高等学校应用型本科创新人才培养计划指定教材
高等学校计算机类专业“十三五”课改规划教材

Android 高级开发及实践

青岛农业大学
青岛英谷教育科技股份有限公司 编著

西安电子科技大学出版社
http://www.xduph.com

高等学校应用型本科创新人才培养计划指定教材

高等学校计算机类专业“十三五”课改规划教材

Android 高级开发及实践

青岛农业大学

青岛英谷教育科技股份有限公司

编著

西安电子科技大学出版社

内 容 简 介

本书在“Android 程序设计及实践”课程的基础上，以理论联系实践的形式深入地讲解了 Android 高级开发的相关知识与技术。全书共有 8 章，具体介绍了 Content Provider、图形图像与动画、高级网络编程、高级用户体验、传感器、Wi-Fi 与 Bluetooth、NFC 以及资源与国际化等知识。另外，本书还讲解了移动物联网的相关概念及有关程序的实现、NFC 近场通信技术等。

本书案例基于 Eclipse 开发工具编写，使用的 SDK 版本为 Android4.3(API 18)。

本书适用范围较广，可作为高等院校计算机科学与技术、移动互联网、软件工程、网络工程、计算机软件、计算机信息管理以及电子商务等专业的程序设计课程的教材，也可作为科研、程序设计等人员的参考书籍。

图书在版编目(CIP)数据

Android 高级开发及实践/青岛农业大学，青岛英谷教育科技股份有限公司编著.
—西安：西安电子科技大学出版社，2016.11
高等学校计算机类专业“十三五”课改规划教材
ISBN 978-7-5606-4316-8

Ⅰ. ① A… Ⅱ. ① 青… ② 青… Ⅲ. ① 移动终端—应用程序—程序设计—高等学校—教材
Ⅳ. ① TN929.53

中国版本图书馆 CIP 数据核字(2016)第 269047 号

策　　划　毛红兵
责任编辑　万晶晶　毛红兵
出版发行　西安电子科技大学出版社(西安市太白南路 2 号)
电　　话　(029)88242885　88201467　　邮　　编　710071
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西天意印务有限责任公司
版　　次　2016 年 11 月第 1 版　　2016 年 11 月第 1 次印刷
开　　本　787 毫米×1092 毫米　1/16　印　张　19
字　　数　447 千字
印　　数　1～3000 册
定　　价　46.00 元
ISBN 978-7-5606-4316-8/TN
XDUP 4608001-1
如有印装问题可调换

高等学校计算机类专业
“十三五”课改规划教材编委会

主编　吕健波

编委　王　燕　李言照　孔祥木　王吉华
薛庆文　李长明　李树金　张广渊
陈龙猛　唐述宏　李大明　刘　斌
孔祥和　刘汉平　吴海峰　武　华

❖❖❖ 前　　言 ❖❖❖

本科教育是我国高等教育的基础，而应用型本科教育是高等教育由精英教育向大众化教育转变的必然产物，是社会经济发展的要求，也是今后我国高等教育规模扩张的重点。应用型创新人才培养的重点在于训练学生将所学理论知识应用于解决实际问题，这主要依靠课程的优化设计以及教学内容和方法的更新。

另外，随着我国信息技术的迅猛发展，社会对具备信息技术能力的人才需求急剧增加，“全面贴近企业需求，无缝打造专业实用人才”是目前高校计算机专业教育的革新方向。为了适应高等教育体制改革的新形势，积极探索适应 21 世纪人才培养的教学模式，我们组织编写了高等院校计算机专业系列课改教材。

该系列教材面向高校计算机专业应用型本科人才的培养，强调产学研结合，经过了充分的调研和论证，并参照多所高校一线专家的意见，具有系统性、实用性等特点。旨在使读者在系统掌握计算机知识的同时，着重培养其综合应用能力和解决问题的能力。

该系列教材具有如下几个特色：

1. 以培养应用型人才为目标

本系列教材以应用型软件人才为培养目标，在原有体制教育的基础上对课程进行了改革，强化“应用型”技术的学习。使读者在经过系统、完整的学习后能够掌握如下技能：

- ✧ 掌握信息系统开发所需的理论和技术体系以及系统开发过程规范体系；
- ✧ 能够熟练地进行信息系统设计和编码工作，并具备良好的自学能力；
- ✧ 具备一定的项目经验，包括代码的调试、文档编写、软件测试等内容；
- ✧ 达到信息技术企业的用人标准，做到学校学习与企业的无缝对接。

