return super.onKeyDown(keyCode, event);
    }
}

private class CustomRenderer implements
    GLSurfaceView.Renderer {
    TriangleSmallGLUT mTriangle = new TriangleSmallGLUT(3);
    boolean fAnimPaused = false;

    public void onDrawFrame(GL10 gl) {
        if (!fAnimPaused) {
            gl.glClear(GL10.GL_COLOR_BUFFER_BIT |
                GL10.GL_DEPTH_BUFFER_BIT);
            gl.glRotatef(1f, 0, 0, 1f);

            if (mTriangle != null) {
                mTriangle.drawColorful(gl);
            }
        }
    }

    public void togglePause() {
        if (fAnimPaused == true) {
            fAnimPaused = false;
        } else {
            fAnimPaused = true;
        }
    }

public void onSurfaceChanged(GL10 gl, int width,
    int height) {
    gl.glViewport(0, 0, width, height);

    // configure projection to screen
    gl.glMatrixMode(GL10.GL_PROJECTION);
    gl.glLoadIdentity();
    gl.glClearColor(0.5f, 0.5f, 0.5f, 1);
    float aspect = (float) width / height;
    GLU.gluPerspective(gl, 45.0f, aspect, 1.0f, 30.0f);
}

public void onSurfaceCreated(GL10 gl,
    EGLConfig config) {
    gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
```

Download at http://www.pin5i.com/

```
            // configure model space
            gl.glMatrixMode(GL10.GL_MODELVIEW);
            gl.glLoadIdentity();
            GLU.gluLookAt(gl, 0, 0, 10f, 0, 0, 0, 0, 1, 0f);
            gl.glColor4f(1f, 0f, 0f, 1f);
        }
    }}
```

正如你所看到的，上面这段代码演示了创建一个新的 GLSurfaceView 和 GLSurfaceView.Renderer 的方法。结合适当的三角绘制类（包含于本书的代码中，且在本章稍前部分有所讲解）的实现，代码最终运行的结果是一个旋转的三角，在用户按下“P”键时可以被暂停。GLSurfaceView 的实现包含了它自身的 Renderer，这与通过外部关联的方式相比不太普遍。这两个类必须紧密地结合使用。

GLSurfaceView 通过覆写常规 View 类的 onKeyDown() 方法来实现按键处理。这一操作通过一个称为 queueEvent() 的辅助方法被传入 Renderer。queueEvent() 方法向 GLSurfaceView 控制的 Renderer 线程中传入一个 Runnable 对象。

接下来，Renderer 的实现提供了在 onDrawFrame() 方法中进行绘制的方法。它既可以不断地被调用，也可以在接收到指令后被调用，这依赖于通过 GLSurfaceView.SetRenderMode() 方法所设定的渲染模式。我们在 onSurfaceChanged() 方法中建立屏幕投影（projection），这是一个很好的地方，因为在“平面（surface）”的方向和尺寸改变时该方法将被调用。之后，在 onSurfaceCreated() 方法中，执行了基本的 GL 配置，包括设置客户状态（client state）和静态数据，如模型视图（model view）等。

现在，所有的 EGL 配置均在 GLSurfaceView 内部完成了，所以应用程序不用对此过于担心。但是，如果应用程序需要对 EGL 进行自定义的配置，那么也可以将配置参数传入 onSurfaceCreated() 方法中。

如果你决定使用上面的这种方法在 Android 中建立 GL 平面，那么实现的所有渲染代码都不需要有丝毫更改。

17.8 使用 OpenGL ES 2.0

Android 从 Android 2.2（API Level 8）开始支持 OpenGL ES 2.0，尽管如此，使用 Android NDK Release 3 的应用程序仍然可以在低至 API Level 5 的平台上使用 2.0 所提供的功能。在这一节中，我们将讨论针对 OpenGL ES 2.0 的 Android Java API 支持。另外，它也向下兼容，支持 OpenGL ES 1.x，但 OpenGL ES 2.0 本身并不兼容 OpenGL ES 1.x。不同版本的 OpenGL ES 提供不同的方法来处理 3D 图形。

① OpenGL ES 1.x 提供一个固定的函数来进行渲染和材质处理。也就是说，用来进行变形、控制灯光和着色的函数是同一个。

② OpenGL ES 2.0 使用定点和局部面（fragment）着色程序替代了之前单一的函数，当然，这需要你，也就是开发人员来完成。自己编写着色程序，一方面扩展灵活性，另一方面也给开发增添了一些难度。

选择哪个 OpenGL ES 版本取决于你自己。在本节中，我们将向你展示如何初始化并且实现一个简单的 OpenGL ES 2.0 程序，而使用 NDK 的方法将在第 18 章中讨论。

Download at http://www.pin5i.com/

Tip

这一部分中的许多代码示例都来自于 SimpleOpenGL2 应用程序。在本书的网站上可以下载该应用程序的源代码。

17.8.1 为 OpenGL ES 2.0 配置应用程序

如果你将要使用 Android OpenGL ES 2.0 API，而不打算支持其他的可选功能，那么你需要在 Manifest 文件中指定两项：通过<uses-sdk>标记指定应用程序需要 Android 2.2 及更高版本；通过<uses-feature>标记指定应用程序需要 OpenGL ES 2.0 支持。

```
<uses-sdk
    android:targetSdkVersion="8"
    android:minSdkVersion="8" />
<uses-feature
    android:glEsVersion="0x00020000" />
```

17.8.2 获取 OpenGL ES 2.0 平面

我们从创建自定义的 SurfaceView 开始，它通常在 Activity 的 onCreate()方法中完成，如下所示：

```
private class CustomGL2SurfaceView extends GLSurfaceView {
    final CustomRenderer renderer;

    public CustomGL2SurfaceView(Context context) {
        super(context);
        setEGLContextClientVersion(2);
        renderer = new CustomRenderer();
        setRenderer(renderer);
    }
}
```

上面代码中最重要的一行是对 setEGLContextClientVersion()方法的调用，它的作用在于获取一个 OpenGL ES 1.x（当参数是 1 时）或者 OpenGL ES 2.0（当参数是 2 时）的 EGL 上下文。这样，自定义的渲染器就设置好了。

Renderer 的方法传入的是 GL10 对象，这看起来有些令人不解，而之后你又如何实现 OpenGL ES 2.0 的调用呢？答案很简单：新的 GLES20 类是纯静态的，你只需要忽略 GL10 参数，而直接使用 GLES20 类即可。

CustomRenderer 从初始化顶点（vertices）开始，这与我们之前遇到的类似。之后，当 onSurfaceCreate()方法被调用时，我们就可以初始化着色程序了，如下所示：

```
@Override
public void onSurfaceCreated(GL10 unused, EGLConfig unused2) {
    try {
        initShaderProgram(R.raw.simple_vertex, R.raw.simple_fragment);
        initialized = true;
    } catch (Exception e) {
        Log.e(DEBUG_TAG, "Failed to init GL");
    }
}
```

simple_vertex和simple_fragment这两个资源标识符简单地关联了两个作为原始

Download at http://www.pin5i.com/

资源的文本文件。现在，我们来看看如何初始化着色器：

```
private int shaderProgram = 0;
private void initShaderProgram(int vertexId, int fragmentId)
    throws Exception {
    int vertexShader =
        loadAndCompileShader(GLES20.GL_VERTEX_SHADER, vertexId);
    int fragmentShader =
        loadAndCompileShader(GLES20.GL_FRAGMENT_SHADER, fragmentId);
    shaderProgram = GLES20.glCreateProgram();
    if (shaderProgram == 0) {
        throw new Exception("Failed to create shader program");
    }
    // 向程序中关联着色器
    GLES20.glAttachShader(shaderProgram, vertexShader);
    GLES20.glAttachShader(shaderProgram, fragmentShader);
    // 向点着色器绑定属性
    GLES20.glBindAttribLocation(shaderProgram, 0, "vPosition");
    // 连接着色器
    GLES20.glLinkProgram(shaderProgram);
    // 检查连接器状态
    int[] linkerStatus = new int[1];
    GLES20.glGetProgramiv(shaderProgram, GLES20.GL_LINK_STATUS,
        linkerStatus, 0);
    if (GLES20.GL_TRUE != linkerStatus[0]) {
        Log.e(DEBUG_TAG, "Linker Failure: "
            + GLES20.glGetProgramInfoLog(shaderProgram));
        GLES20.glDeleteProgram(shaderProgram);
        throw new Exception("Program linker failed");
    }
    GLES20.glClearColor(0.5f, 0.5f, 0.5f, 1);
}
```

在上面的代码中，我们首先为每一个着色器载入了文本并且进行了编译；然后我们创建了一个新的着色器程序引用（shader program reference），将两个着色器与其关联，并将属性位置（attribute position）与我们唯一的输入参数相绑定，而后连接程序；最后，执行检查以确定程序成功连接。

每一个着色器的载入均由 `loadAndCompileShader()` 方法完成。下面是该方法的一个实现：

```
private int loadAndCompileShader(int shaderType, int shaderId)
    throws Exception {
    InputStream inputStream =
        AndroidGL2Activity.this.getResources().openRawResource(shaderId);
    String shaderCode = inputStreamToString(inputStream);
    int shader = GLES20.glCreateShader(shaderType);
    if (shader == 0) {
        throw new Exception("Can't create shader");
    }
    // 向 GL 传入代码
    GLES20.glShaderSource(shader, shaderCode);
    // 执行编译
    GLES20.glCompileShader(shader);
    // 获取编译状态
    int[] status = new int[1];
    GLES20.glGetShaderiv(shader, GLES20.GL_COMPILE_STATUS, status, 0);
```

Download at http://www.pin5i.com/

```
        if (status[0] == 0) {
            // 失败
            Log.e(DEBUG_TAG, "Compiler Failure: "
                + GLES20.glGetShaderInfoLog(shader));
            GLES20.glDeleteShader(shader);
            throw new Exception("Shader compilation failed");
        }
        return shader;
    }
```

loadAndCompileShader()方法以字符串的方式读取原始资源，然后源代码通过glShaderSource()方法被递交给 GLES20，最后，通过调用 glCompileShader()方法完成着色器的编译。为了确保编译无误，需要对结果进行检查。OpenGL ES 2.0 在内存中存有编译完成的二进制结果，这样就可以在随后的连接中使用了。

onSurfaceChanged()方法看起来应该比较熟悉，它只改变了一点点：对视点（viewport）进行了重新配置，并且定义了清除色（clear color）。再次注意你可以简单地使用静态 GLES20 调用而非 GL10 参数。

```
@Override
public void onSurfaceChanged(GL10 unused, int width, int height) {
    Log.v(DEBUG_TAG, "onSurfaceChanged");
    GLES20.glViewport(0, 0, width, height);
    GLES20.glClearColor(0.5f, 0.5f, 0.5f, 1);
}
```

最后，我们准备渲染这个三角。在每次系统调用我们实现的 onDrawFrame()方法时，场景都会被渲染。

```
@Override
public void onDrawFrame(GL10 unused) {
    if (!initialized) {
        return;
    }
    GLES20.glClear(GLES20.GL_COLOR_BUFFER_BIT);
    GLES20.glUseProgram(shaderProgram);
    GLES20.glVertexAttribPointer(0, 3, GLES20.GL_FLOAT, false, 12,
        verticesBuffer);
    GLES20.glEnableVertexAttribArray(0);
    GLES20.glDrawArrays(GLES20.GL_TRIANGLES, 0, 3);
}
```

至此，代码看起来都比较熟悉。主要的区别在于对 glUseProgram()方法的调用，在其中我们必须传入需要编译和连接的程序的数值标识符。程序运行的最终结果是在屏幕上显示一个简单的静止三角形。考虑到我们花费的代码量，这着实不太能够振奋人心。灵活的着色器是很强大的，但许多应用程序并不需要这些随 OpenGL ES 2.0 而来的扩展功能。

到目前为止，你也许在疑惑着色器到底是个什么样子。由于 Android 的资源系统是使用文件名除去扩展名的部分，所以在这里我们决定给着色器文件仔细地命名，以便能够强调它们：simple_vertext.shader 和 simple_fragment.shader，它们已经是可以定义的最简单的着色器了。

首先，我们来看看点着色器（vertex shader），它是第一个被传入管道的：

```
attribute vec4 vPosition;
void main()
{
```

Download at http://www.pin5i.com/

```
        gl_Position = vPosition;
    }
```

它只有一个输入：vPosition，被简单地赋予了输出。其中没有应用任何变形，也没有附加任何材质。现在，我们把目光转移到局部面着色器：

```
precision mediump float;
void main()
{
    gl_FragColor = vec4(0.0, 1.0, 0.0, 1.0);
}
```

这个着色器更加简单，它向输出指定了一个固定的颜色值，具体地讲是绿色。

着色器的定义可以很复杂，可以应用光照、材质、烟雾效果和其他不能通过 OpenGL ES 1.x 实现的有趣的 OpenGL ES 2.0 特性，这些具体的效果超出了本书的范围。不过，我们推荐你能够阅读一些有关 OpenGL ES 2.0 的书籍，比如 Aaftab Munshi、Dan Gisnburg 和 Dave Shreiner 所著的 *OpenGL ES 2.0 Programming Guide*（ISBN：0321502795）或者 Richard S. Wright、Jr. Nicholas Haemel、Graham Sellers 和 Benjamin Lipchak 所著的 *OpenGL SuperBible*（ISBN：0321712617），或者其他的在线资料。

小　结

在本章中，你学习了在 Android 应用程序中使用 OpenGL ES 的基本知识，你还学习了 Android 平台所支持的不同版本的 OpenGL ES。

你还掌握了在 Android 应用程序中使用 OpenGL ES 的几种不同的方法，以及如何在独立的线程中初始化 OpenGL ES。之后，学习了如何通过一系列的 OpenGL 和 Android 辅助方法来进行物体的绘制、着色以及光源的设置。然后，你学习了应用程序线程和 OpenGL 线程之间进行相互通信的方法。最后，学习了如何清理 OpenGL。

创建具有完整功能的 3D 应用程序是一个庞大的话题，远远超出本书所能容纳的范围。不过，你已经学习了在 Android 中进行三维绘制的足够知识，并且可以使用这些知识将 OpenGL 的一般概念应用在 Android 中。

参考资料和更多信息

Khronos OpenGL ES 概览：http://www.khronos.org/opengles/.

OpenGL ES 1.1 API 文档：www.khronos.org/opengles/sdk/1.1/docs/man/.

OpenGL ES 2.0 API 文档：www.khronos.org/opengles/sdk/2.0/docs/man/.

OpenGL ES 相关信息：www.opengl.org/.

Download at http://www.pin5i.com/

第 18 章

使用 Android NDK

虽然 Android 应用程序主要由 Java 编写，但有时开发人员也可能需要使用原生的 C 或 C++库。Android 原生开发包（Android Native Development Kit，NDK）提供了一系列可以在 Android 应用程序中包含并使用原生库的工具。在本章中，你将学习在什么情况下需要使用 Android NDK，以及如何配置和使用它。

18.1 何时使用 Android NDK

大多数 Android 应用程序都使用 Android SDK 以 Java 语言进行开发，并且在 Dalvik 虚拟机下流畅而高效地运行。但是，通过 Java 调用某些原生代码（native code）的情况也是存在的。Android NDK 提供支持结合 Android Java 应用程序编译和使用原生 C 和 C++库的工具链。使用 Android NDK 通常基于以下两个理由。

① 执行高处理器占用的操作，如复杂的物理计算，这些操作使用 C 和 C++实现会更为高效，从而带来性能上的本质提升。

② 使用不便于移植的现有代码，通常为共享或专有的 C 或 C++库。这也是使用单一代码支持多个平台时的常见情况。

Warning

Android NDK 生成的原生库只能用于运行 Android 1.5 及以上版本的设备，而不能针对较早版本的平台使用 Android NDK 进行开发。

决定是否在 Java 内调用原生代码需要一定的权衡。应用程序开发人员需要仔细地对应用程序进行设计，权衡使用 NDK 所带来的收益和代价，包括：

① 增加代码的复杂性；

② 增加调试的复杂性；

③ 每次调用原生代码的性能开销；

④ 更加复杂的构建（build）过程；

⑤ 开发人员需要同时掌握 Java 和 C/C++。

虽然开发人员不能完全使用 C 或 C++开发整个应用程序，但是当条件允许时他们仍然可以在 Java 中使用已编译的代码。如果你的应用程序需要进行复杂的数学、物理、图形计算或者其他高强度的操作，那么 Android NDK 或许是一个正确的选择。以下 C 和 C++ API 可以在 Android 工程中使用：

Download at http://www.pin5i.com/

① C 库的头文件（libc）；
② 数学库的头文件（libm）；
③ zlib 压缩库的头文件（libz）；
④ 3D 图形库的头文件（OpenGL ES 1.1 和 2.0）；
⑤ 用于在运行时检测设备 CPU 特性的库；
⑥ 其他涉及 C++、日志记录、JNI 等的头文件。

18.2 安装 Android NDK

你可以在 Windows、Mac OSX 或 Linux 操作系统上安装 Android NDK，当然之前需要正确地安装 Android SDK 及其相关工具。另外，你还需要安装以下组件：

① GNU Make 3.81 或更高版本（http://www.gnu.org/software/make/）；
② GNU Awk（Gawk）或 Nawk（http://www.gnu.org/software/gawk/）；
③ Cygwin 1.7 或更高版本（只限于 Windows，http://cygwin.com）。

你可以通过下面的 Android 开发人员网站下载 Android NDK：http://developer.android.com/sdk/ndk/。

18.3 探索 Android NDK

Android NDK 包含许多不同的工具和文件，下面是其中最主要的：

① 向上兼容 Android 平台（1.5 及之后）的原生系统库及头文件；
② 针对 ARMv5TE 和 ARMv7-A 设备（即将支持 x86）编译和链接原生库的工具；
③ 将原生库嵌入 Android 应用程序的构建文件；
④ 使用 ndk-gdb 的原生代码调试工具；
⑤ 位于/docs 子目录下的 NDK 文档；
⑥ 位于/samples 子目录下的 NDK 示例应用程序。

让自己尽快熟悉 Android NDK 的最佳方法是构建某个示例应用程序，比如 hello-jni。以下是实现这一目标的步骤。

① 定位到 NDK 目录的`/samples/hello-jni` 子文件夹，在命令行（Windows 系统的 Cygwin 内）键入：`ndk-build`，构建 hello-jni 原生库。

② 在 Eclipse 内，执行 New→Android Project，并选择 Create from Existing Source，从 NDK 安装目录下的`/samples/hello-jni/project/`子目录中导入工程，并执行清理（clean）。

③ 为工程创建 Debug Configuration（调试配置）。

④ 必要时创建适当的 AVD，然后正常地运行应用程序。

⑤ 如果遇到错误，你可以在运行了 `ndk-build clean` 和 `ndk-build` 之后再次在 Eclipse 中“清理工程（Clean project）”。Eclipse 与原生库的构建状态不同步是比较常见的现象。

18.4 创建自己的 NDK 工程

现在，我们来看看如何使用 Eclipse 在自己的 Android 应用程序中搭建并使用 NDK。完

Download at http://www.pin5i.com/

成下面的步骤可以创建一个新的 Android 应用程序并调用原生代码。

① 在 Eclipse 中创建一个新的 Android 工程。

② 定位至你的工程目录，创建名为/jni 的子文件夹。

③ 在/jni 子目录内，创建一个名为 Android.mk 的文件。下面是 Android.mk 文件的示例代码，它将用于这个我们正在创建的示例应用程序：

```
LOCAL_PATH := $(call my-dir)
include $(CLEAR_VARS)
LOCAL_LDLIBS := -llog -lGLESv2
LOCAL_MODULE := simplendk
LOCAL_SRC_FILES := native_basics.c native_opengl2.c
include $(BUILD_SHARED_LIBRARY)
```

④ 编辑 Android.mk 文件并且根据构建的具体情况对其进行修改。默认情况下，只针对 ARMv5TE 指令集。对于我们的应用程序而言，这已经足够了。当然，你也有可能会只针对一定数量的手机设备，比如，支持 ARMv7 代码可以获得原生的浮点运算特性，以提升数学运算能力。

⑤ 导航回工程目录，执行 ndk-build 脚本，以构建原生库并将其嵌入你的 Android 应用程序。或许你需要为这个批处理文件（batch file）建立一定的路径；ndk-build 脚本位于 ndk 的安装目录下。

⑥ 在 Eclipse 中，更新你应用程序的 Manifest 文件，不过需要确定的是<uses-tag>标记中的 android:minSdkVersion 属性设置为了大于等于 3 的值。在我们的例子中，最后加入了 OpenGL ES 2.0，所以需要设置为 8。

⑦ 创建 Eclipse Debug Configuration 和所有需要的 AVD。

Tip

本章中的许多代码示例都来自于 SimpleNDK 应用程序。在本书的网站上可以下载该应用程序的源代码。

18.4.1 使用 Java 调用原生代码

实现 Java 对原生代码的调用需要下面三个步骤。

① 在 Java 类文件中加入新函数的声明，并将其定义为原生类型。

② 添加一个库的静态初始化器（initializer），原生函数将编译进这个库中。

③ 向原生源文件中添加函数，函数的命名需要遵循一定的命名规则。

其实这个过程并不像上面讲的这么复杂，现在我们就来按部就班地实现它。SimpleNDK 工程拥有一个名为 NativeBasicsActivity.java 的类，我们从这里开始添加原生函数的声明。向类中添加以下声明：

```
private native void basicNativeCall();
```

现在，我们需要导入上面声明的这个函数的原生库。这个操作并不需要在每次调用时进行，而只是针对每一个库。在 Android.mk 文件中，我们将这个库标识为了 simplendk，所以我们根据这个标识符导入它。向 NativeBasicsActivity 类中添加下面的静态初始化器：

Download at http://www.pin5i.com/

```
NativeBasicsActivity class:
static {
    System.loadLibrary("simplendk");
}
```

最后，这个函数需要被添加到编译的原生库中的某个 C 文件中。每一个函数都必须遵照一定的命名约定：函数名的每一个部分都需要使用下画线分隔开，如下所示：

```
Java_package_name_ClassName_functionName
(JNIEnv *env, jobject this, your vars...);
```

对于这个例子而言，我们的函数应该看起来像下面这样：

```
void Java_com_androidbook_simplendk_NativeBasicsActivity_basicNativeCall
    (JNIEnv *env, jobject this)
{
    // do something interesting here
}
```

虽然这个函数名看起来很冗长，但是不这样的话会出现错误。此后，当在 Java 中声明为 basicNativeCall() 的方法被调用时，实际上将调用这个函数。怎么验证呢？向函数中加入下面的代码，注意需要在 C 文件中包含“android/log.h”：

```
__android_log_print(ANDROID_LOG_VERBOSE, DEBUG_TAG, "Basic call");
```

这样就实现了！首先你从 Android Java 调用 C 原生代码，如果你熟悉 JNI 的话，会意识到这基本上是相同的，而主要的不同点在于可用的库。熟悉 JNI 的开发人员会觉得 NDK 非常简单。

18.4.2 处理参数和返回值

现在你可以实现一个基本的调用了，让我们向 C 中传入一些参数并且获得它的返回值。在这里，我们实现一个简单的 C 函数，它使用一个格式化字符串（能够在 stdio sprintf() 中使用）和两个待相加的数值作为其输入。数值相加后的结果被放入格式化字符串中，并返回新生成的字符串。虽然很简单，但它演示了处理 Java 对象的方法，并且提醒了我们需要在原生 C 代码中很小心地管理内存：

```
jstring Java_com_androidbook_simplendk_NativeBasicsActivity_formattedAddition
    (JNIEnv *env, jobject this, jint number1,
    jint number2, jstring formatString)
{
    // 从 Java 对象中获取一个 C 字符串
    jboolean fCopy;
    const char * szFormat =
        (*env)->GetStringUTFChars(env, formatString, &fCopy);
    char * szResult;
    // 将两个数值相加
    jlong nSum = number1+number2;
    // 确保有足够的空间存储 nSum
    szResult = malloc(sizeof(szFormat)+30);
    // 执行调用
    sprintf(szResult, szFormat, nSum);
    // 获得一个 Java 字符串对象
    jstring result = (*env)->NewStringUTF(env, szResult);
```

Download at http://www.pin5i.com/

```
    // 释放 C 字符串
    free(szResult);
    (*env)->ReleaseStringUTFChars(env, formatString, szFormat);
    // 返回 Java 字符串对象
    return(result);
}
```

上面代码中的 JNI 环境对象用来同 Java 对象进行交互，而一般的 C 函数用来进行一般的 C 内存管理。

18.4.3 在原生代码中使用异常

原生代码可以抛出异常，并且能够被 Java 端捕获，进而检测错误、调用相关的 Java 代码进行处理。这会很频繁地使用到 JNIEnv 对象，熟悉 JNI 的开发人员也会熟悉它。下面的原生代码将在传入 number 不在某个范围内时抛出一个异常：

```
void Java_com_androidbook_simplendk_NativeBasicsActivity_throwsException
    (JNIEnv * env, jobject this, jint number)
{
    if (number < NUM_RANGE_MIN || number > NUM_RANGE_MAX) {
        // 抛出异常
        jclass illegalArgumentException =
            (*env)->FindClass(env, "java/lang/IllegalArgumentException");
        if (illegalArgumentException == NULL) {
            return;
        }
        (*env)->ThrowNew(env, illegalArgumentException,
            "What an exceptional number.");
    } else {
        _ _android_log_print(ANDROID_LOG_VERBOSE, DEBUG_TAG,
            "Nothing exceptional here");
    }
}
```

相应的 Java 声明需要一个 throws 子句：

```
private native void throwsException(int num)
    throws IllegalArgumentException;
```

通常，异常类将通过反射（reflection）被找到，而后，JNIEnv 对象的 ThrowNew()方法被调用来进行实际的异常抛出。

要演示原生 C 代码中的异常，我们还需要看看从 C 中如何调用 Java 方法，如下所示：

```
void Java_com_androidbook_simplendk_NativeBasicsActivity_checksException
    (JNIEnv * env, jobject this, jint number)
{
    jthrowable exception;
    jclass class = (*env)->GetObjectClass(env, this);
    jmethodID fnJavaThrowsException =
        (*env)->GetMethodID(env, class, "javaThrowsException", "(I)V");
    if (fnJavaThrowsException != NULL) {
        (*env)->CallVoidMethod(env, this, fnJavaThrowsException, number);
        exception = (*env)->ExceptionOccurred(env);
        if (exception) {
            (*env)->ExceptionDescribe(env);
```

Download at http://www.pin5i.com/

```
            (*env)->ExceptionClear(env);
            _ _android_log_print(ANDROID_LOG_ERROR,
                DEBUG_TAG, "Exception occurred. Check LogCat.");
        }
    } else {
        _ _android_log_print(ANDROID_LOG_ERROR,
            DEBUG_TAG, "No method found");
    }
}
```

有关代码中 GetMethodID()函数的更多内容可以查阅 JNI 参考或在线资料，它基本上是通过反射的方式获得方法的引用，但是要求第 4 个参数必须正确地传递。这里，传入一个整数并且返回 void。

由于方法返回 void，所以我们使用 CallVoidMethod()函数执行调用，并使用 ExceptionOccurred()函数来检查方法是否抛出了一个异常。如果是，那么 ExceptionDescribe()函数则实际地向 LogCat 写入异常信息，但这看起来与一般的异常输出有些不同。之后，异常得以清除，程序就此为止。

上面代码中被调用的 Java 方法如下所示：

```
@SuppressWarnings("unused") // 从原生代码中调用
private void javaThrowsException(int num)
    throws IllegalArgumentException {
    if (num == 42) {
        throw new IllegalArgumentException("Anything but that number!");
    } else {
        Log.v(DEBUG_TAG, "Good choice in numbers.");
    }
}
```

使用@SuppressWarnings 选项的原因在于该方法并未直接被任何 Java 代码所使用，而只是从原生代码中对其进行了调用。在这个过程中，你还可以调用任何 Android SDK 所提供的 Java 方法。记住，使用 NDK 确实会带来性能的提升，如果你发现你进行了太多的 Android 调用，那么不妨把简单的事情放在 Java 端，而把那些“繁重”的工作交给原生代码去完成。

18.5 提升图形处理性能

使用 Android NDK 的重要理由之一便是使用 OpenGL ES 1.1 和 2.0 的原生库，以承担复杂的数学计算，提升效率，加速移植。虽然可以使用 Android SDK 提供的有关 OpenGL ES 支持的 Java API，但是如果你拥有 C 或 C++编写的核心图形库的话，可能就会选择使用 NDK 了。下面是使用 Android NDK 开发和使用图形库的一些小技巧和建议。

① Android 1.6（API Level 4）及更高版本支持 OpenGL ES 1.1 原生库，而 Android 2.0（API Level 5）及更高版本支持 OpenGL ES 2.0 原生库。在访问其中的函数之前，需要确保包含了“GLES2/gl2.h”，或许还有“GLES2/gl2ext.h”。函数的命名符合标准的 OpenGL 方式（例如，glClearColor）。

② 记得使用<uses-sdk> Manifest 标记来限定应用程序使用的 OpenGL ES 所支持的最低 SDK 版本。

③ 记得使用<uses-feature> Manifest 标记指定应用程序所使用的 OpenGL ES 版本，

Download at http://www.pin5i.com/

这样 Android Market 可以根据这些内容筛选应用程序，并将它们提供给兼容的设备。

例如，下面的代码片段显示了 OpenGL 那一章中涉及的 drawFrame() 方法在 NDK 中的样子。你可以在 SimpleNDK 工程中找到这段代码：

```
const GLfloat gVertices[] = {
    0.0f, 0.5f, 0.0f,
    -0.5f, -0.5f, 0.0f,
    0.5f, -0.5f, 0.0f
};

void Java_com_androidbook_simplendk_NativeOpenGL2Activity_drawFrame
    (JNIEnv * env, jobject this, jint shaderProgram)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glUseProgram(shaderProgram);
    glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 12, gVertices);
    glEnableVertexAttribArray(0);
    glDrawArrays(GL_TRIANGLES, 0, 3);
}
```

它被 CustomRenderer 类的 onDrawFrame() 方法所调用。由于这段代码是在一个相对较密集的循环中执行的，所以以原生代码的方式编写它确实能够起到一定的作用。当然，上面的代码并不能完全体现出这一点，但当我们使用复杂的数学运算、变形计算等的时候，对比就明显了。不过，只有在真实设备上测试时所显示的效果才是具有参考价值的。

小　结

Android NDK 提供了在 Android 1.5 及更高版本设备上调用原生 C 和 C++库的许多有用的工具。安装 Android NDK 工具集是一个相对简单明了的过程。使用 Android NDK 包括在 Android 应用程序包文件中创建脚本，以包含共享原生库。虽然 Android NDK 并不适用于每一个应用程序，但某些应用程序可能会从中大受裨益。

参考资料和更多信息

Android NDK Download Site（下载站点）：http://developer.android.com/sdk/ndk/index.html.

Google Discussion Group（讨论组）：Android NDK：http://groups.google.com/group/android-ndk.

Download at http://www.pin5i.com/

第 19 章

使用 Android 可选硬件 API

Android 软件开发包（Software Development Kit，SDK）提供了各种应用程序编程接口（API）来访问手机上的底层硬件特性。除了摄像头之外，Android 设备可能还拥有一些其他的传感器。Android 设备上可用的设备传感器包括地磁和方向传感器、光和温度传感器，以及支持 Wi-Fi 和蓝牙信号的相关硬件。另外，应用程序还可以访问设备电池的状态信息。在本章中，你将探索 Android SDK 提供的一些可选硬件的 API。

19.1 与设备硬件进行交互

Android 平台空前地提供了对设备底层硬件安全而又稳定的访问。由于不是所有的 Android 设备都支持或拥有所有的可选硬件，所以在对底层设备硬件进行访问时，很有必要遵循下面的原则。

① 不能在代码或其他相关内容中假定某些底层硬件的存在性和可用性。

② 在试图使用程序访问硬件之前，始终对可选硬件功能进行检查和确认。

③ 当使用硬件 API 时，要特别注意异常处理，包括错误处理和返回值的检验。

④ 充分理解硬件特性是一种设备资源，一定要在必需时才占用，并且尽早释放。换句话说，要充分照顾到其他的应用程序。不要滥用硬件资源，也不要消耗过多的电池电量。

Warning

Android 模拟器对硬件传感器的模拟不是很强大，包括 Wi-Fi、蓝牙和设备电池在内。所以在真实设备上进行测试是非常重要的。本章中涉及的许多代码和 API 都只能运行在 Android 硬件上，而在 Android 模拟器上基本或完全不能运行。

不同 Android 设备所具有的可选硬件功能是其重要的市场区分因素。例如，某些设备以支持 Wi-Fi 作为其产品闪光点，而其他的可能支持蓝牙，还有一些则更关心从各个传感器中收集数据。除此之外，应用程序还可以访问电池及电源管理状态的有关数据，另外还包括我们在第 14 章“使用位置服务 API”和第 15 章“使用 Android 多媒体 API”中分别讨论过的位置服务摄像头。

Tip

本章中的许多代码示例都来自于 SimpleHardware 应用程序。在本书的网站上可以下载它的源代码。

Download at http://www.pin5i.com/

19.2 使用设备传感器

Android SDK 提供设备上的传感器对原始数据的访问。传感器及其精度和特性根据设备的不同而各有区别。有些设备还提供地磁传感器和加速度传感器供应用程序使用，前者可以被用作指南针，而后者可以用来检测运动。

你可以通过 SensorManager 对象（android.hardware.SensorManager）来访问设备传感器。SensorManager 对象监听来自传感器的数据。这是一个系统服务，所以可以通过 getSystemService()方法来获取它的实例，如下所示：

```
SensorManager sensors =
    (SensorManager) getSystemService(Context.SENSOR_SERVICE);
```

19.2.1 使用不同的传感器

Sensor 类（android.hardware.Sensor）为设备上可能存在的各种不同传感器定义了一系列标识符。不是每一个设备都拥有所有的传感器。下面列出了其中比较常用的。

- **TYPE_ACCELEROMETER**：在三个方向上测量加速度（国际标准单位[m/s^2]）。
- **TYPE_GYROSCOPE**：在三个方向上测量旋转角（单位为度）。
- **TYPE_ORIENTATION**：测量三个方向上的旋转（单位为度，已废弃）。
- **TYPE_LIGHT**：测量环境光线（国际标准单位 lux）。
- **TYPE_MAGNETIC_FIELD**：罗盘，测量三个方向上的磁力（单位为微特斯拉[μT]）。
- **TYPE_PRESSURE**：测量大气压。
- **TYPE_PROXIMITY**：测量离某个对象的距离（单位为厘米或“near”、“far”）。
- **TYPE_TEMPERATURE**：测量温度。

SensorManager 对象还拥有针对特定传感器的若干常量值。例如，STANDARD_GRAVITY 常量可以用在加速度测量仪中，而 LIGHT_SUNLIGHT 常量则可用于光线传感器。

Tip

不是所有的设备都提供所有的传感器。例如，HTC Evo 4G Android 手机拥有加速度、磁场和距离传感器，但不具有温度、压力或陀螺仪。

不幸的是，模拟器并不能提供任何传感器数据。所有的传感器测试必须依赖于物理设备。不过，OpenIntents.org 网站提供了一些方便实用的传感模拟器（http://code.google.com/p/openintents/wiki/SensorSimulator），它能够模拟加速度仪、罗盘和温度计，并且能够将数据传递给模拟器。

19.2.2 获取对传感器的访问

你可以使用 SensorManager 类的 getDefaultSensor()方法来获得对特定传感器进行访问，它需要传入一个传感器类型参数。例如，你可以通过下面的方式获得对默认加速度传感器的访问：

```
Sensor accelSensor = sensors.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
```

Download at http://www.pin5i.com/

19.2.3 读取传感器数据

在你获得了一个有效的 Sensor 对象后，就可以周期性地读取传感器数据了。传感器的值使用 SensorEventListener 对象传回给应用程序，这一对象必须由应用程序实现并使用 registerListener() 方法予以注册。

```
boolean isAvailable = sensors.registerListener(SensorsActivity.this,
    accelSensor, SensorManager.SENSOR_DELAY_NORMAL);
```

在上面的例子中，加速度传感器会被观测。onSensorChanged() 方法会以一定的时间间隔被调用，这一时间间隔在 registerListener() 中定义，这里我们使用了默认值。

SensorEventListener 接口拥有两个必须实现的方法：onAccuracyChanged() 和 onSensorChanged()。onAccuracyChanged() 方法在给定传感器的精度发生改变时被调用，而 onSensorChanged() 方法则是在传感器数据发生改变时调用，它通常用于监测传感器的信息。

下面是对 onSensorChanged() 方法的一个实现，它将显示各种类型的传感器数据（不只是加速度仪）：

```
@Override
public void onSensorChanged(SensorEvent event) {
    StringBuilder sensorMessage =
        new StringBuilder(event.sensor.getName()).append(" new values: ");
    for (float value : event.values) {
        sensorMessage.append("[").append(value).append("]");
    }
    sensorMessage.append(" with accuracy ").append(event.accuracy);
    sensorMessage.append(" at timestamp ").append(event.timestamp);

    sensorMessage.append(".");

    Log.i(DEBUG_TAG, sensorMessage);
}
```

onSensorChanged() 方法拥有一个参数：一个 SensorEvent 对象。SensorEvent 类包含了有关传感器的所有数据，包括哪个传感器触发的事件、传感器的精度以及传感器的当前读数和时间戳。有关各个类型传感器的具体数据细节，可以参考 Android SDK 中提供的 SensorEvent 类的文档。

加速度传感器提供了 *x*、*y* 和 *z* 轴三个方向上相对于重力加速度的加速度（acceleration minus Gravity）。图 19.1 显示了一个普通的 Android 设备的加速度传感器的数值。

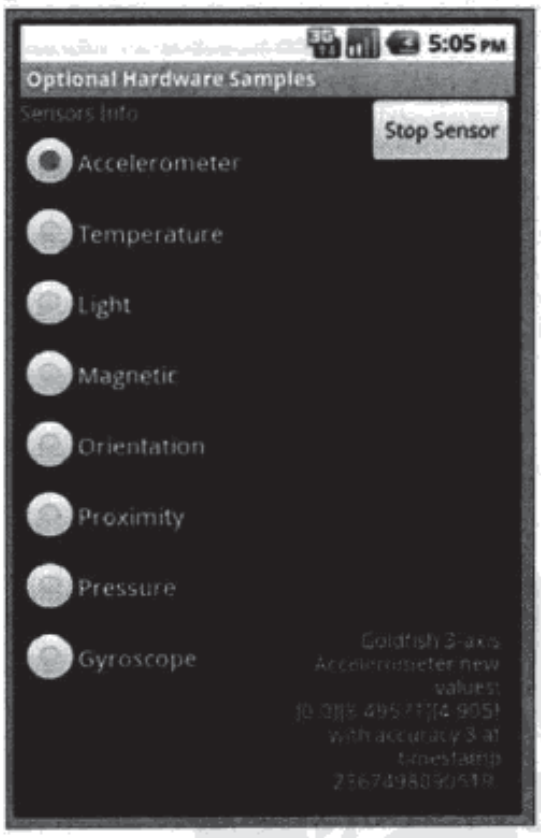


图 19.1 显示加速度值的传感器示例应用程序

Warning

由于使用的传感器不同，传感器数据的产生频率可能会非常高。你需要注意的是在应用程序的 onSensorChanged()方法中不能实现太多的功能。

Download at http://www.pin5i.com/

19.2.4 校正传感器

传感器的数据只有在校准之后才能够为应用程序所使用。一种校准的方式是通过用户按下某个按钮来进行校准。应用程序可以存储下此时的值。之后，新的值可以以此作为参照，计算出其较之原始值的变化量（偏差）。虽然手机的传感器有其特定的方向，但这能够允许用户既能在“竖向”模式下使用也能在“横向”模式下使用，并且既能够握在身前也能够举在头上（仰躺的情况）。

当注册一个传感器时，如果传感器可用并且能够被激活，那么 `registerListener()` 方法将会返回 `true`，否则它将返回 `false`。

传感器通常很敏感，所以其值也容易受到影响。在大多数使用场合下，应用程序可能需要对数据进行一些平滑处理来降低噪音和抖动（shaking）造成的影响。具体如何实现依赖于应用程序的功能。例如，一个模拟的气泡水平仪（bubble level）需要较为精确的数据；相反，在某些游戏中就需要对这些数据进行适当的平滑处理，以降低其敏感性，不然很容易让用户感到沮丧。

方向数据可能只在某些需要测定手机方向的场合下使用，其变动频率远没有加速度仪和指南针那么高。

19.2.5 确定设备的方向

你可以使用 `SensorManager` 类来获得设备的旋转方向。虽然 `Sensor.TYPE_ORIENTATION` 值被废弃不用了，但很多流行的设备仍然使用它。不过，最新推荐的方式是使用 `SensorManager` 类的 `getOrientation()` 方法。

> **Note**
>
> 我们这里说“最新推荐的方式”是因为从 Android 最初开发以来，获取设备旋转方向的最佳方法几经改动。

`getOrientation()` 方法需要传入两个参数：一个旋转矩阵和一个包含 3 个浮点值的数组（方位[azimuth][z]、倾斜[pitch][y]以及旋转[roll][y]）。

19.2.6 指北

除了 `SensorManager` 以外，还有一个有用的类，称为 `GeomagneticField`，它位于 `android.hardware` 包中。`GeomagneticField` 类使用世界磁场模型（World Magnetic Model）来估计地球上任何一个位置的磁场，它通常用于确定地磁北极与地理北极之间的偏差。这一模型由美国国家空间地理情报机构（United States National Geospatial-Intelligence Agency，NGA）开发，并且每隔 5 年进行一次精确的校准。它会在 2015 年过期，当然，即使过了那个时间也不影响大多数情况下的使用，不过 Android GeomagneticField 类也应该会在那时得到更新。

19.3 使用 Wi-Fi

Wi-Fi 传感器可以读取网络状态信息，并且确定附近的无线接入点。Android SDK 提供一系列的 API 来获取设备可用的 Wi-Fi 网络信息，以及网络连接的细节。这些信息可以用来检

Download at http://www.pin5i.com/

测信号强度、查找接入点，或者在连接到特定接入点后执行某些操作。这一节我们将介绍如何获取 Wi-Fi 信息。不过，如果你想要了解有关网络的更详细信息，可以参考第 12 章“使用 Android 网络 API”的有关内容。

下面的例子需要在 AndroidManifest.xml 文件中显式地声明两个许可权限。在应用程序获取 Wi-Fi 网络的信息或开启 Wi-Fi 无线电通信设备时，CHANGE_WIFI_STATE 许可权限是必需的。而当应用程序需要请求 Wi-Fi 设备的信息时，就需要 ACCESS_WIFI_STATE 许可权限。它们可以像下面这样被加入 AndroidManifest.xml 文件：

```
<uses-permission
   android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission
   android:name="android.permission.ACCESS_WIFI_STATE" />
```

应用程序需要做的下一件事情是实例化 WifiManager 对象。它是一个系统服务，所以 getSystemService()方法能够派上用场。

```
WifiManager wifi =
   (WifiManager) getSystemService(Context.WIFI_SERVICE);
```

在获取了有效的 WifiManager 对象之后，应用程序就可以利用它来实现一些有用的功能了。首先，应用程序执行一次 Wi-Fi 扫描，以发现在附近区域可供接入的入口点。要执行扫描，需要执行以下几个步骤。

① 使用 WifiManager 对象的 startScan()方法启动扫描。

② 注册针对 SCAN_PRESULTS_AVAILABLE Intent 的 BroadcastReceiver。

③ 调用 getScanResult()方法获取 ScanResult 对象列表。

④ 遍历结果，利用它们实现一些功能。

其中，前两步可以由下面的代码实现：

```
wifi.startScan();

registerReceiver(rcvWifiScan,
   new IntentFilter(WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));
```

下面代码中的 BroadcastReceiver 对象将完成后面两步操作。它会被周期性地调用直到调用了 WifiManager 对象的 stopScan()方法。

```
rcvWifiScan = new BroadcastReceiver() {

   public void onReceive(Context context, Intent intent) {
      List<ScanResult> resultList = wifi.getScanResults();
      int foundCount = resultList.size();

      Toast.makeText(Wifi.this,
         "Scan done, " + foundCount + " found",
         Toast.LENGTH_SHORT).show();
      ListIterator<ScanResult> results = resultList.listIterator();
      String fullInfo = "Scan Results : \n";
      while (results.hasNext()) {
         ScanResult info = results.next();
         String wifiInfo = "Name: " + info.SSID +
            "; capabilities =" + info.capabilities +
```

Download at http://www.pin5i.com/

```
                "; sig str = " + info.level + "dBm";

            Log.v("Wifi", wifiInfo);

            fullInfo+= wifiInfo + "\n";
        }

        status.setText(fullInfo);
    }
};
```

ScanResult 对象拥有比上面演示的要多的字段（fields）。SSID 属性也就是网络名称，对于用户来讲应该是最熟悉的。capabilities 属性列出了一些诸如所使用的安全模型（如“WEP”）这样的信息。而给出的信号强度属性（level）可能对用户而言显得不太直观。

不过，WifiManager 对象提供两个辅助类来处理信号强度的问题。第一个是 calculateSignalLevel()，它可以有效地将信号强度的数值转换成可供指示条使用的数值。第二个是 compareSignalLevel()，它可以用来比较两个结果的相对信号强度。

Note

模拟器并不提供对 Wi-Fi 的模拟，不过 WifiManager API 仍然可以使用，只不过没有返回扫描结果而已。所以，对 Wi-Fi API 的测试需要在实际的拥有 Wi-Fi 无线电设备的硬件上进行。

WifiManager 对象用来列举扫描出来的接入点。通常的接入点为用户所配置的，或者是之前连接过的。下面的代码演示了如何使用 getConfiguredNetworks()方法：

```
ListIterator<WifiConfiguration> configs =
    wifi.getConfiguredNetworks().listIterator();

String allConfigs = "Configs: \n";
while (configs.hasNext()) {
    WifiConfiguration config = configs.next();
    String configInfo = "Name: " + config.SSID +
        "; priority = " + config.priority;

    Log.v("Wifi", configInfo);

    allConfigs += configInfo + "\n";
}

status.setText(allConfigs);
```

返回的 WifiConfiguration 对象并不包含其所能包含的全部信息。例如，它并没有填充网络密码字段。它所填充的字段与在 ScanResult 对象中的类似。例如，当设备处于已知的 Wi-Fi 网络范围内时，在用户没有设置自动连接的情况下，这可以用来通知用户相关的情况。

WifiManager 对象还可以用来配置 Wi-Fi 网络，获取 Wi-Fi 无线电信号的状态等。更多信息可以参考 android.net.wifi 包。

19.4 使用蓝牙

在 Android 2.0 SDK 中引入了蓝牙 API。当然，不是所有的 Android 设备都拥有蓝牙硬件，

Download at http://www.pin5i.com/

但是，蓝牙是深受消费者喜欢的功能，所以 Android 开发人员可以充分利用它的优势。如果蓝牙设备可用的话，Android 应用程序可以：

① 扫描与发现蓝牙设备并同蓝牙适配器进行交互；

② 建立 RFCOMM 连接并且以数据流的方式与设备进行数据传输；

③ 使用蓝牙设备建立点对点和多点连接，并对它们进行管理。

蓝牙 API 是 android.bluetooth 包的一部分，应用程序必须拥有一定的许可权限才能使用蓝牙服务。android.permission.BLUETOOTH 权限用户连接蓝牙设备，而类似地，android.permission.BLUETOOTH_ADMIN 权限则用于管理蓝牙硬件及其相关服务，包括开启或关闭硬件设备、执行扫描等。

蓝牙 API 包含以下几个有用的类。

① BluetoothAdapter 类代表本地设备上的蓝牙无线电硬件。

② BluetoothDevice 类代表远程蓝牙设备。

③ BluetoothServerSocket 类用来打开监听到达连接的套接字，并且在连接后提供一个 BluetoothSocket 对象。

④ BluetoothSocket 类被客户端用来建立同远程设备的连接。在设备连接以后，BluetoothSocket 对象被双方用于处理连接和获取输入输出流。

19.4.1 检查可用的蓝牙硬件

要在应用程序中使用蓝牙功能需要做的第一件事情就是确定设备是否具有可用的蓝牙无线电硬件。这可以通过调用 BluetoothAdapter 类的静态方法 getDefaultAdapter() 并检查其返回值来实现：

```
BluetoothAdapter btAdapter = BluetoothAdapter.getDefaultAdapter();
if (btAdapter == null) {
    Log.d(DEBUG_TAG, "No bluetooth available.");
    // ...
} else {
    // 蓝牙可用
}
```

19.4.2 开启蓝牙

在确定了设备确实拥有蓝牙无线电后，你需要通过 BluetoothAdapter 类的 isEnabled() 方法来检查其是否已经开启。如果适配器已经开启，那么你可以继续后面的操作；否则，你还需要请求开启它。这可以通过以下几种方式来实现。

① 使用 startActivityForResult() 方法来启动 BluetoothAdapter.ACTION_REQUEST_ENABLE Intent。它会运行一个 Activity 来让用户选择是否开启蓝牙适配器。如果结果是 RESULT_OK，那么证明蓝牙设备已经开启了；否则，证明用户终止了开启蓝牙的过程。

② 调用 BluetoothAdapter 类的 enable() 方法。它仅用于那些需要显式开启蓝牙无线电设备的应用程序，并且需要 BLUETOOTH_ADMIN 权限。另外，它只应该在用户采取了某些直接的操作后执行，比如用户按下了一个按钮，或者菜单项、对话框等。

③ 让 Android 蓝牙设备可见的过程本身就会开启蓝牙。这可以使用 startActivityForResult() 方法来启动 BluetoothAdapter.ACTION_REQUEST_ DISCOVERABLE Intent。

Download at http://www.pin5i.com/

它会运行一个 Activity 来让用户选择是否在一定时间内开启设备发现。

19.4.3 查询配对设备

你可以使用 BluetoothAdapter 来查询可供连接的蓝牙设备。getBondedDevices() 方法会返回一组 BluetoothDevice 对象，它们代表了与蓝牙适配器配对的设备。

```
Set<BluetoothDevice> pairedBtDevices = btAdapt.getBondedDevices();
```

19.4.4 搜索设备

新的蓝牙设备在使用前必须被搜索到并且与适配器进行配对。你可以使用 BluetoothAdapter 来启动和终止搜索可用蓝牙设备的过程。startDiscovery() 方法异步地启动搜索过程，它需要 android.permission.BLUETOOTH_ADMIN 权限的许可。

在你开启了搜索过程后，应用程序需要进行注册以便能够接收下面 Intent 的广播。

- **ACTION_DISCOVERY_STARTED**：当搜索开始时发生。
- **ACTION_FOUND**：每次在发现一个远程蓝牙设备时发生。
- **ACTION_DISCOVERY_FINISHED**：当搜索过程完成时发生。

设备发现的过程是非常消耗资源与时间的。你可以使用 isDiscovery() 方法来检测设备搜索过程是否正在进行，而 cancelDiscovery() 方法可以用来终止这一过程，它应该在即将同远程蓝牙设备建立连接时调用。

19.4.5 建立设备间连接

两个设备间建立连接实际上是一个设备通过各种方法搜寻另一个设备，包括事先已经建立的配对和通过搜索发现的设备。在找到设备以后，设备将调用 connect() 方法。这样，两个设备都各自拥有了一个有效的 BluetoothScocket 对象，它可以用来获取 InputStream 和 OutputStream 对象，以便在两个设备间进行数据通信。

现在，一切都准备就绪了。如果相同的应用程序在两个设备上同时运行，这也是通常的情况，那么意味着两个设备都可以被发现，并且通过 BluetoothServerSocket 对象开启监听，这样它们就可以接收到来的连接请求了。另外，我们需要注意一个事实，那就是对 BluetoothServerSocket 类的 accept() 方法和 BluetoothSocket 类的 connect() 方法的调用会阻塞同步调用，所以你应该能够觉察到这里需要使用一些线程机制。设备搜寻的过程需要消耗一定的硬件资源，所以你需要在适当的时候取消这一过程而后重启它。在连接状态或者尝试连接的状态下进行设备搜索操作可能会极大地降低设备的性能。

Tip

蓝牙这一节中的代码片段均摘自 SimpleBluetooth 应用程序，它的完整代码可以在本书网站上下载。在两个设备间建立并维持连接的代码量非常大，所以，我们在这里只能泛泛而谈，不能对完整的代码进行讲解。你可以仔细地研究这个示例工程，包括让远程设备发出“嘀”声（像声纳一样）的方法。

图 19.2 显示了一个蓝牙应用程序的布局，同时列出了在 SimpleBluetooth 工程中的线程使用情况。

Download at http://www.pin5i.com/

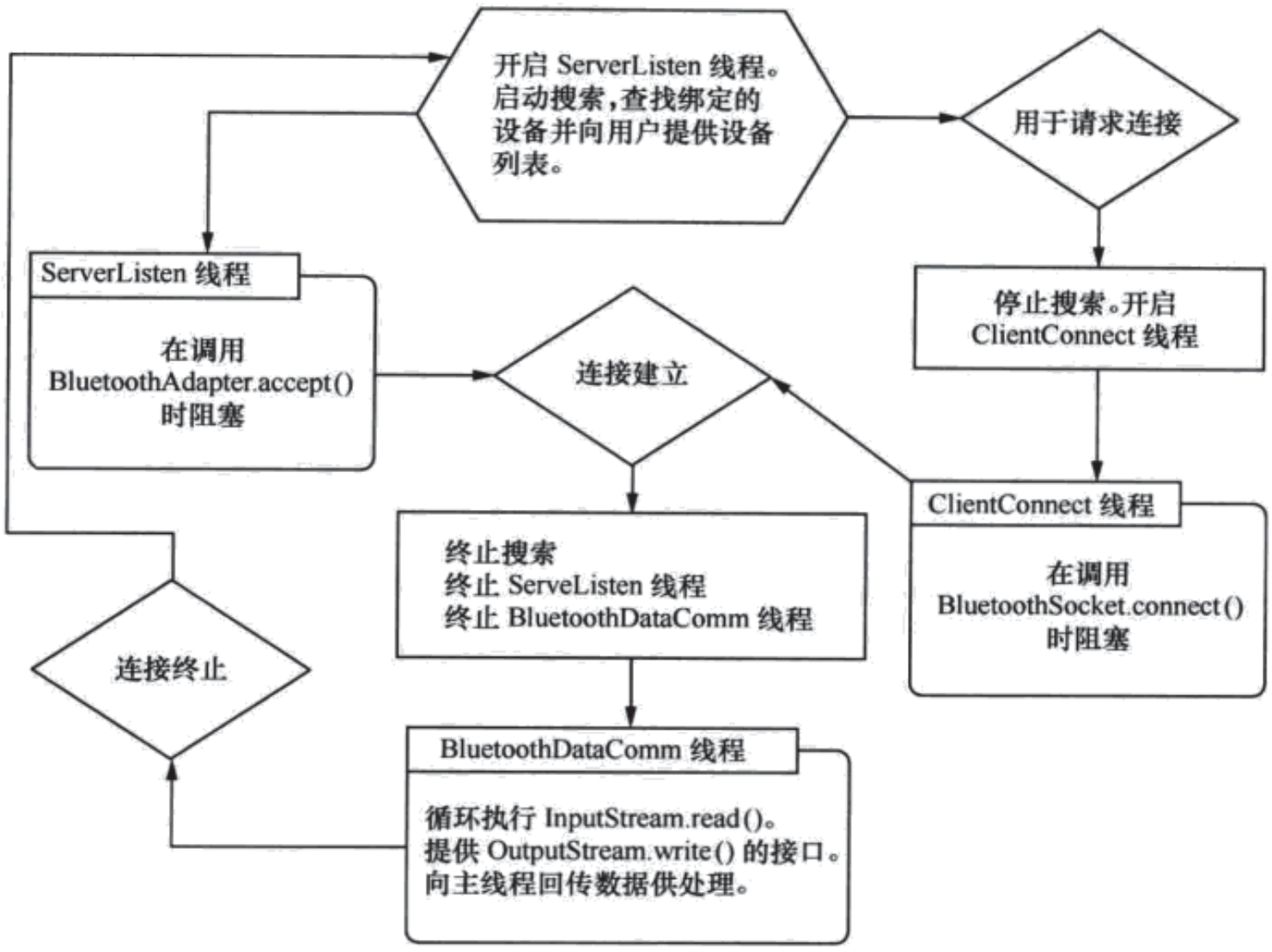


图 19.2 Android 蓝牙应用程序的行为流（behavior flow）

19.5 监视电池

移动设备通过电池维持其工作。虽然许多应用程序并不需要知晓电池的状态，但是某些应用程序可能会需要根据电池的电量、充电状态或电源管理设置来改变自己的行为。例如，一个监视应用程序在电池电量较低时可以降低其监视的扫描频率，而当使用外部电源补充电量后又可以提高其频率。电池的电量级别还可以监测应用程序的执行效率，并且发现某些可以通过改变行为来维持电量的操作。这样做一定能够受到用户的欢迎。

要对电池进行监测，应用程序必须拥有 BATTERY_STATS 许可权限。将下面的 XML 添加到 AndroidManifest.xml 文件中即可满足需要：

```
<uses-permission
   android:name="android.permission.BATTERY_STATS" />
```

之后，应用程序需要注册特定的 BroatcastIntent。在这里为 Intent.ACTION_BATTERY_CHANGED。下面的代码演示了这一操作：

```
registerReceiver(batteryRcv,
   new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
```

下一步，应用程序需要提供对 BroadcastReceiver 的实现。下面给出了一个 BroadcastReceiver 的示例：

```
batteryRcv = new BroadcastReceiver() {

   public void onReceive(Context context, Intent intent) {
      int level =
         intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
```

Download at http://www.pin5i.com/

```
int maxValue =
    intent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);
int batteryStatus =
    intent.getIntExtra(BatteryManager.EXTRA_STATUS, -1);
int batteryHealth =
    intent.getIntExtra(BatteryManager.EXTRA_HEALTH, -1);
int batteryPlugged =
    intent.getIntExtra(BatteryManager.EXTRA_PLUGGED, -1);
String batteryTech =
    intent.getStringExtra(BatteryManager.EXTRA_TECHNOLOGY);
int batteryIcon =
    intent.getIntExtra(BatteryManager.EXTRA_ICON_SMALL, -1);
float batteryVoltage =
    (float) intent.getIntExtra(BatteryManager.EXTRA_VOLTAGE,
       -1) / 1000;
boolean battery =
    intent.getBooleanExtra(BatteryManager.EXTRA_PRESENT,
       false);
float batteryTemp =
    (float) intent.getIntExtra(
       BatteryManager.EXTRA_TEMPERATURE, -1) / 10;
int chargedPct = (level * 100)/maxValue ;

String batteryInfo = "Battery Info:\nHealth=" +
    (String)healthValueMap.get(batteryHealth)+"\n" +
    "Status="+(String)statusValueMap.get(batteryStatus)+"\n" +
    "Charged % = "+chargedPct+"%\n"+
    "Plugged = " + pluggedValueMap.get(batteryPlugged) + "\n" +
    "Type = " + batteryTech + "\n" +
    "Voltage = " + batteryVoltage + " volts\n" +
    "Temperature = " + batteryTemp + "°C\n"+
    "Battery present = " + battery + "\n";

status.setText(batteryInfo);
icon.setImageResource(batteryIcon);

Toast.makeText(Battery.this, "Battery state changed",
    Toast.LENGTH_LONG).show();
  }
};
```

在上面的代码中有几个需要引起注意的地方。第一，注意电池的电量级别并没有被直接使用，而是通过其最大值得到一个百分比。原始的数据对用户而言没有太大意义。第二，状态（status）属性的值及其含义由 android.os.BatteryManager 对象定义，一般而言是电池的充电状态。第三，电池的“健康（health）”程度（仍然在 android.os.BatteryManager 对象中定义）指示了电池的损耗程度，还包括过热情况等。除此之外，Plugged 值指示设备是否连接电源，如果是，还将指示其连接的是交流电源还是 USB 电源。

Warning

在某些设备上，部分信息可能获取不到，或者信息不够准确。例如，即使我们可以获得大多数字段的值，但某些设备对于 present 字段却返回 false。在根据这些字段值做出判断之前可能需要对设备进行一定的检测。

Download at http://www.pin5i.com/

除了上面讨论的之外，还可能返回一些其他的信息，包括图标指示符，它可以直观地显示电池的状态和其他一些技术细节，例如电池类型、当前电压以及温度等。图 19.3 给出了显示所有这些信息的界面。

图 19.3 来自物理设备电池监视器的返回结果

Tip

在模拟器中可以获得测试电池的部分返回结果。对于电源控制的更多信息可以参考附录 A “Android 模拟器快速入门”。

小 结

不同于其他的许多移动平台，Android 允许对设备上的底层硬件进行访问，包括读取原始设备传感器数据、使用内置 Wi-Fi 支持、蓝牙硬件及其服务和监视电池使用情况等。需要记住的是不同的手机设备拥有不同的底层硬件，所以，在 Android 工程的设计阶段要始终考虑每一种目标手机平台上可用的功能。

参考资料和更多信息

Android API Reference：Sensor Data Details.http://developer.android.com/reference/android/hardware/SensorEvent.html#values.

NOAA：World Magnetic Model.http://www.ngdc.noaa.gov/geomag/WMM/DoDWMM.shtml.

Android Dev Guide（开发指南）：Bluetooth.http://developer.android.com/guide/topics/wireless/bluetooth.html.

Android Sample Application（示例应用程序）：Bluetooth Chat.http://developer.android.com/resources/samples/BluetoothChat/index.html.

Download at http://www.pin5i.com/

第五部分

更多 Android 应用程序设计原则

第 20 章 使用通知

第 21 章 使用服务

第 22 章 扩展 Android 应用程序

第 23 章 用户账户管理与数据同步

第 24 章 处理高级用户输入

第 25 章 面向不同设备配置和语言

Download at http://www.pin5i.com/

第20章

使用通知

在很多情况下，应用程序需要与用户进行交互，即使是在后台运行的时候。应用程序可以通过文字、震动、彩光甚至声音来告知用户某些事情或提醒用户进行某些操作。在本章中，你将学习如何在应用程序中构建不同类型的通知。

20.1 向用户发出通知

应用程序使用通知可以很大程度上提升其用户体验。例如如下情况。

① 电子邮件应用程序在有新邮件到达时需要提醒用户，阅读器应用程序在有新文章可供阅读的时候需要提醒用户。

② 游戏程序在有好友加入、发送了一次邀请或者打破了高分记录等情况发生时可能需要通知用户。

③ 天气预报应用程序需要在有特殊天气情况时向用户告警。

④ 股票应用程序也需要在到达预定股价时将这一情况告知用户。

用户对此会非常欢迎，因为它可以帮助用户完成应用程序所设置的工作流程，提醒用户需要运行某些应用程序。当然，这些通知必须与应用程序和所发生的事件相匹配，打扰用户是需要足够的理由的。每一个应用程序都有适合自己的通知类型，并且能够针对用户所关心的事件发出通知。例如，应用程序不能发出一个紧急样式的通知（想象一下带有闪光、铃声、亟待答复的通知）然后仅仅用来告知用户他的图片已经上传成功或者是完成了某个文件的下载。

Android 平台提供向用户发出通知的几种不同的途径。通知通常显示在屏幕顶端的状态栏（status bar）中，包括以下几种类型：

① 文字信息；

② 图像指示器；

③ 声音指示器；

④ 设备震动；

⑤ 指示灯的控制。

Warning

虽然 Android SDK 提供了创建各类通知的方法，但不是所有的设备都完整地支持它们，比如，有些设备就不具有指示灯或震动功能。另外，不同的设备在处理通知时进行设备震动的强度也是不一样的。所以，你始终不要忘记在目标设备上进行实际的测试。

Download at http://www.pin5i.com/

现在，我们来看看如何在应用程序中使用各种不同类型的通知。

20.2 使用状态栏发布通知

Android 设备上用于显示通知和指示器的标准区域位于屏幕顶端的状态栏。通常，状态栏显示了诸如当前日期和时间这类信息。在必要时，它还用于显示通知（比如收到的短消息），收起时显示简略的版本，而当用户向下拖动后则显示完整的通知列表，用户还可以在此时单击 Clear（清空）按钮清除所有的通知。

开发人员可以使用通知来告知用户重要的事件，这能够增强应用程序的功能。例如，当一个下载项完成时可能需要向用户发出一个简单的通知，它有以下几个重要的组成部分：

① 一个图标（显示在状态栏和完整的通知中）；
② 一行 Ticker 文本（ticker text）（显示在状态栏中）；
③ 通知标题文本（显示在完整的通知中）；
④ 通知正文（显示在完整的通知中）；
⑤ 一个 Intent（当用户单击完整通知时运行）。

在这一节中，你将学习如何创建一个基本的通知。

Tip

本章中的许多代码示例都来自于 SimpleNotifications 应用程序。在本书的网站上可以下载它的源代码。

20.2.1 使用 NotificationManager 服务

所有的通知都使用 NotificationManager 进行创建。NotificationManager（在 android.app 包中）是一个必须予以请求的系统服务。下面的代码演示了如何使用 getSystemService()方法获得一个有效的 NotificationManager 对象：

```
NotificationManager notifier = (NotificationManager)
    getSystemService(Context.NOTIFICATION_SERVICE);
```

如果没有一个有效的 Notification 对象供 notify()方法使用的话，NotificationManager 是毫无用处的。Notification 对象定义了当其被触发时需要向用户显示的信息。这包含了在状态栏显示的文字、展开状态下显示的多行文字、两种状态下均使用的图标、Notification 被触发的次数以及上次被触发的时间。

20.2.2 创建一个带有图标的简单文本通知

在状态栏中显示的图标和文字条均在 Notification 的构造函数中进行设置，如下所示：

```
Notification notify = new Notification(
    R.drawable.android_32, "Hello!", System.currentTimeMillis());
```

之后，你还可以通过类的公有成员设置通知的其他信息，如下所示：

```
notify.icon = R.drawable.android_32;
notify.ticketText = "Hello!";
```

Download at http://www.pin5i.com/

```
notify.when = System.currentTimeMillis();
```

在调用 notify()方法之前，还需要对另外几个的信息进行设置。我们需要调用 setLastEventInfo()方法，它可以配置一个 View 来显示展开的状态栏。下面给出了一个示例：

```
Intent toLaunch = new Intent
    (SimpleNotificationsActivity.this,
    SimpleNotificationsActivity.class);
PendingIntent intentBack = PendingIntent.getActivity
    (SimpleNotificationsActivity.this, 0, toLaunch, 0);

notify.setLatestEventInfo(SimpleNotificationsActivity.this,
    "Hi there!", "This is even more text.", intentBack);
```

然后，使用 notify()方法提供通知的标题和正文，以及在用户点击通知时需要触发的 Intent。这里，我们使用自己的 Activity，所以当用户单击通知时，我们的 Activity 将再次运行。

Note

当展开的状态栏被拖曳下来后，当前的 Activity 会被视为前台（显示状态）运行的 Activity。在前台运行时发出系统通知一般来讲是没有必要的，因为运行在前台的应用程序可以使用 Dialog 或 Toast 来通知用户。

20.2.3 使用通知队列

现在应用程序已经做好通知用户的准备了。所有需要做的就是使用一个标识符和我们配置的 Notification 来调用 NotificationManager 的 notify()方法。下面的代码演示了这个过程：

```
private static final int NOTIFY_1 = 0x1001;
// ...
notifier.notify(NOTIFY_1, notify);
```

如果 Notification 的标识符与之前相同类型的 Notification 实例相匹配的话，原先的 Notification 将被更新而非重新创建一个新的。例如，你可能会在下载某个文件时给出一个 Notification，然后在下载完成时更新这个 Notification，而不是在 Notification 队列中加入一个新的 Notification，因为它很快就会变得多余。这样，应用程序只需要唯一的一个 Notification 标识符就可以了。

通知会在状态栏内显示一个图标和一行文字条，在图 20.1 的顶部可以看到它。

短暂地显示了 Ticker 文字条之后，状态栏将会恢复原来的状态，即显示每一个通知的图标。如果用户展开状态栏，那么可以见到类似图 20.2 所示的界面。

图 20.1 显示了图标和 Ticker 文字条的状态栏通知

Download at http://www.pin5i.com/

图 20.2 显示了图标、两段文本和通知时间的扩展状态栏

20.2.4 更新通知

应用程序所发出的通知不应该一直堆在通知栏里，所以，你可能需要复用或者更新已有的通知。例如，假设现在有一个通知，告诉用户某个文件正在下载，那么当文件下载完成后，如果你想发布另一个通知，那么就没有必要让这条通知依然存在了。或者，你也可以简单地修改第一条通知的内容。

当标识符相匹配时，原先的通知将会被更新。当提交了标识符相匹配的通知时，Ticker 文字条并不会再次被绘制。如果要告知用户这一改变，可以使用一个计数器。Notification 对象的 number 成员变量将会记录并显示这一计数。例如，我们可以像下面这样将其设置为 4：

```
notify.number = 4;
```

这将会在图标上显示一个小的数字。它只在状态栏中显示，而不显示在其展开状态下，不过应用程序可以对其显示的信息进行更新。图 20.3 给出了这种情况下状态栏的显示。

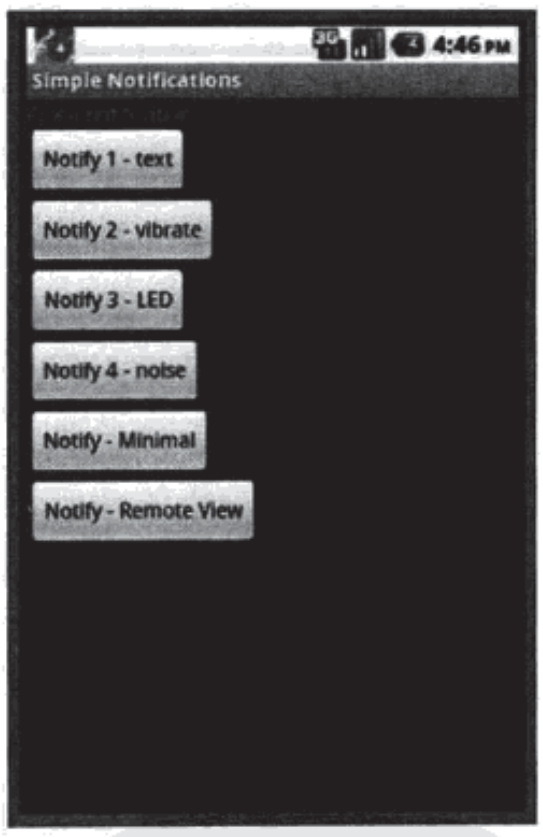


图 20.3 图标上显示了计数“4”的状态栏

20.2.5 清除通知

当用户单击通知时，相关联的 Intent 会被触发。在此之后，应用程序就可以从系统通知队列中清除这一通知了。这可以通过 NotificationManager 对象的 cancel() 方法来实现。例如，使用下面的调用即可取消我们之前创建的通知：

```
notifier.cancel(NOTIFY_1);
```

上面的代码将取消带有相同标识符的通知。如果应用程序在用户单击通知后并不执行

Download at http://www.pin5i.com/

其他的操作，那么上面的实现是一种相对比较简单的方式。可以像下面这样简单地设置一个标签：

```
notify.flags |= Notification.FLAG_AUTO_CANCEL;
```

设置了 Notification.FLAG_AUTO_CANCEL 标签之后，当用户单击通知时它将被自动清除。这对于只需要运行 Intent 的应用程序而言是一种方便而简单的操作。

Notification 对象与你所接触的其他 Android 对象稍微有所不同。大部分与它交互的方式是直接访问其公有变量，而不是通过某些辅助方法。这对于某些后台应用程序或者服务（service）而言非常有用，它们将在第 21 章“使用服务”中讨论。Notification 对象可以一直被保留着，每次只需要修改其参数值即可。在修改后，需要再次调用 notify() 方法提交这一新的通知。

20.3 让手机震动

在吵闹的环境中，或者在不适宜采用视觉或听觉方式的情况下，震动是用来提醒用户的最佳方式。（虽然当手机放置在硬板桌面上时，震动也会制造一些噪音。）Android 的通知机制给予了震动以很好的控制模式。不过，在应用程序使用震动作为通知的手段之前，还需要显式地声明一个许可权限。下面的代码给出了用于实现震动所需的 XML，它应该被包含在 AndroidManifest.xml 文件中：

```
<uses-permission
    android:name="android.permission.VIBRATE" />
```

Warning

震动功能必须在手机设备上进行测试。模拟器并不会以任何方式指示手机的震动，同样地，某些 Android 设备也不支持震动功能。

如果没有声明上述许可权限，那么将无法实现震动，或者会有某种错误发生。一旦获得了这一许可权限，那么应用程序就可以使手机以任何模式震动。震动的模式可以通过 vibrate 成员变量进行设定，它使用一个 long 型数组描述震动的时长。下面的这行代码定义了一个简单的震动模式，它将在通知出现时触发手机的震动：

```
notify.vibrate = new long[] {0, 200, 200, 600, 600};
```

上面定义的通知模式为：开始震动 200ms，然后暂停 200ms，之后再震动 600ms，最后停止 600ms。如果需要重复震动，可以为其设置一个标签，这样手机将不断地震动下去直到用户清除这一通知。

```
notify.flags |= Notification.FLAG_INSISTENT;
```

应用程序可以针对不同类型的事件或者当前的通知计数来使用不同的震动模式。例如，可以实现一个类似钟摆（grandfather clock）的报时方式，这样你可以根据震动的次数推测出当前的时间。

Tip

使用简短、特征性强的震动非常有用，因为这样可以让用户习惯于它们。

Download at http://www.pin5i.com/

20.4 灯光闪烁

灯光闪烁是向用户传递信息的一种比较安静的方式，在其他方式不可用或显得过于鲁莽的时候，灯光闪烁是很好的选择。Android SDK 提供了对彩色指示灯的控制。它一般用于服务指示或者电量报警，当然，在改变了闪烁频率或者灯光颜色后，应用程序也同样可以使用它。

Warning

并不是所有的 Android 设备都具有指示灯，同样地，模拟器也不能显示灯光的状态，所以，你需要在实际设备上测试这一功能。

要使用指示灯，必须为 Notification 对象设置一个标签。之后，还需要设置诸如灯光颜色等控制其闪烁方式的其他信息。下面的这段代码对指示灯进行了配置，它将发出绿光，并以 1 秒亮、1 秒灭的频率进行闪烁：

```
notify.flags |= Notification.FLAG_SHOW_LIGHTS;

notify.ledARGB = Color.GREEN;
notify.ledOnMS = 1000;
notify.lesOffMS = 1000;
```

虽然我们可以任意指定颜色值，但通常指示灯的物理实现是三个小的 LED，颜色分别为红、绿和蓝。虽然颜色能够比较柔和的变化，但仍然不会像屏幕显示那样精确。例如，在 T-Mobile G1 上，白光看起来略带粉色。

Warning

在部分设备上，某些特定的通知具有使用指示灯的优先权。例如，我们发现 T-Mobile G1 手机当连接上 USB 后，指示灯始终显示为绿色，不随其他的操作而改变。只有在拔下接头后灯光的颜色才能发生改变。

应用程序可以使用不同的颜色和闪烁频率来向用户传递不同的信息。例如，某个事件发生的次数越多，指示灯的闪烁就会越急促。下面的这段代码给出了基于通知的触发次数来修改灯光状态的示例：

```
notify.number++;
notify.flags |= Notification.FLAG_SHOW_LIGHTS;

if (notify.number < 2) {
    notify.ledARGB = Color.GREEN;
    notify.ledOnMS = 1000;
    notify.lesOffMS = 1000;
} else if (notify.number < 3) {
    notify.ledARGB = Color.BLUE;
    notify.ledOnMS = 750;
    notify.ledOffMS = 750;
} else if (notify.number < 4) {
    notify.ledARGB = Color.WHITE;
    notify.ledOnMS = 500;
    notify.lesOffMS = 500;
```

Download at http://www.pin5i.com/

```
} else {
    notify.ledARGB = Color.RED;
    notify.ledOnMS = 50;
    notify.lesOffMS = 50;
}
```

指示灯的闪烁将持续进行直到用户清理 Notification。与震动不同，Notification.FLAG_INSISTENT 标签并不对灯光产生影响。

颜色和闪烁频率也可以用来指定其他信息。例如，通过天气服务获知的气温信息可以使用红蓝光加之一定的闪烁频率来指示。针对类似这样的数据使用灯光指示会非常实用，即使其他的方法也能够达到相同的目的，但是灯光会显得更平和。比如，用户只需要一瞥就知道其所传达的信息，而无需中断手里的工作或是另外运行某个应用程序。

20.5 发出声响

有时，手机必须制造出一些声响以引起用户的注意。Android SDK 通过 Notification 对象提供了实现它的一种途径。在需要播放声音时，我们首先需要配置音频流的类型。一般而言，最常用的流格式为 STREAM_NOTIFICATION。你可以使用如下所示的代码来配置通知的音频流类型：

```
notify.audioStreamType = AudioManager.STREAM_NOTIFICATION;
```

现在，我们将 sound 成员变量与某个 Uri 对象相关联即可，这样，在通知被触发时也会随之发出声音。下面的代码给出了如何播放一个在工程资源中包含的声音：

```
notify.sound = Uri.parse(
    "android.resource://com.androidbook.simplenotifications/" +
    R.raw.fallbackring);
```

默认情况下，音频文件只会播放一次。不过，如同震动一样，可以使用 Notification.FLAG_INSISTENT 标签来使音频不断地播放，直到用户终止通知提示。这种通知的方式不需要任何许可权限。

Note

这个例子中使用的声音文件以原始资源的方式包含在了工程中。不过，你可以使用任何设备上的声音文件，需要注意的是不同的 Android 设备上的声音文件可能会不同。

20.6 自定义通知

在扩展状态栏中默认的通知显示在大多数情况下能够满足需求。不过，开发人员仍然可以使用 RemoteViews 对象来自定义通知的外观和风格。

下面的代码创建了一个 RemoteViews 对象并且为其关联了一段文本：

```
RemoteViews remote =
    new RemoteViews(getPackageName(), R.layout.remote);

remote.setTextViewText(R.id.text1, "Big text here!");
remote.setTextViewText(R.id.text2, "Red text down here!");
```

Download at http://www.pin5i.com/

```
notify.contentView = remote;
```

为了更好地理解上面的代码，我们给出其引用的 remote.xml 文件，如下所示：

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:id="@+id/text1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="31dp"
        android:textColor="#000" />
    <TextView
        android:id="@+id/text2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textSize="18dp"
        android:textColor="#f00" />
</LinearLayout>
```

上面示例所定义的通知与默认定义的非常相似，只是它不包含图标。setLatestInfo()方法通常用于为默认的布局关联文本。只不过在这个例子中，我们使用了自定义的布局。另外，仍然需要关联 Intent，如下所示：

```
    Intent toLaunch = new Intent
        (SimpleNotificationsActivity.this,
SimpleNotificationsActivity.class);
    PendingIntent          intentBack          =
PendingIntent.getActivity
        (SimpleNotificationsActivity.this,  0,
toLaunch, 0);

    notify.contentIntent = intentBack;
    notifier.notify(NOTIFY_5, notify);
```

最终运行的结果类似于图 20.4 所示。

图 20.4 只显示两行文本的自定义通知

使用自定义的通知布局能够提供对在扩展状态栏上显示的通知以更好的控制。另外，通过提供主题化或品牌化的风格，它还能够让你的应用程序通知与其他应用程序的相区别。

Note

对于给定的硬件设备，扩展状态栏上可用的布局区域大小是固定的。不过，某些特定的细节可能会因设备的不同而不同。在设计自定义通知的布局时需要时刻考虑这一点。另外，一定要在所有目标设备上测试这一布局，并且还要考虑各种不同的屏幕模式和状态，以保证它始终能够被正确绘制。

默认的布局包含两个文本域、一个图标和一个通知触发时间的显示。用户会很习惯这些信息。应用程序在使用自定义通知时至少都应该提供这些信息。

Download at http://www.pin5i.com/

20.7 设计有用的通知

Android 平台的通知功能非常强健，以至于你可能会过度地使用它，从而造成用户的反感。下面是设计良好的通知所需的一些技巧和建议。

① 只有在你的应用程序不处于前台时才使用通知，否则使用 Toast 或者 Dialog 控件。

② 允许用户选择他想要的通知类型（文字、灯光、声音、震动）和频率，以及在什么情况下触发通知。

③ 在可能的情况下，尽量更新和重用已有的通知而不是重新创建一个新的。

④ 经常性地清理通知以免向用户提供过时的信息。

⑤ 在不确定的情况下，使用柔和的通知语言。

⑥ 确保在通知的 Ticker 文字、标题和正文中包含有用的信息，并且运行有意义的 Intent。

通知框架虽然轻量但却很强大。不过，诸如闹钟或股票监控器这样的应用程序可能会需要提供超越通知框架的功能。在这种情况下，它们可能会使用后台服务并且在特定事件到达时使用自己完整的 Activity。在 Android 2.0 及之后的版本中，开发人员可以使用 WindowManager.LayoutParams 类来开启 Activity 窗口并显示，甚至在屏幕处于键盘锁定的状态下。

小 结

应用程序可以使用通知来与用户进行交互，而超越其自身 Activity 的界限。通知既可以是视觉上的，也可以是听觉上的，或者使用设备的震动功能。可以使用各种方法来自定义通知以便向用户传递更丰富的信息。不过需要特别注意的是通知要适当而且数量要适宜，否则用户将对应用程序感到厌烦，甚至干脆放弃不用。

参考资料和更多信息

Android Dev Guide（开发指南）：Notifying the User.http://developer.android.com/guide/topics/ui/notifiers/index.html.

