Android 编程

Programming in Android

钟元生　高成珍　编著

Android编程就这么简单
从Android菜鸟到达人就一步之遥

清华大学出版社

Android 编程

主　编　钟元生　高成珍

参　编　朱文强　徐　军　涂云钊

清华大学出版社

二 00 五年四月

内容简介

本书基于教学实践，反复提炼而成，包括 Android 起步、Android 界面设计基础、Android 事件处理、Android 活动与意图(Activity 与 Intent)、Android 服务(Service)、Android 广播接收器（BroadcastReceiver）、Android 文件与本地数据库(SQLite)、Android 内容提供者（ContentProvider）、Android 图形图像处理、Android 界面设计进阶、Android GPS 位置服务与地图编程、Android 编程综合案例等。

全书内容全面、材料新颖、案例丰富、条理清晰，既可作为大学教材，又可作为自学 Android 编程的快速入门参考书。

作者简介

钟元生，江西财经大学软件与通信工程学院教授、学术委员会主任，电子商务专业博士生导师，教育技术学研究生导师组组长，浙江大学博士毕业，美国加州大学尔湾分校访问学者，江西省计算机学会理事，江西省政府学位委员会学科评议组成员，江西省中青年学科带头人。曾任江西财经大学本科教学质量评建创优专家组副组长、用友软件学院教学副院长，科技部科技支撑计划项目评审专家、江西省教学成果奖评审专家，多次担任 IEEE 电子商务国际学术会议程序委员。

主持或参与国家自然科学基金、全国教育科学规划教育部重点课题、江西省自然科学基金、江西省工业支撑计划项目和江西省科技型中小企业技术创新基金项目等 10 多项，江西省教育厅科技项目等其他省级以上项目多项。作为第一完成人获江西省教学成果一等奖两项，作为第二、第三完成人获省教学成果二、三等奖多项，获全国高校计算机基础教育优秀教材二等奖一项。出版专著 2 部，近年来主编《Android 应用开发教程》、《Android 编程经典案例解析》和《移动电子商务》等教材多部。

江西省大学生手机软件设计赛发起人、总策划和前三届竞赛的专家委员会主任，正在联合全国百所高校举办全国大学生手机软件邀请赛。

创办倚动实验室，基于软件工厂思想，探索移动互联网领域的软件设计、服务创新和人才培养等。在软件工程、计算机科学与技术、电子商务、教育技术、MBA 等专业培养了一大批研究生。

高成珍，本科院校计算机专业教师，江西财经大学软件与通信工程学院教育技术学专业----移动学习与手机软件开发方向硕士毕业，作为骨干开发完成《Android 手机编程》网络课程，曾任江西省大学生手机软件设计赛-----Android 编程指导教师培训班主讲教师、竞赛命题专家和评审教师。参与创建的 Android 编程网络学习社区----倚动实验室，影响越来越大。主编教材《Android 编程经典案例解析》在清华大学出版社出版。

阅读指南

本书假定您懂一些基本的 Java 语法知识，具有一定的 Java 编程经验。如果没有 Java 基础，也可阅读本书，但在涉及 Java 知识时，建议去补充学习一些相关的 Java 知识。

书中示例较多，源代码较长。本书注重示例的程序分析，为了方便介绍知识重点、压缩篇幅，仅列出一些关键代码，读者可从本书配套网站下载完整源码。

强烈建议，读者基于书上说明和关键代码自己补充完成程序，而不主张一开始就下载程序、粗看、调通并对比运行结果。仅在反复尝试失败时，才看下载的源码。

为便于教学，书中源码分别添加了行号，为一些关键语句添加了注释，例如：

```
1    public class MainActivity extends Activity {
2        public void onCreate(Bundle savedInstanceState) {
3            super.onCreate(savedInstanceState);                    →调用父类的该方法
4            setContentView(R.layout.activity_main);                →设置Activity对应的界面布
5        }                                                          局文件
6        public boolean onCreateOptionsMenu(Menu menu) {            →创建选项菜单
7            getMenuInflater().inflate(R.menu.activity_main, menu); →指定菜单资源
8            return true;
9        }
10   }
```

其中，左边的 1、2、3、5、……、10 表示行号，中间的“super.onCreate (savedInstanceState);”才是真实的程序代码内容。“→”及后面的内容“调用父类的该方法”表示对中间代码的注释，非真实编程时所需，请读者注意。

为了方便读者学习，倚动实验室网站上提供了本书相关资源下载，包括源码、课件、教学视频、试题等，网址为：**http://WWW.10LAB.CN/resource.html**。

特别是，我们在以往录制的“手把手教你学 android”教学视频基础上，根据本书的新结构重新录制或编辑了具有微课性质的短小精练的小视频，便于读者学习时使用。

在学习或使用本书过程中有什么疑问或有什么好的建议,欢迎通过 QQ 群：287966120（Android 学习交流群）或 QQ：2227351322、645595894 与我们联系。

教学实践中，我们发现很多同学上机调试经常遇到一些错误就措手无策。本书整理了 Android 上机调试中部分常见错误与程序调试方法（见附录），希望对这些同学有用。

前　　言

近年来，移动互联网影响越来越大，Android 终端越来越普及，各种新的 APP 层出不穷。谁更早地掌握了手机编程技术，谁就占有发展先机。现在，越来越多高校开设 Android 编程课，希望有一本好的教材。

为此，我们在江西省大学生手机软件设计赛指导教师“Android 编程”培训班和多年 Android 教学经验的基础上，完成本书。本书努力做到：

（1）既介绍 Android 基本语法、基本知识和基本应用，又介绍可直接运行的应用教学案例。使教师容易教学，学生能寓学于练、寓学于用。

（2）不仅注重讲解语法细节，而且循序渐进地引导和启发学生建构自己的知识体系，包括用图解法详细分析 Android 应用程序的结构、运行过程以及各部分间的调用关系，演示 Android 应用的开发流程，给出一些关键代码由学生自己去重组和实现相应功能。

（3）重点关注手机应用中常见案例，将有关知识串起来。结合学生用 Android 手机的体验，逐步引导他们深入思考其内部实现。每章都有一些练习题，以帮助学生自测。

本书由钟元生担任主编，负责全书的组织设计、质量控制和统稿定稿。各章分工如下：钟元生负责第 1、第 2 和第 10 章，同时指导和参与了其余各章的编写、修改；高成珍负责第 3、第 4、第 7、第 8、第 11 和第 12 章，徐军负责第 5 章，朱文强负责第 6 章，涂云钊负责第 9 章。研究生刘平、何英、章雯、陈海俊、吴微微、高必梵、杨旭、邵婷婷等参与了初稿讨论、编辑加工以及配套教学课件的制作工作。陈海俊做了大量的初稿电脑排版工作。

许多领导与朋友为本书编写、大学生手机软件设计赛提供了无私支援。特别是，江西财经大学校长、博士生导师王乔教授，在百忙之中过问竞赛并特批经费支持；江西省科技厅副厅长（原江西财经大学副校长）、博士生导师卢福财教授对竞赛给予了大力支持；江西省教育厅高等院校科技开发办公室主任陈东林编审、省教育工委党校校长杜侦研究员参与策划竞赛。江西财经大学软件与通信工程学院院长关爱浩博士、党委书记李新海先生、副院长黄茂军博士、副院长白耀辉博士、副院长邓庆山博士，现代经济管理学院院长、博士生导师陆长平教授，经济管理与创业模拟实验中心主任、博士生导师夏家莉教授，协同创新中心监测预警仿真部主任万本庭博士，以及清华大学出版社副社长卢先和先生、计算机分社袁勤勇主任以不同的形式对我们的工作提供了许多帮助。对上述领导与朋友们的帮助，我们深表感谢。

希望本书帮助 Android 任课教师更快地教好 Android 编程课，也能帮助使用本书的学生更快更扎实地掌握 Android 应用开发技能。

编者

于南昌江西财经大学麦庐园

2015 年 4 月

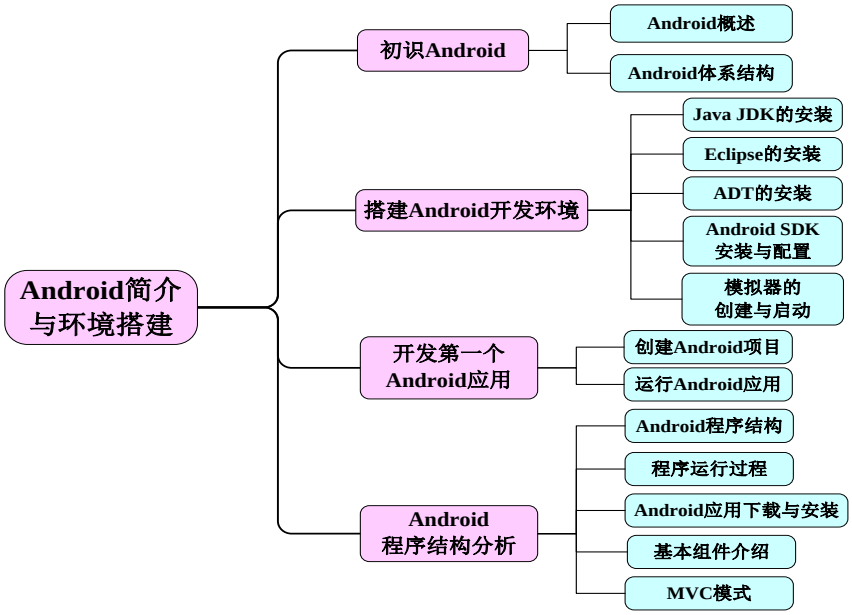
目　录

第 1 章 Android 起步

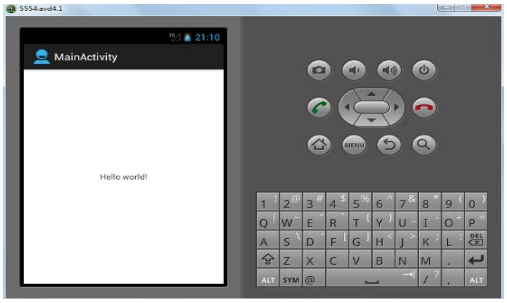
本章要点

- 初识 Android
- 搭建 Android 开发环境
- 开发第一个 Android 应用
- Android 应用结构分析
- Android 应用的下载与安装

本章知识结构图



本章示例



Android 环境搭建成功后，创建第一个 Android 项目，启动模拟器，并运行 Android 程序，效果如左图所示。

本章是Android应用开发的准备章节，主要介绍什么是Android，如何搭建Android开发环境，然后通过一个简单的HelloAndroid程序讲解Android项目的创建、运行过程以及Android应用程序目录结构中各文件的作用等。本章是学好Android的基础，是学习其他章节所必须掌握的内容。

1.1 初识 Android

近年来在开放的手持设备中，Android无疑是发展最快的操作系统之一，覆盖高、中、低端手机系统，在众多系统中，Android为什么能脱颖而出？它究竟有什么特点？下面我们从不同的角度来认识Android。

1.1.1 Android 的概述

1．什么是 Android

Android（英文翻译为机器人，前期版本主要标志是一个绿色机器人，Android 3.0之后的标志改为蜂巢），最早是由安迪•罗宾（Andy Rubin）创办，随后在2005年的时候被Google公司收购。Android是基于Linux平台的开源手机操作系统，Android平台由操作系统、中间件、用户界面和应用软件组成，号称是首个为移动终端打造的真正开放和完整的移动软件。

2008年9月22日，美国运营商T-Mobile USA在纽约正式发布第一款Google手机——T-Mobile G1。该款手机由台湾宏达电代工制造，是世界上第一部使用Android操作系统的手机。目前智能手机的应用已经越来越广泛，市场上已经出现数万种运行于Android平台的手机应用软件，涉及办公软件、影视娱乐软件、游戏软件等应用领域，可以说已深入到移动应用的方方面面。应用软件开发人才的需求数量庞大，据统计，软件应用类Android开发人才的需求约占总需求的72%。

2．Android 的特点：开放性、平等性、无界性、方便性、硬件的丰富性。

1.1.2 Android 的体系结构

Android系统的底层建立在Linux系统之上。该平台由操作系统、中间件、用户界面和应用软件四层组成，采用一种被称为软件叠层（Software Stack）的方式进行构建。这种软件叠层结构使得层与层之间相互分离，以明确各层的分工。这种分工保证了层与层之间的低耦合，当下层的层内或层下发生改变时，上层应用程序无须任何改变。

Android体系结构主要由三部分组成。底层以Linux内核工作为基础，主要由C语言开发，提供基本功能；中间层包括函数库Library和Dalvik虚拟机，由C++语言开发。最上层是各种应用框架和应用软件，包括通话程序，短信程序等。应用软件则由各公司自行开发，主要是以Java语言编写。可以把Android看作是一个类似于Windows的操作系统。Android的体系结构如图1-1所示。

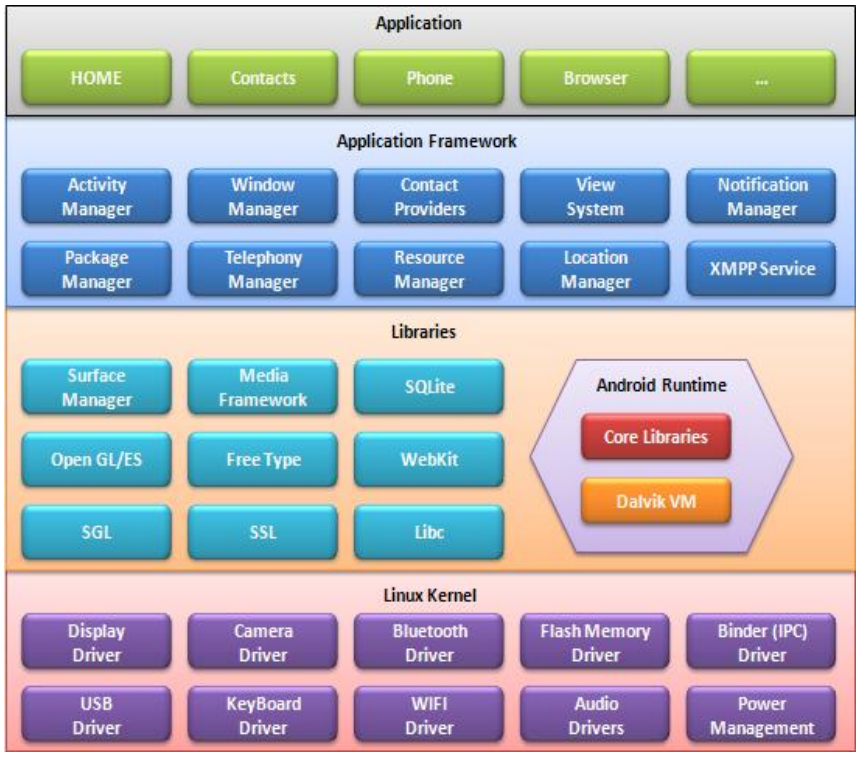


图 1- 1　Android 系统的体系结构

1．应用程序(APPLICATIONS)

Android内有一系列的核心应用，如短信程序、日历工具、地图浏览器、网页浏览器等工具，以及基于Android平台的应用程序框架，所有的应用都是Java语言编写的。

2．应用程序框架(APPLICATION FRAMEWORK)

开发者可以完全使用与那些内核应用程序相同的框架，这些框架用于简化和重用应用程序的组件。若某程序能够“暴露”其内容，则其他程序就可以使用这些内容。例如Android的四大组件：Activity、Service、ContentProvider、BroadcastReceiver。

3．系统运行库层(LIBRARIES)

Android定义了一套C/C++开发库供Android平台的其他组件使用。这些功能通过Android应用程序框架提供给开发者，开发者是不能直接使用这些库的。

4．Linux 内核层(LINUX KERNEL)

Android的核心系统服务依赖于Linux2.6内核，如安全性、内存管理、进程管理、网络协议栈和驱动模型。Linux内核也同时作为硬件和软件栈之间的抽象层。

1.2 搭建 Android 开发环境

本书中所有示例运行环境为：Java JDK1.6+Eclipse4.2+ADT20.0.3+Android SDK 4.4。下面讲述这些工具下载的地址及在Android开发中扮演的功能角色(见表1-1)。

表 1- 1　Android 开发所需软件的下载地址及其功能

软件名称	下载地址	功能角色
Java JDK	http://java.sun.com	Android 是基于 Java 的，需要安装 Java 运行环境。
Eclipse	http://www.eclipse.org	免费、开源的集成开发工具，方便、快捷开发。
Android SDK	http://developer.android.com/sdk/index.html	Android 应用开发工具包，包含 Android 程序运行所需要的各种资源、类库。
ADT	https://dl-ssl.google.com/android/eclipse/	将 Eclipse 和 Android SDK 连接起来的纽带，方便开发 Android 程序。

注意：

（1）本章假定读者已将所有的工具安装包下载并存放在D:\android文件夹下。

（2）上述D:\android文件夹是指下载软件包所存放的文件夹，而不是将来运行的开发环境文件所存放的文件夹。本书不特别指定时，后面假定开发环境均存放在“ F:”盘。

1、上述开发工具中，Java JDK 和 Android SDK 是必需的，而 Eclipse 和 ADT 是可选的。

Eclipse 是一个集成开发工具，能够帮助开发者完成很多繁琐的事情，而 ADT 是 Eclipse 中开发 Android 应用所需要的插件，使用它们可以提高开发者的开发速度和效率。实际上，完全可以通过记事本和命令行来开发和运行 Android 应用程序。

官方下载地址：

- **Java JDK：http://java.sun.com**
- **Eclipse：http://www.eclipse.org**
- **Android SDK：**http://developer.android.com/sdk/index.html
- **ADT：https://dl-ssl.google.com/android/eclipse/**

2、Android4.2 之后官方提供了三合一的安装包，当前最新版本 4.4。

三合一安装包是指包括了 Eclipse 、android SDK 和 Anroid ADT 三部分，只要直接解包即可，大大地简化了安装可能，便利初学者。

三合一安装包下载地址：

http://dl.google.com/android/adt/adt-bundle-windows-x86-20131030.zip (官方网站)

本教程选用三合一安装包来配置 android 开发环境。

这些工具的安装流程与主要步骤如图1-2所示。

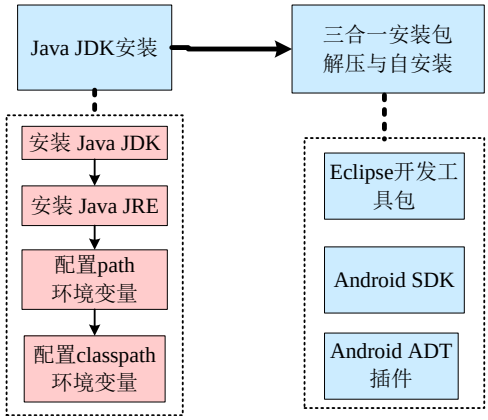


图 1-2 Android 开发环境搭建的流程与主要步骤

1.2.1 安装 JDK 和配置 Java 开发环境

Android程序是基于Java语言的，若要开发和运行Android程序，必须首先安装Java JDK，并对其进行简单配置。

1．JDK1.6 程序的安装

单击下载好的Java JDK安装包，然后弹出提示框，点击下一步，直到选择安装目录如图1-3所示，此处将Java JDK安装在F:\Java\jdk1.6.0_10\目录下，然后继续下一步（**安装目录可任意设置，建议选择的安装目录最好不要包含中文和空格**）。