2. 以新颖的教材架构来引导学习

本系列教材采用的教材架构打破了传统的以知识为标准编写教材的方法，引导读者在学习理论知识的同时，加强实践动手能力的训练。

教材内容的选取遵循“二八原则”，即重点内容由企业中常用的 20%的技术组成。每个章节设有本章目标，明确本章学习重点和难点，章节内容结合示例代码，引导读者循序渐进地理解和掌握这些知识和技能，培养学生的逻辑思维能力，掌握信息系统开发的必备知识和技巧。

另外，本系列教材借鉴了软件开发中的“低耦合，高内聚”的设计理念，组织结构上遵循软件开发中的 MVC 理念，即在保证最小教学集的前提下可以根据自身的实际情况对

此为试读，需要完整PDF请访问：www.ertongbook.com

整个课程体系进行横向或纵向裁剪。

3. 提供全面的教辅产品来辅助教学实施

为充分体现“实境耦合”的教学模式，方便教学实施，该系列教材配备可配套使用的项目实训教材和全套教辅产品。

- ✧ 实训教材：集多线于一面，以辅助教材的形式，提供适应当前课程(及先行课程)的综合项目，遵循系统开发过程，进行讲解、分析、设计、指导，注重工作过程的系统性，培养读者解决实际问题的能力，是实施“实境”教学的关键环节。
- ✧ 立体配套：为适应教学模式和教学方法的改革，本系列教材提供完备的教辅产品，主要包括教学指导、实验指导、电子课件、习题集、实践案例等内容，并配以相应的网络教学资源。教学实施方面，提供全方位的解决方案(课程体系解决方案、实训解决方案、教师培训解决方案和就业指导解决方案等)，以适应信息系统开发教学过程的特殊性。

本书由青岛农业大学和青岛英谷教育科技股份有限公司共同编写，参与本书编写工作的有吕健波、邵作伟、宁维巍、宋国强、侯方超、何莉娟、王千、杨敬熹、刘江林、王万琦等。本书在编写期间得到了各合作院校专家及一线教师的大力支持与协作，在此，衷心感谢每一位老师与同事为本书出版所付出的努力。

由于编者水平有限，书中难免有不足之处，欢迎大家批评指正！读者在阅读过程中发现问题，可以通过邮箱(yinggu@121ugrow.com)发给我们，以期进一步完善。

本书编委会

2016 年 9 月

❖❖❖ 目　　录 ❖❖❖

第 1 章　Content Provider 1
1.1　Content Provider 概述 2
1.1.1　相关 API 2
1.1.2　Content Provider 操作规则 4
1.2　系统通讯录 5
1.2.1　系统通讯录结构 5
1.2.2　操作系统通讯录 7
1.3　自定义 Content Provider 19
1.3.1　创建 Content Provider 19
1.3.2　使用自定义的 Content Provider 25
本章小结 32
本章练习 32
第 2 章　图形图像与动画 33
2.1　图形绘制 34
2.1.1　Color 类 34
2.1.2　Paint 类 35
2.1.3　Path 类 35
2.1.4　Canvas 类 36
2.1.5　绘制几何图形 37
2.2　Property Animation(属性动画) 40
2.2.1　ValueAnimator 40
2.2.2　ObjectAnimator 41
2.2.3　AnimatorSet 41
2.2.4　AnimatorInflater 42
本章小结 46
本章练习 46
第 3 章　高级网络编程 47
3.1　HTTP 概述 48
3.1.1　HttpURLConnection 48
3.1.2　HttpClient 60
3.2　上传文件到服务器 67