Android Dev Guide（开发指南）：Creating Status Bar Notifications.http://developer.android.com/guide/topics/ui/notifiers/notifications.html.

Android Reference（参考）：The NotificationManager Class.http://developer.android.com/reference/android/app/NotificationManager.html.

Download at http://www.pin5i.com/

第21章

使用服务

服务是一种可以很大程度上增强应用程序功能的 Android 应用程序组件。Android 服务可以用在不需要接受用户输入而只是在后台运行的功能上，或者也可以用来向其他的应用程序提供信息。在本章中，你将学习如何使用 Android 接口定义语言（Android Interface Definition Language，AIDL）定义一个远程接口。最后，你还将学习如何实现一个 Parcelable 对象，以及如何使用它通过定义的远程接口传递对象。

21.1 何时使用服务

Android 软件开发包（Software Development Kit，SDK）中的服务（service）意味着两层含义。第一，它是一个后台进程，以有规律的时间间隔执行着某些操作。第二，它为远程（remote）对象提供接口，你可以在你的应用程序中调用它们。无论在哪种情况下，服务对象都继承于 Android SDK 中的 `Service` 类，它既可以独立存在，也可以作为某个有完整界面的应用程序的一部分。

当然，不是所有的应用程序都需要使用服务。不过，当你的应用程序需要使用如下一些功能时，你可以考虑提供一个服务。

① 应用程序需要执行耗时而占用资源的任务，同时又不需要用户输入。

② 应用程序需要周期性地执行某项操作，比如上传或下载最新的内容或者记录当前地理位置等。

③ 应用程序需要执行耗时的操作，而且如果应用程序退出，下次启动时再次执行这一过程会对资源造成很大的浪费，比如下载一个很大的文件。

④ 应用程序需要向其他 Android 应用程序提供数据或者信息服务（如网络服务），而且不需要用户界面。

Note

Android 云到设备通信（Android Cloud to Device Messaging，C2DM）是 Android 2.2 中引入的一个服务。它向开发人员提供了远程启动应用程序事件的功能：在需要时可以高效地远程唤醒应用程序，而不需要每一个应用程序都实现自己的后台服务（从而提高了电池续航能力）。C2DM 服务在设备上运行，并且被相关应用程序所共享。开发人员可以从远程服务器向具有 C2DM 服务的设备发送信息，设备会将这一信息发送至相关的应用程序。对于运行 Android 2.2 及更高版本的设备而言，你可以考虑在其上实现 C2DM 解决方案，前提是它比实现本地服务更好。关于云到设备通信的更多信息，可以参考 Google Project 网站：http://code.google.com/android/c2dm/。

Download at http://www.pin5i.com/

21.2 理解服务的生命周期

在我们进入有关服务创建的细节之前，先来看看服务与 Android 操作系统之间的交互情况。首先，服务必须在应用程序的 Manifest 文件中使用<service>标记予以注册。另外，服务可能还需要定义或者要求一些许可权限，用于服务的启动、终止、绑定和具体的调用。

服务实现以后，可以使用 Context.startService()方法启动它。即使在调用 startService()方法时服务已经处于运行状态，也不会另外启动任何新的服务实例。服务持续运行，直到 Context.stopService()方法被调用，或者服务完成了其自身的任务，使用 stopSelf()方法终止了运行。

应用程序如果需要连接到某个服务，那么可以使用 Context.bindService()方法来获得一个连接。如果服务没有运行，那么它将会被创建。建立连接以后，具有一定许可权限的应用程序可以开始向服务发送请求。例如，预言球（Magic Eight Ball）程序可能会拥有一个底层的服务，它可以接收是与否的问题，并且提供一些玄妙（Yoda-style）的答案，而其他应用程序则可以连接到预言球服务上，问一个问题（“我的应用程序会在 Android Market 上走红吗？”）并且获得回答（“各种迹象表明是的。”），最后使用 Context.unbindService()断开与服务的连接。

Warning

与应用程序一样，Android 操作系统可以在内存不足的情况下终止服务。除此之外，服务也拥有类似应用程序拥有的主线程，并且它可能会导致系统停止响应。所以，即使是实现了服务，也需要把那些繁重而耗时的操作放到辅助线程（worker thread）上。

21.3 创建服务

创建 Android 服务牵涉对 Service 类的继承，还需要在 AndroidManifest.xml 许可权限文件中加入有关服务的代码段。在继承 Service 类时需要覆写的方法包括 onCreate()、onStart()、onStartCommand()、onDestroy()等。定义服务的名称可以使其他应用程序能够启动或停止运行在后台的服务。onStart()和 onStartCommand()方法本质上是一样的，只不过 onStart()方法在 API Level 5 及更高版本中被废弃不用了。（API Level 5 及更高版本中，onStartCommand()方法的默认实现是调用 onStart()方法，并且返回某个值，以便与旧版本相兼容。）在下面的例子中，我们实现了两种方法。

Tip

本章中的许多代码示例都来自于 SimpleService 和 UseService 应用程序。在本书的网站上可以下载这两个应用程序的源代码。

这个例子实现了一个监听 GPS 变化的简单服务，当服务启动后，它将周期性地显示通知，并且通过远程接口提供对最近位置数据的访问。下面的代码给出了一个简单的 Service 类定义，名为 GPXService：

```
public class GPXService extends Service {
```

Download at http://www.pin5i.com/

```
    public static final String GPX_SERVICE =
        "com.androidbook.GPXService.SERVICE";

    private LocationManager location = null;
    private NotificationManager notifier = null;

    @Override
    public void onCreate() {
        super.onCreate();
    }
    @Override
    public void onStart(Intent intent, int startId) {
        super.onStart(intent, startId);
    }

    @Override
    public void onStartCommand(Intent intent, int flags, int startId) {
        super.onStart(intent, startId);
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
    }
}
```

你需要了解一个服务的生存周期，它与 Activity 是有区别的。如果系统通过 Context.StartService()方法启动了一个服务，那么它的 onCreate()方法将首先被调用，紧接着是 onStart()或 onStartCommand()方法。不过，如果服务会通过 Context.bindService()调用被绑定的话，那么在 onCreate()调用之后将紧接着调用 onBind()方法，而 onStart()或者 onStartCommand()方法在这种情况下不会被访问。稍后我们将讨论有关绑定的话题。最后，当服务完成（即服务停止）且没有其他的进程与其绑定，onDestroy()方法将被调用。在这个方法中，所有的服务都必须予以清理。

在牢记了上面的内容之后，我们给出前面介绍的 GPXService 类的 onCreate()方法的完整实现：

```
public void onCreate() {
    super.onCreate();
    location = (LocationManager)
        getSystemService(Context.LOCATION_SERVICE);
    notifier = (NotificationManager)
        getSystemService(Context.NOTIFICATION_SERVICE);
}
```

由于对象此时并不知道下一步将被调用的是 onStart()方法（或者 onStartCommand()方法）还是 onBind()方法，所以，我们进行了快速初始化调用，但并没有启动任何后台进程。如果这些对象甚至都没有被绑定器（binder）提供的接口所使用的话，它们可能都是多余的。

由于我们无法预测我们的代码将运行在何种版本的 Android 上，所以我们可以同时实现 onStart()和 onStartCommand()方法，并且让它们都调用另外一个方法，在其中提供具体的实现。这样，我们的应用程序可以同时向上和向下兼容。当然，这还需要工程编译在 Level

Download at http://www.pin5i.com/

5及以上的SDK下，同时将minSdkValue设定为一个支持的早期版本。当然，我们还是强烈建议能够在多个平台下测试应用程序，以便验证它是否能够如你预期地正常运行。下面，让我们关注onStart()和onStartCommand()方法的实现细节：

```
@Override
public int onStartCommand(Intent intent, int flags, int startId ) {
    Log.v(DEBUG_TAG, "onStartCommand() called, must be on L5 or later");

    if (flags != 0) {
        Log.w(DEBUG_TAG, "Redelivered or retrying service start: "+flags);
    }

    doServiceStart(intent, startId);
    return Service.START_REDELIVER_INTENT;
}

@Override
public void onStart(Intent intent, int startId) {
    super.onStart(intent, startId);
    Log.v(DEBUG_TAG, "onStart() called, must be on L3 or L4");
    doServiceStart(intent,startId);
}
```

接着，我们给出doServiceStart()方法的实现：

```
@Override
public void doServiceStart(Intent intent, int startId) {
    super.onStart(intent, startId);
    updateRate = intent.getIntExtra(EXTRA_UPDATE_RATE, -1);
    if (updateRate == -1) {
        updateRate = 60000;
    }

    Criteria criteria = new Criteria();
    criteria.setAccuracy(Criteria.NO_REQUIREMENT);
    criteria.setPowerRequirement(Criteria.POWER_LOW);

    location = (LocationManager)
        getSystemService(Context.LOCATION_SERVICE);

String best = location.getBestProvider(criteria, true);

location.requestLocationUpdates(best,
    updateRate, 0, trackListener);

Notification notify = new
    Notification(android.R.drawable.stat_notify_more,
    "GPS Tracking", System.currentTimeMillis());
    notify.flags |= Notification.FLAG_AUTO_CANCEL;

Intent toLaunch = new Intent(getApplicationContext(),
    ServiceControl.class);
PendingIntent intentBack =
    PendingIntent.getActivity(getApplicationContext(),
    0, toLaunch, 0);

notify.setLatestEventInfo(getApplicationContext(),
```

Download at http://www.pin5i.com/

```
        "GPS Tracking", "Tracking start at " +
        updateRate+"ms intervals with [" + best +
        "] as the provider.", intentBack);
    notifier.notify(GPS_NOTIFY, notify);
    }
```

后台进程在两个方法中启动。在这个例子中，后台进程实际上是在另一个服务上注册以获取数据的更新。对于更多有关使用位置服务（Location-Based Services）和 LocationManager 的内容，可以参考第 14 章“使用位置服务 API”；对于有关 Notification 调用的更多信息，可以参考第 20 章“使用通知”。

Tip

在使用后台进程获取更新时，推荐使用回调来接收数据的更新。大多数的移动设备都只拥有有限的电池电量。持续地在后台运行甚至不断轮询，可以从本质上影响电池的电量。另外，基于相同的考虑，对用户实现服务的回调也可以带来更好的效率。

在这个例子中，我们将在进程运行期间开启 GPS，它会影响电池的续航时间，即使我们只通过耗能最小的请求来确定位置。在开发服务时要时刻记住这一点。

Intent 对象将提取由请求服务的进程传入的数据。这里，我们只提取一个值，即 EXTRA_UPDATE_RATE，以确定数据更新的频率。变量 update-rate 必须在外部进行发布，可以是在开发人员文档中，也可以是在公有的类文件中，这样服务的使用方才知道相关的情况。

LocationListener 对象和 trackListener 的实现细节这里我们可以不必深究，因为它们与服务关系不大。无论如何，进程都必须保持最小化以避免影响用户在前台进行的操作。在用户注意到性能问题之前，我们需要对此进行测试，以确定特定设备所能承受的后台进程。

有两种方法可以用来与用户通信。第一种是使用 Notification。这是为用户提供应用程序信息最柔和的方式。同时，它还意味着通知触发时用户不一定需要正在使用手机，因为通知可以被加入通知队列。例如，天气预报应用程序可能会使用通知来提供每小时天气情况。

另一种方式是使用 Toast 消息。对于某些服务而言，这是一种很好的方式，特别是在用户需要信息被周期性地更新的时候，并且这些更新信息能够合适地占据一小块屏幕区域，而不影响用户当前的操作。例如，对于一个后台音乐播放器，它可以简洁地覆盖屏幕一小块区域，当音乐切换时可以显示当前播放音乐的标题。

当没有客户（client）绑定在服务上时，onDestroy()方法将被调用，并且通过 Context.stopService()方法请求服务的停止，或者在服务内部使用 stopSelf()方法终止服务。在这个时候，所有的东西都必须“优雅”地被清除掉，因为服务即将不存在。

下面给出了 ondestroy()方法的示例：

```
@Override
public void onDestroy() {
    if (location != null) {
        location.removeUpdates(trackListener);
        location = null;
    }

    Notification notify = new
```

Download at http://www.pin5i.com/

```
        Notification(android.R.drawable.stat_notify_more,
        "GPS Tracking", System.currentTimeMillis());
    notify.flags |= Notification.FLAG_AUTO_CANCEL;

    Intent toLaunch = new Intent(getApplicationContext(),
        ServiceControl.class);
    PendingIntent intentBack =
        PendingIntent.getActivity(getApplicationContext(),
        0, toLaunch, 0);
    notify.setLatestEventInfo(getApplicationContext(),
        "GPS Tracking", "Tracking stopped", intentBack);

    notifier.notify(GPS_NOTIFY, notify);
    super.onDestroy();
}
```

在上面的代码中，我们停止了对 LocationListener 对象的更新。这将停止我们所有的后台进程。之后，我们通知用户服务即将终止。最后，只需要简单地一次调用父类的 onDestroy()方法，而不管之前 onStart()被调用了多少次。

只有在 AndroidManifest.xml 许可权限文件中使用服务标记定义了服务，系统才能够接受它。下面是我们需要添加到 Android Manifest 文件中的代码：

```
<service
    android:enabled="true"
    android:name="GPXService">
    <intent-filter>
        <action android:name=
            "com.androidbook.GPXService.SERVICE" />
    </intent-filter>
</service>
```

这段 XML 定义了服务的名称：GPXService，并且设置服务为可用。之后，定义了一个 Intent 过滤器，我们使用了与类中相同定义的字符串。它将在稍后讲解服务控制时使用。通过使用上面这段包含在 Manifest 中应用程序区段的 XML 代码，系统就能够知道服务的存在而且能够被其他应用程序所使用了。

21.4 控制服务

到目前为止，示例代码已经给出了一个 Service 的完整实现。现在我们编写代码来控制前面定义的服务。

```
Intent service = new Intent("com.androidbook.GPXService.SERVICE");
service.putExtra("update-rate", 5000);
startService(service);
```

启动一个服务非常直接，只需要使用服务名称创建一个 Intent 并且调用 startService()方法就可以了。在这个例子中，我们还设置了 Intent 的扩展参数 update-rate 为 5 秒。这个频率已经非常高了，但对于测试而言还是比较合适的。在实际应用中，我们可以将其设置为 60 秒或更长。这段代码将触发对 onStart()或 onStartCommand()方法的调用，即使服务已经处于运行的状态下。

在我们使用完服务后，需要使用下面的代码来终止它：

Download at http://www.pin5i.com/

```
Intent service = new Intent("com.androidbook.GPXService.SERVICE");
stopService(service);
```

这段代码本质上同启动服务的那段代码是一样的，只不过最后调用的是 stopService() 方法。如果没有其他的绑定，那么它将调用 onDestroy() 方法。这意味着后台进程即使在前台调用了 stopService() 的情况下也有可能继续运行。另外，除了这些系统调用之外，如果需要对后台进程有更深入的控制的话，就需要实现远程接口了。

21.5 实现远程接口

除了用于启动和停止 Activity 的系统服务之外，有时还需要提供对其更多的控制。在客户应用程序可以绑定到服务以执行其他调用之前，需要定义接口。Android SDK 对于实现这种目的的远程接口提供了一个有用的工具和文件格式。

要定义远程接口，你必须在一个 AIDL 文件中声明接口，然后实现这个接口，并且在 onBind() 方法被调用时返回这个接口的实例。

以我们在本章中已经建立好的这个 GPXService 服务为例，现在我们为其创建一个远程接口。这个远程接口需要有一个方法，它可以返回上一次记录下的位置。在远程服务调用中，只能使用实现了 Parcelable 协议（protocol）的基本类型和对象。这是由于跨进程调用时内存并不能被共享。当遵循这一规则时，AIDL 编译器将会处理跨界调用的细节问题。Location 对象实现了 Parcelable 接口，所以它是可以使用的。

下面的 AIDL 文件定义了这个接口，名为 IRemoteInterface：

```
package com.androidbook.services;

interface IRemoteInterface {
   Location getLastLocation();
}
```

当使用 Eclipse 时，AIDL 文件 IRemoteInterface.aidl 可以被添加到某个工程中，Android SDK 插件将处理剩下的事情。现在我们必须实现接口的代码。下面的示例提供了接口的一种实现：

```
private final IRemoteInterface.Stub
   mRemoteInterfaceBinder = new IRemoteInterface.Stub() {
      public Location getLastLocation() {
         Log.v("interface", "getLastLocation() called");
         return lastLocation;
      }
   };
```

由于服务的实现代码已经将上一次记录的位置信息作为其成员变量存储了下来，所以这里我们只需要简单地返回它即可。实现的接口需要在服务的 onBind() 方法中返回：

```
@Override
public IBinder onBind(Intent intent) {
   // 由于我们只有一个，所以没有必要对 Intent 进行检测了
   return mRemoteInterfaceBinder;
}
```

如果实现了多个接口，可以在 onBind() 方法中检测传入的 Intent 来确定需要执行的

Download at http://www.pin5i.com/

操作以及所返回的接口。不过在这个例子中，我们只有一个接口并且在 Intent 中不包含其他的信息，所以只简单地返回接口即可。

我们还将绑定接口的类名添加到了 Action 列表中，它包含在 AndroidManifest.xml 文件内服务的 Intent 过滤器中。这样做并不是必需的，但这是一个很好的约定，可以允许使用类名称。在服务标记的定义中可以加入下面的这段代码：

```
<action android:name =
   "com.androidbook.services.IRemoteInterface" />
```

现在，我们可以通过其接口来使用服务了。这需要通过 ServiceConnection 对象并调用 bindService()方法来实现。完成之后，还必须调用 unbindService()方法以告诉系统应用程序完成了对该服务的访问，否则，即使接口不存在了，连接依然会保持。

下面是对 ServiceConnection 对象的两个方法的实现，它们分别是 onServiceConnection()和 onServiceDisconnected()：

```
public void onServiceConnected(ComponentName name,
   IBinder service) {

   mRemoteInterface =
      IRemoteInterface.Stub.asInterface(service);
   Log.v("ServiceControl", "Interface bound.");
}

public void onServiceDisconnected(ComponentName name) {
   mRemoteInterface = null;
   Log.v("ServiceControl",
      "Remote interface no longer bound");
}
```

调用了 onServiceConnected()方法之后，就获得了一个 IRemoteInterface 实例，它可以用来调用我们之前定义的接口。对于远程接口的调用看起来与其他任何接口调用是一样的：

```
Location loc = mRemoteInterface.getLastLocation();
```

Tip

记住远程接口调用将进行跨进程的操作，并且同步完成。如果调用需要消耗比较长的时间，那么应该像其他的耗时调用一样在另外的线程中进行处理。

要在另外的应用程序中使用这个接口时，需要在工程中添加 AIDL 文件以及相应的包。对 onBind()方法的调用将在服务的 onCreate()方法之后触发对 onServiceConnected()方法的调用，记住，在这种情况下，onStart()或 onStartCommand()方法并没有被调用。

```
bindService(new Intent(IRemoteInterface.class.getName()),
   this, Context.BIND_AUTO_CREATE);
```

在这种情况下，执行调用的 Activity 也应该实现 ServiceConnection 接口。这段代码还反映出了为什么在 Intent 过滤器中使用类名的约定非常有用。因为我们拥有两个 Intent 过滤器，而且不会在 onBind()调用中检测其行为，当然我们也可以使用另外的

Download at http://www.pin5i.com/

Intent 过滤器，但现在的代码显得更简洁一些。

当使用完接口后，需要调用 unbindService()方法来断开与接口的连接。不过，对于 onServiceDisconnected()方法的回调并不意味着服务就断开了绑定；绑定依然是存在的，只是断开了连接。

21.6 实现可包装（Parcelable）类

在这个例子中，非常幸运的是 Location 类实现了 Parcelable 接口。但如果需要通过远程接口传递一个新的对象，又怎么办呢？

我们以下面的 GPXPoint 类为例讲解这个问题，如下所示：

```
public final class GPXPoint {
    public int latitude;
    public int logtitude;
    public Date timestamp;
    public double elevation;

    public GPXPoint () {
    }
}
```

GPXPoint 类定义了与 GeoPoint 相似的位置点，但还包含了位置的海拔高度和被记录的时间。这些数据在 GPX 文件所定义的格式中非常常见。但对于我们来讲，它并不是系统能够识别的并且可以通过远程接口传递的基本格式。不过，如果类实现了 Parcelable 接口并且为其创建了 AIDL 文件，那么这个对象就可以在远程接口中使用了。

要完整地支持 Parcelable 类型，我们需要实现一些方法以及 Parcelable.Creator<GPXPoint>。下面给出了上面相同的类在实现为 Parcelable 类之后的代码：

```
public final class GPXPoint implements Parcelable {

    public int latitude;
    public int longitude;
    public Date timestamp;
public double elevation;

public static final Parcelable.Creator<GPXPoint>
    CREATOR = new Parcelable.Creator<GPXPoint>() {

    public GPXPoint createFromParcel(Parcel src) {
        return new GPXPoint(src);
    }

    public GPXPoint[] newArray(int size) {
        return new GPXPoint[size];
    }

};

public GPXPoint() {
```

Download at http://www.pin5i.com/

```
    }

    private GPXPoint(Parcel src) {
        readFromParcel(src);
    }

    public void writeToParcel(Parcel dest, int flags) {
        dest.writeInt(latitude);
        dest.writeInt(longtitude);
        dest.writeDouble(elevation);
        dest.writeLong(timestamp.getTime());
    }

    public void readFromParcel(Parcel src) {
        latitude = src.readInt();
        longitude = src.readInt();
        elevation = src.readDouble();
        timestamp = new Date(src.readLong());
    }

    public int describeContents() {
        return 0;
    }
}
```

writeToParcel()方法是必需的，在Parcel中，它可以按照一定的顺序将对象“扁平化（flatten）”为支持的基本类型。当在一个Parcel中创建这个类时，Creator将被调用，它会依次调用私有的构造函数。为了数据的可读性，我们还创建了readFromParcel()方法，它将上面提到的“扁平化”过程反过来进行，即按照其写入的顺序读取这些基本类型，并创建新的Date对象。

现在需要为这个类创建AIDL文件了。它应该与Java文件放在一起，并命名为GPXPoint.aidl以满足匹配性。内容应该类似如下代码：

```
package com.androidbook.services;

parcelable GPXPoint;
```

现在，GPXPoint类就可以用在远程接口中了。对于其他的原始（native）类型或者Parcelable对象都可以采取这样的方式。此时，IRemoteInterface.aidl文件应该大致被修改为如下形式：

```
package com.androidbook.services;

import com.androidbook.services.GPXPoint;

interface IRemoteInterface {
    Location getLastLocation();
    GPXPoint getGPXPoint();
}
```

另外，我们可以在接口中提供getGPXPoint()方法的实现，如下所示：

```
public GPXPoint getGPXPoint() {
    if (lastLocation == null) {
        return null;
```

Download at http://www.pin5i.com/

```
    } else {
        Log.v("interface", "getGPXPoint() called");
        GPXPoint point = new GPXPoint();

        point.elevation = lastLocation.getAltitude();
        point.latitude =
            (int) (lastLocation.getLatitude() * 1E6);
        point.longitude =
            (int) (lastLocation.getLongitude() * 1E6);
        point.timestamp =
            new Date(lastLocation.getTime());

        return point;
    }
}
```

如同你所看到的，并不需要任何特别的操作。只需要让对象变成 Parcelable 就可以了，现在这个对象可以正常使用了。

小　结

Android SDK 提供了一种 Service（服务）机制，它可以用来后台操作，并且能够在多个应用程序之间实现功能共享。通过使用 AIDL 创建一个接口，Service 可以向其他的应用程序提供其功能，而不需要发布任何库或者包。开发人员可以通过实现 Parcelable 接口来创建可以跨进程传输的对象。

当创建后台服务时需要特别小心。设计粗糙的后台服务可能会对手机的性能和电池续航能力造成本质上的影响。除了进行标准的测试之外，在实现 Service 时还需要综合考虑上面这些因素。

谨慎而有前瞻性地创建 Service 可以显著地增强应用程序或所提供服务的吸引力。创建 Service 是 Android SDK 为开发应用程序所提供的有力工具，而这在其他的移动平台上是不能实现的。

参考资料和更多信息

Android Reference（参考）：The Service Class.http://developer.android.com/reference/android/app/Service.html.

Android Dev Guide（开发指南）：Service Lifecycle.http://developer.android.com/guide/topics/fundamentals.html#servlife.

Android Dev Guide（开发指南）：Processes and Threads.http://developer.android.com/guide/topics/fundamentals.html#procthread.

Android Application Framework FAQ（Android 应用程序框架常见问题与解答）.http://developer.android.com/guide/appendix/faq/framework.html.

Download at http://www.pin5i.com/

第22章

扩展 Android 应用程序

Android 应用程序有着极强的扩展性，它能够突破传统功能的局限，与操作系统紧密结合。开发人员可以使用一系列的平台特性来提升应用程序的功能。在本章中，你将学习到有关增强应用程序的各种途径，使它变得更为强大、更受用户欢迎。

22.1 增强你的应用程序

Android 平台最令人兴奋的特性之一是其与应用程序的互动性。不同于过去的移动开发平台，只是简单地让每一个应用程序运行在自己的“小气泡”中，Android 允许应用程序之间以及应用程序与操作系统之间安全共享数据和功能。在你开发了应用程序的核心部分之后，就可以开始思考如何对应用程序进行扩展，超越传统的使用模式，比如用户启动应用程序、用户运行应用程序、用户关闭应用程序。

虽然我们需要支持这些场景，但这并不是用户与你的应用程序及其特性进行交互的唯一方式。Android 框架包含了许多打破常规的途径，你可以通过下面的这些方法来扩展和增强应用程序。

① 以应用程序小插件的形式分享一部分应用程序的功能，它可以驻留在用户的主页。

② 以 Live Wallpaper（壁纸）的形式为用户提供一个与你的应用程序相关联的互动壁纸。

③ 使用动态文件夹让用户能够快速而简单地组织应用程序数据。

④ 让应用程序的内容在整个设备上能够被搜索到。

⑤ 让你的应用程序充当内容类型处理器（Content Type Handler），使其有能力处理一些常用的数据类型，如图片或视频。

⑥ 充当内容提供器（Content Provider），这样可以向其他应用程序提供内部数据，如果其他应用程序也如此的话，你也可以共享它们的数据，我们在第 11 章“使用内容提供器在应用程序间共享数据”中讨论过有关话题。

⑦ 使用 Intent 过滤器开启应用程序的多个入口点，而非仅仅通过运行默认的 Activity 来启动应用程序。

⑧ 充当广播接收器（Broadcast Receiver）以便对重要的事件做出响应，同时还可以向其他应用程序发布事件广播。

⑨ 实现服务，向其他应用程序提供数据服务和某些具体的功能，还可以使用其他的服务（系统服务或其他应用程序提供的服务）来增强你自己的应用程序，我们在第 21 章“使用服务”中详细地讨论了这一话题。

Download at http://www.pin5i.com/

上面就是我们对于创造性地扩展应用程序功能给出的一些点子。通过这些方法，你的应用程序能够给用户留下一个很好的印象，并且很难将它遗忘。下面，我们来看看上面一些功能的具体实现细节。

22.2 使用应用程序小插件

应用程序小插件（App Widget）在 API Level 3 中被正式引入，它将应用程序与 Android 操作系统之间的集成提升到了一个新的水平，而这在之前开发人员是不可能实现它的。发布小插件的应用程序称为小插件提供器（provider），包含小插件的部件被称作小插件宿主（host）。小插件是一个轻量而功能简单的应用程序（如同桌面插件），它们可以安装在一定的宿主中，比如主页。

小插件通常与某些底层的应用程序相绑定。例如，一个日历应用程序可能会具有一个小插件，用于显示当前的日期，并且让用户能够查看当天的计划事务；在某个具体的事件上单击可能就会启动完整的日历应用程序，从而提供完整的服务。类似地，一个音乐软件可能也会提供一个具有一些简单控件的小插件，让用户在主页就能够方便地开始和停止音乐的播放。本书的示例代码中包含一个简单小插件的实现，它显示美国国土安全部所提供的咨询系统中的风险等级（红/严重、橙/高风险、黄/警戒、蓝/注意、绿/低风险），这种类型的小插件可能适用于与旅游有关的应用程序。

小插件可以周期性地使用最新的内容进行更新。这样，小插件可以用来实现应用程序的一些次级功能，因为如果使用通知的话，它将运行整个应用程序，而且更适用于一些需要用户做出响应的事件。

22.2.1 创建应用程序小插件

首先你需要仔细地考虑是否有必要为应用程序添加小插件功能。虽然小插件轻巧而袖珍，功能也比较简单，但它能够在主页提供对应用程序功能的直接访问。应用程序小插件还能够起到维系用户的作用，因为它位于主页，时刻都在提醒用户自己的存在。某些应用程序只需要运行一个小插件实例（如音乐播放器），但另外一些则可能需要使用多个相同的小插件实例同时运行，只不过显示不同的内容（如图片浏览工具）。

你需要像下面这样对应用程序做出修改，以便能够支持小插件。

① 提供 XML 的小插件配置。

② 确定小插件是否需要一个配置 Activity。

③ 提供对 `AppWidgetProvider` 类的实现。

④ 提供一个 `Service` 类的实现，以便能够在必要时处理小插件的内容更新。

⑤ 修改应用程序的 Android Manifest 文件，注册小插件提供器的相关信息，以及有关更新服务的信息。

现在，我们来具体看看上面这些过程。

Tip

这一部分中的许多代码示例都来自于 SimpleAppWidget 应用程序。在本书的网站上可以下载该应用程序的源代码。

Download at http://www.pin5i.com/

1. 创建应用程序小插件配置

首先，应用程序需要提供一个 XML 的小插件定义。你可以把它存储在工程资源目录/res/xml 下。下面是一个名为/res/xml/simple_widget_info.xml 的小插件定义示例：

```
<?xml version="1.0" encoding="utf-8"?>
<appwidget-provider
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:minWidth="146dp"
    android:minHeight="72dp"
    android:updatePeriodMillis="28800000"
    android:initialLayout="@layout/widget">
</appwidget-provider>
```

上面这个简单的应用程序小插件定义包含了<appwidget-provider> XML 标记。minWidth 和 minHeight 属性定义了小插件的尺寸（需要一个度量单位，这里是 dp），updatePeriodMillis 属性则用来定义小插件内容刷新的周期（使用小插件更新服务），这里定义的是每 8 个小时。最后，initialLayout 属性定义了小插件的布局引用，我们稍后即将介绍这个布局文件。

Note

应用程序小插件的更新周期最短为 30 分钟（1800000 毫秒）。确保你的小插件的 updatePeriodMillis 属性不会低于这个值。你可以考虑实现一个工具或者类似刷新按钮这样的东西来强制进行更新。如果出于某种必要的考虑，你的 Widget 必须打破这条限制，那么你需要自己实现计时器（timer），不过还需要把对电池电量的消耗和对性能的影响综合考虑进来。

要在主页绘制漂亮的应用程序小插件，它的尺寸必须遵循一定的规范。主页被划分成了一定尺寸的若干小格。当用户试图安装一个小插件时，系统会进行检查以确保有足够的空间能够容纳它（通过小插件定义的最小高宽）。

Tip

确定应用程序小插件尺寸的最好办法就是用 74 乘以占用的格数，然后再减去 2。这里，我们想要小插件占 2 格的高和宽，所以，我们把尺寸设定为 146，即（(74 × 2) −2）。如果你需要有关小插件外观设计的详细信息，可以参考下面这个网址：http://developer.android. com/guide/practices/ui_guidelines/widget_design.html。

在<appwidget-provider>标记中，还有许多其他可用的属性。例如，你可以使用 configure 属性指定用于配置小插件的 Activity 类，这在需要支持多个小插件同时运行时非常有用（可以参考本书博客中有关小插件的内容）。要获得小插件提供器的完整配置细节，可以参考 android.appwidget.AppWidgetProviderInfo 类的文档。

2. 确定应用程序小插件是否需要配置 Activity（Configuration Activity）

通常来讲，如果有不止一个小插件的实例，那么每一个的外观或行为都应该不同。当然，这并不是严格的要求；如果每一个小插件实例的外观和功能都相同，那么用户会很快地发现，并且决定只安装其中一个。

Download at http://www.pin5i.com/

如果你需要几个不同的小插件实例，那么就需要为每一个实例都提供一定的设置，这样，你可能就得创建一个配置 Activity 了。配置 Activity 就是一个普通的 Activity，只不过它会在运行时读取特定的 Intent 信息，它必须在小插件的 XML 配置文件中予以定义。

每次创建一个新的小插件实例，都将运行配置 Activity。它的启动需要一个唯一的小插件标识符，并通过启动 Intent 的 EXTRA_APPWIDGET_ID 传入。Activity 在执行完成时，需要向结果 Intent 传回一个表示结果状态的值，比如 RESULT_OK。

配置 Activity 不仅仅是提供用户对特定小插件实例进行配置的功能，还需要更新 RemoteViews 对象，因为小插件在第一次使用 XML 配置中定义的配置 Activity 创建时，并不会收到更新事件。不过，后面的更新将会被接收。

3. 创建一个小插件提供器

应用程序小插件本质上是一个处理特定动作（action）的 BroadcastReceiver。既然如此，与小插件的基本交互是从 onReceive()方法开始的。不过，默认的 AppWidget Provider 类将处理 onReceive()，并且，反过来将操作委派给其他的方法，这才是你需要实现的。

（1）实现 AppWidgetProvider 类

AppWidgetProvider 类通过为开发人员提供一个实现小插件的框架来简化这些操作。AppWidgetProvider 需要实现以下方法。

① onEnabled()方法在小插件被创建时调用。在这里，我们可以进行一些用于整个 Widget 提供器的配置操作。这个方法只在第一个小插件实例被添加到 Widget 宿主（通常是主页）时调用一次。

② onDisabled()方法在小插件被删除时调用，且仅在所有的小插件实例均被从 Widget 宿主移除时调用。例如，如果在主页上有某个提供器的 5 个小插件，那么只有在第 5 个并且是最后一个小插件被删除时它才会被调用。

③ onUpdate()方法周期性地被调用，调用周期取决于定义在小插件配置文件中的值。时间频率并不是很精确，所以不要依赖它做一些精确性较高的操作。如果你需要在精确的时间进行更新，那么可以考虑使用 AlarmManager 类实现计划更新（scheduling updates）。在调用这个方法时，将传入一个 Widget 标识符的列表（list）。其中的每一个标识符代表一个小插件宿主中的一个唯一的小插件实例。小插件提供器需要对每一个实例加以区分，通常指定不同的配置。

④ onDeleted()方法在某一个小插件的实例被删除时调用。

Warning

在 onDelete()方法中存在一个漏洞，它需要通过覆写 onReceived()方法来创建一个有效的补丁，这在文档中有详细的说明。不过，这个问题只存在于 Android 1.5 平台上。如果你需要对这一平台进行开发，那么你可以参考下面的网址：http://developer.android.com/guide/topics/appwidgets/index.html#AppWidgetProvider。

（2）使用远程视图

Android 应用程序小插件并不在应用程序的进程中运行，而是运行在宿主的进程中。所以，小插件使用 RemoteViews 类，以便定义它的用户界面。RemoteViews 类是全部 View

Download at http://www.pin5i.com/

层级结构的一个子集，用于在其他进程中进行显示。通常而言，你需要配置 RemoteViews 对象，然后在 onUpdate()方法中将其传入小插件管理器（App Widget Manager）。不过，在创建了某个实例或者存在配置 Activity 时也需要进行更新。

使用 RemoteViews 定义的视图层级结构仅仅包含有限的控件，包括 Button、ImageButton、ImageView、TextView、AnalogClock、Chronometer 和 ProgressBar，且只具有 FrameLayout、LinearLayout 和 RelativeLayout 布局。除此之外，这些控件的派生对象也不能使用。RemoteViews 的配置应该越简单越好，因为对其视图层级的访问是通过一些辅助类来实现的，比如 setImageViewResource()、setTextViewText()、setProgressbar()、setShort()、setString()和 setChronometer()。简而言之，你可以为小插件生成一个 XML 布局定义，但是需要特别注意只使用那些 RemoteView 类支持的控件。

下面，让我们来看看一个非常简单的布局定义，它位于上面我们提到过的“风险等级”小插件应用程序的资源文件/res/layout/widget.xml 中：

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:id="@+id/widget_view">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/widget_text_threat"
        android:layout_centerInParent="true">
    </TextView>
</RelativeLayout>
```

上面的布局已经简单得不能再简单了，是吧？它只拥有一个包裹在 RelativeLayout 布局中的 TextView 控件，用于显示“风险等级”信息。现在，你可以在应用程序中将这个布局载入到 RemoteViews 对象中，以便能够在小插件提供器中使用。当我们把一个布局塞进 RemoteViews 对象中后，事情会变得更复杂。

为了将布局资源（比如先前定义的 wedget.xml）载入到 RemoteViews 对象中，你需要使用下面的代码：

```
RemoteViews remoteView =
    new RemoteViews(context.getPackageName(), R.layout.widget);
```

当你想要修改布局中 TextView 控件的文本时，你需要使用 RemoteViews 类的 setTextViewText()方法，如下所示：

```
remoteView.setTextViewText(R.id.widget_text_threat, "Red alert!");
```

如果你想让用户能够在小插件的 RelativeLayout 控件内通过单击启动底层应用程序，那么可以使用 RemoteViews 类的 setOnClickPendingIntent()方法。例如，下面的代码创建了一个 PendingIntent，它可以启动 SimpleAppWidgetActivity Activity：

```
Intent launchAppIntent =
    new Intent(context, SimpleAppWidgetActivity.class);
PendingIntent launchAppPendingIntent = PendingIntent.getActivity(
```

Download at http://www.pin5i.com/

```
        context, 0, launchAppIntent,
        PendingIntent.FLAG_UPDATE_CURRENT);
    remoteView.setOnClickPendingIntent
        (R.id.widget_view, launchAppPendingIntent);
```

最后，当 RemoteViews 对象完全设置好后，小插件提供器需要告诉小插件管理器有关更新的 RemoteViews 对象的信息：

```
ComponentName simpleWidget = new ComponentName(context,
    SimpleAppWidgetProvider.class);
AppWidgetManager appWidgetManager =
    AppWidgetManager.getInstance(context);
appWidgetManager.updateAppWidget(simpleWidget, remoteView);
```

要更新某个小插件，AppWidgetManager 对象需要其组件名称（component name）。之后，小插件管理器会使用你在 updateAppWidget()方法中提供的 RemoteViews 对象的内容更新具有特定名称的小插件的内容。每一次 RemoteViews 对象更新时，都会被重新构建。虽然并不会频繁地进行更新，但还是需要尽量把它设计得简单一点，以提高性能。

Note

遗憾的是，SimpleAppWidget 示例中 AppWidgetProvider 类的完整实现过于冗长，不便在这里引用。你可以查看示例工程中的 SimpleAppWidgetProvider 类来了解“风险等级”小插件的实现细节。

（3）更新应用程序小插件

当小插件提供器的 onUpdate()方法被调用时，一个标识符列表将会被传入。每一个标识符代表着该提供器下的一个特定小插件实例。也就是说，用户可以向宿主中添加多个某种类型的小插件。在更新事件期间，每一个标识符都需要被遍历并且单独对每一个 RemoteViews 对象进行更新（如果你同时支持多个不同实例存在的话）。

小插件在更新期间必须要有很强的响应性，因为它是在宿主进程的 UI 线程上执行的。当更新完成后，并不保证小插件提供器对象仍然存在，所以，如果某个小插件的刷新是一个耗时的操作，那么它必须使用服务，以便能够在后台创建一个单独的线程执行这项操作。

在“风险等级”小插件示例中，我们已经实现了一个服务，用于执行一些网络操作以下载最新的风险等级数据。这个服务能够同时满足小插件和应用程序的需求，所以它们可以共享它。很方便，不是吗？

4. 创建小插件更新服务

大多数应用程序小插件并不会包含静态的内容，而是不断地进行更新。通常来讲，Android 服务会被用来为小插件提供内容更新，包括生成线程、连接 Internet 等。我们可以在小插件更新时调用小插件提供器的 onUpdate()方法中启动更新服务。当服务完成自己的工作后，它应该被关闭，直到下次更新时再启动。让我们再来看看“风险等级”小插件，它使用了两个服务。

① SimpleDataUpdateService 类在小插件更新时运行（在小插件提供器的 onUpdate()方法中）。服务连接到 Internet，检查当前的风险等级，并且将结果存储到应用程序的共享首选项（preference）中。最后，服务关闭自身。可以把应用程序理解成服务的“拥有者”，服务通过将数据存储在共享首选项中，同时向应用程序和小插件提供信息。

Download at http://www.pin5i.com/

② PrefListenerService 类监听应用程序首选项的改变。除了使用 onUpdate() 方法外，服务也会在小插件开启时启动，这样当数据改变时也可以进行更新（例如，当底层应用程序自己检查了风险等级进而修改了共享首选项后）。当风险等级首选项改变时，服务将触发对小插件提供器 updateAppWidget() 方法的调用，它将继而更新小插件的 RemoteViews 对象，从而绕过了小插件管理器的更新频率限制。我们可以把小插件理解成服务的“拥有者”，它在小插件的生命周期中运行，并且只有在更新小插件的内容时存在。

当然，也有一些更简单的方法来实现小插件的更新。例如，小插件可以使用它的某一个服务来完成所有的工作，包括下载风险等级数据和更新小插件的内容，不过这样应用程序就只有靠自己了。这里描述的方法显示了如何绕过小插件更新频率的限制，并且在小插件及其底层应用程序间共享内容。

Tip

更新 RemoteViews 对象不一定非要在小插件提供器中触发。它也可以直接从应用程序中被调用。在这里的例子中，用于下载风险等级数据的服务同时被应用程序和小插件所使用。使用服务来实现在线数据的下载有许多好处。不过，如果没有下载服务可供使用，我们也可以只使用一个服务来完成所有的功能。这个服务完全由小插件所控制，我们不仅可以用它来实现下载，还可以直接用来更新 RemoteViews 对象。当然，这种方式也失去了在小插件服务中监听共享首选项改变的机会。

5. 为小插件配置 Android Manifest 文件

为了让 Android 系统了解应用程序的小插件，你需要在应用程序的 Android Manifest 文件中包含<receiver>标记，以将其注册成小插件提供器。小插件经常会使用服务，并且这些服务需要在 Android Manifest 文件中使用<service>标记进行注册，这与其他的服务是一样的。下面是 SimpleAppWidget 工程中 Android Manifest 文件的代码片段：

```
<receiver android:name="SimpleAppWidgetProvider"
    android:label="@string/widget_desc"
    android:icon="@drawable/threat_levels_descriptions">
    <intent-filter>
        <action android:name=
            "android.appwidget.action.APPWIDGET_UPDATE" />
    </intent-filter>
    <meta-data
        android:name="android.appwidget.provider"
        android:resource="@xml/simple_widget_info" />
</receiver>
<service android:name="SimpleDataUpdateService" />
<service android:name="SimpleAppWidgetProvider$PrefListenerService" />
```

注意，不同于普通的<receiver>定义，<meta-data>部分引用了一个 XML 文件资源。<receiver>标记包含有关小插件配置的一些信息，包括小插件的一个标签（label）和一个图标，它们将显示在小插件的“选择器（picker）”中（就是用户从系统中选择需要的小插件的地方）。<receiver>标记还包含一个 Intent 过滤器，以便处理 android.appwidget.action.APPWIDGET_UPDATE Intent 动作，以及一个<meta-data>标记，它引用了存储在 XML 资源目录中的小插件配置文件。最后，注册了用于更新小插件的服务。

Download at http://www.pin5i.com/

22.2.2 安装应用程序小插件

在应用程序实现了小插件的功能之后，用户（安装你的应用程序的人）可以按照以下步骤将其安装到主页中。

① 长按主页。

② 从弹出的菜单中选择 Widgets 选项，如图 22.1（左）所示。

③ 在 Widget 菜单中，选择你想添加的小插件，如图 22.1（右）所示。

④ 如果有足够的空间，小插件将被显示在屏幕上，如图 22.2 所示。你可以在屏幕上随意移动这个小插件，或者把它拖进位于主页底部的“垃圾筐”中移除它。

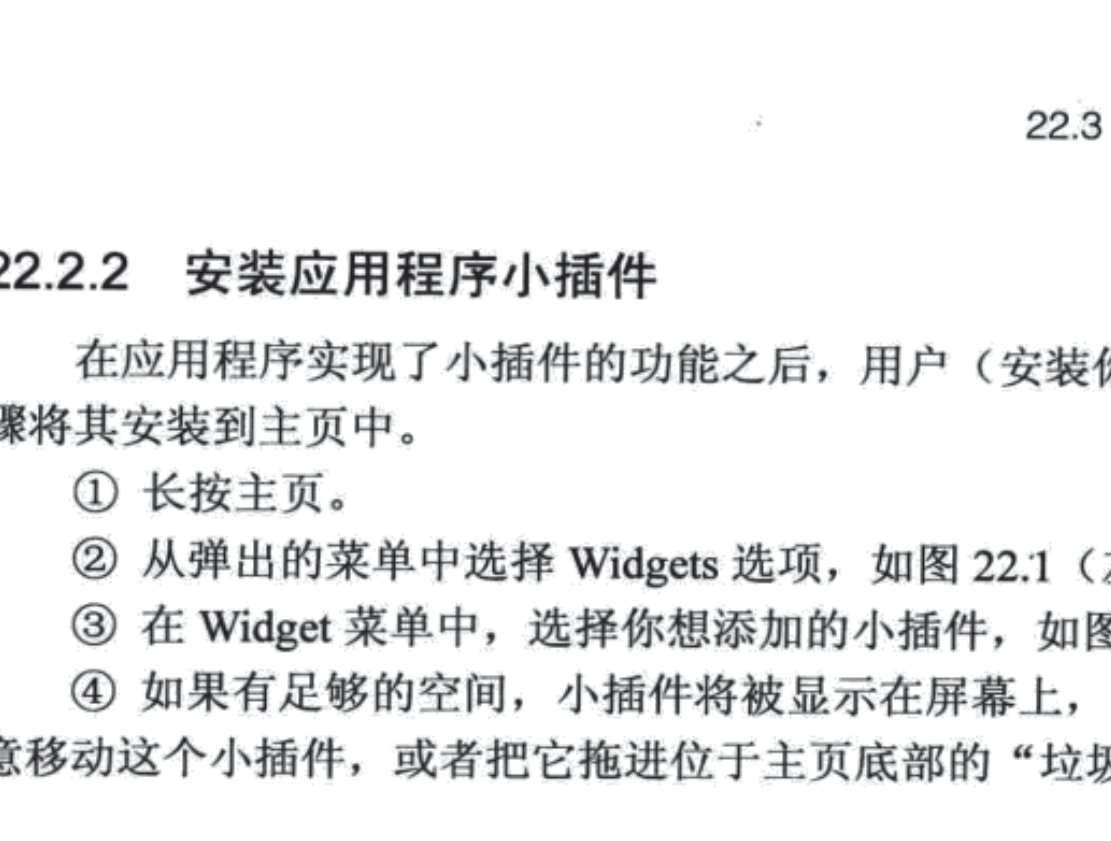


图 22.1 使用 Widget 选择器来安装小插件到主页中

图 22.2 主页上的一个简单的小插件，它显示了安全风险等级

22.2.3 成为小插件“宿主（Host）”

应用程序也可以成为小插件“宿主”。小插件“宿主”（android.appwidget.AppWidgetHost）只是用于嵌入和显示小插件的简单容器。最常见的小插件宿主就是主页。要获取更多有关成为小插件“宿主”的信息，可以参考 Android SDK 文档。

22.3 使用动态壁纸（Live Wallpaper）

除了静态的壁纸图片外，Android 支持动态壁纸的概念。用户可以设定一个具有交互性的壁纸，也就是动态壁纸，而不是简单的静态背景图片。在动态壁纸上可以显示任何东西，包括图像和动画。动态壁纸在 Android 2.1（API Level 7）中被引入。

你的应用程序可以使用 3D 图形、动画以及有趣的应用程序内容作为动态壁纸。下面是动态壁纸中可以包含的一些对象的例子：

① 描绘抽象图形的 3D 动画场景；

Download at http://www.pin5i.com/

② 从在线图片服务上获取图片，然后使用本地服务切换显示；
③ 具有交互性的池塘，当用手点击时会泛起涟漪；
④ 会随季节、天气和时间变化的壁纸。

Tip

静态图片壁纸的有关内容在第 15 章“使用 Android 多媒体 API”中有详细讨论。

22.3.1 创建动态壁纸

动态壁纸类似于 Android 的服务，只不过它提供一个用于显示其他对象的平面。为了支持动态壁纸，你需要对你的应用程序做如下修改。

① 提供一个 XML 的壁纸配置。
② 提供对 WallpaperService 类的实现。
③ 修改 Android Manifest 文件，以注册壁纸服务，以及授予适当的许可权限。

现在，让我们来看看具体的细节。

Tip

这一部分中的许多代码示例都来自于 SimpleLiveWallpaper 应用程序。在本书的网站上可以下载该应用程序的源代码。

1. 创建动态壁纸服务

动态壁纸功能的核心由 WallpaperService 的实现提供，且大部分的功能由 WallpaperService.Engine 实现驱动。

（1）实现壁纸服务

你的应用程序需要继承 WallpaperService 类，其中，需要覆写的最重要的方法为 onCreateEngine()。下面是 SimpleDroidWallpaper 服务实现的壁纸代码示例：

```
public class SimpleDroidWallpaper extends WallpaperService {
    private final Handler handler = new Handler();

    @Override
    public Engine onCreateEngine() {
        return new SimpleWallpaperEngine();
    }
    class SimpleWallpaperEngine extends WallpaperService.Engine {
        // 你的壁纸服务引擎实现由此开始...
    }
}
```

上面这个壁纸服务并没有什么功能。onCreateEngine()方法只是简单地返回应用程序的自定义壁纸引擎，而由它提供具体动态壁纸的所有功能。你还可以在必要时覆写其他的壁纸服务方法。另外，代码中还初始化了一个 Handler 对象，以便进行壁纸的绘制操作。

（2）实现壁纸服务引擎

现在我们更仔细地来看看壁纸服务引擎的实现。壁纸服务引擎处理在具体的动态壁纸生命周期中的所有细节。这与第 17 章“通过 OpenGL ES 使用 Android 3D 图形”中的图形示例

Download at http://www.pin5i.com/

非常相似，动态壁纸使用一个 Surface 对象来实现屏幕绘制。

壁纸服务引擎里有许多很有用的回调方法，如下所示。

① 你可以覆写 onCreate()和 onDestroy()方法，以便建立和删除动态壁纸。在生命周期的这个阶段，Surface 对象是无效的。

② 你可以覆写 onSurfaceCreated()和 onSurfaceDestroyed()方法，以建立和删除用于进行动态壁纸绘制的 Surface。

③ 你需要覆写 onVisibilityChanged()方法来处理壁纸的显示。当其不可见时，壁纸不能继续运行。对该方法的处理与对 Activity 的暂停或返回事件的处理类似。

④ onSurfaceChanged()是另外一个用于 Surface 管理的方法。

⑤ 你可以覆写 onOffsetsChanged()方法以使动态壁纸能够响应用户对主页的切换（通过在屏幕上轻扫）。

⑥ 你可以覆写 onTouchEvent()方法来处理触屏事件。它需要一个 MotionEvent 对象作为其输入参数（我们将在第 24 章“处理高级用户输入”中讨论有关 MotionEvent 类的更多细节）。另外，你还需要使用 setTouchEventsEnabled()方法开启动态壁纸的触屏事件（默认情况下是关闭的）。

动态壁纸的实现细节取决于开发人员。通常，动态壁纸的实现会使用 OpenGL ES 调用来进行屏幕绘制。例如，本书提供的示例动态壁纸就实现了一个能在屏幕上像弹球一样移动的“机器人”，它还能够以改变方向的方式对触摸事件做出响应。这个动态壁纸的引擎使用了一个线程来管理绘制操作，并且使用壁纸服务中定义的 Handler 对象将结果送回给系统。

Tip

动态壁纸可以对触屏等用户事件做出响应，它还能监听用户向屏幕添加新内容的事件。更多有关这方面的内容，你可以参考文档中的 WallpaperService.Engine 类。

Note

遗憾的是，由于调用 OpenGL ES 屏幕绘制代码的缘故，SimpleLiveWallpaper 示例应用程序中壁纸引擎的完整实现过于冗长，不便在这里引用。你可以查看示例工程中的 SimpleDroidWallpaper 类来了解更多的实现细节。

Warning

在设计动态壁纸时，你还应该将设备的响应性和电池电量纳入进来综合考虑。

2. 创建动态壁纸配置

接下来，应用程序需要提供以 XML 方式实现的壁纸定义，你可以将它存储在工程资源目录/res/xml 下。例如，下面是一个名为/res/xml/droid_wallpaper.xml 的简单壁纸定义：

```
<?xml version="1.0" encoding="utf-8"?>
<wallpaper xmlns:android="http://schemas.android.com/apk/res/android"
    android:thumbnail="@drawable/live_wallpaper_android"
    android:description="@string/wallpaper_desc" />
```

Download at http://www.pin5i.com/

在上面的示例中，<wallpaper> XML 标记内包含了一个简单的壁纸定义。其中，description 和 thumbnail 属性会显示在壁纸选择器（wallpaper picker）中，这个选择器可供用户选取他们需要的壁纸。

3. 为动态壁纸配置 Android Manifest 文件

最后，你需要修改应用程序的 Android Manifest 文件，以便发布动态壁纸服务。特别地，WallpaperService 需要使用<service>标记进行注册，其中应该包含下面一些重要的信息：

① WallpaperService 类；
② BIND_WALLPAPER 许可权限；
③ 针对 WallpaperService 动作的 Intent 过滤器；
④ 引用动态壁纸配置的壁纸元数据。

我们来看一个例子，其中的<service>标记实现了一个简单的动态壁纸：

```
<service
    android:label="@string/wallpaper_name"
    android:name="SimpleDroidWallpaper"
    android:permission="android.permission.BIND_WALLPAPER">
    <intent-filter>
        <action
            android:name="android.service.wallpaper.WallpaperService" />
    </intent-filter>
    <meta-data
        android:name="android.service.wallpaper"
        android:resource="@xml/droid_wallpaper" />
</service>
```

除了服务的定义以外，你还需要使用<uses-sdk> Manifest 标记将应用程序所需的 API Level 限制为 7 或以上（以便提供对动态壁纸的支持）：

```
<uses-sdk android:minSdkVersion="7" android:targetSdkVersion="8" />
```

需要注意的是，你实现的动态壁纸所使用的 API（比如 OpenGL ES 2.0 API）可能会需要比 7 更高的 API Level，所以你需要根据实际情况适当地修改 minSdkVersion 属性。你还可能需要使用<uses-feature>标记，比如当你的应用程序需要支持动态文件夹时，以便能够为 Android Market 提供需要的信息：

```
<uses-feature android:name="android.software.live_wallpaper" />
```

22.3.2 安装动态壁纸

当你实现了应用程序对动态壁纸的支持后，就可以按照以下步骤将其装在主页上了。

① 长按主页。
② 从弹出的菜单中，选择“壁纸”选项，如图 22.3（左）所示。
③ 从“壁纸”菜单中，选择“动态壁纸”选项，如图 22.3（中）所示。
④ 从“动态壁纸”菜单中，选择你将要安装的动态壁纸，如图 22.3（右）所示。
⑤ 在选择了动态壁纸后，它将在预览模式下显示。然后只需要选择“设置壁纸”按钮确认即可。之后，动态壁纸将显示在主页上，如图 22.4 所示。

Download at http://www.pin5i.com/

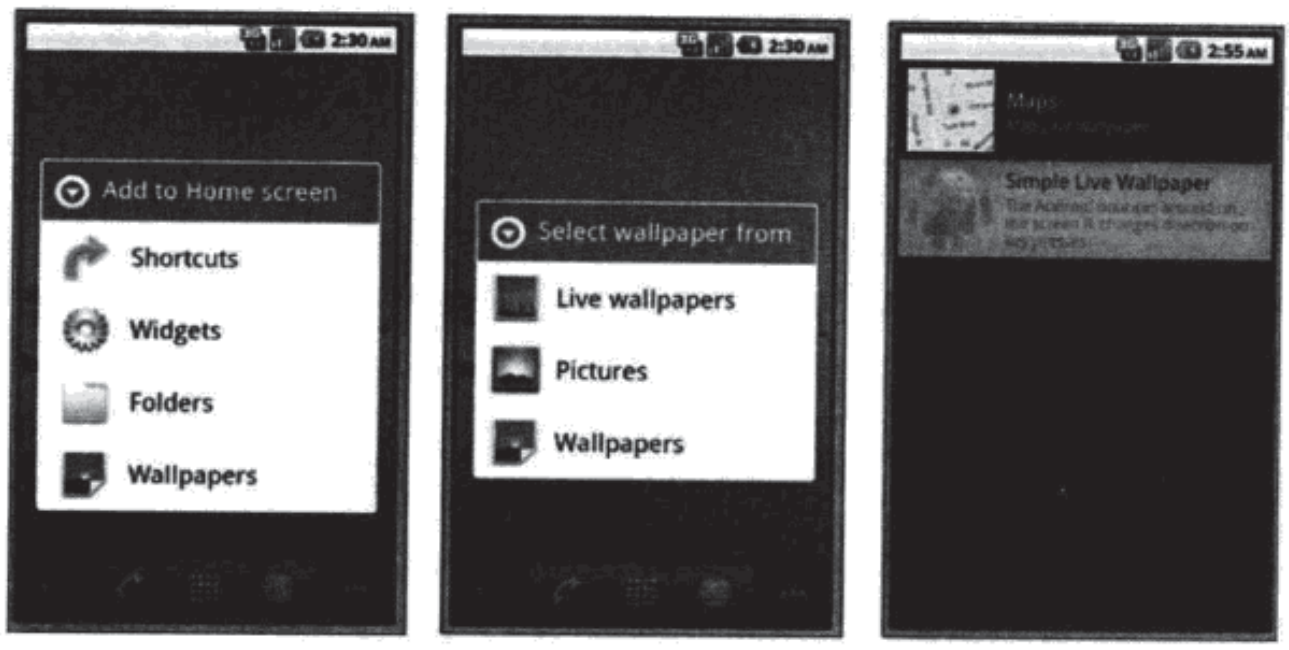


图 22.3 将动态壁纸安装到主页上

图 22.4 显示在主页的动态壁纸

22.4 成为内容类型处理器（Content Type Handler）

你的应用程序可以作为一个内容类型过滤器（content type filter），也就是说，能够处理针对某些 MIME 类型的诸如 VIEW、EDIT 或者 SEND 这样的 Intent 动作。

Tip

查看 android.content.Intent 类以获得标准 Activity 动作的列表。

一个照相机应用程序可能需要实现为内容类型处理器，以处理各种格式的 VIEW 动作，比如 JPG、PNG 或者 RAW 图像文件。类似地，一个社交网络应用程序可能需要处理 SEND 这样的 Intent 动作，以便在底层数据包含某些 MIME 类型（例如文本、图像或者视频）时能够处理。这意味着，当用户试图使用带有 SEND 动作的 Intent 发送数据（即这个社交网络应用程序关心的 MIME 类型）时，这个社交网络应用程序将会出现在可选择的列表中，供用户选择用来完成 SEND 请求。如果用户选择了这个应用程序发送内容，那么应用程序就会运行一个 Activity 来处理这一请求（例如，一个用于将内容上传到某个社交网站上的 Activity）。

Download at http://www.pin5i.com/

最后，内容类型处理器使得应用程序很容易就能够成为一个内容提供器，提供搜索或者动态文件夹的功能。使用自定义的 MIME 类型定义数据记录，这样，无论 Intent 是如何触发的（在应用程序以内或之外），相关动作都可以被应用程序流畅地处理。

要让你的应用程序成为内容类型处理器，你需要对其做如下修改。

① 确定你应用程序需要处理的 Intent 动作和 MIME 类型。

② 你需要实现一个 Activity，以便处理 Intent 动作或者你想处理的动作。

③ 你还需要使用<activity>标记在应用程序的 Android Manifest 文件中注册这个 Activity，这与平常是一样的。然后，你需要为这个 Activity 配置一个<intent-filter>标记，以提供应用程序可以处理的 Intent 动作和 MIME 类型。

22.5 确定 Intent 动作和 MIME 类型

让我们来看一个简单的示例。在本章的剩下部分中，我们将不断地在一个示例的基础上进行改动，这个应用程序使用的一个简单的内容提供器来提供非洲野生狩猎动物的记录，每一条记录都有标题和正文（正文的内容来自我们自己几年前在非洲时写下的有关野生狩猎动物的博文，这些博文颇受小学生们的欢迎）。在整个例子中，应用程序作为一个内容类型处理器，针对自定义的 MIME 类型处理 VIEW 请求：

```
vnd.android.cursor.item/vnd.androidbook.live.fieldnotes
```

Tip

MIME 类型有两种形式，大多数开发人员对此都很熟悉，比如 text/plain 或者 image/jpeg（在 RFC 2045 & RFC 2046 中定义），它们作为标准被广泛地使用。因特网地址编码分配机构（The Internet Assigned Numbers Authority，IANA，http://www.iana.org）负责管理这些 MIME 类型。

开发人员通常需要创建自己的 MIME 类型，但并不需要它成为全球标准。当然，这些类型必须足够唯一，不能与 MIME 类型的命名相冲突。当你在实现 Android 内容提供器时，有两个很好的前缀可供使用。ContentResolver.CURSOR_DIR_BASE_TYPE 前缀（“vnd.android.cursor.dir”）用于目录或文件夹；而 ContentResolver.CURSOR_DIR_BASE_TYPE 前缀（“vnd.android.cursor. item”）用于具体的某个类型。斜杠后面的部分则必须要唯一。在包名称或者其他唯一标识符的后面接续 MIME 类型的方式并不常见。

22.5.1 实现 Activity 处理 Intent

接下来，应用程序需要一个 Activity 类来处理接收到的 Intent。例如，我们需要载入一个页面来显示野生记录。下面是这个 Activity 实现的示例，它将解析 Intent 数据并且显示具体某种动物的野生记录：

```
public class SimpleViewDetailsActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.details);
        try {
```

Download at http://www.pin5i.com/

```
Intent launchIntent = getIntent();
Uri launchData = launchIntent.getData();
String id = launchData.getLastPathSegment();
Uri dataDetails = Uri.withAppendedPath
    (SimpleFieldnotesContentProvider.CONTENT_URI, id);
Cursor cursor =
    managedQuery(dataDetails, null, null, null, null);
cursor.moveToFirst();
String fieldnoteTitle = cursor.getString(cursor
    .getColumnIndex(SimpleFieldnotesContentProvider
    .FIELDNOTES_TITLE));
String fieldnoteBody = cursor.getString(cursor
    .getColumnIndex(SimpleFieldnotesContentProvider
    .FIELDNOTES_BODY));
TextView fieldnoteView = (TextView)
    findViewById(R.id.text_title);
fieldnoteView.setText(fieldnoteTitle);
            TextView bodyView = (TextView) findViewById(R.id.text_body);
            bodyView.setLinksClickable(true);
            bodyView.setAutoLinkMask(Linkify.ALL);
            bodyView.setText(fieldnoteBody);
        } catch (Exception e) {
            Toast.makeText(this, "Failed.", Toast.LENGTH_LONG).show();
        }
    }
}
```

SimpleViewDetailActivity 类使用 getIntent()方法获取用来启动 Activity 的 Intent。然后检查 Intent 的细节，使用 getLastPathSegment()方法抽取特定的野生记录标识符。剩下的代码向底层内容提供器提出特定野生记录的查询请求，并使用布局进行显示。

22.5.2 注册 Intent 过滤器

最后，Activity 类必须在应用程序的 Android Manifest 文件中进行注册，并且配置 Intent 过滤器，这样，应用程序仅仅接收特定动作的 Intent 和特定的 MIME 类型。例如，SimpleViewDetailsActivity 的注册代码如下：