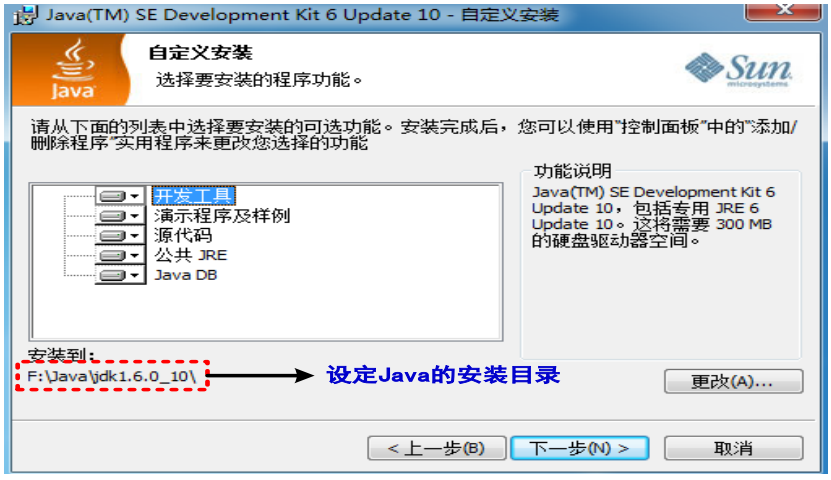


图 1-3 设定 JDK 安装目录图

JDK（Java开发工具）安装过程中，系统会自动安装JRE（Java 运行时环境），更改JRE的安装目录，将其与JDK放在同一目录下，如图1-4所示。

“p6~25所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

课后练习

1．Android 的四大基本组件是_____、_____、_____、_____。

2．搭建 Android 开发环境必需的工具是_____、_____。

3．Android 系统的底层建立在什么操作系统之上（　）。

A）Java　　B）Unix　　C）Windows　D）Linux

4．Android 系统中安装的应用软件是什么格式的（　）。

A）exe　　B）java　　C）apk　　D）jar

5．Android 中启动 Android SDK 和 AVD 管理器的命令是（　）。

A）adb　　B）aidl　　C）android　　D）emulator

6．Android 中启动模拟机(Android Virtual Device)的命令是（　）。

A）adb　　B）android　　C）avd　　D）emulator

7．Android 中完成模拟器文件与电脑文件的相互复制以及安装应用程序的命令是（　）。

A）adb　　B）android　　C）avd　　D）emulator

8．Android 项目工程下面的 assets 目录的作用是什么（　）。

A）放置应用到的图片资源

B）主要放置一些文件资源，这些资源会被原封不动打包到 apk 里面

C）放置字符串，颜色，数组等常量数据

D）放置一些与 UI 相应的布局文件，都是 xml 文件

9．关于 res\raw 目录说法正确的是（　）。

A）该目录下的文件将原封不动的存储到设备上，不会转换为二进制的格式

B）该目录下的文件将原封不动的存储到设备上，会转换为二进制的格式

C）该目录下的文件最终以二进制的格式存储到指定的包中

D）该目录下的文件最终不会以二进制的格式存储到指定的包中

10．当我们创建一个 Android 项目时，该项目的图标是在哪个文件中设置的（　）。

A）AndroidManifest.xml　B）string.xml　C）main.xml　D）project.properties

11．在第一个 Android 项目的 AndroidManifest.xml 文件中，<Application>标签内的 android:label 对应的属性值是什么？该值会显示在模拟器的哪个位置？

12．请简要描述 HelloAndroid 程序的执行过程。

13．res 目录下各文件夹与 R.java 中的类与成员变量之间有什么关系？

14. 在手机及手机模拟器中下载并安装“校园通”APP，简述其基本过程。

15．创建一个 Android 项目，该项目的应用名称为 Name，包名为：com.text.book，为它设置一个自定义的图标，并实现下面的 Android 运行结果。（**注意标题文字和中间显示文字的变化**）

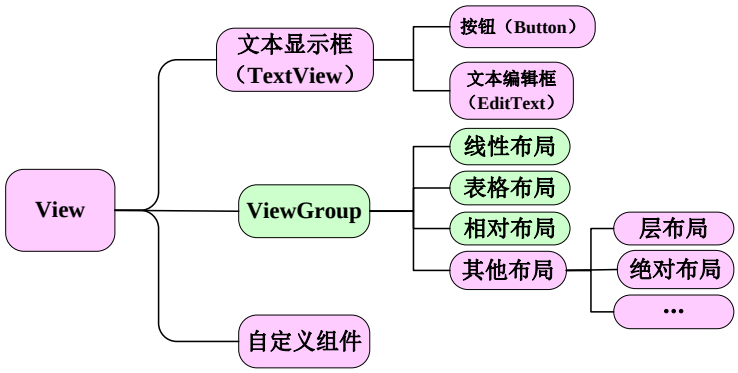
练习15要求实现的效果图

第 2 章 Android 界面设计基础

本章要点

- View 与 ViewGroup 的理解
- 文本显示框的功能和用法
- 文本编辑框的常用属性
- 按钮的简单用法
- 线性布局的功能和用法
- 表格布局的功能和用法
- 相对布局的功能和用法
- 布局的嵌套使用
- 开发自定义 View 的方法和步骤

本章知识结构图



本章示例

第1章中，我们通过一个简单的程序熟悉了Android应用程序的运行过程。Android程序开发主要分为三部分：界面设计、代码流程控制和资源建设。代码和资源主要是由开发者进行编写和维护的，对于大部分用户来说是不关心的，展现在用户面前最直观的就是界面设计。作为一个程序设计者，必须首先考虑用户的体验，只有用户满意了你开发的产品，应用才能推广，才有价值，因此界面设计尤为重要。

Android系统为我们提供了丰富的界面控件，开发者熟悉这些界面控件的功能和用法后，只需要直接调用就可以设计出优秀的图形用户界面。除此之外，Android系统还允许用户开发自定义的界面控件，在系统提供的界面控件基础之上设计出符合自己要求的个性化界面控件。本章将详细讲解Android中的一些最基本的界面控件以及简单的布局管理。通过本章的学习，读者应该能开发出简单的图形用户界面。

本书中有时候将会提到控件、界面控件或界面组件，不特指时均指界面控件。

2.1 基础 View 控件

2.1.1 View 与 ViewGroup 控件

Android中所有的界面控件都继承于View类。View类代表的就是屏幕上的一块空白的矩形区域，该空白区域可用于绘画和事件处理。不同的界面控件，相当于是对这个矩形区域做了一些处理，例如文本显示框、按钮等。

View类有一个重要的子类：ViewGroup。ViewGroup类是所有布局类和容器控件的基类，它是一个不可见的容器，它里面还可以添加View控件或ViewGroup控件，主要用于定义它所包含的控件的排列方式，例如网格排列或线性排列等。通过View和ViewGroup的组合使用，从而使得整个界面呈现一种层次结构。ViewGroup内包含的控件如图2-1所示。

ViewGroup并没有定义其包含的view布局，而其中子类中定义。

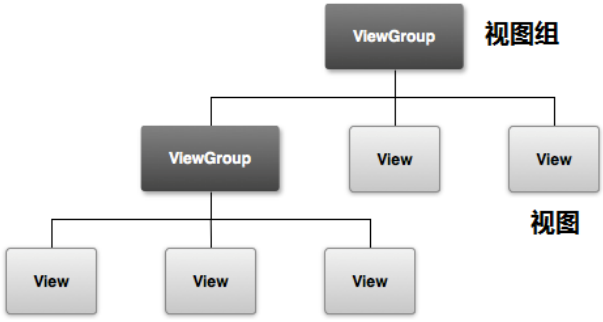


图 2- 1 ViewGroup 控件的层次结构

Android中控制控件的显示有两种方式：一种是通过XML布局文件来设置控件的属性进行控制；另一种是通过Java代码调用相应的方法进行控制。这两种方式控制Android界面显示的效果是完全一样的。实际上，XML文件的属性与Java代码中方法之

间存在着一一对应的关系。从Android API文档中View类的介绍中，可查看所有的属性与方法之间的对应关系，在此只列出一些常用的属性供参考。

表 2- 1 View 类的常见 XML 属性、对应方法及说明

XML 属性	对应的 Java 方法	说明
android:alpha	setAlpha(float)	设置控件的透明度
android:background	setBackgroundResource(int)	设置控件的背景
android:clickable	setClickable(boolean)	设置控件是否可以触发点击事件
android:focusable	setFocusable(boolean)	设置控件是否可以得到焦点
android:id	setId(int)	设置控件的唯一 ID
android:minHeight	setMinimumHeight(int)	设置控件的最小高度
android:minWidth	setMinimumWidth(int)	设置控件的最小宽度
android:padding	setPadding(int,int,int,int)	在控件四边设置边距
android:scaleX	setScaleX(float)	设置控件在 X 轴方向的缩放
android:visibility	setVisibility(int)	设置控件是否可见

几乎每一个界面控件都需要设置android:layout_height、android:layout_width这两个属性，用于指定该控件的高度和宽度，主要有以下三种取值。

- fill_parent：表示控件的高或宽与其父容器的高或宽相同。
- wrap_content：表示控件的高或宽恰好能包裹内容，随着内容的变化而变化。
- match_parent：该属性值与 fill_parent 完全相同，Android2.2 之后推荐使用 match_parent 代替 fill_parent。

虽然两种方式都可以控制界面的显示，但是它们又各有优缺点。（1）完全使用 Java 代码来控制用户界面，不仅繁琐，而且界面设计代码和业务处理代码相混合，不利于软件设计人员的分工合作；（2）完全使用 XML 布局文件虽然方便、便捷，但灵活性不好，不能动态改变属性值。

因此，我们经常会混合使用这两种方式来控制界面，一般来说，习惯将一些变化小的、比较固定的、初始化的属性放在 XML 文件中管理，而对于那些需要动态变化的属性则交给 Java 代码控制。例如，可以在 XML 布局文件中设置文本显示框的高度和宽度以及初始时的显示文字，在代码中根据实际需要动态的改变显示的文字。

2.1.2 文本显示框 TextView

TextView类直接继承于View类，主要用于在界面上显示文本信息，类似于一个文本显示器，从这个方面来理解，有点类似于Java编程中的JLable的用法，但是比JLable的功能更加强大，使用更加方便。TextView可以设置显示文本的字体大小、颜色、风格等属性，TextView的常见属性如表2-2所示。

表 2- 2 TextView 类的常见 XML 属性、对应方法及说明

XML 属性	对应的 Java 方法	说明
android:gravity	setGravity(int)	设置文本的对齐方式
android:height	setHeight(int)	设置文本框的高度（pixel 为单位）
android:text	setText(CharSequence)	设置文本的内容
android:textColor	setTextColor(int)	设置文本的颜色
android:textSize	setTextSize(int,float)	设置文本的大小
android:textStyle	setTextStyle(Typeface)	设置文本的风格
android:typeface	setTypeface(Typeface)	设置文本的字体
android:width	setWidth(int)	设置文本框的宽度（pixel 为单位）

这些是TextView文本显示控件都具有的功能。

除此之外，Android中的TextView还能自动识别文本中的各种链接，能够显示字符串中的Html标签的格式。识别自动链接的属性为：android:autoLink，该属性的值有：

- none：不匹配任何格式，这是默认值。
- web：只匹配网页，如果文本中有网页，网页会以超链接的形式显示。
- email：只匹配电子邮箱，电子邮箱会以超链接的形式显示。
- phone：只匹配电话号码；电话号码会以超链接的形式显示。
- map：只匹配地图地址。
- all：匹配以上所有。

当匹配时，相应部分会以超链接显示，单击超链接，会自动的运行相关程序。例如电话号码超链接，会调用拨号程序，网页超链接会打开网页等。

而显示 Html 标签格式，则需要通过 java 代码来控制。首先为该文本框添加一个 id 属性，然后在 onCreate()方法中，通过 findViewById(R.id.***)，获取该文本框，最后通过 setText()方法来设置显示的内容。例如：

```
1    TextView tv=((TextView)findViewById(R.id.myText);
     //在布局中有一个TextView,其id为myText
2    tv.setText(Html.fromHtml (“欢迎参加<font color=blue>手机软件设计赛</font>”));
```

该代码的显示效果是：

手机软件设计赛这几个字为蓝色，其他字的颜色为布局文件中设置的颜色。

上面的例子中，fromHtml()方法可识别HTML标签，返回值为spanned类型。Html类实现了CharSequence接口，可作为参数传入方法setText（）。

在Android中经常需要设置尺寸，包括组件的宽度和高度、边距、文本大小等，这些尺寸的单位各不相同，在Android提供了多种尺寸单位，常见有：

- **px (像素pixels)** ：屏幕上真实像素表示，不同设备显示效果相同，用于表示清晰度，像素越高越清晰。
- **dip或dp (device independent pixels)** ：设备独立像素，是一个抽象单位，基于

屏幕的物理密度，1dp在不同密度的屏幕上对应的px不同，从而整体效果不变，dp可消除不同类型屏幕对布局的影响。

- **sp (Scale-independent Pixels—best for text size)** ：比例独立像素，主要处理字体的大小，可以根据屏幕自适应。

为了适应不同分辨率、不同的屏幕密度的设备，推荐使用dip单位，文字使用sp单位。

2.1.3 文本编辑框 EditText

TextView的功能仅仅是用于显示信息而不能编辑，好的应用程序往往需要与用户进行交互，让用户进行输入信息。为此，Android中提供了EditText控件，EditText是TextView类的子类，与TextView具有很多相似之处。它们最大的区别在于，EditText允许用户编辑文本内容，使用EditText时，经常使用到的属性有:

- android:hint：设置当文本框内容为空时，文本框内显示的提示信息，一旦输入内容，该提示信息立即消失，当删除所有输入的内容时，提示信息又会出现。
- android:minLines：设置文本框的最小行数。
- android:inputType：设置文本框接收值的类型，例如只能是数字、电话号码等。当其值为“textPassword”时，可表示密码输入，输入的内容将会以点替代。

2.1.4 按钮 Button

Button也是继承于TextView，功能非常单一，就是在界面中生成一个按钮，供用户单击。单击按钮后，会触发一个单击事件，开发人员针对该单击事件可以设计相应的事件处理；从而实现与用户交互的功能。我们可以设置按钮的大小、显示文字以及背景等。当我们想把一张图片作为按钮时，有两种方法：

- 将该图片作为 Button 的背景图片；
- 使用 ImageButton 按钮，将该图片作为 ImageButton 的 android:src 属性值即可。

需注意的是ImageButton按钮不能指定android:text属性，即使指定了，也不会显示任何文字。

2.1.5 应用举例

下面我们以一个简单的例子，介绍这三种简单控件一些属性的用法。程序运行效果如图2-2所示。在此界面中包含两个TextView、两个EditText、两个Button，界面布局文件如下。

图 2- 2 文本框、编辑框和按钮使用举例的程序运行效果图

在正式学习前，读者可以直接从网站将应用解包后直接引入，直接看代码运行结果，也可以先看完后面的代码再按第一章的创建Android项目的步骤来从头建立该程序，根据个人的学习习惯自由选择。但是，不管那种方法，强烈建议一定要自己亲自从头到尾将本项目例子创建起来，这样才学得扎实。

为便于读者阅读，这里给出引入代码后执行，并查看代码的具体过程：

1. 下载第二章源代码包，并将代码解包（见图2-3所示）

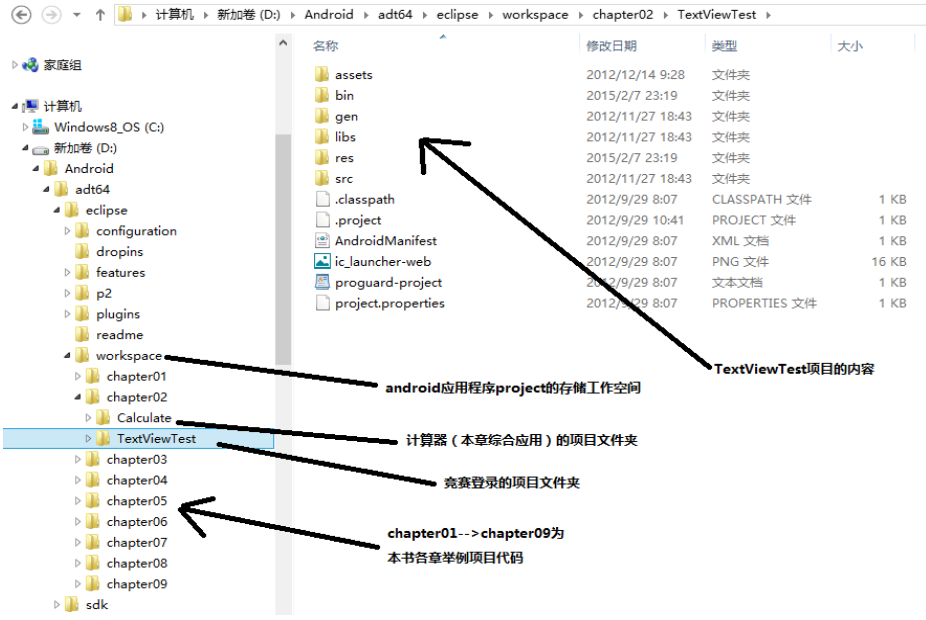


图 2-3 本书全部代码包下载前解包到 workspace 文件夹下

2. 选择 Eclips 的 File/import..菜单，显示引入项目的对话框（见图 2-4），选择 Android/ Existing Android Code Into Workspace,则显示引入项目对话框（Import Projects,见图 2-5）

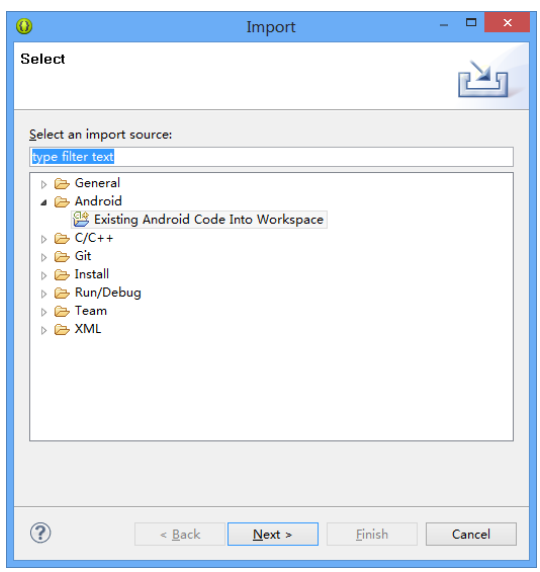


图 2-4 引入资源对话框

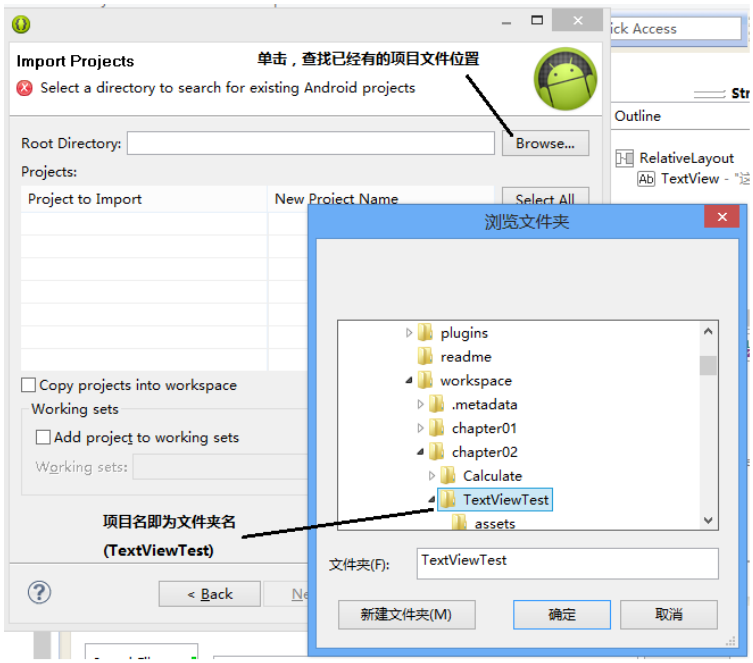


图 2-5　引入项目对话框

3. 选定项目后，单击“确定”，返回图 2-4 所示的引入资源对话框，再单击 Finish 按钮。

在 Package Explorer 下面将出来刚才引入的项目名称（本例中为 TextViewTest），可以打开它看到类似于图 1-28 中所示的程序结构(见图 2-6)。