3.3 断点续传下载文件 73
3.3.1 断点续传的流程及原理 73
3.3.2 断点续传的实现 74
本章小结 89
本章练习 90
第 4 章 高级用户体验 91
4.1 图片自适应 92
4.1.1 Draw9-patch 概述 92
4.1.2 绘制图片缩放 93
4.1.3 绘制内容填充区域 95
4.2 ListView 列表视图 98
4.2.1 ListView 事件处理 98
4.2.2 Adapter 概述 99
4.2.3 ArrayAdapter 100
4.2.4 SimpleAdapter 103
4.2.5 自定义 Adapter 106
4.2.6 自定义 Adapter 的优化 111
4.3 PopupWindow 113
4.3.1 PopupWindow 概述 114
4.3.2 PopupWindow 的使用 115
4.4 ViewPager 118
4.4.1 ViewPager 概述 118
4.4.2 编写简易图片查看器 119
本章小结 124
本章练习 124
第 5 章 传感器 125
5.1 传感器简介 126
5.1.1 传感器相关类 126
5.1.2 查看本机传感器 128
5.2 传感器的应用 131
5.2.1 光线传感器 132
5.2.2 距离传感器 134
5.2.3 气压传感器 137
5.2.4 温度传感器 140
5.2.5 加速度传感器 140
5.2.6 陀螺仪传感器 143
5.2.7 磁场传感器 147
5.2.8 相对湿度传感器 149
5.2.9 环境温度传感器 150

5.2.10 旋转矢量传感器 150
5.2.11 重力传感器 150
5.2.12 线性加速度传感器 153
5.2.13 方向传感器 153
本章小结 156
本章练习 156
第 6 章 Wi-Fi 与 Bluetooth 157
6.1 Wi-Fi 158
6.1.1 Wi-Fi 概述 158
6.1.2 扫描周围的 Wi-Fi 159
6.1.3 Wi-Fi 相关广播事件 162
6.1.4 连接到指定 Wi-Fi 网络 169
6.1.5 Wi-Fi 技术与设备通信 176
6.2 Bluetooth(蓝牙) 192
6.2.1 传统蓝牙概述 192
6.2.2 传统蓝牙通信 196
6.2.3 BLE 技术概述 219
6.2.4 通过 BLE 技术与设备通信 221
本章小结 232
本章练习 232
第 7 章 NFC 233
7.1 NFC 概述 234
7.1.1 RFID 射频识别技术 234
7.1.2 NFC 工作模式 235
7.2 数据格式 236
7.3 Tag(标签)调度系统 239
7.4 NFC 开发配置 239
7.5 NFC 标签数据操作 242
7.5.1 开发前的准备 242
7.5.2 读写 MifareClassic 标签数据 248
7.5.3 读写 NDEF 纯文本数据 262
本章小结 272
本章练习 272
第 8 章 资源与国际化 273
8.1 Android 资源 274
8.1.1 Android 资源概述 274
8.1.2 资源的创建与使用 278
8.2 国际化 283
8.2.1 跟随系统国际化 283

8.2.2 程序内国际化 287
本章小结 292
本章练习 293
附录 国家地区语言代码表 294

第1章 Content Provider

本章目标

- 了解 Content Provider 运行机制
- 理解 Content Provider 共享数据的规则
- 掌握获取系统通讯录的流程
- 掌握自定义 Content Provider

此为试读，需要完整PDF请访问：www.ertongbook.com

Content Provider(内容提供者)是 Android 系统的四大组件之一，使用 Content Provider 组件可以实现多个程序之间数据的存储和读取。本章主要讲解如何通过 Content Provider 获取系统通讯录，以及如何在自己的程序中自定义 Content Provider 组件，以向外界提供数据。

1.1 Content Provider 概述

虽然 Android 为每个程序开辟了一块独立的内存区域，却没有提供一个公共的内存区域以供程序之间共享数据。Content Provider 组件恰好可以解决这一问题。

通过 Content Provider 组件能够实现如下功能：

✧ 访问系统程序或其他程序提供的数据，例如系统通讯录、短信等；

✧ 开发的程序中，有部分数据可供给其他程序操作。

如图 1-1 所示，程序 A 提供了 Content Provider 接口，这个接口允许其他程序对程序 A 中的数据进行操作，这些数据可以是普通文件、XML 文件、数据库或网络；程序 B 只需要按照一定的规则访问 Content Provider 接口，即可获取相应的数据。在这里，程序 A 只需公开可共享的数据，程序 B 通过接口即可获取自己关心的数据。