```
<activity
    android:name="SimpleViewDetailsActivity">
    <intent-filter>
        <action
            android:name="android.intent.action.VIEW" />
            <category
                android:name="android.intent.category.DEFAULT" />
            <data android:mimeType =
                "vnd.android.cursor.item/vnd.androidbook.live.fieldnotes" />
    </intent-filter>
</activity>
```

<activity>标记剩下的部分与一般情况下相同。<intent-filter>标记是这里比较重要的部分。首先，使用<action>标记定义了应用程序想要处理的动作，具体为 VIEW。然后，<category>标记设定为了 DEFAULT，这也是比较普遍的情况。最后，<data>标记进一步将 VIEW Intent 限定为了与野生记录相关联的 MIME 类型。

Download at http://www.pin5i.com/

Tip

本章示例应用程序的剩下部分（SimpleSearchIntegration 和 SimpleLiveFolder）实现了上面应用程序的内容类型处理器。这些应用程序源代码可以在本书网站上下载，你可以进一步地研究实现它们的方法。

22.6 让应用程序的内容可搜索

如果你的应用程序具有很多的内容，无论是用户建立的还是你提供的，如果开发人员提供了搜索的功能，那么将大大提升其用户体验。这样，应用程序的数据就成为了整部手机的一部分，获得更容易访问的特性，从而能够得到更多的运行机会。

大多数的 Android 设备都拥有一组类似的硬件按钮：Home（首页）、Menu（菜单）、Back（后退）和 Search（搜索）。使用 Android 框架，应用程序可以实现强大的搜索功能。有两种方法可以在 Android 应用程序中引入搜索功能。

① 应用程序实现一个可以响应用户点击 Search 按钮的框架，并且在应用程序内执行搜索。

② 应用程序可以向全局提供其内容，系统范围内的搜索包含应用程序和网络内容。

搜索框架所提供的功能包括搜索和访问应用程序的数据（搜索结果），同时还能够在用户输入搜索内容时给出提示。当用户选择某一项搜索建议后，应用程序还能够提供一个 Intent。

Tip

本章中的许多代码示例都来自于 SimpleSearchIntegration 应用程序。在本书的网站上可以下载到它的源代码。

我们来看看之前提到过的那个“非洲野生记录”的应用程序。它使用了一个简单的内容提供器来提供野生动物的信息。在这个应用程序中实现搜索是一个很好的选择，它能够让用户快速地检索到某个特定的动物。当结果找到以后，应用程序需要使用一个 Intent 来运行适当的页面以显示它的野生记录，这是实现一个简单的内容类型处理器以让应用程序处理“查看野生记录”动作的很好的机会，如图 22.5 所示。

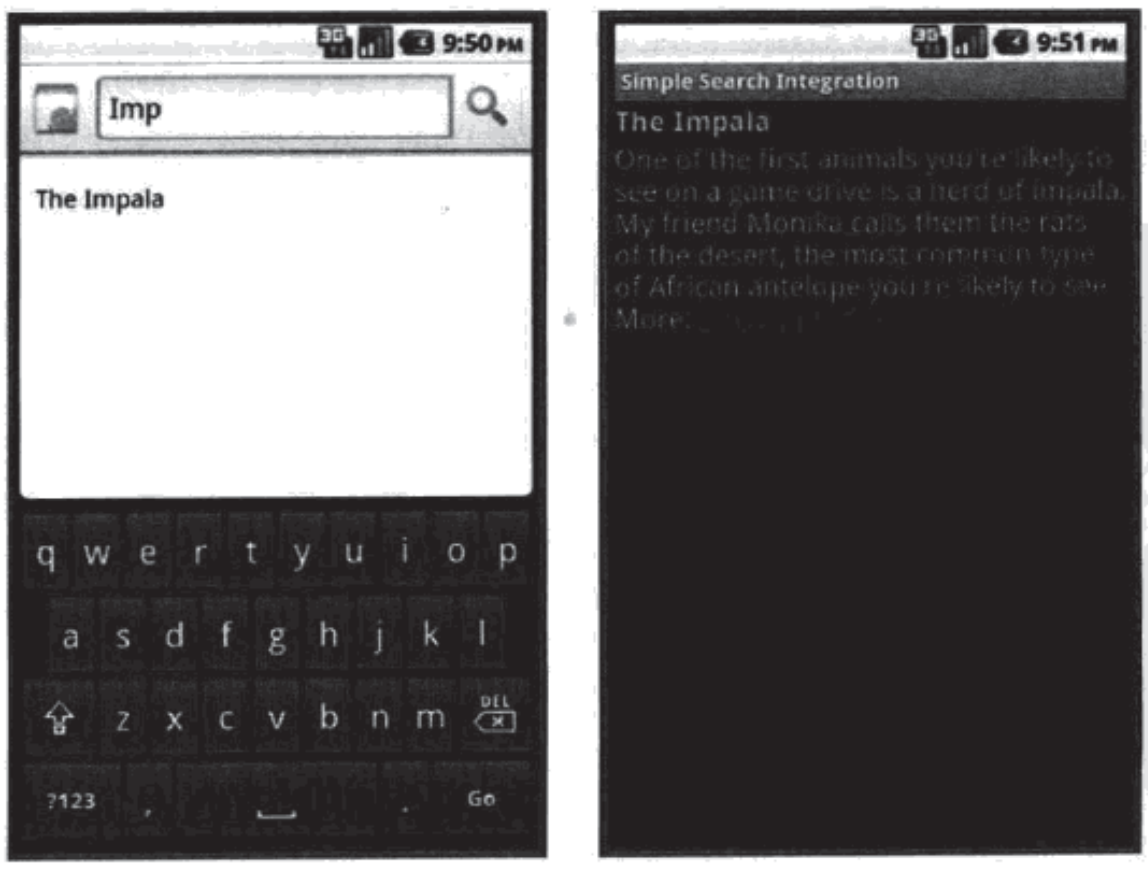


图 22.5 处理应用程序内部搜索以及搜索结果

Download at http://www.pin5i.com/

22.6.1 在应用程序内启用搜索

为了启用搜索，你需要对应用程序做出许多修改。虽然这些改动看起来会很复杂，但一旦完成了，后面启用全局搜索的过程就会非常简单。实现内容搜索需要你的应用程序成为内容提供器，或者至少具有某种类型的可以进行搜索的底层数据库。

Note

SearchManager 类（android.app.SearchManager）提供的搜索框架并不实际地执行搜索请求，而是由开发人员完成。SearchManager 类仅仅是管理搜索服务以及控制搜索对话框。如何搜索数据、搜索什么数据以及返回什么结果都是具体的实现细节。

要在应用程序中使用搜索，你需要实现下面步骤。

① 开发一个具有数据的应用程序，理想的情况是实现内容提供器。

② 创建一个 XML 搜索配置文件。

③ 实现一个 Activity 用于处理搜索。

④ 为搜索配置应用程序的 Android Manifest 文件。

下面我们来看看实现上面步骤的更多细节。

1. 创建搜索配置

为应用程序创建一个搜索配置就是创建一个带有特定搜索标记的 XML 文件。它通常存储在 XML 资源目录下（例如：/res/xml/searchable.xml）并且在应用程序的 Android Manifest 文件中引用。

（1）启用基本搜索

下面是“野生记录”应用程序的搜索配置示例：

```
<?xml version="1.0" encoding="utf-8"?>
<searchable
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:label="@string/app_name"
    android:hint="@string/search_hint"
    android:searchSettingsDescription="@string/search_settings_help">
</searchable>
```

搜索配置的基本属性相对比较简单。label 域通常设置为应用程序的名称（也就是提供搜索结果的应用程序名称）。hint 域用于显示无输入情况下，在搜索框的 EditText 控件内显示的提示文字。另外，如果愿意的话，你还可以进一步修改搜索按钮的文字和输入法选项。

（2）启用搜索建议

如果你的应用程序实现了内容提供器并且你想启用搜索建议，那么你需要在搜索配置中包含几个附加的属性。你需要指定用于提供搜索建议的内容提供器，包括它的“顾问（authority）”、路径信息以及用于返回搜索建议的查询。你还需要提供一些有关 Intent 的信息，这样，在用户点击特定搜索建议时能够触发它。

我们再次回到“野生记录”示例中。下面是用于支持搜索建议的搜索配置，它将在查询野生记录标题时给出提示：

Download at http://www.pin5i.com/

```
android:searchSuggestAuthority =
    "com.androidbook.simplesearchintegration
    .SimpleFieldnotesContentProvider"
android:searchSuggestPath="fieldnotes"
android:searchSuggestSelection="fieldnotes_title LIKE ?"
android:searchSuggestIntentAction="android.intent.action.VIEW"
android:searchSuggestIntentData = "content://com.androidbook.simplesearch
    integration.SimpleFieldnotesContentProvider/fieldnotes"
```

第一个属性 searchSuggestAuthority 用于设定处理搜索建议查询的内容提供器（“顾问”）。第二个属性定义了附加在 Authority 后面的路径。第三个属性提供了搜索查询的 SQL WHERE 子句（这里只有“记录标题”字段，而没有正文，这样可以让检索更快速）。而后，提供了一个 Intent 动作，即用户点击搜索建议。最后，定义了用于启动 Intent 的 Uri。

你还可以设定一个阈值（android:searchSuggestThreshold），它表示在进行搜索提示前用户需要键入的最小字符数。你可以考虑把这个值设定为 3 或者 4（默认为 0），这样可以减小搜索的命中率。在值为 0 时，甚至一个空的搜索域（没有任何过滤条件）都会显示搜索提示。

每次用户在开始输入搜索条件时，系统都将执行一次内容提供器查询以获取提示。所以，应用程序的内容提供器接口需要进行修改以便能够处理这些查询。为了实现这一目的，你需要定义一个内容提供器数据列向搜索框架中搜索建议列表内容的映射（projection）。例如，下面的代码定义了一个映射，它将野生记录的唯一标识符和标题分别映射到_ID 和 SUGGEST_COLUMN_TEXT_1 中，并且使用 SUGGEST_COLUMN_INTENT_DATA_ID 域的内容作为搜索提示：

```
private static final HashMap<String, String>
FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP;
static {
    FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP =
        new HashMap<String, String>();
    FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP.put(_ID, _ID);
    FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP.put(
        SearchManager.SUGGEST_COLUMN_TEXT_1, FIELDNOTES_TITLE + " AS "
        + SearchManager.SUGGEST_COLUMN_TEXT_1);
    FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP.put(
        SearchManager.SUGGEST_COLUMN_INTENT_DATA_ID, _ID + " AS "
        + SearchManager.SUGGEST_COLUMN_INTENT_DATA_ID);
}
```

每次在需要显示提示时，系统都会使用搜索配置中定义的 Uri 执行一次查询。所以不要忘记定义 Uri 并且在内容提供器的 UriMatcher 对象中注册它（使用 addURI()方法）。例如，“野生记录”应用程序使用下面的 Uri 作为搜索建议查询：

```
content:// com.androidbook.simplesearchintegration.
SimpleFieldnotesContentProvider/fieldnotes/search_suggestion_query
```

通过为内容提供器查询提供特殊的 Uri，你可以简单地修改内容提供器的 query()方法来处理特定的查询，包括构建映射、执行查询以及返回结果。下面我们来仔细看看野生记录内容提供器的 query()方法：

Download at http://www.pin5i.com/

```
@Override
public Cursor query(Uri uri, String[] projection, String selection,
    String[] selectionArgs, String sortOrder) {
    SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();
    queryBuilder.setTables(SimpleFieldnotesDatabase.FIELDNOTES_TABLE);
    int match = sURIMatcher.match(uri);
    switch (match) {
    case FIELDNOTES_SEARCH_SUGGEST:
        selectionArgs = new String[] { "%" + selectionArgs[0] + "%" };
        queryBuilder.setProjectionMap(
            FIELDNOTES_SEARCH_SUGGEST_PROJECTION_MAP);
        break;
    case FIELDNOTES:
        break;
    case FIELDNOTE_ITEM:
        String id = uri.getLastPathSegment();
        queryBuilder.appendWhere(_ID + "=" + id);
        break;
    default:
        throw new IllegalArgumentException("Invalid URI: " + uri);
    }
    SQLiteDatabase sql = database.getReadableDatabase();
    Cursor cursor =
        queryBuilder.query(sql, projection, selection,
        selectionArgs, null, null, sortOrder);
    cursor.setNotificationUri(getContext().getContentResolver(), uri);
    return cursor;
}
```

query()方法的实现处理了常规的内容查询和特殊的搜索建议查询（使用搜索建议 Uri 执行的查询）。当搜索建议查询到达时，我们在搜索条件的两边加上通配符，并使用 QueryBuilder 对象的 setProjectionMap()方法建立并执行一个查询。由于我们需要快速地返回结果，所以只搜索与标题相匹配的搜索条件，而非野外记录的正文。

Tip

在 SQLite 中使用通配符和 LIKE 表达式会降低查询的效率，我们可以使用 SQLite FTS3 扩展，它能够提供快速的全文查询功能。由于它对数据记录的条数有所限制，而且要求数据表的关系表达比较简单，所以不太适合我们这个例子。另外，这种方式不支持索引，所以查询效率会受到一定的影响。有关 SQLite FTS3 的更多信息，可以参考 http://www.sqlite.org/fts3.html。

（3）启用语音搜索

你还可以在应用程序中启用语音搜索。它可以让用户通过语音输入搜索条件，而非键入。启用语音搜索需要在搜索配置中添加几个属性。其中最重要的是 voiceSearchMode，它将开启语音搜索并且设定适当的模式：showVoiceSearchButton 值使搜索对话框中能够显示一个小的语音录制按钮；launchRecognizer 值告诉 Android 系统使用语音录制 Activity；而 launchWebSearch 值则将打开语音网络搜索 Activity。

下面给出了在“野外记录”应用程序中引入语音搜索的搜索配置代码：

```
android:voiceSearchMode="showVoiceSearchButton|launchRecognizer"
```

Download at http://www.pin5i.com/

其他一些你可以设置的语音搜索属性包括语音语言模型（自由形式或网络搜索）、语音语言、最大语音结果以及语音识别对话框的文本提示。更多细节可以参考 Android SDK 文档中有关“搜索配置（Searchable Configuration）”的内容：http://developer.android.com/guide/topics/search/searchable-config.html。

2. 创建搜索 Activity

下一步，你需要实现实际执行搜索的 Activity 类。它将在应用程序接收到带有 ACTION_SEARCH 动作的 Intent 时运行。

查询请求包含名为 SearchManager.QUERY 扩展域（extra field）中的搜索字符串。Activity 将使用这个值进行搜索，而后返回结果。

我们来看看“野外记录”示例中的搜索 Activity。你可以像下面这样实现一个名为 SimpleSearchableActivity 的搜索 Activity：

```
public class SimpleSearchableActivity extends ListActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Intent intent = getIntent();
        checkIntent(intent);
    }

    @Override
    protected void onNewIntent(Intent newIntent) {
        // update the activity launch intent
        setIntent(newIntent);
        // handle it
        checkIntent(newIntent);
    }

    private void checkIntent(Intent intent) {
        String query = "";
        String intentAction = intent.getAction();
        if (Intent.ACTION_SEARCH.equals(intentAction)) {
            query = intent.getStringExtra(SearchManager.QUERY);
            Toast.makeText(this,
                "Search received: " + query, Toast.LENGTH_LONG)
                .show();
        } else if (Intent.ACTION_VIEW.equals(intentAction)) {
            // pass this off to the details view activity
            Uri details = intent.getData();
            Intent detailsIntent =
                new Intent(Intent.ACTION_VIEW, details);
            startActivity(detailsIntent);
            finish();
            return;
        }
        fillList(query);
    }

    private void fillList(String query) {
        String wildcardQuery = "%" + query + "%";
        Cursor cursor =
```

Download at http://www.pin5i.com/

```
        managedQuery(
            SimpleFieldnotesContentProvider.CONTENT_URI,
            null,
            SimpleFieldnotesContentProvider.FIELDNOTES_TITLE
            + " LIKE ? OR "
            + SimpleFieldnotesContentProvider.FIELDNOTES_BODY
            + " LIKE ?",
            new String[] { wildcardQuery, wildcardQuery }, null);
    ListAdapter adapter =
        new SimpleCursorAdapter(
            this,
            android.R.layout.simple_list_item_1,
            cursor,
            new String[] {
                SimpleFieldnotesContentProvider.FIELDNOTES_TITLE },
            new int[] { android.R.id.text1 });
    setListAdapter(adapter);
    }

    @Override
    protected void onListItemClick(
        ListView l, View v, int position, long id) {
        Uri details = Uri.withAppendedPath(
            SimpleFieldnotesContentProvider.CONTENT_URI, "" + id);
        Intent intent =
            new Intent(Intent.ACTION_VIEW, details);
        startActivity(intent);
    }
}
```

上面的代码同时实现了 onCreate()和 onNewIntent()方法，这是因为 Activity 的 launchMode 被设置为了 singleTop。当用于点击 Search 按钮时，这个 Activity 可以启动搜索对话框，如同示例中的其他 Activity 一样。当用户执行一次搜索时，系统将运行 SimpleSearchableActivity——与用户看到的是同一个 Activity。我们并不想为搜索结果创建一系列的 Activity 栈，所以只在最顶端实现一个实例，这也就是 singleTop 的意义。

处理搜索相对比较简单。使用 managedQuery 调用，结果会以 Cursor 对象的形式获得，然后我们可以使用 SimpleCursorAdapter 对象来填充 Activity 类的 ListView 控件。

对于列表项的点击，这里的实现只是简单地创建了一个新的 VIEW Intent，然后让系统来处理这一事件。在这个例子中，另外一个 Activity 将显示适当的野生记录正文。为什么不直接启动 Activity 类呢？因为这样处理很简单，而且这种启动方式更灵活稳定。

当用户在点击搜索建议列表中的某一条时，这个 Activity 将接收到 ACTION_VIEW 而不是 ACTION_SEARCH。它将这一事件直接交给了另外一个视图 Activity 来处理，它被设计用来绘制更为细节的内容。

3. 为搜索配置 Android Manifest 文件

现在我们在应用程序的 Manifest 文件中注册上面的 Activity 类，包括配置与 ACTION_SEARCH 动作相关联的 Intent 过滤器。你还需要使用<meta-data> Manifest 文件标记来标明

Download at http://www.pin5i.com/

应用程序的可搜索性。

下面是 Android Manifest 文件中有关搜索 Activity 注册的代码片段：

```
<activity
    android:name="SimpleSearchableActivity"
    android:launchMode="singleTop">
    <intent-filter>
        <action android:name="android.intent.action.SEARCH" />
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
    </intent-filter>
    <meta-data
        android:name="android.app.searchable"
        android:resource="@xml/searchable" />
</activity>
```

这里配置的<activity>标记和一般的 Activity 的主要不同之处在于添加了针对 SEARCH 动作 Intent 的 Intent 过滤器。除此之外，代码中还包含一些元数据，以便能够为系统提供搜索配置细节。

下一步，让我们来看看在应用程序中使用 Search 按钮的例子。我们需要在<application>标记中添加<meta-data>代码段。

```
<meta-data
    android:name="android.app.default_searchable"
    android:value =
        "com.androidbook.simplesearchintegration.SimpleSearchableActivity"/>
```

<meta-data>标记配置了在整个应用程序中处理搜索结果的默认 Activity。这样，当按下 Search 按钮后，搜索对话框将从应用程序的任何一个 Activity 中弹出。如果你并不打算在每个 Activity 中都应用这一功能，那么你需要逐一地在想要启用搜索的 Activity 中添加定义。

Note

并不是所有的 Android 设备都具有 Search 按钮。如果你一定要在应用程序中使用它，那么可以考虑以其他的方式实现搜索，比如手动地向应用程序页面中添加 Search 按钮，或者将搜索选项实现在选项菜单里。

22.6.2 启用全局搜索

在启用了应用程序搜索之后，经过简单的几个步骤，我们就可以使它成为全局设备搜索的一部分。全局搜索通常在“快速搜索框（Quick Search Box）”中使用。为了在你的应用程序中启用全局搜索，你需要：

① 以我们前面描述过的应用程序内部搜索为基础；

② 修改搜索配置文件；

③ 修改设备的搜索设置，将你的应用程序添加到全局搜索中。

现在，我们来具体看看这些步骤。在这里，我们仍然使用与前面相同的应用程序——野生记录。图 22.6 显示了全局搜索框，它可以从主页启动。

Download at http://www.pin5i.com/

图 22.6 包含在全局搜索结果中的应用程序内容，用户可以在主页按下搜索按钮进行搜索

1. 为全局搜索修改搜索配置

修改现存的搜索配置非常简单。你只需要在配置中添加 includeInGlobalSearch 属性，并且像下面这样将其设为 true：

```
android:includeInGlobalSearch="true"
```

此时，你需要确保你的应用程序已经实现为了一个内容类型处理器，以处理提供给全局搜索的结果。这样，用户可以选择应用程序提供的搜索建议。所以这里你需要再次使用应用程序的内容类型处理器，以便在用户选择某个搜索建议时能够运行它。

Tip

你可以使用 SearchManager.INTENT_ACTION_GLOBAL_SEARCH Intent 启动全局搜索。

2. 为全局搜索修改搜索设置

无论如何，对于全局搜索的范围，用户具有最终的选择权。默认情况下，你的应用程序不会添加到全局搜索中，而需要显式地将其引入进来，为此用户需要执行“设置→搜索→可搜索的项”菜单命令，修改设备的搜索设置，如图 22.7 所示。

如果应用程序拥有适合全局搜索的内容，那么你可以为上面的搜索设置提供一个快捷方式。SearchManager 类的 INTENT_ACTION_SEARCH_SETTINGS Intent 可以实现这一效果：

```
Intent intent = new Intent(SearchManager.INTENT_ACTION_SEARCH_SETTINGS);
startActivity(intent);
```

上面这个 Intent 将启动搜索设置页，如图 22.7（左）所示。

在 Android 开发人员网站中的“Search Dev Guide（搜索开发指南）”部分有关于 Android 搜索框架更为专业的内容：http://developer.android.com/guide/topics/search/index.html。

Download at http://www.pin5i.com/

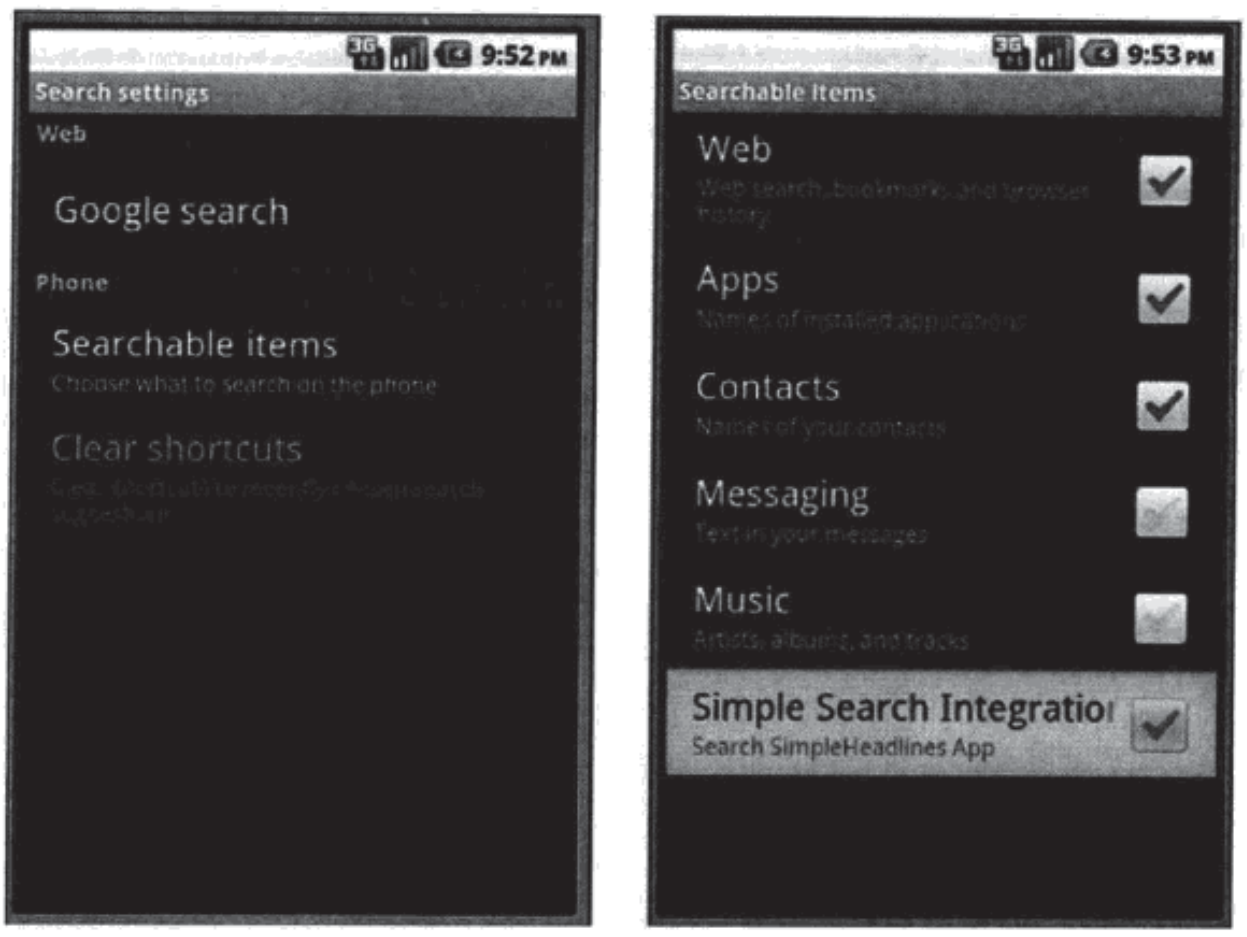


图 22.7 将你的应用程序添加到设备搜索设置中

22.7 使用动态文件夹

动态文件夹（Live Folder）是让你的应用程序丰富起来并且让用户更易使用的另一种途径。在 Android 1.5（API Level 3）中引入的动态文件夹是一种特殊类型的文件夹，用户可以将一些东西放入其中，比如主页，当用户点击其中的内容时，可以向作为内容提供器的应用程序发送查询，并显示其内容。文件夹中的每一个数据都有与其相关联的 Intent。你也可以把动态文件夹看做是一个装有应用程序快捷方式的文件夹。比如，一个音乐播放器可以让用户创建自己喜爱音乐的动态文件夹；一个日程表应用程序可以提供一个包含每日任务的动态文件夹；再如，一款游戏可以创建一个动态文件夹用来保存存盘点。应用程序可以在动态文件夹中存储各种不同类型的东西，这完全取决于应用程序想提供什么内容。

让我们再次回到前面那个非洲野生记录应用程序，对它进行一些修改，以便用户可以创建包含野生记录标题的动态文件夹。单击其中的某条记录则可以通过 VIEW 动作运行应用程序并显示完整的野生记录内容（这是通过内容类型处理器实现的，我们在本章前面部分已经讨论过了）。

Tip

这一部分提供的大部分代码都来自于应用程序 SimpleLiveFolder。该程序的源代码可以在本书的网站上下载。

22.7.1 创建动态文件夹

要在应用程序中创建内置的动态文件夹，你需要：

① 一个向外部提供数据的内容提供器；

② 一个用于处理动态文件夹中数据类型的内容类型处理器（例如：VIEW 野生记录）；

③ 实现一个 Activity 类来处理动态文件夹的创建；

④ 修改应用程序的内容提供器接口，以处理动态文件夹查询；

Download at http://www.pin5i.com/

⑤ 为动态文件夹配置应用程序的 Android Manifest 文件。

现在让我们来关注更多的细节。

1. 创建动态文件夹 Activity

任何支持动态文件夹的应用程序都需要实现一个动态文件夹创建 Activity，它将在触发 ACTION_CREATE_LIVE_FOLDER Intent 动作时运行。动态文件夹创建 Activity 只负责一件事情：返回动态文件夹配置的实例。如果你回忆 startActivityForResult() 方法的工作原理，你就知道动态文件夹创建 Activity 是如何工作的了。Activity 需要获取到达的 Intent，然后执行创建动态文件夹的请求、制作一个动态文件夹并且使用 setResult() 方法设置 Activity 的返回结果。setResult() 方法的参数 resultCode 表示动态文件夹是否创建成功，并且作为结果 Intent 的 data 参数返回具体的动态文件夹实例。

看起来有点复杂，是吧？我们还是来看一个例子吧。下面是“野生记录”应用程序动态文件夹创建 Activity 的实现：

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.provider.LiveFolders;

public class SimpleLiveFolderCreateActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        final Intent intent = getIntent();
        final String action = intent.getAction();
        if (LiveFolders.ACTION_CREATE_LIVE_FOLDER.equals(action)) {
            final Intent baseIntent = new Intent(Intent.ACTION_VIEW,
                SimpleFieldnotesContentProvider.CONTENT_URI);
            final Intent resultIntent = new Intent();
            resultIntent.setData(
                SimpleFieldnotesContentProvider.LIVE_URI);
            resultIntent.putExtra(LiveFolders.EXTRA_LIVE_FOLDER_NAME,
                getResources().getString(R.string.livefolder_label));
            resultIntent.putExtra(LiveFolders.EXTRA_LIVE_FOLDER_ICON,
                Intent.ShortcutIconResource.fromContext(
                this, R.drawable.foldericon));
            resultIntent.putExtra(
                LiveFolders.EXTRA_LIVE_FOLDER_DISPLAY_MODE,
                LiveFolders.DISPLAY_MODE_LIST);
            resultIntent.putExtra(
                LiveFolders.EXTRA_LIVE_FOLDER_BASE_INTENT,
                baseIntent);
            setResult(RESULT_OK, resultIntent);
        } else {
            setResult(RESULT_CANCELED);
        }
        finish();
    }
}
```

如你所见，SimpleLiveFolderCreateActivity 的生命周期很短。它等待 ACTION_

Download at http://www.pin5i.com/

CREATE_LIVE_FOLDER 请求，然后生成适当的 Intent 对象，并将其作为 Activity 结果的一部分返回。其中最重要的代码是创建一个名为 resultIntent 的新 Intent。这个 Intent 对象包含新动态文件夹实例的所有配置细节。setData()方法用于提供动态 Uri（填充文件夹数据的请求 Uri）。

上面的代码还为动态文件夹实例设置了几个扩展属性（extra），包括 label 和 icon，以及动态文件夹的显示模式。动态文件夹有几种显示模式：DISPLAY_MODE_LIST 值使所有的动态文件夹内容显示在一个 ListView 控件中（对于文字内容非常理想），DISPLAY_MODE_GRID 则使用一个 GridView 控件显示动态文件夹中的内容（非常适合用来显示图像内容）。最后，代码为每一个动态文件夹项设置了基底（base）Intent 对象。这里，基底 Intent 具有一个名为 VIEW 的动作类型，你已经很熟悉了，它将用在内容类型处理器技术上。有关动态文件夹配置的更多细节，可以参考 Android SDK 文档的 android.provider.LiveFolders 包。

2. 处理动态文件夹内容提供器查询

每次用户打开动态文件夹时，系统都会执行内容提供器查询。所以，应用程序的内容提供器接口需要进行修改以处理填充动态文件夹的查询数据。如同搜索建议，你需要定义一个从内容提供器数据列向动态文件夹数据的映射，动态文件夹内的显示模式可以为列表或网格。例如，下面的代码定义了从“野生记录”的唯一标识符和标题向动态文件夹项的映射：

```
private static final HashMap<String, String>
FIELDNOTES_LIVE_FOLDER_PROJECTION_MAP;
static {
    FIELDNOTES_LIVE_FOLDER_PROJECTION_MAP = new HashMap<String, String>();
    FIELDNOTES_LIVE_FOLDER_PROJECTION_MAP
        .put(LiveFolders._ID, _ID + " AS "
        + LiveFolders._ID);
    FIELDNOTES_LIVE_FOLDER_PROJECTION_MAP.put(
        LiveFolders.NAME, FIELDNOTES_TITLE
        + " AS " + LiveFolders.NAME);
}
```

当用户打开动态文件夹时，系统根据动态文件夹配置中提供的 Uri 执行一次查询。所以不要忘记定义动态 Uri 地址，并且在内容提供器的 UriMatcher 对象中注册（使用 addURI()方法）。例如，“野生记录”应用程序使用下面的 Uri：

```
content:// com.androidbook.simplelivefolder.
SimpleFieldnotesContentProvider/fieldnotes/live
```

通过为内容提供器查询提供一个特殊的动态文件夹 Uri，你可以简单地修改内容提供器的查询方法，以处理特定的查询，包括构建映射、执行查询以及返回显示在动态文件夹中的结果。让我们来仔细看看野生记录内容提供器的 query()方法：

```
@Override
public Cursor query(Uri uri, String[] projection, String selection,
    String[] selectionArgs, String sortOrder) {
    SQLiteQueryBuilder queryBuilder = new SQLiteQueryBuilder();
    queryBuilder.setTables(SimpleFieldnotesDatabase.FIELDNOTES_TABLE);
    int match = sURIMatcher.match(uri);
    switch (match) {
    case FIELDNOTE_ITEM:
        String id = uri.getLastPathSegment();
```

Download at http://www.pin5i.com/

```
            queryBuilder.appendWhere(_ID + "=" + id);
            break;
        case FIELDNOTES_LIVE:
            queryBuilder.setProjectionMap(
                FIELDNOTES_LIVE_FOLDER_PROJECTION_MAP);
            break;
        default:
            throw new IllegalArgumentException("Invalid URI: " + uri);
        }
        SQLiteDatabase sql = database.getReadableDatabase();
        Cursor cursor = queryBuilder.query(sql,
            projection, selection, selectionArgs, null,
            null, sortOrder);
        cursor.setNotificationUri(getContext().getContentResolver(), uri);
        return cursor;
    }
```

上面的 query()方法处理了常规的内容查询和特殊的动态文件夹查询(也就是那些带有动态 Uri 的查询)。当动态文件夹查询发生时，我们使用 QueryBuilder 对象的 setProjectMap()方法像通常一样设置与执行。

3. 为动态文件夹配置 Android Manifest 文件

最后，我们需要在 Android Manifest 文件中使用带有 CREATE_LIVE_FOLDER 动作的 Intent 过滤器注册动态文件夹 Activity 类。例如，下面是一段摘自“野生记录”应用程序 Android Manifest 文件的代码：

```
<activity
    android:name="SimpleLiveFolderCreateActivity"
    android:label="@string/livefolder_label"
    android:icon="@drawable/foldericon">
    <intent-filter>
        <action
            android:name="android.intent.action.CREATE_LIVE_FOLDER" />
        <category
            android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

我们比较熟悉这种类型的 Activity 注册。SimpleLiveFolderCreateActivity 类负责处理 CREATE_LIVE_FOLDER Intent（由 Intent 过滤器决定），另外，你还可以设置动态文件夹的文本 label 和 icon。

Note

在动态文件夹选择器中使用的图标和文本标签与显示在动态文件夹中的之间没有关联。在 Android Manifest 文件中，你可以使用<activity>标记的 android:label 和 android:icon 属性分别设置供动态文件夹选择器使用的资源，这个<activity>标记与处理 android.intent.action.CREATE_LIVE_FOLDER 动作的 Activity 相对应。而针对主页（或其他动态文件夹宿主）上各个动态文件夹实例的图标和标签则是在 Activity 类处理 LiveFolders.ACTION_CREATE_LIVE_FOLDER 时作为 Intent 的扩展域（extra fields）进行设置。

Download at http://www.pin5i.com/

22.7.2 安装动态文件夹

在应用程序能够处理动态文件夹的创建之后，你的开发过程就告一段落了。作为用户，你可以在主页上安装一个动态文件夹，如下所示。

① 长按主页上的空白处。

② 从弹出的菜单中选择“Folders（文件夹）”选项，如图 22.8（左）所示。

③ 从“Select folder（选择文件夹）”菜单中选择想要添加的文件夹，如图 22.8（右）所示。

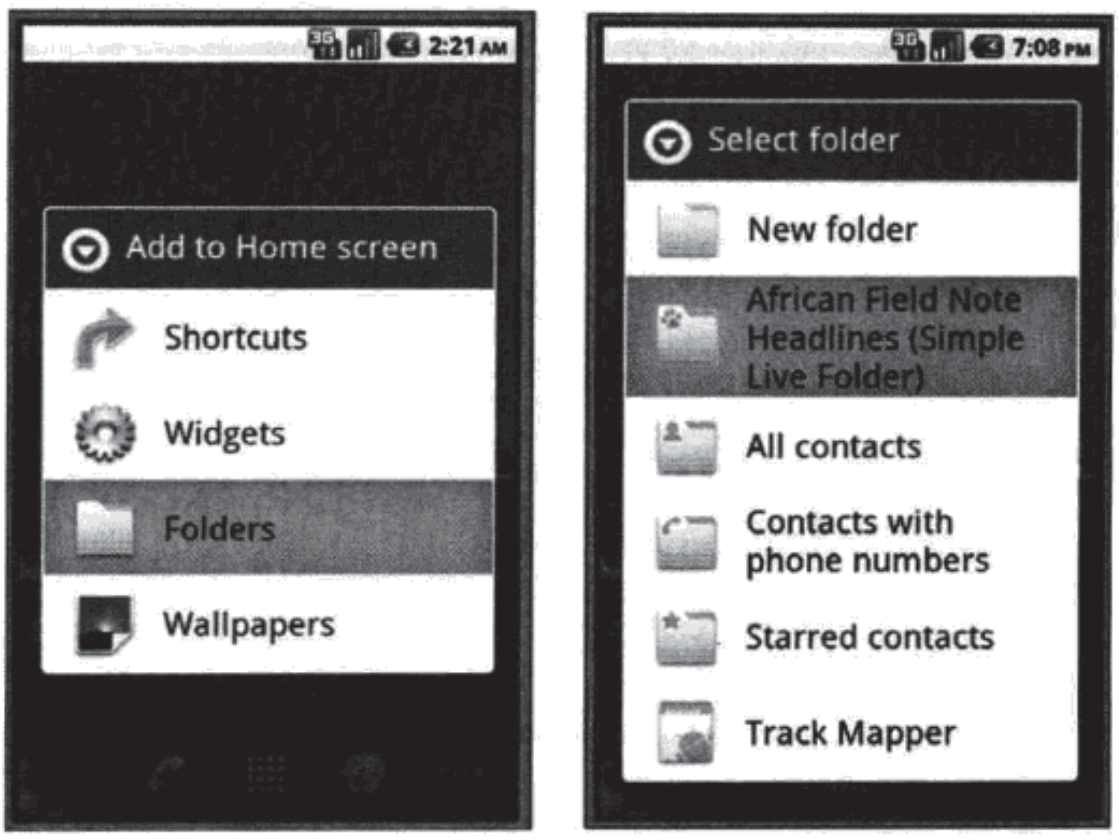


图 22.8 在主页上安装一个动态文件夹

④ 现在，你刚刚选择的动态文件夹显示在了主页上，如图 22.9 所示。

图 22.9 显示在主页上的一个动态文件夹

⑤ 如果你点击这个动态文件夹，它将打开并显示其中的内容（这由开发人员决定具体是什么）。选中其中的某一项将运行底层应用程序。例如，选择一个野生记录的标题将会运行用

Download at http://www.pin5i.com/

来显示特定记录及其细节的 Activity，如图 22.10 所示。

图 22.10　从动态文件夹中选择某一项野生记录（左）将运行应用程序以显示其内容（右）

小　结

Android 平台提供了一系列使应用程序与操作系统之间密合的途径，从而拓展了你应用程序的功能，大大超越传统应用程序之所能及。在本章中，你学习了如何通过创建简单的应用程序小插件、动态壁纸、动态文件夹等来扩展你的应用程序。你还学习到了如何在应用程序中启用搜索功能，以及如何将它引入到全局搜索中。

参考资料和更多信息

Our Series of Articles on App Widgets（我们有关应用程序小插件的一些文章）：http://j.mp/a8mpdH.

Android Dev Guide（开发指南）：App Widgets.http://developer.android.com/guide/topics/appwidgets/index.html.

Android Dev Guide（开发指南）：Search.http://developer.android.com/guide/topics/search/index.html.

Android Technical Articles（技术文章）：Live Wallpapers.http://developer.android.com/resources/articles/live-wallpapers.html.

Android Technical Articles（技术文章）：Live Folders.http://developer.android.com/resources/articles/live-folders.html.

Download at http://www.pin5i.com/

第23章

用户账户管理与数据同步

Android 支持云服务。Android 应用程序可以与远程服务无缝密合，使得用户可以方便地传输数据。Android 应用程序可以使用同步适配器（sync adapter）与远程云服务（Internet）进行数据同步。开发人员还可以使用 Android 基于云的备份服务来保护应用程序数据，安全而高效地进行数据转移。在本章中，你将学习有关用户账户和应用程序内置数据同步的内容，以及使用 Android 提供的备份和恢复功能保护应用程序数据。

23.1 使用账户管理器管理账户

从用户的角度讲，Android 2.0 平台引入了许多令人兴奋的设备功能。例如，用户可以注册并使用多个账号，用于处理电子邮件和联系簿的管理。这一功能的实现基于新引入的数据同步和账户服务，开发人员可以充分利用它们。虽然你可以使用账户和同步处理任何类型的数据，但一般情况下，Android 应用程序所提供的这类功能通常会与面向公司的商务应用相集成。

Android 用户账户使用 android.accounts 包中的类进行管理。它们通常用于与联系簿、电子邮件或其他类似信息相关的用户账户。比如，一个社交网络类应用程序可以包含好友的联系信息、好友近况等。这些信息通常使用 Android 设备所提供的同步服务进行分发与管理（我们将在本章后面部分详细地讨论有关同步的话题）。

首先，我们来讲讲用户账户。在 Android 账户管理器（Android Account Manager）中注册的账户提供了对某些类型数据的支持，联系人信息是其中的主要部分。AccountManager 类可以用来针对不同用户注册不同的账户。每一个账户包含用于使用服务的授权信息，通常是供在线使用的某些服务账户的授权证书。Android 服务，比如平台内置的同步服务，可以访问这些账户，并从其中挖掘出适当类型的数据（联系人信息仍然是最主要的，也可以包含其他一些信息，比如好友的状态等）。

接下来，我们来看看如何使用由 AccountManager 类和 Account 类提供的账户信息。需要访问服务器的应用程序可以从系统上注册一系列的账户。如果其中的某一个包含服务器所需的证书，那么应用程序就可以请求该账户的“令牌（token）”。之后，应用程序使用这一令牌登录远程服务器并访问其所提供的服务。这在保障证书安全性的同时，还使用户操作得到了很大的简化，因为无论有多少个应用程序使用某一个证书，用户只需要进行一次选择。所有的这些功能都需要通过 AccountManager 类来实现。对 getAccountByType() 方法可以获取一个账户列表，然后调用 getAuthToken() 方法则可以获取与某一个账户相关联

Download at http://www.pin5i.com/

的令牌，而应用程序则可以使用这个令牌安全地与 Web 服务等进行远程通信。

另一方面，与后台服务器进行证书验证的过程是由账户提供器（account provider）完成的，账户提供器是向用户提供账户并且控制应用程序获得授权证书令牌的服务。为了实现这些操作（处理向远程服务器验证 Account 对象的系统请求），账号提供器需要实现一个账户认证器（account authenticator）。通过认证器，账户提供器可以请求特定的证书，并且通过一些账户验证的机制（通常是线上服务器）进行确认。要实现一个账户验证器，你需要对应用程序做一些修改。首先要做的是实现 AbstractAccountAuthenticator 类。此外，你还需要修改应用程序的 Android Manifest 文件，提供验证器的配置文件（XML），并且提供验证器的配置页面，以便用户能够流畅地完成认证操作。

Tip

有关创建系统范围内账号的更多信息，可以参考 Android SDK 文档中有关 AbstractAccountAuthenticator 类的部分，而 AccountManager 类的文档则提供了更多有关使用账户的信息。

Android SDK 中提供的同步功能需要我们刚刚讨论过的账户类。这一服务主要为内置应用程序提供针对联系人信息、电子邮件以及日程安排等与后台数据存储之间进行同步的功能——将后台服务器数据映射到现有的内容提供器中。也就是说，这一服务一般不用于同步其他一些 Android 应用程序的数据。理论上讲，应用程序可以使用这一服务来进行数据同步，但更好的方式可能是实现一个内部的同步机制。这可以通过 AlarmManager 类将系统网络数据同步实现成计划任务，这或许需要使用 Android 服务。

如果你正在处理的数据很适合同步到内部应用程序中，比如可以放到内置应用程序或内容提供器中的联系人或者日程安排，那么你可以实现一个同步适配器（sync adapter）。这可以让 Android 系统来管理你的同步 Activity。

账户服务必须通过继承 stractThreadSyncAdapter 类实现同步适配器。当同步发生时，同步适配器的 onPerformSync() 方法将被调用，其参数将告诉适配器所使用的用户账户（通过 Account 对象进行定义），从而提供所需的认证令牌，而无需用户认证。一同告知适配器的还有待写入数据的内容提供器，以及数据所属的 Authority。

这样，当系统发出请求时，同步操作能够在自己的线程内执行。在同步期间，适配器从服务器获得更新信息，并且将其与指定内容提供器进行同步。具体的实现细节非常灵活，需要由开发人员确定。

Tip

Android 开发人员网站上提供了 Sync Adapter 示例应用程序，你可以从中学习到创建同步适配器的方法：http://developer.android.com/resources/samples/SampleSyncAdapter/。

23.2 使用备份服务

Android 2.2（API Level 8）中引入了 Android 备份服务。应用程序可以使用这一备份系统服务请求应用程序数据，比如需要备份或者恢复的共享首选项和文件。备份服务可以向远程备份服务发送备份包，也可以从远程备份服务获取。

Download at http://www.pin5i.com/

备份服务不应用被用于同步应用程序的内容，因为备份和恢复操作并不能即时完成，所以应该使用本章前面讨论过的同步策略，如同步适配器等。Android 备份服务应该只用于重要的应用程序数据。

Tip

这一部分中的许多代码示例都来自于 SimpleBackup 应用程序。在本书的网站上可以下载该应用程序的源代码。另外，你还需要使用 adb bmgr 命令来执行备份与恢复操作。更多的信息可以参考附录 C “Android 调试桥快速入门”。

23.2.1 选择远程备份服务

在实现应用程序数据备份时需要考虑的一个重要问题就是将数据备份到哪里。你选择的远程备份服务应该安全可依赖，并且要随时可用。Google 提供的 Android 备份服务（Android Backup Service）是许多开发人员的首选。

Note

也可以使用其他的第三方远程备份服务。如果你想要完全控制备份过程的话，可以考虑自己创建一个。不过，这已经超过了本书的范围。

为了在你的应用程序中使用 Android 备份服务，你需要在 Google 注册你的应用程序，并且获得一个唯一的备份服务 Key，以便在应用程序的 Manifest 文件中使用。

Android 备份服务网站提供签署 Google 备份服务 Key 的功能：http://code.google.com/android/backup/signup.html。

Warning

备份服务可以在大多数运行 Android 2.2 及以上版本的设备上使用，但并不是全部。其底层实现可能会有所不同。另外，不同的远程备份服务可能会具有对设备的不同限制。所以，你需要在目标设备上进行完整的备份测试，以确定你的应用程序是否真的支持备份。

在选择了远程备份服务之后，你还需要完成一些操作。以 Google 的 Android 备份服务为例，你需要注册一个特定的 Key。在获得了 Key 之后，你可以在应用程序的 Manifest 文件中使用<application>块的<meta-data>标记注册它：

```
<meta-data android:name="com.google.android.backup.api_key"
    android:value="KEY HERE" />
```

23.2.2 实现备份代理（Agent）

备份系统服务依赖于应用程序的备份代理（agent），以确定具体需要备份或恢复的应用程序数据。

1. 提供备份代理实现

现在，我们来实现应用程序的备份代理。备份代理决定了需要发送到备份服务的应用程序数据。如果你仅仅想要备份共享首选项数据和应用程序文件的话，可以简单地使用 BackupAgentHelper 类。

Download at http://www.pin5i.com/

Tip

如果你需要自定义应用程序备份数据的方法，那么你需要继承 BackupAgent 类，进而处理方法回调。onBackup()方法将在应用程序发出备份请求时调用，并且向备份服务提供待备份的应用程序数据。onRestore()方法在发出恢复请求时调用。备份服务提供数据文档，而 onRestore()方法则处理应用程序数据的恢复。

下面是实现备份代理类的示例：

```
public class SimpleBackupAgent extends BackupAgentHelper {
    @Override
    public void onCreate() {
        // 在这里注册 Helper
    }
}
```

你的应用程序备份代理需要为待备份的每一种数据类型包含备份 Helper。

2. 为共享首选项实现备份 Helper

要备份共享首选项文件，你需要使用 SharedPreferencesBackupHelper 类。添加对共享首选项的支持简单而直接。你只需要修改备份代理的 onCreate()方法，创建一个有效的 SharedPreferencesBackupHelper 对象，并且使用 addHelper()方法将其添加入代理：

```
SharedPreferencesBackupHelper prefshelper = new
SharedPreferencesBackupHelper(this,
    PREFERENCE_FILENAME);
addHelper(BACKUP_PREFERENCE_KEY, prefshelper);
```

上面的这个 Helper 将通过名称备份所有的共享首选项。这里，addHelper()方法使用了两个参数。

① Helper 的唯一名称（在这里，备份 Key 存储在一个名为 BACKUP_PREFERENCE_KEY 的 String 变量中）。

② 配置好的一个有效的 SharedPreferencesBackupHelper 对象，通过共享首选项的名称（在这里，首选项文件名存储在名为 PREFERENCE_FILENAME 的 String 变量中）控制备份和恢复。

就是这样。事实上，如果你的应用程序仅仅备份共享首选项，那么你并不需要实现备份代理类的 onBackup()和 onRestore()方法。

Tip

首选项里有不止一组数据怎么办？没有问题。SharedPreferencesBackupHelper 的构造函数可以接收任意数量的首选项文件名，而且仅仅需要一个唯一的 Key。

3. 为文件实现备份帮助

要备份应用程序文件，你可以使用 FileBackupHelper 类。备份文件比备份共享首选项稍微复杂一点，因为它们并不是线程安全（thread-safe）的。我们从修改备份代理的

Download at http://www.pin5i.com/

onCreate()方法开始，继而创建一个有效的 FileBackupHelper 对象，并且使用 addHelper()方法将其加入代理：

```
FileBackupHelper filehelper = new FileBackupHelper(this, APP_FILE_NAME);
addHelper(BACKUP_FILE_KEY, filehelper);
```

文件 Helper 通过文件名备份特定的文件。在上面的代码中，addHelper()方法使用了两个参数。

① Helper 的唯一名称（在这里，备份 Key 存储在一个名为 BACKUP_FILE_KEY 的 String 变量中）。

② 配置好的一个有效的 FileBackupHelper 对象，通过文件的名称（在这里，首选项文件名存储在名为 APP_FILE_NAME 的 String 变量中）控制备份和恢复。

Tip

有多个文件需要备份？没问题。FileBackupHelper 的构造函数可以接收任意数量的文件名，而且仅仅需要一个唯一的 Key。备份服务通常设计用于备份配置数据，而非所有的文件，目前还没有可备份文件大小的限制。例如，一个图书阅读器可以备份图书的标题和阅读状态，但不必备份书籍的内容。在恢复以后，这些数据可以用来再次下载图书的内容。对于用户而言，状态仍然是保持不变的。

你还需要确保应用程序中的所有文件操作都是线程安全的，因为备份请求很有可能会在某个文件打开时产生。Android 网站给出了有关在 Activity 内实现一个“锁”数组的方法，如下所示：

```
static final Object[] fileLock = new Object[0];
```

无论执行应用程序逻辑还是执行备份，每次进行文件操作时都可以使用这个锁。例如：

```
synchronized(fileLock){
    // 执行文件操作逻辑
}
```

最后，你需要覆写备份代理的 onBackup()和 onRestore()方法，并且确保所有的文件操作都使用了线程安全锁的机制进行。下面是一个备份代理的完整实现，它备份了一组名为 AppPrefs 的共享首选项和一个名为 appfile.txt 的文件：

```
public class SimpleBackupAgent extends BackupAgentHelper {
    private static final String PREFERENCE_FILENAME = "AppPrefs";
    private static final String APP_FILE_NAME = "appfile.txt";
    static final String BACKUP_PREFERENCE_KEY = "BackupAppPrefs";
    static final String BACKUP_FILE_KEY = "BackupFile";

    @Override
    public void onCreate() {
        SharedPreferencesBackupHelper prefshelper = new
            SharedPreferencesBackupHelper(this,
            PREFERENCE_FILENAME);
        addHelper(BACKUP_PREFERENCE_KEY, prefshelper);
        FileBackupHelper filehelper =
            new FileBackupHelper(this, APP_FILE_NAME);
        addHelper(BACKUP_FILE_KEY, filehelper);
```

Download at http://www.pin5i.com/

```
}
@Override
public void onBackup(ParcelFileDescriptor oldState,
    BackupDataOutput data, ParcelFileDescriptor newState)
    throws IOException {
        synchronized (SimpleBackupActivity.fileLock) {
            super.onBackup(oldState, data, newState);
        }
}
@Override
public void onRestore(BackupDataInput data, int appVersionCode,
    ParcelFileDescriptor newState) throws IOException {
        synchronized (SimpleBackupActivity.fileLock) {
            super.onRestore(data, appVersionCode, newState);
        }
}
```

在上面的代码中，为了让 doBackup()和 doRestore()方法线程安全，我们通过文件锁使用 synchronized 包装其父类。

4. 在应用程序 Manifest 文件中注册备份代理

最后，你需要在应用程序的 Manifest 文件中使用<application>标记的 android:backupAgent 属性注册备份代理类。例如，如果你的备份代理类名为 SimpleBackupAgent，那么可以使用其完成的路径名进行注册：

```
<application
    android:icon="@drawable/icon"
    android:label="@string/app_name"
    android:backupAgent="com.androidbook.simplebackup.SimpleBackupAgent">
```

23.2.3 备份和恢复应用程序数据

BackupManager 系统服务管理备份和恢复请求，它在后台运行在自己的计划任务中。实现了备份代理的应用程序可以请求执行备份或恢复，不过可能不会立即进行。要获得 BackupManager 的实例，简单地使用 Activity 类创建一个即可，如下所示：

```
BackupManager mBackupManager = new BackupManager(this);
```

1. 请求备份

应用程序可以使用 dataChanged()方法请求执行备份。通常来说，这个方法应该在应用程序发生改变时就会调用。它可以被调用很多次，但当执行备份的时间到达时，只会执行一次，而不管 dataChanged()方法之前被调用过多少次。

```
mBackupManager.dataChanged();
```

通常情况下，用户并不会主动地启动备份。相反，在保存应用程序更改的过程中，应该调用 dataChanged()方法。这样，在未来的某个时间点，备份管理器将在后台执行备份。

Download at http://www.pin5i.com/

Warning

避免在远程服务器上备份敏感数据。开发人员是最终对用户数据安全负责的人，而非你使用的备份服务提供商。

2. 请求恢复

在用户重置（reset）设备或者不小心让它掉进锅里、洗衣机里（比你想象的更常见）之后，恢复操作会自动地启动。当恢复开始后，用户数据将从远程备份服务器获取数据，而后应用程序备份代理将刷新相关数据，并且覆盖掉旧的。

应用程序可以使用 requestRestore()方法直接请求恢复。requestRestore()方法接收一个参数：RestoreObserver 对象。下面的代码显示了如何请求一次恢复：

```
RestoreObserver obs = new RestoreObserver(){
    @Override
    public void onUpdate(int nowBeingRestored, String currentPackage) {
        Log.i(DEBUG_TAG, "RESTORING: " + currentPackage);
    }

    @Override
    public void restoreFinished(int error) {
        Log.i(DEBUG_TAG, "RESTORE FINISHED! ("+error+")");
    }

    @Override
    public void restoreStarting(int numPackages) {
        Log.i(DEBUG_TAG, "RESTORE STARTING...");
    }
};

try {
    mBackupManager.requestRestore(obs);
} catch (Exception e) {
    Log.i(DEBUG_TAG,
        "Failed to request restore. Try adb bmgr restore...");
}
```

Warning

备份服务是 Android SDK 中相对较新的功能。目前，在模拟器上运行备份服务存在一些问题（抛出异常、服务丢失）。所以，对备份服务的测试最好能够在运行 Android 2.2 及以上版本的设备上进行，结合 adb bmgr 命令，我们可以实现备份或恢复的即时执行。

小　结

Android 应用程序并不活在真空中。用户需要他们的数据能够在他们所使用的所有设备间共享（当然是安全的），无论是手机掉进了锅里还是换了部新手机。Android 平台提供了维持本地应用程序数据与远程云服务同步性的服务，通过使用远程备份和恢复服务对用户数据进行保护。

Download at http://www.pin5i.com/

参考资料和更多信息

Wikipedia on Cloud Computing（维基百科“云计算”）.http://en.wikipedia.org/wiki/Cloud_computing.

Android Reference（Android 参考）：The AccountManager Clas.http://developer.android.com/reference/android/accounts/AccountManager.html.

Android Sample App（Android 示例程序）：Sample Sync Adapter.http://developer.android.com/resources/samples/SampleSyncAdapter/.

Android Dev Guide（开发指南）：Data Backup.http://developer.android.com/guide/topics/data/backup.html.

Google's Android Backup Service（Android 备份服务）.http://code.google.com/android/backup/index.html.

Download at http://www.pin5i.com/

第24章

处理高级用户输入

Android 设备上有许多用户交互模式，如键盘、轨迹球（trackball）、触屏手势，甚至语音。不同的设备支持不同的输入方法，并且拥有不同的硬件。例如，某些设备具有硬件键盘，而其他的则完全依赖软键盘。在本章中，你将学习到开发人员能够使用的各种不同的输入方法，以及它们能够为你的应用程序所带来的巨大影响。

24.1 使用文本输入法

Android SDK 提供一系列的输入法框架类，它们能够让开发人员使用强大的输入法，同时还能够创建自己的输入法，比如自定义软键盘和其他的输入法编辑器（Input Method Editors，IME）。用户可以下载自定义的 IME 供设备使用。开发人员可以创建许多有意思的自定义键盘，如《指环王》（*Lord of the Rings*）中的精灵族（Elvish）语字符、各种笑脸，或者希腊符号。

Tip

大多数设备的输入法设置可以通过“Settings（设置）→Language & Keyboard（语言和键盘）”菜单项启动。用户可以在这里选择语言、配置自定义用户词典，以及修改键盘设置。用户可以通过长按 `EditText` 控件来修改设备输入法，然后就会弹出一个上下文菜单，以供用户修改输入法（默认一般为 Android 键盘）。

24.1.1 使用软键盘

由于文本输入法依赖于本地信息（不同的国家使用不同的字母表和键盘）和具体情境（数字、字母或者特殊按键），Android 平台倾向于使用软键盘而非依赖硬件制造商所提供的专用硬件键盘。

1. 选择适当的软键盘

Android 平台拥有一系列可供选择的软键盘。指定每一个文本输入域的输入类型可以简化用户的输入过程，并且提供输入数据的效率。

Tip

这一部分中的许多代码示例都来自于 SimpleTextInputTypes 应用程序。在本书的网站上可以下载到它的源代码。

例如，要将某个 `EditText` 指定为只输入大写文本，你可以像下面这样设置 `inputType` 属性：

Download at http://www.pin5i.com/

```
<EditText android:layout_height="wrap_content"
    android:layout_width="fill_parent"
    android:inputType="text|textCapCharacters">
</EditText>
```

图 24.1 显示了一系列具有不同 inputType 配置的 EditText 控件。

输入类型决定了默认情况下所使用的软键盘，同时限定了一些规则，比如将输入字符限定在某个范围内。

图 24.2（左）显示了 EditText 控件的 inputType 属性设置为全大写情况下的软键盘。注意软键盘中的字符全部是大写的。如果你将 inputType 设置为 textCapWords，那么键盘将在键入每个词的首字母后转换为小写，然后在空格之后，又切换回大写。图 24.2（中）显示了 EditText 控件的 inputType 属性设置为 number 情况下的软键盘。图 24.2（右）显示了 EditText 控件的 inputType 属性设置为文本情况下的软键盘，此时每一句话的首字母将大写，并且文本可以为多行。

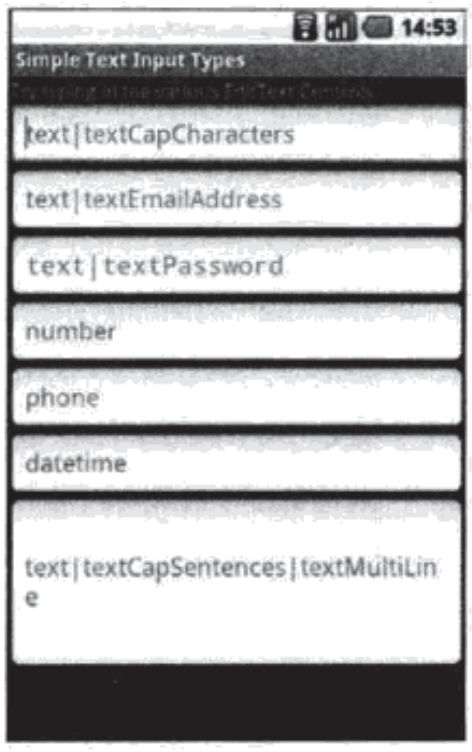


图 24.1 具有不同输入类型的 EditText 控件

图 24.2 软键盘的类型与输入类型相关联

如果用户在设备的键盘设置中开启了联想（Suggestions）选项和单词自动完成（Auto-complete）选项后，用户还可以看到输入联想或者在键入时进行的拼写修正。

inputType 属性值的完整列表以及它们各自的功能可以参考如下地址：http://developer.android.com/reference/android/R.attr.html#inputType。

Tip

你还可以让你的 Activity 对软键盘的显示做出响应（例如，调整显示的域），这可以通过请求 WindowManager 系统服务，并且修改与 softInputMode 域相关联的布局参数来实现。

更多有关输入法的微调控制，可以参考 android.view.inputmethod.InputMethodManager 类。

2. 提供自定义软键盘

如果你对开发自己的软键盘感兴趣的话，下面的参考或许可以帮助你。

① IME 实现为 Android 服务。你可以从一个名为 android.inputmethodservice

Download at http://www.pin5i.com/

和 android.view.inputmethod 包开始学习，它们可以用来实现自定义输入法。

② Android SDK 中提供的 SoftKeyboard 示例应用程序实现了一个软键盘。

③ Android 开发人员技术文档中有屏幕输入法的内容（http://developer.android.com/resources/articles/on-screen-inputs.html）和创建输入法的内容（http://developer.android.com/resources/articles/creating-input-method.html）。

24.1.2 使用文本预测和用户词典

文本预测是 Android 设备提供的一个强大而灵活的特性。我们已经在本书中的其他几个章节中讨论了很多有关它的话题，但它仍然值得在这里再次提及。

在第 7 章“探索用户界面屏幕元素”中，你学习了如何使用 AutoCompleteTextView 和 MultiAutoCompleteTextView 控件来帮助用户输入常用的单词和字符串。

在第 10 章“使用 Android 数据和存储 API”中，你学习了如何将底层 SQLite 数据表与 AutoCompleteTextView 控件相绑定。

在第 11 章“使用内容提供器在应用程序间共享数据”中，你学习了有关 UserDictionary 内容提供器的知识（android.provider.UseDictionary），它可以用来向自定义词典中添加常用的单词。

24.2 探索辅助功能（Accessibility）框架

Android SDK 提供了一系列方便视觉或听觉障碍用户使用的特性和服务，其他用户也能从中得到体验上的提升，特别是当他们没有在关注自己设备的时候（比如在开车时）。在 Android 1.6 和 2.0 中引入了许多强大的辅助功能，你需要在决定使用它们之前进行检查，确定你所使用的 API Level 是否包括所需的类或方法。Android SDK 提供了下面一些辅助功能。

① 语音识别框架。

② 语音合成（Text-To-Speech，TTS）框架。

③ 在 View 对象（API Level 3 及以上）上的触觉反馈（当你在按下某个按钮时感觉到的震动）。你可以查看 View 类的 setHapticFeedbackEnabled() 方法获取更多信息。

④ 在 View 对象（API Level 4 及以上）上设置相关的元数据，比如对 ImageView 控件的文本描述。这一功能对于有视觉障碍的用户非常有用。你可以查看 View 类的 setContentDescription() 方法获取更多信息。

⑤ 结合 Android 辅助功能（Accessibility）框架创建和扩展应用程序的辅助功能。你可以参考下面这些包，从它们开始学习应用程序的辅助功能：android.accessibilityservice 和 android.view.accessibility。另外还有许多已有的具有辅助功能的应用程序，如 KickBack、SoundBack 和 TalkBack，它们是平台自带的。更多的信息可以参考 Settings（设置）→Accessibility（辅助功能）中的设备设置。

Tip

你可以考虑在应用程序中提供辅助功能，比如提供视图元数据，因为用户会非常欣赏这些很小的细节，而且这些细节也决定了某些用户是否能够使用你的应用程序。同时，确保你的质量管理团队将这一部分的测试纳入了他们的工作流程。

Download at http://www.pin5i.com/

由于语音识别和语音合成应用程序非常流行，这些技术经常用在导航应用程序上（例如，美国某些州的法律规定在驾车时占用双手使用移动设备是违法的），所以我们更深入地讲解有关这两个技术的话题。

Android 应用程序可以使用语音输入和输出。语音输入可以通过使用语音识别服务获得，而语音输出则可以通过语音合成服务实现。并不是所有的设备都支持这些服务，但是某些类型的应用程序，特别是解放了双手的导航程序，却可以从这两个服务中大受裨益。

语音服务由 Android SDK 的 `android.speech` 包提供，而实现这些技术的底层服务可能会因设备的不同而不同，某些设备在使用这些功能时可能需要连接网络。

Tip

这一部分中的许多代码示例都来自于 SimpleSpeech 应用程序。在本书的网站上可以下载该应用程序的源代码。语音服务最好能够在真实的 Android 设备上进行测试，我们使用的是运行 Android 2.2 的 HTC Nexus One。

24.2.1 使用语音识别服务

你可以使用 Android SDK 提供的语音识别框架来提供应用程序的语音识别支持，从而增强其功能。语音识别的过程包括设备麦克风的语音采集、软件的探测和翻译，进而生成字符串。语音识别倾向于识别无停顿的短语，而不是很长的句子。如果你需要更多健壮的语音识别，那么需要实现自己的解决方案。

在 Android SDK 2.1 及更高版本上，语音识别被内置在了大部分的弹出键盘上。所以，应用程序可能无需任何修改就能直接支持语音识别。不过，对识别器的直接访问可以为应用程序带来更多有意义的识别控制。

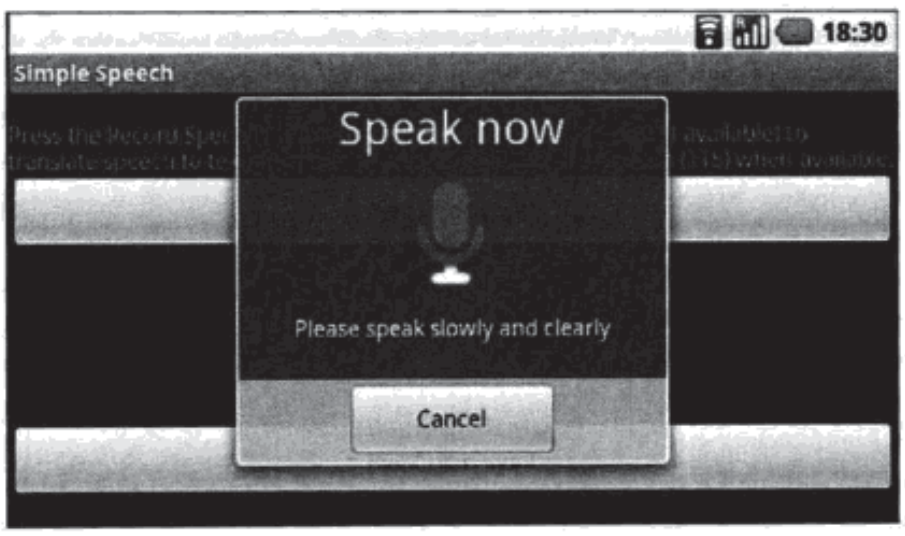


图 24.3 使用 `RecognizerIntent` 录制语音

你可以使用 `android.speech.RecognizerIntent` Intent 来启动内置的录音器，以录制用户的语音（如图 24.3 所示）。

之后，声音文件将发送到底层识别服务器上进行处理，所以，对于完全没有网络支持的设备而言，这一功能没有太大的实际意义。然后，你可以获取语音识别处理的结果，并在应用程序中使用它们。注意，对于一个语音片段，你可能会获得多个结果。

Note

语音识别技术不断地在演进与提升，但是在对着设备说话时，仍然需要发音准确、清晰可辨。有时在获得正确的翻译之前，可能需要尝试很多次。

下面的代码演示了如何在应用程序中通过 `RecognizerIntent` Intent 使用语音录制功能：

```
public class SimpleSpeechActivity extends Activity
{
    private static final int VOICE_RECOGNITION_REQUEST = 1;

    @Override
    public void onCreate(Bundle savedInstanceState) {
```

Download at http://www.pin5i.com/

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
    public void recordSpeech(View view) {
        Intent intent =
            new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
        intent.putExtra(RecognizerIntent.EXTRA_PROMPT,
            "Please speak slowly and clearly");
        startActivityForResult(intent, VOICE_RECOGNITION_REQUEST);
    }

    @Override
    protected void onActivityResult(int requestCode,
        int resultCode, Intent data) {
        if (requestCode == VOICE_RECOGNITION_REQUEST &&
        resultCode == RESULT_OK) {
            ArrayList<String> matches = data.getStringArrayListExtra(
                RecognizerIntent.EXTRA_RESULTS);
            TextView textSaid = (TextView) findViewById(R.id.TextSaid);
            textSaid.setText(matches.get(0));
        }
        super.onActivityResult(requestCode, resultCode, data);
    }
}
```

在上面的代码中，Intent 的启动由点击一个 Button 控件触发，它将调用 recordSpeech() 方法。RecognizerIntent 的配置如下。

① Intent 动作设置为 ACTION_RECOGNIZE_SPEECH，以便提示用户进行语音输入，并且发送声音文件供后期识别。

② 一个名为 EXTRA_LANGUAGE_MODEL 的 Intent 扩展属性（extra）设置为了 LANGUAGE_MODEL_FREE_FORM，以执行标准的语音识别。还有一种供网络搜索的语言模型：LANGUAGE_MODEL_WEB_SEARCH。

③ 一个名为 EXTRA_PROMPT 的 Intent 扩展属性被设置为了在用户进行语音输入时所显示的提示文字。

在配置好 RecognizerIntent 对象之后，可以使用 startActivityForResult() 方法启动 Intent，之后，结果将由 onActivityResult() 方法捕获，并在名为 TextSaid 的 TextView 中显示。我们这里只显示返回结果中的第一条。例如，用户按下按钮启动 record Speech() 方法，说出“We're going to need a bigger boat”，之后识别的文字将显示在应用程序的 TextView 控件中，如图 24.4 所示。

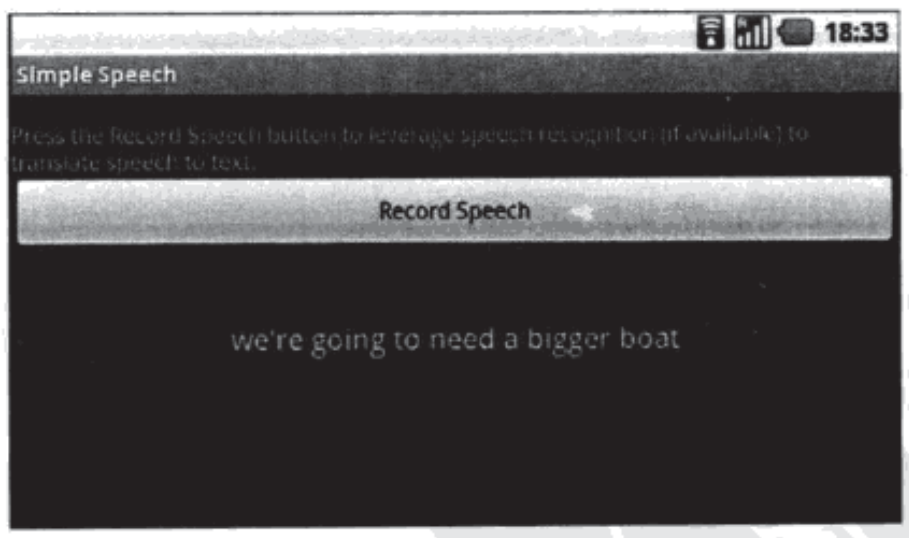


图 24.4 RecognizerIntent 返回的结果文本字符串

24.2.2 使用语音合成（TTS）服务

Android 平台包含 TTS 引擎（android.speech.tts），它可以让设备进行语音合成。你可以使用 TTS 引擎来让你的应用程序为用户“念”一段文字。你也许在许多位置