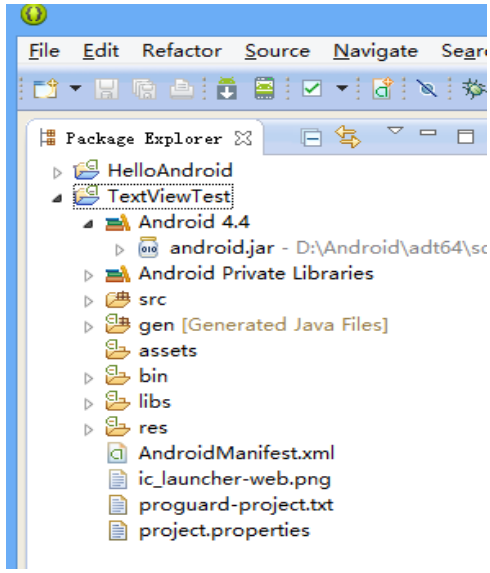


图 2-6 引入了新的项目----TextViewTest

4. 运行引入的项目：右击 TextViewTest 项目，然后在弹出菜单中选择 Run as/ Android Application（见图 2-7），系统将自动对程序进行编译、启动模拟器、装入目标程序，直至最后显示出前述运行结果。

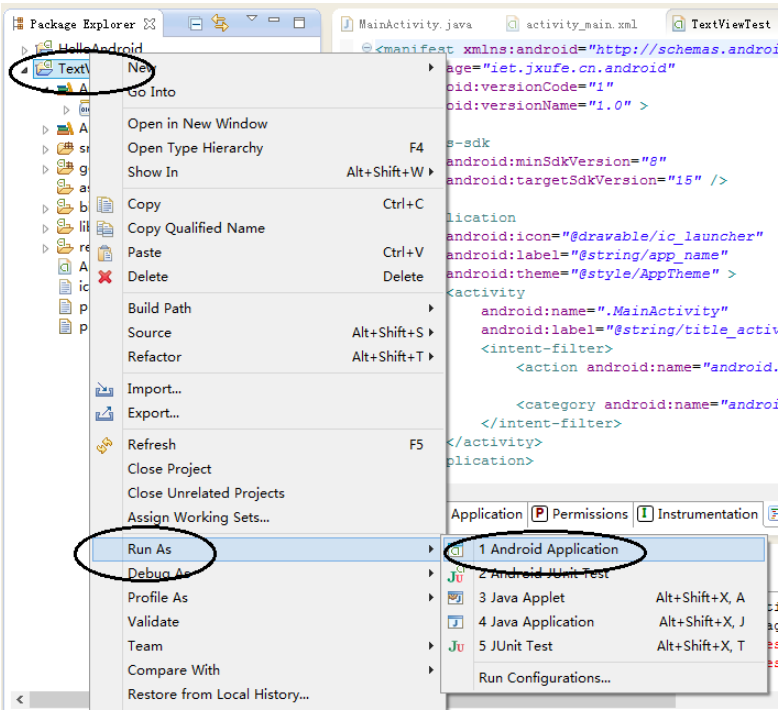


图 2-7　启动项目运行的菜单选择过程

5. 查看所需看的代码，这里主要是看布局文件。依次单击 res/layout/ activity_main.html，选择这个主布局文件，再双击，即可在屏幕中央右边显示该文件的代码。（见图 2-8）

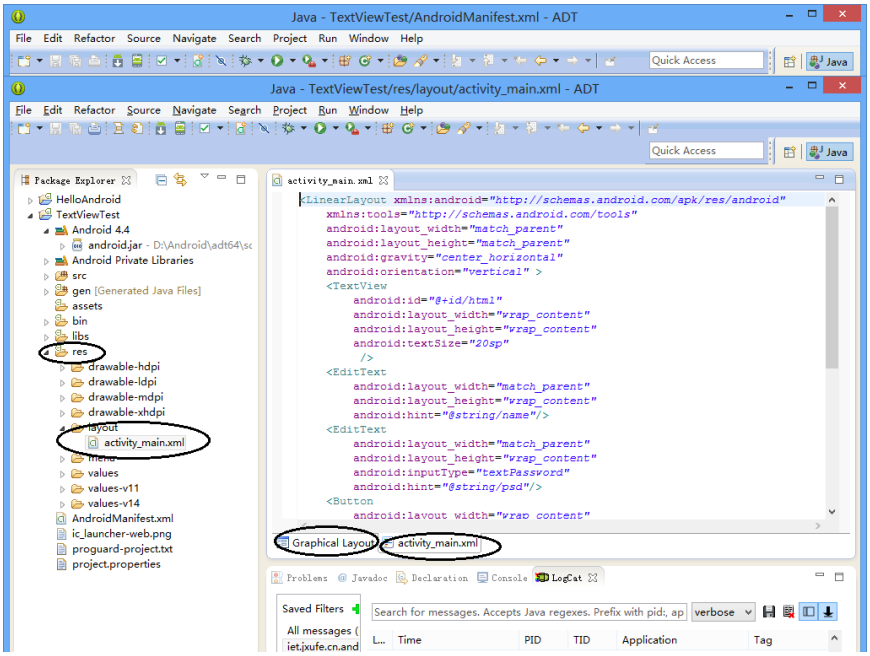


图 2-8 查看布局文件的代码

同样，可以查看该项目的清单文件(AndroidManifest.xml)和主程序(MainActivity. java)等。**清单文件代码 codes\chapter02\TextViewTest\AndroidManifest.xml:**

```
1   <manifest xmlns:android="http://schemas.android.com/apk/res/android"
2       package="iet.jxufe.cn.android"
3       android:versionCode="1"
4       android:versionName="1.0" >
5
6       <uses-sdk
7           android:minSdkVersion="8"
8           android:targetSdkVersion="19" />
9
10      <application
11          android:icon="@drawable/ic_launcher"
12          android:label="@string/app_name"
13          android:theme="@style/AppTheme" >
14          <activity
15              android:name=".MainActivity"
16              android:label="@string/title_activity_main" >
17              <intent-filter>
18                  <action android:name="android.intent.action.MAIN" />
19                  <category android:name="android.intent.category.LAUNCHER" />
20              </intent-filter>
21          </activity>
22      </application>
23  </manifest>
```

布局文件：codes\chapter02\TextViewTest\res\layout\activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"              →线性布局内控件的对齐方式：水平居中
    android:orientation="vertical" >                 →线性布局方向为垂直
     <TextView
        android:id="@+id/html"                       →为TextView添加id属性
        android:layout_width="wrap_content"          →控件宽度为内容包裹
        android:layout_height="wrap_content"         →控件高度为内容包裹
        android:textSize="20sp"    />                →设置文本大小为20像素
     <EditText
         android:layout_width="match_parent"         →控件宽度为填充父容器
         android:layout_height="wrap_content"        →控件高度为内容包裹
         android:hint="@string/name"/>               →设置文本编辑框的提示信息
     <EditText
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:inputType="textPassword"            →设置文本编辑框的输入类型为密码
         android:hint="@string/psd"/>                →设置文本编辑框的提示信息
     <Button
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/login"/>              →设置按钮的显示文本
<Button
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/register"/>           →设置按钮的显示文本
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="18sp"                      →设置文本字体大小为18像素
        android:textColor="#0000ff"                  →设置文本颜色为蓝色
        android:autoLink="all"                       →自动识别所有链接
        android:text="@string/test"/>                →设置显示的文本
</LinearLayout>
```

在布局文件中多次用到@string/***作为android:text的属性值，表示引用R.java中string内部类的***成员变量所代表的资源。这些常量值是在strings.xml文件中定义的。查看strings.xml文件的内容如下。

变量文件：codes\chapter02\TextViewTest\res\values\strings.xml

```
<resources>
    <string name="app_name">TextView</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">竞赛登录</string>
```

```
6   <string name="test">如有疑问请联系我们\n联系电话:0791-83840363\nE-mail:
    iet2011@163.com\n网址：http://iet.jxufe.cn</string>
7       <string name="name">请输入用户名</string>
8       <string name="psd">请输入密码</string>
9       <string name="login">登录</string>
10  <string name="register">注册</string>
11  </resources>
```

其实在设置android:text属性时，可以直接将这些字符串常量赋值给该属性，但是建议不要这么做。因为一些字符串常量可能会在多处被使用，如果都在属性里写，不仅占用更多的内存，而且修改起来也比较麻烦，需要一个个进行修改；另一方面，统一放在strings.xml文件中，还有利于以后软件语言的国际化。针对不同的语言，写一个相应的资源文件就可以了，而不用去更改别的文件，可扩展性比较好。

由于本程序中，还涉及html格式标签的使用，因此需要在Java代码中进行简单设置，首先通过findViewById()方法获取文本控件，然后进行设置显示文本。该过程调用了Html类的静态方法fromHtml()，代码如下。

```
1   public void onCreate(Bundle savedInstanceState) {
2           super.onCreate(savedInstanceState);
3           setContentView(R.layout.activity_main);
4           TextView html=(TextView )findViewById(R.id.html);          →根据id获取文本控件
5   html.setText(Html.fromHtml("欢迎参加<font color=red>" +
6   "手机软件设计赛</font>"));                                    →设置文本控件的显示文本
7   }
```

读者也可以尝试自己来创建这一个Android项目,下面简要给出其主要步骤。

1. 创建android应用程序project,项目名：TextViewTest, 包名：iet.jxufe.cn.android；
2. 打开 res\values\strings.xml,修改有关变量值，主要是应用程序名称的变量；
3. 打开 res\layout\activity_main.xml，修改文件内容，设置有关的布局文件内容；
4. 打开 src\iet.jxufe.cn.android\MainActivity.java文件，在其onCreate（）方法中，增加相应的java代码，使之实现所需要的功能；
5. 保存并检查相应的错误；
6. 若语法无误，则运行之，比较结果，根据结果再修改，直至正确为止。

读者可参照第一章创建Android应用程序的方法，按以上步骤创建本用例程序。

2.2 布局管理器

上一节，我们学习了几种简单的界面控件，并通过一个简单的示例，演示了几种控件的常用属性的基本用法，但是程序的运行界面并不是很美观，控件排列杂乱。本节我们将学习Android中为我们提供的几种管理界面控件的布局管理器。

Android中布局管理器本身也是一个界面控件，所有的布局管理器都是ViewGroup类的子类，都可以当做容器类来使用。因此，可以在一个布局管理器中嵌套其他布局管理器。Android中布局管理器可以根据运行平台来调整控件的大小，具有良好的平台无关性。Android中用得最多的布局主要有：线性布局、表格布局、相对布局。

2.2.1 线性布局

线性布局是最常用也是最基础的布局方式。前面的示例中，我们就使用到了线性布局，它用 LinearLayout 类表示。线性布局和 Java 编程中 AWT 编程里 FlowLayout 有些相似，它们都会将容器里的所有控件一个挨着一个排列。

它提供了水平和垂直两种排列方向，通过 android:orientation 属性进行设置，默认为垂直排列。

- 当为水平方向时，不管控件的宽度是多少，整个布局只占一行，当控件宽度超过容器宽度时，超出的部分将不会显示。
- 当为垂直方向时，整个布局文件只有一列，每个控件占一行，不管该控件宽度有多小。

线性布局与 AWT 编程中 FlowLayout 的最明显的区别：在 FlowLayout 中控件一个个地排列到边界就会自动从下一行重新开始；在线性布局中如果一行的宽度或一列的高度超过了容器的宽度或高度，那么超出的部分将无法显示，如果希望超出的部分能够滚动显示，则需在外边包裹一个滚动控件，ScrollView（垂直滚动）或 HorizontalScrollView（水平滚动）。

在线性布局中，除了设置高度和宽度外，主要设置如下两个属性：

- android:gravity：设置布局管理器内控件的对齐方式，可以同时指定多种对齐方式的组合，多个属性之间用竖线隔开，但竖线前后不能出现空格。例如 bottom|center_horizontal 代表出现在屏幕底部，而且水平居中。
- android:orientation：设置布局管理器内控件的排列方向，可以设置为 vertical（垂直排列）或 horizontal（水平排列）。

2.2.2 表格布局

表格布局是指以行和列的形式来管理界面控件，由TableLayout类表示，不必明确声明包含几行几列，而通过添加TableRow来添加行，在TableRow中添加控件来添加列。

TableRow就是一个表格行，本身也是容器，可以不断地添加其他控件，每添加一个控件就是在该行中增加一列，如果直接向TableLayout中添加控件，而没有添加TableRow，那么该控件将会占用一行。

在表格布局中，每列的宽度都是一样的，列的宽度由该列中最宽的那个单元决定，整个表格布局的宽度则取决于父容器的宽度，默认总是占满父容器本身。

TableLayout继承了LinearLayout，因此它完全支持LinearLayout所支持的全部XML属性，另外，TableLayout还增加了自己所特有的属性。

- android:collapseColumns：隐藏指定的列，其值为列所在的序号，从 0 开始，如

果需要隐藏多列，可用逗号隔开这些序号。

- android:shrinkColumns：收缩指定的列以适合屏幕，使整行能够完全显示不会超出屏幕，用于当某一行的内容超过屏幕的宽度时，会使该列自动换行，其值为列所在的序号。如果没有该属性，则超出屏幕的部分会自动截取，不会显示。
- android:stretchColumns：尽量把指定的列填充空白部分。该属性用于某一行的内容不足以填充整个屏幕，这样指定某一列的内容扩张以填满整个屏幕，其他列的宽度不变。如果某一列有多行，而每行的列数可能不相同，那么可扩展列的宽度是一致的，不会因为某一行有多余的空白而填充整行。也就是说，不管在哪一行，它的宽度都是相同的。
- android:layout_column：控件在 TableRow 中所处的列。如果没有设置该属性，默认情况下，控件在一行中是一列挨着一列排列的。通过设置该属性，可以指定控件所在的列，这样就可以达到中间某一个列为空的效果。
- android:layout_span：该控件所跨越的列数，即将多列合并为一列。

2.2.3 相对布局

表 2-3 相对布局中常用属性设置

属性	说明
android:layout_centerHorizontal	设置该控件是否位于父容器的水平居中位置
android:layout_centerVertical	设置该控件是否位于父容器的垂直居中位置
android:layout_centerInParent	设置该控件是否位于父容器的正中央位置
android:layout_alignParentTop	设置该控件是否与父容器顶端对齐
android:layout_alignParentBottom	设置该控件是否与父容器底端对齐
android:layout_ alignParentLeft	设置该控件是否与父容器左边对齐
android:layout_ alignParentRight	设置该控件是否与父容器右边对齐
android:layout_toRightOf	指定该控件位于给定的 ID 控件的右侧
android:layout_toLeftOf	指定该控件位于给定的 ID 控件的左侧
android:layout_above	指定该控件位于给定的 ID 控件的上方
android:layout_below	指定该控件位于给定的 ID 控件的下方
android:layout_alignTop	指定该控件与给定的 ID 控件的上边界对齐
android:layout_ alignBottom	指定该控件与给定的 ID 控件的下边界对齐
android:layout_ alignLeft	指定该控件与给定的 ID 控件的左边界对齐
android:layout_ alignRight	指定该控件与给定的 ID 控件的右边界对齐

相对布局，顾名思义就是相对于某个控件的位置，由RelativeLayout类表示，这种布局的关键是找到一个合适的参照物，如果甲控件的位置需要根据乙控件的位置来确定，那么要求先定义乙控件，再定义甲控件。

在相对布局中，每个控件的位置可通过它相对于某个控件的方位以及对齐方式来确定，因此相对布局中常见的属性如表2-3所示。由于父容器是确定的，所以与父容器方位与对齐的关系取值为true或false。

2.2.4 其他布局

除以上几种常用的布局方式外，Android还提供了层布局、绝对布局。在此简介之。

层布局也叫帧布局，由FrameLayout类表示。其每个控件占据一层，后面添加的层会覆盖前面的层，后面的控件会叠放在先前的控件之上。如果后面控件的大小大于前面的控件，那么前面的控件将会完全被覆盖，不可见；如果后面控件无法完全覆盖前面的控件，则未覆盖部分，显示先前的控件。这样看来，层布局的显示效果有些类似于Java中AWT编程里的CardLayout，都是把控件一个接一个的叠在一起，但CardLayout通过使用first、last、previous等能够看到所有的控件，但是层布局没有这种功能。

绝对布局，即指定每个控件在手机上的具体坐标，每个控件的位置和大小都是固定的。由于不同手机屏幕可能不同，绝对布局只适合于固定的手机或屏幕，不具有通用性，现在已很少使用。

2.2.5 布局的综合运用

通过上面对几种布局方式的介绍，我们发现每种布局方式都有自己的优缺点，在实际的开发中，往往很难通过一种布局方式就能完成我们的界面设计，需要多种布局方式的嵌套使用，方能达到我们要求的效果。

下面我们以一个简单的示例，演示多种布局管理器的综合运用。该程序设计出一个通用计算器的界面，程序运行效果如图2-9所示。

图 2-9 计算器界面设计图

该界面中包含一个用于显示输入的数字和计算结果的文本编辑框和28个按钮。其中两个按钮比较特别，一个高度是普通按钮的两倍，一个宽度是普通按钮的两倍。单独采用某一种布局方式，例如线性布局，也可以达到该效果，在线性布局中不断的嵌套线性布局。在此，我们根据各控件的特点，综合运用多种布局。该界面整体采用垂直线性布局，先添加一个文本编辑框，然后添加一个四行五列的表格布局，最后再添加相对布局，摆放剩余的按钮。

由于所有的按钮都需要设置高度、宽度、对齐方式、字体大小等属性，在此我们定义三种按钮样式，分别对应于普通按钮，较高的按钮以及较宽的按钮，样式代码如下。

程序清单：codes\chapter02\Calculater\res\values\styles.xml

```
1   <resources xmlns:android="http://schemas.android.com/apk/res/android">
2       <style name="btn01">                                    →普通按钮的风格
3           <item name="android:layout_width">60dp</item>       →设置按钮宽度
4           <item name="android:layout_height">50dp</item>      →设置按钮高度
5           <item name="android:textSize">20sp</item>           →设置按钮上字体大小
6           <item name="android:gravity">center_horizontal</item>  →设置按钮文本对齐方式
7       </style>
8       <style name="btn02">                                    →较宽按钮的风格
9           <item name="android:layout_width">120dp</item>      →按钮宽度为120dp
10          <item name="android:layout_height">50dp</item>      →按钮高度为50dp
11          <item name="android:textSize">20sp</item>           →按钮上字体大小为20sp
12              <item name="android:gravity">center_horizontal</item>  →按钮上文字水平居中
13      </style>
14      <style name="btn03">                                    →较高按钮的风格
15          <item name="android:layout_width">60dp</item>       →按钮宽度为60dp
16          <item name="android:layout_height">100dp</item>     →按钮高度为100dp
17          <item name="android:textSize">20sp</item>           →按钮上字体大小为20sp
18          <item name="android:gravity">center_horizontal</item>  →按钮上文字水平居中
19      </style>
20  </resources>
```

每一种样式都以<style>标签开始，样式中每一个属性值都用<item>标签表示，<item>标签的name属性指定具体的属性，<item>标签的内容为属性的值。引用时，只需将控件的style属性值设置为@style/样式名即可。

首先整体采用线性布局，代码如下。

```
1   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2       xmlns:tools="http://schemas.android.com/tools"
3       android:layout_width="match_parent"
4       android:layout_height="match_parent"
5       android:orientation="vertical">                         →垂直线性布局
6       <EditText                                               →文本编辑框
7           android:layout_width="match_parent"
8           android:layout_height="wrap_content"
9           android:minLines="2" />                             →高度最少两行
10      <TableLayout.../ >                                      →表格布局
11  <RelativeLayout.../>                                        →相对布局
12  </LinearLayout>
```

表格布局中包含四行五列，具体代码如下。

```
1   <TableLayout                                                →表格布局
2           android:layout_width="match_parent"                 →宽度填充父容器
3           android:layout_height="wrap_content" >              →高度包裹内容
4           <TableRow                                           →表格行（第一行）
```