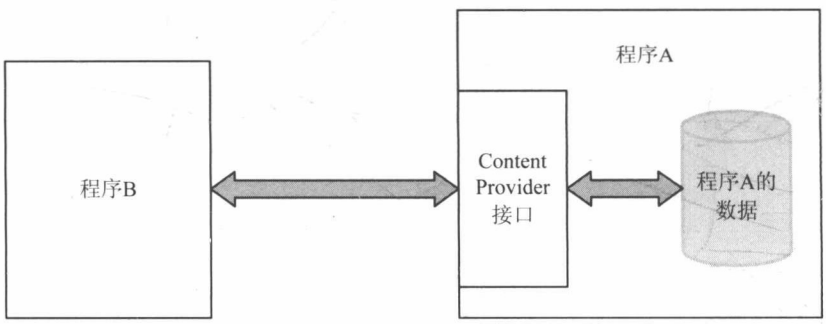

图 1-1 Content Provider 实现流程图

1.1.1 相关 API

Android API 提供了一系列的类来实现或操作 Content Provider 相关功能，主要涉及以下几类：

1. Content Provider

Content Provider 是内容提供程序的基类。该类负责将要共享的数据进行封装，向外界程序提供统一的数据接口，实现数据共享。数据接口使用 Uri 协议规范，通常实现一个内容提供程序，需要定义多个 Uri 接口。需要注意的是：Content Provider 类只能作“内容提供者”，而非“使用者”。

创建一个自定义的内容提供程序，需要继承 Content Provider 类，重写该类的方法如下：

- ✧ onCreate()：只执行一次，通常用于初始化必要的程序设置，例如创建数据库实例等。
- ✧ insert(Uri uri, ContentValues values)：插入数据。把新数据插入内容提供程序中，返回 Uri 对象，表示新插入项的 URI。参数如下：
  - ➢ uri：操作数据的 URI。
  - ➢ values：表示一条数据的参数，以“键值对”格式存放。
- ✧ delete(Uri uri, String selection, String[] selectionArgs)：删除数据。删除内容提供程序中的数据，返回 int 类型值，表示删除操作受影响的行数。参数如下：
  - ➢ selection：对应 SQL 语句中的条件部分，表示删除数据的条件。
  - ➢ selectionArgs：条件语句中对应的占位符填充。
- ✧ update(Uri uri, ContentValues values, String selection, String[] selectionArgs)：更新数据。更新内容提供程序中已有的数据，返回 int 类型值，表示更新操作受影响的行数。
- ✧ query(Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder)：查询数据。返回相应的查询结果，返回 Cursor 游标对象。参数如下：
  - ➢ projection：指定要查询的列，如果为空，则表示查询所有列。
  - ➢ sortOrder：指定查询的排序规则。
- ✧ getType(Uri uri)：查询 URI 类型。返回 String 类型值，是传入 Uri 对象的 MIME 数据类型。此数据类型以“vnd.android.cursor.item/”开头，表示一条记录；以“vnd.android.cursor.dir/”开头，表示多条记录。

2. Content Resolver

Content Resolver 类在 Content Provider 中担任“使用者”的角色，用于根据指定的 URI 操作对应的内容提供程序，实现数据的“增删改查”操作。因此，Content Resolver 的主要方法如下：

- ✧ insert(Uri url, ContentValues values);
- ✧ delete(Uri url, String where, String[] selectionArgs);
- ✧ update(Uri uri, ContentValues values, String where, String[] selectionArgs);
- ✧ query(Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder)。

以上四个“增删改查”方法，与 Content Provider 中对应方法的功能是类似的，区别在于：Content Provider 是对内容提供程序本身的数据进行操作的，而 Content Resolver 用于操作其他程序提供的 Content Provider。

3. ContentValues

ContentValues 类用于以“键值对”的格式存储一系列数值，通常描述数据表中一条记录。该类常用方法如表 1-1 所示。

表 1-1　ContentValues 类常用方法

方　法　名	描　　述
put(String key, String value)	添加一个 String 类型的数值，该方法为重载方法，支持一系列的基本数据类型数值的添加
putAll(ContentValues other)	添加 other 中所有值
putNull(String key)	添加一个空值
Object get(String key)	获取一个值
String getAsString(String key)	获取一个 String 类型值，与 put()方法相对应。与之类似的还有 getAsInteger()、getAsBoolean()、getAsByteArray()等方法
remove(String key)	删除一个值
clear()	清除所有值
size()	获取值的数量
Set<String> keySet()	获取“key”的集合
boolean containsKey(String key)	查看是否存在与“key”对应的值