Download at http://www.pin5i.com/

服务（LBS）应用程序中见过这种功能，它可以让用户在不使用双手的情况下进行导航。其他应用程序则面向具有阅读或视觉障碍的用户提供这一功能。合成的语音可以立即播放或者保存成普通的音频文件。

> **Note**
>
> 要向用户提供 TTS 服务，Android 设备必须同时具有 TTS 引擎（Android SDK 1.6 及以上版本提供）和适当的语言资源文件。在某些情况下，用户需要从远程地址安装适当的语言资源文件（当然需要一定的空间），这可以通过 Settings（设置）→Voice Input & Output Settings（语音输入与输出）→Text-to-Speech（文字转语音设置）→Install Voice Data（安装语音数据）操作实现。不同于其他的设置页面，它并不具有定义在 `android.provider.Settings` 下的 Intent 动作。另外，应用程序可以验证需要的数据是否已经正确地安装了，如果没有，则可以在必要时启动安装程序。

举个例子，我们可以让设备读出我们刚才在语音识别示例中输入的文字。首先，我们需要修改 Activity 以实现 `TextToSpeech.OnInitListener` 接口，如下所示：

```
public class SimpleSpeechActivity extends Activity
    implements TextToSpeech.OnInitListener
{
    // 具体的类实现代码
}
```

然后，你需要在 Activity 内启动 TTS 服务：

```
TextToSpeech mTts = new TextToSpeech(this, this);
```

启动 TTS 引擎是异步的。`TextToSpeech.OnInitListener` 接口只有一个方法：`onInit()`，它会在 TTS 完成启动时调用（无论是否启动成功）。下面是 `onInit()` 方法的实现：

```
@Override
public void onInit(int status) {
    Button readButton = (Button) findViewById(R.id.ButtonRead);
    if (status == TextToSpeech.SUCCESS) {
        int result = mTts.setLanguage(Locale.US);
        if (result == TextToSpeech.LANG_MISSING_DATA
            || result == TextToSpeech.LANG_NOT_SUPPORTED) {
            Log.e(DEBUG_TAG, "TTS Language not available.");
            readButton.setEnabled(false);
        } else {
            readButton.setEnabled(true);
        }
    } else {
        Log.e(DEBUG_TAG, "Could not initialize TTS Engine.");
        readButton.setEnabled(false);
    }
}
```

我们使用 `onInit()` 方法来检验 TTS 引擎的状态。如果启动成功，那么名为 `readButton` 的 `Button` 控件将启用；否则，它将不可用。`onInit()` 方法也是配置 TTS 引擎的好地方。例如，你可以使用 `setLanguage()` 方法设置引擎所使用的语言。这里，我们将其设定为美式英语（American English）。那么 TTS 引擎所使用的语音将是美式发音。

Download at http://www.pin5i.com/

Note

Android TTS 引擎支持各种语言，包括英语（美式或英式）、法语、德语、意大利语和西班牙语。你可以在 onInit() 方法中像下面这样配置英式英语：

```
int result = mTts.setLanguage(Locale.UK);
```

我们可以使用下面这句话来测试美式英语与英式英语之间的区别："We adjusted our schedule to search for a vase of herbs in our garage"。

如果你愿意的话，可以把你喜欢的本地短语发给我们，我们将把它们发布在网站上。另外，我们也欢迎你发送一些有趣的语音识别错误（比如，"our garage" 经常被错误地识别成 "nerd haha"）。

最后，你可以进行文字到语音文件的转换了。这里，我们取用存储在 TextView 控件（在前面一节有关语音识别的内容中使用过的）中的文本字符串，使用 speak() 方法将其传入 TTS 中：

```
public void readText(View view) {
    TextView textSaid = (TextView) findViewById(R.id.TextSaid);
    mTts.speak((String) textSaid.getText(),
        TextToSpeech.QUEUE_FLUSH, null);
}
```

speak() 方法需要 3 个参数：需要朗读的字符串、排队策略和语音参数。排队策略决定了文本是添加到队列还是替换已有的文本，这里，我们使用 QUEUE_FLUSH 策略，这样只会朗读我们添加的文字。我们也不需要设定其他的特殊语音参数，所以只向第三个参数中传入 null 即可。最后，当你用完 TextToSpeech 引擎后，记得使用 shutdown() 方法释放资源（比如在 Activity 的 onDestroy() 方法中）。

```
mTts.shutdown();
```

现在，你只需将 Button 按钮的点击事件关联到 readText() 方法即可完成 TTS 的实现。当与之前的语音识别例子结合起来之后，应用程序则可以录制用户语音，翻译成字符串并将其显示在屏幕上，再把它朗读出来。实际上，这也是名为 SimpleSpeech 的示例所实现的功能。

24.3 使用手势

Android 设备的用户输入通常会依赖于触屏。用户已经非常习惯于在设备上使用手指动作进行操作了。Android 应用程序可以探测并响应单指（单点触控）和双指（多点触控）手势。

Note

稍早的 Android 设备也可能支持单点触控手势。但对多点触控的支持在 Android 2.2 SDK 中才被引入。当然，只有配备触屏硬件的设备才有触控功能。

手势识别中比较困难的地方在于多点触控事件，或者是手指动作（motion）。不同的动作序列合起来表示不同的手势。比如，Fling 手势包含三个过程：将手指按触在屏幕上，然后快速扫过，最后抬起手指，并且在抬起后手指仍然在运动（也就是说，在抬起手指前，运动的速度并不会减小）。每一步操作都会触发相应的运动事件，应用程序可以对此做出响应。

到目前为止，你已经理解了 Android 应用程序的用户界面是使用各种类型的 View 控件构建的。在 View 控件内，开发人员可以像处理普通的单击事件时使用 setOnClickListener()

Download at http://www.pin5i.com/

和 setOnLongClickListener()方法一样处理手势。onTouchEvent()回调方法用于探测 View 区域内的用户动作。

onTouchEvent()回调方法只接收一个参数：MotionEvent 对象。MotionEvent 对象包含在 View 内触发的所有类型动作的细节，通过收集和分析连续的 MotionEvent 对象，开发人员可以确定具体产生了何种手势。你可以使用 MotionEvent 数据来识别和探测任何你能想象的手势。不过，你也可以使用 Android SDK 内置的手势探测器来探测普通的用户动作。Android 目前拥有两个不同的类用于探测手势。

① GestureDetector 类可以用来探测一般的单点触控手势。

② ScaleGestureDetector 可以用来探测多点缩放操作手势。

在今后的 Android SDK 中，很有可能会加入更多的手势探测器。你也可以实现你自己的手势探测器来探测内置探测器所不支持的手势。例如，你可以创建一个探测两个手指旋转手势的探测器，用于诸如图像旋转这样的操作，或者还可以创建三个手指的手势，用于弹出选项菜单等。

除了普通的方向性手势外，你还可以使用 API Level 4 中引入的 android.gesture 包中的 GestureOverlayView 来识别命令手势。例如，你可以创建一个 S 形的手势来执行搜索，或者是 Z 形的手势在绘图应用程序中用于清除屏幕。有一些工具可以用来录制和创建这种类型手势的库。由于它覆盖了一些探测，所以并不适用于所有类型的应用程序。

Warning

触控屏的类型和灵敏度会因设备的不同而不同。不同的设备可以同时探测不同数量的触控点，这会直接影响到应用程序所支持的手势的复杂度。

24.4 处理普通单点触控手势

在 API Level 1 中引入的 GestureDetector 类可以用来探测单个手指的手势。GestureDetector 类所支持的单指手势包括以下几种。

- **onDown：** 当用户第一次按触屏幕时触发。
- **onShowPress：** 在用户按触屏幕，并且在抬起或移动手指之前触发；用于显式地指示按触事件确已探测到。
- **onSingleTapUp：** 作为单击（single-tap）事件的一部分，在用户从触屏上抬起手指（使用抬起 MotionEvent）时触发。
- **onSingleTapConfirmed：** 单击事件发生时调用。
- **onDoubleTap：** 双击（double-tap）事件发生时调用。
- **onDoubleTapEvent：** 在任何双击手势发生时调用，包括按下（down）、移动或抬起（up）MotionEvent。
- **onLongPress：** 与 onSingleTapUp 类似，但只在用户保持按触状态且不移动一定时间，并且不是标准的单击操作时调用。
- **onScroll：** 用户按下并且匀速移动手指后，在抬起手指前调用，通常也称为“拖曳（dragging）”。
- **onFling：** 在用户按下并且加速移动手指后，在抬起手指前调用，通常也称为“拂动（flick）”手势。

Download at http://www.pin5i.com/

你可以使用 GestureDetector 类中的接口监听特定的手势，比如单击和双击（参考 GestureDetector.OnDoubleTapListener），以及 Scroll 和 Fling 手势（参考 GestureDetector.OnGestureListener）。Scroll 手势包含按触屏幕，并移动手指；而 Fling 手势则通常会在手指抬起后仍然使对象继续移动。这给用户产生了在屏幕上投掷或拂动的印象。

Tip

你可以使用 GestureDetector.SimpleOnGestureListener 类来监听由 GestureDetector 识别的任何手势。

让我们来看一个简单的例子。我们假设你开发了一个游戏，可以让用户通过手势操作屏幕上的图像。我们可以创建一个名为 GameAreaView 的 View 类，它可以根据每一个手势确定位图图像在屏幕上的移动。GameAreaView 类可以使用 onTouchEvent() 方法并向 GestureDetector 传入 MotionEvent 对象。这样，GameAreaView 就可以响应并翻译简单的手势，进而对位图做出修改，比如将它们从屏幕的一个位置移到另一个位置。

Tip

如何翻译手势以及它们将触发怎样的操作完全由开发人员决定。比如，你可以将 Fling 手势翻译成去掉屏幕上的位图……但这有意义吗？在应用程序中手势的含义以及所执行的操作应该是符合逻辑的，以免造成用户混乱。不仅如此，用户已经习惯于某些特定的屏幕手势操作了，所以最好能够将它们沿袭下来。

在这里，GameAreaView 类可以像下面这样翻译手势。

① 双击手势可以让位图回到起初的位置。

② 拖曳（Scroll）手势可以让位图“跟随”手指的移动。

③ 拂动（Fling）手势可以让位图向拂动的方向“飞”去。

Tip

这一部分中的许多代码示例都来自于 SimpleGestures 应用程序。在本书的网站上可以下载该应用程序的源代码。

要启用手势，GameAreaView 类需要包含适当的手势探测器，它将触发对位图图像的操作。在探测到的具体手势的基础上，GameAreaView 类需要执行所有的手势翻译，以及对位图的相关操作。下面是编写 GameAreaView 类所需实现的几个重要方法。

① 类构造函数需要初始化所有用到的手势探测器和位图图像。

② 需要覆写 onTouchEvent() 方法，以便向手势探测器传递 MotionEvent 数据。

③ 需要覆写 onDraw() 方法，以便将位图图像绘制在适当的位置。

④ 执行图像移动、抛掷和复位操作所需的其他各种方法。

所有上面提及的方法都定义在 GameAreaView 类中：

```
public class GameAreaView extends View {

    private static final String DEBUG_TAG =
        "SimpleGesture->GameAreaView";
    private GestureDetector gestures;
    private Matrix translate;
    private Bitmap droid;
    private Matrix animateStart;
```

Download at http://www.pin5i.com/

```
private Interpolator animateInterpolator;
private long startTime;
private long endTime;
private float totalAnimDx;
private float totalAnimDy;

public GameAreaView(Context context, int iGraphicResourceId) {
    super(context);
    translate = new Matrix();
    GestureListener listener = new GestureListener(this);
    gestures = new GestureDetector(context, listener, null, true);
    droid = BitmapFactory.decodeResource(getResources(),
        iGraphicResourceId);
}

@Override
public boolean onTouchEvent(MotionEvent event) {
    boolean retVal = false;
    retVal = gestures.onTouchEvent(event);
    return retVal;
}

@Override
protected void onDraw(Canvas canvas) {
    Log.v(DEBUG_TAG, "onDraw");
    canvas.drawBitmap(droid, translate, null);
}

public void onResetLocation() {
    translate.reset();
    invalidate();
}

public void onMove(float dx, float dy) {
    translate.postTranslate(dx, dy);
    invalidate();
}

public void onAnimateMove(float dx, float dy, long duration) {
    animateStart = new Matrix(translate);
    animateInterpolator = new OvershootInterpolator();
    startTime = System.currentTimeMillis();
    endTime = startTime + duration;
    totalAnimDx = dx;
    totalAnimDy = dy;
    post(new Runnable() {
        @Override
        public void run() {
            onAnimateStep();
        }
    });
}

private void onAnimateStep() {
    long curTime = System.currentTimeMillis();
    float percentTime = (float) (curTime - startTime) /
        (float) (endTime - startTime);
    float percentDistance = animateInterpolator
        .getInterpolation(percentTime);
```

Download at http://www.pin5i.com/

```
            float curDx = percentDistance * totalAnimDx;
            float curDy = percentDistance * totalAnimDy;
            translate.set(animateStart);
            onMove(curDx, curDy);

            if (percentTime < 1.0f) {
                post(new Runnable() {
                    @Override
                    public void run() {
                        onAnimateStep();
                    }
                });
            }
        }
    }
```

如上面的代码所示，GameAreaView 类随时跟踪了位图图像应该绘制的位置。onTouchEvent()方法用来捕捉动作事件，并且将其传递给手势探测器，同时，我们还需要实现它的 GestureListener（稍后介绍）。通常而言，GameAreaView 的每一个方法都会对位图图像采取某种操作，继而调用 invalidate()方法，强制屏幕重绘。现在我们将注意力转移到实现具体手势所需的方法上。

① 对于双击手势，我们实现一个名为 onResetLocation()的方法来进行位图图像的复位，即在起初的位置重新绘制。

② 对于拖曳手势，我们实现一个名为 onMove()的方法，以便在新的位置绘制位图图像。

③ 对于拂动手势，事情会比较复杂。为了能够让屏幕动画流畅地进行，我们使用了一系列的异步调用和内置的 Android 插补器（interpolator），根据动画播放的时长计算当前图像的位置。拂动动画的完整实现可以参考 onAnimateMove()和 onAnimateStep()方法。

现在我们需要实现 GestureListener 类以翻译手势并且调用我们刚刚实现的 GameAreaView 的方法。下面是 GameAreaView 类所使用的 GestureListener 类的实现：

```
    private class GestureListener extends
            GestureDetector.SimpleOnGestureListener {

        GameAreaView view;

        public GestureListener(GameAreaView view) {
            this.view = view;
        }

        @Override
        public boolean onDown(MotionEvent e) {
            return true;
        }

        @Override
        public boolean onFling(MotionEvent e1, MotionEvent e2,
                final float velocityX, final float velocityY) {
            final float distanceTimeFactor = 0.4f;
            final float totalDx = (distanceTimeFactor * velocityX / 2);
            final float totalDy = (distanceTimeFactor * velocityY / 2);

            view.onAnimateMove(totalDx, totalDy,
                    (long) (1000 * distanceTimeFactor));
```

Download at http://www.pin5i.com/

```
            return true;
        }

        @Override
        public boolean onDoubleTap(MotionEvent e) {
            view.onResetLocation();
            return true;
        }

        @Override
        public boolean onScroll(MotionEvent e1, MotionEvent e2,
            float distanceX, float distanceY) {
            view.onMove(-distanceX, -distanceY);
            return true;
        }
}
```

注意，对于想探测的任何手势或动作，你都需要返回 true。所以，在拖曳手势开始时所触发的 onDown()方法需要返回 true。GestureListener 类方法的大部分实现都包含了对手势数据的翻译。例如如下。

① 我们使用 GameAreaView 类的 onResetLocation()方法响应双击事件，将位图复位到起初的位置。

② 我们使用 onScroll()方法中的距离数据来确定移动的方向，并将其传入 GameAreaView 类的 onMove()方法中。

③ 我们使用 onFling()方法中的速度数据来确定位图移动的方向和速度。TimeDistanceFactor 变量在这里设定为了 0.4，虽然这是一个主观设定的值，但它能够让整个“减速运动直至停止”的过程可见，同时又能让这一过程尽量短，以保障可控性和响应性。你可以把它想象成是在一个高摩擦力的平面上。GameAreaView 类的 onAnimateMove()方法中使用了这一信息。

现在我们完整地实现了 GameAreaView 类，你可以将其显示在屏幕上了。例如，你可以创建一个具有 FrameLayout 控件的 Activity，并且使用 addView()方法添加一个 GameAreaView 的实例。图 24.5 显示了拖曳和拂动手势的效果。

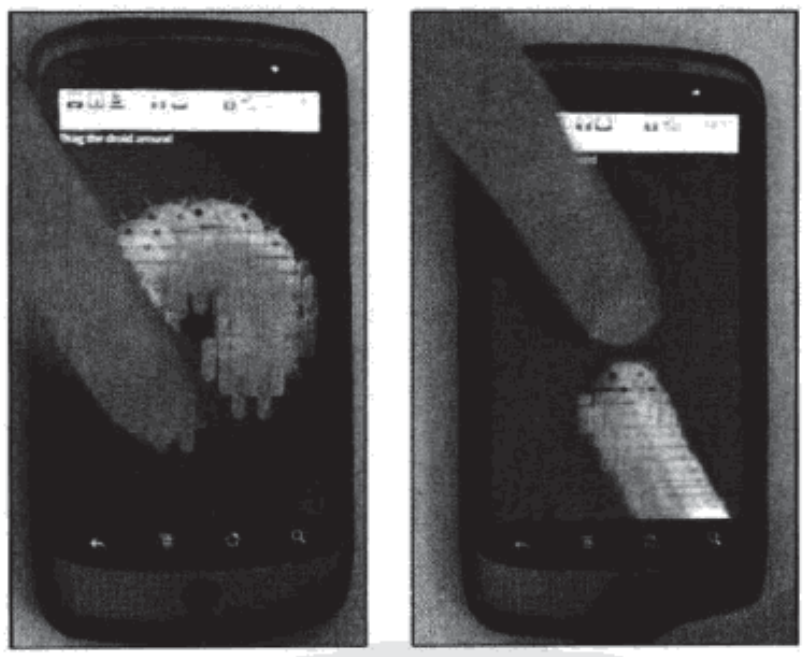


图 24.5 拖曳（左）和拂动（右）手势

Tip

为了支持更大范围的设备，我们建议你为那些不支持多点触控的设备，提供可替代的单指操作方式。不过，用户现在已经对多点触控手势充满期待了。所以，你可以适时地使用它。不过，电阻（resistive）触屏在一些较为低端的设备上不是很普及。

24.4.1 处理普通多点触控手势

在 API Level 8（Android 2.2）中引入的 ScaleGestureDetector 类可以用来探测双指缩放手势，即用户可以相向或相背地移动两根手指。当手指相背移动时，通常被认为放大；而手指相向移动时，则是缩小。这种“挤压缩放（pinch-to-zoom）”经常被用在地图或照片应用程序中。

Download at http://www.pin5i.com/

Tip

你可以使用 ScaleGestureDetector.SimpleOnScaleGestureListener 类来监听有 ScaleGestureDetector 探测的手势。

让我们再来看一个示例。我们仍然使用之前用过的 GameAreaView 自定义视图类，但这次我们将处理多点缩放事件。这样，GameAreaView 可以对缩放手势做出响应，并且对位图进行适当的改变，如在屏幕上进行缩放等。

Tip

这一部分中的许多代码示例都来自于 SimpleMultiTouchGesture 应用程序。在本书的网站上可以下载到它的源代码。

为了处理缩放手势，GameAreaView 类需要包含适当的手势探测器：ScaleGestureDetector。GameAreaView 类需要与缩放手势相关联，这与前面我们实现单点触控手势时是相同的，包括在类构造函数中初始化手势探测器、覆写 onTouchEvent()方法并向探测器传入 MotionEvent，以及覆写 onDraw()方法实现适当的视图绘制。我们还需要修改 GameAreaView 类，以便能够记录位图图像的尺寸（使用一个 Matrix），并且为图像放大和缩小提供一个 Helper 类。下面是我们对新 GameAreaView 类的实现，它将支持缩放手势：

```
public class GameAreaView extends View {
    private ScaleGestureDetector multiGestures;
    private Matrix scale;
    private Bitmap droid;

    public GameAreaView(Context context, int iGraphicResourceId) {
        super(context);
        scale = new Matrix();
        GestureListener listener = new GestureListener(this);
        multiGestures = new ScaleGestureDetector(context, listener);
        droid = BitmapFactory.decodeResource(getResources(),
            iGraphicResourceId);
    }

    public void onScale(float factor) {
        scale.preScale(factor, factor);
        invalidate();
    }

    @Override
    protected void onDraw(Canvas canvas) {
        Matrix transform = new Matrix(scale);
        float width = droid.getWidth() / 2;
        float height = droid.getHeight() / 2;
        transform.postTranslate(-width, -height);
        transform.postConcat(scale);
        transform.postTranslate(width, height);
        canvas.drawBitmap(droid, transform, null);
    }

    @Override
    public boolean onTouchEvent(MotionEvent event) {
        boolean retVal = false;
        retVal = multiGestures.onTouchEvent(event);
```

Download at http://www.pin5i.com/

```
            return retVal;
        }
    }
```

正如你所看到的，GameAreaView 类使用一个名为 scale 的 Matrix 变量随时跟踪并记录位图的当前尺寸。onTouchEvent() 方法用来捕获运动时间，并将其传递给 ScaleGestureDetector 手势探测器。与之前一样，GameAreaView 的 onScale() Helper 方法将缩放应用于位图图像，并且调用 invalidate() 方法强制屏幕重绘。

现在，我们来看看 GestureListener 类的实现，它将用于翻译缩放手势，并且调用我们刚刚实现的 GameAreaView 的方法。下面是 GestureListener 类的实现：

```
    private class GestureListener implements
        ScaleGestureDetector.OnScaleGestureListener {

        GameAreaView view;

        public GestureListener(GameAreaView view) {
            this.view = view;
        }

        @Override
        public boolean onScale(ScaleGestureDetector detector) {
            float scale = detector.getScaleFactor();
            view.onScale(scale);
            return true;
        }

        @Override
        public boolean onScaleBegin(ScaleGestureDetector detector) {
            return true;
        }

        @Override
        public void onScaleEnd(ScaleGestureDetector detector) {
        }
    }
```

记住，对于想探测的任何手势或动作，你都需要返回 true。所以，在缩放手势开始时所触发的 onScaleBegin() 方法需要返回 true。GestureListener 类方法的大部分实现都包含了对缩放手势数据的翻译。具体而言，我们使用缩放因子（由 getScaleFactor() 方法提供）来计算我们应该缩小还是放大位图图像，以及缩放的程度。我们将这一信息传递给我们刚刚实现的 GameAreaView 类的 onScale() Helper 方法。

现在，如果你在应用程序中使用 GameAreaView 类，那么缩放手势将类似图 24.6 所示。

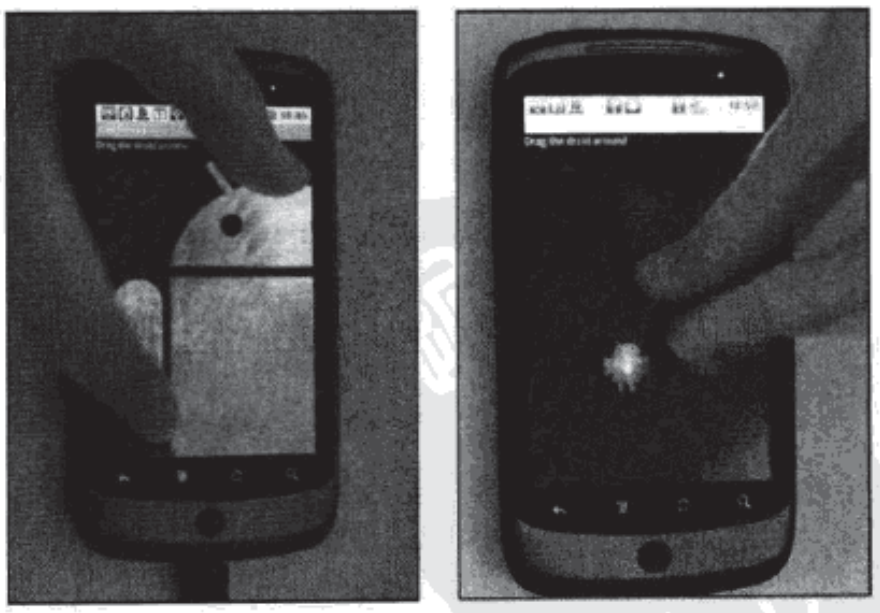


图 24.6 放大（左）和缩小（右）手势

Note

Android 模拟器目前并不支持多点触控输入。所以你需要在运行 Android 2.2 或更高版本的设备上运行和测试具有多点触控（如缩放操作）支持的应用程序。

Download at http://www.pin5i.com/

24.4.2 让手势更自然

手势可以以更加新潮、有趣、直观的方式增强 Android 应用程序的用户体验。将屏幕上需要执行的操作与用户手指的动作直接关联起来能够让手势自然而直观。让应用程序的操作看起来非常自然需要开发人员具有相关的经验。不过你需要记住的是不同设备的处理性能会不同，这或许是影响手势操作自然性的一个重要因素。

24.5 使用轨迹球

某些 Android 设备拥有硬件轨迹球（trackball），但不是全部。开发人员可以在 View 控件内处理轨迹球事件，这与处理点击或手势事件是相似的。要处理轨迹球事件，你可以使用 View 类的一个名为 onTrackballEvent() 的方法。它与手势处理的方法类似，拥有一个参数：MotionEvent 对象。你可以使用 MotionEvent 类的 getX() 和 getY() 方法来确定轨迹球的相对移动。诸如 Droid Incredible 设备上使用的光学触控板（optical track-pad）也可以以相同的方式进行处理。

Tip

如果你的应用程序要求设备具有轨迹球，那么你应该在应用程序的 Android Manifest 文件中设置<uses-configuration>标记，并将其指定为轨迹球。

24.6 处理屏幕方向改变

市场上的许多 Android 设备都拥有横屏（风景）和竖屏（肖像）两种模式，并且可以流畅地在这两种屏幕方向间切换。Android 操作系统将为你的应用程序自动地处理屏幕方向的改变，如果你选择这么做的话。你还可以提供替换资源，比如不同的布局，以便能够在肖像和风景两种模式下使用（第 25 章“面向不同设备配置和语言”中将讨论更多的内容）。另外，你还可以直接访问诸如加速度仪这样的设备传感器，我们已经在第 19 章“使用 Android 可选硬件 API”中讨论了有关设备的三个旋转方向的问题。

如果你想在程序中监听简单的屏幕方向改变，并让应用程序对此做出响应的话，可以在 Activity 中使用 OrientationEventListener 类。

Tip

这一部分中的许多代码示例都来自于 SimpleOrientation 应用程序。在本书的网站上可以下载它的源代码。方向改变最好能够在设备上进行测试，而非使用模拟器。

在你的 Activity 实现方向改变事件的处理非常简单。只需要初始化一个 OrientationEventListener 并且提供它的实现即可。例如，下面名为 SimpleOrientationActivity 的 Activity 类记录了方向信息，并将其输出到 LogCat：

```
public class SimpleOrientationActivity extends Activity {

    OrientationEventListener mOrientationListener;
```

Download at http://www.pin5i.com/

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mOrientationListener = new OrientationEventListener(this,
        SensorManager.SENSOR_DELAY_NORMAL) {

        @Override
        public void onOrientationChanged(int orientation) {
            Log.v(DEBUG_TAG,
                "Orientation changed to " + orientation);
        }
    };

    if (mOrientationListener.canDetectOrientation() == true) {
        Log.v(DEBUG_TAG, "Can detect orientation");
        mOrientationListener.enable();
    } else {
        Log.v(DEBUG_TAG, "Cannot detect orientation");
        mOrientationListener.disable();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        mOrientationListener.disable();
    }
}
```

你可以将检测屏幕方向改变的频率设定为各种不同的值。对于游戏和其他应用程序可能需要设置不同的值。默认的频率 SENSOR_DELAY_NORMAL 比较适合用于简单的屏幕方向改变。而其他值，如 SENSOR_DELAY_UI 和 SENSOR_DELAY_GAME 等，则可能适用于某些特殊的应用程序。

在你拥有了一个有效的 OrientationEventListener 对象后，可以使用 canDetectOrientation() 方法检测其是否能够探测方向的改变，还可以使用 enable() 和 disable() 方法开启和关闭监听器。

OrientationEventListener 拥有一个回调方法用于监听方向的改变：onOrientationChanged()。它具有一个参数：一个 integer，通常表示设备当前的倾斜角，范围从 0 至 359。

① 结果为 ORIENTATION_UNKNOWN（-1）意味着设备平放着（可能放置于桌面）并且旋转角未知。

② 结果为 0 表示设备处于“正常”方向，即设备顶端朝上。（何为“正常”是由制造商决定的。你需要在实际的设备上进行测试，以确定它的含义。）

③ 结果为 90 表示设备的倾斜角为 90°，即设备左侧朝上。

④ 结果为 180 表示设备的倾斜角为 180°，即设备底部朝上（倒置）。

⑤ 结果为 270 表示设备的倾斜角为 270°，即设备右侧朝上。

图 24.7 显示了设备倾角为 90° 的状态。

Download at http://www.pin5i.com/

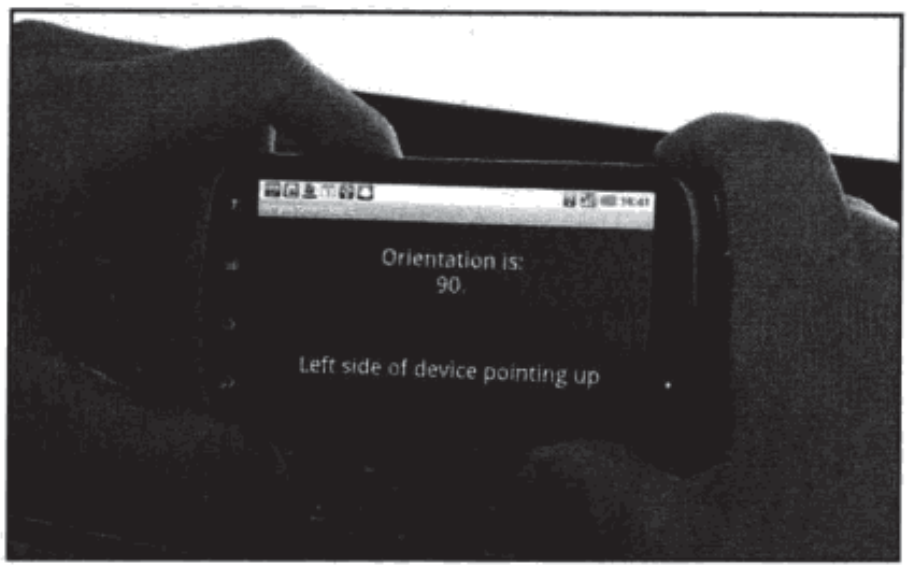


图 24.7 OrientationEventListener 报告的设备旋转角

Warning

早期版本的 Android SDK 包含一个名为 OrientationListener 的类，许多开发人员用它来处理屏幕方向的改变。不过现在这个类已经废弃不用了。

小 结

Android 平台为用户数据输入提供了许多强大而灵活的方式。视图控件内置了许多强大的输入法，这对开发人员将大有裨益。应用程序可以充分地利用这些内置的输入法，比如软键盘，甚至还可以自己定义，以达到特殊的效果。Android 框架还包含诸如手势这样的强大特性，以及许多辅助功能，包括语音识别和语音合成支持。在应用程序中支持各种不同的输入方法是非常重要的，因为不同的用户可能会有不同的习惯，而且不是所有的设备都支持所有的输入方法。

参考资料和更多信息

Android Reference（Android 参考）：Faster Orientation Changes.http://j.mp/9P3yTy.

Android Reference： Screen Orientation and Direction.http://j.mp/b2zY1t.

Download at http://www.pin5i.com/

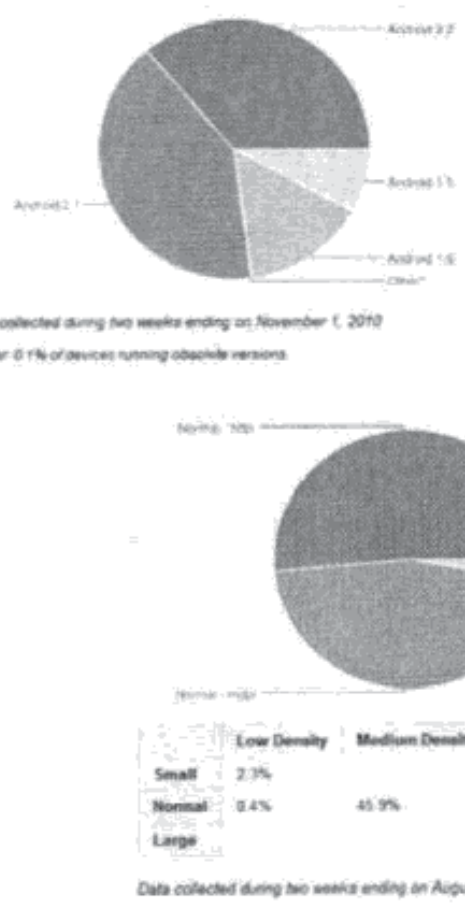
第25章

面向不同设备配置和语言

全球范围内的市场上有超过60种不同的Android设备。在本章中，你将学习如何设计并开发对不同屏幕尺寸、硬件或平台版本设备相兼容的Android应用程序。我们还给出了一些有助于设计和开发兼容不同设备应用程序的技巧和诀窍。最后，你将学习如何使你的应用程序国际化，以便能够在国外销售和使用。

25.1 最大化应用程序兼容性

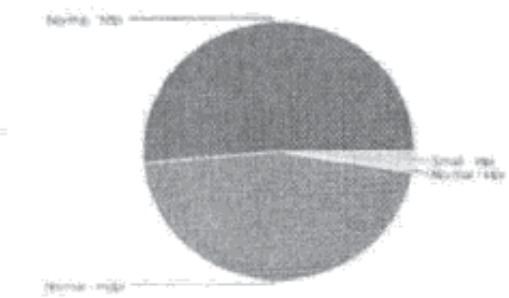
现在有越来越多的制造商在研发Android设备，各种不同的型号极大地丰富了市场，每一种设备都有它自己的独特之处，同时占有一定的市场份额。用户有了很大的选择空间，但这也带来了一些问题。设备的迅速增加导致了开发人员常说的“非一致性（fragmentation）”，也有人称之为“兼容性问题（compatibility issues）”。不管采用何种术语，开发具有对设备广泛兼容性的Android应用程序成为了一种挑战。开发人员需要与各种不同的版本“作战”，还要考虑设备是否具有一些可选硬件（如摄像头、键盘），以及各种不同的屏幕尺寸和分辨率（如图25.1所示）。设备的市场份额表相当冗长，并且在每发布一款新设备时都会随之增加。



Platform	API Level	Distribution
Android 1.5	3	7.9%
Android 1.6	4	15.0%
Android 2.1	7	40.8%
Android 2.2	8	36.2%

Data collected during two weeks ending on November 1, 2010

** Other: 0.1% of devices running obsolete versions*



	Low Density	Medium Density	High Density
Small	2.3%		
Normal	0.4%	45.9%	51.2%
Large			

Data collected during two weeks ending on August 2, 2010

图25.1 有关平台版本和屏幕像素密度的Android设备统计（来源：http://j.mp/bnW70V）

Download at http://www.pin5i.com/

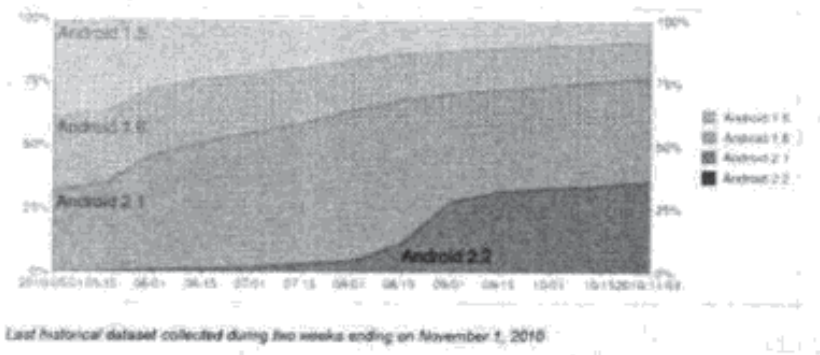


图 25.1 有关平台版本和屏幕像素密度的 Android 设备统计（来源：http://j.mp/bnW70V）（续）

虽然“非一致性”问题让 Android 应用程序开发人员的生活变得更为复杂，但开发能够对相当一部分设备（甚至所有设备）兼容的应用程序还是有可能的。当开始考虑最大化兼容性这一问题时，下面的一些策略和建议也许对你有用。

① 在可能的情况下，始终面向大多数设备进行开发。

② 如果某个开发决策局限了应用程序的兼容性（例如，使用了 API Level 较高的 API，或者引入了硬件需求，如摄像头支持），那么首先进行风险评估，然后将这一限制写入文档。你还需要确定是否为一些不支持的设备提供替代方案。

③ 在设计应用程序用户界面时，将屏幕尺寸和分辨率纳入考量范围。设计出具有很强的灵活性，能够适应横与竖屏两种模式、不同屏幕分辨率和尺寸的布局通常是有可能的。如果你不对此早做打算的话，后期再做修改是一件非常痛苦的事情。

④ 尽早地在尽可能多的设备上进行测试，避免后期带来一些痛苦的“惊喜”。确保设备拥有不同的硬件、软件，包括不同版本的 Android 平台，以及不同的屏幕尺寸和硬件特性。

⑤ 在需要时，为硬件之间的差异性提供替代资源（我们将在本章中详细讲解有关替代资源的话题）。

⑥ 如果你在应用程序中引入了软件和硬件需求，那么确保在 Android Manifest 文件中使用相应的标记进行了注册。Android 平台以及某些第三方机构，如 Android Market，将使用这些标记来确保应用程序安装在了满足需求的设备上。

现在，我们来看看面向不同设备配置和语言进行开发的策略。

25.2 为兼容性设计用户界面

在我们讲解有关提供自定义应用程序资源和代码，以支持特定硬件配置的方法之前，你首先需要了解的是，通常这些问题是可以在第一时间解决的。方法就是在最初进行设计时，默认的解决方案要足够灵活，以应对各种变化。当涉及用户界面时，务必让它们保持简单，千万不能挤在一起。另外，你还可以使用以下许多任由你支配的强大的工具。

① 从经验上讲，针对中等尺寸和大尺寸屏幕，且中等分辨率进行设计。随着技术的发展，设备都会倾向于拥有更大的屏幕和更高的分辨率。

② 对于 View 和 Layout 控件的宽、高属性，使用 fill_parent（现在为 match_parent）和 wrap_content，这样，控件可以在不同的屏幕尺寸和方向下自动进行拉伸。

③ 对于度量单位，可以使用灵活的单位，如 dp 和 sp，而尽量避免使用固定的单位类型，如 px、mm 和 in。

④ 避免使用 AbsoluteLayout 和其他精确像素的设置和属性。

Download at http://www.pin5i.com/

⑤ 使用灵活性强的布局控件（如 RelativeLayout、LinearLayout、TableLayout 和 FrameLayout）来设计屏幕，以便能够在不同的屏幕方向、尺寸和分辨率下完美地显示。尝试使用“工作方块（Working Square）”原理来组织屏幕内容，我们稍后将讨论这一话题。

⑥ 将屏幕内容包装在可延伸的容器控件中，如 ScrollView 和 ListView 等。通常，屏幕的延伸只涉及一个方向（水平或垂直）。

⑦ 不要为屏幕元素、尺寸和度量单位设定绝对值。相反，你可以使用相对位置、重量（weight）和重力（gravity）等。预先在这些问题上花点时间可以为后期节约很多时间。

⑧ 为应用程序提供一定质量的图像，并且始终保留原始尺寸和质量（通常较大）的图像，以便今后需要在兼容不同分辨率的应用程序中使用。图像质量和尺寸之间始终是一种平衡。找到一个平衡点，既使图像质量在可接受的范围内，又避免因尺寸过大而对应用程序造成性能和响应性上的影响。可能的话，使用可拉伸图像，如“九格拉伸图像（Nine-Patch）”，它能够允许图像根据显示区域的大小改变其尺寸。

Tip

有关设备屏幕的信息可以参考 DisplayMetrics 功能类，当它与窗口管理器结合时，可以在运行时确定设备的各种显示特征：

```
DisplayMetrics currentMetrics = new DisplayMetrics();
WindowManager wm = getWindowManager();
wm.getDefaultDisplay().getMetrics(currentMetrics);
```

25.2.1 支持特定屏幕类型

虽然通常而言，你都想开发对屏幕无依赖（也就是说兼容各种类型、不同尺寸、不同像素密度的屏幕）的应用程序，但是，你也可以使用 Android Manifest 文件中的<supports-screens>标记来显式地制定应用程序所支持的屏幕类型。默认情况下，应用程序支持所有类型的屏幕。如果你的应用程序与视频相关，或者是一个电子书阅读器——它们不适合小屏幕的设备，那么你可以考虑为应用程序配置这一选项。更多有关 Android Manifest 文件标记的信息，可以参考 Android SDK 文档：http://developer.android.com/guide/topics/manifest/supports-screens-element.html。

Tip

Android 操作系统结合了缩放技术（scaling techniques）以实现在不同屏幕尺寸、分辨率和高宽比例下的平滑显示。有关这一内容的更多细节，可以参考本章末尾“参考资料和更多信息”中有关支持多屏幕的文章。

25.2.2 使用九格拉伸图像

手机屏幕拥有各种不同的尺寸。想象一个可以根据不同的屏幕尺寸和方向或者不同长度的文字调整大小的图像，它将非常方便而适用。这将极大地节约开发人员为各种不同屏幕尺寸创建图像的时间。Android 支持九格拉伸（Nine-Patch Stretchable）图像以达到这一目的。九格拉伸图像是拥有小块图像区域的简单 PNG 图像，这些图像区域被定义为可以进行适当的缩放，而非把整个图像作为一个整体进行缩放。通常中间区域为透明。图 25.2 演示了图像（显示为方块）是如何被分成九“格”的。

九格拉伸图像可以通过 Android SDK Tools 目录下的 draw9patch 工具由一个 PNG 文

Download at http://www.pin5i.com/

件创建。draw9patch 工具的界面（如图 25.3 所示）相当直观。在左侧的面板中，你可以为你的图像指定“九格”导引，以定义拉伸时的缩放方式。在右侧的面板中，你可以预览根据你的定义图像将如何进行拉伸缩放。

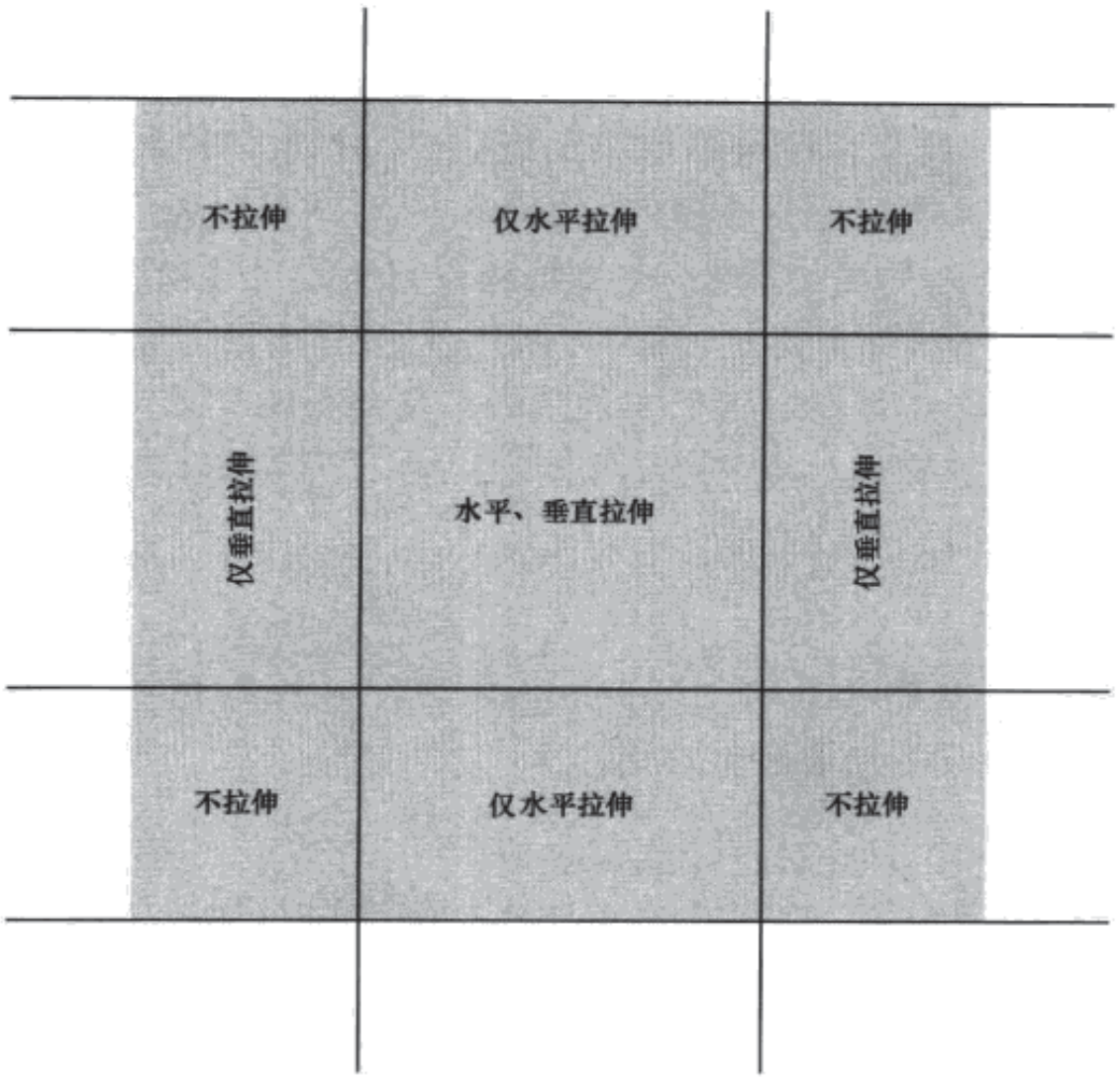


图 25.2 由一个方块定义的九格图像是如何进行缩放的

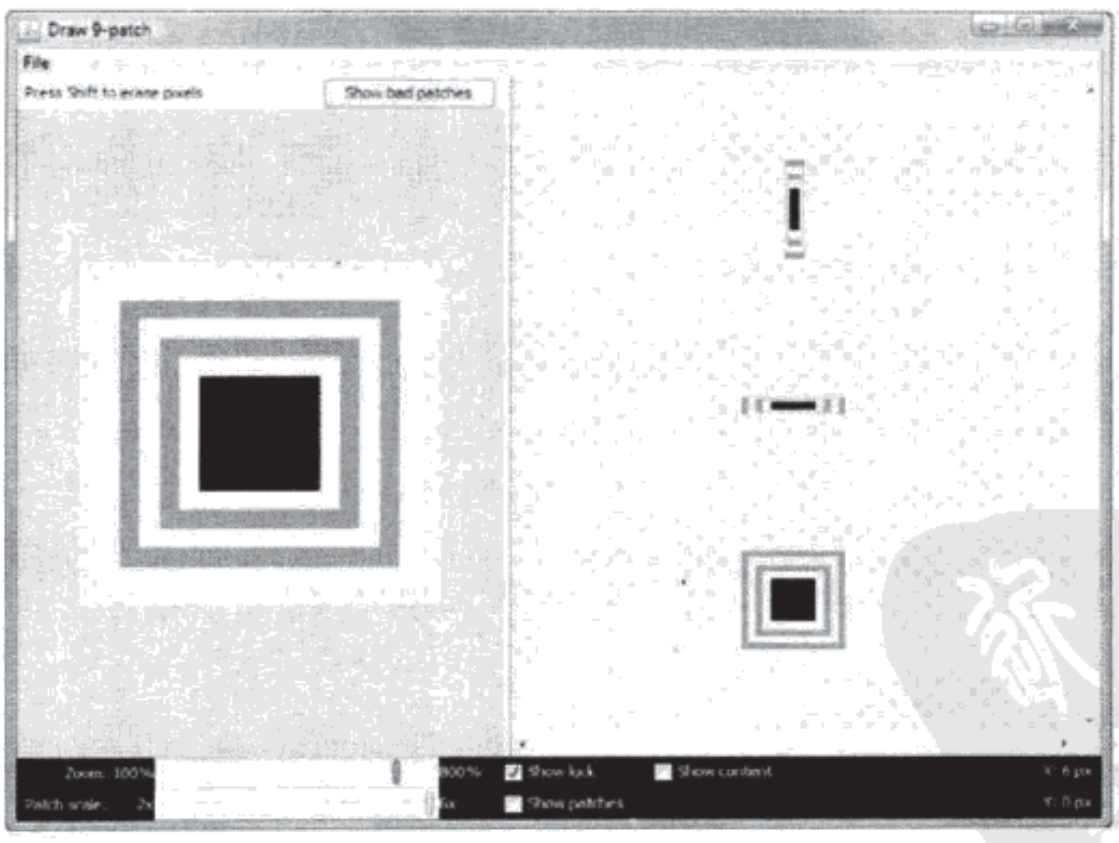


图 25.3 在进行“九格”处理前的简单 PNG 文件

要使用 draw9patch 工具创建九格图像，可以进行以下操作。

① 运行 Android SDK tools 目录下的 draw9patch.bat 文件。

② 将一个 PNG 文件拖入左侧的面板中（或者执行 File→Open Nine-Patch）。

③ 单击左侧面板底部的 Show Patches 复选框。

Download at http://www.pin5i.com/

④ 将 Patch Scale 设置为合适的值（比能够看见标记结果稍大）。

⑤ 沿着图像的右边沿单击，以设置水平“格”导引。

⑥ 沿着图像的上边沿单击，以设置垂直“格”导引。

⑦ 在右侧面板中查看结果；移动“格”导引直到图像按照预期的效果进行拉伸，如图 25.4 和图 25.5 所示。

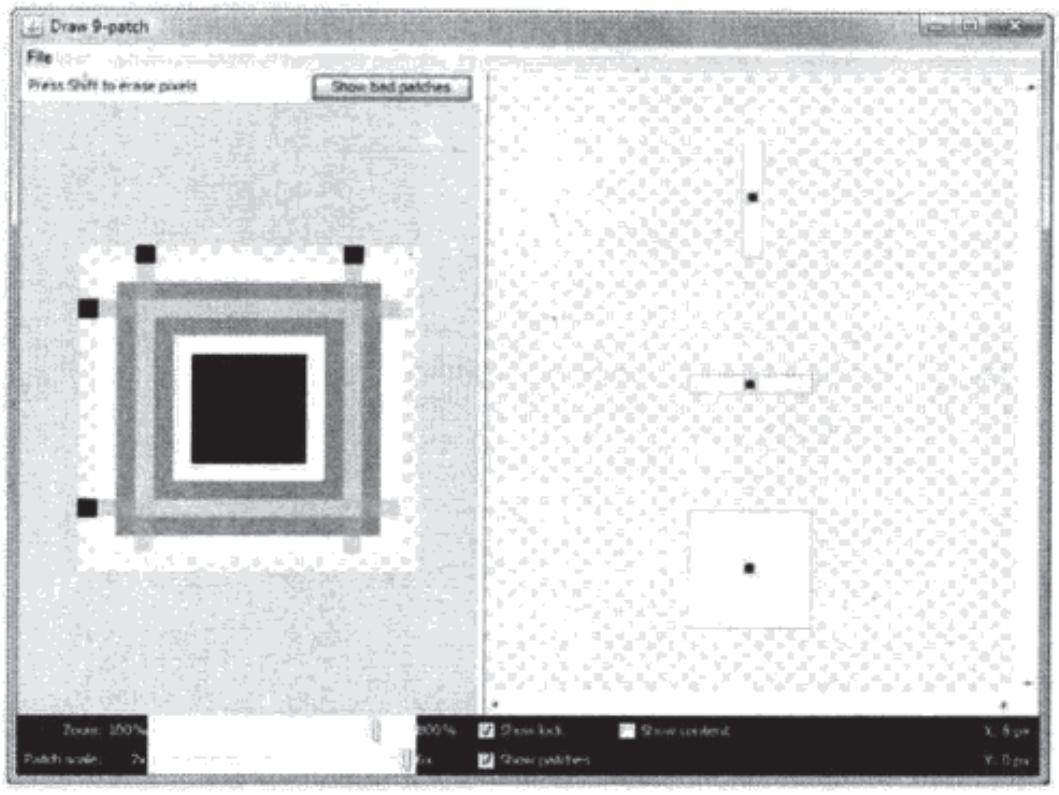


图 25.4 使用定义的导引进行了“九格”处理后的“九格”PNG 文件

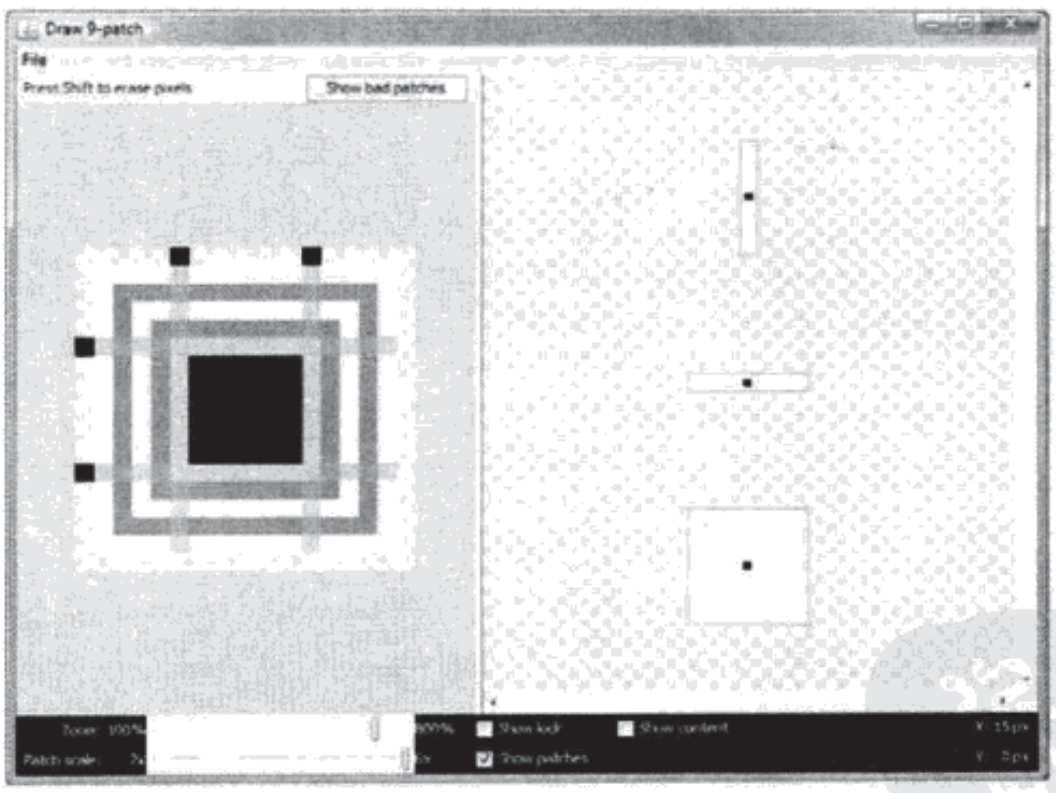


图 25.5 使用定义的不同导引进行了“九格”处理后的“九格”PNG 文件

⑧ 要删除一个“格”导引，按住 Shift 键在导引的像素（黑色）上单击即可。

⑨ 以 .9.png 为扩展名保存图像（例如：little_black_box.9.png）。

25.2.3 使用 Working Square 原理

另一种设计不同屏幕方向显示的方法是在应用程序的 Activity 内使用“工作方块（working square）”。当屏幕方向改变时，这一部分保持不变（或者仅仅只是简单的旋转）。只

Download at http://www.pin5i.com/

有位于“工作方块”以外的部分才会随着屏幕方向的改变而发生本质的变化，如图 25.6 所示。

“工作方块”的一个典型例子是 HTC Evo 4G 手机上的 Camera（照相机）应用程序。在肖像模式下，摄像头控件位于取景框的下面，如图 25.7（左）所示；当顺时针旋转成风景模式后，摄像头控件仍处于相同位置，但是位于取景框的右侧，如图 25.7（右）所示。那么，取景框所在的区域则可以认为是工作方块——始终保持不变的区域。其他的控件和滑动抽屉则在这个区域之外，这样用户可以随时进行拍照或摄像。

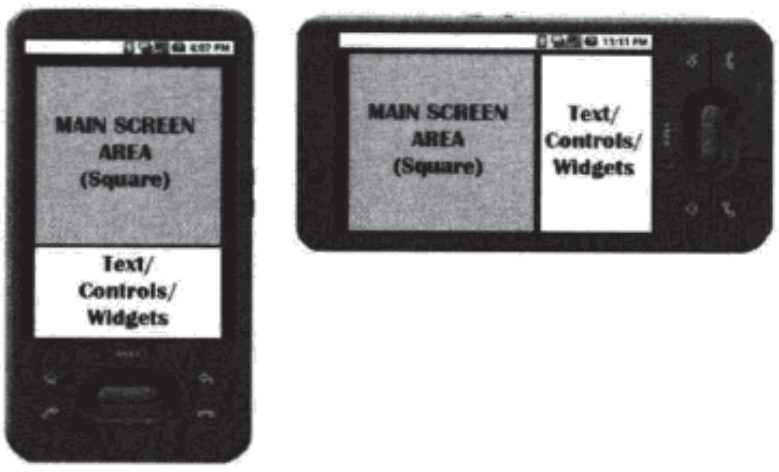


图 25.6 “工作方块”原理

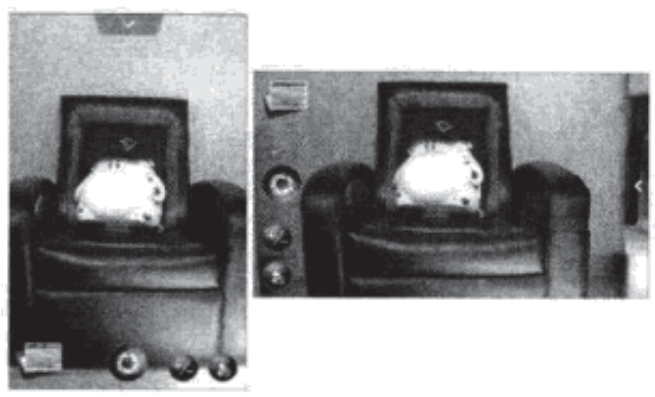


图 25.7 Evo 4G Camera 应用程序，它使用了“工作方块”原理

当你使用这个应用程序时，旋转屏幕后几乎没有什么变化。只是控件从取景框的底部移到了它的右侧。就是这样，屏幕主体部分的内容几乎没有变化，这是对“工作方块”原理的很好的应用。

25.3 提供应用程序替代资源

Android 应用程序每一次进行屏幕绘制时，Android 操作系统都会尝试使用最佳的资源。在许多情况下，应用程序只提供一组资源——默认资源。开发人员可以在应用程序包内添加相同资源的可替代版本。Android 操作系统始终会尝试载入定义最详细而具体的可用资源，开发人员无需关心到底载入哪个资源，因为操作系统会完成这一任务。

25.3.1 使用替代资源标识符

替代资源可以按照许多标准进行创建，包括但不局限于屏幕特性、设备输入方法，以及语言或区域设置。这些替代资源被层级化地组织在工程资源目录/res 下。你可以使用目录修饰符（以目录名后缀的形式）将某个资源指定为替代资源，从而只在某些特殊的情况下使用。

为了理解这一概念，我们举一个简单的例子。使用替代资源最多的场合是有关 Eclipse 自动创建的 Android 工程中所包含的应用程序图标。应用程序可以仅仅提供一个图标资源，将其存储在/res/drawable 目录下。但是，不同的 Android 设备拥有不同的屏幕分辨率。所以，就需要使用替代资源：/res/drawable-hdpi/icon.png 适用于高像素密度的屏幕，/res/drawable-ldpi/icon.png 适用于低像素密度的屏幕，而/res/drawable-mdpi/icon.png 则适用于中等像素密度的屏幕。注意在每种情况下，替代资源本身的命名是相同的。这就是为什么 Android 系统可以载入适当的资源的原因——通过资源的名称。

有意思的是，在这个例子中，你是见不到默认资源（即存储在/res/drawable 目录下的 icon.png 文件）的，因为它只有在系统环境的屏幕分辨率既不匹配 hdpi 也不匹配 mdpi 或者 ldpi 的时候才会使用。你见不到默认资源的原因在于，根据定义，Android 设备的屏幕解析度只可能为这三者之一，所以不会出现“退而求其次”的情况。

Download at http://www.pin5i.com/

Note

我们也不喜欢这样的解释。但作为开发人员，出于安全考虑，我们需要为 switch 表达式实现一个 default 情况，替代资源也是如此。我们更倾向于稳妥的做法，尝试始终定义默认资源，这样无论设备的设置中有关声明，应用程序都有资源可供载入。我们想要避免一些理论上的情况，比如某些设备忘记了指定 dpi 设置，那么我们的应用程序就不能正确地载入资源。解决这一问题的办法是将最具普适性的资源存储在默认目录下，而仅在必要时使用替代资源。例如，你可以将 mdpi 版本的图标放到/res/drawable 文件夹下作为默认资源，然后只创建/res/drawable-ldpi 和/res/drawable-hdpi 目录存储替代资源。总之，如何组织你的应用程序资源完全由你决定（本章稍后部分将介绍更多有关内容）。

下面是有关替代资源的一些重要信息。

① 替代资源的目录修饰符始终附加在默认资源目录名的后面。例如，/res/drawable-qualifier、/res/values-qualifier、/res/layout-qualifier。

② 替代资源目录修饰符（及资源文件名）必须为小写，只有一种除外：区域修饰符。

③ 一个资源目录名中只能附加一种类型的一个修饰符。当然，这有一定的弊端，你可能不得不将同一个资源放在不同的目录下。例如，你不能创建一个名为/res/drawable-ldpi-mdpi 的资源目录以共享相同的图标。相反，你需要分别创建两个文件夹：/res/drawable-ldpi 和/res/drawable-mdpi。其实，如果你想要让不同的修饰符共享同一个资源的话，可以考虑将其视作默认资源，然后为不是 ldpi 和 mdpi（也就是 hdpi）的设备提供替代资源。就像我们前面说过的一样，如何组织应用程序的资源完全取决于你自己，这些都只是我们的建议。

④ 替代资源的目录修饰符可以组合起来，两两之间用短横线隔开。这使得开发人员可以创建特定的目录名称。不过，这些修饰符必须按照一定的次序排列，并且，Android 操作系统始终会载入最契合的资源（也就是具有最长目录名称的资源）。例如，你可以像下面这样创建针对法语（修饰符为 fr）、加拿大地区（修饰符为 rCA，区域修饰符是大写的）的字符串资源（存储在 values 目录下）：/res/values-fr-rCA/strings.xml。

⑤ 你只需要为特定的资源创建替代资源即可，而不是文件中的每一个资源。如果你只想翻译 strings.xml 文件中一半的字符串的话，那么可以只为它们提供替代字符串资源。换句话说，默认的 strings.xml 资源文件是其替代资源文件的“超集（superset）”，替代资源文件为默认资源文件的子集，只包含需要翻译的字符串。通常来讲，不需要翻译的字符串可能是公司名或品牌名。

⑥ 不允许使用自定义的目录名修饰符，只能够使用 Android SDK 中定义的修饰符。表 25.1 列出了大多数的修饰符。

表 25.1 重要的替代资源修饰符

目录修饰符	示 例 值	说 明
移动国家码和移动网编号	mcc310（美国） mcc310-mnc004（美国，Verizon） mcc208-mnc00（法国，Orange）	从设备上 SIM 卡获得的移动国家码（Mobile Country Code，MCC），后跟可选的短横线及移动网编号（Mobile Network Code，MNC）
语言和区域代码	en（英语） ja（日语）	语言代码（ISO 639-1 2 字母语言码），后跟可选的短横线及区域代码（一个小写的“r”后跟 ISO 3166-1-alpha-2 中定义的区域代码）

Download at http://www.pin5i.com/

续表

目录修饰符	示 例 值	说 明
语言和区域代码	de（德语） en-rUS（美式英语） en-rGB（英式英语）	
屏幕尺寸	Small normal large	泛化的屏幕尺寸值 small 通常指低解析度的 QVGA 或稍高一点的 VGA 屏幕 normal 通常指中等解析度的 HVGA 屏幕 large 通常指 HVGA 更高的屏幕 在 API Level 4 中被引入
屏幕高宽比	long notlong	设备是否是宽屏 WQVGA、WVGA 和 FWVGA 屏幕为 long QVGA、HVGA 和 VGA 为 notlong， 在 API Level 4 中被引入
屏幕方向	port land	肖像模式、风景模式
停驻（dock）模式	car desk	当设备处于车载或桌面停驻模式下载入特定的资源 在 API Level 8 中被引入
夜间模式	night notnight	当设备处于夜间模式或不处于夜间模式下载入特定的资源 在 API Level 8 中被引入
屏幕像素密度	ldpi mdpi hdpi nodpi	ldpi 表示低屏幕解析度（大约 120dpi） mdpi 表示中等屏幕解析度（大约 160dpi） hdpi 表示高屏幕解析度（大约 240dpi） nodpi 选项意味着你不想让资源适应屏幕解析度 在 API Level 4 中被引入
触屏类型	notouch stylus finger	notouch 供不具备触屏的设备使用 stylus 用于带有触笔触屏（电阻屏）的设备 finger 用于带有手指触屏（电容屏）的设备
键盘类型及可用性	keysexposed keyshidden keyssoft	keysexposed 选项针对键盘（硬件键盘或软键盘）可用时的资源 keyshidden 选项针对键盘（硬件键盘或软键盘）不可用时的资源 keyssoft 选项针对软键盘可用时的资源
文本输入法	nokeys qwerty 12key	nokeys 选项用于设备不具有硬件键盘时的资源 qwerty 选项用于设备具有 QWERTY 键盘时的资源 12key 选项用于设备具有 12 键数字键盘时的资源
导航键可用性	navexposed navhidden	navexposed 用于硬件导航按钮可用时的资源 navhidden 用于硬件导航按钮不可用时的资源
导航方式	nonav dpad trackball wheel	在设备不具有导航按钮时使用 nonav 当主导航方式是导航面板时使用 dpad 当主导航方式是轨迹球时使用 trackball 当主导航方式是方向滚轮时使用 wheel

Download at http://www.pin5i.com/

续表

目录修饰符	示 例 值	说 明
Android 平台	v3（Android 1.5） v4（Android 1.6） v5（Android 2.0） v7（Android 2.1） v8（Android 2.2） v9（姜饼）	根据 Android 平台的版本载入资源 注意：这一修饰符可能存在一些问题，具体情况可以参考 Android 文档

⑦ 始终尝试包含默认资源，即那些存储在不带任何修饰符的目录下的资源。当 Android 操作系统不能找到匹配的替代资源时的后备选择。如果你没有提供默认资源，那么系统将依照资源修饰符选择最接近匹配的资源（也可能起不到作用）。

既然你已经理解了替代资源的工作方式，让我们来看看你可以使用的不同目录修饰符。修饰符以严格的顺序附加在现有的资源目录名后，表 25.1 给出了它们的排序。

下面是带有修饰符的替代资源目录的正确例子：

```
/res/values-en-rUS-port-finger
/res/drawables-en-rUS-land-mdpi
/res/values-en-qwerty
```

下面是错误的例子：

```
/res/values-en-rUS-rGB
/res/values-en-rUS-port-FINGER-wheel
/res/values-en-rUS-port-finger-custom
/res/drawables-rUS-en
```

第一个错例中使用了同种类型的两个不同的修饰符：rUS 和 rGB。第二个的错误在于，除了区域修饰符外，其他所有修饰符都应该是小写的。第三个错例包含了一个由开发人员自定义的属性，目前并不支持这种情况。而最后一个则没有按照修饰符的顺序进行排列：首先应该是语言，然后才是区域，等等。

25.3.2 提供用于不同屏幕方向的资源

让我们来看一个简单的示例，它通过替代资源，针对不同的屏幕方向提供了自定义的屏幕内容。

Tip

这一部分中的许多代码示例都来自于 SimpleAltResources 应用程序。在本书的网站上可以下载该应用程序的源代码。

这个示例应用程序并没有需要在这里解释的代码，不信的话可以看看 Activity 类的代码。在该示例中，所有的功能都依赖于资源目录修饰符。这些资源包括以下内容。

① 应用程序的默认资源，包括应用程序图标，存储在/res/drawable 目录下的图片，存储在/res/layout 目录下的布局文件，以及存储在/res/values 目录下的颜色和字符串资源。当没有定义更具体的资源时，它们将被载入，也就是说，这些默认资源是后备的。

② 在/res/drawable-port 目录下存储了一幅肖像（竖屏）模式下的图片，同时还提供了肖像屏幕模式下的字符串和颜色资源，它们存储在/res/values-port 目录下。如果设备处于竖屏方向，那么这些资源（肖像模式下的图片、字符串和颜色）将被载入并由默认布局使用。

Download at http://www.pin5i.com/

③ /res/drawable-land 目录下还存储了风景（横屏）模式下的替代图片，以及该模式下所使用的字符串和颜色（前景和背景反色）资源，它们存储在/res/values-land 目录中。如果设备处于竖屏方向，那么这些资源（风景模式下的图片、字符串和颜色）将被载入并由默认布局使用。

这样，应用程序会根据运行时设备的方向载入不同的资源，如图 25.8 所示。图中显示了工程的布局资源，以及不同设备方向下的屏幕显示。

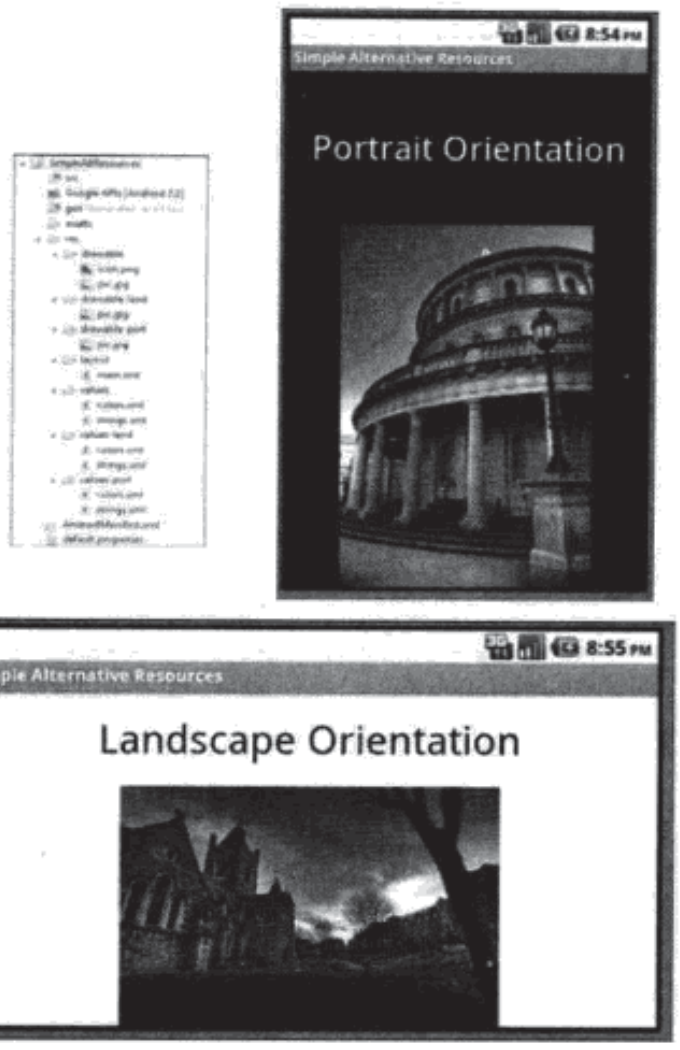


图 25.8 在肖像和风景模式下使用替代资源

25.3.3 在程序中使用替代资源

目前，我们不能在程序中请求具有特定配置的资源。例如，开发人员不能在程序中请求某个字符串资源的法语或英语版本。相反，Android 系统会在运行时决定使用何种资源，而开发人员只需要使用普适的资源名称即可。

25.3.4 高效地组织应用程序资源

对替代资源的过度使用是很容易出现的。你可以为设备屏幕、语言或者输入法的各种组合提供自定义的图像资源。但是，每次当你向工程中添加应用程序资源时，其尺寸都会增大。

另外，过度地使用替代资源也会造成性能上的降低——一般发生在运行时每次进行资源切换时。每次在某个运行时事件（如方向改变、键盘状态改变）发生时，Android 操作系统都需要重启底层的 Activity，并且重新载入资源。如果你的应用程序载入了过多的资源和内容，那么应用程序的性能和响应性将会为此而付出代价。

你需要仔细地选择资源的组织模式。通常来讲，你应该将最常使用的资源作为默认资源，然后仅在必要时小心地使用替代资源覆盖它。例如，如果你正在开发一个应用程序，它会经常性地显示视频或者游戏画面，那么你可以将风景模式设为默认，并且提供肖像模式下的替

Download at http://www.pin5i.com/

代资源，因为它并不常使用。

（1）在配置变化时保持数据

Activity 可以使用 onRetainNonConfigurationInstance() 方法在模式切换时保存数据，并且在切换后使用 getLastNonConfigurationInstance() 方法恢复数据。当 Activity 有许多设置需要完成，或者有许多预载入的事务需要处理时，这一功能将非常有用。

（2）处理配置改变

在模式切换发生，并且不需要 Activity 重新载入替代资源的情况下，你可以考虑让 Activity 类来处理模式的切换，从而避免 Activity 的重启。我们之前提及的摄像头应用程序就使用了这一技术，它能够处理屏幕方向的改变，并且能够避免重新初始化摄像头，重新显示取景窗，以及重新显示摄像头控件（Button 按钮只需要在方向改变时稍做旋转——非常流畅）。

要让 Activity 类处理配置的改变，你的应用程序需要做如下修改。

① 修改 Android Manifest 文件中特定 Activity 类的<activity>标记，为其添加 android:configChanges 属性。其中必须指定 Activity 类处理的特定事件类型。

② 实现 Activity 类的 onConfigurationChanged() 方法，以处理特定的改变（通过类型）。

25.4 应用程序国际化

Android 的用户来自世界各地，他们讲不同的语言，使用不同的货币，日期的格式也不尽相同，这仅仅是举几个例子。Android 应用程序的国际化通常涉及三个方面。

① 提供不同语言下的替代资源，如字符串或者图像。

② 实现区域无关或者针对特定区域的代码，以及其他与程序相关的内容。

③ 为在国外市场销售应用程序而进行配置，我们将在第 29 章“销售你的 Android 应用程序”中讨论更多有关这方面的话题。

我们已经讨论过如何使用替代资源了，但可能还需要另外一个实例，让我们来看看一个能够根据设备语言设置而载入不同字符串资源的例子。具体来讲，我们来关注一个示例应用程序，它能够依照设备语言和区域的设定，载入不同的字符串和图像资源。

25.4.1 使用替代资源实现国际化

让我们来看两个有关如何使用替代资源的例子，这次，我们使用不同的语言和区域设置本地化一个应用程序。“语言”是指不同国家和地区所使用的不同语言，如英语、法语、西班牙语、德语、日语等。“区域”则更为具体，如英语（美国）、英语（英国）或者英语（澳大利亚）等。有些时候你可以存着侥幸心理只提供一种语言资源，但也有些时候这样行不通。

我们的第一个例子是理论上的。如果你想让你的应用程序既能支持英语又能支持法语（除去默认字符串），那么你可以简单地创建两个资源目录：/res/values-en（针对英语版本的 strings.xml 文件）和/res/values-fr（针对法语版本的 strings.xml 文件）。两个 strings.xml 文件内的资源名称是相同的。例如，/res/values-en/strings.xml 文件可能类似于下面这样：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
```