```
5                android:layout_width="match_parent"                →宽度填充父容器
6                android:gravity="center_horizontal" >              →内容水平居中对齐
7                <Button                                            →插入一列（第一列）
8                    style="@style/btn01"                           →引用样式btn01
9                    android:text="MC" />                           →按钮文字为MC
10               <Button                                            →插入一列（第二列）
11                   style="@style/btn01"
12                   android:text="MR" />
13               <Button                                            →插入一列（第三列）
14                   style="@style/btn01"
15                   android:text="MS" />
16               <Button                                            →插入一列（第四列）
17                   style="@style/btn01"
18                   android:text="M+" />
19               <Button                                            →插入一列（第五列）
20                   style="@style/btn01"
21                   android:text="M-" />
22           </TableRow>
23           <TableRow                                              →表格行（第二行）
24               android:layout_width="match_parent"
25               android:gravity="center_horizontal" >
26               <Button
27                   style="@style/btn01"
28                   android:text="←" />
29               <Button
30                   style="@style/btn01"
31                   android:text="CE" />
32               <Button
33                   style="@style/btn01"
34                   android:text="C" />
35               <Button
36                   style="@style/btn01"
37                   android:text="±" />
38               <Button
39                   style="@style/btn01"
40                   android:text="√" />
41           </TableRow>
42           <TableRow                                              →表格行（第三行）
43               android:layout_width="match_parent"
44               android:gravity="center_horizontal" >
45               <Button
46                   style="@style/btn01"
47                   android:text="7" />
48               <Button
49                   style="@style/btn01"
50                   android:text="8" />
51               <Button
52                   style="@style/btn01"
53                   android:text="9" />
```

```
54                  <Button
55                      style="@style/btn01"
56                      android:text="/" />
57                  <Button
58                      style="@style/btn01"
59                      android:text="%" />
60              </TableRow>
61              <TableRow                                          →表格行（第四行）
62                  android:layout_width="match_parent"
63                  android:gravity="center_horizontal" >
64                  <Button
65                      style="@style/btn01"
66                      android:id="@+id/four"                     →添加ID属性
67                      android:text="4" />
68                  <Button
69                      style="@style/btn01"
70                      android:text="5" />
71                  <Button
72                      style="@style/btn01"
73                      android:text="6" />
74                  <Button
75                      style="@style/btn01"
76                      android:text="*" />
77                  <Button
78                      style="@style/btn01"
79                      android:text="1/x" />
80  </TableRow>
81  </TableLayout>
```

相对布局需要一个参照物。本例中，按钮“2”以“1”按钮(id为“one”，见下面代码第6行)为参考，与“1”顶端对齐（见代码14行），在“1”的右边（见代码15行），详见下面代码。

```
1   <RelativeLayout                                    →相对布局
2               android:layout_width="match_parent"
3               android:layout_height="wrap_content"
4               android:gravity="center_horizontal" >  →水平居中对齐
5               <Button
6                   android:id="@+id/one"              →添加ID属性，供其他组件参考
7                   style="@style/btn01"
8                   android:text="1"                   →按钮文字“1”
9                   android:textColor="#0000ff"        →按钮文字为蓝色，“#RRGGBB”
10                  android:textStyle="bold" />        →按钮文字为粗体字
11              <Button
12                  android:id="@+id/two"
13                  style="@style/btn01"
14                  android:layout_alignTop="@id/one"  →“2”与“1”顶端对齐
15                  android:layout_toRightOf="@id/one" →“2”在“1”的右边
16                  android:text="2" />
```

```
17          <Button
18              android:id="@+id/three"
19              style="@style/btn01"
20              android:layout_alignTop="@id/two"          →“3”与“2”顶端对齐
21              android:layout_toRightOf="@id/two"         →“3”在“2”的右边
22              android:text="3" />
23          <Button
24              android:id="@+id/minus"
25              style="@style/btn01"
26              android:layout_alignTop="@id/three"        →“一”与“3”顶端对齐
27              android:layout_toRightOf="@id/three"       →“一”在“3”的右边
28              android:text="-" />
29          <Button
30              android:id="@+id/equal"
31              android:layout_alignTop="@id/minus"        →“=”与“一”顶端对齐
32              android:layout_toRightOf="@id/minus"       →“=”在“一”的右边
33              android:gravity="center"
34              style="@style/btn03"                       →高度为2倍，在styles.xml定义
35              android:text="=" />
36          <Button
37              android:id="@+id/plus"
38              style="@style/btn01"
39              android:layout_alignBottom="@id/equal"
40              android:layout_toLeftOf="@id/equal"
41              android:text="+" />
42          <Button
43              android:id="@+id/dot"
44              style="@style/btn01"
45              android:layout_alignTop="@id/plus"
46              android:layout_toLeftOf="@id/plus"
47              android:text="." />
48          <Button
49              style="@style/btn02"
50              android:layout_alignTop="@id/dot"
51              android:layout_toLeftOf="@id/dot"
52              android:text="0" />
53      </RelativeLayout>
```

本界面设计中最关键的就是两个特殊按钮的摆放，对于表格布局而言，每列的宽度是一致的，并且每一行中，各列的高度也是相同的。而这两个按钮，一个过高，一个过宽，因此采用表格布局不好处理这两个按钮，而对于线性布局而言，要么处于同一行，要么处于同一列，对于占多行或多列的控件需组合使用水平线性布局和垂直线性布局，比较麻烦，在此采用相对布局来处理。

2.3 开发自定义 View

Android中所有的界面控件都是继承于View类，View本身仅仅是一块空白的矩形区域，不同的界面控件在这个矩形区域上绘制外观即可形成风格迥异的控件，基于这个原

理，开发者完全可以通过继承View类来创建具有自己风格的控件。

开发自定义 View 时的一般步骤如下：

1）定义自己控件的类名，并让该类继承 View 类或一个现有的 View 的子类。

2）重写父类的一些方法，通常需要提供一个构造器，构造器是创建自定义控件的最基本方式，当Java代码创建该控件或根据XML布局文件加载并构建界面时都将调用该构造器，根据业务需要重写父类的部分方法。例如onDraw()方法，用于实现界面显示，其他方法还有onSizeChanged()、onKeyDown()、onKeyUp()等。

3）使用自定义的控件，既可以通过Java代码来创建，也可以通过XML布局文件进行创建，在XML布局文件中，该控件的标签是完整的包名+类名，而不再仅仅是原来的类名。例如我们自定义一个圆形控件。程序运行效果如图2-10所示。

图 2- 10 自定义控件

```
1    public class MyView extends View {                    →定义自定义控件类
2        public MyView(Context context, AttributeSet attrs) {   →构造方法，调用父类构造方法
3            super(context, attrs);
4        }
5        protected void onDraw(Canvas canvas) {            →重写父类的onDraw()方法
6            Paint paint = new Paint();                    →创建一个画笔
7            paint.setColor(Color.BLUE);                   →设置画笔颜色——蓝色
8            canvas.drawCircle(50, 50, 50, paint);         →画一个圆，半径为50
9         }
10   }
```

通过XML布局文件来使用该控件，代码如下。

```
1    <iet.jxufe.cn.android.MyView                          →完整的包名+类名
2        android:layout_width="wrap_content"
3        android:layout_height="wrap_content"/>
```

2.4 本章小结

本章主要讲解了Android中界面控件的基本知识，Android中所有的界面控件都继承于View类，View类代表的是一块空白的矩形区域，不同的控件在此区域中进行绘制从而形成了风格迥异的控件。View类有一个重要的子类：ViewGroup，该类是所有布局类或容器类的基类，在ViewGroup中可以包含View控件或ViewGroup，ViewGroup的这种嵌套功能从而形成了界面上控件的层次结构。除此之外，我们详细介绍了几种最基本的界面控件的功能和常用属性，包括文本显示框、文本编辑框和按钮等，并通过“竞赛登录”（图2-2）示例演示了具体的用法。

为了使这些控件排列美观，我们继续学习了Android中几种常见的布局管理器，包括线性布局、表格布局和相对布局，它们各有优缺点，线性布局方便，需使用的属性较少，但不够灵活；表格布局中通过TableRow添加行，每列的宽度一致；相对布局则通过提供一个参照物来准确定义各个控件的具体位置，通常我们在一个实例中会用到多种布局，把各种布局结合起来达到我们所要的界面效果。本章最后通过一个综合的示例演示了如何综合运用多种布局设计一些比较复杂的界面。

课后练习

1．下列哪个属性可做 EditText 编辑框的提示信息（　　）。

A）android:inputType　　B）android:text　C）android:digits　D）android:hint

2．为下面控件添加 android:text=”Hello”属性，运行时无法显示文字的控件是（　　）。

A）Button　　B）EditText　　C）ImageButton　　D）TextView

3．下列选项中，前后两个类不存在继承关系的是（　　）。

A）TextView、EditText　　B）TextView、Button

C）Button、ImageButton　　D) ImageView、ImageButton

4．假设手机屏幕宽度为 400px，现采取水平线性布局放置 5 个按钮，设定每个按钮的宽度为 100px，那么该程序运行时，界面显示效果为（　　）。

A）自动添加水平滚动条，拖动滚动条可查看5个按钮

B）只可以看到4个按钮，超出屏幕宽度部分无法显示

C）按钮宽度自动缩小，可看到5个按钮

D）程序运行出错，无法显示

5．表格布局中，设置某一列是可扩展的正确的做法是（　　）。

A）设置TableLayout的属性：android:stretchColumns="x"，x表示列的序号；

B）设置TableLayout的属性：android: shrinkColumns="x"，x表示列的序号；

C）设置具体列的属性：android:stretchable=”true”；

D）设置具体列的属性：android: shrinkable=”true”；

6．相对布局中，设置以下属性时，属性值只能为 true 或 false 的是（　　）。

A）android:layout_below　　B）android:layout_alignParentLeft

C）android:layout_alignBottom　　D）android:layout_toRightOf

7．布局文件中有一个按钮（Button），如果要让该按钮在其父容器中居中显示，正确的

做法的设置是（　　）。

A）设置按钮的属性：android:layout_gravity="center"

B）设置按钮的属性：android:gravity="center"

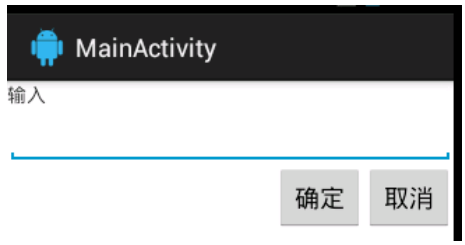
C）设置按钮父容器的属性：android:layout_gravity="center"

D）设置按钮父容器的属性：android:gravity="center"

8．Android 中的水平线性布局不会自动换行，当一行中控件的宽度超过了父容器的宽度时，超出的部分将不会显示，如果想以滚动条的形式显示超出的部分应该怎么做呢？

9．运用表格布局设置 3 行 3 列的按钮，要求：第一行中有一列空着，第三列被拉伸。第三行中有一个按钮占两列，运行效果如图所示。

10．根据所学的相对布局的知识，设计出下图所示界面，要求在文本编辑框内只能输入数字，并且输入的内容会以“点”显示。



11. 在 View 类的 XML 属性中 android:layout_gravity 和 android:gravity 都用于设置对齐方式，它们之间有什么区别？

12．学习了开发自定义 View 的知识，试着编写一个自己的控件。

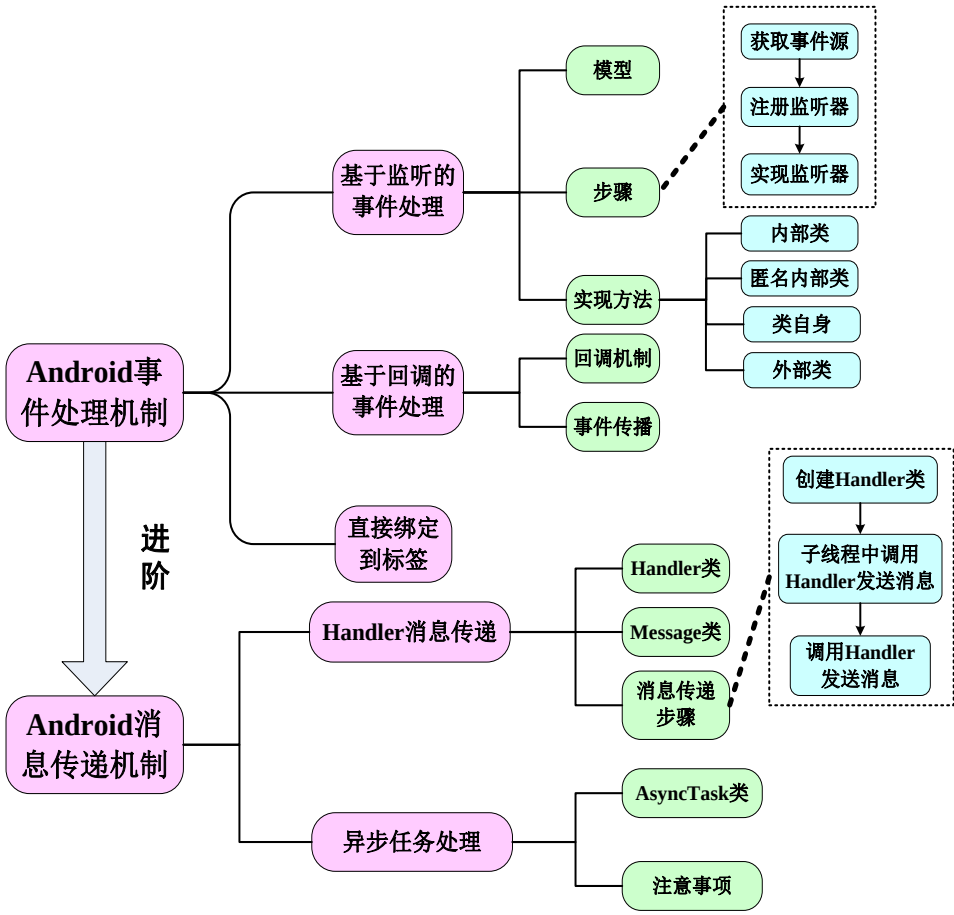
13．运用所学知识，设计下图所示界面。要求：“用户登录”这几个字大小为 28sp，红色。“登录”、“注册”、“找回密码”这几个按钮水平排列并且居中显示。

第 3 章 Android 事件处理

本章要点

- 基于监听的事件处理模型
- 实现事件监听器的四种方式
- 基于回调的事件处理模型
- 事件传播
- 事件直接绑定到标签
- Hanlder 消息传递机制
- 使用 Handler 动态生成随机数
- AsyncTask 异步任务处理

本章知识结构图



本章举例

简单文本编辑器

文件下载提示进度条

前面我们学习了Android所提供的一些基本控件，将来在第10章还会介绍其他功能强大的界面控件，android提供的其他控件读者可以查找有关参考资料。但是，这些控件主要是用来进行数据的显示，如果用户想与之进行交互，实现具体的功能，则还需要相应事件处理的辅助。当用户在程序界面上执行各种操作时，如单击一个按钮，应用程序必须为用户动作提供响应动作，这种响应动作就需要通过事件处理来完成。

Android提供了三种事件处理方式：基于回调的事件处理、基于监听的事件处理和事件直接绑定到标签。熟悉传统图形界面编程的读者对于基于回调事件处理可能比较熟悉；熟悉Java AWT/Swing开发方式的读者对于基于监听的事件处理可能比较熟悉；对于熟悉JavaScript编程的读者对于直接绑定到标签的事件处理可能比较熟悉。Android系统充分利用了三种事件处理的优点，允许开发者采用自己熟悉的事件处理方式来为用户操作提供响应。

在Android中，用户界面属于主线程，而子线程无法更新主线程的界面状态，那么，如何才能动态地显示用户界面呢？本章我们将学习，通过Handler消息传递来动态更新界面。

如果在事件处理中需要做一些比较耗时的操作时，直接放在主线程中将会阻塞程序的运行，给用户以不好的体验，甚至程序会没有响应或强制退出。本章我们将学习，通过AsyncTask异步方式来处理耗时的操作。

学完本章之后，再结合前面所学知识，读者将可以开发出界面友好、人机交互良好

的Android应用。

3.1 Android 的事件处理机制

不管是什么手机应用，都离不开与用户的交互，只有通过用户的操作，才能知道用户的需求，从而实现具体的业务功能，因此，应用中经常需要处理的就是用户的操作，也就是需要为用户的操作提供响应，这种为用户操作提供响应的机制就是事件处理。

Android提供了强大的事件处理机制，包括三种事件处理机制：

(1) 基于监听的事件处理：主要做法就是为Android界面控件绑定特定的事件监听器，在事件监听器的方法里编写事件处理代码，前面我们已经见过大量这种事件处理的示例。

(2) 基于回调的事件处理：主要做法就是重写Android控件特定的回调方法，或者重写Activity的回调方法。Android为绝大部分界面控件都提供了事件响应的回调方法，我们只需重写它们即可，由系统根据具体情景自动调用。

(3) 直接绑定到标签：主要做法就是在界面布局文件中为指定标签设置事件属性，属性值是一个方法的方法名，然后再在Activity中定义该方法，编写具体的事件处理代码。

一般来说，直接绑定到标签只适合于少数指定的事件，实际应用中比较少见；基于回调的事件处理代码比较简洁，可用于处理一些具有通用性的系统为我们定义好的事件。但对于某些特定的事件，无法使用基于回调的事件处理，只能采用基于监听的事件处理。实际应用中，基于监听的事件处理方法应用最广泛。

3.1.1 基于监听的事件处理

Android的基于监听的事件处理模型与Java的AWT、Swing的处理方式几乎完全一样，只是相应的事件监听器和事件处理方法名有所不同。在基于监听的事件处理模型中，主要涉及三类对象：

EventSource（事件源）：产生事件的控件即事件发生的源头，如按钮、菜单等；

Event（事件）：具体某一操作的详细描述，事件封装了该操作的相关信息，如果程序需要获得事件源上所发生事件的相关信息，一般通过Event对象来取得，例如按键事件按下的是哪个键、触摸事件发生的位置等；

EventListener(事件监听器)：负责监听用户在事件源上的操作，并对用户的各种操作做出相应的响应，事件监听器中可包含多个事件处理器，一个事件处理器实际上就是一个事件处理方法。

那么在基于监听的事件处理中，这三类对象又是如何协作的呢？实际上，基于监听的事件处理是一种委托式事件处理。普通控件（事件源）将整个事件处理委托给特定的对象（事件监听器）；当该事件源发生指定的事情时，系统自动生成事件对象，并通知所委托的事件监听器，由事件监听器相应的事件处理器来处理这个事件。具体的事件处理模型如图3-1所示。当用户在Android控件上进行操作时，系统会自动生成事件对象，并将这个事件对象以参数的形式传给注册到事件源上的事件监听器，事件监听器调用相应的事件处理器来处理。

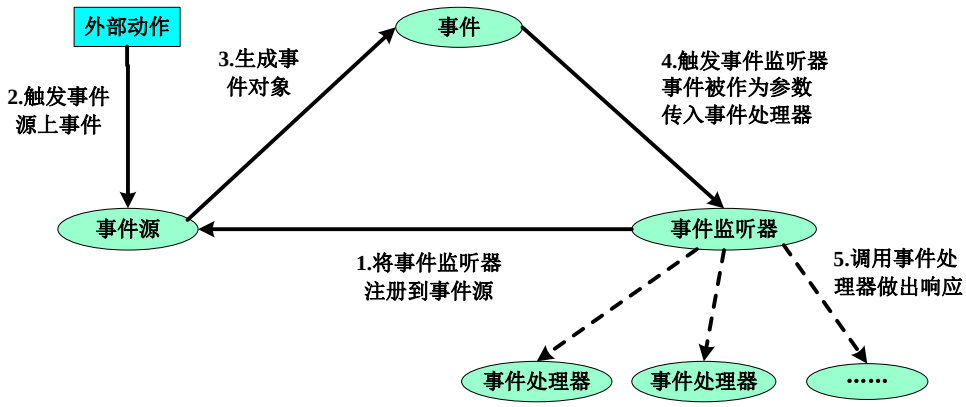


图 3-1 基于监听的事件处理模型

委托式事件处理非常好理解，就类似于生活中我们每个人能力都有限，**当碰到一些自己处理不了的事情时，就委托给某个机构或公司来处理。**你需要把你所遇到的事情和要求描述清楚，这样，其他人才能比较好地解决问题，然后该机构会选派具体的员工来处理这件事。其中，我们**自己**就**是事件源**，你**遇到的事情**就**是事件**，该**机构**就是**事件监听器**，**具体**解决事情的**员工就是事件处理器**。

基于监听的事件处理模型的编程步骤主要有：

(1) 获取普通界面控件（事件源），也就是被监听的对象；

(2) 实现事件监听器类，该监听器类是一个特殊的Java类，必须实现一个XxxListerner接口，并实现接口里的所有方法，每个方法用于处理一种事件；

(3) 调用事件源的setXxxListener方法将事件监听器对象注册给普通控件（事件源），即将事件源与事件监听器关联起来，这样，当事件发生时就可以自动调用相应的方法。

在上述步骤中，事件源比较容易获取，一般就是界面控件，根据findViewById()方法即可得到；调用事件源的setXxxListener方法是由系统定义好的，我们只需要传入一个具体的事件监听器；所以，我们所要做的就是实现事件监听器。所谓事件监听器，其实就是实现了特定接口的Java类的实例。在程序中实现事件监听器，通常有如下几种形式。

- 内部类形式：将事件监听器类定义为当前类的内部类；
- 外部类形式：将事件监听器类定义成一个外部类；
- 类自身作为事件监听器类：让 Activity 本身实现监听器接口，并实现事件处理方法；
- 匿名内部类形式：使用匿名内部类创建事件监听器对象；

下面以一个简单的程序来示范基于监听的事件处理模型的实现过程。该程序实现简单文本编辑功能，程序界面布局中定义了一些文本显示框、若干个按钮，以及一个文本编辑框。为所有的按钮注册了单击事件监听器，为文本编辑框注册了编辑事件监听器，为了演示各种实现事件监听器的方式，该程序中使用了四种实现监听器的方式。界面分析与运行效果如图3-2所示。

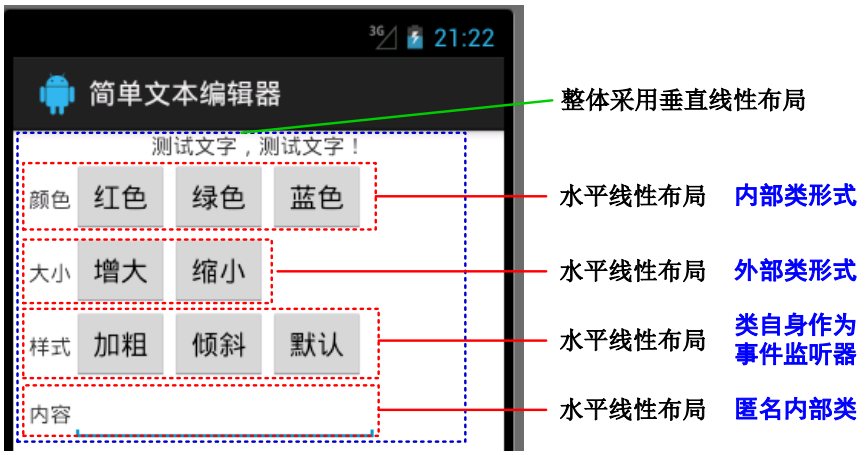


图 3- 2 简单文本编辑器

界面布局文件见后。在该布局文件中，省略了一些类似的代码，保留了整体结构，整体采用垂直线性布局，里面又嵌套了若干个水平线性布局。

界面设计完成后，运行程序，得到上述界面效果，但此时单击按钮时没有任何反应，下面为这些按钮添加事件监听器。

程序清单：codes\ chapter03\TextEditor\res\layout\ activity_main.xml

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:orientation="vertical" >                    →垂直线性布局
6      <TextView
7          android:id="@+id/testText"                      →为文本框添加ID，便于查找
8          android:layout_width="match_parent"             →文本框的宽度为填充父容器
9          android:layout_height="wrap_content"            →文本框的高度为内容包裹
10         android:gravity="center_horizontal"             →文本内容水平居中
11         android:text="@string/test_text" />             →设定文本显示内容
12     <LinearLayout
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:layout_marginLeft="10dp"                →左边距为10dp
16         android:orientation="horizontal" >              →水平线性布局
17         <TextView
18             android:layout_width="wrap_content"
19             android:layout_height="wrap_content"
20             android:text="@string/color" />
21         <Button
22             android:id="@+id/red"                       →为按钮添加ID属性
23             android:layout_width="wrap_content"
24             android:layout_height="wrap_content"
25             android:text="@string/red" />
26           <Button      ... />                           →按钮属性与上面相似，略
```