4. ContentUris

ContentUris 类用于处理 Uri 对象，针对 URI 提供了以下常用方法：

- ✧ Uri withAppendedId(Uri contentUri, long id)：静态方法，为 Uri 对象附加给定的 ID，返回 Uri 对象。
- ✧ long parseId(Uri contentUri)：静态方法，解析给定的 Uri 对象，返回解析后的 long 类型 ID 值。

5. UriMatcher

UriMatcher 类是一个工具类，用于匹配 Content Provider 中用到的 URI。常用方法如下：

- ✧ UriMatcher(int code)：构造方法，“code”为 URI 的根节点，通常传入常量“UriMatcher.NO_MATCH”。
- ✧ addURI(String authority, String path, int code)：添加一个 URI 匹配规则。参数如下：
  - ➢ authority：权限匹配部分。
  - ➢ path：路径，通常表示要操作的数据表的名称。
  - ➢ code：匹配成功的返回码。
- ✧ match(Uri uri)：尝试匹配 Uri 路径，返回 int 类型数值，表示匹配的“code”。

1.1.2　Content Provider 操作规则

访问程序提供的 Content Provider，必须遵循一个规则，就像浏览网页时必须遵循 HTTP 协议一样。

Content Provider 通过 Uri 对象来实现数据共享，相当于浏览网页时输入的域名。标准 URI 如下所示：

```
content://authority/path/id
```

- ✧ 第一部分(content)：协议头，必须以“content://”开头，对应的常量为“ContentResolver.SCHEME_CONTENT”。
- ✧ 第二部分(authority)：权限部分，要保证每个 Content Provider 具有唯一的“authority”，权限部分必须与 AndroidManifest.xml 文件中声明 ContentProvider 时的“authority”保持一致。
- ✧ 第三部分(path)：资源部分，多个层次使用“/”分隔，通常用于表示要操作的数据表的名称。
- ✧ 第四部分(id)：资源部分的一个子集，通常用于表示数据表中的一条记录。如果操作的是整个资源(数据表)，该部分可省略。

以下协议是一个标准 URI，用于操作“user”表中 ID 为“1”的记录：

```
content://com.yg.providers.myprovider/user/1
```

其中：“com.yg.providers.myprovider”对应授权部分；“user/1”对应资源部分。

Content Provider 使用的 Uri 协议中，协议头和权限部分是不可或缺的。

1.2 系统通讯录

Android API 将系统通讯录以 Content Provider 的方式提供给开发者，使其可以对通讯录进行一系列的操作。

1.2.1 系统通讯录结构

Android 系统通过一个单独的 Content Provider 程序来向其他程序提供系统通讯录相关操作。想要了解系统通讯录数据在数据库中存储的结构，需要将此程序的数据库文件导出，之后通过 SQLite 数据库管理工具进行查看。

自 Android4.0 起，DDMS 程序已经无法查看手机中已安装程序的数据，但可以查看模拟器中的程序信息。因此，接下来将使用模拟器进行导出操作。

在“DDMS”界面中，打开“File Explorer”文件浏览器窗口，系统通讯录 Content Provider 程序位于“data/data/com.android.providers.contacts”目录下，数据保存在该目录中的“databases/contacts2.db”数据库文件中，将其导出，并使用 SQLite 数据库管理工具打开。

系统通讯录数据库中存在若干张数据表，但与本小节相关的表只有三张：“mimetypes”表、“raw_contacts”表和“data”表。

1. mimetypes 表

“mimetypes”表用于存放数据类型，这些数据类型用于区分“data”表中的数据。

Android 提供了相应的常量与“mimetypes”表中列出的数据类型相对应，这些常量均由 CommonDataKinds 类中的内部类描述，CommonDataKinds 类中常用内部类如表 1-2 所示。