Download at http://www.pin5i.com/

```
    <string name="hello">Hello in English!</string>
</resources>
```

而/res/values-fr/strings.xml 文件则可能是这样的：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Bonjour en Français!</string>
</resources>
```

定义在/res/layout 目录下的默认布局文件使用变量名@string/hello 引用上面的字符串，并不指定其所使用的语言或者是资源具体存放的目录。Android 操作系统会在运行时决定载入哪个版本的字符串（法语、英语或者是默认）。下面是一个带有 TextView 控件并在其中显示字符串的布局：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:text="@string/hello" >
</LinearLayout>
```

在程序中可以下面的方式访问这个字符串：

```
String str = getString(R.string.hello);
```

就是这么简单。下面，我们来看一个稍微复杂点的例子，它将向你展示如何组织应用程序替代资源，以便能够提供基于设备语言和区域设置的功能。

Tip

本章中的许多代码示例都来自于 SimpleInternationalization 应用程序。在本书的网站上可以下载该应用程序的源代码。

与上面一样，这个应用程序也不包含任何有意义的代码。相反，所有的功能都依赖于资源目录修饰符，通过它们指定了需要载入的具体资源。以下这些资源包括。

① 应用程序的默认资源，包括存储在/res/drawable 目录下的应用程序图标，存储在/res/layout 目录下的布局文件，以及存储在/res/values 目录下的 strings.xml 字符串资源文件。当没有定义更具体的资源时，它们将被载入，也就是说，这些默认资源是后备的。

② 在/res/values-en 目录下存储了英语的字符串资源。如果设备的语言设置为了英语，那么将在默认布局中使用其中定义的字符串。

③ 在/res/values-fr-rCA 目录下存储了法语（加拿大地区）的字符串资源。如果设备的语言设置为了法语（加拿大），那么将在默认布局中使用其中定义的字符串。另外，还有一个替代布局存储在了/res/layout-fr-rCA 目录下，它将绘制存储在/res/drawable-fr-rCA 目录下的一幅图片（魁北克旗）。

④ 最后，/res/values-fr 目录下存储了法语的字符串资源。如果设备的语言设置为了法语（除加拿大以外的任何地方），那么将在默认布局中使用其中定义的字符串。

Download at http://www.pin5i.com/

这样，应用程序将会在不同的语言和区域设置下载入不同的资源，如图 25.9 所示。图中显示了工程的布局资源，以及不同语言和区域设置下的屏幕显示。

图 25.9 使用不同语言的字符串资源

通常而言，设备默认会移植一种语言，比如，在美国，语言为英语（美国）。而在法国购买的设备则会是法语（法国），在英国及其其他领土则可能会是英语（英国），也就是说，澳大利亚和新西兰拥有自己的英语语言设置，加拿大同时具有英语（加拿大）和法语（加拿大）两个语言选项。

要在模拟器或设备上修改区域设置，你可以执行以下步骤。

① 导航至首页。

② 按下 Menu（菜单）按钮。

③ 选择 Settings（设置）选项。

④ 向下滑动并选择 Language & keyboard（语言和键盘）选项，如图 25.10（左）所示。

⑤ 选择 Select Language（选择语言）选项，如图 25.10（中）所示。

⑥ 选择你想要修改的区域，比如，法语（法国）、英语（美国）或英语（英国），如图 25.10（右）所示。

记住上面的操作过程（以及相关的图标，比如图 25.11 中显示的 Settings 图标），因为你可能会需要在一种完全不懂的语言下重新修改语言设置。

图 25.10 更改设备语言设置

图 25.11 英语和法语下的 Settings 图标(左)和菜单(右)

Download at http://www.pin5i.com/

Tip

你还可以使用 Android 模拟器提供的 Custom Locale 应用程序创建自定义的区域，以供测试使用。例如，你可以为墨西哥创建西班牙语的区域设置。当你使用自定义区域设置时，模拟器将显示默认语言，但是当前区域将是你所设置的，从而使你能够本地化应用程序。

国际化是开发团队在设计时需要做出的选择。例如，是构建一个适应所有语言的大工程，还是将其按照不同的区域进行拆分？对于一些轻量的国际化，你或许可以在一个工程中包含所有的国际化资源，但对于更深入的国际化，你就应该避免应用程序过大，进而降低用户体验。

由于大部分的应用程序本地化工作都与替代资源相关，这意味着这些工作可能不会由使用 Eclipse 和 Android 工具编写代码的开发人员来完成，而是交给了一些资深的设计人员，他们知道 Android 国际化是如何实现的，知道如何排布资源，以及过于国际化的缺点（导致过于庞大的应用程序包）。从积极方面来讲，这使得开发人员能够更加专注于自己擅长的工作：编写代码。

最后，你或许已经意识到了，Android 替代资源的结构并不完美，特别是对于一些具有多官方语言的国家而言。可以尝试通过程序基于当前的本地化设置决定应用程序所载入的资源，但只有在万不得已时才这么做。

25.4.2 在程序中实现本地化支持

有时你应该确保你的应用程序能够对本地化设置做出响应，也就是说，你的代码需要足够灵活，以至于能够在不同的本地化设置的情况下流畅地运行。当你的应用程序需要实现这一点时，比如需要从远程应用程序服务器下载适当的内容，你需要依赖一些“区域友好(locale-friendly)”的方法。

Tip

要确定设备的区域设置，开发人员可以使用 `java.util.Locale` 类的 `getDefault()` 方法。类似地，你可以使用 `getAvailableLocale()` 方法来获取设备可用的本地化设置列表。不是所有的设备都支持全部的区域设置；设备制造商或者运营商有可能只引入 Android 操作系统所支持的全部本地化设置中的一部分。例如，在美国你可以看到只支持英语和西班牙语的设备。另外，你还可以使用 `android.content.res.Configuration` 类获取有关设备配置的更多信息，包括本地化设置。

下面是开发能够在各种本地化设置下工作的应用程序的一些建议。

① 首先要明白世界上不同地区的人表达日常信息的方式是不同的。尊重这些差异，并且在可能的情况下提供对这些差异的支持。

② 在国际化的 Java 应用程序中使用标准的方法。

③ 大多数与本地化相关的应用程序代码都与使用和格式化字符串、数字、货币、日期和时间相关。包含语言的音频和视频通常也在本地化的范围内。

④ 不要先入为主地确定你的应用程序将运行在什么区域，以及所使用的字符编码。

⑤ 在可能的情况下，始终使用“区域友好”的方法。通常，一个类拥有两个版本的方法：一个使用当前的本地化设置，而另一个则包含区域参数，它将在必要时覆盖前者的行为。

Download at http://www.pin5i.com/

Tip

Android SDK 为本地化及特定区域的字符串（如 String.format()方法）、日期（如 java.text.DateFormat 类）和电话号码（如 android.telephony.PhoneNumberUtils 类）处理提供了一系列功能类及其方法。需要注意的是，还有许多方法允许传入本地化参数，以便开发人员能够覆盖在当前本地化设置下相关方法的行为。

25.5 面向不同设备配置

Android 设备具有各种不同的颜色、形状和尺寸。有时你会想在一个应用程序中支持所有的设备，而有时你也可能会希望限制应用程序能够支持的设备。当然，这完全取决于你的用户群，这样他们能够有很好的体验。

开发人员可以通过修改 Android Manifest 标记而使应用程序面向不同的设备配置，包括<supports-screen>、<uses-configuration>、<uses-feature>以及<uses-sdk>。开发人员还可以使用特定的编程技巧使应用程序具有前向和后向的兼容性。

25.5.1 支持硬件配置

让我们来仔细看看你可以在 Android Manifest 文件中使用的标记，它们可以在应用程序中使用，帮助你指定目标设备。

① 你可以使用<supports-screen>标记来限制应用程序所支持的屏幕类型，这在之前的章节中已经讨论过了。

② 你可以使用<uses-configuration>标记来指定应用程序所支持的设备配置(通常与输入方法有关)，以及其是否影响应用程序的正常运行。

③ 你可以使用<uses-feature>标记来指定应用程序所支持的更多特性，以及其是否影响应用程序的正常运行。例如，可以使用这一标记来指定应用程序所支持的 OpenGL ES 版本，或者可选硬件（摄像头、麦克风、传感器）等。关于<uses-feature>标记的完整列表，可以参考定义在 PackageManager 类文档中的常量。

④ 你可以使用<uses-sdk>标记来限制应用程序所支持的平台，即 SDK 版本的范围(由 API Level 定义)。该信息是 Android 操作系统强制要求的。

更多有关以上标记的细节，可以参考 Android SDK 文档：http://developer.android.com/guide/topics/manifest/manifest-intro.html。

Tip

通过 Android Market 发布应用程序的开发人员还可以指定应用程序所部属的国家。在第 29 章“销售你的 Android 应用程序”中，我们会详细地讨论 Android Market。

25.5.2 面向不同 Android SDK 版本

当一个新版的 Android 发布后，开发人员需要关注新的 SDK 组件中增加了什么、去掉了什么。首要的是，你需要确定在应用程序 Android Manifest 文件中所使用的<uses-sdk>标记是正确的。

Download at http://www.pin5i.com/

Tip

要在运行时确定设备构建（device build）的细节，你可以使用 `Build` 类（具体地，`android.os.Build.VERSION`）。

你还可以使用其他的一些技术来确定应用程序对于不同 Android SDK 的兼容性，包括：

① 确定是否使用基于某 API Level 的 Android SDK 所提供的功能；

② 当支持的设备运行在较旧版本平台上的时候，使用 Java 反射（reflection），以便条件化地载入新的 SDK 特性。

1. 确定何时引入 SDK 特性

在开发面向多版本平台的应用程序时，开发人员需要特别注意 SDK 中与特定包、类、方法或常量相关联的 API Level，它一般会在 Android SDK 的文档中给出（如图 25.12 所示）。

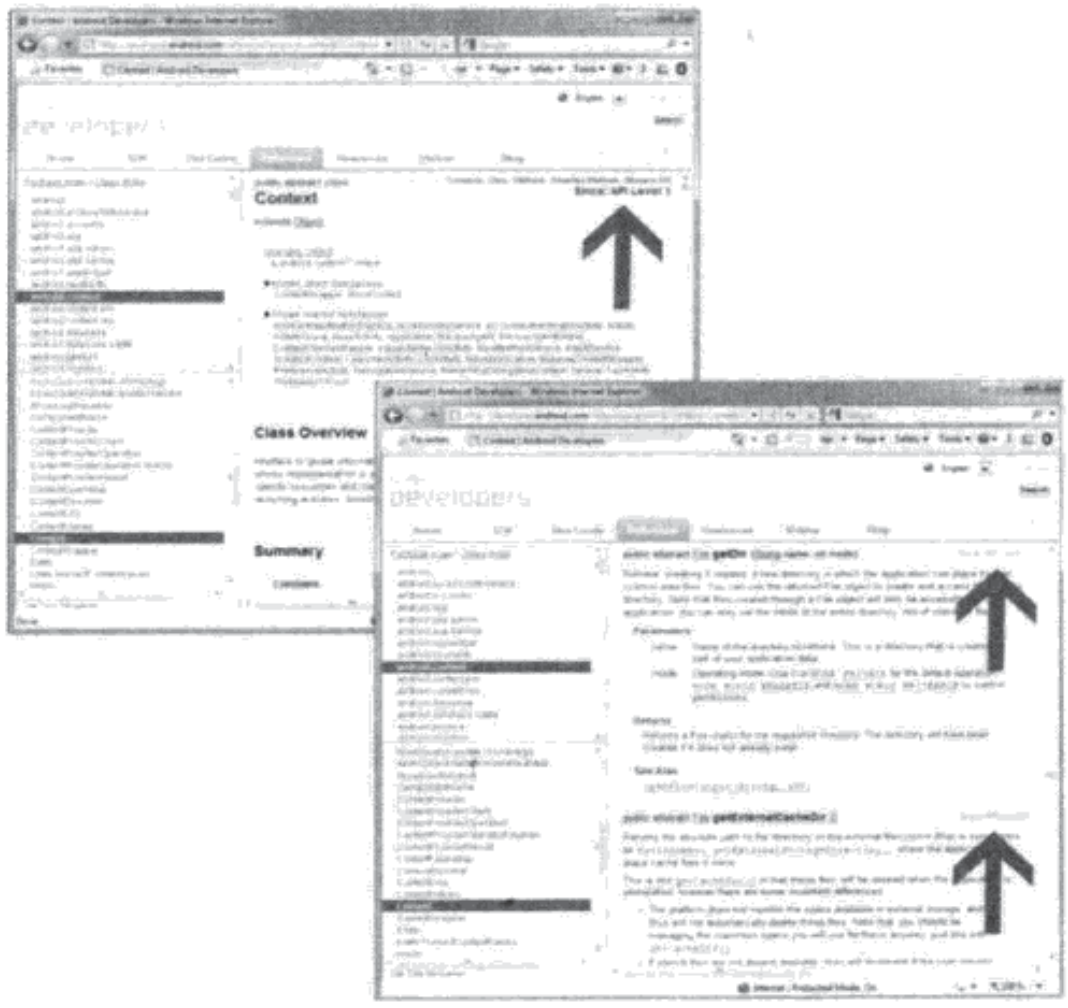


图 25.12 如何确定特定包、类或方法的 API Level

特定 SDK 组件的 API Level 可能会在文档中不同的地方给出。

① 对于包、类和接口，其 API Level 会显示在 Java 文档的右上角。

② 对于方法和常量，其 API Level 显示在方法描述中方法名称的右侧。

2. 使用 Java 反射确保后向兼容

幸运的是，Android 应用程序使用 Java 进行编写，而 Java 语言支持反射。这意味着开发人员可以在运行时通过程序获取方法和类。

当你的应用程序想要使用 Android SDK 的新功能，但仍支持旧版本的时候，它将非常有用。使用这一技术，你需要在旧版本的 SDK 上进行工程的构建，然后在新版本的 SDK 上运行。这一技术有效的原因在于你不会在代码中引用不存在的新方法或类。相反，可以通过向 Java 反射方法传入字符串来实现条件载入。

Download at http://www.pin5i.com/

Tip

有关 Java 反射和后向兼容的更多内容，可以参考 Android 开发人员网站（http://developer.android.com/resources/articles/backward-compatibility.html）以及 Android 开发人员博客（http://android-developers.blogspot.com/2010/06/allowing-applications-to-play-nicer.html）。

我们来看一个示例。要通过名称获取一个方法，你可以使用你想引用的类的静态对象的 `getMethod()` 方法。下面的代码使用 Java 反射获取了 `Environment` 类的 `getExternalStoragePublicDirectory()` 方法。这一方法在 API Level 8（Android 2.2）中被引入，之前的平台并不支持：

```
Method methodExternalStoragePublicDirectory;
try{
    methodExternalStoragePublicDirectory =
        Environment.class.getMethod(
        "getExternalStoragePublicDirectory",
        new Class[] {String.class});
} catch (Exception e) {
    // 失败，不会运行 API Level 8
    // 提供替代功能
}
```

如果方法可用，那么可以使用 `methodExternalStoragePublicDirectory` 对象通过 `invoke()` 方法执行调用，如下所示：

```
methodExternalStoragePublicDirectory.invoke(null, picture_directory);
```

使用这一技术可以使开发人员在保证后向兼容性的情况下，使用新版本 Android SDK 提供的特性，它可以节约开发的时间和精力，并且可以避免针对不同平台开发不同版本的应用程序（这通常会使用户感觉混乱）。

不过，也不能滥用这一技术，因为 Java 反射仍然会带来性能上的消耗。通过 `invoke()` 方法执行调用比正常的调用要慢，通常的做法是将所有的反射逻辑都放在一个新的抽象类中，并且简化对它的使用，不过这又会增加一层调用堆栈。另外，这一技术还会让你的代码变得更难阅读和维护。

Tip

Android 开发人员博客中有一些很方便的小技巧，可以在不使用反射的情况下最大化后向兼容性。你可以从以下网址获得更多内容：http://android-developers.blogspot.com/2010/07/how-to-have-your-cupcake-and-eat-it-too.html。

小　结

兼容性是一个广阔的话题，在本章中我们进行了一些思考。在应用程序的设计和实现阶段，随时考量你的决定是否为设备兼容性带来障碍。质量保障人员也应该在尽可能多的设备上进行测试。根据你的目标，应用程序既可以具有对设备的广泛支持，也可能对其进行限制。你需要做的是让应用程序具有尽可能强的兼容性，并且维护好与兼容性相关的资源和代码。

从本章的内容中你应该可以总结出，替代资源的用处非常大。无论是针对屏幕差异还是国际

Download at http://www.pin5i.com/

化，它们都能够很好地提供兼容性和灵活性。另外，特定的 Android Manifest 文件标记还能够保障应用程序安装在其所支持的设备上，比如对 OpenGL ES 的版本，或者是否具有摄像头硬件等。

参考资料和更多信息

Android Dev Guide（开发指南）：Alternative Resources.http://developer.android.com/guide/topics/resources/providing-resources.html#AlternativeResources.

Android Dev Guide：How Android Finds the Best-Matching Resource.http://developer.android.com/guide/topics/resources/providing-resources.html#BestMatch.

Android Dev Guide：Handling Runtime Changes.http://developer.android.com/guide/topics/resources/runtime-changes.html.

ISO 639-1 Languages：http://www.loc.gov/standards/iso639-2/php/code_list.php.

ISO 3166-1-alpha-2 Regions：http://www.iso.org/iso/country_codes/iso_3166_code_lists/.

Android Best Practices（最佳实践）：Compatibility.http://developer.android.com/guide/practices/compatibility.html.

Android Best Practices（最佳实践）：Supporting Multiple Screens.http://developer.android.com/guide/practices/screens_support.html.

Download at http://www.pin5i.com/

第六部分

对外发布你的 Android 应用程序

第 26 章 移动软件开发过程

第 27 章 设计和开发“防弹”Android 应用程序

第 28 章 测试 Android 应用程序

第 29 章 销售你的 Android 应用程序

Download at http://www.pin5i.com/

第26章

移动软件开发过程

移动开发过程与传统的桌面软件开发过程大体相似，但某些方面还是具有明显的区别。理解两者的差异对于成功地运营整个开发团队有着积极的影响。无论是对于经验丰富的开发人员，还是刚刚踏入移动开发领域的初学者而言，深入了解移动开发过程都有着积极的意义，同时这些知识还将帮助项目管理人员更好地计划和管理项目团队，包括开发人员和测试人员。在本章中，我们将纵览移动软件开发的过程，并深入每一个步骤，详细探讨移动开发的独特之处。

26.1 移动开发过程概览

移动开发团队通常规模较小且项目开发周期较短。整个项目的开发周期一般会被压缩，无论你是一个人的团队还是拥有一个一百人的团队，理解和审视开发过程中的每一个部分将为你节约宝贵的时间与精力。要成功地运营整个移动开发团队，你需要克服下面这些困难：

① 为你的移动项目选择一种合适的软件开发方法；
② 理解目标设备将怎样实现你应用程序的功能；
③ 持续进行全面、精确的可行性分析；
④ 使用试验性的设备降低风险；
⑤ 通过配置管理跟踪设备的功能变化；
⑥ 在内存受限的系统上设计稳定且响应性强的应用程序；
⑦ 设计能够适用于不同设备且具有不同用户体验的用户界面；
⑧ 在目标设备上对应用程序进行全面、彻底的测试；
⑨ 积极引入第三方需求，以便为应用程序的销售带来便利；
⑩ 部署和维护移动应用程序。

26.2 选择一种软件开发方法

开发人员可以很容易地将最新、最流行的软件开发方法引入移动开发当中。不过，无论你的项目团队选择的是传统的快速应用程序开发（Rapid Application Development，RAD）原则，还是诸如 Scrum 这样的更为新潮的敏捷软件开发（Agile Software Development），移动应用程序都有其特殊的需求。

Download at http://www.pin5i.com/

26.2.1 理解“瀑布模型”的危险

由于开发周期较短，某些开发人员可能会倾向于选择“瀑布模型”作为开发方法，但开发人员应该意识到这种选择会带来灵活性方面的问题。不重视在开发周期中有可能出现的许多变化，而一味对整个应用程序进行设计和开发，不是一种明智的选择（如图 26.1 所示）。目标设备（特别是试验性设备）的变化、不断发展的可行性、对性能的要求，以及提早引入 QA 人员并在目标设备（而不只是模拟器）上进行测试，这些需求都使得“瀑布”方法难以为移动项目的开发带来成功。

图 26.1 “瀑布”开发的危险（感谢 Amy Tam Badger 绘制此图）

26.2.2 认识迭代式开发的价值

由于移动项目开发进程较快，迭代式方法（iterative methods）成为了适用于移动开发的最成功的策略。快速的原型开发（prototyping）使得开发人员和 QA 人员能够有充足的机会对移动应用程序在目标设备上的可行性与性能进行评估，并且能够使应用程序适应开发过程中不可避免地产生的一些变化。

26.3 整合应用程序需求

移动应用程序的需求分析较之传统的桌面应用程序更复杂。需求往往会被裁剪，以适应在多种设备上运行——不同的设备往往在用户界面和输入法上具有较大的差异。这使得开发的前提和假设更为复杂。

26.3.1 确定工程需求

在引入了多个设备之后，通常有两种确定工程需求的方法：

① 最小公倍数（lowest common denominator）方法；

② 定制（customization）方法。

这两种方法都有其各自的优缺点。在使用“最小公倍数”方法时，你需要更好地设计应用程序以使其顺利地运行在多种设备上。在这种情况下，主要的目标设备通常应该具有最少的特性——基本上为最低级的设备。只有匹配所有设备的需求才能够被加入到规范（specification）

Download at http://www.pin5i.com/

中，以最大范围地适应设备，比如对于输入法、屏幕分辨率和软件开发包（Software Development Kit）版本的需求。

Note

“最小公倍数”方法大致上与开发符合以下需求的桌面应用程序相似：① Windows 2000；② 128MB RAM 且假设应用程序能够与最新版本（及其中间版本）的 Windows 相兼容。这并不理想，但在某些情况下，这种“交易”是可以接受的。

某些轻微的自定义，比如资源和最终编译的二进制文件（以及版本信息）在最小公倍数方法中通常是可行的。这种方法的主要优点在于只用维护一个主要的源代码，在一个地方修正了 bug，就等同于在所有设备上都进行了修正。除此之外，你还可以轻松地引入新的设备，而不必修改太多的代码，因为这一设备应该能够满足最低的硬件需求。当然，这种方法也存在一些缺点，比如，开发出来的应用程序显得较为普通，并不能最大化设备的特有功能，也无法充分发挥新平台的优势。而且，如果产生了针对某个特定设备的问题，或者你错误地估计了“最小公倍数”，之后发现某个设备不能满足最低需求，那么整个团队将对此进行补救，或者衍生出新的代码，这样，就失去了这种方法的优点反而凸显了其劣势。

Tip

Android 提供的一系列工具使得应用程序的多版本平台支持变得更容易。开发人员应该提早地在设计阶段就考虑目标设备的问题。也就是说，用户可能会经常在线进行固件更新，所以设备平台的版本可能会随时变动。因此，你需要始终把应用程序的向上兼容性纳入考量范围，并且在必要时能够及时发布应用程序的升级补丁。

使用“定制”方法时，应用程序会针对特定的设备或设备类型进行裁剪（例如，所有使用 OpenGL ES 2.0 的设备）。这种方法适用于只针对小范围目标设备进行开发的情况，从资源管理的角度而言不具有普适性。通常存在一个核心的应用程序框架（类或包），它为所有版本的应用程序所共享。所有版本的客户/服务器应用程序共享同一个服务器，并且以相同的方式与其进行交互，但客户端的实现却根据特定设备的特性进行裁剪。这是这种方法的关键优势。它的缺点在于源代码不够统一（同一代码具有许多衍生）、测试需求增大，而且今后新添加设备时会显得比较困难。

事实上，移动开发团队通常将这两种方法混合起来使用，而取各自的优点。我们经常能够看到开发人员将某些特定的设备根据其功能分成一组。例如，游戏应用程序可能会依照设备的图形性能、屏幕分辨率或输入法对其进行分组；位置服务（Location-Based Service，LBS）应用程序可能会依照设备的内部感应器对其进行分组；其他应用程序可能会对具有内置摄像头和不具有内置摄像头的设备分别开发不同的版本。这些分组方式由开发人员决定，目的在于便于代码的编写和测试。他们在很大程度上会关注特定应用程序的细节和任何支持性方面的需求。在许多情况下，这些功能也可以在运行时检测到，但这样会使代码变得比较繁杂。

Tip

具有单一而统一版本的应用程序往往比具有明确版本区分的应用程序更廉价且更易获得广泛的支持。不过，具有定制版本的游戏往往会销售得更好，因为它能够发挥出特定设备组的显著优势；而一个统一的垂直商务应用程序却往往会获得更大的好处，因为它使得用户在不同的设备上都能够顺利地使用，同时可以为其节约一定的商务成本。

Download at http://www.pin5i.com/

26.3.2 为移动应用程序开发用例

具有概括性的应用程序用例（use case）应该在适用于特定设备之前进行开发，因为考虑特定的设备会具有一定的局限性。例如，一个针对应用程序的高级（high-level）用例可能会仅仅是“输入表单数据”，但在特定设备上时却可能需要采用不同的方法以适应设备的硬件环境，比如考虑设备使用的是硬件键盘还是软键盘等。

Tip

为不同的设备开发应用程序与开发不同操作系统和输入设备（比如处理 Mac 键盘和 Windows 键盘上的快捷键）上的应用程序非常相似，都必须考虑微妙的和不那么微妙的差异。这些差异可能会非常显著，比如是否拥有一个用来进行输入的键盘；也可能会很微妙，比如特定设备上的 bug，或软键所使用的不同约定。在第 25 章“面向不同设备配置和语言”中，我们就设备兼容性进行了讨论。

26.3.3 吸纳第三方需求

除了在你进行内部需求分析时所产生的需求之外，你的团队还需要考虑其他人所提出的需求。第三方需求可以来自于任何方面，包括：

① Android 许可协议（License Agreement）需求；
② Google Maps API 许可协议需求（如果涉及的话）；
③ 第三方 API 需求（如果涉及的话）；
④ Android Market 需求（如果涉及的话）；
⑤ 移动运营商需求（如果涉及的话）；
⑥ 其他应用程序商店的需求（如果涉及的话）；
⑦ 应用程序证书（certification）需求（如果涉及的话）。

在项目设计阶段考虑并吸纳这些需求是非常重要的，它不但使你的项目能够如期完成，而且，由于你在设计之初就对它们进行了考虑，所以还降低了事后再弥补所产生的风险。

26.3.4 管理设备数据库

随着你的移动开发团队逐渐为越来越多的设备开发了应用程序，跟踪记录这些目标设备的信息就显得越来越重要，它对收益估计和程序、设备的维护都非常重要。为设备创建数据库是用于跟踪市场以及目标设备特性细节的很好的方法。我们这里提到的“数据库”既可以指 Microsoft Excel 电子表格中所记录的信息，也可以指一个小型的 SQL 数据库。其关键点在于这些信息可以在团队或公司内部进行共享，同时获得更新。将设备分为若干类将会很有用，例如将支持 OpenGL ES 2.0 的设备分开，或是将没有硬件摄像头的设备分开。

在建立起了项目需求、确定了目标设备之后，设备的数据库需要及早地实现。图 26.2 说明了如何在应用程序开发团队内部跟踪记录并使用设备的信息。

1. 确定跟踪的设备

一些公司仅仅跟踪其经常用来进行开发的设备，但其他一些公司也会跟踪那些开发“优先权”不高或今后可能用来进行开发的设备。设备可以在项目需求阶段被包含进数据库，可以在

Download at http://www.pin5i.com/

项目的整个周期中进行修改，也可以在初始应用程序发布之后将其添加到后续的移植项目中。

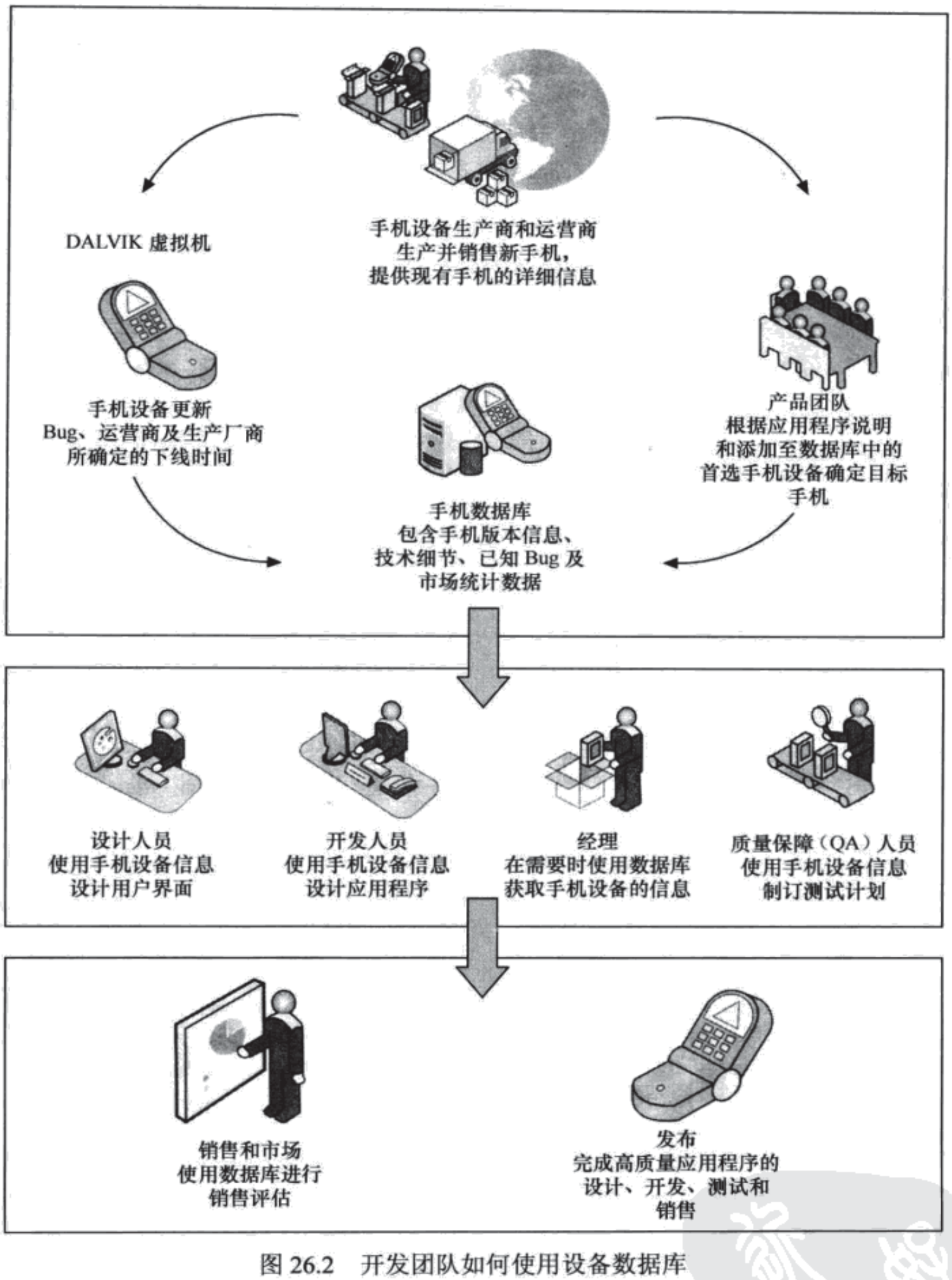


图 26.2 开发团队如何使用设备数据库

2. 存储设备数据

设备数据库应该被设计为能够囊括特定设备对于应用程序的开发和销售有帮助的所有信息。这可能需要专人负责跟踪来自运营商和制造商的持续不断的信息。同样地，这些信息可以在公司内的所有移动项目开发团队中进行共享。这些数据应该包括：

① 重要的设备技术说明细节（屏幕分辨率、硬件细节、支持的媒体格式、输入法和本地化等）；

② 有关设备的任何已知情况（bug 和重要的限制等）；

③ 设备运营商信息（固件定制、发布和停产期以及期望的用户统计信息等，期望的用户统计

Download at http://www.pin5i.com/

信息包括是否具有较高的预期、是否有较大的市场以及在垂直市场上是否受到了广泛的欢迎等）；

④ 固件升级信息（当具有了这一信息之后，固件的改变就不会对应用程序产生影响，或者能够保证设备的独立性）；

⑤ 实际的测试用设备信息（哪些设备是从制造商或运营商处购买或租借的，有多少是可用的等）。

设备运营商的信息可以与来自运营商、应用程序商店和内部统计的销售数据互相参照。

实际的测试用设备信息通常可以实现为一个仓库的“进出仓”系统。团队成员可以为测试和开发目的而持有设备。当租借来的设备需要归还给制造商时，使用这个系统就能够很方便地对其进行跟踪了。这还可以减轻在不同团队之间共享设备的困难。

3. 使用设备数据

记住数据库可以在多个移动开发团队之间共享。设备资源可以共享，销售统计数据可以用来进行比较，以确定应用程序在哪个设备上获得了最大的成功。设备数据库对于不同的团队成员（或角色）可能有不同的作用。

① 产品设计人员可以使用数据库来开发针对目标设备最合适的应用程序用户界面。

② 媒体设计师可以使用数据库来生成应用程序组件，如图像、视频、音频等，它们都需要匹配目标设备所支持的媒体格式以及屏幕分辨率等。

③ 项目经理可以使用数据库来确定必需的设备，以供开发和测试使用，并且可以了解到它们的优先级。

④ 软件开发人员可以使用数据库来设计和开发与目标设备兼容的应用程序。

⑤ QA 人员可以使用数据库来设计和编制针对目标设备的测试计划，并且对应用程序进行全面的测试。

⑥ 市场和销售人员可以使用数据库来评估应用程序的销售数据。例如，在设备的占有率下降时需要意识到对应应用程序的销售也会受到影响。

另外，数据库中存储的信息还能够帮助预测出最有前景目标设备，以便在今后进行开发和移植。

4. 使用第三方设备数据库

第三方设备数据库的信息中可能包含屏幕尺寸、内部硬件细节和运营商支持细节等，但订阅这些信息对于小公司而言可能开销过大。许多移动开发人员则选择仅对它们感兴趣的设备创建相应的数据库，记录与这些设备相关的数据，而这些数据往往难以在免费、开放的数据库中获得。

26.4 项目风险评估

除了必须确定的任何软件项目都存在的普通风险之外，移动项目的开发还需要注意那些会对项目进度和需求造成影响的外部因素。这些风险因素包括目标设备的确定和获取，以及对应用程序可行性的持续分析。

26.4.1 确定目标设备

明智的软件开发人员在开发桌面应用程序之前都需要首先确定其所运行的操作系统（及

Download at http://www.pin5i.com/

其版本），同样地，移动开发人员也必须首先确定应用程序所运行的目标设备。每一种不同的设备都拥有不同的功能、用户界面及其自身局限。

目标设备的选择通常依据以下两个原则中的一种。

① 你想要开发运行在流行的“杀手级”设备上的应用程序。

② 你想要开发的应用程序能够运行在尽量多的设备上。

在第一种情况下，你需要找到一款这样的目标设备；而在第二种情况下，你需要了解市面上已有（或即将上市）的设备，然后调节应用程序的功能及特性，以适应最大数量的设备。

Tip

在 Android 平台上，你通常不会针对某一款设备进行开发，而是基于一定的设备功能（例如，运行了特定版本平台的设备或者拥有某个硬件配置的设备）。你可以使用 Android Manifest 标记来限定应用程序所安装的设备，同时它也供在线市场对应用程序进行筛选。在第 5 章“使用 Android Manifest 文件定义应用程序”中，我们讨论了很多有关 Android Manifest 标记和市场筛选器的话题。

1. 理解制造商和运营商的作用

我们需要注意到，一些很流行的设备，比如 Android 设备的 Droid 产品线，都被许多运营商所定制。运营商可以向设备移植自己的版本，包括不同的用户体验、不同的皮肤，以及大量的自定义应用程序（它们会消耗掉大量的空间），运营商甚至会去掉设备的某些特性（比如蓝牙或 Wi-Fi），它们可能会让你的应用程序无法正常运行。你必须在分析应用程序需求和功能时将所有的这些因素都纳入考虑的范畴。应用程序的运行需求必须与这些所有目标设备所共有的特性相匹配，并且能够对可选特性进行适当的处理。

2. 理解时间对市场的影响

设备不断推陈出新，运营商和制造商也不断地将“退休”的设备撤下柜台，不同的运营商可能会运营相同（或相似）的设备，但将它们撤离市场的时间却有所区别。

Tip

开发人员应该发表声明，让用户能够明确地知道，在运营商或制造商停止生产某台设备之后，软件提供者对应用程序的支持能够持续多长时间。这一声明可能需要针对不同的运营商而适当调整，因为不同的运营商有其自身的需求。

开发人员需要确定在全球市场上发布的设备型号有什么不同。某些设备只在一部分区域发布（或流行）。有些时候手机是全球发布的，但更普遍的是只在局部地区发售。例如，T-Mobile G1 首先在美国市场发售，后来才在全球范围内普及。类似地，摩托罗拉的 Droid 曾一度仅在美国市场发售，而其 Milestone 型号产品则仅在美国以外的市场发售。

另外需要注意的是，相同设备在世界上的不同地区的发售时间会有所不同，也会因为其市场需求而有一定的差异。

26.4.2 获取目标设备

拿到一部目标设备的时间越早，对于你的开发工作而言就会越好。有时获取一部目标设

Download at http://www.pin5i.com/

备也就是到商店里把它买回来而已，有些时候，你需要通过一些特殊途径获取设备。

对于应用程序开发人员而言，瞄准一部即将上市的目标设备是很常见的，这些设备可能还处在移植当中，无法被消费者所购买。在消费者第一时间买到新设备时，你的应用程序就能够在其上正常地运行，这是非常具有竞争力的。对于试验设备而言，你可以加入到制造商和运营商对开发人员所提供的项目当中。这些项目将帮助你与设备的生产线（即将推出的型号、停产的型号）并驾齐驱。许多项目还包含试验设备的租借服务，它使得开发人员可以先于消费者而获得需要的设备。

Tip

如果你正准备购买 Android 设备，我们建议你考虑 Android 开发用机，如 Google Nexus One。Android 开发用机解锁了 SIM，并且无需任何合约，意味着你可以在任何 GSM 网络下使用它们。Android 开发用机还提供引导功能，以便你安装自定义的系统镜像。你可以在本章“参考资料和更多信息”中找到购买这种设备的方法。

对于开发人员而言，在某个特定型号的试验设备上开发应用程序是要面临一定风险的，因为设备的移植时间可能会有所变化，平台也可能会时常出现固件问题，也就是说，设备可能会被推迟或取消发布。另外，设备所拥有的特性（特别是那些新奇而有趣的特性）也不是毫不动摇的，只有在确实地实现了移植并且开发人员验证了这些特性之后才能够最终确定下来。因为不断地有人宣称一些令人兴奋的新特性，而你可能又想利用这些特性开发应用程序，所以，你的项目计划需要有足够的灵活性以适应市场的变化。

26.4.3 确定应用程序需求的可行性

移动开发人员在面临设备的各种局限时只能任由其摆布，这些局限包括内存容量、处理器性能、屏幕类型和开发平台版本等。移动开发人员不具有传统桌面应用程序开发人员所拥有的那种奢侈的特权，他们不能宣称应用程序需要“更多的内存”、“更多的空间”。设备的局限性基本上可以说是固有的，所以移动应用程序要么在这个限制下运行，要么根本不能运行。从技术上讲，绝大部分 Android 设备都有一定的硬件弹性，例如可以使用 SD 卡一类的外部存储设备，但资源的局限性依然存在。

真正的可行性分析只能在真实的物理设备上进行，而不能由模拟器代劳。应用程序在模拟器上可能可以完美地运行，但在实际的设备上却步履蹒跚。所以，移动开发人员需要在整个开发过程中不断地进行可行性的评估以及性能和响应性的测试。

26.4.4 理解质量保障的风险

质量保障（Quality Assurance，QA）团队会因为测试环境的不佳而相应地减少其工作量。

1. 早测试、常测试

越早获得目标设备越好。从制造商处获得一部试验设备可能需要好几个月的时间。参与运营商的设备租借项目或者从零售点购买设备虽然比较麻烦，但很多时候也是有必要的。总之，不要等到最后一刻才动手去收集需要的测试设备。

2. 在真实设备上进行测试

模拟器固然有用，但真实设备才是“王道”。从事实上讲，应用程序能否在模拟器上运行

Download at http://www.pin5i.com/

并不重要——没有人在现实生活中会使用模拟器。

你可以将设备“恢复原厂设置”或是清除所有用户数据，但是一般而言，并没有哪种方法能够将设备“清理”成最初的洁净的状态，所以，QA 人员需要确定设备处于某种洁净状态，将其作为测试的原则之一，并严格遵守。测试人员可能需要掌握使用不同的固件版本刷新设备的技巧，以及不同版本平台间的微妙差异，还需要了解底层应用程序数据在设备中是如何存储的（例如 SQLite 数据库，私有应用程序文件以及缓存的使用）。

3. 降低在有限的真实环境下测试的风险

从某种意义上讲，QA 测试人员工作在一个可控的环境中，对于移动测试人员更是如此。他们所使用的设备往往不能连接真实的网络，而且试验设备可能与市面上销售的不吻合。除此之外，由于测试通常都是在实验室进行的，所以位置（包括主蜂窝基站、卫星定位坐标和相关设备信号强度、数据服务的可用性、LBS 信息以及本地化信息等）是固定的。因此，QA 团队需要充分发挥创造性才能减小类似由于范围狭窄等因素而产生的测试风险。例如，在手机无信号（以及飞行模式等）的情况下测试应用程序是非常有必要的，这样可以检测应用程序是否会在这种我们经常遇到的情况下崩溃。

4. 测试客户/服务器应用程序

确保 QA 团队明确并理解他们的职责。移动应用程序往往拥有网络组件和服务器端功能实现。要保证完整的服务器测试被纳入了总体测试，而非仅仅测试设备上实现的客户端部分。这可能需开发相应的桌面应用程序或网络应用程序，来实现对总体解决方案中服务器端的测试。

26.5 编写基本项目文档

由于移动应用程序开发周期较短、人员较少且功能较简单，你可能会认为编写移动软件工程的文档会相对轻松一些。不幸的是，事实并非如此，与你期望的恰恰相反。和传统的软件项目一样，好的项目文档也会给移动开发带来各种好处。在你的项目中应该考虑编写如下文档：

① 需求分析及优先级别；
② 风险评估和管理；
③ 应用程序构架与设计；
④ 可行性研究与性能评估；
⑤ 技术规范（总体、服务器、特定设备客户端）；
⑥ 详细用户界面规范（总体、特定设备）；
⑦ 测试计划、测试脚本、测试实例（总体、特定设备）；
⑧ 范围（Scope）变更文档。

在一般的软件开发项目中，上面这些文档都是很常见的。但也有可能你的团队会发现省略上面的某些文档在过去也是可行的。在你决定省略移动开发项目中的某些文档之前，需要仔细考虑它对成功完成这一项目所起的作用。某些项目文档可能会比那些稍大规模的软件项目要复杂，但其他的一些文档可能需要填充很多细节，特别是用户界面和可行性研究。

Download at http://www.pin5i.com/

1. 为质量保障制定测试计划

质量保障强烈地依赖于功能文档和用户界面文档。对于屏幕较小的移动设备而言，屏幕尺寸异常宝贵；对于成功的移动项目来说，用户体验至关重要。

（1）理解用户界面文档的重要性

没有一个“杀手级”应用程序拥有一个设计简陋的用户界面。在任何移动软件项目的设计阶段，深思熟虑地敲定用户界面设计都是非常重要的。应用程序的工作流（应用程序状态）必须完整地在文档中表示出来，要深入到每一个屏幕的显示，并且包含按键使用模式的详细说明，以及在错过这些按键或相关功能时该如何回退。用例需要提前予以清晰地定义。

（2）使用第三方测试

一些公司选择将质量保障交给第三方来完成；大多数的质量保障团队需要详细的说明文档，包括用来确定应用程序行为的用例工作流图表。如果你没有为测试人员提供足够充分且详细的文档，那么你将不可能获得深入而细致的测试。通过提供详细的文档，你可以使你应用程序的级别从“能运行”提高到“能正常运行”，原因是显而易见的。

2. 提供第三方所需的文档

如果你需要将应用程序提交给软件认证项目，甚至在移动应用程序商店里进行发布，你所提交的资料当中应该包含有关应用程序的文档。例如，某些应用程序商店要求应用程序必须包含帮助特性或者技术支持等信息。认证项目可能需要你提供详细的应用程序功能文档、用户界面工作流以及应用程序状态图。

3. 提供维护和移植所需的文档

移动应用程序还经常移植到另外一些设备以及其他移动平台上。这些移植工作通常由第三方完成，这使得完整的功能和技术说明显得更为重要。

26.6 使用配置管理系统

有许多种源代码控制系统可供开发人员使用，其中大部分能供传统开发使用的也能够用于移动项目上。不过，从另一个侧面来看，为应用程序划分版本也不一定就像你想象的那么直截了当。

1. 选择一个源代码控制系统

移动开发需要一个源代码控制系统，这一点也不奇怪。开发人员在为移动项目处理配置管理问题时，应该考虑如下几个方面：

① 能够跟踪源代码（Java）和二进制文件（Android 包等）；

② 能够通过设备配置跟踪应用程序资源（图像等）；

③ 能够与开发人员所选择的开发环境（Eclipse）相融合。

其中需要考虑的一点是开发环境（IDE，如 Eclipse）与源代码控制系统之间的整合。通常使用的源代码控制系统（如 Perforce、Subversion 和 CVS）都能够很好地运行在 Eclipse 下。

Download at http://www.pin5i.com/

2. 实现一个可用的版本管理系统

开发人员应该提早决定版本所使用的模式（schema），以兼顾设备细节和软件构建（build）。通常只针对软件构建来编制版本号（即“版本 1.0.1”）是不够的。

移动开发人员通常将传统的版本模式与目标设备或目标设备组的配置结合起来（版本 1.0.1.重要特性/设备组名称）。这有助于质量保障和技术支持人员，以及不了解设备型号或者只了解产品名称的最终用户。例如，带有摄像头支持的应用程序的版本号可以是“1.0.1.Cam”，其中，“Cam”代表的是“支持摄像头”；这样，不带有摄像头支持的相同应用程序的版本号就应该为“1.0.1.NoCam”，其中“NoCam”代表“不支持摄像头”。如果你有两组不同的维护工程师分别负责为不同的代码分支提供支持，那么只通过版本名称就能够区分它们了。

再复杂一点，你可能需要考虑特定设备的不同固件版本。如果某次升级导致你需要重新构建你的应用程序，你可能需要用适当的方式将版本变更标注出来：1.0.1.NoCam.Upg1 之类的。当然，这样下去会无止境的，所以你需要控制别太过头了，但如果你提前就聪明地设计好了整个版本系统，那么之后就会非常有用，特别是当你或者用户拥有各种不同版本的设备时。最后，你需要谨慎管理应用程序的版本代号。

26.7 设计移动应用程序

在设计移动应用程序时，开发人员必须考虑设备硬件的约束，以及采用什么样的应用程序构架。

26.7.1 认识移动设备的局限

应用程序需要快速、灵敏、稳定，但开发人员能够使用的却是有限的资源。在设计和开发移动应用程序时，必须时刻照顾目标设备内存和处理器性能的局限。

26.7.2 移动应用程序体系结构探索

传统的移动应用程序通常有两种基本模型：单机应用程序和网络驱动应用程序。

单机应用程序将其所需要的所有东西封装进一个安装包中，并且只依赖设备实现其提供的功能。所有的操作都在本地内存中进行，所以会受到设备限制的影响。单机应用程序也可以使用网络功能，但核心功能并不依赖于它们。单机应用程序的一个典型例子就是“纸牌（Solitaire）”游戏。用户可以在飞行模式下玩游戏，程序不会出现任何问题。

网络驱动的应用程序提供轻量级的客户端，而依赖于网络（或“云”）提供其主体内容和主要功能。网络驱动的应用程序通常会将繁重的处理工作交给服务器来完成，而且它们还能够在安装了很长时间之后获取额外的内容或功能的拓展。除此之外，开发人员只需要搭建一个完善的服务器，之后就可以在许多操作系统下开发客户端，从而获得大量的用户，因而网络驱动的应用程序非常受到开发人员的欢迎。网络驱动应用程序的例子包括以下几种：

① 使用云服务、应用服务或网络服务的应用程序；

② 提供内容定制服务的应用程序，包括铃音、壁纸等；

③ 拥有高内存占用和非关键性操作的应用程序，它们可以把这些操作交给强大的服务器来完成，只需要获得返回的结果即可；

Download at http://www.pin5i.com/

④ 任何提供后续功能补充的应用程序，它们并不是一次性地将完整的功能封装进二进制文件中。

在你的应用程序中，网络功能占多重分量是由你决定的。你可以使用网络来仅仅提供内容的更新（流行的新铃音），或者操作应用程序的外观和行为（例如，在线添加新的菜单选项或者新特性）。

26.7.3 可扩展和维护设计

应用程序可以被编写为具有固定的用户界面和功能特性，但也可以不这样做。网络驱动的应用程序设计起来可能更复杂，但对于长期来讲，它更灵活且易于扩展，下面举一个例子。我们假设你想要编写一个壁纸应用程序。你的应用程序可以是单机的，也可以是部分网络驱动或完全网络驱动。无论如何，应用程序都需要以下两项功能。

① 显示一系列的图片，并且允许用户从中进行选择。

② 获取选中的图片并将其设置为设备的壁纸。

一个超级简单的单机应用程序可能会提供一组壁纸。如果它们只具有针对大部分设备的普通尺寸，那么你可能还需要重新进行调整。你可以开发这样的应用程序，但它会浪费很多空间与处理操作，而且你还不能更新可选的壁纸，这基本上是一个糟糕的设计。

部分网络驱动的壁纸应用程序可以为用户提供一个固化的菜单，其中包含壁纸的分类，用户可以使用它们浏览服务器提供的壁纸。应用程序将下载特定的图片，并且将尺寸调整为适应设备屏幕的大小。你可以随时向服务器添加新的图片，但是当每次你想要添加一种新的设备配置或屏幕尺寸时，都需要构建新的应用程序。如果你想要修改菜单，比如添加动画壁纸，你就需要编写一个新版本的应用程序。这种应用程序是可行的，但它仍然没有明智地利用它的资源，也不具有特别强的可扩展性。不过，你可以使用一个单一的服务器来为各种平台的客户端提供服务，包括 Android、BREW、J2ME 和黑莓等，所以，这种模式仍占有一个很好的地位，至少同单机壁纸应用程序比起来是这样的。

完全网络驱动的壁纸应用程序尽其所能在手机上完成最少的操作。客户端允许服务器决定用户界面的外观、所需显示的菜单及其指向的位置。用户可以浏览服务器上存储的这些图片，这与部分网络驱动的版本是一样的，不同的是，当用户选中的某张壁纸后，移动应用程序只向服务器发送一个请求："我想要这张壁纸，并且我是这种类型的设备，具有这种类型的屏幕，尺寸就是如此。"服务器将图像重新格式化并调整好尺寸（密集性操作），最后将经过精确修剪的图像下传给应用程序，应用程序再将其设为手机壁纸。添加对更多手机的支持很简单：发布经过略微修改的轻量级客户端，并且在服务器上添加相应的配置。针对动画壁纸添加新的菜单只是在服务器上所做的更改，但它能够反映到所有的手机（或者说接受服务器指令的手机）上，从而实现一个新的分类。这种应用程序的响应时间依赖于网络性能，但应用程序本身是最动态的，其扩展性也是最强的。当然，这种应用程序在设备处于"飞行模式"时基本上是没有用的。

单机应用程序非常简单，这种模式适合于一次性开发的应用程序，或者是那些不依赖网络的应用程序。网络驱动应用程序需要一点前瞻性，并且在开发时往往会复杂一些，但它在今后可能会省去不少麻烦，同时，长期来看，网络驱动应用程序能够为用户提供更多新的内容和功能。

Download at http://www.pin5i.com/

26.7.4 应用程序交互设计

移动应用程序设计人员应该考虑它们如何与设备上的其他应用程序进行交互，包括由相同开发人员开发的其他应用程序。一些需要思考的问题如下：

① 你的应用程序会依赖于其他的内容提供器吗？
② 这些内容提供器是否保证已经安装在了设备上？
③ 你的应用程序会充当内容提供器吗？它将提供什么数据？
④ 你的应用程序具有后台特性吗？是否作为一个服务？
⑤ 你的应用程序依赖于第三方服务或可选组件吗？
⑥ 你的应用程序是否通过远程接口（AIDL）共享其功能？

26.8 开发移动应用程序

移动应用程序的实现遵循与其他平台开发相同的原则。移动开发人员在实现应用程序时所需的步骤相对简单一点。

① 编写、编译代码。
② 在软件模拟器中运行应用程序。
③ 在软件模拟器中测试与调试应用程序。
④ 封装、打包应用程序，并将其部署在目标设备上。
⑤ 在目标设备上进行测试与调试。
⑥ 综合整个团队的修改意见，对应用程序进行改进，直到顺利完成。

Note

我们将在第 27 章“设计和开发‘防弹’Android 应用程序”中讨论有关构建健壮的 Android 应用程序的相关开发策略。

26.9 测试移动应用程序

测试人员面临着许多的挑战，包括设备分化（许多种设备，每一种都有自己的特性，有人把这称作“适应性”）、设备状态定义（什么是洁净状态）以及现实世界事件的处理（通话、信号丢失）等。为测试而收集设备可能花费不菲，并且也比较困难。

Note

我们将在第 28 章“测试 Android 应用程序”中讨论有关 Android 应用程序的测试相关细节。

对于移动 QA 团队来讲，好消息是 Android SDK 为在模拟器和真实设备上测试应用程序提供了若干有用的工具，有许多机会可以实现“白盒（white box）”测试。

缺陷跟踪系统必须进行修改，以处理不同设备配置和运营商的测试。对于完整测试而言，通常不能交给 QA 团队一台设备，然后让他们“破坏”它。在“黑盒”测试与“白盒”测试之间还有许多“灰色”的地带。测试人员需要了解 Android 模拟器以及其他 Android SDK 提

Download at http://www.pin5i.com/

供的功能。

Tip

本书附录中讨论的工具（尤其是附录 A、B、C 和 D）不仅对开发人员而言意义重大，同时也为测试人员提供了对设备配置更高的控制力。

移动质量保障涉及大量的边缘测试，而且还要考虑到试验型号的设备与最终移植的消费者所使用的设备之间可能会有区别。

26.10 发布移动应用程序

开发人员需要确定发布应用程序所使用的方法。对于 Android 来讲，你有多个不同的选择。你既可以自己销售应用程序，也可以将其交由第三方处理，比如 Android Market。一些成熟的移动市场，比如 Handango，也提供了 Android 的发布频道，你可以使用它们。

Note

我们将在第 29 章详细讲解 Android 应用程序的发布过程。

开发人员必须考虑来自提供应用程序发布机制的第三方的需求。

特定的发布者可能会对他们所发布的应用程序类型有所规定。他们还可能提出一些质量需求，比如测试认证（尽管在本书出版时还没有针对 Android 应用程序的认证）、伴随的技术支持和相关文档，以及对于一般用户界面工作流标准的遵守（也就是说，“Back”按钮应该具有“回退”功能）和出于应用程序敏捷性考虑的性能衡量。另外，发布者可能还会对内容有所限制，比如禁止一些令人不愉快的内容。

26.11 支持和维护移动应用程序

一般而言，如果你具有传统软件开发背景的话，移动应用程序的支持需求可能会显得很简单，但它确实存在。运营商通常站在对最终用户提供技术支持的前沿。作为一个开发人员，你通常不需要 24 小时全年无休的技术支持团队，也无需类似免费服务电话这样的东西。事实上，应用程序维护的主体部分应该落在服务器端，并且限制为内容维护，比如发布新的铃音、壁纸、视频或其他内容媒体。

众所周知，市场上的硬件设备更新非常快速，而移动开发团队需要始终站在市场前列。下面给出一些移动应用程序开发中在提供维护和支持时需要考虑的要素。

Warning

开发人员不能简单地开发一个应用程序、发布它，然后将它抛诸脑后，即便是最简单的应用程序，也需要一定程度的维护和升级。

1. 维护适当的应用程序文档

维护通常不是由应用程序最初的开发人员来完成的。所以，提供足够的开发和测试文档，包括规范和测试脚本，显得更加重要。

Download at http://www.pin5i.com/

2. 管理服务器的动态更改

要始终小心地处理动态的服务器更改。这意味着备份和更新需要适当地同步。在任何情况下，都需要保证数据的安全以及用户隐私的保密。在推出产品时必须特别注意，因为动态移动应用程序的用户可能很大程度上依赖于它的可利用性。千万不要低估服务器端的开发和测试。在实现服务器的动态化之前需要首先在安全的环境中进行测试。

3. 确定低风险移植机会

如果你已经实现了我们在本章中讨论过的设备数据库，那么就可以开始分析设备的相似性，并从中确定比较容易开展的移植项目。例如，你可能会遇到下面的情况：某个应用程序最初针对某种特定设备而开发，而现在市场上已经有了几款类似的设备。将一个现存的应用程序移植到这些新设备的过程有时就简单到生成一个新的构建（包含适当的版本信息）并在设备上进行测试。如果你正确地对设备进行了分类，幸运的话，在某个新设备面世时，你可能不需要对程序进行任何改动，它也能够在新设备上顺畅地运行。

小　结

移动软件的开发随着时间而不断演变，逐渐形成了与传统桌面软件开发的重要差异。在本章中，我们为你提出了一些从传统软件开发过渡到移动开发所需的建议，这些建议涉及从确定目标设备到测试并发布应用程序相关的话题。对于软件开发过程而言，提升的空间是始终存在的，我们提出的一些建议能够帮助你避免许多陷阱，而一些新的移动开发公司往往会跌落到这些陷阱当中；这些建议或许还能够帮助你的团队更快地完成项目的开发工作。

参考资料和更多信息

Wikipedia 软件开发过程（Software Process）：http://en.wikipedia.org/wiki/Software_development_process.

Wikipedia 快速应用程序开发（Rapid Application Development，RAD）：http://en.wikipedia.org/wiki/Rapic_application_development.

Wikipedia 迭代式开发（Iterative Development）：http://en.wikipedia.org/wiki/Iterative_and_incremental_development.

Wikipedia 瀑布开发过程（Waterfall Development Process）：http://en.wikipedia.org/wiki/Waterfall_model.

Extreme Programming：www.extremeprogramming.org/.

Purchase Nexus One and Android Dev Phones: http://android.brightstarcorp.com/index.htm.