“p55~62所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

程序清单：codes\ chapter03\EventBinding\src\iet\jxufe\cn\android \MainActivity.java

```
1   public class MainActivity extends Activity {
2   //      private Button myBtn;
3           public void onCreate(Bundle savedInstanceState) {
4                   super.onCreate(savedInstanceState);
5                   setContentView(R.layout.activity_main);
6   //                myBtn = (Button) findViewById(R.id.mybtn);
7   //                      myBtn.setOnClickListener(new OnClickListener() {
8   //                              public void onClick(View v) {
9   //                                      Toast.makeText(MainActivity.this, "监听器中的处理方法",
10  //                                              Toast.LENGTH_SHORT).show();
11  //                              }
12  //                      });
13          }
14          public void clickEventHandler(View source) {
15                  Toast.makeText(this, "自定义事件处理方法", Toast.LENGTH_SHORT).show();
16          }
17  }
```

3.2 Handler 消息传递机制

Android平台不允许Activity新启动的线程访问该Activity里的界面控件，也不允许将运行状态外送出去，这样就会导致新启动的线程无法动态改变界面控件的属性值，与Activity进行交互。但在实际Android应用开发中，尤其是涉及动画的游戏开发，需要让新启动的线程周期性地改变界面控件的属性值，这就需要借助Handler的消息传递机制实现。Handler类的常用方法如表3-3所示。

表 3- 3 Handler 类的常用方法

方法签名	描述
public void handleMessage (Message msg)	通过该方法获取、处理信息
public final boolean sendEmptyMessage (int what)	发送一个只含有 what 值的消息
public final boolean sendMessage (Message msg)	发送消息到 Handler，通过 handleMessage 方法接收
public final boolean hasMessages (int what)	监测消息队列中是否有 what 值的消息
public final boolean post (Runnable r)	将一个线程添加到消息队列

从Handler类的方法可知，Handler类主要有两个作用：

- 在新启动的线程中发送消息；
- 在主线程中获取、处理消息。

那么新启动的线程何时发送消息？主线程又如何去获取并处理消息呢？

为了让主线程能“适时”地处理新启动的线程所发送的消息，显然只能通过回调的方式来实现——我们只要重写Handler类中处理消息的方法，当新启动的线程发送消息时，Handler类中处理消息的方法被自动回调。

开发带有Handler类的程序步骤如下。

(1) 创建Handler类对象，并重写handleMessage()方法；

(2) 在新启动的线程中，调用Handler对象的发送消息方法；

(3) 利用Handler对象的handleMessage()方法接收消息，然后根据不同的消息执行不同的操作。

下面的程序通过一个新线程来动态生成随机数，然后显示在主线程的文本显示框上。该程序界面布局非常简单，只有一个TextView控件，在此不给出界面布局代码。

（1）最初想法：通过启动一个线程，在线程中动态的改变主线程的界面

程序清单：codes\ chapter03\HandlerTest\src\iet\jxufe\cn\android\MainActivity.java

```
1   public class MainActivity extends Activity {
2       private TextView myText;
3       public void onCreate(Bundle savedInstanceState) {
4           super.onCreate(savedInstanceState);
5           setContentView(R.layout.activity_main);
6           myText=(TextView)findViewById(R.id.myText);
7           myText.setText("生成的随机数为："+Math.random());
8           new Thread(new Runnable(){                    →单独启动一个线程动态生成的随机数
9             public void run() {
10                try {
11                    while(true){
12                        Thread.sleep(300);                    →程序休眠0.3秒
13                        Double random=Math.random();          →生成随机数
14                        myText.setText("生成的随机数为："+random);
15            //这句代码无法执行，控制台打印错误信息，Only the original thread that
16            //created a view hierarchy can touch its views.即该线程不能改变
17            //TextView的显示，只有创建TextView的线程可以改变
18                        }
19                } catch (Exception e) {
20                    e.printStackTrace();
21                }
22            };
23          }).start();
24      }
25  }
```

该程序显示的是第一个生成的随机数，没有动态变化的效果。因为Android中不允许子线程更改主线程的界面控件，在控制台会打印出错误信息，但程序不会强制退出。

（2）既然子线程不能更改主线程的界面控件，那么我们模拟一下在主线程中进行更改，代码见后。

```
1   public class MainActivity extends Activity {
2       private TextView myText;
3       public void onCreate(Bundle savedInstanceState) {
4           super.onCreate(savedInstanceState);
5           setContentView(R.layout.activity_main);
6           myText = (TextView) findViewById(R.id.myText);
7           myText.setText("生成的随机数为：" + Math.random());
```

```
8                   try {
9                       for (int i = 0; i < 5; i++) {
10                          Thread.sleep(300);                  →程序休眠0.3秒
11                          Double random = Math.random();
12                          System.out.println(random);         →控制台打印生成的随机数
13                          myText.setText("生成的随机数为：" + random);
14                      }
15                  } catch (Exception e) {
16                      e.printStackTrace();
17                  }
18              }
19  }
```

如果将上面的for循环，改为while(true)循环，程序将进入死循环，没有任何显示。在此模拟5次生成随机数，运行结果发现仍然不能达到效果。查看控制台打印信息发现有5条信息，而此时TextView显示的结果与**最后生成**的随机数相同。原因是，Android中是通过调用onCreate()方法来完成界面的显示，我们这里是在onCreate()方法内进行线程休眠，只是将onCreate()方法的执行过程延迟了，因此无法达到动态改变界面的显示，只能显示一次。除非我们能多次调用onCreate（）方法，而该方法是由系统自动调用的。

（3）使用消息传递机制实现该功能，界面每隔0.3秒更新一次。主要思路，在子线程里发送消息，然后主线程收到消息后进行相应的处理即在主线程修改界面显示。

```
1   public class MainActivity extends Activity {
2       private TextView myText;
3       private Handler myHandler;
4       public void onCreate(Bundle savedInstanceState) {
5           super.onCreate(savedInstanceState);
6           setContentView(R.layout.activity_main);
7           myText = (TextView) findViewById(R.id.myText);
8           myText.setText("生成的随机数为：" + Math.random());
9           myHandler = new Handler() {
10              public void handleMessage(Message msg) {
11                  super.handleMessage(msg);
12                  if(msg.what==0x12){                  →如果该消息是本程序
                                                         所发送的，前后标记一致
13                      myText.setText("生成的随机数为：\n" + Math.random());
14                  }
15              }
16          };
17          new Thread(new Runnable() {
18              public void run() {
19                  try {
20                      while (true) {
21                          Thread.sleep(300);
22                          Message msg=new Message();
23                          msg.what=0x12;               →消息的标记
24                          myHandler.sendMessage(msg);
25                      }
```

```
26                                  } catch (Exception e) {
27                                      e.printStackTrace();
28                                  }
29                              };
30                      }).start();
31              }
32      }
```

该程序重写了Handler类的handleMessage（Message msg）方法，该方法用于处理消息，当新线程发送消息时，该方法会被自动回调，然后根据消息的标记，对不同的消息进行不同的业务逻辑处理，由于handleMessage（Message msg）方法依然位于主线程，所以可以动态的修改TextView控件的文本。

注意：发送消息和处理消息的是同一个Handler对象。

3.3 异步任务处理

在开发Android移动客户端的时候往往要使用多线程来进行操作，我们通常会将耗时的操作放在单独的线程执行，避免其占用主线程而给用户带来不好的用户体验。但是在子线程中无法去操作主线程（UI 线程），在子线程中操作UI线程会出现错误。因此Android提供了一个类Handler在子线程中来更新UI线程，用发消息的机制更新UI界面，呈现给用户。这样就解决了子线程更新UI的问题。但是费时的任务操作总会启动一些匿名的子线程，给系统带来巨大的负担，随之带来一些性能问题。因此Android提供了一个工具类AsyncTask，即异步执行任务。这个AsyncTask生来就是处理一些后台的比较耗时的任务，给用户带来良好用户体验的，从编程的语法上显得优雅了许多，不再需要子线程和Handler就可以完成异步操作并且刷新用户界面。

Android的AsyncTask类对线程间通讯进行了包装，提供了简易的编程方式来使后台线程和UI线程进行通讯，即后台线程执行异步任务，并把操作结果通知UI线程。

AsyncTask是抽象类，AsyncTask定义了三种泛型类型 **Params**、**Progress**和**Result**。

Params：启动任务执行的输入参数，如HTTP请求的URL；

Progress：后台任务执行的百分比；

Result：后台执行任务最终返回的结果，如String，Integer等。

AsyncTask类中主要有以下几个方法：

onPreExecute()：该方法将在执行实际的后台操作前被**UI线程**调用。可以在该方法中做一些准备工作，如在界面上显示一个进度条，或者一些控件的实例化，这个方法可以不用实现。

doInBackground(Params...)：将在onPreExecute 方法执行后马上执行，该方法运行在**后台线程**中。这里将主要负责执行那些比较耗时的后台处理工作。可以调用publishProgress方法来实时更新任务进度。**该方法是抽象方法，子类必须实现。**

onProgressUpdate(Progress...)：在publishProgress方法被调用后，UI 线程将调用这个方法从而在界面上展示任务的进展情况，例如通过一个进度条进行展示。

onPostExecute(Result)：在doInBackground执行完成后，onPostExecute 方法将被**UI线程**调用，后台的计算结果将通过该方法传递到UI线程，并且在界面上展示给用户。

onCancelled()：在用户取消线程操作的时候调用。在主线程中调用onCancelled()的时候调用。

doInBackground方法和onPostExecute的参数必须对应，这两个参数在AsyncTask声明的泛型参数列表中指定，第一个为doInBackground接收的参数，第二个为显示进度的参数，第三个为doInBackground返回值和onPostExecute传入的参数。

为了正确使用AsyncTask类，必须遵守以下几条准则：

（1）AsyncTask的实例必须在UI 线程中创建；

（2）execute(Params...)方法必须在 UI 线程中调用；

（3）不要手动的调用 onPreExecute(), onPostExecute(Result)，doInBackground (Params...), onProgressUpdate(Progress...)等方法，需要在UI线程中实例化这个task来调用；

（4）该 task 只能被执行一次，否则多次调用时将会出现异常。

下面以一个简单的示例，演示AsyncTask的使用，该程序通过睡眠来模拟耗时操作，程序代码如下。

程序清单：codes\ chapter03\ AsyncTaskTest\res\layout\ activity_main.xml

```
1   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2       xmlns:tools="http://schemas.android.com/tools"
3       android:layout_width="match_parent"
4       android:layout_height="match_parent"
5       android:orientation="vertical" >
6        <Button
7           android:id="@+id/myBtn"
8           android:layout_width="wrap_content"
9           android:layout_height="wrap_content"
10          android:text="@string/down"/>
11      <TextView
12          android:id="@+id/myText"
13          android:layout_width="match_parent"
14          android:layout_height="wrap_content" />
15      <ProgressBar
16          android:id="@+id/myBar"
17          android:layout_width="match_parent"
18          android:layout_height="wrap_content"
19          android:visibility="invisible"                    →初始时进度条不可见
20          android:max="100"                                 →进度条最大值为100
21          style="?android:attr/progressBarStyleHorizontal"/>  →设置进度条样式，调
22  </LinearLayout>                                           用系统资源
```

异步任务处理类代码如下。

程序清单：codes\ chapter03\ AsyncTaskTest\src\iet\jxufe\cn\android\DownTask.java

```
1   public class DownTask extends AsyncTask<Integer, Integer, String> {
2       private TextView tv;
3       private ProgressBar pb;
```

```
4       public DownTask(TextView tv,ProgressBar pb){
5           this.tv=tv;                                          →初始化控件
6           this.pb=pb;
7       }
8       public DownTask(){                                       →提供一个无参的构造方法
9       }
10      protected String doInBackground(Integer... param) {
11          for(int i=0;i<=100;i++){
12              publishProgress(i);
13              try{
14                  Thread.sleep(param[0]);
15              }catch (Exception e) {
16                  e.printStackTrace();
17              }
18          }
19          return "下载完毕";
20      }
21      protected void onPreExecute() {
22          super.onPreExecute();
23      }
24      protected void onPostExecute(String result) {            →执行结束后，相关界面控件属性
25          tv.setText(result);                                  的设置
26          tv.setTextColor(Color.RED);
27          tv.setTextSize(20);
28          pb.setVisibility(View.INVISIBLE);
29          super.onPostExecute(result);
30      }
31        protected void onProgressUpdate(Integer... param) {    →更改界面控件的属性
32          tv.setText("当前完成任务的"+param[0]+"%");
33          pb.setProgress(param[0]);
34          tv.setVisibility(View.VISIBLE);
35          pb.setVisibility(View.VISIBLE);
36          super.onProgressUpdate(param);
37      }
38  }
```

程序清单：codes\ chapter03\ AsyncTaskTest\src\iet\jxufe\cn\android\MainActivity.java

```
1   public class MainActivity extends Activity {
2       private Button myBtn=null;
3       private TextView myText=null;
4       private ProgressBar myBar=null;
5       public void onCreate(Bundle savedInstanceState) {
6           super.onCreate(savedInstanceState);
7           setContentView(R.layout.activity_main);
8           myBtn=(Button)findViewById(R.id.myBtn);
9           myText=(TextView)findViewById(R.id.myText);
10          myBar=(ProgressBar)findViewById(R.id.myBar);
11          myBtn.setOnClickListener(new OnClickListener() {
```

```
12                      public void onClick(View v) {
13                          DownTask downTask=new DownTask(myText,myBar);
14                          downTask.execute(100);                    →每隔0.1秒更新一次
15                      }
16                  });
17          }
18  }
```

上面程序中，异步处理类是单独作为一个外部类，放在外面，因此，需要把主线程中的相应的界面控件以参数的形式传递给异步处理类。其实，为了方便可以把异步处理类放在Activity内部，作为它的一个内部类，这样就省去了控件初始化的步骤，可自由调用Activity中的相关控件，更简洁些。但并不提倡这样做，因为异步处理类一般来说业务逻辑比较复杂，放在Activity中会显得比较臃肿，结构比较混乱。

程序的运行效果与执行流程如图3-4所示。初始化时，文本显示框和进度条都是不可见的，界面中只有一个开始下载按钮，单击该按钮后，文本显示框和进度条都显示出来，并且它们的值是动态变化的。当下载完毕后，进度条消失，文本显示框给出下载完毕的提示。

图 3-4 程序运行结果及说明

异步任务处理方法调用顺序如图3-5所示。图中，较密的虚线框里的方法是在主线程中执行的，实线框中的方法是在子线程中执行的，较疏的虚线框里的方法会循环多次调用该方法。具体过程如下：execute()方法传入的参数，将会传给doInBackground()方法，在doInBackground()方法中，循环调用publishProgress()方法，而该方法又会触发onProg ressUpdate()方法，并且publishProgress()方法传入的参数会传递给onProgressUpdate()方法。doInBackground()方法执行结束后，会将结果作为参数传递给onPostExecute()方法，而这些参数的类型，在类声明的时候就已经指定了。

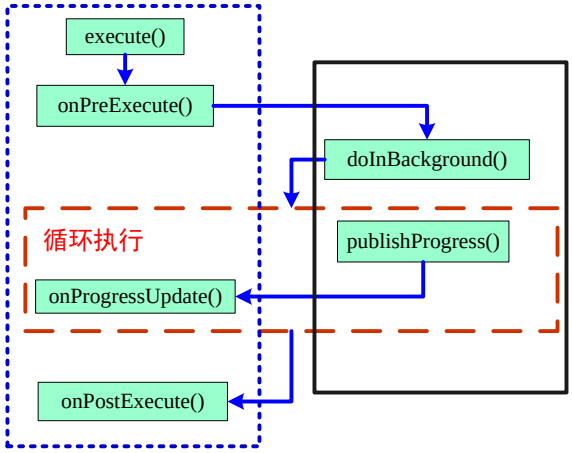


图 3- 5 方法调用流程图

注意：以上方法中只有doInBackground()方法以及publishProgress()方法是在子线程中执行的，其他的方法都是在主线程中执行的，所以可以在这些方法中更新界面控件。