表 1-2 CommonDataKinds 类中常用内部类

类 名	描 述
StructuredName	描述姓名类型数据
Phone	描述电话号码类型数据
Email	描述电子邮箱类型数据
Photo	描述头像类型数据
StructuredPostal	描述联系地址类型数据

表 1-2 中的类分别描述了各种数据类型和对应表中的字段名等常量。其中，这些类有一个共同的常量“CONTENT_ITEM_TYPE”，对应“mimetypes”表中列出的数据类型，对应关系如表 1-3 所示。

表 1-3 “mimetypes”表中常用数据类型对应常量

数 据 类 型	常 量
vnd.android.cursor.item/name	StructuredName.CONTENT_ITEM_TYPE
vnd.android.cursor.item/phone_v2	Phone.CONTENT_ITEM_TYPE
vnd.android.cursor.item/email_v2	Email.CONTENT_ITEM_TYPE
vnd.android.cursor.item/photo	Photo.CONTENT_ITEM_TYPE
vnd.android.cursor.item/postal-address_v2	StructuredPostal.CONTENT_ITEM_TYPE

2. raw_contacts 表

“raw_contacts”表是通讯录的主表，用于存放联系人的详细数据，但通常用于操作的表是“data”表，在“data”表中添加新联系人的信息之前，必须在“raw_contacts”表中生成新联系人的 ID。

操作“raw_contacts”表的 Uri 是“content://com.android.contacts/raw_contacts”，对应的常量为：RawContacts.CONTENT_URI。

3. data 表

“data”表用于存放具体联系人的数据，是“raw_contacts”表的子表。其主要有三个字段：“raw_contact_id”、“data1”和“mimetype_id”。

- ✧ “raw_contact_id”描述联系人唯一标识，与“raw_contacts”表相关联。
- ✧ “data1”描述联系人资料，例如：姓名、电话号码、电子邮箱等信息。
- ✧ “mimetype_id”与“mimetypes”表相关联，用于描述“data1”中数据的类型。

在系统通讯录中添加几条联系人数据后，导出数据库，“data”表中主要列的内容如图 1-2 所示。

观察图 1-2 不难发现，联系人的不同信息都被保存到了“data1”字段中，这些不同的信息用“mimetype_id”来区分。因此，如果要对联系人信息进行操作，对象主要是“data”表中的“data1”字段。

操作“data”表的 Uri 是“content://com.android.contacts/data”，对应的常量为：

ContactsContract.Data.CONTENT_URI。

mimetype_id	raw_contact_id	data1
(empty)	(empty)	(empty)
5	1	1 562-889-1233
8	1	Shandong Qingdao
1	1	lilei@mail.com
7	1	LiLei
5	2	1 866-192-9369
8	2	Beijing
1	2	tom@mail.com
7	2	Tom
5	3	1 585-678-8959
8	3	Shandong Jinan
1	3	lucy@mail.com
7	3	Lucy

图 1-2　“data”表中主要列的内容

1.2.2　操作系统通讯录

在了解了系统通讯录数据组织结构后，本小节将编写程序，对系统通讯录进行操作。

下述示例用于实现：利用系统提供的通讯录 Content Provider，对通讯录进行“增删改查”操作。要求如下：

✧ 打开程序，查询通讯录中所有联系人，并显示到界面中。

✧ 添加新的联系人到通讯录。

✧ 修改、删除通讯录中的联系人。

对通讯录进行操作的步骤如下：

(1) 创建项目“ch01_ContactManager”，首先创建一个联系人实体类，用于封装联系人信息，包括联系人 ID、名称和电话号码等属性。在“com.yg.ch01_contactmanager.entity”包中创建“Contact.java”类，代码如下：

```
public class Contact {
    private String id;
    private String name;
    private String phone;

    // 属性 get set 方法略
    @Override
    public String toString() {
        return "Contact [id=" + id + ", name=" + name + ", phone="
                + phone + "]";
    }
}
```

(2) 编写工具类，对系统通讯录进行“增删改查”等操作。创建“ContactHelper.java”类，该类是项目中的核心类。首先添加 addContact()方法，代码如下：

此为试读，需要完整PDF请访问：www.ertongbook.com