Download at http://www.pin5i.com/

第27章

设计和开发“防弹”Android应用程序

在本章中，我们将向你介绍一些经验和技巧，它们是这若干年以来我们在移动软件的设计和开发中所积累和总结的。我们还将提醒你各种各样的陷阱，以及如何避免它们，而无论你是一个设计师、软件开发人员还是移动应用程序项目的管理人员。

如果你刚刚从事移动开发，那么一次性地读完本章可能会有些困难，也没有太大的必要。相反，你可以考虑在整个设计、开发过程的某些环节中，有针对性地阅读相关的章节。在本章中，不是所有的建议都适用于你的项目，也不一定能够加速你的项目进程，不过其中包含的一些成功（或失败）的移动项目开发的例子确实能够为你提供一定的参考，以帮助你提高移动开发项目成功的几率。

27.1 “防弹”移动应用程序设计最佳实践

移动应用程序设计的“规则”非常简单，而且适用于所有的移动平台。这些规则提醒我们，我们所开发的应用程序在设备上始终是第二位的，而这个设备通常来说是一部电话。同时，这些规则还让我们意识到，我们只能在一定程度上进行操作，因为设备的基础硬件是由运营商和设备制造商所掌握的。这些规则还能够在Android软件开发包（Software Development Kit，SDK）的许可协议和第三方应用程序市场的条款、合约中找到。有如下这些“规则”。

① 不要妨碍手机的语音通话和消息服务。
② 不要破坏或试图削弱或扩展设备的硬件、固件、软件和OEM组件。
③ 不要滥用运营商的网络，不要造成网络上的任何问题。
④ 不要辜负用户的信任。

上面这些规则看上去像是不用动脑筋就能想到的，但即使是没有不良动机的开发人员也有可能在不小心或没有经过仔细测试的情况下跌落到上面所提到的“陷阱”中。这对于使用设备网络支持和底层硬件API以及需要存储私有用户数据（如姓名、位置和联系人信息等）的应用程序来讲特别如此。

27.1.1 满足移动用户的需求

移动用户对于安装在设备上的应用程序都有其要求。应用程序应该满足以下需求。

① 具有很强的响应性、稳定性和安全性。
② 拥有简洁直观的用户界面，容易上手。
③ 避免用户在使用过程中遇到障碍。
④ 服务器要全天候地工作（远程服务或服务器随时可用）。

Download at http://www.pin5i.com/

⑤ 包含“帮助”和/或“关于”页面，以提供反馈和支持信息。

27.1.2 为移动设备设计用户界面

为移动设备设计有效的用户界面（UI），特别是需要在各种不同的设备上运行的应用程序，是一种技巧性很强的工作。我们也都见过比较糟糕、过于笨重的移动应用程序 UI，它们会丧失很多用户，而一个设计良好的 UI 却能够在很长时间内保持用户的忠实度。而且，即使应用程序的功能相似，拥有不同 UI 的应用程序之间还是有所区别的；那些拥有良好 UI 的应用程序即使功能上逊色一点，也不会影响用户的使用，所以，好的 UI 对于移动应用程序来讲是非常重要的。下面是对设计出好的移动应用程序 UI 的一些小建议。

① 节约、合理地使用屏幕空间；在屏幕上一次性地显示过多信息会使用户感到厌倦。

② 用户界面的工作流、菜单类型和按钮要始终统一，并且充分考虑设备硬件的行为模式，两者要统一。

③ 要有足够大的触摸模式“点击区域”，并且合理分布。

④ 使用大号且易于阅读的字体，图标尺寸也要尽量大。

⑤ 使用标准控件，例如快速联系人图标、内容提供器和搜索适配器，以达到和其他应用程序的紧密结合。

⑥ 当设计含有较多文本的用户界面时，需要充分考虑本地性。某些语言可能本身就比其他语言要长。

⑦ 尽量减少按键或点击的次数。

⑧ 不要假设某种输入机制(例如特定的按钮或键盘一定存在)能够在所有的设备上使用。

⑨ 尽量在设计每一屏幕的默认用例时，让用户只使用大拇指就能满足需求。某些特殊的用例可能需要其他的按钮、输入法，但默认应该尽量使用大拇指。

⑩ 要为设备合理地调整图片尺寸。不要使用超大的资源和组件，即使是能够进行适当调整，因为它们会消耗宝贵的设备资源，并且载入缓慢。除此之外，还可以考虑移除一些不必要的信息，比如 EXIF 或 IPTC 元数据，并且使用类似 ImageMagick 或 PNGOptimizer 这样的工具。另外，“九格图像”绘制工具也可以帮助优化你的 Android 图像文件。

⑪ 出于“友好”用户界面的考虑，我们需要假设用户在安装应用程序时并没有真正地阅读应用程序许可协议。如果你的应用程序所要进行的操作将会导致用户付费或公开用户私人信息，那么在执行之前可以酌情考虑再次通知用户。

Note

在第 25 章“面向不同设备配置和语言”中，我们讨论了如何设计适用范围更广的应用程序，包括如何协调不同设备的不同屏幕尺寸和分辨率。

27.1.3 设计稳定、灵敏的移动应用程序

虽然移动设备的硬件在过去几年中取得了长足的发展，但开发人员仍然需要在资源受限的条件下工作。用户没有升级手机这种设备的 RAM 的奢望，不过他们可以依靠诸如 SD 卡这样的移动存储设备为媒体存储提供“额外的”空间。提早多花一些时间来设计一个稳定、灵敏的应用程序是很重要的，会在今后带来丰厚的回报。下面是对设计健壮、灵敏的移动应用程序的一些小建议。

① 不要在主 UI 进程中执行大量占用资源的操作，使用异步任务或线程消解过于复杂的操作。

Download at http://www.pin5i.com/

② 使用高效的数据结构和算法，它们将满足应用程序的响应性需求，并获得愉快的用户体验。

③ 谨慎地使用循环（recursion），对于此处的代码应该反复检查并进行性能测试。

④ 时刻记录应用程序所处的状态。Android Activity 栈能很好地支持它，但你仍然需要特别注意不要超越了这些状态。

⑤ 记得保存应用程序状态，因为它随时都有可能被挂起（suspended）或终止。如果你的应用程序被挂起或关闭了，用户是不可能进行查验（按下某个按钮等）的。如果应用程序能够“优美”地返回，那么用户一定不吝赞美。

⑥ 迅速启动、迅速返回。你不能让用户悬着大拇指等待应用程序的启动。相反，你需要在预载入与实际需求的数据之间取得一个巧妙的平衡，因为你的应用程序随时都有可能毫无征兆地被挂起（或关闭）。

⑦ 在执行长时操作时使用进度条告知用户。不过你可以考虑把这些繁重的任务交给服务器来操作，而不是在设备上完成，因为它们可能会显著地削减电池的电量，这一点许多用户是难以接受的。

⑧ 在执行长时操作前确保操作是可能成功的。例如，如果你的应用程序将要下载较大的文件，那么在执行之前可以首先检查网络连接、文件尺寸和可用的空间。

⑨ 尽量减少占用本机存储空间，因为大部分设备的资源都非常有限。如果可以的话，使用外部存储。需要注意的是，SD 卡（最常见的外部存储）可能随时会被用户取出或更换，你的应用程序需要顺畅地处理此类事件。

⑩ 你要知道，通过数据调用内容提供器以及跨越 AIDL 障碍是需要消耗性能的，所以，明智地使用此类操作。

⑪ 确保应用程序的资源消耗情况与目标用户相匹配。游戏玩家应该可以接受具有复杂绘图（graphics-intensive）功能的游戏消耗他们很多的电能，而工作中所使用的应用程序则不能消耗不必要的电池电量，并且要做得足够轻巧以满足那些整天奔波的人们，因为他们往往没有闲暇给设备充电。

Tip

对于 Android 开发人员来说，由 Google Android 开发团队撰写及维护的 Android 开发人员博客（http://android-developers.blogspot.com/）是一个很好的资源。这个网站提供了 Android 平台许多深入的、在普通文档中没有涉及的细节资料。在网站上你可以找到许多提示、技巧、最佳实践操作以及 Android 开发相关话题的链接，比如内存管理（如同上下文管理）、视图优化（避免过深的视图层级结构）以及可以提升 UI 速度的布局技巧等。有经验的 Android 开发人员会经常访问这个网站，并且将其提供的实例和提示应用到他们的项目中。

27.1.4 设计安全的移动应用程序

许多移动应用程序都与设备本身的核心特性相整合，比如通话和联系人数据库。确保你采取了必要而适当的预防措施来保护应用程序所使用的私有数据，比如姓名、地址和联系人信息。这其中包括在服务器端及网络传输过程中保护用户的个人信息。

Tip

如果你的应用程序需要访问或使用私有数据，特别是用户名、密码或联系人信息，那么最好将最终用户许可协议（End User License Agreement，EULA）和隐私协定（Privacy Policy）包含到应用程序中。

Download at http://www.pin5i.com/

1. 处理设备上的私有数据

首先，尽你所能地限制应用程序存储的私有或敏感数据。不要以纯文本（plain text）的方式存储这类信息，也不要不加以保护地传输这些数据。不要尝试着去“变通”Android 框架（Framework）提供的安全机制。将用户隐私数据保存在应用程序的私有文件夹中，仅对你的应用程序开放，不要向操作系统开放共享。在没有取得其他应用程序许可的前提下，不要将应用程序数据暴露在内容提供器中，在必要的时候，可以使用 Android 框架中的加密类。

2. 通过网络传输私有数据

上述原则同样适用于任何远程网络数据存储（例如服务器端程序或云存储）以及网络传输。确保你的应用程序所依赖的所有服务器均具有针对身份盗窃（identity theft）和隐私入侵（invasion of privacy）的安全保障措施。把你应用程序所依赖的任何服务器当做应用程序的其他部分来对待——全面地对它们进行测试。必须对所有传输的隐私数据使用诸如 SSL 这样的典型安全保障机制。当使用 Android 备份服务时，以上规则也同样适用。

27.1.5 在设计移动应用程序时考虑利益的最大化

从收费的方式和收入的产生来看，移动应用程序通常属于以下四类中的一类：

① 免费应用程序（包括以广告赚取收益的）；

② 单付费（一次性付清费用）；

③ 订阅费用（周期性地付费，常见于生产和服务类应用程序）；

④ 内容点播收费（为特定内容付费，如铃音、游戏装备或是升级包）。

应用程序可以使用多种类型的收费机制，这取决于它们所处的市场和所使用的收费（billing）API（例如，Android Market 指定 Google Checkout 作为收费途径）。虽然目前为止 Android 框架中还没有收费专用 API，不过有传言称未来版本中会加入。对于 Android 而言，可以使用由第三方提供的付费方法，所以技术上来说，限制是不存在的。

在设计移动应用程序时，你需要仔细考虑用来向用户收取费用的屏幕功能区域，并将其纳入整个应用程序的设计中。另外，你还需要考虑应用程序特定工作流区域内的交易完整性，以便顺利地完成付费操作。例如，如果你的应用程序可以向设备传输数据，那么一定要确立一个自然而然的交易过程，这样，一旦你决定对这项功能收取费用，收费功能可以顺利地展开。除此之外，当用户付款后，数据传输能够顺利进行，否则，整个交易过程能够毫无障碍地回退。

> **Note**
>
> 你将在第 29 章“销售你的 Android 应用程序”中学习到更多目前可用的销售应用程序的方法。

27.1.6 在 Android 应用程序设计时使用第三方标准

目前没有针对 Android 应用程序的证书（certification）。不过，随着应用程序数量的增长，第三方可能会推出自己的标准，以区分应用程序质量的优劣。例如，移动软件市场可以对软件的质量提出自己的要求，某些应用程序可能会得到一些业内知名人士的认可与肯定。

Download at http://www.pin5i.com/

Warning

Android 市场能够进行自我管理。不过，千万不要把“Android 只有很少的规则”错误地理解为“没有规则”。有些很严厉的许可条款能够保护用户不受到恶意软件或恶意代码的侵害。

分析其他移动平台下的认证程序是非常有用的，Android 可以对其稍作调整。你也可以分析桌面应用程序和其他平台的认证程序，并且考虑如何将它们应用到 Android 当中。例如，如果应用程序需要某种加密机制，而 Android 恰恰提供这一功能，那么你就可以考虑在应用程序当中使用它。

27.1.7 设计便于维护和升级的移动应用程序

一般而言，在开发应用程序时，对设备的配置做越少的假设越好。当你需要移植应用程序或提供简单的升级时，就会尝到这样做的甜头。而那些你不得不做出的假设一定要是深思熟虑后的结果。

1. 使用网络诊断

除了提供足够的文档和便于理解的代码之外，你还可以使用一些技巧来维护和监视移动应用程序。大部分的技巧只适用于包含应用程序服务器的移动应用程序，但有时你也可以从第三方获取数据报告，例如来自移动软件市场的应用程序销售图表，你也可以在应用程序中加入收集反馈的功能。开发人员还可以通过设备上的程序崩溃时发送的错误日志收集数据。

对于网络应用程序而言，在应用程序的服务器端引入一些轻量级的统计（auditing）、日志和报表功能会非常有用，这样可以使你自己拥有一些统计数据，而非完全依赖于第三方。例如，你可以很方便地跟踪以下数据：

① 有多少用户第一次运行你的应用程序？
② 有多少用户习惯性地使用你的应用程序？
③ 最常见的使用模式和倾向是什么？
④ 最不常见的使用模式和特性是什么？
⑤ 哪款设备（由应用程序版本或其他相关数据决定）最流行？

通常上面的这些数据都可以大致地解读为对销售预期的估计，而产生的数据可以同来自第三方市场的实际销售数据进行比较。最常见的使用模式可能是最流畅而高效的方式，而最不常用的特性可能需要重新进行考量和改进。有时你甚至可以找出潜在的 bug，比如通过发现某些特性完全没有人使用过，则它有可能根本就无法使用。除此之外，你还可以确定你的应用程序运行在哪款设备上是最合适的，或者说哪款设备对于现有的用户基础而言是最合适的。

Tip

绝对不要在不告知用户的情况下收集用户的个人数据。在这类问题中，进行匿名的诊断统计是很常见的，不过要注意避免保留那些会被认为是隐私的数据。另外，你还需要确保你的样本尺寸足够大，不至于显现出个体细节，还要将动态的 QA 测试数据从结果中移除（特别是在考虑销售数据时）。

Download at http://www.pin5i.com/

2. 设计易于更新和升级的应用程序

Android 应用程序都能够很方便地进行升级，虽然移动应用程序的更新和升级的确给开发人员带来了一定的挑战。更新是指修改 Android manifest 版本信息，并将升级后的版本重新发布到用户的设备上去。升级是指创建一个全新的带有新功能的应用程序包，将其作为一个独立的可选应用程序让用户选择是否安装。

从更新的角度来说，你需要考虑哪种情况下需要进行更新。例如，程序崩溃时或出现功能性需求时。你还需要考虑发布更新信息的频率，你要确保更新的频率足够高，以真正满足用户需求，但是又不能过于频繁，因为更新应用程序会对用户体验造成影响。

Tip

你应该将应用程序的内容更新纳入其功能特性（通常需要网络驱动）当中，而非像应用程序在线升级一样必需。通过启用在线内容更新，用户对你应用程序的满意度将会随之提升。

在考虑升级时，你需要决定以什么样的方式使用户从旧的版本升级为新的。你是否需要使用 Android 备份功能，以便用户能够无缝地将应用程序从一台设备上转移到另一台？或者提供自己的解决方案？你还需要考虑在有可用版本更新时如何提示用户。

27.1.8 在应用程序设计中使用 Andriod 工具

Android SDK 和开发人员社区中提供了许多有用的工具和资源，它们可以用来帮助你设计应用程序。在你的项目开发过程中，可能会使用到下面这些工具。

① Android 模拟器是在你拥有特定设备之前快速验证想法的好地方。你可以使用不同的 AVD 配置模拟不同配置的硬件设备和开发平台版本。

② DDMS 工具在用于内存分析时非常有用。

③ 层级阅览器（Hierarchy View）的精细像素（Pixel Perfect）视图可以帮助你实现精确的用户界面设计。

④ 九格绘图（Draw Nine Patch）工具可以为移动应用程序创建可拉伸的图像。

⑤ 实际的物理设备（如果能够获得的话）。如果可以的话，尽量使用实际的物理设备进行可行性研究和概念验证的工作，不要仅依赖模拟器进行设计。

⑥ 设备的技术规范可以从制造商和运营商处获得，这些信息在确定目标设备的配置细节时非常宝贵。

27.1.9 避免设计时的愚蠢错误

下面，我们列出一些 Android 设计人员应该尽力避免的愚蠢错误，虽然我们把它放在最后，但其重要性却不可被忽视。

① 经过了数月的设计或开发，却没有在真实设备上进行任何可行性测试。

② 只在特定设备上、使用单一语言、针对唯一运营商进行设计。

③ 在不考虑存储空间和处理器性能限制的情况下进行设计，并且总是引入庞大的资源。

④ 在错误的 Android SDK 版本下进行开发（需要检验设备 SDK 版本）。

⑤ 在确知了设备屏幕尺寸后才考虑让应用程序适应小屏幕。

⑥ 在部署应用程序时添加了过大的图像或媒体资源，而未对其进行适当的裁剪。

Download at http://www.pin5i.com/

27.2 “防弹”移动应用程序开发最佳实践

正如我们前面所讨论过的，开发移动应用程序与进行传统的桌面应用程序开发之间并没有太大的差别。但是，开发人员也许会发现移动应用程序的开发会受到更多的限制，特别是资源的约束。与前面相似，我们仍然以针对移动应用程序开发的一些建议开始本节。

① 提早进行可行性测试，并且经常在目标设备上进行。
② 尽量减小应用程序的尺寸，并且提高其效率。
③ 选择适用于移动应用程序的高效数据结构和算法。
④ 执行成熟、稳健的内存管理。
⑤ 假定设备主要运行在电池供电的情况下。

27.2.1 设计有效的移动开发过程

一个成功项目的支柱在于设计精良的软件开发过程。它能够带来标准而良好的沟通和交流，同时能够降低风险。我们在第 26 章中讨论了移动开发的总体过程，这里，我们为进行成功的项目开发提供几点建议。

① 使用迭代式开发过程。
② 结合完善的版本机制使用合理、可持续的构建过程。
③ 针对范围变更同各方进行沟通，这些变更往往会给测试造成重大影响。

27.2.2 提早和经常地测试应用程序可行性

我们不厌其烦地重复这句话：一定要在真实设备上对开发人员的设想进行测试。当你花费了数月时间设计和开发的应用程序在设备上使用时却需要从头来过的时候，你会觉得没有比这更糟糕的事情了。你的应用程序虽然可以顺利地运行在模拟器上，但却不能保证一定能在设备上正常使用。需要仔细进行验证的功能包括以下几个方面：

① 需要同外设和设备硬件进行交互的功能；
② 网速和延迟；
③ 内存占用和使用情况；
④ 算法效率；
⑤ 用户界面对于小屏幕的适应性；
⑥ 与必需的输入方式相比时的设备输入限制；
⑦ 文件尺寸和存储耗用。

有时我们一遍又一遍地看到了同样的错误在发生。当不能获取目标设备时，项目开发会变得异常脆弱。其原因在于开发人员使用“瀑布”方式进行着软件开发，而在使用了数周甚至数月的模拟器之后，才为其犯下的巨大而严重的错误感到震惊。

我们不需要再重复“瀑布”方式的危险性，你也不能对此放松警惕。把它想象成飞机起飞前最后的安全确认。

27.2.3 使用代码标准、代码检查和单元测试以提高代码质量

花费了必要的时间和精力开发出高效的移动应用程序的开发人员始终能够从用户那里获

Download at http://www.pin5i.com/

得回报。下面是一些具有代表性的建议，它们可以帮助你朝着适当的方向努力。

① 将核心的功能集中在共享的 Java 包中（或者，如果你有共享 C 或 C++库，考虑使用 Android NDK）。

② 针对与目标设备兼容的特定的 Android SDK 版本进行开发。

③ 使用合适的内置控件和 Widget，只在需要时进行自定义，以维持代码的简短。

可以使用系统设置来继承重要的设备特性（屏幕类型、语言、日期、时间、输入方式、可用硬件等）。如果你在应用程序中修改了系统设置，那么必要的情况下要在应用程序暂停或退出时将其恢复。

1. 定义代码标准

为开发团队定义一组能够在成员之间实现良好沟通的代码标准有助于理解重要的移动应用程序需求。这些标准应该大致包括如下内容。

① 实现健壮的错误处理机制，并且能够“优雅地”处理这些异常。

② 将冗长而耗时的操作从主 UI 线程中移开。

③ 在不使用对象和资源时将其释放。

④ 实现成熟、稳健的内存管理。内存泄漏会使你的应用程序毫无用处。

⑤ 为今后的本地化考虑，适当地使用资源。不要将字符串和其他资源硬性地定义代码或布局文件中。

⑥ 为了避免引起困惑，注释是很有必要的。同时，你需要在开发过程中可能会出现的困惑和保护软件不被剽窃二者之间中找到平衡点。一般来说，使用一些不容易理解的设置来保存文件名和详细行数这类信息能够让你更容易追踪盗版。

⑦ 考虑使用标准文档生成工具，例如 Javadoc。

⑧ 建立并强制使用命名约定（无论是在代码中还是在数据库模式设计中）。

2. 执行代码检查

执行代码检查可以提高项目代码的质量，帮助遵守代码标准，而且能够在 QA 人员接手并耗费时间和资源进行测试之前找出问题。

它还能够在开发人员和测试相应代码的 QA 人员之间建立联系。如果测试人员能够深入理解应用程序和 Android 操作系统的功能，则可以进行更全面而成功的测试。这可以作为正式的代码检查环节而引入项目的开发过程中，也可以不这样。例如，测试人员可以仅通过表单域的布局或者“Submit”或“Save”按钮的处理函数就能发现一个类型安全（type-safety）的漏洞。

3. 使用代码诊断

Android SDK 提供了一系列与代码诊断相关的包。通过构建针对日志记录、单元测试和诊断信息收集的架构，可以帮助你开发出稳健、高效和有效的移动应用程序，其中，诊断信息的收集包括方法调用的频率和算法的效率等。需要注意的是这些诊断功能的高优先级一般会在应用程序发布后取消，因为它们可能会导致严重的性能下降，从而影响应用程序的响应性。

（1）使用应用程序日志

在第 3 章“编写第一个 Android 应用程序”中，我们讨论了如何使用内建的日志类 `android.util.Log` 来实现各种级别的诊断日志，并且可以在若干个不同的 Android 工具

Download at http://www.pin5i.com/

（例如 LogCat，它在 DDMS、ADB 和 Android Eclipse 插件中均可使用）中监视这些信息。

（2）使用单元测试

单元测试可以帮助开发人员慢慢地向百分之百的代码测试靠近。Android SDK 中包含了可用于测试的扩展 JUnit 框架。自动化测试可以通过创建测试用例来完成，测试用例能够验证你的应用程序确实如你所设计地运行。单元测试、功能测试甚至用户交互测试中都可以使用自动化测试的方式。

基本的 JUnit 支持是通过 junit.framework 和 junit.runner 包提供的。其中你将会发现用于运行单元测试的框架，以及针对独立测试案例（test case）的辅助类。这些测试案例可以结合成“测试套装（test suite）”。另外，它还提供了针对标准断言（assertion）和测试结果推理（logic）的功能类。

这种 Android 特有的单元测试类是 android.test 包中的一部分，其中含有大量专门针对 Android 应用程序而设计的特殊工具。这个包构建在 JUnit 框架的基础之上，并且添加了许多有趣的特性，如下所示。

① 包含 android.test.InstrumentationTestRunner 的简化“测试仪器挂钩（Hooking of Test Instrumentation）”（android.app.Instrumentation），它可以通过 ADB Shell 命令运行。

② 性能测试（android.test.PerformanceTestCase）。

③ 单一 Activity（或 Context）测试（android.test.ActivityUnitTestCase）。

④ 完整应用程序测试（android.test.ApplicationTestCase）。

⑤ 服务测试（android.test.ServiceTestCase）。

⑥ 生成类似触屏事件的工具包（android.test.TouchUtils）。

⑦ 更多专用断言（android.test.MoreAsserts）。

⑧ View 验证（android.test.ViewAsserts）。

如果你对设计和实现单元测试框架感兴趣的话，我们建议你学习我们提供的指南：“Android SDK：使用 JUnit 测试框架进行单元测试”。你可以在我们的博客上找到这篇指南：http://androidbook.blogspot.com/2010/08/unit-testing-with-android-junit.html。

Tip

更多有关 JUnit 的信息，可以查看以下网站：http://www.junit.org，或者相关书籍。

27.2.4 处理某单一设备的缺陷

偶尔地，你可能会遇到需要为某一特定设备提供代码的情况。Google 和 Android 团队告诉你，如果出现了这种情况，你应该马上告知他们这一情况，采用任何手段都可以，因为这是个 bug。然而，这在短期内可能不会对你有帮助。处理只出现在某个单一设备上的 bug 比较麻烦，而你又不希望为其专门编写代码，那么下面这些选择可能会有用。

① 如果可能的话，尽量让客户端具有普适性，使用服务器来提供针对特定设备的服务。

② 如果在客户端的程序中可以确定出现 bug 的条件，尝试着为其设计一种普适的解决方案，以允许开发人员继续进行开发，而不必要派生出新的代码分支。

③ 如果这个设备并不处于较高的优先级，那么在性价比合适的情况下可以考虑放弃它。

④ 如果有必要，为其派生出新的代码分支以解决这个问题。确保你在 Android Manifest 文件中进行了正确的设置，保证这个版本的应用程序只会安装在目标设备上。

Download at http://www.pin5i.com/

⑤ 如果以上所有办法都失败了，则只能将这个问题记录下来，等待bug被解决。在此过程中，保证用户体验不要被中断。

27.2.5 在应用程序开发中使用Android工具

Android SDK和开发人员社区为应用程序开发提供了许多有用的工具和资源。在项目的开发过程中，你可能会用到下面这些工具：

① 带有ADT插件的Eclipse开发环境；

② 用于测试的Android模拟器和物理设备；

③ 用于与模拟器和设备进行交互的Android Dalvik调试监视服务（Dalvik Debug Monitor Service，DDMS）工具；

④ 用于记录日志、调试和实现Shell访问的Android调试桥（Android Debug Bridge，ADB）工具；

⑤ 用于访问应用程序数据库的`sqlite3`命令行工具（通过ADB Shell访问）；

⑥ 用于视图用户界面调试的层级阅览器（Hierarchy Viewer）。

Android SDK中还有许多其他工具，Android帮助文档中都有详细说明。

27.2.6 避免开发时的愚蠢错误

下面是Android开发人员应该避免的一些愚蠢而令人沮丧的错误。

① 忘记将新的应用程序Activity和必要的许可权限添加到`AndroidManifest.xml`文件中。

② 忘记使用`show()`方法显示`Toast`。

③ 将类似网络信息和测试用户信息等硬编码信息加入到应用程序中。

④ 在发布前忘记清除诊断性日志。

⑤ 在开启调试模式的情况下发布动态应用程序。

小 结

出于响应性、稳定性和安全性的考虑，Android开发具有一些原则。在本章中，我们使用来源于经验丰富的移动开发人员所提供的实用知识和技术来“武装”你（无论你是软件设计人员、开发人员还是测试人员）。对于特定的项目，你可以任意选择适用的信息，并且记住，软件的开发过程，特别是移动软件的开发过程，是始终存在上升空间的。

参考资料和更多信息

Android Best Practices（最佳实践）：Designing for Performance：http://developer.android.com/guide/practices/design/performance.html.

Android Best Practices：Designing for Responsiveness：http://developer.android.com/guide/practices/design/responsiveness.html.

Android Best Practices：Designing for Seamlessness：http://developer.android.com/guide/practices/design/seamlessness.html.

Android Best Practices：User Interface Guidelines：http://developer.android.com/guide/practices/ui_guidelines/index.html.

Download at http://www.pin5i.com/

第28章

测试 Android 应用程序

早测试，常测试，真机测试。这是在测试移动应用程序时我们认为最重要的质量保障十字箴言。应用程序测试不需要成为繁重的工作，相反，你可以相对容易地将一些传统应用程序开发中的质量保障技术移植到 Android 移动应用开发中来，例如自动化测试和单元测试。在本章中，我们将讨论一些 Android 应用程序测试中的技巧和窍门。我们还要提醒所有的项目经理、软件开发人员和移动应用测试人员一些需要注意和避开的陷阱。

如同所有的 QA 过程，移动项目开发也能够从漏洞跟踪系统、周期性构建和系统而有计划的测试中受益。另外还有许多使用“白盒”（或“灰盒”）测试的机会，以及一些有限的自动测试功能。

28.1 设计一个应用程序漏洞跟踪系统

大部分的漏洞跟踪系统可以为测试移动应用程序而进行自定义。漏洞跟踪系统必须围绕特定设备漏洞和集中的应用程序服务问题进行跟踪。

1. 记录重要的漏洞信息

一个好的移动应用程序漏洞跟踪系统应该包含特定设备漏洞的如下相关信息。

① 构建（build）的版本信息、语言等。

② 设备配置和状态信息，包括设备类型、固件版本、屏幕方向、网络状态以及运营商信息等。

③ 再现问题的步骤，包括与所使用的输入法（触摸或点击）相关的特定细节。

④ 设备屏幕截图，这可以使用 Android SDK 提供的 DDMS 或层级阅览器（Hierarchy Viewer）工具来实现。

> **Tip**
>
> 为设备上的特定行为定义一组标准化的术语是非常有用的，比如触摸模式中的手势，单击或轻点，长按和按下并保持，清除和回退等。这可以方便操作步骤的重复，对所有参与进来的相关人员而言，表述也更加精确。

2. 为移动应用程序重新定义术语“漏洞”

我们有必要考虑“漏洞”一词的更广阔的含义。漏洞可以发生在所有的设备或某些设备上。应用程序环境的其他部分，例如远程应用程序服务器，也有可能会产生漏洞。移动应用程序中发生的典型漏洞包括如下。

① 程序崩溃和异常的终止。

Download at http://www.pin5i.com/

② 应用程序功能没有正确运行（安装有误）。

③ 消耗了设备上大量的磁盘或内存空间。

④ 没有进行充分的输入检测（通常使用“按钮混乱（button mashing）”的方式）。

⑤ 状态管理的问题（启动、关闭、挂起、返回与关机）。

⑥ 应用程序响应性的问题（启动、关闭、挂起与返回缓慢）。

⑦ 测试异常状态改变（内置状态改变时的异常，如保存过程中的意外中断）。

⑧ 与输入方式、字体尺寸和屏幕空间利用相关的可用性问题。美工问题导致屏幕显示不正常。

⑨ 主 UI 线程暂停或“冻结”（没有很好地实现异步线程操作）。

⑩ 反馈指示符丢失（指示进度失败）。

⑪ 应用程序在与设备上其他应用程序进行交互时出现问题。

⑫ 应用程序导致设备出现某些问题（过度消耗电量、使节能模式失效、滥用网络资源、产生大量费用以及弹出令人讨厌的通知等）。

⑬ 消耗过多内存资源，没有恰当地冻结内存或释放资源，任务结束时没有关闭进程。

⑭ 应用程序不遵守第三方协议，比如 Android SDK 授权许可协议、Google Maps API 条款、市场条款以及任何其他适用于该应用程序的条款。

⑮ 应用程序的客户端或服务器并没有对受保护信息或私有数据进行安全性操作。这包括保障应用程序服务器足够的运行时间以及相关的安全手段确保不被“黑客”攻击等。

28.2 管理测试环境

测试移动应用程序给 QA 人员带来了挑战，特别是在涉及配置管理的问题时。测试的难度往往会被低估。不要因为移动应用程序所拥有的特性比桌面应用程序少就认为它测试起来更简单，这是完全错误的。目前市场上种类繁多的 Android 设备使得配置测试用的不同安装环境显得格外棘手。

Warning

要确保所有在项目范围内的更改都经过了 QA 团队的测试。增添新的设备一般不会对开发进度造成太大的影响，但却会大大改变测试的进度。

1. 管理设备配置

设备型号的分化或许是移动开发人员面临的最大挑战。设备具有各种不同的屏幕尺寸、平台版本和底层硬件，输入方式（比如按钮和触屏）也各式各样，它们还带有各种可选的特性，比如摄像头和增强图形处理支持等。许多 Android 设备是智能手机，但并不是所有设备都是智能手机。跟踪记录所有设备的功能和特性是一项浩大的工程，而这些工作大部分都需要测试团队来完成。

QA 人员必须详细地掌握目标设备的可用功能，包括熟悉它们各自的特性和设备特有的“脾性”。只要条件允许，他们都需要在这些设备上进行测试，而这些设备可能并不是应用程序所默认的目标设备，它们可能具有不同的配置和语言等。这意味着需要改变输入方式、屏幕方向和本地化设置，以及使用电池供电，而不是坐在桌子边连着数据线。

Tip

要特别注意与运营商相关联固件的变更，它们可能会影响你的应用程序在设备上的使用。例如，假设你拿到了一部某个牌子的手机，它具有合适的操作系统版本，并且测试也很顺利。但是，如果某个运营商移除了这部手机上的某些应用程序而替换成了其他的，对

Download at http://www.pin5i.com/

于开发人员来讲，这非常重要。因为就算你的应用程序能够完美地运行在标准形态手机上，也并不意味着大多数用户的手机也具有这些默认的配置。所以你需要在尽量与市场接近的手机设备上测试应用程序。

百分之百完整的测试是不可能的，所以 QA 人员需要仔细地设计优先级别。正如我们在第 26 章中所讨论的，开发一个设备的数据库可以大大降低移动配置管理的困难，它可以帮助确定优先级，并对物理硬件进行跟踪记录，以供测试使用。善用 AVD，当某些设备和情景无法被真实复现时，模拟器也是提供良好扩展性的有力工具。

Tip

如果对于你来讲配置一个接近现实状况的设备有困难的话，可以考虑看一看由一些运营商提供的设备“实验室”。与租借项目不同，开发人员可以访问运营商的在线实验室，并租用特定的“设备”。开发人员可以在这里安装他们的应用程序并且进行测试，对于持续性的测试而言并不理想，但有总比没有好。除此之外，某些实验室可能还聘请了一些专家，他们可以帮助你解决针对特定设备的问题。

2. 确定设备的“洁净”初始状态

目前没有任何一种方式可以将设备进行“镜像”，以便能够一次又一次地返回到某个相同的起点。而 QA 测试团队出于测试的目的，需要定义一种设备的“洁净”状态。这涉及一些卸载过程和手动的清除操作。

Tip

使用 Android SDK 提供的工具，比如 DDMS 和 ADB，可以使开发人员能够访问 Android 文件系统，包括应用程序的 SQLite 数据库。这些工具可以用来监视和操作设备以及模拟器上的数据。例如，测试人员可以使用 `sqlite3` 命令行接口来清除应用程序的数据库，或者为特定的测试需求载入测试数据等。

3. 模仿现实世界的活动

为移动应用程序的测试设置一个完全独立的环境是几乎不可能的（当然对大多数公司来说性价比也会很低）。虽然对于一个网络应用程序来讲，先将其在测试用（模拟）应用程序服务器上进行测试，然后再接入具有相应配置的实时服务器中，是一件很常见的事情，但对于设备配置而言，移动软件的测试人员却需要在提供真实服务的真实设备上进行适当的测试。如果测试设备是手机，它们需要适当地拨打和接听电话，发送和接收文字消息，使用 LBS 服务确定地点，以及做任何正常情况下一部手机可以完成的事情。

对应用程序进行测试远不止确定应用程序能够正常工作那么简单。在现实世界里，你的应用程序并不处于真空中，而是安装在设备上的众多应用程序之一。对移动应用程序的测试还包含确保这个软件能够与设备的其他功能和应用程序良好地融合。例如，假设你要开发一个游戏，测试人员就必须验证游戏过程中有电话呼入的情况，看应用程序是否能够自动地暂停（保持状态）以允许接听电话。

这也意味着测试人员有时必须要在设备上安装另外一些应用程序，最好从那些流行的软件开始。不仅要在安装了这些应用程序的设备上进行测试，还要尝试着同时运行它们，有时

Download at http://www.pin5i.com/

你会从中发现用户情境中的一些协调问题，或是设备其他方面的缺陷。

有些时候测试人员需要具有一定的创造性，比如在需要产生某一类型的事件时。例如，测试人员可以在手机没有服务时对应用程序的行为进行测试，以确保其能正常工作。

> **Tip**
>
> 某些移动平台可能光靠不正确的握机姿势就可以使设备无法接收信号，但 Android 测试人员是需要采取一些非常手段才能做到这一点的。要测试信号丢失，你可以走出去在公路隧道或电梯中测试应用程序，或者干脆将手机放到冰箱里。不过不要在低温的地方放置太长时间，因为这会减损电池寿命。金属罐也是很好的选择，特别是内含饼干的罐子：可以首先吃掉饼干，然后再将手机放进罐子里。

28.3 尽量全面地进行测试

所有的测试团队都力求做到百分之百全面的测试，但大多数会意识到这一目标过于感性或性价比很低（尤其在 Android 设备数量大得惊人的今天）。测试人员必须尽力拓展测试场景，以覆盖更大的范围，有时其“深度”和“广度”令人吃惊，特别是刚刚进入移动领域的人。我们来看看几种特定的测试类型，以及 QA 人员实现它们的办法，某些是被经验所证明的，而另一些则是新奇而具有创意的，这都是为了做到尽量全面。

1. 构建验证与“冒烟”测试

除了常规的构建（build）过程以外，建立“构建许可测试（build acceptance test）”也非常有用，“构建许可测试”也被称作“构建验证（build validationg）”、“冒烟测试（smoke testing）”、“理智测试（sanity testing）”。“构建许可测试”是一种短小但对关键功能具有针对性的测试，它可以在进行完整测试之前对应用程序的构建进行检验。这也是一种快速检验 bug 修正的方法，它可以在新一轮完整测试开始之前进行。另外，还要考虑为多个版本的 Android 平台开发能够同时运行的验收测试。

2. 为“构建许可”引入自动功能测试

移动应用程序的“构建许可测试”通常是在具有最高优先级的目标设备上手工完成的，不过，这也正是进行自动“理智”测试的理想条件。模拟器是一个桌面软件，所以它可以使用传统的 QA 自动化平台，例如 Borland Silk Test（www.borland.com/us/products/silk/silktest/index.html）。通过为模拟器创建主干功能测试，测试团队可以进一步确信这一构建可以进行更深入的测试，同时可以使提交到 QA 人员那里的有问题的构建数量达到最小。

3. 模拟器与设备测试

如果你能够获得用户将会实际使用的设备，那么就使用这个设备进行测试。然而，硬件设备及其伴随的服务合约通常价格不菲，测试团队却不可能针对每个运营商和每个用户所在的国家来设置环境。有时 Android 模拟器可以降低开销并且扩展测试范围。使用模拟器的好处如下所示。

① 可以模拟已退出市场或即将上市的设备。

② 可以在一些实际设备无法实现的复杂条件下进行测试。

③ 可以像其他桌面软件一样使用自动化测试。

Download at http://www.pin5i.com/

（1）在拥有设备前使用模拟器进行测试

开发人员通常会瞄准那些即将上市的设备或平台版本。这些设备通常会拥有比较庞大的用户群，而在其发布的第一天就推出应用程序的开发人员往往会获得销售上的巨大回报，因为此时可用的应用程序还不多，竞争小，利润就大。

最新版 Android SDK 对开发人员发布的时间一般都会比正式对用户公开要早几个月。通常来说，开发人员可以通过运营商和制造商的开发人员项目获得试验设备。不过，开发人员和测试人员应该注意在试验设备上进行测试的风险：这些设备都不是正式发布的。最终的技术规范和固件可能已被更改，而你不会收到任何通知。而且，这些设备的发布日期也可能被推迟，甚至永远不能面世。

如果不能获取试验设备，那么测试人员可以使用模拟器的配置进行一些功能性的测试，这些模拟器的配置需要尽量地与目标平台接近，以降低今后重复测试时的风险，使得应用程序能够早日发布。

（2）使用模拟器的自动测试功能

Android 测试人员有多种不同的测试方法可供选择。当然，装配一个自动测试的软件来实现软件模拟是完全有可能的，许多工具（例如 monkey）都能够支持这种操作。然而，自动化硬件测试却没有那么多方式可供选择。我们都可以想象出这一解决方案：在我们的想象中，这种设备应该类似于美国总统用来在照片和圣诞卡片上签名的自动签名机。问题在于每一个设备无论是外观还是行为都各不相同，这样所有的动作机器人（animatronic）的“手”都需要针对每一种设备进行调校。另一个问题在于如何确定应用程序的失败或成功。所以，使用程序来完成这一测试是非常困难的，也基本上只有移动软件测试咨询公司才会着眼于开发移动软件的自动测试工具，对于普通的移动软件开发人员来讲，是没有能力负担这笔开销的。

（3）理解依赖模拟器的危险性

不幸的是，模拟器仅仅是一个模仿设备内部工作原理的一般 Android“设备”（AVD 配置中涵盖的可选功能除外）。

Tip

考虑为特殊的 AVD 配置建立一个管理测试环境的文档，将不同的设备配置作为测试计划的一部分保存起来。

模拟器并不代表任何特定 Android 平台的实现，因为它对于给定设备而言是唯一的。它并不使用与之相同的内部硬件来确定信号、网络或位置信息。模拟器可以“假装”拨打或接听电话、收发信息、拍摄照片或视频，但是到最后，应用程序有可能可以很好地在模拟器上运行，但却不能运行在实际的设备上。

4. 测试策略：“白盒”测试

Android 为“黑盒”和“白盒”测试都提供了大量工具。

“黑盒”测试的测试人员可能只需要测试设备和文档即可。对于“黑盒”测试而言，更重要的是测试人员拥有特定设备的使用经验，所以为他们提供设备的手册和技术规范可以帮助其进行更完整的测试。除了这些细节之外，知晓设备的细微差异和技术标准对可用性测试具有很大的帮助。比如，如果设备可以座驻（dock），那么知晓其是横屏模式还是竖屏模式就会非常有用。

移动应用程序的“白盒”测试从来都很困难。“白盒”测试人员可以使用各种可用的工具，包括 Eclipse 开发环境，它不仅免费，而且许多 Android SDK 提供的调试工具都可以在其中

Download at http://www.pin5i.com/

运行。“白盒”测试人员可以使用 Android Emulator（模拟器命令行）、DDMS 和 ADB。他们也可以使用强大的单元测试框架，我们在前一章已经详细讨论过了。在这种情况下，测试人员需要在计算机上设置一个与开发人员相似的开发环境。

5. 测试移动应用程序服务器和服务

测试人员通常都将目光集中在应用程序的客户端部分，他们往往会忽略在服务器端进行完整的测试。许多移动应用程序都依赖网络或“云”，并且使用一个应用程序服务器。如果你的应用程序依赖于一个服务器而运行，那么测试应用程序的服务器端是非常重要的。即使这项服务不是你提供的，你也需要仔细测试，确保它会如你所期望的那样提供服务。

Warning

用户会期望应用程序在任何时间都可用，无论白天黑夜，7×24 小时。所以，你需要尽量减少服务器无响应的时间，并且确保在服务器无响应时应该以适当的方式告知用户（应用程序不能崩溃）。如果某项服务在你的掌控范围之外，你可能需要仔细研究一下对方提供的服务等级协议（Service Level Agreement）。

以下是关于测试远程服务器或服务的几条建议。

① 为服务器构建设定版本。应该像管理构建过程中的其他部分一样管理服务器的构建。服务器应该以某种可重复利用的方式实现，并且要进行适当的版本管理。

② 使用测试服务器。通常，QA 测试会在一个可控的模拟服务器的环境下进行。特别是在实际的服务器已经处于运行状态时，这种方式更为普遍。

③ 验证可扩展性。在负载情况下测试服务和服务器，包括压力测试（大量用户，模拟客户端）。

④ 测试服务器的安全性（攻击、SQL 注入等）。

⑤ 确保你的应用程序能够很好地处理远程服务器的维护和中断（编制计划任务或者其他方式）。

⑥ 测试服务器的升级和回退（rollback），确定一套策略，当你的服务不可用时，你应该何时、使用何种方式告知用户。

上面的这些测试为你带来了又一次使用自动测试的机会。

6. 测试应用程序的视觉效果和可用性（Usability）

对移动应用程序的测试不仅包括查找功能上的障碍，还包括对应用程序可用性的评估，比如缺乏视觉效果、出现导航或使用障碍等。在谈及移动应用程序用户界面时，我们喜欢将其比喻为“边走边嚼口香糖”。移动应用程序的用户通常不会全神贯注，而是一边使用应用程序一边做着别的事情。所以，应用程序应该设计得就像嚼口香糖一样简单。

Tip

在进行可用性研究时，可以收集对软件不太熟悉的用户的反馈。只依赖于开发团队的成员，可能会掩盖一些应用程序的缺陷，因为他们对应用程序已经很熟悉了。

7. 为 Android 测试引入第三方标准

养成将传统软件的测试原则引入到移动开发中的习惯，鼓励公司的 QA 人员实践并分享此类经验。

我们再次说明，目前 Android 应用程序无需认证程序；没有什么能够阻止移动应用程序

Download at http://www.pin5i.com/

市场的发展。可以考虑仔细地分析其他移动平台中使用的认证程序，例如苹果 iPhone 和 TRUE BREW 测试中所使用的大量测试脚本，将它们做适当调整，然后应用到 Android 应用程序中来。无论你是否打算使用认证，始终遵守质量准则的确能够有效地提升应用程序的质量。

8. 处理专门的测试场景

除了功能测试以外，还有几个 QA 团队可以使用的专门的测试场景。

（1）测试应用程序融合点

测试应用程序如何同 Android 操作系统的其他部分进行交互是很有必要的。例如以下方面。

① 来自操作系统中断的处理（消息到达、电话呼入、关机等）。

② 验证你的应用程序提供的 Content Provider，包括动态文件夹一类。

③ 通过 `Intent` 验证其他应用程序中的功能触发器。

④ 通过 `Intent` 验证你的应用程序中提供的所有已知功能触发器。

⑤ 验证 `AndroidManifest.xml` 文件中定义的应用程序的次入口点，例如应用程序快捷方式。

⑥ 验证应用程序的其他形式，例如应用程序小插件。

⑦ 验证与服务相关的特性（如果提供服务的话）。

（2）测试升级

如果可以的话，应在客户端和服务器端同时进行升级测试。如果在应用程序中使用了升级功能，那么可以开发一个模拟的升级 Android 应用程序，这样 QA 人员可以验证数据的迁移是否正确，即使这个升级的应用程序并不使用这些数据。

Tip

用户通常都会通过无线的方式获取 Android 平台的最新更新。你的应用程序安装的平台版本随时有可能改变。某些开发人员可能已经发现了固件升级有时会损坏应用程序，致使开发人员不得不对应用程序进行升级。当新的 SDK 版本发布之后，一定要重新测试应用程序，这样才能在用户手上的应用程序崩溃前将其升级。

（3）测试产品的国际化（Internationalization）

在开发过程的早期进行国际化测试是一个很好的想法，当然这个测试既包含客户端也包含服务器。在这个方面，你有可能会遇到有关屏幕占用、字符串、日期和格式的问题。

Tip

如果你的应用程序会使用若干种语言进行本地化，那么你需要在外语下进行测试，特别是对于较长的语言。比如，应用程序可能在英语下拥有完美的显示，但若使用较长的德语，则有可能无法使用。

（4）测试合宜性（Conformance）

确保你查看过应用程序所必须满足的政策、协议和条款，并且确保应用程序遵守了这些条文。例如，Android 应用程序默认必须遵守 Android 开发人员协议（Android Developer Agreement）和 Google Maps 服务条款（如果要使用地图的话）。

（5）安装测试

一般而言，安装 Android 应用程序是很简单的。不过，你需要在资源短缺和内存匮乏的设备上进行安装测试，并且当你的应用程序可以从市场获得时，尝试从那里下载并安装。如果 Manifest

Download at http://www.pin5i.com/

的安装路径允许引用外部媒体资源，确保各种低速或不可用资源的场景也纳入了测试范围。

（6）备份测试

不要遗漏了那些对用户而言不那么显而易见的功能，例如备份和恢复功能，还包括我们在第 23 章“用户账户管理与数据同步”中讨论过的同步功能。

（7）性能测试

移动应用程序的性能非常重要。Android SDK 中提供了对计算应用程序性能的支持，并且提供了用于监视内存和资源耗用的工具。QA 团队需要熟悉这些工具并经常使用它们，以帮助找出性能瓶颈、危险的内存泄漏和资源滥用。

（8）测试应用程序付费功能

付费功能是绝不能凭主观臆断的，一定要对其进行测试。在 Android 市场上你会发现很多测试应用程序。记得说明你的应用程序是一个测试程序。

（9）测试无预见性事件（Unexpected）

尽管你设计了工作流，但是你要理解用户会随机地进行操作，这是无法预料的，无论是故意的还是不小心的。有些用户特别钟爱按钮，而有些则是忘记锁上键盘锁就把手机塞进了口袋里，这都会导致“诡异”的按键，有时候甚至会拨出一个电话或发出一条短信。你的应用程序必须要足够健壮，以应付这类情况。前面讲过的 Monkey 命令行工具非常适合用来进行这类事件的测试。

（10）“杀手级”应用程序测试

每一个移动开发人员都想开发出“杀手级”的应用程序——那些像吃了兴奋剂一样在排行榜上以火箭般速度上升的应用程序，一个月就能获得上百万的回报。大多数开发人员认为，只要他们有了正确的想法，就已经在手里攥着一个“杀手级”应用程序了。他们总是在搜寻列表的前十位，尝试着开发出下一个“杀手”。但让我告诉你一个秘密：如果所有这些“杀手”应用程序都有一个共同点的话，则一定是高于平均水平的质量。绝对没有缓慢、难于使用或使人不愉快的应用程序登上排行榜。测试并硬性使用质量标准可以将一个中等的应用程序升级为“杀手”应用程序。

如果你花时间查看移动应用程序市场的话，将会发现许多由较大的移动软件开发公司发布的各类应用程序，它们都是具有共同外观和感受的高品质应用程序。这些公司在各种产品中保持用户界面的一致性，且高于平均水平，以建立品牌的认同感和忠实度，同时扩大市场份额，一旦有某一个产品创造了奇迹，那么其他的产品也能从中获利。相反，一些小公司通常拥有很好的想法，但却始终在移动软件开发的质量方面进行努力，其最终的结果就是一些拥有奇思妙想的应用程序却运行在丑陋的用户界面或严重的漏洞下。

28.4 在应用程序测试中使用 Android 工具

Android SDK 和开发人员社区提供了一系列用于应用程序测试和质量保障的工具及资源。在项目开发的过程中，你可以使用这些工具：

① 用于测试和 bug 检测的物理设备；

② 在没有实际设备可用的情况下，用于自动测试和构建测试的 Android 模拟器；

③ 用于调试、操作模拟器和设备以及截屏的 Android DDMS；

④ 用于记录日志、调试和实现 Shell 访问的 Android ADB；

⑤ 用于进行应用程序输入压力测试的 Exerciser Monkey 命令行工具（通过 ADB Shell 访问）；

⑥ 用于访问应用程序数据库的 `sqlite3` 命令行工具（通过 ADB Shell 访问）；

Download at http://www.pin5i.com/

⑦ 用于用户界面导航和验证的层级阅览器（Hierarchy Viewer），它还可以用于屏幕截图；

⑧ 带有 ADT 和“白盒”测试所需的相关日志和调试工具的 Eclipse 开发环境。

虽然我们在 Eclipse 中使用诸如 Android 模拟器和 DDMS 调试工具这样的 Android 工具，但应该注意的是，它们也可以作为单独的工具供 QA 人员使用，而无需源代码或开发环境。

28.5 避免测试时的愚蠢错误

下面是 Android 测试人员应该尽力避免的一些愚蠢而令人沮丧的错误：

① 没有像测试应用程序客户端一样完整地测试服务器；

② 没有使用适当的 Android SDK 版本进行测试；

③ 没有在实际设备上进行充分的测试，而假定模拟器就足够了；

④ 没有使用用户实际使用的操作系统版本即时性地测试应用程序（付费、安装等）；

⑤ 忽略了测试应用程序的所有入口点；

⑥ 忽略了在不同的网络状况下测试应用程序；

⑦ 忽略了只在电池供电的情况下进行测试。不要始终将设备连接着电源。

28.6 外包测试的责任

移动应用程序的质量保障可以外包。不过需要记住的是， 外包 QA 任务成功与否取决于你所提供的文档的质量和细节。但是，QA 人员和开发人员之间关系的密切程度关系到测试的完整性和综合性，而外包对此不利。

小　结

在本章中，我们使用来源于经验丰富的移动开发人员所提供的实用知识和技术来“武装”你——应用程序质量的保卫者，不管你的团队只有你一个人还是有上百人，测试应用程序是项目成功的关键。幸运地，Android SDK 提供了许多用于测试应用程序的工具和强大的单元测试框架。通过遵守 QA 技术标准，并且使用这些工具，你一定能够开发出最佳的应用程序。

参考资料和更多信息

Android Dev Guide（开发指南）：Testing & Instrumentation：http://developer.android.com/guide/topics/testing/testing_android.html.

Android Dev Guide：Testing：http://developer.android.com/guide/developing/testing/index.htm.

Android Tools（Android 工具）：UI/Application Exerciser Monkey：http://developer.android.com/guide/developing/tools/monkey.html.

Wikipedia 软件测试：http://en.wikipedia.org/wiki/Software_testing.

Software Testing Help：http://www.softwaretestinghelp.com.

Download at http://www.pin5i.com/

第29章

销售你的 Android 应用程序

在你完成一个 Android 应用程序的开发之后，下一步应该就是发布它，以便更多的人可以使用你的工作成果，甚至欣赏它，并借此赚取一些外快。开发人员可以通过若干种途径利用 Android 应用程序获得收入，许多人会选择向诸如 Android Market 这样的移动市场提交应用程序，也有人会发展一些个人的分发机制，例如将其发布在某个网站上。无论如何，程序员在程序处于设计和开发程序的初期就应该考虑好使用何种发布方式，因为某些发布方式可能会对代码有一定的要求，或是对内容有一定的限制。

29.1 选择合适的发布模式

你选用的应用程序发布方式应该取决于你想要达成的效果以及你的目标用户群。首先，问问自己以下几个问题。

① 这个程序已经是最终版本了吗？抑或你还预留了一段时间的测试期解决其中的问题？

② 你是想让更多的人使用你的应用程序呢，还是想开发一个垂直市场应用程序？确定你的用户是哪一类人，他们使用哪些设备，以及他们习惯于使用哪种方式寻找和下载应用程序。

③ 你要如何为你的应用程序定价？它是个免费软件吗？你期望的付费方式（一次性付费或支付订金或广告驱动式）是否为被所使用的发布机制支持？

④ 你愿意从你的收入中抽出一定比例用于支付手续费吗？Android Market 这类发布机制一般都会从每笔交易中收取一定比例的费用，作为为你的应用程序提供发布平台并聚合人气的报酬。

⑤ 你是否想要完全控制发布过程？或是你愿意遵守第三方市场的强制要求和界限？这些要求中可能会包括遵守后续协议的承诺。

⑥ 如果你打算独立发布应用程序，你想怎么做？你可能需要开发更多服务，用于管理用户、部署应用程序以及收费。如果选择这种方式，你如何保护用户数据？你应该遵守哪些法律条款？

⑦ 你是否考虑过为你的应用程序创建一个试用版本？如果你将要选用的发布系统中包含有返还政策，你可以考虑使用分支发布的方式，你需要创建一个“门槛”，将试用后购买的用户数量最大化。例如，一个游戏的试用版可能很简单，完整版包含更多的关卡，用户需要在试用后的一段特定时长内购买完整版才能解锁，这就是合适的“门槛”的作用。

接下来，我们看看打包并发布你的应用程序需要哪些步骤。

29.2 为发布而打包你的应用程序

在准备发布一个 Android 应用程序之前，有几项工作是开发人员必须完成的。你的应用

Download at http://www.pin5i.com/

程序同样需要满足几个市场需求。下面的步骤是发布一个应用程序所必需的。

① 为应用程序创建用于发布的版本。

② 确认应用程序满足正确地配置 Android Manifest 文件这类发布市场的规定。例如，确保应用程序的名称和版本信息没有错误、调试属性被设置为否。

③ 为应用程序打包并进行数字签名。

④ 仔细检查准备发布的应用程序包。

⑤ 发布应用程序。

以上的步骤很重要，但仅仅如此并不能确保部署一定成功。开发人员还应该进行以下步骤。

① 在所有目标设备上仔细地测试应用程序。

② 关闭编译选项，包括 Log 语句和其他所有的日志记录。

③ 对许可权限进行验证，确保已经添加了服务所需使用的许可权限，并移除其他不需要的，而无论它们是不是被设备所强制要求。

④ 在所有调试和日志选项都关闭了的情况下再一次测试最终已签名的版本。

接下来，我们按照上面的顺序，详细地探索每一个步骤的细节。

29.2.1 准备封装你的代码

一个经过了完整测试周期的应用程序在准备进行产品发布之前需要做一些调整，这些调整会使其由一个可编译的、试验性的应用程序转变为成熟的、可发布的软件。

1. 设置应用程序名称及图标

每一个 Android 应用程序都会有默认的图标和标签设置。图标会出现在应用程序 Launcher 页面或者其他许多地方，包括市场中。既然如此，每一个应用程序都应该拥有自己的图标。你需要为分辨率不同的屏幕分别准备可选的图标图像资源。标签（也就是应用程序的名称）会显示在与图标相似的地方，其默认名称为包名称。你应该为你的应用程序赋予一个简洁好记、表意明晰的名称。

2. 为应用程序指定版本号

下一步，我们需要一个合适的版本号，尤其是在考虑到未来很有可能会进行升级的情况下。版本号是由开发人员自行指定的，但是它同时会被 Android 操作系统用来作为后台判断一个应用程序是否是最新版本的依据。应用程序每更新一次，版本号都应该相应地增加一点。增加的具体数值无关紧要，但至少要比上一次发布的版本大一些。在第 5 章“使用 Android Manifest 文件定义应用程序”中我们曾经详细地讨论过关于 Manifest 文件中的版本号的问题。

3. 验证平台版本

确保你的程序在 Android Manifest 文件中正确地设置了<uses-sdk>标记。这个标记用于指定应用程序能够运行的目标平台版本号和最低版本号。除了应用程序名称和版本信息外，这可能是整个应用程序中最重要的设置了。

4. 为 Android Market 准备你的应用程序包

Android Market 对软件包有着严格的要求。当你上传应用程序到 Android Market 网站，软件包会被验证，如果发现任何问题，网站都会与你确认。很多时候，出现问题的原因都是

Download at http://www.pin5i.com/

因为没有对 Android Manifest 文件进行合适的配置。

Android Market 通过 Android Manifest 文件中<manifest>标记的 android:versionName 属性来获知应用程序的版本信息，并将其显示给用户。另外，它还在后台使用 android:versionName 来控制应用程序的升级情况。android:icon 和 android:label 属性也需要进行设置，它们分别对应应用程序的名称和图标，Android Market 会将其显示给用户。

Warning

Android SDK 允许 android:versionName 属性引用字符串资源，但 Android Market 却不允许。如果使用了字符串资源则会产生一个错误。

5. 关闭调试和日志选项

接下来，我们需要关闭调试和日志选项。关闭调试选项需要将 AndroidManifest.xml 文件中<application>标签下的 android:debuggable 移除，或是将其置为 false。在 Java 中关闭日志选项可以有很多种不同的途径，既可以简单地将其注释掉，也可以使用构建系统来自动完成这一操作。

Tip

对调试代码进行条件编译的最常用方法为使用一个带有 public、static 和 final 的布尔值。当使用 if 表达式且设置为 false 时，由于它是不可能被修改的，所以编译器并不会将这段永远也执行不了的代码包含进去。我们推荐使用除了将 Log 行注释掉以外的其他方法，即使你可能不太习惯。

6. 验证应用程序的许可

最后，我们应该验证应用程序所使用的许可权限，包括确认应用程序申请的所有许可，移除那些没有必要的许可，否则，用户可能会感到疑惑。

29.2.2 为你的应用程序打包并签名

既然应用程序已经做好了发布的准备，那么就需准备文件包（.apk 文件）了。Android 设备的包管理器不会安装一个未经过数字签名的软件包。在软件开发的整个过程中，Android 工具通过使用调试密钥（key）进行签名解决了这一问题，但是调试密钥不能用于应用程序的发布。相反，应用程序需要一个真正的可以用于数字签名的密钥。你可以使用私钥对你的 Android 应用程序发布包以及它的每一次升级进行数字签名，这保证了整个应用程序（作为一个整体）是你授权的，不是任何其他来源。

Warning

私钥是开发人员的“身份证”，是建立开发人员和用户之间关系的重要纽带，保证私钥信息安全是十分重要的。

Android Market 要求你数字签名的有效期在 2033 年 10 月 22 之后。这个时间看起来遥遥无期，当然，对于手机来说的确也是如此，然而，原因在于应用程序必须使用相同的密钥来进行升级，而且，一些需要在特殊权限下密切合作的应用程序也必须使用同一个密钥，一个密钥可以串联多个应用程序。所以，Google 规定密钥必须有足够长的有效期，以保证更新和升级的流畅。

Download at http://www.pin5i.com/

Note

找到一个第三方的认证授权且又能够保证有效期足够长的密钥是很不容易的，所以自签名是最直接的解决方案。实际上，在 Android Market 中，使用第三方的认证授权也没有太大好处。

虽然自签名对于 Android 应用程序来讲非常普遍，而且无需官方证书。但是，创建一个适当的密钥并好好保管它还是非常重要的。Android 应用程序的数字签名可能会对一些功能造成影响。在安装时将对签名的时效进行验证，但一旦安装成功，即使签名过期，应用程序也可以继续使用。

除此之外，对应用程序的升级或更新需要使用相同的密钥，否则安装将不能进行。

你可以使用 Eclipse 中的 Android 开发插件导出你的 Android 软件包并对其签名，或者你也可以使用命令行工具。使用 Eclipse 导出软件包并签名的步骤如下。

① 在 Eclipse 中，右键单击你的应用程序项目，选择“导出（Export）”选项。

② 在“导出”菜单中展开 Android 面板，选择“导出 Android 应用程序（Export Android Application）”。

③ 单击“下一步”按钮。

④ 选择需要导出的项目（默认是你刚才打开的项目）。

⑤ 在选择密钥库页面，选中“新建密钥库（Create New Keystore）”选项，输入文件路径（你想要保存密钥库文件的位置）和管理密钥库使用的密码。（如果你已经有了一个密钥库，单击“浏览”按钮选择你的密钥库文件，然后输入正确的密码。）

Warning

确保已为密钥库选用了一个足够强壮的密码，同时谨记密钥库所在的位置，因为发布应用程序升级包时你需要用到同一个密钥库。如果密钥库已经在版本控制系统核查过，密码可以帮助你保护它，不过你需要增加一个权限优先层来使用这一功能。

⑥ 单击“下一步”按钮。

⑦ 在“创建密钥（Key Creation）”页面，输入密钥详情，如图 29.1 所示。

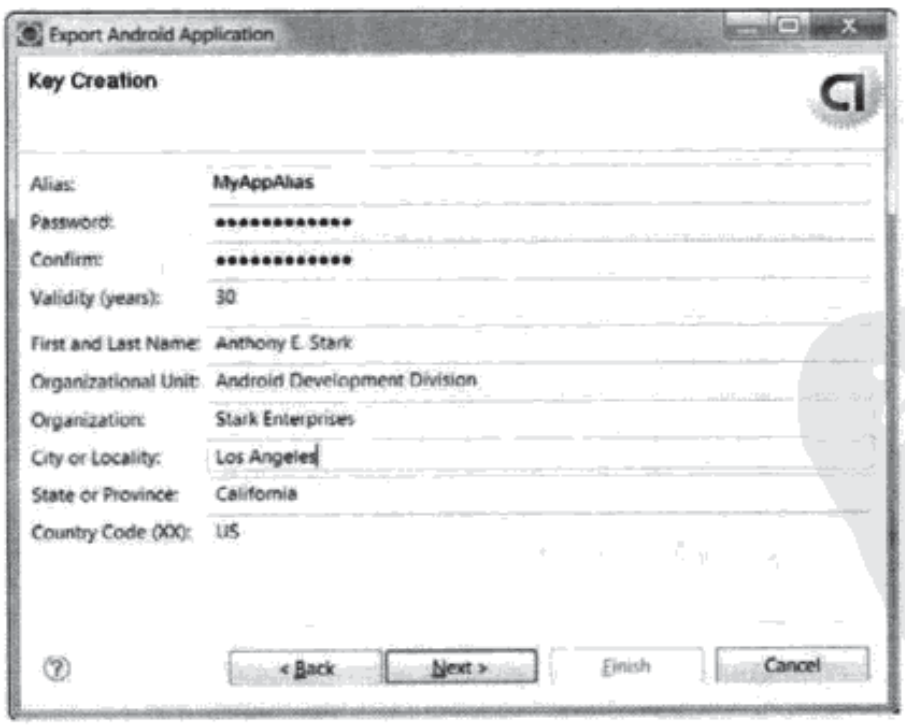


图 29.1 在 Eclipse 中导出 Android 应用程序并对其进行签名

⑧ 单击“下一步”按钮。

⑨ 在“目标和密码/证书检查（Destination and Key/Certification Check）”页面，为应用程序包文件键入目标地址。

Download at http://www.pin5i.com/

现在，你已经创建了一个完全签名的、通过了验证的应用程序包文件了，应用程序包已经准备好可以发布了。

Note

如果你没有使用 Eclipse 和 Android 开发插件，除了 Android SDK 中提供的 `zipalign` 之外，你还可以使用 JDK 中的 `keytool` 和 `jarsigner` 命令行工具来创建合适的密钥并用对应用程序包文件（.apk）进行签名。虽然 `zipalign` 工具并不直接和签名相关，但它将 Android 中的应用程序包进行了优化。Eclipse 中的 ADT 插件能够在签名步骤完成后自动运行 `zipalign`。

29.2.3 测试软件包的发布版本

既然你已经为产品发布配置好了应用程序，那么应该执行一次完整的测试，其间特别注意安装过程的微小变化。这个过程的重要部分就是检验是否已经关闭了所有的调试特性，并且确保日志记录不会对应用程序的功能和性能造成负面影响。

29.2.4 认证你的 Android 应用程序

如果你熟悉其他的移动平台，那么可能会熟悉它们所采用的许多严格的认证程序，包括 TRUE BREW 和 Symbian Signed 程序。这些程序对应用程序的质量做了一定的限制。

就目前而言，Android 并没有任何认证或测试需求。它是一个开放的市场，只包含一点对内容的限制和需要遵守的规则。但这并不意味着就完全不需要认证，也并不表示所有的发布方式都不需要进行认证。

一般情况下，认证程序需要进行严格而完整的测试，必须满足某些可用性约定以及遵守其他一些称为“最佳实践”所需的约束，同时，还有可能需要满足运营商的特定需求。开发能够通过认证的应用程序的最好办法是在设计之初就将这些需求纳入到项目的设计中。不过，遵循 Android 开发的“最佳实践”原则，开发出高效、可用、动态、健壮的应用程序终究能够获得收益（无论你的应用程序是否需要认证）。

29.3 发布你的应用程序

现在你已经完成了发布你的应用程序所需的全部准备工作了，是时候让它与广大用户见面了——在为他们带去乐趣的同时为你带来收入！和其他移动平台不同，大部分 Android 的发布机制可以同时支持免费发布和收费计划。

29.3.1 在 Android Market 上销售你的应用程序

目前，Android Market 是发布 Android 应用程序的主要渠道。通常用户将会从这里购买和下载你的应用程序。到目前为止，每一个 Android 手机都支持 Android Market。下面我们将介绍如何检查你准备好的软件包，如何在 Android Market 上注册开发人员账号，以及如何将你准备出售的软件提交至 Android Market。

Note

Android Market 能够提供许可服务。它是一种 Google API 附加组件，但是仅在 Android 1.5 及以上版本中可用，仅可用于付费程序，仅可用于通过 Android Market 发布的应用程

Download at http://www.pin5i.com/

序。它需要应用程序提供对代码添加的完全支持，所以你需要仔细考虑一下如何混淆你的代码。此项服务旨在确认安装在设备上的应用程序是用户正确付费获取的。以下网页中有更多关于此项服务的说明：http://developer.android.com/guide/publishing/licensing.html。如果想要了解如何创建一个不易被破解的程序，可访问 http://j.mp/9HoP4t and http://j.mp/9A0cRv，Dan Galpin 有一篇关于混淆器、Android 和许可服务器的文章（以下网址可以访问：http://android-developers.blogspot.com/2010/09/proguard-android-and-licensing-server.html），其中提供了一些使用混淆器混淆代码的实用建议。

1. 在 Android Market 上注册一个开发人员账号

想要在 Android Market 上发布应用程序，你需要首先注册一个开发人员账号。这一举动有两个意义：第一，向 Google 证实了你的身份；第二，在 Google 上注册了一个支取（Checkout）账户，你可以用它来收取别人购买你应用程序应付的款项。

Note

截至此刻，只有居住于以下国家和地区的开发人员（“商人”）能够在 Android Market 上销售标价的应用程序：阿根廷、澳大利亚、奥地利、比利时、巴西、加拿大、丹麦、芬兰、法国、德国、中国香港、爱尔兰、以色列、意大利、日本、墨西哥、荷兰、新西兰、挪威、葡萄牙、俄罗斯、新加坡、南非、韩国、西班牙、瑞士、中国台湾、英国和美国（参考：http://market.android.com/support/bin/answer.py?hl=en&answer=150324）。来自于其他国家和地区的开发人员可以注册发布账号，但是暂时只能发布免费的应用程序。如果想要查看完整的支持发布的国家和地区列表，访问 http://market.android.com/support/bin/answer.py?hl=en&answer=136758。以上列表可能会随时调整。

注册 Android Market 的步骤如下。

① 打开 Android Market 注册页面：http://market.android.com/publish/signup，如图 29.2 所示。

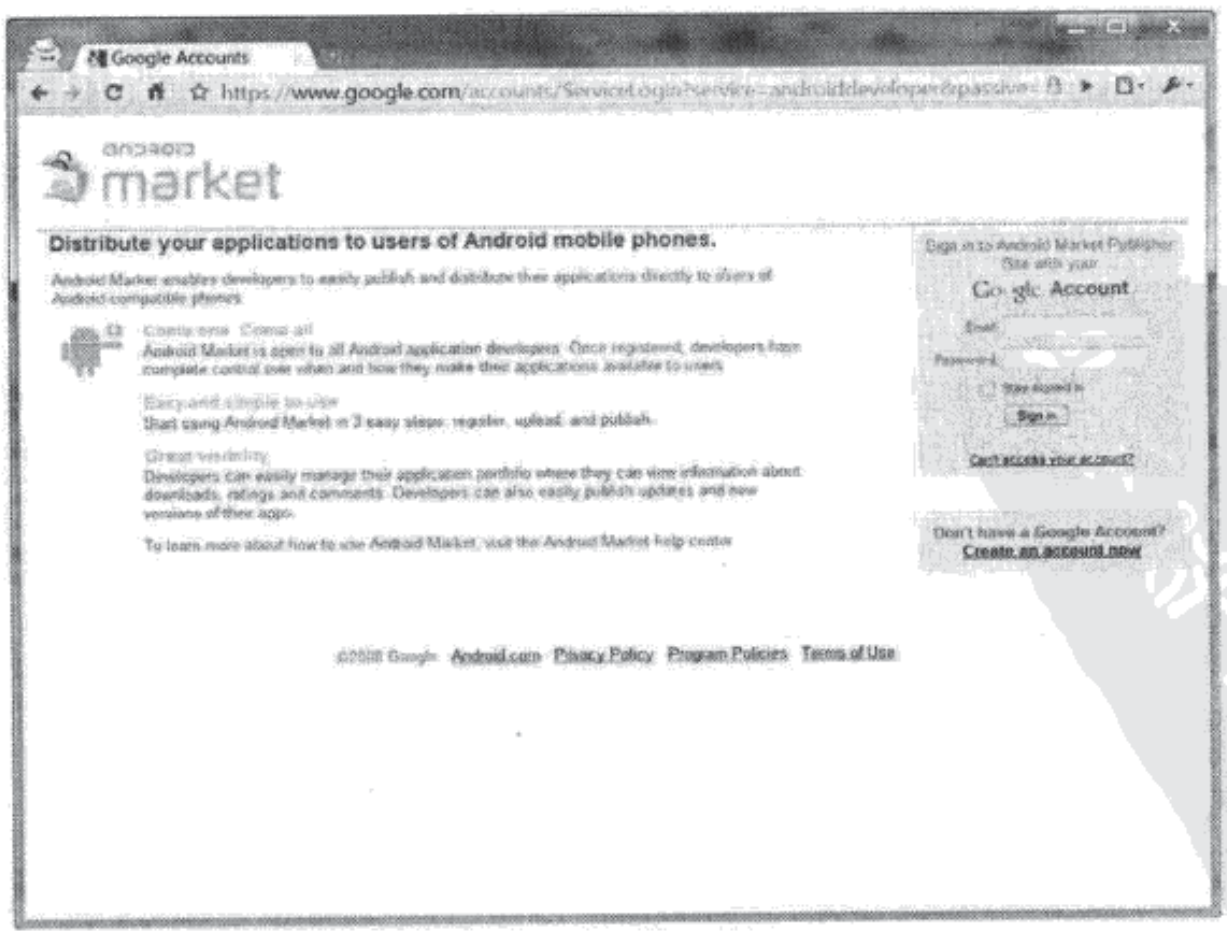


图 29.2 Android Market 注册页面

Download at http://www.pin5i.com/

② 注册你想用的 Google 账户。（目前为止，账户在注册后是不能更改的，但是用于联系的 E-mail 地址可以单独更改。）

③ 填写开发人员的基本信息，比如开发人员姓名、电子邮件地址、个人主页等，如图 29.3 所示。

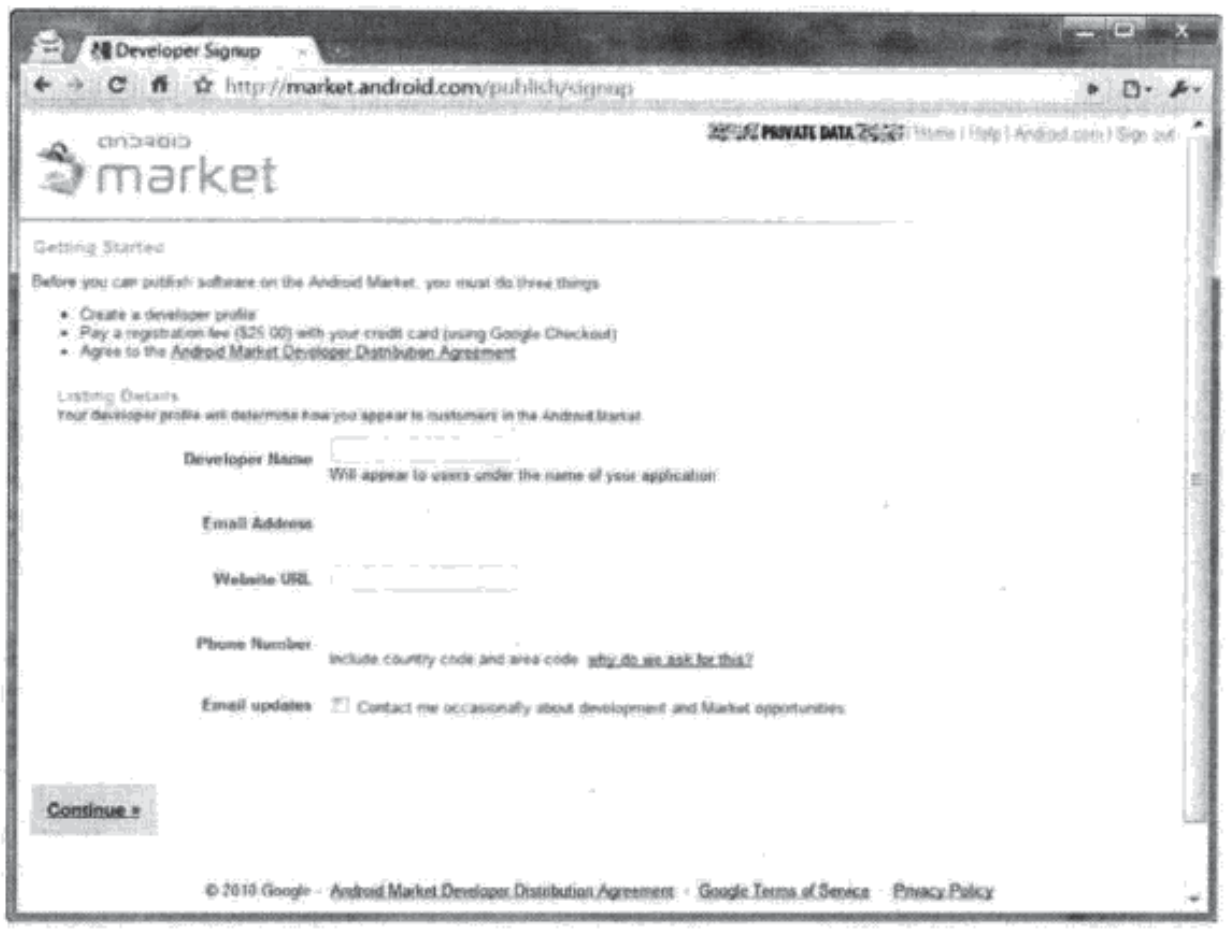


图 29.3 Android Market 发布人员资料页面

④ 确认缴纳注册费用（目前为止是 25 美元）。需要注意的是，在缴纳注册费的过程中会用到 Google Checkout。

⑤ 注册及支付成为一个 Android 开发人员的同时会创建一个必需的 Google Checkout Merchant 账号，你也需要为这个账号提供所需的信息。这个账号用于处理支付。

⑥ 在《Android Market 开发人员发布协议（Developer Distribution Agreement）》中同意绑定你的信用卡，并且确认你注册的账号。美国地区版的基础协议在以下网址可以访问：http://www.android.com/us/developer-distribution-agreement.html。最好能够把每一次你签署同意的协议打印下来，作为注册流程的一部分，因为它随时有可能改变。

当成功完成以上步骤后，页面将会跳转至 Android Market 主页，该页面会确认你创建的 Google Checkout 账号。

2. 将你的应用程序上传至 Android Market

现在，你拥有了一个可以用来上传应用程序的 Android Market 开发人员账号，以及一个已经完成签名的软件包，上传的所有准备工作已经就绪。打开 Android Market 主页（http://market.android.com/publish）并用你的开发人员账户信息登录，之后，你将会看到显示你的账户信息的页面，如图 29.4 所示。

在这个页面中，你可以配置开发人员账户设置，查看支付交易历史详单，以及管理你发布的应用程序。想要发布新的应用程序，只需要单击此页面的“上传应用程序（Upload Application）”按钮即可，页面会弹出一个用于上传应用程序的表格，如图 29.5 所示。

以下信息是你必须填写的。

Download at http://www.pin5i.com/

图 29.4 Android Market 开发人员应用程序列表

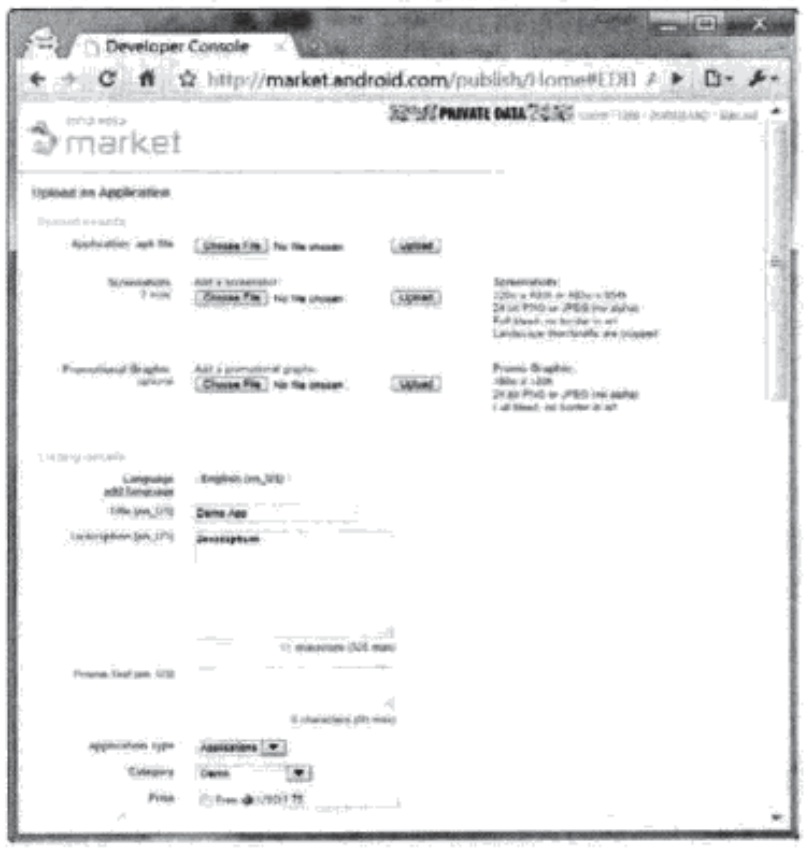


图 29.5 Android Market 应用程序上传页面

① 应用程序软件包名称（.apk）。

② 用于促销的程序截图和用于 Market 列表的图像。

③ 若干种不同语言的应用程序名称和描述。

④ 应用程序类型和类别。

Warning

花一些时间为你应用程序的所属领域做一个精确定位，这样你的应用程序才能被呈现在真正需要的顾客面前，分错类的应用程序一般很难有好的销量。Android Market 帮助列表中包含有完整的类型和类别清单：http://market.android.com/support/bin/answer.py?hl=en&answer=113475。

⑤ 应用程序价格——免费还是付费（后期不可更改）。

Note

需要注意的是，Android Market 目前会向每一个在其中流通的应用程序收取交易金额的 30%作为托管费。应用程序价格可浮动于$0.99～$200，其他种类货币也有类似的价格区间。目前为止仅支持单次付费模式。

Download at http://www.pin5i.com/

⑥ 版权保护信息——此选项能够阻止应用程序在未经授权的情况下在终端之间传输或发布，由于 Google 新增了许可证服务，此选项可能会在近期内有变动。

⑦ 目标发行区域——选择将会发布此应用程序的国家和地区。

Note

发布地区会涉及不同国家的法律问题，所以请慎重选择你要发布的地区。目前为止，有 50 个可选国家和地区，并正不断更新。此外，你还可以为每一个位置选择特定的运营商，以进一步限制应用程序的流通。你可以为你的应用程序选择特定的发行地区，也可以简单地选择全部现有以及未来会添加的地区。不是所有的地区都支持付费购买应用程序，有些地区的市场上甚至都还没有出现 Android 手机。不过，一般而言，对于新添加的国家和地区来说，免费软件是最早进入的，随着费用机制的完善，付费软件也会逐渐进入。Android Market 的帮助页面给出了一些国家和地区的列表，获得了资质的卖家可以在其中发布免费或付费的应用程序：http://market.android.com/support/bin/answer.py?hl=en&answer=138294。

⑧ 联系信息——默认为你在开发人员账号中提供的信息，不过你也可以为每一个应用程序设定不同的内容，这为发布多个应用程序的开发人员提供了许多便利。

⑨ 同意协议——不管你的国籍是什么，在发布之前都必须要勾选 Android 内容准则（Android Content Guideline）前的复选框，表示你同意其中的内容以及美国的出口法律。

Tip

应用程序包成功上传之后，可以将以上信息保存成草稿，以便在最终发布之前进行复核。同样地，应用程序图标、名称、版本信息、本地化信息以及需要的许可协议都会显示出来，你可以再检查一次以确保你的信息符合 Android Manifest 文件规范。

3. 在 Android Market 上发布你的应用程序

现在，你已经为按下“Publish”按钮做好了充分的准备。你的应用程序几乎会立即出现在 Android Market 中。在发布以后，包括评分等级、评价、下载数量等统计信息都会显示在开发人员账号主页的“你的 Android Market 列表（Your Android Market Listings）”部分中。这些统计信息并不会即时进行更新，也不能直接查看评价的详情。单击应用程序列表可允许你编辑一些字段。

4. 理解 Android Market 的退款政策

Android Market 支持 24 小时退款政策，虽然这点有些争议。也就是说，在 24 小时之内，可以将应用程序退回，并且用户能够收到等额的退款。对于开发人员来讲，这意味着在开始的 48 小时之内，销售并没有最终达成。不过，这一政策只适用于第一次下载和第一次退回。如果某个用户已经退还了你的应用程序，并且想要再进行一次尝试，那么就需要进行最终付款，这次就不能再退回了。虽然它对滥用这一政策进行了限制，但是你还是需要注意，如果你的应用程序没有再次使用的必要，或者它的价值仅在于开始的几个小时（甚至更短）之内，那么你应用程序的退回率可能会高于你的想象。

5. 从 Android Market 上移除你的应用程序

你还可以通过撤销发布操作将你的应用程序从 Market 和开发人员账号中移除。撤销发布操作是立即执行的，但如果查看或下载过这个应用程序的话，它在 Market 应用程序中的列表

Download at http://www.pin5i.com/

项可能仍然缓冲在手机上。

6. 开发人员账号的其他好处

除了在 Android Market 中管理你的应用程序之外，拥有注册 Android 开发人员账号的好处还在于可以为 Android 开发版本的手机付费。这些手机对于一般的开发和测试而言很有用，但可能不太适合用于最终的目标手机测试，因为它的某些功能可能受到了限制，并且固件版本也可能与消费者所使用的不同。

29.3.2 在自己的服务器上销售你的应用程序

你可以通过一个网站或服务器直接发布 Android 应用程序。这最适用于垂直市场应用程序、开发移动市场的公司以及品牌网站，它们可能想要用户使用具有其品牌的 Android 应用程序。这也是从最终用户那里获得反馈的一种很好的途径。

虽然自行发布可能是应用程序发布中最简便的方式，但其市场化也是最不容易的，并且很难对应用程序提供保护，获取利润也比较困难。自行发布应用程序唯一所需要的就是找个地方存放你的应用程序包文件。

自行发布处于下降趋势的原因在于最终用户必须对其设备进行配置，以允许来自未知源的应用程序包。在设备的“Settings”应用程序中的“Application”部分有对于这个选项的设置，如图 29.6 所示。这一选项并不是所有客户端设备都适用的，最值得一提的是，AT&T 在美国市场发行的 Android 设备只能安装来自于 Android Market 中的应用程序，第三方资源是不被允许的。

图 29.6 勾选了“Unknown sources（未知源）”的“Settings”应用程序

在此之后，用户需要进行的最后一步就是在手机的 Web 浏览器中输入应用程序包的 URL 并下载这一文件（或者点击它的链接）。下载完成后，将开始标准的 Android 安装过程，询问用户是否同意所有的许可权限，并且如果在已有旧版本应用程序的情况下，还将向用户确认对现有应用程序的更新或替换。

29.3.3 销售应用程序的其他途径

Android Market 并不是销售应用程序的统一场所。Android 是一个开放的平台，这意味着任何一个手机制造商或运营商（甚至是你）都可以运行一个 Android 市场网站，或者开发另一个提供市场功能的 Android 应用程序。许多以移动为主体的商店，如 Handango 等，都有将 Android 应用程序纳入到它们的销售范围。

下面是一些其他可供选择的 Android 应用程序发布场所。

- **Handango：**它使用各种付费模型发布各种手机设备的移动应用程序（www.handango.com）。
- **SlideME：**它是一个针对 Android 的发布社区，可以使用手机设备发布免费或商业的应用程序（http://slideme.org）。
- **AndAppStore：**它是一个针对 Android 的发布网站，可以使用手机设备发布免费的应用程序（www.andappstore.com）。
- **MobiHand：**它发布各种设备的免费或商业移动应用程序（www.mobihand.com）。

Download at http://www.pin5i.com/

这个列表并不完整，我们也没有在背后与这些网站有任何合约。我们认为告诉开发人员发布应用程序可以有很多种选择是非常重要的。各个商店对于应用程序的要求不同，第三方应用程序商店可以自由地提出自己的需求，所以你需要仔细地阅读其中的条文，它们可能会对内容有所要求、需要额外的技术支持或者强制数字签名等。只有你和你的团队才能确定哪一个最适合你们。

Tip

另一个获取大范围发布途径的方法是与设备生产商或移动运营商合作，他们通常会挑选一些应用程序预装在设备中，你可以寻找一些特殊的开发项目，帮助你和生产商、运营商等环节建立合作关系，将你的应用程序发布出去。

29.3.4 保护你的知识产权

你已经花了许多时间、金钱和精力来构建一个有价值的 Android 应用程序。现在你想要发布它，但也许会担心一些反向制造技术（reverse engineering）泄漏你的商业机密，或者担心软件的版权问题。随着技术的发展，上面两种情况都是不可能完全避免的。

如果你习惯于开发 Java 应用程序，那么你可能会熟悉“代码混乱（code obfuscation）”工具。它们被设计用来将易于阅读的信息从已编译的 Java 字节码中剔除，使得反编译后的应用程序更加难以理解。但是对于 Android 而言，应用程序均针对 Dalvik 虚拟机进行编译。这样，现有的 Java 工具应该不能直接发挥作用，而需要一些更新。某些工具，比如 ProGuard（http://proguard.sourceforge.net），可以支持 Android 应用程序，因为它在 `jar` 文件创建之后、被 Android 转换成最终的包文件之前运行。

Android Market 支持某种形式的拷贝保护，这在你发布应用程序时可以勾选。目前，它使用的方法并没有在文档中予以说明。不过，如果对于你或你的公司而言很重要的话，也可以使用自己的拷贝保护方法或者其他市场所提供的方法。

29.3.5 让用户付费

不同于你可能会使用的其他移动平台，Android 目前并不提供内建的付费 API，也就是说不能直接在应用程序当中进行付款或者直接从用户的手机付费。相反，Android Market 使用 Google Checkout 来实现付费的过程。当应用程序获得了付款后，用户将可以获得这笔费用。（虽然它们使用了 24 小时退款机制。）

1. 可再生（Recurring）付费和特定内容付费

如果你的应用程序需要收取服务费或收取应用程序中其他对象的费用（比如铃音、音乐、电子书等），那么应用程序开发人员必须开发自定义的付费机制。大多数的 Android 设备可以使用因特网，所以使用在线付费服务和 API 是比较普遍的选择，比如 Paypal、Google、Amazon 等均如此。你需要验证你所选择的付费服务确实可以为移动服务提供支持。

2. 通过广告获取收益

从用户那里获取收益的另一种途径是使用支持广告的移动商务模型。这是在应用程序中使用的一种相对比较新的模型，因为许多早期发布的应用程序都不支持它。不过，Android 对于在应用程序中引入广告并没有特殊的规定。这一点也不奇怪，看看 Google 的 AdSense 就知道了。

Download at http://www.pin5i.com/

小　结

你已经学会了如何设计、开发、测试和部署专业级别的 Android 应用程序。在最后这一章中，你还学习了如何针对发布进行应用程序安装包的准备，以及使用一系列的付费模型来赚取利润。无论你选择通过自己的网站发布，还是通过 Android Market，或者使用许多其他的途径（或者将它们结合起来），现在你都可以从头开始构建一个健壮的应用程序，来获取收入或名望。

所以现在你完全合上书本，启动 Eclipse，打造一个令人兴奋的应用程序。我们鼓励你跳出思维定式，因为 Android 平台比起其他大部分平台来说，都为开发人员提供了更多的灵活和自由。取长补短、去糟取精，你终究能够开发出一个“杀手级”的应用程序。

如果你确实已经急不可耐、跃跃欲试，那么我们期待着你开发出的令人耳目一新的应用程序。你可以在本书的开头部分找到我们的联系信息和博客地址。

最后，祝你好运！

参考资料和更多信息

The Android Market Website：http://market.android.com/.