3.4 本章小结

图形界面编程肯定需要与事件处理相结合，当我们设计了界面友好的应用之后，我们必须为界面上的相应控件提供响应，从而当用户操作时，能执行相应的功能，这种响应动作就是由事件处理来完成的。本章重点是掌握Android的事件处理机制：基于监听的事件处理、基于回调的事件处理以及直接绑定到标签的事件处理。了解事件传播的顺序、常见的事件监听器接口及其注册方法。还着重讲解了动态改变界面控件的显示，需要注意的是，Android不允许在子线程中更新主线程的界面控件。因此，我们讲解通过Handler消息处理机制，当需要子线程需要更改界面显示时，子线程就向主线程发送一条消息，主线程接收到消息后，自己对界面显示进行修改。最后，我们讲解了异步任务处理，异步任务处理主要处理一些比较耗时的操作，是对消息处理机制的一种补充。

课后练习

1．Android 中事件处理方式主要有哪三种？
2．基于监听的事件处理模型中，主要包含的三类对象是什么？
3．简单描述基于监听的事件处理的过程？
4. 实现事件监听器的方式有_____、_____、_____和_____。
5．当一个控件，既重写了该控件的事件回调方法、同时重写了该控件所在 Activity 的回调方法、还为其添加了相应的事件监听器，当事件触发时，事件处理的顺序是怎样的？
6．简要描述 Handler 消息传递机制的步骤？
7．使用异步任务处理时，以下方法中，不能更改界面控件显示的是（　）

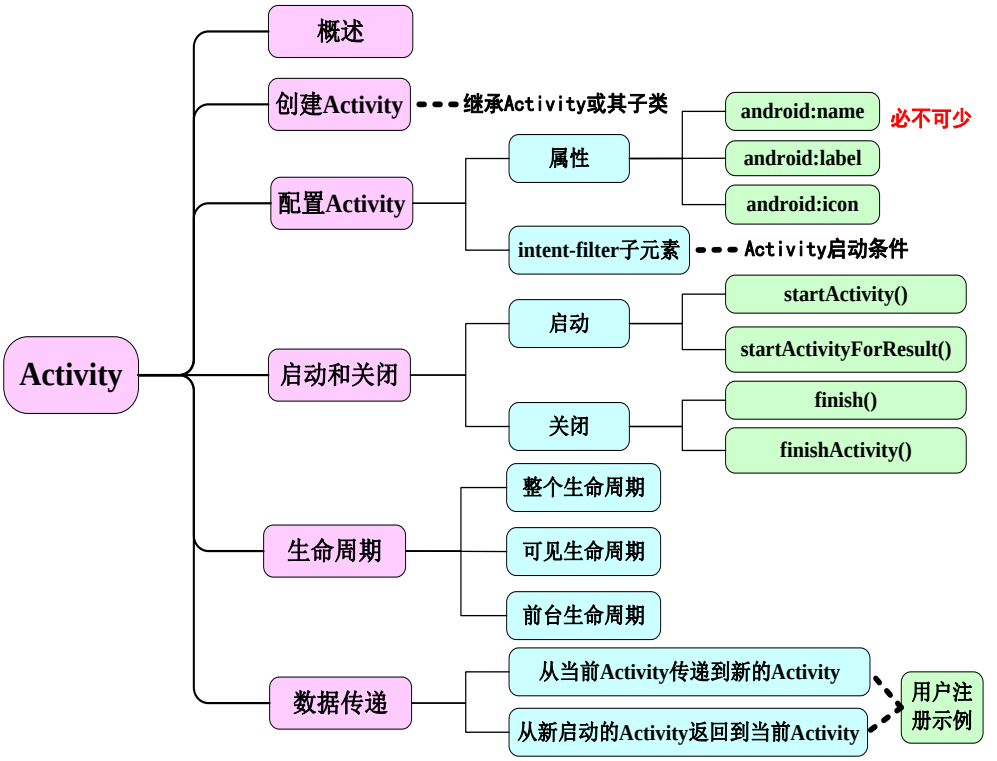
A）onPreExecute()　　B）doInBackground()
C）onPostExecute()　　D）onProgressUpdate()

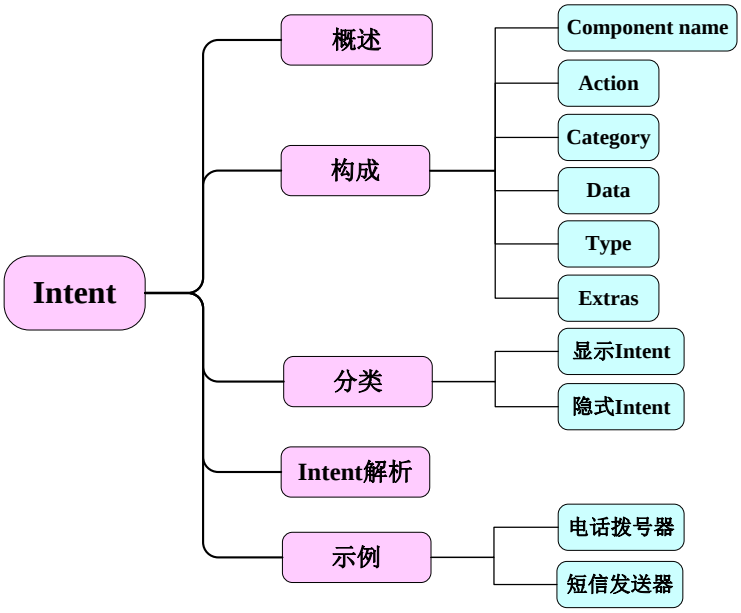
第 4 章 Android 活动与意图(Activity 与 Intent)

本章要点

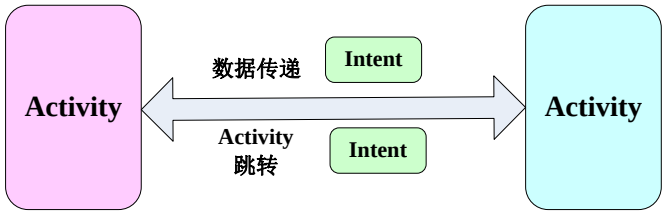
- 理解 Activity 的功能与作用
- 创建和配置 Activity
- 在程序中启动、关闭 Activity
- Activity 的生命周期
- Activity 间的数据传递
- 理解 Intent 的功能与作用
- Intent 的 Action 属性的作用
- Intent 的 Category 属性的作用
- Intent 的 Data 属性的作用
- Intent 的分类
- Intent 的解析

本章知识结构图





Activity 与 Intent 的关系：



本章示例

Android应用通常由一个或多个组件组成，Android中主要包含四大组件：Activity、Service、BroadcastReceiver、ContentProvider。其中Activity是最基础也是最常见的组件，前面我们所写的程序通常都只包含一个Activity。本章我们将详细讲解Activity的相关知识，包括Activity的创建、配置、启动、停止、数据传递以及它的完整生命周期。

Activity是Android应用中负责与用户交互的组件，它为Android应用提供了可视化的用户界面，通过setContentView()方法来指定界面上的组件。如果该Android应用需要多个用户界面，那么这个Android应用将会包含多个Activity，多个Activity组成Activity栈，当前活动的Activity位于栈顶。

一个应用程序往往由多个Activity或其他组件组成，那么Activity间以及各组件间是如何交互或通信的呢？Android中是通过Intent对象来完成这一功能的。本章将详细讲解Intent对象是如何封装组件间的交互，并讲解Intent对象的各种属性以及Intent的过滤机制。

通过本章的学习，读者将可以实现Activity间数据的传递以及通过Intent调用系统中的某些应用，完成诸如用户注册、登录、打电话、发短信等功能。

4.1 Activity 详解

Activity是Android应用中重要组成部分之一，如果把一个Android应用看成是一个网站的话，那么一个Activity就相当于该网站的一个具体网页。Android应用开发的一个重要组成部分就是开发Activity，下面我们将由浅入深详细地讲解Activity的创建、配置、启动、传值以及生命周期等相关知识。

4.1.1 Activity 概述

Activity是Android的一种应用程序组件，该组件为用户提供了一个屏幕，用户在这个屏幕上进行操作即可完成一定的功能，例如打电话、拍照、发送邮件或查看地图等。每一个Activity都有一个用于显示用户界面的窗口。该窗口通常会充满整个屏幕，但有可能比这个屏幕更小或者是漂浮在其他窗口之上。Activity类包含了一个setTheme()方法来设置其窗口的风格，例如我们希望窗口不显示标题、以对话框形式显示窗口，都可通过该方法来实现。

一个应用程序通常是由多个彼此之间松耦合的Activity组成。通常，在一个应用程序中，有一个Activity被指定为主Activity。当应用程序第一次启动的时候，系统会自动运行主Activity，前面的所有例子都只有一个Activity，并且该Activity为主Activity。每个Activity都可以启动其他的Activity用于执行不同的功能。当一个新的Activity启动的时候，先前的那个Activity就会停止，但是系统会在堆栈中保存该Activity。当一个新的Activity启动时，它将会被压入栈顶，并获得用户焦点。堆栈遵循后进先出的队列原则。因此，当用户使用完当前的Activity并按Back键时，该Activity将从堆栈中取出并销毁，然后先前的那个Activity将恢复并获取焦点。

当一个Activity因为新的Activity的启动而停止时，系统将会调用Activity的生命周期的回调方法来通知这一状态的改变。Activity类中定义了一些回调方法，对于具体Activity对象而言，这些回调方法是否会被调用，主要取决于具体状态的改变——系统

“p74~81所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

问题与讨论：

(1) 问题：当MainActivity正在运行时，若此时直接按Home键，返回到桌面，MainActivity是否还存在？控制台会打印什么消息？

提示：并不会调用onDestroy方法。

(2) 问题：前面我们返回到原来的Activity都是使用返回键，如果我们在新启动的Activity中添加一个按钮，单击按钮后，跳转到原来的Activity，这样做与单击返回键有区别吗？

提示：可以在SecondActivity中添加一个Go to MainActivity按钮，并添加相应的处理事件，来观察两者的区别。其关键代码如下。

```
1    Button main=(Button)findViewById(R.id.main);
2        main.setOnClickListener(new OnClickListener() {
3            public void onClick(View v) {
4                Intent intent=new Intent(SecondActivity.this,MainActivity.class);
5                startActivity(intent);
6            }
7        });
```

区别在于：通过Go to MainActivity按钮跳转到MainActivity只是表面现象，实际上系统是重新创建了一个MainActivity，即此时在Activity堆栈中包含两个MainActivity对象。如果重复操作多次，那么Activity堆栈中将会存在多个这样的MainActivity，而通过返回键操作，则是销毁当前的Activity，从而使上一个Activity获取焦点，重新显示在前台，它是不断地从堆栈中取出Activity。

4.1.5 Activity 间的数据传递

1．Activity间数据传递的方法——采用intent对象

前面我们学习了Activity的生命周期，Activity间的跳转，实际应用中，仅仅有跳转还是不够的，往往还需要进行通信，即数据的传递。在Android中，主要是通过Intent对象来完成这一功能的，Intent对象就是它们之间的信使。

数据传递方向有两个：一个是从当前Activity传递到新启动的Activity，另一个是从新启动的Activity返回结果到当前Activity。下面详细讲解这两种情景下数据的传递。

在介绍Activity启动方式时，我们知道Activity提供了一个startActivityForResult (Intent intent，int requestCode)方法来启动其他Activity。该方法可以将新启动的Activity中的结果返回给当前Activity。如果要使用该方法，还必须做以下操作。

(1)在当前Activity中重写onActivityResult(int requestCode ,int resultCode,Intent intent)方法，其中requestCode代表请求码，resultCode代表返回的结果码;

(2)在启动的Activity执行结束前，调用该Activity的setResult(int resultCode,Intent intent)方法，将需要返回的结果写入到Intent中。

整个执行过程为：当前Activity调用startActivityForResult(Intent intent，int requestCode)方法启动一个符合Intent要求的Activity之后，执行它相应的方法，并将执行

结果通过setResult(int resultCode,Intent intent)方法写入Intent，当该Activity执行结束后，会调用原来Activity的onActivityResult(int requestCode ,int resultCode,Intent intent)，判断请求码和结果码是否符合要求，从而获取Intent里的数据。

请求码和结果码的作用：因为在一个Activity中可能存在多个控件，每个控件都有可能添加相应的事件处理，调用startActivityForResult()方法，从而就有可能打开多个不同的Activity处理不同的业务。但这些Activity关闭后，都会调用原来Activity的onActivityResult（int requestCode, int resultCode, Intent intent）方法。通过请求码，我们就知道该方法是由哪个控件所触发的，通过结果码，我们就知道返回的数据来自于哪个Activity。

Intent保存数据的方法：从当前Activity传递数据到新启动的Activity相对来说比较简单，只需要将需要传递的数据存入到Intent即可。上面两种传值方式，都需要将数据存入Intent，**那么Intent是如何保存数据的呢？**Intent提供了多个重载的方法来存放额外的数据，主要格式如下：

putExtras(String name, Xxx data)：其中Xxx表示数据类型，向Intent中放入Xxx类型的数据，例如int、long、String等；

此外还提供了一个putExtras(Bundle data)方法，该方法可用于存放一个数据包，Bundle类似于Java中的Map对象，存放的是键值对的集合，可把多个相关数据放入同一个Bundle中，Bundle提供了一系列的存入数据的方法，方法格式为putXxx(String key, Xxx data)，向Bundle中放入int、long、String等各种类型的数据。为了取出Bundle数据携带包中的数据，Bundle还提供了相应的getXxx（String key）方法，从Bundle中取出各种类型的数据。

2．Activity间的数据传递举例

下面我们用一个完整的注册案例讲解Activity间的数据传递。程序运行效果如图4-10所示。当我们单击所在地时，程序会弹出省份下拉列表，选择某一省份后，会显示该省份下的城市列表供用户选择，如图4-11、图4-12所示。

填写完相应信息后，单击注册按钮，系统会对用户填写的信息进行简单的验证，如果用户名未填写，则会弹出如图4–13所示对话框；如果密码位数过短或过长则弹出如图4–14所示对话框；如果两次密码不一致，则弹出如图4–15所示对话框。如果用户注册信息符合要求，则跳转到注册成功页面，如图4–16所示。

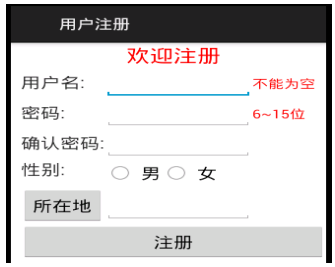


图 4- 10 程序运行界面图

图 4- 11 省份选择列表

图 4- 12 城市选择列表

图 4- 13 注册信息提示图（1）

图4- 14 注册信息提示图（2）

图4- 15 注册信息提示图（3）

图 4- 16 注册成功界面图

用户注册界面设计方案如图4-17所示。图4-18给出实现用户注册的程序结构。

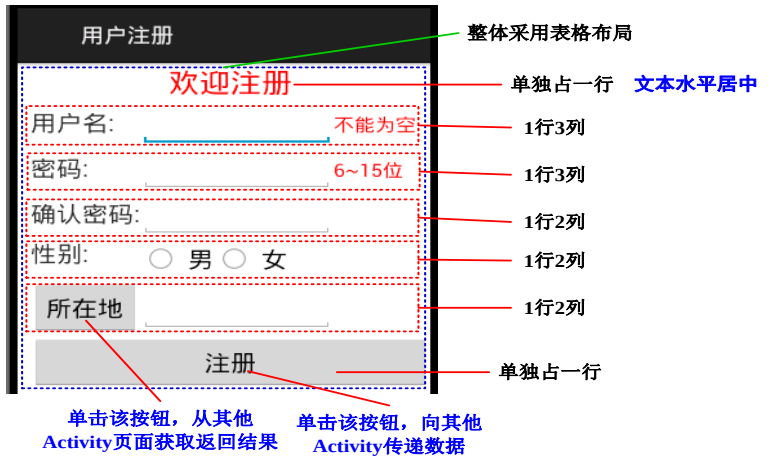


图 4- 17 用户注册界面设计方案图

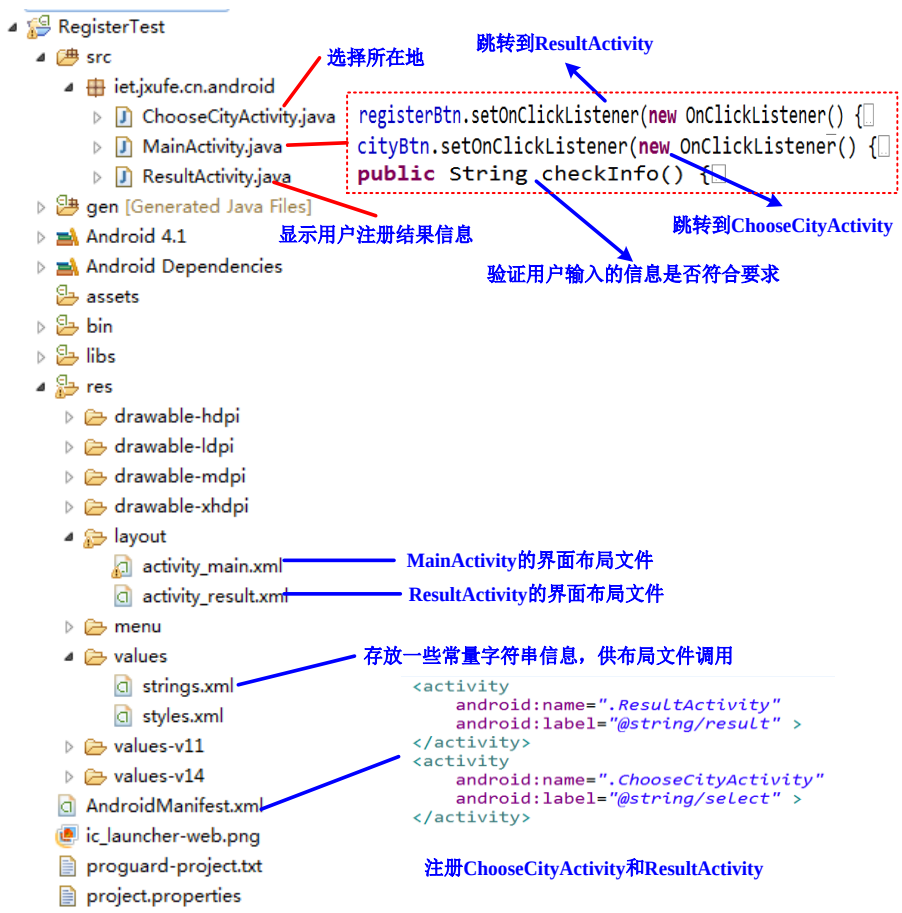


图 4- 18 用户注册程序结构图

“p86~90所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

```
65                          data.putString("city" ,cities[groupPosition][childPosition]);      Activity对应的Intent
66                              intent.putExtras(data);
67                              ChooseCityActivity.this.setResult(0 , intent);                 → 设置结果码和退回的Activity
68                              ChooseCityActivity.this.finish();                              →结束
69                              return false;                                                  SelectCityActivity
70                          }
71                  });
72          }
73  }
```

结果显示界面的Activity为ResultActivity，该Activity主要就是获取Intent中的数据，然后一个个显示在对应的TextView上，布局文件比较简单，在此不列出，详细代码如下。

程序清单：codes\chapter04\RegisterTest\src\iet\jxufe\cn\android\ResultActivity.java

```
1   public class ResultActivity extends Activity {
2       protected void onCreate(Bundle savedInstanceState) {
3           super.onCreate(savedInstanceState);
4           setContentView(R.layout.activity_result);
5           TextView resultName=(TextView)findViewById(R.id.resultName);
6           TextView resultPsd=(TextView)findViewById(R.id.resultPsd);
7           TextView resultGender=(TextView)findViewById(R.id.resultGender);
8           TextView resultCity=(TextView)findViewById(R.id.resultCity);
9           Intent intent=getIntent();                          →获取传递过来的Intent
10          resultName.setText(intent.getStringExtra("name"));  →从Intent中获取值
11          resultPsd.setText(intent.getStringExtra("psd"));
12          resultGender.setText(intent.getStringExtra("gender"));
13          resultCity.setText(intent.getStringExtra("city"));
14      }
15  }
```

注意：要实现这个功能，必须要在AndroidManifest.xml文件中，配置ChooseCity-Activity和ResultActivity，配置信息如下。

```
1   <activity
2           android:name=".ResultActivity"                  →注册ResultActivity
3           android:label="@string/result" >
4       </activity>
5       <activity
6           android:name=".ChooseCityActivity"              →注册ChooseCityActivity
7           android:label="@string/select" >
8       </activity>
```

4.2 Intent 详解

在前面介绍启动Activity以及Activity间传值时，发现都需要传递一个Intent对象作为参数。事实上，Android应用程序中的三种核心组件：Activity、Service、

BroadcastReceiver彼此之间是独立的，它们之间之所以可以互相调用、协调工作，最终组成一个真正的Android应用，主要是通过Intent对象协助来完成的。下面将对Intent对象进行详细的介绍。

4.2.1 Intent 概述

“Intent”中文翻译为“意图”，是对一次即将运行的操作的抽象描述，包括操作的动作、动作涉及数据、附加数据等，Android系统则根据Intent的描述，负责找到对应的组件，并将Intent传递给调用的组件，完成组件的调用。因此，Intent在这里起着媒体中介的作用，专门提供组件互相调用的相关信息，实现调用者与被调用者之间的解耦。

例如，我们想通过联系人列表查看某个联系人的详细信息，点击某个联系人后，希望能够弹出此联系人的详细信息。为了实现这个目的，联系人Activity需要构造一个Intent，这个Intent 用于告诉系统，我们要做“查看”动作，此动作对应的查看对象是“具体的某个联系人”，然后调用startActivity (Intent intent)，将构造的Intent 传入，系统会根据此Intent中的描述，到AndroidManifest.xml中找到满足此Intent 要求的Activity，最终传入Intent，对应的Activity则会根据此Intent 中的描述，执行相应的操作。

Intent实际上就是一系列信息的集合，既包含对接收该Intent的组件有用的信息，如即将执行的动作和数据，也包括对Android系统有用的信息，如处理该Intent的组件的类型以及如何启动一个目标Activity。

4.2.2 Intent 构成

Intent封装了要执行的操作的各种信息，那么，Intent是如何保存这些信息的呢？事实上，Intent对象中包含了多个属性，每个属性就代表了该信息的某个特征，对于某一个具体的Intent对象而言，各个属性值都是确定的，Android应用就是根据这些属性值去查找符合要求的组件。从而启动合适的组件执行该操作。下面我们就来详细学习Intent中的各种属性及其作用和典型用法。

(1) Component name（组件名）：指定Intent的目标组件名称，即组件的类名。通常Android会根据Intent 中包含的其他属性信息进行查找，比如action、data/type、category，最终找到一个与之匹配的目标组件。但是，如果component这个属性有指定的话，将直接使用它指定的组件，而不再执行上述查找过程。指定了这个属性以后，Intent的其他所有属性都是可选的。Intent的Component name属性需要接受一个ComponentName对象，创建ComponentName对象时需要指定包名和类名，从而可唯一确定一个组件类，这样应用程序即可根据给定的组件类去启动特定的组件。

```
1  ComponentName comp=new ComponentName(Context con,Class class);   →创建一个ComponentName对象
2  Intent intent=new Intent();
3  intent.setComponent(comp);                                       →为Intent设置Component属性
```

实际上，上面三行代码完全等价于我们前面所用的创建Intent的一行代码，见下面：

```
1  Intent intent=new Intent(Context con Class class);
```

在被启动的组件中，通过以下语句即可获取相关 ComponentName 的信息：

```
1  ComponentName comp=getIntent().getComponent();
2  comp.getPackageName();                          →获取组件的包名
3  comp.getClassName();                            →获取组件的类名
```

(2) Action（动作）：Action代表该Intent所要完成的一个抽象“动作”，这个动作具体由哪个组件来完成，Action这个字符串本身并不管。比如Android提供的标准Acton:Intent.ACTION_VIEW，它只表示一个抽象的查看操作，但具体查看什么，启动哪个Activity 来查看，它并不知道（这取决于Activity的<intent-filter.../>配置，只要某个Activity的<intent-filter.../>配置中包含了该ACTION_VIEW，该Activity就有可能被启动）。Intent类中定义了一系列的Action常量，具体的可查阅**Android SDK→reference**中的**Android.content.intent**类，通过这些常量我们能调用系统为我们提供的功能。

Intent类中为我们提供的一些Action常量，如表4-1所示。

表 4- 1 Intent 类中部分 Action 常量表

编号	Action 名称	AndroidManifest.xml 配置名称	描述
1	ACTION_MAIN	android.intent.action.MAIN	作为应用程序的入口，不需要接收数据
2	ACTION_VIEW	android.intent.action.VIEW	用于数据的显示
3	ACTION_DIAL	android.intent.action.DIAL	调用电话拨号程序
4	ACTION_EDIT	android.intent.action.EDIT	用于编辑给定的数据
5	ACTION_PICK	android.intent.action.PICK	从特定的一组数据中进行数据的选择操作
6	ACTION_RUN	android.intent.action.RUN	运行数据
7	ACTION_SEND	android.intent.action.SEND	调用发送短信程序
8	ACTION_CHOOSER	android.intent.action.CHOOSER	创建文件操作选择器

(3) Category（类别）：执行动作的组件的附加信息。例如LAUNCHER_CATEGORY表示Intent 的接受者应该在Launcher中作为顶级应用出现；而ALTERNATIVE_CATEGORY表示当前的Intent是一系列的可选动作中的一个，这些动作可以在同一块数据上执行。同样的，在Intent类中定义了一些Category常量。

一个**Intent对象最多只能包括一个Action属性**，程序可调用的setAction(String str)方法来设置Action属性值；但一个Intent对象可以**包含多个Category属性**，程序可调用Intent的addCategory（String str）方法来为Intent添加Category属性。当程序创建Intent时，该Intent默认启动Category属性值为Intent.CATEGORY_DEFAULT常量的组件。

Intent 中部分 Category 常量及对应的字符串和作用如表 4-2 所示。

表 4- 2 Intent 类中部分 Category 常量表

编号	Category 常量	对应字符串	简单描述
1	CATEGORY_DEFAULT	android.intent.category.DEFAULT	默认的 Category
2	CATEGORY_BROWSABLE	android.intent.category. BROWSABLE	指定该 Activity 能被浏览器安全调用
3	CATEGORY_TAB	android.intent.category. TAB	指定 Activity 作为 TabActivity 的 Tab 页
4	CATEGORY_LAUNCHER	android.intent.category. LAUNCHER	Activity 显示在顶级程序列表中

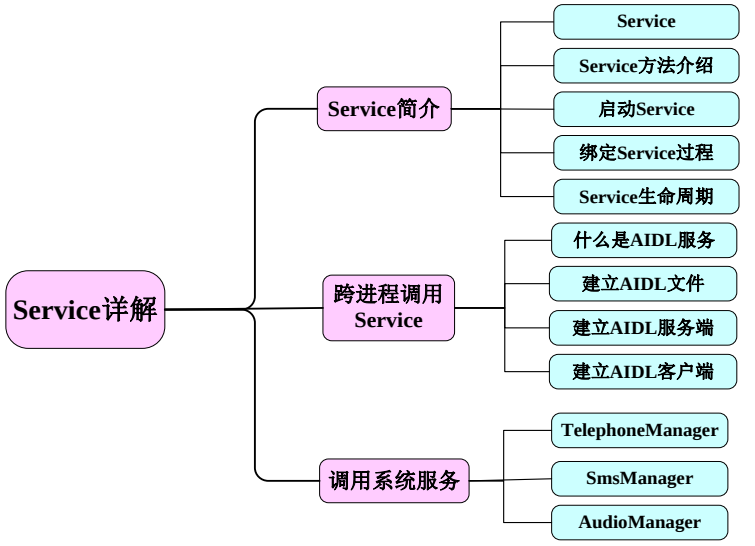
“p94~100所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 5 章 Android 服务(Service)

本章要点

- Service 组件的作用和意义
- Service 与 Activity 的区别
- 运行 Service 的两种方式、绑定 Service 执行的过程
- Service 的生命周期
- 跨进程调用 Service
- 调用系统打电话、发短信服务
- 调用系统播放音频的服务

本章知识结构图



本章示例

根据Activity的生命周期，程序中每次只有一个Activity处于激活状态，并且Activity的执行时间有限，不能做一些比较耗时的操作。当我们需要多种工作同时进行时，如一边听音乐，一边浏览网页，则比较困难。针对这种情况，Android为我们提供了另一种组件——服务（Service）。

5.1 Service 概述

Service是一种Android应用程序组件，可在后台运行一些耗时但不显示界面的操作。

5.1.1 Service 介绍

Service与Activity类似，都是Android中四大应用程序组件，并且二者都是从Context派生而来，最大的区别在于Service没有实际的界面，而是一直在Android系统的后台运行，相当于是一个没有图形界面的Activity程序，它不能自己直接运行，需要借助Activity或其他Context对象启动。

Service主要实现有两种用途：后台运行和跨进程访问。通过启动一个服务，可以在不显示界面的前提下在后台运行指定的任务，这样可以不影响用户做其他事情，如后台运行音乐播放，前台显示网页信息。而通过AIDL服务可以实现不同进程之间的通信，这也是Service的重要用途之一。

其他的应用程序组件一旦启动服务，该服务将会一直运行，即使启动它的组件跳转到其他页面或销毁了。此外，组件还可以与Service绑定，从而与之进行交互，甚至执行一些进程内通信。例如：服务在后台执行网络连接、播放音乐、执行文件操作或者是与内容提供者交互等等。

5.1.2 启动 Service 的两种方式

在Android系统中，常采用以下两种方式启动Service。

（1）通过Context的startService()启动Service后，访问者与Service之间没有关联，该Service将一直在后台执行，即使调用startService的进程结束了，Service仍然还存在，直到有进程调用stopService()，或者Service自杀（stopSelf()）。这种情况下，Service与访问者之间无法进行通信、数据交换，往往用于执行单一操作，并且没有返回结果。例如通过网络上传、下载文件，操作一旦完成，服务应该自动销毁。

（2）通过Context的bindService()绑定Service，绑定后Service就和调用bindService()的组件同生共死了，也就是说当调用bindService()的组件销毁了，那么它绑定的Service也要跟着被结束，当然期间也可以调用unbindservice()让Service提前结束。**注意：**一个服务可以与多个组件绑定，只有当所有的组件都与之解绑后，该服务才会被销毁。

以上两种方式也可以混合使用，即一个Service既可以启动也可以绑定，只需要同时实现onStartedCommand()（用于启动）和onBind()（用于绑定）方法，那么只有调用stopService()，并且调用unbindservice()方法后，该Service才会被销毁。

注意：服务运行在它所在进程的主线程，服务并没有创建它自己的线程，也没有

运行在一个独立的进程上（单独指定的除外），这意味着，如果你的服务做一些消耗CPU或者阻塞的操作，你应该在服务中创建一个新的线程去处理。通过使用独立的线程，你就会降低程序出现ANR（Application No Response程序没有响应）的风险，程序的主线程仍然可以保持与用户的交互。

5.1.3 Service 中常用方法

与开发其他Android组件类似，开发Service组件需要先开发一个Service子类，该类需继承系统提供的Service类，系统中Service类包含的方法主要有：

- abstract IBinder onBind(Intent intent)：该方法是一个抽象方法，因此所有 Service 的子类必须实现该方法。该方法将返回一个 IBinder 对象，应用程序可通过该对象与 Service 组件通信；
- void onCreate()：当 Service 第一次被创建时，将立即回调该方法；
- void onDestroy()：当 Service 被关闭之前，将回调该方法；
- void onStartCommand(Intent intent, int flags,int startId)：该方法的早期版本是 void onStart(Intent intent, int startId)，每次客户端调用 startService(Intent intent)方法启动该 Service 时都会回调该方法；
- boolean onUnbind(Intent intent)：当该 Service 上绑定的所有客户端都断开连接时将会回调该方法。

定义的Service子类必须实现onBind()方法，然后还需在AndroidManifest.xml文件中对该Service子类进行配置，配置时可通过<intent-filter.../>元素指定它可被哪些Intent启动。下面的具体来创建一个Service子类并对它进行配置，代码如下。

代码清单：codes\ chapter05\FirstService\src\iet\jxufe\cn\android\MyService.java

```
1   public class MyService extends Service {                          →自定义服务类
2       private static final String TAG = "MyService";
3       public IBinder onBind(Intent arg0) {                          →重写onBind方法
4           Log.i(TAG, "MyService onBind invoked!");
5           return null;
6       }
7       public void onCreate() {                                      →重写onCreate方法
8           Log.i(TAG, "MyService onCreate invoked!");
9           super.onCreate();
10      }
11      public void onDestroy() {                                     →重写onDestroy方法
12          Log.i(TAG, "MyService onDestroy invoked!");
13          super.onDestroy();
14          this.quit = true;
15      }
16      public int onStartCommand(Intent intent, int flags, int startId) {   →重写onStartCommand方法
17          Log.i(TAG, "MyService onStartCommand invoked!");
18  return super.onStartCommand(intent, flags, startId);
19      }
20  }
```

“p104~115所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

```
20                                  ex.printStackTrace();
21                              }
22                          }
23                  });
24          }
25          private ServiceConnection conn=new ServiceConnection() {        →创建ServiceConnection对象
26                  public void onServiceDisconnected(ComponentName name) {
27                          songBinder=null;
28                  }
29                  public void onServiceConnected(ComponentName name, IBinder service) {
30                          songBinder=Song.Stub.asInterface(service);      →将代理类转换成IBinder对象
31                  }
32           };
33           protected void onDestroy() {                                   →销毁时解绑Service
34                  super.onDestroy();
35                  unbindService(conn);                                   →解绑Service
36          };
37  }
```

客户端通过Song.Stub.asInterface(service);来得到对象代理，从而获取AIDL接口。运行该程序，单击“获取其他应用信息”按钮，可以看到如图5-13所示的输出。

图 5- 13 获取其他应用信息运行效果

5.3 调用系统服务

在Android系统中提供了很多内置服务类，通过它们提供的系列方法，可以获取系统相关信息。本节将通过调用系统的短信服务来介绍调用系统服务的一般步骤和流程，帮助读者理解和使用系统提供的服务。

发送短信需要调用系统的短信服务，主要用到SmsManager管理类，该类可以实现短信的管理功能，通过sendXxxMessage()方法进行短信的发送操作，例如sendTextMessage()方法用于发送文本信息，该类中包含若干常量如表8-1所示来反映短信的状态。

表 5- 1 SmsManager 类常用常量

序号	常量	类型	描述
1	RESULT_ERROR_GENERIC_FAILURE	常量	表示普通错误
2	RESULT_ERROR_NO_SERVICE	常量	当前没有可用服务
3	RESULT_ERROR_NULL_PDU	常量	表示没有 PDU 提供者
4	RESULT_ERROR_RADIO_OFF	常量	关闭无线广播
5	STATUS_ON_ICC_FREE	常量	表示自由空间
6	STATUS_ON_ICC_READ	常量	短信已读
7	STATUS_ON_ICC_SENT	常量	短信已发送
8	STATUS_ON_ICC_UNREAD	常量	短信未读
9	STATUS_ON_ICC_UNSENT	常量	短信未发送

以下程序提供两个文本框，分别输入收信人的号码和发送的短信内容，通过点击“发送短信”按钮即可将短信发送出去，程序见**code\05\SendMessage\src\iet\jxufe\cn\android\MainActivity.java**，关键代码解析如图5-14所示。

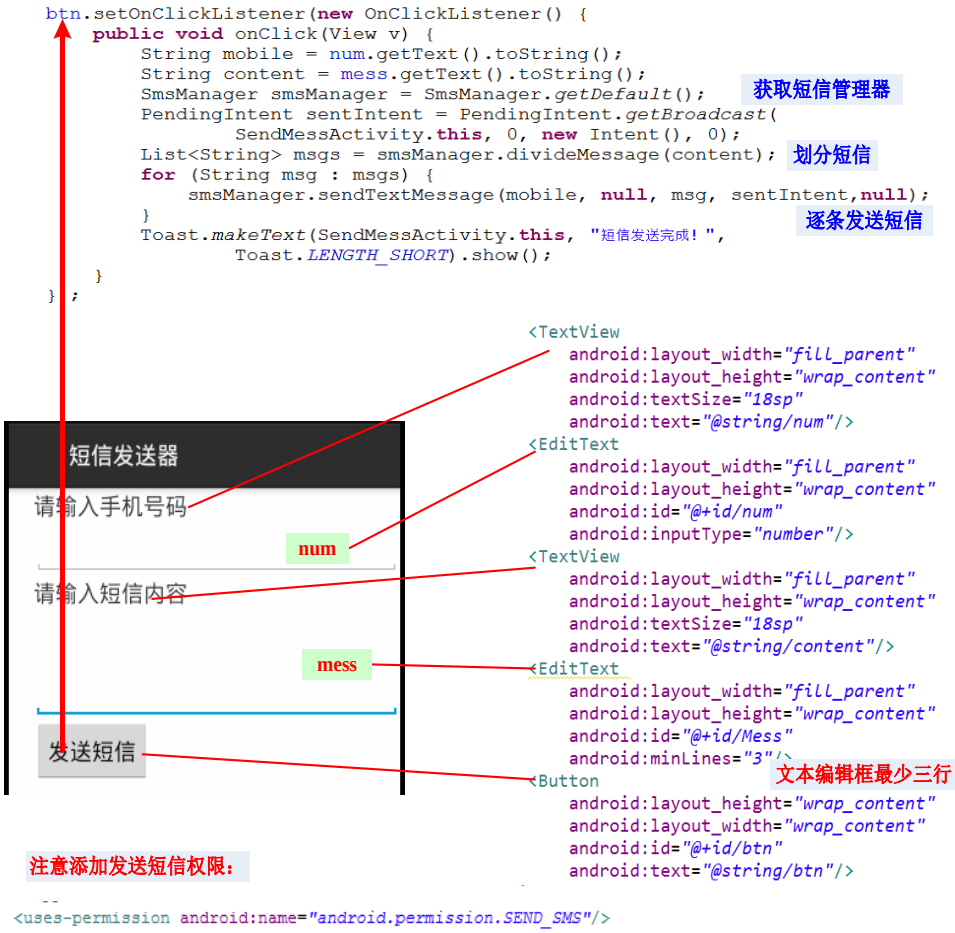


图 5- 14 发送短信

上面程序中用到了一个PendingIntent对象，PendingIntent是对Intent的包装，一般通过调用Pendingintent的getActivity()、getService()、getBroadcastReceiver()静态方法来获取PendingIntent对象。与Intent对象不同的是：PendingIntent通过会传给其他应用组件，从而由其他应用程序来执行PendingIntent所包装的“Intent”。

此外，本程序调用了系统的短信服务，因此，还需要在AndroidManifest.xml文件中添加相应的操作权限，代码如下。

```
1    <uses permission android:name="android.permission.SEND_SMS"/>          →发送短信的权限
```

5.4 本章小结

本章主要介绍了Service的相关知识，Service是Android中的四大组件之一，它与Activity非常类似，都是从Context类派生而来，主要区别是：Service没有用户界面，主要是在后台运行，执行一些比较耗时操作，而不影响用户体验。学习了Service的创建、配置、启动以及Service的生命周期，理解了两种不同的运行Service方式之间的差别，以及如何在Activity中获取Service执行的状态信息。在此基础上学习了如何通过Service实现跨进程的信息访问。主要是通过AIDL服务来完成的，应熟悉AIDL文件的创建、跨进程访问信息的步骤。

课后练习

1．运行服务的两种方式是_____和_____。

2．简述运行服务的两种方式的区别。

3．简述绑定服务执行的过程。

4．在创建 Service 子类时，必须重写父类的以下哪个方法（　　）。

A) onCreate()　　B) onBind()