Android Dev Guide（开发指南）：Market Filters：http://developer.android.com/guide/appendix/market-filters.html.

Download at http://www.pin5i.com/

Download at http://www.pin5i.com/

第七部分

附录

附录 A　Android 模拟器快速入门

附录 B　Android DDMS 快速入门

附录 C　Android 调试桥快速入门

附录 D　Eclipse 技巧和诀窍

附录 E　SQLite 快速入门

Download at http://www.pin5i.com/

附录 A

Android 模拟器快速入门

模拟器是 Android 软件开发包（Software Development Kit，SDK）提供的最有用的工具。开发人员可以在模拟器上快速地开发出 Android 应用程序进而开发能够运行在各种硬件设备上的强大应用程序。这里提供的“快速入门”并不是模拟器命令的完整文档，而是提供给你用于快速入门并执行常规的任务。如果你需要获取完整的特性、启动命令和功能列表，可以参阅 Android SDK 提供的模拟器文档。

Android 模拟器通过 Android Eclipse 开发工具插件（Android Development Tools Plug-in）被整合到了 Eclipse 集成开发环境（Integrated Development Environment，IDE）中。模拟器位于 Android SDK 中的/Tools 目录下，它也可以在单独的线程中运行。

A.1 模拟器的目的：虚拟现实

Android 模拟器（如图 A.1 所示）虚拟了一个能够运行你的应用程序的真实设备环境。作为开发人员，你可以将模拟器配置为与你的应用程序部署目标设备相似的工具。

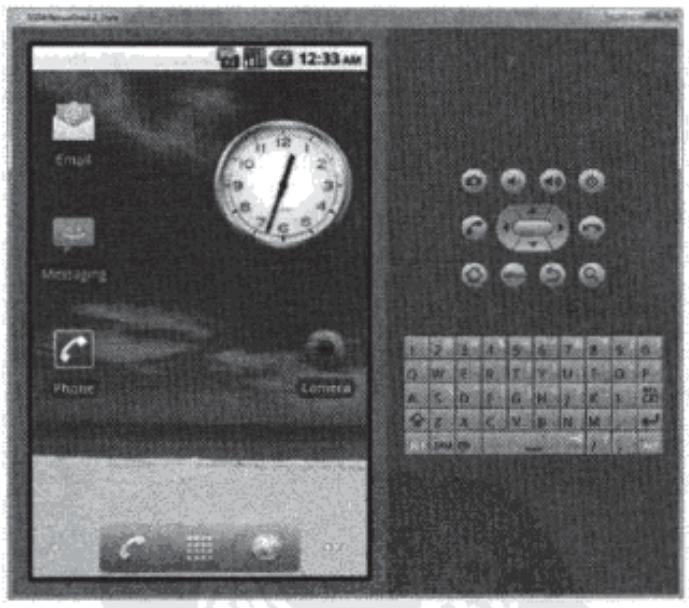


图 A.1 典型的 Android 模拟器界面

以下是几点从零开始使用模拟器的建议。

① 你可以简单地使用键盘命令来控制模拟器。

② 可以使用鼠标进行单击、滚动或者拖曳，也可以使用方向键完成这些动作。不要忘记侧边按钮，例如音量控制，它们都是可以操作的。

③ 如果你有 Internet 连接，那么模拟器也可以连接到互联网，浏览器可以使用。你可以使用 F8 键开启或关闭网络功能。

④ 不同版本平台的运行方式可能会有微小的区别，用户体验层面是感受不到的（Android 操作系统最底层的运行方式）。例如，老版本的平台使用应用程序抽屉来存储安装好的应用程序，而新平台使用滑动控件（sleeker control）和新的主页。模拟器使用的是“Google 体验版”界面设计，界面会经常被制造商和运营商覆写。换句话说，操作系统功能可能和用户实际使用的不一样。

⑤ “设置（Settings）”应用程序对于管理系统设置非常有用。你可以使用“Settings”应用程序来设置无线控制、通话、声音与显示（铃声、亮度等）、数据同步、安全性和位置、应用程序管理、SD 卡和手机存储、日期和时间、文本输入以及“关于”信息（手机状态、软

Download at http://www.pin5i.com/

件版本等）。“开发工具（Dev Tools）”应用程序可以用来设置开发选项。

⑥ 可以使用计算机数字键盘上的“7”键和“9”键（或 Ctrl+F11、Ctrl+F12 组合键）来切换模拟器的“肖像”和“风景”模式。

⑦“菜单（Menu）”按钮可以为给定的界面提供上下文菜单，替代了传统的位于屏幕底部的标签，例如“添加（Add）”、“编辑（Edit）”和“删除（Delete）”等。

⑧ 应用程序生存周期控制：要暂停应用程序只需要按下“Home”；返回应用程序只需要再次运行它，它会从上次离开的地方开始，除非在内存不足的情况下应用程序在后台被终止了。

⑨ 诸如新信息到达这样的通知将会显示在主页顶部的通知栏中，其中还会显示电池电量等。你可以将状态栏拉下来以查看消息。

Warning

Android 模拟器是很有用的工具，但它不能成为测试实际目标设备的替代品。

A.2 使用 Android 虚拟设备（AVD）

Android 模拟器并不是一个真实的物理设备，只是一个为测试目的而开发的一般性的 Android 系统模拟器。开发人员可以通过创建 Android 虚拟设备（Android Virtual Devices，AVD）来模拟特定的 Android 设备。Android 模拟器可以通过 AVD 配置模拟的内容包括：

① 不同的目标平台版本；

② 不同的屏幕尺寸和分辨率；

③ 不同的输入法；

④ 不同的底层硬件配置；

⑤ 不同的外部存储配置。

每一个模拟器的配置都是唯一的，它由 AVD 描述并且持久地存储在模拟的 SD 卡中。图 A.2 显示了具有不同 AVD 配置的模拟器。

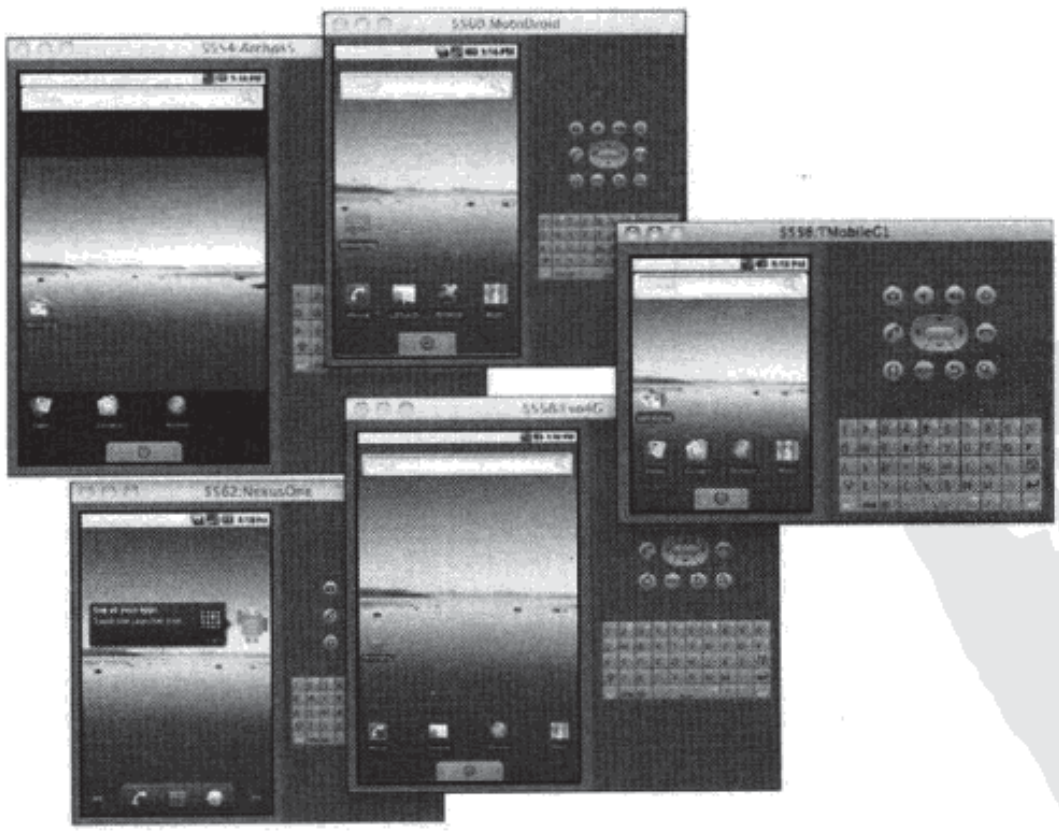


图 A.2 不同模拟器设置中的 AVD 配置

Tip

你可以把 AVD 想象成是为模拟器提供“人格”，没有 AVD 的话，模拟器就是一个空壳子。

Download at http://www.pin5i.com/

A.2.1 使用 Android SDK 和 AVD 管理器

要在 Android 模拟器中运行应用程序，你需要配置一个 Android 虚拟设备（Android Virtual Device，AVD）。你可以使用 Android SDK 和 Eclipse 中的 AVD 管理器（作为 ADT 插件的一部分）或者使用 Android SDK /tools 目录下的 Android 命令行工具来创建和管理 AVD。

> **Note**
>
> 在本书中，我们假设你使用 Eclipse 开发环境以及 Android SDK 和 AVD 管理器来管理 AVD 配置。不过，如果你使用的并不是 Eclipse 及其相关插件的话，可以通过命令行的方式使用 `emulator` 工具，创建和管理 AVD 的 `android` 工具，以及 `mksdcard` 命令行工具来创建虚拟 SD 卡镜像。你可以在命令行下运行这些工具，或者是通过 Android 网站获取更多的帮助：http://developer.android.com/guide/developing/tools/。

不管你使用何种工具以何种方式运行 AVD，每一个 AVD 都应该包含下面的信息，用于描述这个特定的 Android 设备：

① 为每一个 AVD 配置取一个友好而直观的名称；

② 目标 Android 操作系统；

③ 硬件设备支持的细节和特性，包括 RAM、输入法以及可选硬件等，可选硬件包括摄像头和位置感应器等；

④ 模拟外部存储（虚拟 SD 卡）。

图 A.3 展示了如何使用 Android SDK 和 AVD 管理器创建和管理 AVD 配置。

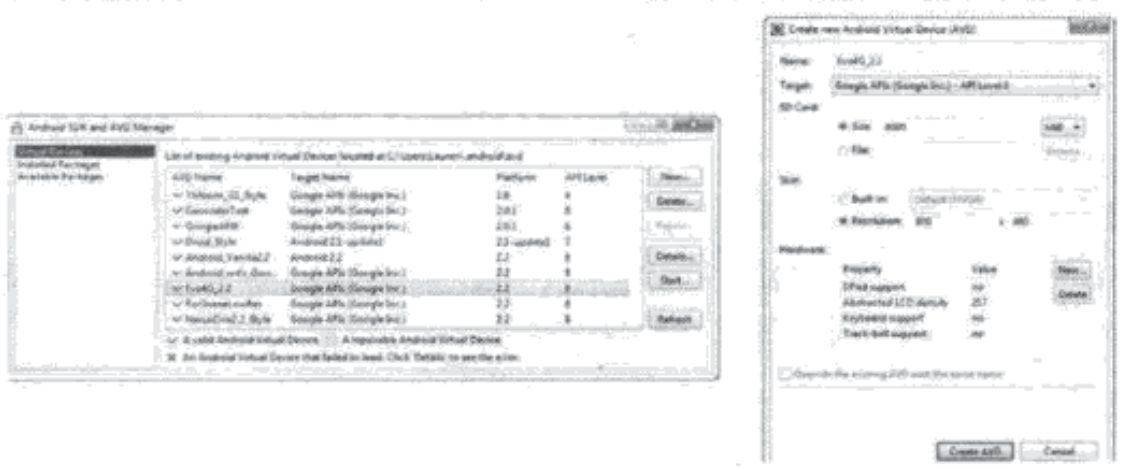


图 A.3 Android SDK 和 AVD 管理器（左）可以用来创建 AVD 配置（右）

A.2.2 创建 AVD

在 Eclipse 中创建新的 AVD 配置遵循以下几个步骤。

① 单击 Eclipse 工具栏上带有箭头的小 Android 图标（），或者执行 Eclipse 菜单命令 Window→Android SDK and AVD Manager 运行 Android SDK 和 AVD 管理器。

② 单击左侧菜单中的“虚拟设备（Virtual Device）”菜单项（如图 A.3（左）），会出现不同配置的 AVD 列表。

③ 单击“新建（New）”按钮创建一个新的 AVD（如图 A.3（右））

④ 为 AVD 命名。如果你是在模拟某个特定的设备，你可以直接使用设备名称。例如，“NexusOne2.2_Style”描述的可能是一部 Nexus One 手机，其上运行着 Android 2.2 平台，并且带有 Google API。

⑤ 选择一个构建目标。它表示运行在模拟器上的 Android 平台的版本，只不过都以 API

Download at http://www.pin5i.com/

Level 的形式表示。例如，Android 2.2 为 API Level 8。另外，在这里你还可以选择是否包含可选的 Google API。如果你的应用程序依赖于 Maps 应用程序或者其他的 Google Android 服务，那么你应该选择带有 Google API 的构建目标。

⑥ 选择 SD 卡容量，容量可配置为千字节或兆字节。

Tip

每一个 SD 卡镜像都会占用硬盘上的一定空间，并且需要一定的时间生成，所以不要把 SD 卡的容量设得过大，否则将浪费大量硬盘空间。1024MB 或者更小一点比较合适，最小为 9MB。

⑦ 选择一种皮肤，它将决定模拟器的屏幕特点。对于每一个目标平台而言，都有一系列预定义的皮肤（HVGA 等），它们代表了一般 Android 设备的特点。如果所有的皮肤都不能满足你的需求的话，还可以使用自己的屏幕设置。

⑧ 配置或修改任何不与默认值相匹配的硬件特征（稍后将讨论更多有关这一话题的内容）。有时预定义的皮肤会自动地为你设置好一些特征，比如屏幕分辨率等。

⑨ 单击“创建 AVD”按钮，等待配置完成。

⑩ 单击“完成（Finish）”按钮。

Note

由于 AVD 管理器在创建 AVD 时会为虚拟 SD 卡分配内存空间，所以创建一个 AVD 配置会需要一段时间。

1. 探索 AVD 皮肤选项

不同的 Android 设备拥有不同的屏幕特征。在模拟器中测试应用程序是否能够在适当的屏幕尺寸和分辨率下正常运行是非常重要的。让我们来看看一些在 AVD 配置中提供的预定义皮肤。表 A.1 给出了这些皮肤的相关信息。另外，你还可以创建自定义的皮肤以供模拟器模拟其他的硬件设备，如 Android Tablets（平板电脑）。

表 A.1 导入默认模拟器皮肤选项

皮肤名称	皮肤描述
HVGA	320 × 480 屏幕 像素密度：160
QVGA	240 × 320 屏幕 像素密度：120
WQVGA400	240 × 400 屏幕 像素密度：120
WQVGA432	240 × 432 屏幕 屏幕密度：120
WVGA800	480 × 800 屏幕 屏幕密度：240 最大虚拟机堆（Heap）空间：24
WVGA854	480 × 854 屏幕 像素密度：240 最大虚拟机堆（Heap）空间：24

Download at http://www.pin5i.com/

2. 使用传统硬件配置创建 AVD

正如我们之前所提到的，你可以在 AVD 配置中指定特定的硬件配置参数。如果想要修改这些参数的话，你需要知道默认的设置是什么。表 A.2 中给出了可用的硬件选项。

表 A.2 硬件选项

硬件选项	说明	默认值
设备物理内存大小 hw.ramSize	设备的物理 RAM。单位为 MB	96
触摸屏支持 hw.touchScreen	设备上是否存在触屏	是
轨迹球支持 hw.trackBall	设备上是否存在轨迹球	是
键盘支持 hw.keyboard	设备上是否存在 QWERTY 键盘	是
方向键盘支持 hw.dPad	设备上是否存在方向键盘	是
GSM 调制解调器支持 hw.gsmModem	设备上是否存在 GSM 调制解调器。默认为 yes	是
hw.camera	设备上是否存在摄像头	否 注意：这项设置并不起任何作用，摄像头始终都存在。事实上，运行 Android 2.0 及以后版本的设备都拥有摄像头
摄像头像素（水平） hw.camera.maxHorizontalPixels	摄像头最大水平尺寸	640
摄像头像素（垂直） hw.camera.maxVerticalPixels	摄像头最大垂直尺寸	480
GPS 支持 hw.gps	设备上是否存在 GPS	是
电池支持 hw.battery	设备是否可以在电池上运行	是
加速度感应器支持 hw.accelerometer	设备上是否存在加速度感应器	是
音频录制支持 hw.audioInput	设备是否可以录制音频	是
视频播放支持 hw.audioOutput	设备是否可以播放音频	是
SD 卡支持 Hw.sdCard	设备是否支持可移除的 SD 卡	是
缓存分区支持 disk.cachePartition	设备是否支持缓存分区	是
缓存分区大小 disk.cachePartition.size	设备缓存分区大小，单位为 MB	66
抽象的 LCD 密度 hw.lcd.density	泛化的屏幕像素密度	160
最大虚拟堆（Heap）空间 vm.heapSize	在被操作系统终止前应用程序分配的最大堆尺寸	16

Download at http://www.pin5i.com/

Tip

在配置与应用程序目标硬件接近的 AVD 上花一点时间，可以为你节约很多时间、金钱，避免很多麻烦。另外，你还需要将这些具体的设置与开发团队中的其他开发人员和测试人员共享。例如，下面的网址提供了有关常用 Android 设备配置的文章：http://androidbook.blogspot.com/2010/08/creating-useful-avds.html。

A.3 使用指定 AVD 启动模拟器

当你配置好了想要使用的 AVD 之后。你就可以启动模拟器了。有很多方法可以启动模拟器，以下列举常用的 4 种。

① 在 Eclipse 中，配置应用程序的调试（Debug）或运行（Run）选项，选用指定的 AVD。

② 在 Eclipse 中，配置应用程序的调试（Debug）或运行（Run）选项，允许开发人员在启动时手动选择 AVD。

③ 在 Eclipse 中通过 Android SDK 和 AVD 管理器直接启动模拟器。

④ 模拟器存储在 Android SDK 的/tools 目录下，它可以从命令行中以独立进程的方式运行（通常只有在不使用 Eclipse 时才有这种需求）。

Tip

在开发过程中，在各个调试会话间保持模拟器处于运行状态，以便快速地重装和调试应用程序。这能够节约每次启动模拟器的几分钟时间，只需要在 Eclipse 中简单地运行“Debug configuration”，然后调试器就能被成功地关联。

A.3.1 配置模拟器启动项

Android 模拟器拥有一系列的配置选项，它们在 Eclipse 中特定应用程序的 Debug 和 Run Configuration 中进行配置，或者在从命令行中启动模拟器时进行设定。某些模拟器的启动设置包含网速、延迟、媒体设置、开启或关闭开机动画以及其他许多的系统设置，还可能会有调试设置，如代理服务器的支持、DNS 地址等。完整的模拟器启动选项可以从 Android 模拟器文档中获得。

A.3.2 启动模拟器运行应用程序

最常见的启动模拟器的方式就是运行一个模拟器实例（带有特定的 AVD 配置）安装或卸载你的最新完成的应用程序。你可以通过使用 Eclipse 的运行与调试配置完成这一操作，在第 3 章“编写第一个 Android 应用程序”中，我们曾讨论过。

Tip

你可以分别创建运行配置和调试配置，使用不同的设置项、不同的启动项，甚至不同的 AVD。

在 Eclipse 中为特定的项目创建调试配置，遵循以下几个步骤。

① 选中“运行（Run）”→“调试配置（Debug Configuration）”（或者右键单击“项目（Project）”然后选中“调试为（Debug As）”）。

② 双击“Android 应用程序（Android Application）”。

③ 为你的调试配置命名（一般使用项目名称）。

Download at http://www.pin5i.com/

④ 单击“浏览”按钮选择项目。

⑤ 切换至目标标签，选中合适的“Deployment Target Selection Mode”。你可以选择特定的 AVD，或是使用模拟器（只有那些符合应用程序目标设备要求的 AVD 才会显示在列表中），或者选用手动选项，在启动时选择可用 AVD。

⑥ 在 Target 标签页内配置模拟器的任意启动选项。你还可以在 Additional Emulator Command-Line Options（附加模拟器命令行选项）域中输入任何标签页中没有显示的选项。

最后的调试配置结果，如图 A.4 所示。

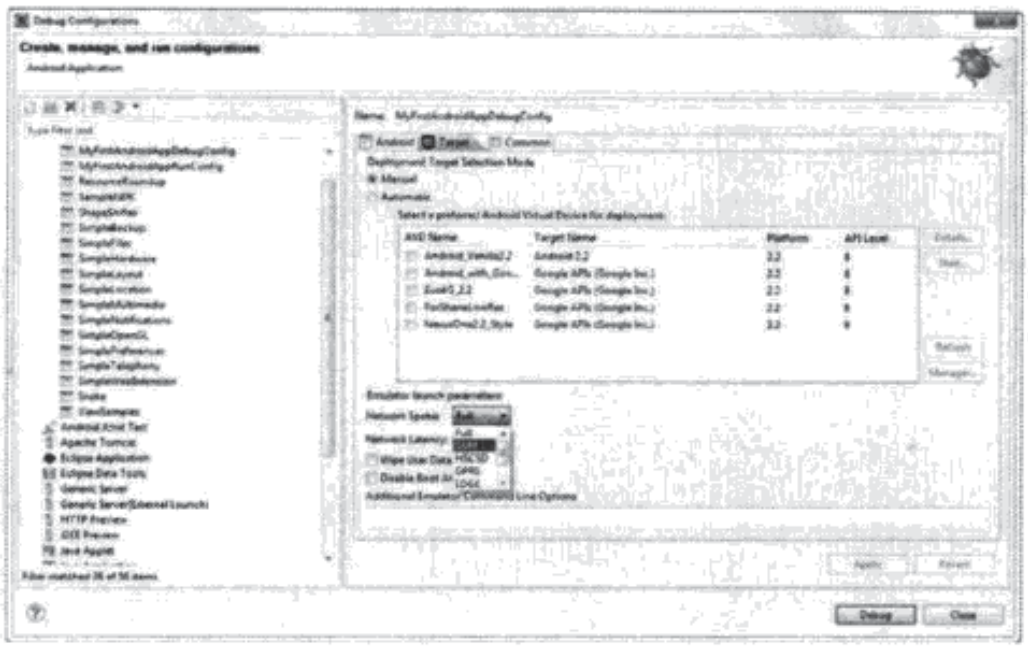


图 A.4 在 Eclipse 中创建调试配置

你可以以相似的方式创建 Run Configuration（运行配置）。如果你为在“Deployment Target Selection Mode”模式下使用而设置了特定的 AVD 的话，那么在 Eclipse 中调试应用程序时，模拟器都会使用这个 AVD。不过，如果你选择了“Manual（手工）”选项的话，那么在你第一次尝试调试应用程序时，将会提示你从 Androi 设备选择器（Android Device Chooser）中选择一个 AVD，如图 A.5 所示。在你运行了模拟器之后，Eclipse 会将它与当前功能关联，并在今后的调试中使用它。

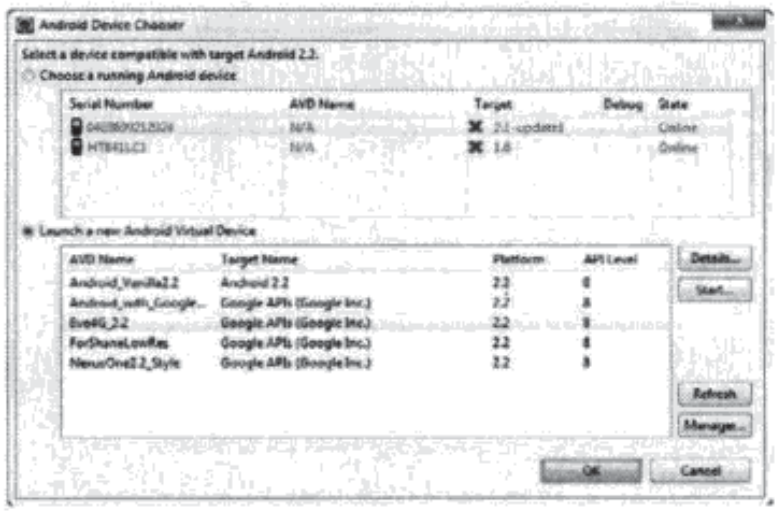


图 A.5 Android 设备选择器

Note

如果你的 Android 设备通过 USB 连接了计算机，那么在你运行或调试应用程序时，将会提示选择运行时的目标设备，而无论你是否已经选择了某个 AVD。这样，你可以将安装或者重装操作定向到连接的设备，而非模拟器。

A.3.3 从 Android SDK 和 AVD 管理器中启动模拟器

有时你会想让模拟器模拟通信，比如启动另外一个模拟器实例，然后模拟拨打电话、发送消息等。在这种情况下，你可以在 Android SDK 与 AVD Manager 中启动它，步骤如下。

① 单击 Eclipse 工具栏上带有箭头的小 Android 图标（），或者执行 Eclipse 菜单命令 Window→Android SDK and AVD Manager 运行 Android SDK 和 AVD 管理器。

② 在左侧的菜单中单击 Virtual Devices（虚拟设备）菜单项，配置好的 AVD 会以列表的方式显示出来。

Download at http://www.pin5i.com/

③ 选中列表中已有的某个 AVD 配置，或者创建一个符合你要求的新 AVD。
④ 单击“开始（Start）”按钮。
⑤ 配置需要的启动项。
⑥ 单击“启动（Launch）”按钮，模拟器就会使用你要求的 AVD 启动了。

Warning

你不能同时运行一个 AVD 配置的多个实例。仔细思考你就会发现这是很有道理的，因为每个 AVD 配置都会保持自己的状态和固有数据。

A.4 配置模拟器的 GPS 位置信息

要开发和测试带有 Google Maps 支持位置服务的应用程序，你需要创建带有 Google API 的目标 AVD。创建完成后，你需要为它配置相应的位置信息。模拟器并不带有位置感应器，所以，你首先需要为模拟器指定 GPS 坐标：在调试模式下运行模拟器，然后按照以下步骤执行。

① 运行模拟器。如果你正在运行某个应用程序，那么按下“主页（Home）”按钮或导航至主页。

② 选择地图（Maps）应用程序。如果你没有在这个目标 AVD 平台中配置 Google API 的话，其中就不会有地图应用。

③ 在地图应用程序中单击“菜单（Menu）”按钮。

④ 选择“我的位置（My Location）”菜单项。

⑤ 在 Eclipse 中，单击右上角的 Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）视图。

⑥ 在左上角的面板中，选择你想要发送位置信息的模拟器实例（如果正在运行的模拟器不止一个的话）。

⑦ 在屏幕左侧你将看到一个“模拟器控制（Emulator Control）”面板。向下拉至“位置控制（Location Control）”。

⑧ 手动键入你想要发送给模拟器位置的经纬度。例如，Yosemite 的坐标为经度：−119.588542，纬度：37.746761。

⑨ 单击“发送（Send）”。

回到模拟器，你将会发现 Google Maps 现在显示了你所指定的位置。如果你指定的是“Yosemite”，那么模拟器看起来将和图 A.6 所示类似。现在，你的模拟器就具有了模拟的位置。

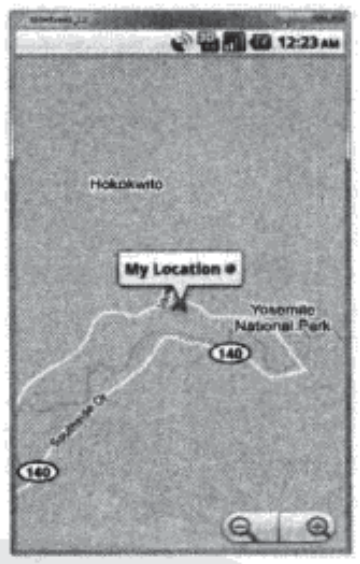


图 A.6 将模拟器位置设置为“Yosemite Valley”

如果你有需要的话，则也可以使用 GPX 坐标文件通过 DDMS 向模拟器发送一系列 GPS 坐标。

Tip

要查找一个指定位置的 GPS 坐标，可以借助 http://maps.google.com。导航至该位置，并且在地图上右键单击使其居中放置。点击“分享链接（link to map）”，并且复制 URL。仔细查看这个 URL，找到“ll”变量，它代表某个位置的“经纬度（latitude/longitude）”。例如，“Yosemite Valley”的链接就具有值“ll=37.746761,−119.588542”，它代表的是“纬度：37.746467”；经度：−119.588542。

Download at http://www.pin5i.com/

A.5 在两个模拟器实例之间拨打电话

你可以让两个模拟器的实例通过“拨号器（Dialer）”应用程序相互拨打电话。模拟器的“电话号码”为其端口号，这可以在模拟器窗口的标题栏中找到。要模拟在两个模拟器之间的电话拨打，你需要执行以下步骤。

① 运行模拟器的两个同时运行的实例。（使用 Android AVD 和 SDK 管理器显得最简单。）

② 查看需要接收呼叫的模拟器端口号。

③ 在发送呼叫的模拟器中运行拨号器应用程序。

④ 键入待拨打的端口号，按下“确定（Enter）”（或“发送”按钮）。

⑤ 在接收端的模拟器实例中将会看到（并且听到）到达的呼叫。图 A.7 显示了一个端口号为 5554 的模拟器（右）使用拨号器应用程序拨打位于 5556 端口上的另一个模拟器。

⑥ 按下“发送”按钮或滑过拨号器接听电话。

⑦ 假装通话一段时间。图 A.8 显示了正在通话的状态。

图 A.7 模拟两个模拟器之间的电话

图 A.8 正在通话中的两个模拟器

⑧ 可以在任何时候按下“结束通话（End）”键终止模拟器通话。

A.6 在两个模拟器实例之间发送消息

你可以像前面介绍的使用端口号拨打电话那样，在两个模拟器之间收发短信息。要模拟在两个模拟器之间发送短消息，你需要执行以下步骤。

① 运行模拟器的两个实例。

② 查看需要接收短消息的模拟器端口号。

③ 在发送短消息的模拟器中运行“信息（Messaging）”应用程序。

④ 将刚才记录的接收短消息的模拟器端口号键入信息的“收件人（To）”一栏。键入一条信息，如图 A.9 所示。点击“发送”按钮。

⑤ 你会在接收方模拟器上看到（同时听见）一条新到达的短息。图 A.10 显示了一个端口号为 5554 的模拟器收到了一条来自端口号 5556 的模拟器发送的短消息。

⑥ 通过下拉通知或启动信息应用程序查看新短消息。

Download at http://www.pin5i.com/

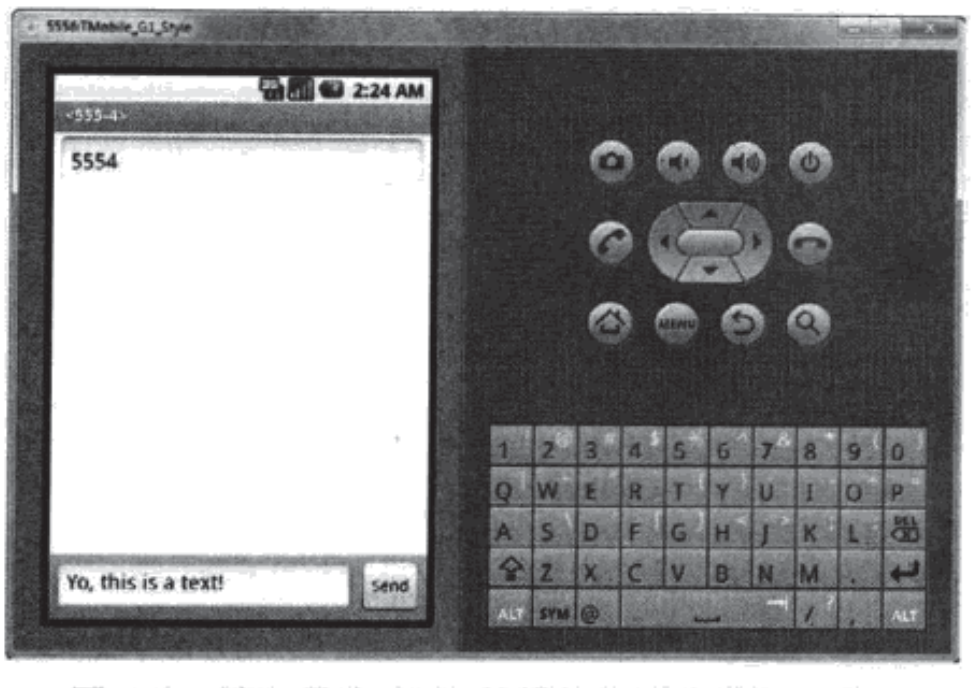


图 A.9 端口号为 5556 的模拟器正在编写一条准备发送给端口号 5554 的模拟器的信息

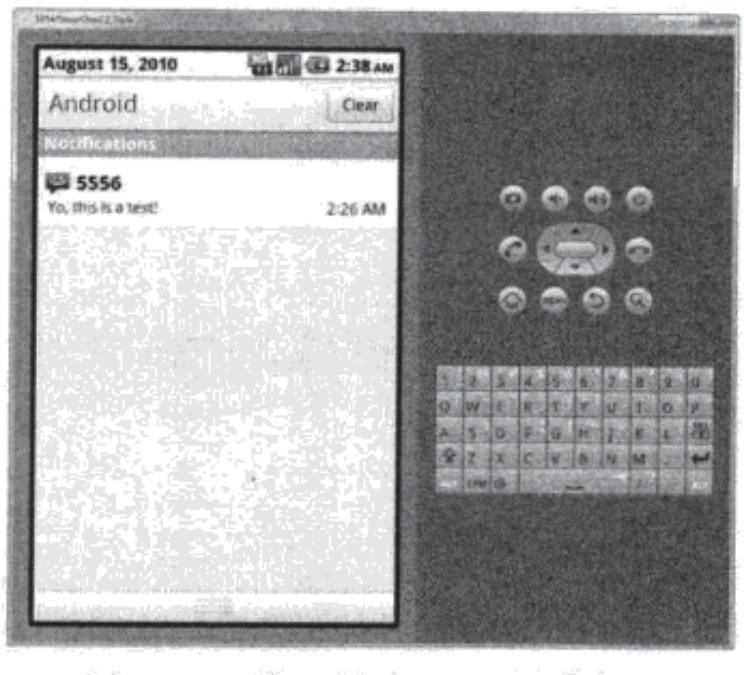


图 A.10 端口号为 5554 的模拟器接收到一条新信息

⑦ 假装聊天。图 A.11 显示了一段短消息对话。

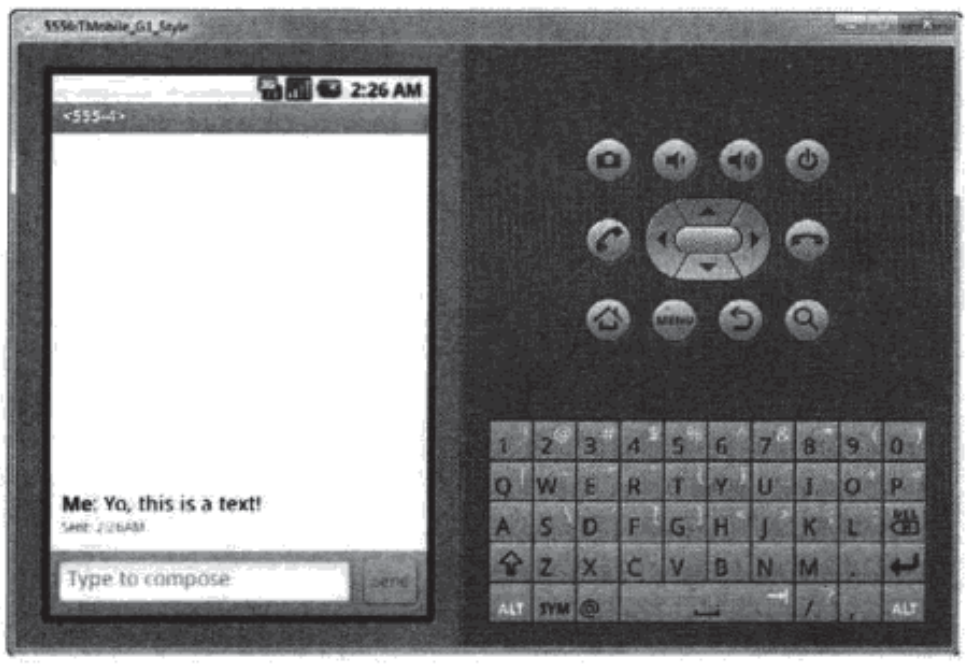


图 A.11 两个模拟器间的短消息进程

Note

你也可以使用 DDMS 或控制台来向模拟器实例发送短消息。

A.7 通过控制台与模拟器交互

你可以通过 telnet 连接到模拟器控制台与其进行交互。例如，要通过端口 5554 连接到模拟器的控制台，可以使用如下命令：

```
telnet localhost 5554
```

你可以使用模拟器控制台来向模拟器发布指令。要终止这一会话，只需要键入 quit 或 exit。你还可以使用 kill 命令来关闭当前的模拟器实例。

A.7.1 使用控制台模拟来电

你可以在模拟器上模拟特定号码的来电，其命令如下：

```
gsm call <number>
```

Download at http://www.pin5i.com/

例如，要模拟来自号码 555-1212 的通话，可以使用如下命令：

```
gsm call 5551212
```

执行上面命令后的结果显示在图 A.12 中。

图 A.12 通过模拟器控制台模拟的呼入，号码为 555-1212（预设姓名为 Anne Droid）

Tip

你可以使用 gsm 子命令来控制呼入和呼出的通话、线路繁忙的信号等。gsm 子命令包括 list、call、busy、hold、accept、cancel、data、voice 和 status。

A.7.2 使用控制台模拟短消息

你可以模拟来自特定号码的短信息，其命令如下：

```
sms send <number> <message>
```

例如，要模拟一个来自号码 555-1212 且内容为“How are you?”的短信息，可以使用如下命令：

```
sms send 5551212 HRU?
```

在模拟器的白色状态栏上将会出现一个通知，它告诉你有新的短消息到达。稍后它甚至会显示短消息的内容并滚动，然后显示消息图标。你可以将通知条向下拖曳来查看消息的内容或者运行“消息”应用程序。

图 A.13 给出了上面命令行在模拟器中的执行结果。

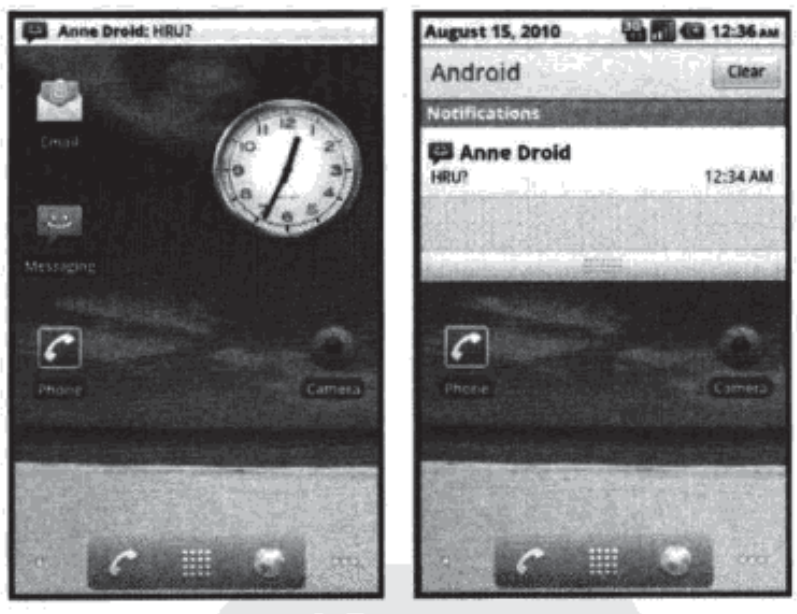


图 A.13 通过模拟器控制台模拟的短消息到达，发件人号码为 555-1212（预设名称 Anne Droid）

Caution

sms send 命令并不能在所有我们测试的平台上都能正常执行。例如，如果你需要发送的短信息中包含空格，那么你可能希望使用一对引号来将整个内容合成一体，以避免模拟器的字符编码出现问题，然而，你所使用的引号却有可能出现在短消息正文中。

A.7.3 使用控制台发送 GPS 坐标

你可以使用模拟器控制台来向模拟器发送简单的 GPS 坐标，其命令如下：

Download at http://www.pin5i.com/

```
geo fix <longtitude> <latitude> [<altitude>]
```

例如，要将模拟器的位置定位在“Mount Everest”，那么首先通过选择“Menu→My Location”运行模拟器上的地图应用程序，然后在模拟器控制台下发布如下命令，设置设备的坐标：

```
geo fix 86.929837 27.99003 8850
```

A.7.4 使用控制台监视网络状态

你可以使用控制台监视模拟器的网络状态并且实时地修改网络连接速度和时延。用于显示网络状态的命令如下：

```
network status
```

一般情况下，该命令返回的结果应该大致如下所示：

```
network status
Current network status:
  download speed:        0 bits/s (0.0 KB/s)
  upload speed:          0 bits/s (0.0 KB/s)
  minimum latency: 0 ms
  maximum latency: 0 ms
OK
```

A.7.5 使用控制台操作电源设置

你可以使用电源管理命令来管理虚拟的电源设置。比如，你可以像下面这样将电池的剩余电量修改为 99%：

```
power capacity 99
```

你可以将交流充电状态设置为 off（或 on），如下所示：

```
power ac off
```

你可以将电池状态（status）设置为 unknow、charging、discharging、not-charging 或 full，如下所示：

```
power status full
```

你可以将电池“Present（存在）”状态设置为 true（或 false），如下所示：

```
power present true
```

你可以将电池的“健康”状态设置为 unknown、good、overheat、dead、overvoltage 或 failure，如下所示：

```
power health good
```

你还可以通过下面的命令查看当前的电源设置：

```
power display
```

通常情况下，该命令返回的结果应该大致如下所示：

```
power display
AC: offline
status: Full
health: Good
present: true
capacity: 99
OK
```

Download at http://www.pin5i.com/

A.7.6 使用其他控制台命令

除了上面介绍的命令之外，还有许多其他的控制台命令，包括模拟硬件事件、端口重定向（redirection）、对虚拟机的检测、启动和终止等。

QA（质量保障）人员应该会查看更详尽的事件子命令，因为它们可以自动模拟按键事件（key events）。这类似于 ADB Exerciser Monkey 所使用的相同接口，它会模拟随机的按键来测试应用程序是否会崩溃。

A.8 享受模拟器

出于“好玩”这一目的，我们给出使用模拟器的一些小窍门。

① 在主页上，按住桌面不放可以切换墙纸，还可以向桌面添加应用程序、快捷方式和小插件。

② 如果你在应用程序托盘中按住一个图标（通常为应用程序图标）不放，则可以在主页上放置一个它的快捷方式，以便今后快速访问。

③ 如果你在主页上按住一个图标不放，则可以将它随意移动或抛弃到“垃圾桶”里。

④ 可以快速地将主页向左或向右拨动，以显示更多的空间。其右侧为 Google 搜索，左侧则拥有更多可以放置桌面项的空间。

⑤ 另一种修改墙纸和添加桌面图标的方法为：在桌面按下“菜单（Menu）”键，然后选择“添加（Add）”。此时，你可以添加快捷方式和图片小插件，比如你的家庭照片等，如图 A.14 所示。

图 A.14 使用应用程序快捷方式和小插件来自定义模拟器主页

A.9 认识模拟器的局限

模拟器虽然很强大，但它仍然具有以下几个不可忽视的局限。

① 它不是一个硬件设备，所以并不能反映出真实的行为，而仅仅是模拟。

② 虽然它可以模拟通话，但你并不能真正地进行通话。

③ 对于设备状态（网络状况、电池充放电情况）的模拟是很有局限性的。

④ 对于外设（摄像头的图像捕捉，耳机、SD 卡的插入，以及位置服务等）的模拟具有很大的局限性。

⑤ 对 API 的支持具有很大的局限性（例如，目前不支持 OpenGL ES 2）。

⑥ 性能受限（现在设备的性能通常会比在模拟器中表现出来的强很多，例如视频和动画）。

⑦ 对制造商和运营商定制的设备特性、主题和用户体验的支持很有限。某些制造商，例如摩托罗拉，提供了模拟器插件，以更好地模拟特定设备的真实情况。

⑧ 不支持 USB 和蓝牙。

Download at http://www.pin5i.com/

附录 B

Android DDMS 快速入门

Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）是由 Android 软件开发包（Software Development Kit，SDK）提供的调试工具。开发人员可以使用 DDMS 提供的窗口来监视模拟器或真实设备的调试，包括对文件和进程的管理等。它是几个工具的完美融合：任务管理器（Task Manager）、文件浏览器（File Explorer）、模拟器控制台（Emulator console）和日志控制台（Logging console）。这里提供的“快速入门”并不是 DDMS 各项功能的完整文档，而是提供给你用于快速入门并执行常规的任务。如果你需要获取其完整的特性说明，可以查看 Android SDK 提供的 DDMS 文档。

B.1 独立或结合 Eclipse 使用 DDMS

如果你使用的是安装了 Android 开发工具插件（Android Development Tools Plug-In）的 Eclipse 集成开发环境（Integrated Development Environment，IDE），那么 DDMS 工具已经紧密地融合到了你的开发环境中。通过 DDMS 视图（如图 B.1 所示，它正使用文件浏览器查看模拟器实例上的文件），你可以浏览任何一个在计算机上运行的模拟器实例，并且能够查看通过 USB 连接的 Android 设备。

图 B.1 在 Eclipse 下的 DDMS 视图，其中包含一个模拟器和一个连接的 Android 设备

如果你没有使用 Eclipse，那么 DDMS 也可以在单独的进程中运行，它位于/Tools 目录。在这种情况下，DDMS 将运行在自己的进程中，如图 B.2 所示（正在向模拟器发送 SMS）。

Download at http://www.pin5i.com/

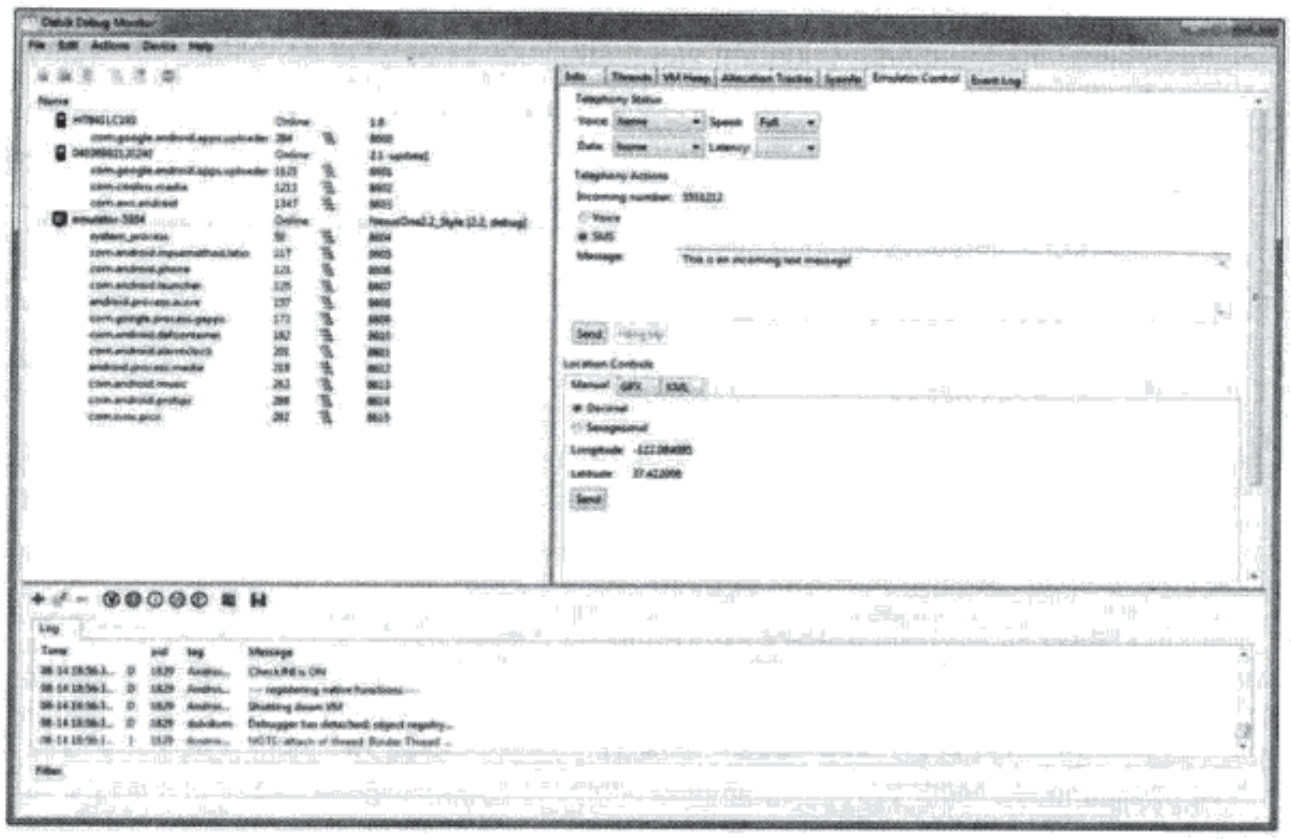


图 B.2 DDMS 工具，其中包含一个模拟器和一个连接的 Android 设备

Tip

在同一时间只允许运行一个 DDMS 实例。其他运行的 DDMS 会被忽略，包括 Eclipse 视图。如果你正在运行 Eclipse，想从命令行启动 DDMS 的话，你可能会看到问号，而不是进程名称，以及一个调试输出，提示 DDMS 实例被忽略了。

B.2 熟悉 DDMS 的关键特性

无论你是在 Eclipse 中使用 DDMS，还是单独使用它，都需要注意以下几个关键特性。

① 处于运行状态的模拟器和连接的设备处于右上角区域。

② 文件浏览器允许你查看模拟器和设备上的文件（包括应用程序文件、目录和数据库），并且可以进行提取或添加。

③ LogCat 窗口能够让你监视 Android 日志控制台（LogCat）。在这里将显示 `Log.i()`、`Log.e()` 和其他 Log 方法调用产生的消息。

④ 可以查看每一个进程（堆和线程更新），也可以查看每一个线程，还可以终止进程。你可以触发进程上的“垃圾回收（garbage collection）”，并随后查看应用程序所使用的堆。

⑤ 你可以使用内存跟踪面板（Allocation Tracker pane）跟踪应用程序的内存分布。

⑥ 你可以使用“Screen Capture（屏幕捕捉）”按钮来捕捉模拟器和设备上的屏幕画面。

⑦ 你随时可以使用模拟器控制台，比如发送 GPS 信息，模拟来电或 SMS 到达等。

某些功能只有在 Eclipse DDMS 视图下可用，比如你可以在模拟器或设备的某个进程上单击，然后点击那只绿色的小虫图标为其关联（attach）一个调试器，这样就可以在提供了源代码的情况下调试这个进程了。这个过程在其他调试器上也都大同小异，例如 JSwat 或 jdebug。

B.3 使用进程

DDMS 非常有用的一个特性在于可以同进程打交道。可能你还记得，每一个 Android 应用

Download at http://www.pin5i.com/

程序都使用其自己的用户 id 运行在操作系统的单独的 VM（虚拟机）中。通过 DDMS 左侧的面板，你可以查看所有在设备上运行的 VM 实例，每一个均以其包名称作为标识。你可以：

① 在 Eclipse 中关联（attach）并调试应用程序；

② 监视线程；

③ 监视堆；

④ 终止进程；

⑤ 强制进行垃圾回收（Garbage Collection，GC）。

例如，在图 B.3 中，我们可以看到一个名为 com.androidbook.SuperDuperPetTracker 的包运行在模拟器上。你可以在右侧面板中查看应用程序线程。

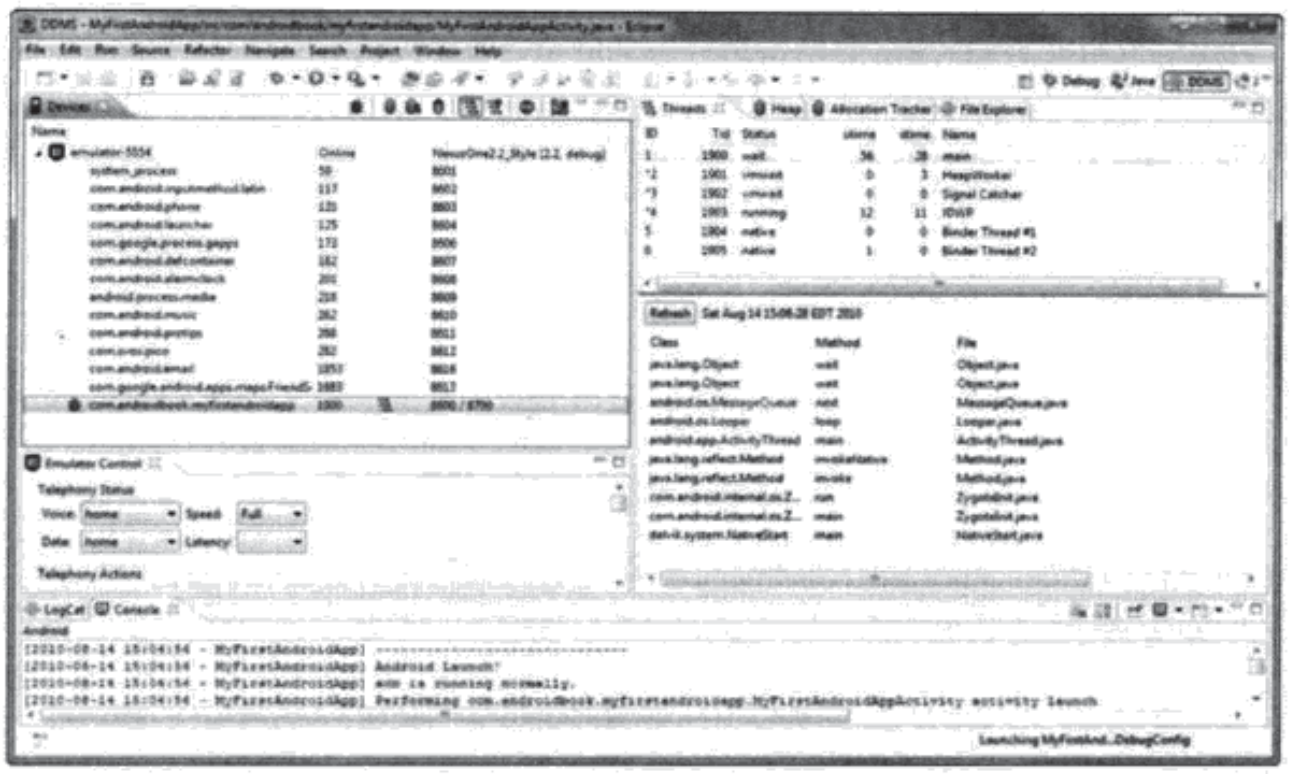


图 B.3 使用 DDMS 显示应用程序的线程信息

B.3.1 向 Android 应用程序关联调试器

虽然大多数情况下你会使用 Eclipse 调试参数来运行并调试应用程序，但你也可以使用 DDMS 来选择任何需要调试的应用程序，并直接关联并调试它。要为一个进程关联调试器，你需要在 Eclipse 工作区中打开对应包的源代码。然后执行以下步骤进行调试。

① 在模拟器或设备上，确认你想要调试的应用程序处于运行状态。

② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。

③ 单击绿色的小虫图标（）开始调试。

④ 在必要时切换到 Eclipse 的调试视图，像通常一样进行调试。

B.3.2 监视 Android 应用程序的线程活动

你可以使用 DDMS 来监视每一个 Android 应用程序的线程活动。图 B.3 给出了正在监视某个应用程序线程活动的 DDMS 界面。可以执行以下步骤。

① 在模拟器或设备上，确认你想要监视的应用程序处于运行状态。

② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。

③ 单击带有三个箭头的小图标（）以显示应用程序的线程。它们将出现在 Thread 标签的右侧。默认情况下，这里显示的数据每 4 秒钟进行一次更新。

④ 在 Thread 标签页中，你可以选择某个特定的线程并且按下 Refresh 按钮来深入查看这

Download at http://www.pin5i.com/

个线程。其中包含的类将会显示在下方区域。

Note

你也可以使用带小红点的三个黑色箭头的图标（）来启动线程管理。

B.3.3 触发垃圾回收（GC）

你可以使用 DDMS 来强制进行垃圾回收（Garbage Collection，GC），操作步骤如下。

① 在模拟器或设备上，确认你想要进行 GC 的应用程序处于运行状态。

② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。

③ 单击垃圾箱按钮（）来运行清理垃圾的操作。也可以在 Heap 标签页中执行这一操作，稍后将进行讲解。

B.3.4 监视堆活动

你可以使用 DDMS 来监视每一个 Android 应用程序的堆统计数据。在每次垃圾回收后堆的统计数据将进行更新。图 B.4 显示了正在监视某个应用程序堆状态的 DDMS 界面。

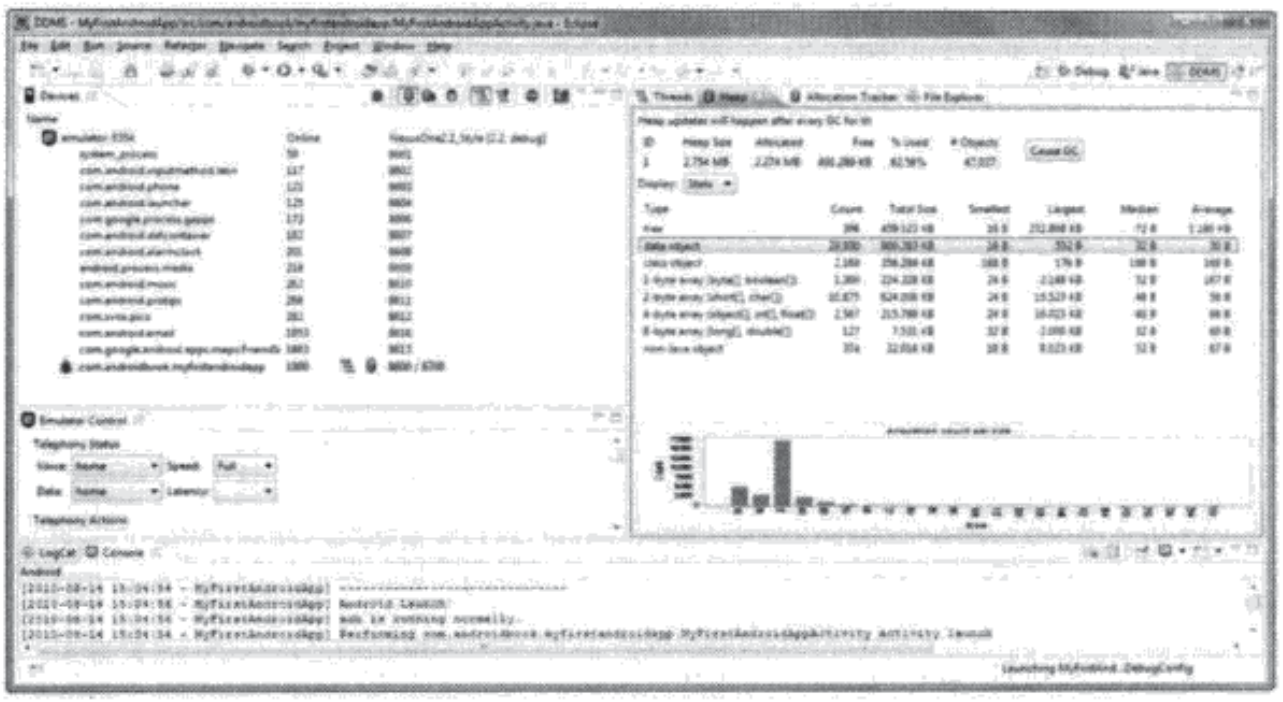


图 B.4 使用 DDMS 显示应用程序的堆信息

可以执行以下步骤。

① 在模拟器或设备上，确认你想要监视的应用程序处于运行状态。

② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。

③ 单击绿色的圆桶图标（）以显示该应用程序的堆信息。统计数据将出现在 Heap 标签的右侧。这一数据将在每次垃圾回收后予以更新。你也可以通过单击 Heap 标签页中的“Cause GC”按钮来触发一个垃圾回收操作。

④ 在 Heap 标签页中，你可以选择特定类型的对象。它的使用情况图表将显示在 Heap 标签页的底部，如图 B.4 所示。

B.3.5 监视内存分配

你可以使用 DDMS 监视某个指定 Android 应用程序的内存分配。内存分配数据可以根据开发人员的需要而更新。执行以下步骤实现内存分配跟踪。

① 在模拟器或设备上，确认你想要监视的应用程序处于运行状态。

Download at http://www.pin5i.com/

② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。
③ 将右侧面板切换至“分配跟踪器 Allocation Tracker”标签页。
④ 单击“开始跟踪”按钮，启动内存分配跟踪操作，即可获得一段时间内的分配情况。
⑤ 如果想要停止跟踪，单击“停止跟踪”按钮。

Android 开发人员网站上有一篇关于如何跟踪内存分配情况的文章：http://developer.android.com/resources/articles/track-mem.html。

B.3.6 终止 Android 进程

你可以使用 DDMS 来终止一个 Android 应用程序，步骤如下。
① 在模拟器或设备上，确认你想要终止的应用程序处于运行状态。
② 在 DDMS 中，找到这个应用程序的包，并且点击它使其高亮。
③ 点击带有红色停止符号的图标（）终止该进程。

B.4 使用文件浏览器

你可以使用 DDMS 来查看并操作模拟器和设备上的 Android 文件系统。表 B.1 给出了 Android 文件系统中的某些重要区域。

表 B.1 Android 文件系统中重要的目录

目　录	说　明
/data/app/	用于存储 Android APK 文件
/data/data/<package name>/	应用程序顶层目录 例如：/data/data/com.androidbook.pettracker
/data/data/<package name>/shared_prefs/	应用程序共享首选项目录 命名的首选项以 XML 文件的方式进行存储
/data/data/<package name>/files/	应用程序文件目录
/data/data/<package name>/cache/	应用程序缓存目录
/data/data/<package name>/databases/	应用程序数据库目录 例如：/data/data/com.androidbook.pettracker/databases/test.db
mnt/sdcard/	外部存储（SD 卡）
mnt/sdcard/download	用于存储浏览器下载图像

B.4.1 浏览模拟器或设备上的文件系统

要浏览 Android 文件系统，可以执行以下步骤。
① 在 DDMS 中，选择你想要浏览的模拟器或设备。
② 切换至 File Explorer（文件浏览器）标签页。你将看到层级显示的目录。
③ 浏览某个文件夹或文件。
记住，当文件浏览器中的目录列表发生了更改，需要过一段时间才能够更新。

Note

某些设备目录，例如/data 目录，可能不能通过 DDMS 文件浏览器访问。

Download at http://www.pin5i.com/

B.4.2 从模拟器或设备上拷贝文件

你可以使用文件浏览器将模拟器或设备上的文件或文件夹拷贝到计算机上，步骤如下。

① 使用文件浏览器导航至需要拷贝的文件或文件夹，点击使其高亮。

② 在文件浏览器的右上角，点击带箭头的光盘图标（）提取设备中的文件。另外，你也可以展开图标旁边的下拉菜单并从中选择 Pull File 来执行这一操作。

③ 键入计算机上用于存放这一文件或文件夹的路径，然后点击 Save。

B.4.3 向模拟器或设备拷贝文件

你可以使用文件浏览器将计算机上的文件拷贝到模拟器或设备的文件系统中，步骤如下。

① 使用文件浏览器导航至需要拷贝文件的文件夹，点击使其高亮。

② 在文件浏览器的右上角，点击带箭头的手机图标（）向设备中添加文件。另外，你也可以展开图标旁边的下拉菜单并从中选择 Push File 来执行这一操作。

③ 选择计算机上待拷贝的文件，然后点击 Open 按钮。

Tip

文件浏览器还支持鼠标拖曳。这也是唯一可以向 Android 文件系统中拷贝文件夹的操作。不过，并不推荐向 Android 文件系统中拷贝文件夹，因为并没有用于删除它们的选项。而你如果拥有许可权限的话，需要使用程序来删除这些文件夹。不过总之，你可以从计算机上将一个文件或文件夹拖到文件浏览器中，并在适当的位置释放它。

B.4.4 删除模拟器或设备上的文件

你可以使用文件浏览器来删除模拟器或设备上的文件（但不能删除文件夹），步骤如下。

① 使用文件浏览器导航至需要删除的文件，点击使其高亮。

② 在文件浏览器的右上角，点击红色的减号图标（）来删除文件。

Warning

执行这一操作时你需要特别小心，因为没有任何确认提示，文件将立即删除并且没有办法恢复它们。

B.5 使用模拟器控制

你可以通过 DDMS 的 Emulator Control（模拟器控制）标签来操作模拟器的实例，这之前必须选中需要操作的模拟器。你针对下面的目的使用模拟器控制标签：

① 修改通话（telephony）状态；

② 模拟语音通话呼入；

③ 模拟 SMS 接收；

④ 发送位置坐标（GPS 坐标）。

B.5.1 模拟语音来电

要使用模拟器控制标签来模拟语音呼入，可以执行以下步骤。

Download at http://www.pin5i.com/

① 在 DDMS 中，选择想要拨打的模拟器。
② 切换到 Emulator 标签页，你将使用 Telephony Actions。
③ 输入模拟呼入的电话号码，它可以包括任意数字、“+”和“#”。
④ 选择 Voice 单选项。
⑤ 点击 Call 按钮。
⑥ 模拟器将会接收到呼入并响铃。接听电话。
⑦ 模拟器可以像正常情况一样挂断电话，也可以使用 DDMS 中的 Hang Up 按钮终止通话。

B.5.2 模拟短消息接收

DDMS 提供了最稳定的向模拟器发送 SMS 的方法。其过程同模拟语音来电类似。要使用模拟器控制标签模拟发送 SMS，可以执行以下步骤。

① 在 DDMS 中，选择需要接收 SMS 的模拟器。
② 切换到 Emulator 标签页，你将使用 Telephony Actions。
③ 输入模拟发送的电话号码，它可以包括任意数字、“+”和“#”。
④ 选择 SMS 单选项。
⑤ 键入 SMS 消息的正文。
⑥ 点击 Send 按钮。
⑦ 模拟器将会接收到 SMS 并显示通知。

B.5.3 发送位置坐标

向模拟器发送 GPS 坐标的步骤在附录 A“Android 模拟器快速入门”中进行了详细的讲解。只需要在模拟器控制标签中简单地输入 GPS 坐标，点击 Send 按钮，然后就可以使用模拟器上的“Maps”应用程序接收当前位置了。

B.6 使用应用程序日志

DDMS 中融合了 `LogCat` 工具，它为 DDMS 用户界面中底部的一个标签。你可以通过点击内含字母的圆圈图标来控制信息的显示量。默认的“V”代表“Verbose”（即显示所有信息），其余的可选图标包括“D”（“Debug”，调试）、“I”（“Information”，信息）、“W”（“Warning”，警告）和“E”（“Error”，错误）。

你还可以创建自定义过滤标签以显示仅与调试标记（Debug Tag）相关的 `LogCat` 信息。可以用“+”按钮来添加一个过滤标签以显示仅与特定标记相匹配的日志信息。对你的应用程序创建专有的调试标记将非常有用，这样，你就可以过滤 `LogCat`，以保证只显示与你的应用程序相关的日志活动。例如，在你的应用程序中有如下代码：

```
public static final String DEBUG_TAG = "MyFirstAppLogging";
Log.i(DEBUG_TAG, "This is info about MyFirstAndroidApp.");
```

那么，你可以使用“+”按钮创建 LogCat 过滤器。为过滤器命名，并将其日志标记设置和你的调试标记统一起来：

```
MyFirstAppLogging
```

Download at http://www.pin5i.com/

就这样，你现在拥有了一个名为“Logging My App”的 LogCat 标签，它将只显示带有你的应用程序标记的日志信息，如图 B.5 所示。

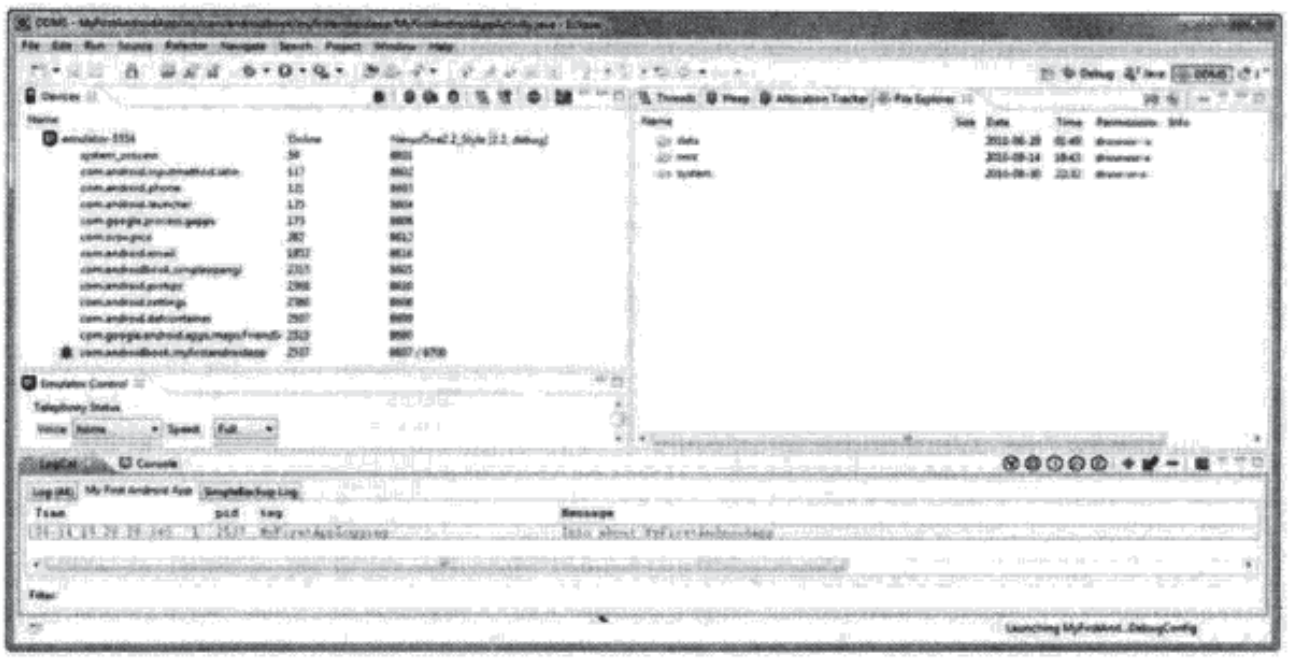


图 B.5 带有 LogCat 自定义标记的 DDMS 工具

Tip

如果 LogCat 日志工具中存储了过量的信息，DDMS 的显示可能会出错，所以最好定期清理一下日志记录。

B.7 在模拟器或目标设备上截屏

你可以在 DDMS 中截取模拟器或设备的屏幕显示。设备截屏对于调试来讲非常有用，它使 DDMS 工具特别适合 QA 人员，并且受到开发人员的欢迎。要进行屏幕截取，可以执行以下步骤。

① 在 DDMS 中，选择需要截屏的模拟器或设备。

② 在模拟器或设备上，确认屏幕显示的正是你想要截取的画面。

③ 点击带有方形彩色图案的图标（）进行截屏。此时将启动一个截屏窗口，如图 B.6 所示。

图 B.6 使用 DDMS 进行屏幕截取

④ 在截屏窗口中，单击 Save 按钮保存屏幕截图。类似地，Copy（复制）按钮可以将截图保存在你的剪贴板中，Refresh（刷新）按钮能够在设备或模拟器屏幕改变时更新屏幕截图，如果你启动了截图窗口的话。

Download at http://www.pin5i.com/

附录 C

Android 调试桥快速入门

Android 调试桥（Android Debug Bridge，ADB）是通过命令行界面直接同 Android 设备和模拟器进行交互的客户/服务器工具。这一工具为 Android 软件开发包（Software Development Kit，SDK）的一部分，你可以使用它来管理和操作与开发计算机相连的模拟器和设备，可以使用它来查看日志和调试信息。ADB 还为其他工具提供支持，如 Android Eclipse 插件（Android Plug-In for Eclipse，ADT）和 Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）。这里提供的“快速入门”并不是 ADB 各项功能的完整文档，而是提供给你用于快速入门并执行常规的任务。如果你需要获取其完整的特性说明，可以查看 Android SDK 提供的 ADB 文档。

ADB 所提供的许多功能（例如：LogCat Android 日志功能，使用文件浏览器进行文件复制等）都通过 DDMS 和 ADT 与开发环境紧密地结合在一起了。开发人员可以使用这些友好的方法来操作设备和模拟器，不过，你也可以通过 ADB 来实现某些自动化和脚本功能。你还可以使用 ADB 来实现自定义功能，而非完全依赖于默认提供的次级工具。

C.1 列表显示连接的设备和模拟器

你可以使用 ADB 来列出所有连接到开发计算机的 Android 设备和模拟器。为此，只需要使用 adb 命令行的 devices 命令即可。例如：

```
adb devices
```

该命令将列出所有连接到计算机的模拟器和设备的序列号以及状态（offline 或 device）。对于模拟器实例，其序列号基于其唯一端口号。例如，在这个例子中，我们拥有一个模拟器实例（端口 5554）和一个 Android 手机：

```
C:\>adb devices
List of devices attached
emulator-5554   device
HT841LC1977     device
```

C.2 针对特定设备使用 ADB 命令

当你已经知道了想要连接的设备的序列号后，就可以像下面这样发布命令了：

```
adb -s <serial number> <command>
```

例如，要获取特定设备的状态，可以键入：

Download at http://www.pin5i.com/

```
adb -s emulator-5554 get-state
```

除了针对唯一的序列号而使用-s 标签之外，还可以使用-d 标签来直接对某个已连接的唯一设备实例发布命令，或者使用-e 标签来对唯一的模拟器实例发布命令，使用这种方式的前提是模拟器和设备都只连接了一个实例。

例如，由于我们连接了唯一的 Android 手机，所以可以使用如下命令来查询它的序列号：

```
adb -d get-serialno
```

C.3 启动和停止 ADB 服务进程

有时你可能需要手动地重启 ADB 服务器进程。比如，当某个模拟器实例运行了很长时间，并且反复连接和断开调试器，导致 LogCat 日志记录丢失时。在这种情况下，你可能需要终止并重启 ADB 服务器（甚至有可能是 Eclipse）。

C.3.1 停止 ADB 服务进程

要终止 ADB 服务器进程，可以使用 kill-server 命令。例如，下面的命令将终止特定模拟器实例的服务器：

```
adb -s emulator-5554 kill-server
```

C.3.2 启动和检测 ADB 服务进程

你可以使用 start-server 命令来启动 ADB 服务器。

```
adb start-server
```

你还可以使用 start-server 命令来检测服务器是否正在运行。如果给出其他命令但服务器没有运行的话，它会自动运行。

C.4 发布 Shell 命令

ADB 包含一个 Shell 接口（ash），你可以使用它来直接操作设备、发布命令以及运行二进制文件（binaries）。ash Shell 拥有通常的文件访问命令，如 pwd 和 ls。

> **Tip**
>
> 要获得有关 ash Shell 的更多信息，可以查看 “Linux Blog Man” 的 ash 页面：http://www.thelinuxblog.com/linux-man-pages/1/ash。

C.4.1 发布单条 Shell 命令

你可以在不启动 Shell 会话的情况下发布单条 Shell 命令，如下所示：

```
adb shell <command>
```

例如，要列出所有位于模拟器/sdcard/download 目录下的文件，可以使用如下命令：

```
adb -e shell ls /sdcard/download
```

Download at http://www.pin5i.com/

C.4.2 使用 Shell 会话

通常你可能需要发布不止一个命令。在这种情况下，可能需要开启一个 Shell 会话，如下所示：

```
adb shell
```

例如，要连接到具有特定序列号的设备实例并启动 Shell 会话，可以使用如下命令：

```
adb -s emulator-5554 shell
# <type commands here>
# exit
```

之后，你就可以发布命令了。要结束会话只需要键入 exit 即可。

C.4.3 使用 Shell 启动和终止模拟器

终止模拟器将会停止其响应，但是它仍然会显示在开发计算机的屏幕上。要终止一个模拟器，你可以在 ADB Shell 中发布一个 stop 命令。

```
adb -s emulator-5554 shell stop
```

之后，你还可以使用 start 命令重新启动它：

```
adb -s emulator-5554 shell start
```

另外，你还可以在一个 Shell 会话中执行这些命令，如下所示：

```
adb -s emulator-5554 shell
# stop
# start
```

C.5 拷贝文件

你可以使用 ADB 命令行将文件在硬盘和 Android 设备之间进行拷贝。你需要知道待拷贝文件的完整路径，而且依赖于你所拥有的用户权限（本地或远程）。

C.5.1 向设备或模拟器发送文件

你可以使用 push 命令来向设备拷贝文件，如下所示：

```
adb push <local file path> <remote file path on device>
```

例如，要从本地硬盘驱动器中将 Pic.jpg 文件拷贝到设备 SD 卡的 download 目录下，可以使用下面的命令：

```
adb -s HT841LC1977 push c:\Pic.jpg /sdcard/download/Pic.jpg
```

C.5.2 从设备提取文件

你可以使用 pull 命令从设备中拷贝文件，如下所示：

```
adb pull <remote file path on device> <local file path>
```

例如，要将 Lion.jpg 文件从设备的 SD 卡 download 目录下拷贝到本地硬盘驱动器中，

Download at http://www.pin5i.com/

可以使用下面的命令：

```
adb -s HT841LC1977 pull /sdcard/download/Lion.jpg C:\Lion.jpg
```

Tip

如果你将图像文件拷贝到了 SD 卡中（无论是否是虚拟的），都需要使用“Media Scanner”（在模拟器的“Dev Tools”应用程序中）来强制 Android 操作系统进行刷新。

C.6 安装和卸载 Android 应用程序

你可以使用 ADB 来安装或卸载指定 Android 设备或模拟器上的应用程序包。虽然 Eclipse 插件可以自动地为开发人员实现这一功能，但它可以提供给那些没有使用 Eclipse 环境的程序开发人员，或者需要创建自动构建（build）过程和测试环境的测试人员。ADB 还可以用于安装第三方的应用程序包，包括那些自行发布，没有在 Android Market 一类的电子市场中出现的应用程序。

Note

应用程序以包的形式进行安装，这些包由 Android 资源打包工具（Android Asset Packaging Tool，aapt）创建，此工具同时也用于 Eclipse 插件。

C.6.1 安装应用程序

要安装应用程序，首先创建一个 Android 包（.apk）文件，之后使用 `install` 命令：

```
adb install <apk file path>
```

例如，要在模拟器上安装示例应用程序 `Snake`，你可以使用如下命令：

```
adb -e install C:\android-sdk\samples\Snake\bin\Snake.apk
821 KB/s (17656 bytes in 0.021s)
 pkg: /data/local/tmp/Snake.apk
Success
```

C.6.2 重装应用程序

你可以使用`-r` 来重装应用程序包而无需覆盖它的数据。例如，现在你可以使用下面的命令重装 Snake 应用程序而不丢失你的数据：

```
adb -e install -r C:\Snake.apk
```

C.6.3 卸载应用程序

要卸载 Android 应用程序，你需要知道应用程序包的名称：

```
adb uninstall <package>
```

例如，要将 `MyFirstAndroidApp` 应用程序从模拟器中卸载，可以使用下面的命令：

```
adb -e uninstall com.androidbook.myfirstandroidapp
```

Tip

如果你经常变换开发计算机或者签名的话，那么你可能会经常使用这个命令。在第 29 章“销售你的 Android 应用程序”中你可以学习到更多有关应用程序签名的内容。

Download at http://www.pin5i.com/

C.7 使用 LogCat 记录日志

Android 的日志记录信息可以通过 LogCat 来访问。这一功能被整合进了 DDMS 和 Eclipse（使用 ADT 插件）中，但你也可以直接通过 ADB 来进行访问，命令如下：

```
adb logcat <option> <filter>
```

这类型的命令最好在 adb shell 中完成。

C.7.1 显示所有的日志信息

例如，如果你想要显示模拟器实例中所有的 LogCat 日志信息，只需要打开 shell，键入如下 logcat 命令：

```
adb -e shell
# logcat
```

默认地，日志记录模式被设置为 brief。例如，下面显示了来自线程 ID 20054 的“信息”类（brief 模式）日志消息，它使用了过滤标记 AppLog：

```
I/AppLog(20054): An Informational Log message.
```

C.7.2 在日志数据中包含日期和时间信息

另一个日志记录模式为 time，它将包含日志消息的日期和时间。要修改日志模式，可以使用-v 标签并且能够指定其格式。例如，要修改成 time 模式，可以使用如下 adb shell 命令：

```
# logcat -v time
```

日志消息的结果会被格式化为带有日期和时间的形式，后面紧跟着显示事件的严重性（类型）、标记、进程 ID 以及日志消息：

```
01-05 21:52:22.465 I/AppLog(20054): Another Log Message.
```

C.7.3 过滤日志信息

通过 LogCat 工具得到的日志信息可能会非常繁杂。大多数时候你都需要一两个过滤器来从中筛选出需要查看的日志记录。过滤器实际上为格式化的标记和事件优先权的组合，格式如下所示：

```
<Tag Name>:<Lowest Event Priority to Print>
```

例如，要从所有日志记录中筛选出“信息”类（更高优先权的消息类型有“警告”、“错误”和“致命性错误（Fatal）”）日志且具有日志标记“AppLog”的过滤器应做如下定义：

```
AppLog:I
```

Tip

你还可以使用星号（“*”）作为通配符，它将匹配所有的字符。那么，如果你在标记一侧使用了星号，则意味着“所有标记”；而如果你将它使用在事件优先权一侧，则相当于使用了“V”优先权——最低的优先权，将显示所有消息。

Download at http://www.pin5i.com/

1. 根据事件严重性进行过滤

你可以针对事件的某个严重性级别进行创建过滤器。事件的严重性类型（从低优先级到高优先级，也就是从冗杂到简练）如下所列：

① 冗余（Verbose，V）；
② 调试（Debug，D）；
③ 信息（Info，I）；
④ 警告（Warning，W）；
⑤ 错误（Error，E）；
⑥ 致命性错误（Fatal，F）；
⑦ 无打扰（Silent，S）。

例如，下面的命令将显示所有的“错误”和“致命性错误”，但滤掉了“警告”、“信息”、“调试”和“冗余”消息：

```
# logcat *:E
```

2. 根据标记进行过滤

你可以使用多个过滤器，这样能获得完整而精确的覆盖。比如，你想查看来自特定应用程序（特定标记）的消息，而滤掉其他的。那么在这种情况下，你可能需要创建一个过滤器来显示所有匹配给定标记的消息，还需要创建一个用于阻止其他消息的过滤器。我们仍然使用 time 模式，这样能够获得日志事件发生的日期和时间。下面的命令显示了所有带有 AppLog 标记的日志信息，并且阻止了其他标记的显示：

```
# logcat -v time AppLog:V *:S
```

上面这个过滤器大体上等价于下面的命令：

```
# logcat -v time AppLog:* *:S
```

返回的结果会带有日期时间，后面紧跟事件的严重性、标记、进程 ID 和消息：

```
01-05 21:52:22.465 I/AppLog(20054): Another Log Message.
```

C.7.4 清除日志

你可以使用-c 标签来清除日志：

```
adb -e logcat -c
```

或使用 ADB shell：

```
# logcat -c
```

C.7.5 将日志输出重定向至一个文件

你可以将日志的输入重定向至设备上的一个文件，这需要使用-f 标签。例如，要将来自模拟器的所有“信息”类日志消息（包括其他具有更高优先级的消息）重定向至 sdcard 目录下的 mylog.txt 文件，可以使用下面的 ADB shell 命令：

Download at http://www.pin5i.com/

```
# logcat -f /sdcard/mylog.txt *:I
```

Note

该文件存储于模拟器或设备中，你需要使用 ADB 或 DDMS 文件浏览器将其拖曳至桌面。

C.7.6 访问次级日志

Android 还拥有几个不同的日志，而默认情况下你所查看的为主日志，另外还存在事件日志（event log）和无线电日志（radio log）。你可以使用-b 标签来连接其他的日志缓冲。例如，要连接到事件日志以回顾所发生的事件，可以使用如下命令：

```
# logcat -b events
```

可以采用类似的方法访问无线电日志：

```
# logcat -b radio
```

C.8 控制备份服务

Android 2.2 引入了备份服务，应用程序可以使用它来备份重要的数据，以便能够在恢复出厂设置或者设备丢失后找回。这一服务通常在后台运行，备份和恢复数据都在自己的计划任务内执行。不过，你可以使用 bmgr Shell 工具来执行备份服务，这对测试备份功能非常有用。你可以使用如下 ADB Shell 命令来检测是否开启了备份服务（Backup Service）：

```
# bmgr enabled
Backup Manager currently disabled
```

你可以在 ADB Shell 中开启备份服务，如下所示：

```
# bmgr enable true
Backup Manager now enabled
```

Tip

用户可以通过 Settings→Privacy→Backup and Restore 菜单命令来启动或关闭设备上的备份管理器。

C.8.1 强制备份操作

在 ADB Shell 中，你可以使用下面的命令将备份操作添加到计划任务：

```
# bmgr backup <package>
```

例如，你可以将 SimpleBackup 应用程序的数据备份添加到计划任务中，如下所示：

```
# bmgr backup com.androidbook.simplebackup
```

上面这条命令只将备份操作添加到了计划任务中，它会在未来的某个时候进行。你也可以使用下面的命令来启动所有的备份任务：

```
# bmgr run
```

Download at http://www.pin5i.com/

C.8.2 强制恢复操作

在 ADB Shell 中，你可以使用下面的命令执行恢复操作：

```
# bmgr restore <package>
```

例如，你可以强制恢复 SimpleBackup 应用程序的数据，如下所示：

```
# bmgr restore com.androidbook.simplebackup
```

与备份命令不同，恢复命令会立即执行，并开始还原操作。

C.8.3 将存档数据清空

在 ADB Shell 中，你可以使用下面的命令将特定应用程序的存档数据清除：

```
# bmgr wipe <package>
```

例如，你可以将 SimpleBackup 应用程序的所有存档数据清空，如下所示：

```
# bmgr wipe com.androidbook.simplebackup
```

C.9 生成 Bug 报告

你可以使用 bugreport 命令来创建详尽的应用程序 bug 报告。例如，要打印在开发计算机上运行的单独模拟器的调试信息，可以使用如下命令：

```
adb -e bugreport
```

如果要打印通过 USB 连接的单独手机设备的调试信息，则可以使用下面的命令：

```
adb -d bugreport
```

C.10 使用 Shell 查看 SQLite 数据库

你可以在 ADB Shell 内使用标准的 sqlite3 数据库工具。这一工具使得你可以直接查看和操作位于模拟器上的 SQLite 数据库。附录 E“SQLite 快速入门”给出了 sqlite3 工具的完整说明。

C.10.1 在 ADB Shell 中进行应用程序压力测试

你可以在 ADB Shell 中使用 Exerciser/Monkey 工具来向指定应用程序发送随机用户事件。你可以把它想象成将你的手机（或模拟器）交给了一只猴子（或者一个小孩）并且允许它们随意按键，从而产生随机的事件——如果你的应用程序没有足够的耐受力的话可能就会崩溃。如果应用程序崩溃了，那么 Monkey 应用程序将终止运行并报告错误。这个工具对于 QA 人员来讲非常有用。

1. 让 Monkey 掌管你的应用程序

要运行 Monkey 工具，可以使用如下 ADB Shell 命令：

Download at http://www.pin5i.com/

```
# monkey -p <package> <options> <event count>
```

例如，要让“猴子”在模拟器的 GroceryList 应用程序中产生 5 个随机事件，可以使用如下命令：

```
adb -s emulator-5554 shell
# monkey -p com.androidbook.grocerylist 5
```

2. 听听 Monkey 怎么说

你可以使用冗余标签-v 来查看每一个生成的事件。例如，要查看所有发送到前面的 GroceryList 应用程序的事件，可以使用如下命令：

```
adb -s emulator-5554 shell
# monkey -p com.androidbook.grocerylist -v 5
```

下面给出了命令执行之后的重要输出：

```
:SendKey: 21    // KEYCODE_DPAD_LEFT
:Sending Trackball ACTION_MOVE x=-4.0 y=2.0
:Sending Trackball ACTION_UP x=0.0 y=0.0
:SendKey: 82    //KEYCODE_MENU
:SendKey: 22    //KEYCODE_DPAD_RIGHT
:SendKey: 23    //KEYCODE_DPAD_CENTER
:Dropped: keys=0 pointers=0 trackballs=0
// Monkey finished
```

从上面的输出你可以看到 Monkey 应用程序向 GroceryList 应用程序发送的 5 个事件：导航时间（左）、两次轨迹球事件、“Menu”按钮和之后的两次导航事件（右、中）。

3. 指导 Monkey 的行为

你可以指定 monkey 应用程序产生事件的类型。你基本上只需要给定不同类型事件的权重（百分比）即可。表 C.1 列出了可用的事件类型。

表 C.1 Monkey 的事件类型

事件类型	说明	默认百分比	命令行标记	在“冗余”模式下显示的事件 ID
触屏（Touch）	在某个屏幕位置的“按下（Down）”、“抬起（Up）”事件	15%	--pct-touch	0
移动（Motion）	在某个屏幕位置的“按下”，进而移动，最后在不同的屏幕位置“抬起”的事件	10%	--pct-motion	1
轨迹球（Trackball）	轨迹球事件，有时会跟随一次单击	15%	--pct-trackball	2
基本导航（Basic Navigation）	上、下、左、右	25%	--pct-nav	3
主导航（Major Navigation）	菜单、回退（Back）、方向键盘（DPAD）的“Center”键	15%	--pct-majornav	4
系统按键（System Key）	“Home”、音量、“Send”、“End”等	2%	--pct-syskeys	5

Download at http://www.pin5i.com/

续表

事件类型	说　明	默认百分比	命令行标记	在“冗余”模式下显示的事件 ID
Activity 切换	随机地切换到应用程序包中的其他 Activity	2%	--pct-appswitch	6
其他事件	按键，其他按钮	16%	--pct-anyevent	7

要使用混合不同类型的事件，你需要包含在表 C.1 中列出事件类型的命令行标签，并跟随指定的百分比。

```
# monkey [<command line flag> <percentage>...] <event count>
```

例如，要告诉 Monkey 只使用触屏事件，则可以使用如下命令：

```
# monkey -p com.androidbook.grocerylist --pct-touch 100 -v 5
```

或者假如你只需要“基本导航”和“主导航”事件，且概率相同：

```
# monkey -p com.androidbook.grocerylist --pct-nav 50 --pct-majornav 50 -v 5
```

就举这两个例子，你应该已经明白了。

4. 训练你的 Monkey 重复它的技巧

你可以使用“种子（seed）”选项来让随机事件可以被重复使用。“种子”特性允许你修改事件序列的一部分，之后你可以重新运行这一序列（比如为了检查 bug 的修改情况）。要设置“种子”，可以使用-s 标签。

```
monkey -p <package> -s <seed> -v <event count>
```

例如，在我们前面使用的命令中，可以通过设置不同的起始“种子”来修改这 5 个事件。在这里，我们将“种子”设为 555：

```
monkey -p com.androidbook.grocerylist -s 555 -v 5
```

改变“种子”则改变了 Monkey 产生的事件序列，所以，作为压力测试的一部分，你可以考虑生成随机的种子，并将其发送给 Monkey 并记录结果。如果应用程序在某个“种子”下出现了问题，则在记录 bug 的同时保留这一“种子”（以及其他所有的命令行选项，例如事件类型百分比），并且在修正 bug 之后重新使用它进行测试。

5. 别让 Monkey 逃出你的手掌

默认情况下，Monkey 将以尽可能快的速度产生事件。不过，你可以使用“阀门（throttle）”选项来减缓这一过程，如下所示：

```
# monkey --throttle <milliseconds> <event count>
```

例如，要发送给 GroceryList 应用程序的 5 次事件中每次事件之间暂停 1 秒（1000 毫秒），可以使用如下命令：

```
monkey -p com.androidbook.grocerylist -v --throttle 1000 5
```

6. 深入学习 Monkey

要获取更多有关 Monkey 应用程序和其他 Monkey 命令的信息，可以访问 Android SDK

Download at http://www.pin5i.com/

参考网站：http://developer.android.com/guide/developing/tools/monkey.html。你还可以不使用任何命令选项直接键入 monkey 来获取命令的列表，如下所示：

```
adb -e shell monkey
```

C.10.2 通过 Shell 安装和使用自定义二进制文件

你可以在模拟器或设备上安装自定义的二进制文件。例如，如果你经常与 Shell 打交道，那么可能需要安装一个 BusyBox，它是一个在 GNU General Public License 协议下的免费软件，它提供了一系列的命令行工具，被称为“嵌入式 Linux 中的瑞士军刀”（感谢维基百科提供这些资料）。BusyBox 提供了一系列熟悉而有用的 UNIX 功能，比如查找等，它们都被打包在了一个单独的二进制文件中。BusyBox 提供了 Android 上许多有用的功能（尽管有些功能可能不被许可），如下所示：

```
[, [[, addgroup, adduser, adjtimex, ar, arp, arping, ash,
    awk, basename, bunzip2, bzcat, bzip2, cal, cat, catv,
    chatter, chgrp, chmod, chown, chpassed, chpst, chroot,
    chrt, chvt, cksum, clear, cmp, comm, cp, cpio, crond,
    crontab, cryptpw, cut, date, dc, dd, deallocvt, delgroup,
    deluser, df, dhcprelay, diff, dirname, dmesg, dnsd, dos2unix, du, dumpkmap,
    dumpleases, echo, ed, egrep, eject, env, envdir, envuidgid, ether-wake,
expand,
    expr, fakeidented, false, fbset, fdflush, fdformat, fdisk, fgrep, find, fold,
    free, freeramdisk, fsck, fsck.minix, ftpget, ftpput, fuser, getopt, getty,
grep,
    gunzip, gzip, halt, hdparm, head, hexdump, hostid, hostname, httpd, hwclock,
id,
    ifconfig, ifdown, ifup, inetd, init, insmod, install, ip, ipaddr, ipcalc,
ipcrm,
    ipcs, iplink, iproute, iprule, iptunnel,kbd_mode, kill, killall, killall5,
    klogd, last, length, less, linux32, linux64, linuxrc, ln, loadfont, loadkmap,
    loger, login, logname, logread, losetup, ls, lsattr, lsmod, lzmacat,
makedevs,
    md5sum, mdev, mesg, microcom, mkdir, mkfifo, mkfs.minix, mknod, mkswap,
mktemp,
    modprobe, more, mount, mountpoint, mt, mv, nameif, nc, netstat, nice, nmeter,
    nohup, nslookup, od, openvt, passwd, patch, pgrep, pidof, ping, ping6,
    pipe_progress, pivot_route, pkill, poweroff, printenv, printf, ps, pscan,
pwd,
    raidautorun, rdate, readlink, readprofile, realpath, reboot, renice, reset,
    resize, rm, rmdir, rmmod, route, rpm, rpm2cpio, run-parts, runlevel, runsv,
    runsvdir, rx, sed, seq, setarch, setconsole, setkeycodes, setlogcons,
setsid,
    setuidgid, sh, sha1sum, slattach, sleep, softlimit, sort, split,
    start-stop-daemon, stat, strings, stty, su, sulogin, sum, sv, svlogd,
swapoff,
    swapon, switch_root, sync, sysctl, syslogd, tail, tar, taskset, tcpsvd, tea,
    telnet, telnetd, test, tftp, time, top, touch, tr, traceroute, true, tty,
    ttysize, udhcpc, udhcpd, udpsvd, umount, uname, uncompress, unexpand, uniq,
    unix2dos, unlzma, unzip, uptime, usleep, uudecode, uuencode, vconfig, vi,
vlock,
    watch, watchdog, wc, wget, which, who, whoami, xargs, yes, zcat, zcip
```

所有你需要做的就是安装这一二进制文件（可以从网上下载它），步骤如下。

Download at http://www.pin5i.com/

① 下载 BusyBox 二进制文件。你可以在下面的地址中找到它：http://benno.id.au/blog/2007/11/14/android-busy-box，这是 Benno 为你提供的。（谢谢 Benno!）

② 使用 ADB Shell 在模拟器上生成名为/data/local/busybox/的文件夹。用于执行这一操作的命令可以是：adb -e shell mkdir /data/local/busybox/。

③ 将 BusyBox 二进制文件拷贝到你刚刚创建的文件夹中，例如：adb -e push C:\busybox /data/local/busybox/busybox。

④ 运行 ADB Shell，例如：adb -e shell。

⑤ 导航至 BusyBox 目录，例如：# cd /data/local/busybox。

⑥ 修改 BusyBox 文件的权限，例如：# chmod 777 busy-box。

⑦ 安装 BusyBox，例如：# ./busybox -install。

⑧ 为了使用方便导出路径。注意：你需要为每一个会话重新设置 PATH，例如：# export PATH=/data/busybox:$PATH。

你可以在 http://www.busybox.net 中找到更多有关 BusyBox 的资料。

C.11 其他 ADB 命令探索

除了前面我们介绍的 ADB 命令之外，还有许多方便实用的 ADB 命令，比如用于配置端口映射（port forwarding）和通过 USB 运行 PPP 的 ADB 网络命令。你可以使用端口映射通过 ID 来将 Java 调试器链接到特定的 JDWP 进程。要获得所有 ADB 命令，可以键入：

```
adb help
```

Download at http://www.pin5i.com/

附录 D

Eclipse 技巧和诀窍

Eclipse IDE 是目前非常流行的 Android 地开发环境。在这篇附录中，我们提供了一些 Eclipse 使用技巧和诀窍，帮助开发人员快速高效地开发 Android 应用程序。

D.1 构建你的 Eclipse 工作空间

在这里，我们列举了一些技巧和诀窍，可以帮助你构建一个最适合于 Android 开发的 Eclipse 工作空间。

D.1.1 集成源代码管理服务

使用 Eclipse 的附加组件和插件，可以集成很多源代码管理包。这使得 Eclipse 能够在你首次编辑代码、提交代码、查看代码状态以及进行其他操作的时候，将代码签出到本地，即让代码允许被编辑。

Tip

通用源代码管理的附加组件，目前有 CVS、Subversion、Perforce、git、Mercurial 等其他许多包。

一般来说，不是所有文件都适用于源代码管理。对 Android 项目来说，任何包含 bin 和 gen 目录的文件都不应该放在源代码管理中。你可以在 Eclipse 中选择 Windows→Preferences→Team→Ignored Resources 设置排除这些文件，点击 Add Pattern 按钮，添加文件后缀如*.apk、*.ap_和*.dex，这样就能对所有源代码管理服务设置生效。

D.1.2 调整视图中的标签面板

Eclipse 默认视图提供了一些相当不错的布局。然而这并不总适用于每个人。某些视图的布局对于 Android 开发来说并不合适，需要做些调整。

Tip

尝试找到一个最适合你开发的面板布局。每一种视图都有自己的布局，而且视图可以是面向任务的。

例如，Properties 标签面板通常出现在视图的下方。对于编码而言这样很合理，因为标签面板只占几行高。但是，要对 Android 进行资源编辑时，这样就不太方便了。幸运的是，在

Download at http://www.pin5i.com/

Eclipse 中，这很容易调整：只需单击标签面板（的标题），并按住不放，拖动到一个新的地方就行了，比如 Eclipse 窗口的右边栏上。这样每次查看数十个属性时，就有了充足的空间。

> **Tip**
>
> 如果你弄乱了视图，或者只是想重新开始，你可以通过选择 Windows→Reset Perspective 来重置面板。

D.1.3 最大化窗口

有时你会觉得编辑器窗口太小了，特别是当所有其他数据窗口和标签面板都围绕在四周的时候。不妨试一下：双击你想要编辑的代码文件标签。“砰”，你会发现窗口扩大到接近 Eclipse 窗口的大小。要想恢复原样，只需再次双击即可。

D.1.4 最小化窗口

你也可以最小化所有面板。例如，如果你不需要底部的控制台面板或左侧的文件浏览视图，你可以通过每个面板右上角的按钮实现最小化。点击那个看起来像两个小窗口的按钮可以复原。

D.1.5 显示并排窗口

你是否有时希望能够同时显示两个源文件？没问题。只需点击其中一个源文件的标签，将其拖动到编辑器的两侧或底部。这时在另一个文件的旁边、顶部或底部，你会看见一圈黑色的轮廓，显示文件将要被放置到的地方。这样就形成了一块平行的编辑区域，同样，你也可以继续拖动其他的文件标签。如此重复多次，你就可以同时浏览 3、4 个或者更多的文件窗口。

D.1.6 显示同一文件的两个部分

你有没有曾经想要同时查看一个源文件里的两处地方？当然可以。右键单击文件标签，在弹出菜单中选择 New Editor，将会出现该文件的第二个标签。再使用上一条技巧，你就能够在两个窗口中显示同样的源文件了。

D.1.7 关掉不需要的标签面板

是不是经常发现，你同时打开了很多已经不再需要编辑的文件标签？我经常这样。有很多办法可以解决这个问题。首先，你可以右键单击一个文件标签，选择 Close Others 来关闭其他打开的文件。你也可以用中键点击某个标签，从而快速关闭一个文件。（在 Mac 电脑上，可以用一个带中键或者滚轮的鼠标来实现。）

D.1.8 保持对窗口的控制

现在，你可以用 Eclipse 来设置限制文件编辑器的打开数量。

① 打开 Eclipse 的 Preferences 对话框。

② 展开 General，选择 Editors，然后勾选 Close Editors Automatically。

③ 设置 Number of Opened Editors Before Closing 框中的值。

Download at http://www.pin5i.com/

可以将 Number of Opened Editors Before Closing 的值设成 8，这样既不会显得杂乱，又能有足够的编辑器来完成工作，同时打开参考代码。另外，如果你勾选了 When All Editors Are Dirty or Pinned 下的 Open New Editor，那么如果你当前编辑的数量超出预设值时，则会打开新的文件标签。因此，当你需要同时编辑很多文件时，这样的设置不仅不会影响工作效率，而且能在绝大多数情况下让事情井然有序。

D.1.9 定制日志过滤器

每一个 Android 日志声明都会有一个标记。你可以通过在 LogCat 中定义来使用这些标记。点击 LogCat 面板上的绿色加号按钮新增一个过滤器。给过滤器命名（可以用标记的名字），然后填入你想要过滤的标记。这时 LogCat 上将会增加一个新标签，用以显示含有这个标记的消息。另外，你也可以创建一个按照严重级别显示的过滤器。

在 Android 的约定中，习惯上用类的名字来创建标记。这种习惯也经常在本书提供的代码中出现。需要注意的是，我们在每个类中都创建了一个变量名相同的常量，使每次日志调用更简单。类似下面这个例子：

```
public static final String DEBUG_TAG = "MyClassName";
```

当然，并非一定要遵循这个约定。你可以围绕那些跨越多个 Activity 的任务组织标记，或者也可以用任何你需要的其他形式结构。

D.2 编写 Java 代码

这里，我们列举了一些技巧和诀窍，帮助你编写 Android 应用程序的代码。

D.2.1 善用自动补全（Auto-Complete）

自动补全是一个非常棒的功能，它能够提高编码的速度。如果你还没有用过或者不小心关掉了，可以用 Ctrl+空格键来开启这个功能。自动补全不但可以节约输入时间，它还能够帮助你想起某个方法的名称，或者找到一个新的方法。你可以用滚轮浏览一个类的所有方法，甚至包括与这些方法相关的 Javadocs。通过类名或者实例变量名，你可以很方便地找到对应的静态方法。在类名或者变量名后输入一个点（也可以按 Ctrl+空格键），就会出现滚动清单。然后你只需键入前几个字母，就能找到想要的结果。

D.2.2 格式化代码

Eclipse 内置一套格式化 Java 代码的处理机制。使用格式化工具，对保持代码风格一致、新样式套用以及匹配客户端或目标风格（例如书或文章）都有很大帮助。

如果要快速格式化一段代码，可以先选中代码，然后在窗口中按 Ctrl+Shift+F（在 Mac 上可以使用 Command+Shift+F），就能以当前设置格式化代码。如果不选中任何代码，则对整个文件进行格式化。有时，你必须选择整段代码（比如一整个方法）才能让格式化后的缩进水平和括号配对符合你的需要。

Eclipse 格式化设置可以在 Java→Code Style→Formatter 的属性面板中找到。这里可以为每个项目或者工作区设置基础格式。你也可以根据自己的风格来修改和应用这些规则。

Download at http://www.pin5i.com/

D.2.3 创建类

你可以按照下面的方法快速创建一个新类并生成相应的源文件：右键单击创建包，选择New→Class，然后填入类的名称，并选择其父类和接口，最后选择是否为父类的构造器或抽象类自动生成注释和方法存根。

D.2.4 创建方法存根（stub）

跟创建类一样，你同样也可以快速地创建方法存根：右键单击一个类，或者在编辑器里点击右键，选择 Source→Override/Implement Methods，依次选择你要创建存根的方法、插入点以及是否自动生成注释。

D.2.5 组织导入

当你在代码中首次引用了一个类时，你可以将鼠标移到新类名上，然后选择 Import“类名”（包名），Eclipse 会自动加入适当的导入声明。

除此之外，使用 Organize Imports 命令（Windows 快捷键 Ctrl+Shift+O，Mac 快捷键 Cmd+Shift+O）能够让 Eclipse 自动组织导入。Eclipse 会移除从未使用过的导入，并在包中插入那些已使用但尚未声明的导入。

如果在自动导入时，类的名称不明确，比如 Android Log 类，Eclipse 就会提示你导入包。最后，你还可以设置 Eclipse 在每次保存文件时自动组织导入，可以配置成是对整个工作空间生效，还是仅对某个单独的项目生效。

当你同时跟进多个项目，不想因此改变某些代码时，针对单独项目的配置会让你更灵活，尽管这些改变是有好处的。你可以按照下面的步骤进行设置。

① 右键单击项目名称，选择 Properties。

② 展开 Java Editor，点击 Save Actions 按钮。

③ 分别勾选 Enable Project Specific Settings、Perform the Selected Actions on Save 和 Organize Imports。

D.2.6 重命名

Eclipse 的重命名功能非常强大。你可以用它重命名变量、方法、类名以及其他东西。通常来说，你只需右键单击你想要重命名的项，然后选择 Refactor→Rename。或者你也可以在选择待改项后，按 Windows 快捷键 Ctrl+Alt+R（或者 Mac 上的 Cmd+Alt+R）开始重命名。如果你重命名了一个文件中的顶级类，那么其文件名也必须做同样的修改，因为如果文件受控于源代码控制，那么 Eclipse 处理代码更改时，通常会要求进行这样的操作。如果 Eclipse 能确定与被重命名项有关的其他同名项，那么所有相关实例都会被重命名。所以有时候连注释中的名称也会被更新，确实是很方便的功能。

D.2.7 重构代码

如果你发现自己写的代码中，有很多看上去重复的部分，比如像这种情况：

```
TextView nameCol = new TextView(this);
nameCol.setTextColor(getResources().getColor(R.color.title_color));
```

Download at http://www.pin5i.com/

```
nameCol.setTextSize(getResources().getDimension(R.dimen.help_text_size));
nameCol.setText(scoreUserName);
table.addView(nameCol);
```

这段代码设置了文字颜色、字体大小和文本内容。如果你已经写了两个或者更多像这样的代码段，那么重构无疑将能有效地改善你的代码。Eclipse 中的两个很有用的工具——本地变量提取（Extract Local Variable）和方法提取（Exact Method）——将会帮助你轻松、高效地完成这一过程。

1. 提取本地变量

按照下面的步骤提取本地变量。

① 选取表达式 `getResources().getColor(R.color.title_color)`。

② 右键单击并选择 Refactor→Exact Local Variable（或者按快捷键 Ctrl+Alt+L）。

③ 在出现的对话框中，键入变量名并勾选 Replace All Occurrences，然后点击 OK 按钮，欣赏神奇的变化。

④ 对字体大小重复以上步骤。

最后结果应该像这样：

```
int textColor = getResources().getColor(R.color.title_color);
float textSize = getResources().getDimension(R.dimen.help_text_size);
TextView nameCol = new TextView(this);
nameCol.setTextColor(textColor);
nameCol.setTextSize(textSize);
nameCol.setText(scoreUserName);
table.addView(nameCol);
```

可以看到后五行代码中所有重复的部分都被修改了，很方便吧！

2. 提取方法

现在我们可以进行第二步了，按照下面的步骤提取方法：

① 选取第一个代码块的全部五行代码。

② 右键单击并选择 Refactor→Exact Method（或者用快捷键 Ctrl+Alt+M）。

③ 为方法命名，并将变量名设成任何你想要的名称（如果需要的话，也可以调整它们的上下位置），然后点击 OK 按钮，欣赏神奇的变化。

默认情况下，新的方法会出现在当前方法的下方。如果有其他的代码块与之相同，这里的相同意味着语句必须是同样的顺序，所有类型完全一致，诸如此类。那么这些代码块同样也会被新方法的调用替换掉！你可以在提取方法的对话框中，看到这些额外情况总共发生的次数。如果这个数字跟你预期的不太一样，不妨检查一下那些形式完全相同的代码。至此，我们得到了类似这样的一段代码：

```
addTextToRowWithValues(newRow, scoreUserName, textColor, textSize);
```

相比原始代码，重构后的代码更加易于使用，而且关键是重构的过程中，我们没有敲入任何代码！如果你有十个实例需要进行重构，那么使用 Eclipse 的这些功能，将帮助你节约大量的时间。

Download at http://www.pin5i.com/

D.2.8 整理代码

有些时候，格式化后的代码还是显得不够整洁和易于阅读。当你完成一个复杂功能的开发后，可能会发现有很多内部类和方法各自散乱在文件里。好在有个快捷的 Eclipse 技巧能够解决这个问题：打开需要处理的文件，记得同时显示大纲视图。

你只需要在大纲视图中，选择并拖动方法和类，将它们以合适的逻辑顺序进行排列。是否发现有个方法仅仅被某个特定的类调用，却对所有类可见？只要把它拖动到那个类中。几乎所有大纲中列的项都能用这个办法，包括类、方法以及变量。

D.2.9 提供 Javadoc 风格的文件

规范的代码注释非常有用（如果是正确的话）。Javadoc 风格的注释会在代码自动补全的对话框以及其他地方出现，这让它们显得更有用。为一个方法或类快速地添加 Javadoc 注释，只需要按 Windows 快捷键 Ctrl+Shift+J（在 Mac 上按 Cmd+Alt+J）。或者你也可以选择 Source→Generate Element Comment 在某块区域预先生成 Javadoc，例如参数名和作者，可以快速创建这种风格的注释。

D.2.10 解决诡异的构建错误

有些时候，你会注意到 Eclipse 会在刚刚还没有问题的地方出现构建错误。遇到这种情况，你可以尝试两个 Eclipse 小技巧。

第一，尝试刷新整个项目：只需右键单击项目名然后选择 Refresh 或者直接按 F5 键。如果这样不起作用，试着在 gen 文件夹下找到被编译的包名，删除其中的 R.java 文件。（不用担心，每一次编译都会重新创建这个文件。）如果启用了 Compile Automatically 选项，该文件则被重新创建。不然的话，你需要重新编译一次项目。

第二，你可以试试清洗项目。点击 Project→Clean，然后选择需要清洗的项目。Eclipse 会移除所有临时文件，并且重新构建项目。如果这是一个 NDK 项目，不要忘了重新编译原代码。

> **Note**
>
> 将你在 Eclipse 上的 Android 开发技巧和诀窍发给我们吧！你可以发送电子邮件到 androidwirelessdev+awad2e@gmail.com。

Download at http://www.pin5i.com/

附录 E

SQLite 快速入门

Android 系统允许单独的应用程序拥有其私有的 SQLite 数据库，以存储应用程序数据。这里提供的“快速入门”并不是 SQLite 命令的完整文档，而是提供给你用于快速入门并执行常规的任务。在第一部分中介绍了 `sqlite3` 命令行工具的特性；之后我们通过一个数据库示例来讲解较为深入的 SQLite 命令。你可以查看 SQLite 在线文档（www.sqlite.org），以了解完整的 SQLite 特性、功能和局限。

E.1 探索 SQLite 的常见任务

SQLite 是一个压缩的、轻量级的、强大的嵌入式关系型数据库引擎，它空间占用低、运行效率高，成为了手机系统上的完美数据库引擎。不同于诸如 Oracle、Microsoft SQL Server 这样运行在数据库服务器上的重量级数据库，SQLite 数据库仅仅包含在磁盘上的一个文件中。

Android 应用程序将其私有数据库（SQLite 或其他）存储在特定的应用程序目录中：

`/data/data/<application package name>/databases/<databasename>`

例如，本书提供的 `PetTracker` 应用程序所使用的数据库就存储在如下目录中：

`/data/data/com.androidbook.PetTracker/databases/pet_tracker.db`

数据库采用了标准的文件格式，可以跨平台使用。你可以使用 Dalvik 调试监视服务（Dalvik Debug Monitor Service，DDMS）中提供的文件浏览器（File Explorer）取出数据库文件，并使用第三方工具进行查看。

> **Tip**
>
> 针对应用程序的 SQLite 数据库为其所私有的，也就是说只能够在应用程序中访问它。如果需要与其他应用程序共享数据，那么应用程序则必须成为一个内容提供器。第 11 章“使用内容提供器在应用程序间共享数据”中介绍了内容提供器的有关内容。

E.2 使用 `sqlite3` 命令行接口

除了在应用程序中使用程序创建和使用 SQLite 数据库（在第 10 章“使用 Android 数据和存储 API”中介绍）以外，你还可以使用熟悉的命令行 `sqlite3` 工具操作数据库，它可

Download at http://www.pin5i.com/

以在 Android 调试桥（Android Debug Bridge，ADB）的远程 Shell 中获得。

SQLite 的命令行接口称为 sqlite3，它由 ADB 工具提供。有关 ADB 的其他内容可以参考附录 C“Android 调试桥快速入门”。

E.2.1 运行 ADB Shell

在使用 sqlite3 命令之前，你必须首先在特定模拟器或设备上运行 ADB Shell 接口（如果安装了的话）。如果只有一个 Android 设备（或模拟器）处于运行当中，那么你可以简单地通过如下命令连接到 ADB Shell：

```
c:\>adb shell
```

而如果你想要连接到特定的模拟器实例，则可以使用下面的命令：

```
adb -s <serialNumber> shell
```

例如，要连接位于端口 5554 的模拟器，可以使用下面的命令：

```
adb -s emulator-5554 shell
```

要获取如何查看特定模拟器或设备实例序列号的更多信息，可以参考附录 C。

E.2.2 连接数据库

现在，你可以通过数据库名称连接到 Android 应用程序的数据库了。例如，要连接我们在 PetTracker 应用程序中创建的数据库，可以像下面这样操作：

```
c:\>adb shell
# sqlite3 /data/data/com.androidbook.PetTracker/databases/pet_tracker.db
SQLite version 3.6.22
Enter ".help" for instructions
sqlite>
```

现在，我们已经看到了 sqlite3 的命令提示，之后就可以向其发布命令了。任何时候你都可以通过键入下面的命令来退出接口：

```
sqlite>.quit
```

或

```
sqlite>.exit
```

sqlite3 程序的命令都以一个点号（.）开头，以与可以直接从命令行中执行的 SQL 命令相区别。这一语法可能与你熟悉的其他应用程序（例如 mysql 命令）有所不同。

Warning

绝大部分 Android 设备不能像模拟器一样直接使用 sqlite3 命令，不过安装了的设备可以提供支持。

E.2.3 探索你的数据库

你可以使用 sqlite3 命令来探索你的数据库，并且操作它们。你可以：

① 列出可用的数据库：

Download at http://www.pin5i.com/

② 列出可用的表；
③ 浏览所有给定表的索引；
④ 显示数据库模式（schema）。

1. 列表显示可用的数据库

你可以列出数据库的名称及其存放的位置。通常，你会拥有一个主数据库和临时数据库，临时数据库用于存放临时的表。可以使用如下命令列出这些信息：

```
sqlite> .databases
seq  name file
---  ---- ----------------------------------------------
0    main /data/data/com.androidbook.PetTracker/databases/...
1    temp
sqlite>
```

2. 列表显示可用的数据库表

你可以列出已连接数据库中所有的数据库表，命令如下：

```
sqlite> .tables
android_metadata  table_pets        table_pettypes
sqlite>
```

3. 列表显示表索引

你可以通过下面的命令列出给定表的索引：

```
sqlite>.indices table_pets
```

4. 列表显示表的数据库模式（Schema）

你可以列出给定表的模式，命令如下：

```
sqlite>.schema table_pets
CREATE TABLE table_pets (_id INTEGER PRIMARY KEY
AUTOINCREMENT,pet_name TEXT,pet_type_id INTEGER);
sqlite>
```

5. 列表显示数据库的模式

你可以使用如下命令列出整个数据库的模式：

```
sqlite>.schema
CREATE TABLE android_metadata (locale TEXT);
CREATE TABLE table_pets (_id INTEGER PRIMARY KEY
AUTOINCREMENT,pet_name TEXT,pet_type_id INTEGER);
CREATE TABLE table_pettypes (_id INTEGER PRIMARY KEY
AUTOINCREMENT,pet_type TEXT);
sqlite>
```

E.2.4 导入和导出数据库

你可以使用 sqlite3 命令来导入和导出数据库数据及模式，并与其进行交互。你可以：

Download at http://www.pin5i.com/

① 将命令输入发送到文件而非 STDOUT（即屏幕）；
② 以 SQL 脚本的方式转储数据库内容（这样可以在今后恢复它们）；
③ 从文件执行 SQL 脚本；
④ 将数据从文件导入到数据库中。

Note

文件路径位于你的 Android 设备上，而非计算机中，所以你需要在 Android 设备上找到一个有读写许可权限的目录。例如：/data/local/tmp/就是一个共享文件夹。

1. 将输入发送到文件

经常地，你需要将 sqlite3 命令的执行结果导出成文件而非显示在屏幕上。要进行这一操作，你只需要键入 output 命令，并且在后面紧跟 Android 系统中的目标路径即可。如下所示：

```
sqlite>.output /data/local/tmp/dump.sql
```

2. 转储数据库内容

通过 dump 命令，你可以使用 SQL 脚本来创建表及其内含数据。dump 命令将创建一个事务（transaction），它包括 CREATE TABLE 和 INSERT 等调用以填充数据库中的数据。这个命令可以接受传入一个可选的表名称，或者转储整个数据库。

Tip

dump 命令是实现数据库档案备份的最好方式。

例如，下面的命令将把 table_pets 表中的数据导出到文件中，之后又将输出模式调整口控制台。

```
sqlite>.output /data/local/tmp/dump.sql
sqlite>.dump table_pets
sqlite>.output stdout
```

你可以使用 DDMS 和文件浏览器来提取存储在 Android 文件系统中的 SQL 文件。上面生成的 dump.sql 文件内容大致如下：

```
BEGIN TRANSACTION;
CREATE TABLE table_pets (
_id INTEGER PRIMARY KEY AUTOINCREMENT,
pet_name TEXT,
pet_type_id INTEGER);

INSERT INTO "table_pets" VALUES(1,'Rover',9);
INSERT INTO "table_pets" VALUES(2,'Garfield',8);
COMMIT;
```

3. 从文件执行 SQL 脚本

你可以创建 SQL 脚本文件并且在控制台中运行它们。这些脚本必须存储在 Android 文件系统中。例如，我们在 Android 文件系统的/data/local/tmp/目录下存放一个名为

Download at http://www.pin5i.com/

myselect.sql 的 SQL 脚本文件。文件共有 2 行：

```
SELECT * FROM table_pettypes;
SELECT * FROM table_pets;
```

之后，我们可以使用如下命令运行这一 SQL 脚本：

```
sqlite>.read /data/local/tmp/myselect.sql
```

你将在命令行中看到运行的结果。

4. 导入数据

你可以使用导入和分隔符命令来执行格式化数据的导入。如同 CSV 这样的文件使用逗号作为数据分隔符，但是在其他的数据格式中，也可能使用空格或制表符（tab）。你可以使用分隔符命令来指定所使用的分隔符，同时使用导入命令来指定需要导入的数据文件。

例如，在 Android 文件系统的/data/local/tmp/目录下存放一个名为 some_data.csv 的 CSV 脚本文件。文件共有 4 行，且使用逗号进行数据分隔，它列出了宠物类型的 id 和名称：

```
18,frog
19,turkey
20,piglet
21,great white shark
```

之后，你可以将这些数据导入到 table_pettypes 表中，这个数据库表拥有两个字段：_id 字段和宠物类型名字段。要导入上面的数据，可以使用如下命令：

```
sqlite>.seperator ,
sqlite>.import /data/local/tmp/some_data.csv table_pettypes
```

现在，如果你查询表，将发现它新增了 4 行记录。

E.2.5 在命令行中执行 SQL 命令

你也可以在命令行中执行原始的 SQL 命令。只需要简单地敲入 SQL 命令，并确保以分号（“;”）结束即可。如果你需要执行数据库的查询，那么可能需要将输入模式改变为“列”方式，这样查询结果显得更清晰易读（以列的方式），同时会打印出字段头（字段名称）。例如：

```
sqlite> .mode colum
sqlite> .header on
sqlite> select * from table_pettypes WHERE _id < 11;
_id         pet_type
----------  ----------
8           bunny
9           fish
10          dog
sqlite>
```

这里能够执行的命令并不局限于查询。如果愿意的话，可以在命令行中执行所有你在 SQL 脚本中看到的 SQL 命令。

Download at http://www.pin5i.com/

Tip

如果使用 QueryBuilder 生成的 Android SQL 查询没能发挥作用的话，使用 sqlite3 命令行来测试 SQL 查询是很有帮助的，特别是用在比较复杂的查询当中。

另外，你还可以使用 width 命令来控制每个列的宽度（这样可以避免将文本字段进行截尾）。例如，下面的命令将使查询结果的格式为：使用 5 字符宽显示第 1 列（通常是 ID 字段），继而使用 50 字符宽显示第 2 列（文本字段）。

```
sqlite> .width 5 50
```

Warning

SQLite 将其数据库模式保存在一个名为 sqlite_master 的特殊表中。你可以认为这个表是只读的。另外，SQLite 将临时表存储在一个名为 sqlite_temp_master 的特殊表中，它本身也是一个临时表。

E.2.6 使用其他的 sqlite3 命令

可以通过键入如下命令获得 sqlite3 的完整命令列表：

```
sqlite> .help
```

E.2.7 认识 SQLite 的局限

SQLite 虽然很强大，但它同传统的 SQL Server 数据库相较而言还是存在几个不可忽视的局限。下面列出了这些局限性。

① SQLite 并不能替代高性能的、服务器驱动的数据库。

② 由于它基于文件（file-based），所以只能串行地访问，而不能以并发的（concurrent）方式进行。它只有一个用户——Android 应用程序。不过它也具有某些并发特性，但仍然具有一定的限制。

③ 对 SQLite 数据库的访问是由文件权限控制的，而非数据库用户权限。

④ 并不进行引用完整性（referential integrity）的维护。例如，FOREIGN KEY 约束能够被解析（例如，在 CREATE TABLE 中）但并不自动强制。不过，可以使用 TRIGGER 函数来强制这一特性。

⑤ 对 ALTER TABLE 的支持有局限性。你只能使用 RENAME TABLE 和 ADD COLUMN，而不能删除或修改列，也不能进行其他类似的操作。这使得数据库的修改工作变得比较棘手。

⑥ 对 TRIGGER 的支持具有局限性。你不能使用 FOR EACH STATEMENT 或 INSTEAD OF。你不能使用递归（recursive）触发器。

⑦ 不能嵌套 TRANSACTION 操作。

⑧ VIEW 是只读的。

⑨ 不能使用 RIGHT OUTER JOINN 和 FULL OUTER JOIN。

⑩ SQLite 不支持 STORED PROCEDURE 或审核（auditing）。

⑪ 对 SQL 语言内建 FUNCTION 的支持具有一定的局限性。

Download at http://www.pin5i.com/

⑫ 对于最大数据库尺寸、表尺寸和行尺寸的限制，可以参考 SQLite 文档，包括 Omitted（省略的）SQL 页面（www.sqlite.org/omitted.html）和 Unsupported（不支持的）SQL 页面（www.sqlite.org/cvstrac/wiki?p=UnsupportedSql）。

E.3 通过实例学习：学生成绩数据库

现在，我们通过一个教师使用的“Grades（成绩）”数据库示例来讲解标准的 SQL 命令，演示数据库的创建和操作。虽然你可以使用 `sqlite3` 命令行来创建这个数据库，但我们推荐使用 Android 应用程序来创建一个空的 Grades 数据库，这样它能够以标准的“Android”方式创建。

我们的目标：学生成绩数据库的目的在于记录班级中每位学生的测试成绩。学生的成绩由以下几个部分综合组成：

① 4 次小测验（每次权重 10%）；
② 1 次期中测试（权重 25%）；
③ 1 次期末测试（权重 35%）。

所有成绩的计分均使用百分制。

E.3.1 设计学生成绩数据库的模式

教师的“Grades”数据库拥有 3 个表：Students（学生）、Tests（测试）和 TestResults（测试结果）。

Students 表包含每个学生的信息。Tests 表包含每次测试的信息，以及其对学生总成绩影响的权值。最后，所有学生的总成绩存储在 TestResults 表中。

`sqlite3` 拥有对于以下几种字段数据类型的支持：

① INTEGER（带符号整数）；
② REAL（浮点值）；
③ TEXT（UTF-8 或 UTF-16 字符串，使用数据库编码方式进行编码）；
④ BLOB（数据块）。

Tip

不要在数据库中存储类似图像这样的文件，而应根据其在应用程序文件目录中的位置，存储其文件名或 URI 路径。

E.3.2 结合 AUTOINCREMENT 创建简单表

首先，我们创建 Students 表。对于每一个学生，需要一个学生 id 来代表。我们可以将它设为主键并且具有自增（AUTOINCREMENT）特性。同时，我们还需要学生的姓（lastname）和名（firstname），而且这些字段都不能缺省（不能为 null）。下面是 SQL 表达式：

```
CREATE TABLE Students (
id INTEGER PRIMARY KEY AUTOINCREMENT,
fname TEXT NOT NULL,
lname TEXT NOT NULL );
```

Download at http://www.pin5i.com/

对于 Tests 表，我们需要一个 id 来代表每一个测试或小测验，这与 Students 表是类似的。同时，我们还需要两个字段来分别存储每一次测试的名称以及其对学生总成绩影响的权重（以百分比的方式）。下面是 SQL 表达式：

```
CREATE TABLE Tests (
id INTEGER PRIMARY KEY AUTOINCREMENT,
testname TEXT,
weight REAL DEFAULT .10 CHECK (weight<=1));
```

E.3.3 向表中插入数据

在我们继续之前，一起来看几个如何向表里插入数据的例子。要向 Students 表中插入数据，你需要按顺序指定列名称及其对应值。例如：

```
INSERT into Students
(fname, lname)
VALUES
('Harry', 'Potter');
```

现在，我们可以向这个表中再加入几条记录，比如“Ron”和“Hermione”。同时，我们还需要向 Tests 表中加入数据。首先加入期中测试，它占总成绩的 25%，如下所示：

```
INSERT into Tests
(testname, weight)
VALUES
('Midterm', .25);
```

然后再加入几次小测验，它们使用默认的 10%作为权值。

```
INSERT into Tests (testname) VALUES ('Quiz 1');
```

最后，再加入占总成绩 35%的期末测试。

E.3.4 使用 SELECT 查询表数据

我们怎么知道数据已经添加到了表中呢？这很简单。我们只需要使用 SELECT 表达式查询表中的所有列即可：

```
SELECT * FROM Tests;
```

它将返回 Tests 表中的所有记录：

```
id     testname          weight
-----  ----------------  ------
1      Midterm           0.25
2      Quiz 1            0.1
3      Quiz 2            0.1
4      Quiz 3            0.1
5      Quiz 4            0.1
6      Final             0.35
```

现在，我们可以使用 SUM 求和函数来累加表中所有的权值，检验其是否等于 1.0：

```
SELECT SUM(weight) FROM Tests;
```

这将返回 Tests 表中所有权值之和：

```
SUM(weight)
```

Download at http://www.pin5i.com/

```
-----------
1.0
```

我们还可以创建我们自己的列并为其添加别名（alias）。例如，我们可以创建一个名为 fullname 的计算列，它使用“||”连接符将学生的 Firstname 和 Lastname 连接起来。

```
SELECT fname||' '||lname AS fullname, id FROM Students;
```

这将返回如下结果：

```
fullname          id
------------      --
Harry Potter       1
Ron Weasley        2
Hermione Granger   3
```

E.3.5 创建具有外键和复合主键的表

现在我们已经设置好了学生和每次测试的信息，下面创建 TestResults 表。它比之前的表更复杂。它是一个“学生—测试”相组合的列表，并且包含分数。

TestResults 表将取自 Students 表的学生 id 与取自 Tests 表的测试 id 组合起来。链接到这两个表的列通常被称为“外键（foreign keys）”。由于我们需要唯一的“学生—测试”组合，所以需要创建复合的主键，它们分别来自“学生”和“测试”外键。最后，我们需要强制约束总成绩为 0～100 的整数。在这个班级里没有额外学分也不允许补考！

```
CREATE TABLE TestResults (
studentid INTEGER REFERENCES Students(id),
testid INTEGER REFERENCES Tests(id),
score INTEGER CHECK (score<=100 AND score>=0),
PRIMARY KEY (studentid, testid));
```

Tip

SQLite 并不强制外键约束，但你可以设置它们并通过创建触发器来进行强制约束。在 SQL 中使用触发器进行强制约束的例子可以查看本书网站中针对第 10 章提供的 FullDatabases 工程。

现在，我们向这个表中插入一些数据。假设 Harry Potter 期中测试得了 82 分：

```
INSERT into TestResults
(studentid, testid, score)
VALUES
(1,1,82);
```

进而，我们输入其余学生的得分。Harry 是个好学生，而 Ron 要差一点，Hermione 当然在每次测试中都能得到高分。当他们的成绩都被添加后，就可以列表显示出来了。我们可以使用 SELECT *来获得所有的列，也可以像下面这样显式地指定需要获取的列：

```
SELECT studentid, testid, score FROM TestResults;
```

下面是这一查询的结果：

```
studentid   testid      score
----------  ----------  -----
1           1           82
```

Download at http://www.pin5i.com/

```
1          2          88
1          3          78
1          4          90
1          5          85
1          6          94
2          1          10
2          2          90
2          3          50
2          4          55
2          5          45
2          6          65
3          6          100
3          5          100
3          4          100
3          3          100
3          2          100
3          1          100
```

E.3.6 修改和更新表数据

Ron 成绩比较差，但却在第 1 次小测验中获得了 90 分。这是很可疑的，所以，作为一个老师，我们检查了他的考卷以确定没有出现录入错误。不幸的是，他只得了 60 分。此时，我们就需要对数据库表中的数据进行更新，以修正这一分数：

```
UPDATE TestResults
SET score=60
WHERE studentid=2 AND testid=2;
```

你可以使用 DELETE 表达式来删除表中的数据行。例如，要删除你刚刚更新的数据记录，可以使用如下语句：

```
DELETE FROM TestResults WHERE studentid=2 AND testid=2;
```

如果不使用 WHERE 子句，则可以删除数据库表中的所有记录：

```
DELETE FROM TestResults;
```

E.3.7 使用 JOIN 进行多表查询

现在我们已经拥有了数据库中的全部数据，是时候使用它们了。前面查询的结果并不易阅读，而如果列出学生的姓名（而非 ID）及其测试情况则会一目了然。

数据的联合通常由 JOIN 来处理，它能够结合多个表中的数据源；JOIN 具有几种不同的类型。当你同时使用多个表时，需要指定数据列所属的表（对于这些不同的 id 列而言更是如此）。你可以通过列名称或表名称加点号（“.”）再加列名称的方式来引用列。

现在我们再次列出成绩记录，只不过这次使用测试的名称和学生的姓名。而且，我们只显示期末测试的成绩（test id 6）：

```
SELECT
Students.fname||' '|| Students.lname AS StudentName,
Tests.testname,
TestResults.score
FROM TestResults
JOIN Students
```

Download at http://www.pin5i.com/

```
    ON (TestResults.studentid=Students.id)
JOIN Tests
    ON (TestResults.testid=Test.id)
WHERE testid=6;
```

我们可以获得如下结果（也可以去掉 WHERE 来返回所有结果）：

```
StudentName          testname         score
-------------------  ---------------  -----
Harry Potter         Final            94
Ron Weasley          Final            65
Hermione Granger     Final            100
```

E.3.8 使用计算列

Hermoine 总想知道自己的水平。当她来询问她的总成绩时，我们可以执行一次查询来显示她所有测试的成绩，并且计算出一个总的得分：

```
SELECT
Students.fname||' '|| Students.lname AS StudentName,
Tests.testname,
Tests.weight,
TestResults.score,
(Tests.weight*TestResults.score) AS WeightedScore
FROM TestResults
JOIN Students
    ON (TestResults.studentid=Students.id)
JOIN Tests
    ON (TestResults.testid=Test.id)
WHERE studentid=3;
```

这样，得到了下面这个可以预见到的结果：

```
StudentName          testname  weight  score  WeightedScore
-------------------  --------  ------  -----  -------------
Hermione Granger     Midterm   0.25    100          25.0
Hermione Granger     Quiz 1    0.1     100          10.0
Hermione Granger     Quiz 2    0.1     100          10.0
Hermione Granger     Quiz 3    0.1     100          10.0
Hermione Granger     Quiz 4    0.1     100          10.0
Hermione Granger     Final     0.35    100          35.0
```

我们可以自己累加带权分数，但也可以通过查询来实现：

```
SELECT
Students.fname||' '|| Students.lname AS StudentName,
SUM((Tests.weight*TestResults.score)) AS TotalWeightedScore
FROM TestResults
JOIN Students
    ON (TestResults.studentid=Students.id)
JOIN Tests
    ON (TestResults.testid=Tests.id)
WHERE studentid=3;
```

这样，我们就得到了一个综合的结果：

```
StudentName          TotalWeightedScore
-------------------  ------------------
```

Download at http://www.pin5i.com/

```
Hermione Granger   100.0
```

如果我们想要获得所有学生的成绩，就需要使用 GROUP BY 子句了。除此，我们还将其进行排序，这样最优秀的学生将排在最前面：

```
SELECT
Students.fname||' '|| Students.lname AS StudentName,
SUM((Tests.weight*TestResults.score)) AS TotalWeightedScore
FROM TestResults
JOIN Students
    ON (TestResults.studentid=Students.id)
JOIN Tests
    ON (TestResults.testid=Tests.id)
GROUP BY TestResults.studentid
GROUP BY TotalWeightedScore DESC;
```

对于老师而言，这样的电子成绩簿能够节省很多体力。

```
StudentName                TotalWeightedScore
-------------------------  ------------------
Hermione Granger           100.0
Harry Potter               87.5
Ron Weasley                46.25
```

E.3.9 对计算列使用子查询

你还可以在查询中包含其他查询。例如，你可以列出所有的学生，同时给出他们“及格”的测试数，“及格”意味着分数超过 60，如下所示：

```
SELECT
Students.fname||' '|| Students.lname AS StudentName,
Students.id AS StudentID,
(SELECT COUNT(*)
FROM TestResults
WHERE TestResults.studentid=Students.id
AND TestResults.score>60)
AS TestsPasses
FROM Students;
```

和之前一样，我们可以看出需要为 Ron 单独开开“小灶”：

```
StudentName          StudentID  TestsPassed
-------------------  ---------  -----------
Harry Potter         1          6
Ron Weasley          2          1
Hermione Granger     3          6
```

E.3.10 删除表

你可以使用 DROP TABLE 命令来删除表。例如，要删除 TestResults 表，可以使用下面的 SQL 命令：

```
DROP TABLE TestResults;
```

Download at http://www.pin5i.com/