C)onStartCommand()　　D）onDestroy()

5．以下关于 startService()与 bindService()运行 Service 的说法不正确的是（　　）

A）startService()运行的 Service 启动后与访问者没有关联，而 bindService()运行的 Service 将于访问者共存亡

B）startService()运行的 Service 将回调 onStartCommand()方法，而 bindService 运行的 Service 将回调 onBind()方法

C）startService()运行的 Service 无法与访问者进行通信、数据传递，bindService()运行 Service 可在访问者与 Service 之间进行通信、数据传递

D）bindService 运行的 Service 必须实现 onBind()方法，而 startService()运行的 Service 则没有这个要求

6．AIDL 的全称是什么？该文件有什么作用？

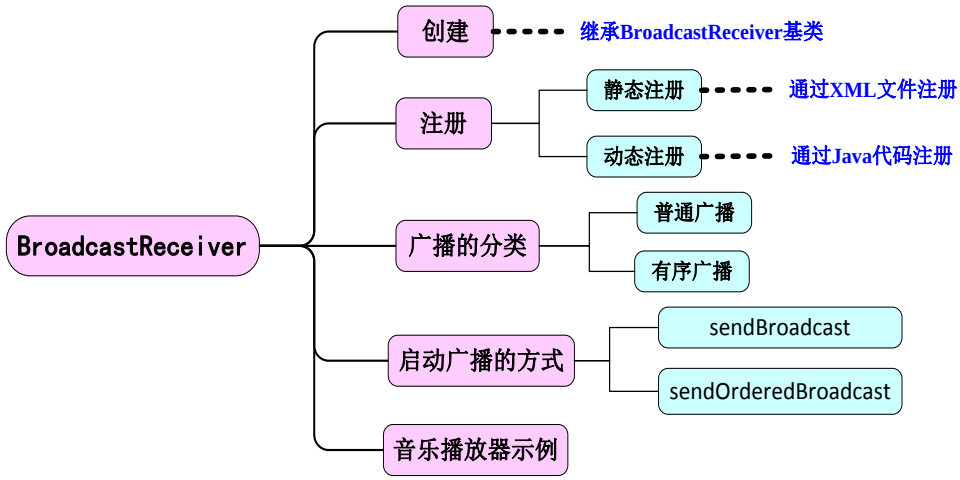
7．简述跨进程访问数据的一般步骤。

第 6 章 Android 广播接收器（BroadcastReceiver）

本章要点

- BroadcastReceiver 的创建
- BroadcastReceiver 的注册
- 发送广播的两种方式
- 普通广播与有序广播
- 简易音乐播放器程序开发

本章知识结构图



本章示例

前面一章我们讲解了Service，我们可以将一些比较耗时的操作放在Service中执行，通过调用相应的方法来获取Service中数据的状态，如果我们需要得到某一特定的数据状态，那么我们需要每隔一段时间调用一次该方法然后判断是否达到我们想要的状态，非常的不方便。如果Service中能够在数据状态满足一定条件时，就主动通知我们，那就非常人性化了。在Android中提供了这样一个组件：BroadcastReceiver（广播接收者），该组件也是Android四大组件之一，它本质上就是一个全局的监听器，一直监听着某一消息，一旦收到该消息则触发相应的方法进行处理，因此可以非常方便地在不同组件间通信。最典型的应用就是电量提醒，当手机电量低于某一设定值时，则发出通知信息。

本章我们将实现一个音乐播放器的示例，当我们单击播放按钮时，界面中按钮发生变化，并显示当前正在播放的歌曲名和演唱者，音乐播放放在Service中执行，当一首歌曲播放结束后，自动播放下一首，同时界面显示也会做相应变化。本程序需要在Activity、Service之间进行双向交互，需要使用到BroadcastReceiver作为中介，通知何时进行更新。

6.1 BroadcastReceiver 介绍

广播是一种广泛运用在应用程序之间传输信息的机制，而BroadcastReceiver是对发送出来的广播进行过滤接收并响应的一类组件。它本质上是一种全局监听器，用于监听系统全局的广播消息，因此它可以非常方便地实现系统中不同组件之间的通信。BroadcastReceiver用于接收广播Intent，广播Intent的发送是通过调用Context.sendBroadcast()、Context.sendOrderedBroadcat()来实现的。通常一个广播Intent可以被订阅了该Intent的多个广播接收者所接收，如同一个广播台，可以被多位听众收听一样。

BroadcastReceiver自身并不实现图形用户界面，但是当它收到某个消息后，可以启动Activity作为响应，或者通过NotificationManager提醒用户，或者启动Service等等。启动BroadcastReceiver与启动Activity、Service非常相似，需要以下两步。

- 创建需要启动的 BroadcastReceiver 的 Intent；
- 调用 Context 的 sendBroadcast()(发送普通广播)或 sendOrderedBroadcast()(发送有序广播)方法来启动指定的 BroadcastReceiver。

当应用程序发出一个 Broadcast Intent 之后，所有匹配该 Intent 的 BroadcastReceiver 都有可能被启动。

BroadcastReceiver是Android四大组件之一，开发自己的BroadcastReceiver与开发其他组件一样，只需要继承Android中的BroadcastReceiver基类，然后实现里面的相关方法即可。

```
1   public class MyBroadcastReceiver extends BroadcastReceiver {      →继承BroadcastReceiver基类
2       public void onReceive(Context context, Intent intent) {       →实现该类的抽象方法
3                                                                      →具体方法的业务处理
4       }
5   }
```

在上面的onReceive()方法中，接收了一个Intent的参数，通过它我们可以获取广播

的数据。创建完我们自己的广播接收者后，并不能马上使用，还必须为它注册一个指定的广播地址，就如同我们有了收音机后，还必须选择收听哪个频道一样。在Android中为BroadcastReceiver注册广播地址有两种方式：静态注册和动态注册。

静态注册：是指在AndroidManifest.xml文件中进行注册。

```
<receiver android:name=".MyBroadcastReceiver">                    →广播接收者对应的类名
    <intent-filter >                                              →设定过滤条件
        <action android:name="iet.jxufe.cn.android.myBroadcastReceiver"></action>
    </intent-filter>
</receiver>
```

动态注册：需要在代码中动态的指定广播地址并注册，通常是在Activity或Service中调用ContextWrapper的registerReceiver（BroadcastReceiver receiver，IntentFilter filter）方法进行注册，代码如下。

```
MyBroadcastReceiver myBroadcastReceiver=new MyBroadcastReceiver();          →创建广播接收者
IntentFilter filter=new IntentFilter("iet.jxufe.cn.android.myBroadcastReceiver");  →设定过滤条件
registerReceiver(myBroadcastReceiver, filter);                              →注册广播接收者
```

注册完成后，即可接收相应的广播消息。一旦广播（Broadcast）事件发生后，系统就会创建对应的BroadcastReceiver实例，并自动触发它的onReceive()方法，onReceive()方法执行完后，BroadcastReceiver的实例就会被销毁。

如果BroadcastReceiver的onReceive()方法不能在10秒内执行完成，Android会认为该程序无响应。所以不要在广播接收者的onReceive()方法里执行一些耗时的操作，否则会弹出ANR（Application No Response）对话框。

如果确实需要根据广播来完成一项比较耗时的操作，则可以考虑通过Intent启动一个Service来完成该操作。不应考虑使用新线程去完成耗时的操作，因为BroadcastReceiver本身的生命周期极短，可能出现的情况是子线程可能还没有结束，BroadcastReceiver就已经退出了。

如果广播接收者所在的进程结束了，虽然该进程内还有用户启动的新线程，但由于该进程内不包含任何活动组件，因此系统可能在内存紧张时优先结束线程。这样就可能导致BroadcastReceiver启动的子线程不能执行完成。

6.2 发送广播的两种方式

广播接收者注册好了以后，并不会直接运行，必须在接收广播后才会被调用，因此，必须首先发送广播，在Android中提供了两种发送广播的方式，调用Context的sendBroadcast()或sendOrderedBroadcast()方法。

- sendBroadcast(Intent intent)：用于发送普通广播，其中 intent 参数表示接收该广播的广播接收者所需要满足的条件，以及广播所传递的数据；
- sendOrderedBroadcast(Intent intent, String receiverPermission)：用于发送有序广播，intent 参数同上，receiverPermission 表示接收该广播的许可权限。

普通广播和有序广播有什么区别呢？

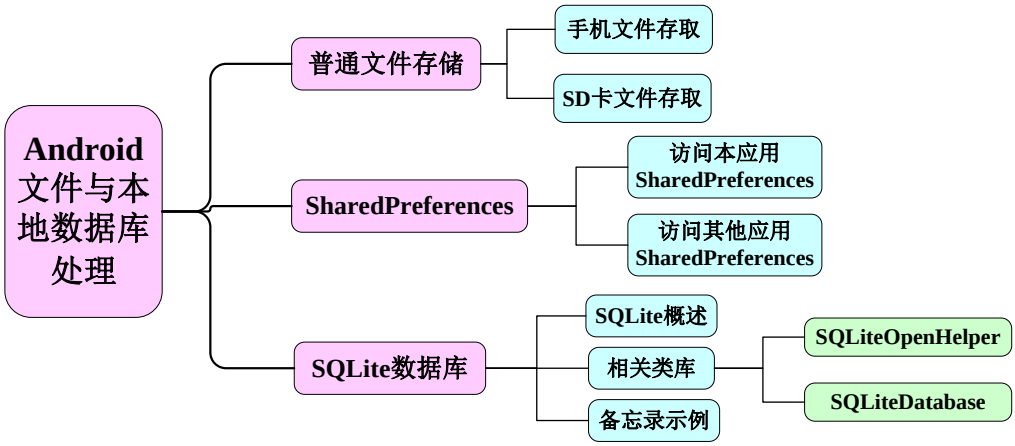
“p122~129所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 7 章 Android 文件与本地数据库(SQLite)

本章要点

- 读、写 Android 手机中存储的普通文件
- 读、写 SharedPreferences（以 XML 存储的配置文件）内容
- SQLite 数据库的使用
- 统一内容提供者——ContentProvider 类及其应用
- 读取网络资源的读取

本章知识结构图



本章示例

文件操作

Sharedpreferences

SQLite

一个比较好的应用程序，应该能够为用户提供一些个性化的设置，能够保存用户的使用记录，而这些都离不开数据的存储。Android系统提供了多种数据存储方式，开发者可根据具体情景选择合适的存储方式，例如你的数据是仅限于本应用程序访问还是允许其他应用程序访问，以及数据所需要的空间等。主要有以下五种方式：

（1）文件存储：以流的方式读取数据；

（2）SharedPreferences：以键值对的形式存储私有的简单的数据；

（3）SQLite数据库：在一个私有的数据库中存储结构化数据；

（4）ContentProvider（内容提供者）：用于在应用程序间共享数据；

（5）网络文件存储：从网络中读取数据，上传数据。

Android应用开发是基于Java的，因此Java IO中的相关经验大部分都可“移植”到Android应用开发上。Android系统还提供了一些专门的输入输出API，通过这些API可以更有效地进行输入、输出操作。

如果应用程序只有少量数据需要保存，那么使用普通文件就可以；如果应用程序只需保存一些简单类型的配置信息，那么使用SharedPreferences就可以；如果应用程序需要保存结构比较复杂的数据时，就需要借助于数据库，Android系统内置了一个轻量级的SQLite数据库，它没有后台进程，整个数据库就对应于一个文件。Android为访问SQLite数据库提供了大量便捷的API；如果想从网络上下载一些资源，则需要用到网络存取。

为了在应用程序之间交互数据，Android提供了一种将私有数据暴露给其他应用程序的方式ContentProvider，ContentProvider是Android的组件之一，是不同应用程序之间进行数据交换的标准API。

本章将详细讲解各种数据存取方式的使用，掌握本章知识，将可以为我们的应用实现普通文件存取和个性化设置参数的设置等操作。

7.1 文件存储

Android是基于Java语言的，在Java中提供了一套完整的输入输出流操作体系，与文件相关的有FileInputStream、FileOutputStream等，通过这些类可以非常方便地访问磁盘上的文件内容。同样的Android也支持这种方式来访问手机上的文件。Android手机中的文件有两个存储位置：内置存储空间和外部SD卡，针对不同位置的文件的存储有所不同，下面分别讲解对它们的操作。

7.1.1 手机内部存储空间文件的存取

Android中文件的读取操作主要是通过Context类来完成的，该类提供了如下两个方法来打开本应用程序的数据文件夹里的文件IO流。

openFileInput(String name)：打开app数据文件夹(data)下name文件对应的输入流；

openFileOutput(String name, int mode)：打开应用程序的数据文件夹下的name文件对应输出流。name参数用于指定文件名称，不能包含路径分隔符“\”，如果文件不存在，Android会自动创建，mode参数用于指定操作模式，Context类中定义了四种操作模

式常量，分别为：

- Context.MODE_PRIVATE=0：为默认操作模式，代表该文件是私有数据，只能被应用本身访问，在该模式下，写入的内容会覆盖原文件的内容。
- Context.MODE_APPEND=32768：模式会检查文件是否存在，存在就往文件追加内容，否则创建新文件再写入内容；
- Context.MODE_WORLD_READABLE=1：表示当前文件可以被其他应用读取；
- Context.MODE_WORLD_WRITEABLE=2：表示当前文件可以被其他应用写入。

提示：如果希望文件既能被其他应用读也能写，可以传入：

Context.MODE_WORLD_READABLE + Context.MODE_WORLD_WRITEABLE或者直接传入数值3，四种模式中除了Context.MODE_APPEND会将内容追加到文件末尾，其他模式都会覆盖掉原文件的内容。

在手机上创建文件和向文件中追加内容的步骤如下：

（1）调用openFileOutput()方法，传入文件的名称和操作的模式，该方法将会返回一个文件输出流；

（2）调用write()方法，向这个文件输出流中写入内容；

（3）调用close()方法，关闭文件输出流。

读取手机上文件的一般步骤如下：

（1）调用openFileInput()方法，传入需要读取数据的文件名，该方法将会返回一个文件输入流对象；

（2）调用read()方法读取文件的内容；

（3）调用close()方法，关闭文件输入流。

下面以一个简单的示例，来演示文件读取的操作，程序运行界面如图7-1所示，界面中包含两个文本输入框，一个用于向文件中写入内容，一个用于显示从文件中读取的内容。界面布局文件如下。

图 7-1 读取手机文件运行界面图

程序清单：codes\chapter07\FileTest\res\layout\ activity_main.xml

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      xmlns:tools="http://schemas.android.com/tools"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:orientation="vertical" >                    →垂直线性布局
6     <EditText
7          android:id="@+id/writeText"
8          android:layout_width="match_parent"
9          android:layout_height="wrap_content"
10         android:minLines="2"                            →设置文本输入框的最少为两行
11         android:hint="@string/hint"/>                   →设置文本输入框的提示信息
12 <Button
13         android:id="@+id/write"
14         android:layout_width="wrap_content"
15         android:layout_height="wrap_content"
16         android:text="@string/write"/>
17 <EditText
18         android:id="@+id/readText"
19         android:layout_width="match_parent"
20         android:layout_height="wrap_content"
21         android:editable="false"                        →设置文本输入框为不可编辑状态
22         android:hint="@string/readhint"/>
23    <Button
24         android:id="@+id/read"
25         android:layout_width="wrap_content"
26         android:layout_height="wrap_content"
27         android:text="@string/read"/>
28 </LinearLayout>
```

在MainActivity.java中分别为写入内容和读取内容按钮添加事件处理，代码如下。

程序清单：codes\ chapter07\FileTest\src\iet\jxufe\cn\android\MainActivity.java

```
1  public class MainActivity extends Activity {
2      private Button read, write;
3      private EditText readText, writeText;
4      private String fileName="content.txt";                  →设置保存的文件名
5      public void onCreate(Bundle savedInstanceState) {
6          super.onCreate(savedInstanceState);
7          setContentView(R.layout.activity_main);
8          read = (Button) findViewById(R.id.read);            →获取读取内容按钮
9          write = (Button) findViewById(R.id.write);          →获取写入内容按钮
10         readText = (EditText)
   findViewById(R.id.readText);
11         writeText = (EditText)
   findViewById(R.id.writeText);
12         read.setOnClickListener(new OnClickListener() {     →添加事件处理
13             public void onClick(View v) {
14                 readText.setText(read());                   →将读取的内容显示在文
15             }                                               本编辑框上
```

```
16                  });
17                  write.setOnClickListener(new OnClickListener() {
18                        public void onClick(View v) {
19                              write(writeText.getText().toString());      →将文本编辑框的内容写
20                        }                                                 入文件
21                  });
22            }
23            public void write(String content) {                         →该方法将字符串内容写
24                  try {                                                   入文件
25                        FileOutputStream fos = openFileOutput(fileName,
      Context.MODE_APPEND);
26                        PrintStream ps = new PrintStream(fos);
27                        ps.print (content);
28                        ps.close();
29                        fos.close();
30                  } catch (Exception e) {
31                        e.printStackTrace();
32                  }
33            }
34            public String read() {                                      →该方法用于读取文件信
35                  StringBuilder sbBuilder = new StringBuilder("");      息，并以字符串返回
36                  try {
37                        FileInputStream is =                            →获取文件输入流
      openFileInput(fileName);
38                        byte[] buffer = new byte[64];                   →定义缓冲区的大小
39                        int hasRead;                                    →记录每次读取的字节数
40                        while ((hasRead = is.read(buffer)) != -1) {
41                              sbBuilder.append(new String(buffer, 0, hasRead));
42                        }
43                  } catch (Exception e) {
44                        e.printStackTrace();
45                  }
46                  return sbBuilder.toString();
47            }
48      }
```

当我们第一次在第一个文本编辑框中，写入一些内容，单击写入内容按钮后，系统首先会查找手机上是否存在该文件，如果不存在则创建该文件。**应用程序的数据文件默认保存在/data/data/<package name>/files目录下，文件的后缀名由开发人员设定。**其中**package name**为当前应用的包名。生成的文件如何查看呢？将当前视图切换到DDMS视图，切换方法是在Eclipse的右上角选择DDMS视图，如果没有DDMS视图，可通过Eclipse的菜单栏中**Window→Open Perspective→other...→DDMS**打开该视图。在该视图中有一个File Explorer面板，可浏览机器上的所有文件，如图7-2所示。

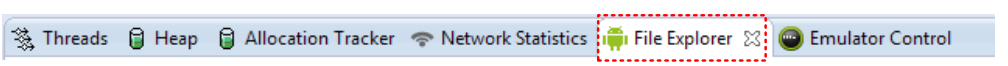


图 7- 2 DDMS 视图中 File Explorer 面板位置

运行程序后，会发现在模拟器的/data/data/iet.jxufe.cn.android/files目录下多了一个context.txt文件。如图7-3所示。这个文件是不能直接在Eclipse中打开的，需要先下载到电脑上才能查看里面的内容。方法是在该面板的右上方，有一个pull a file from the device按钮(从设备上取出文件)，同样我们也可以将电脑上的文件上传到设备中。

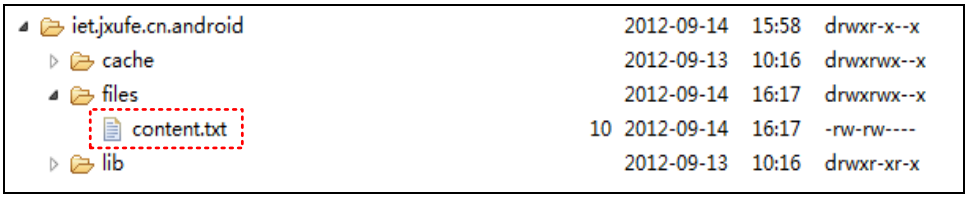


图 7- 3 DDMS 视图中文件生成的位置

默认情况下，内存中的文件只属于当前应用程序，而其他应用程序是不能访问的，当用户卸载了该应用程序时，这些文件也会被移除。

问题与讨论：

(1) 当手机上不存在该文件时，我们先写后读与先读后写有区别吗？程序会不会出错？

具体做法：将手机上的context.txt文件删除，重新启动程序，然后分别进行先写后读与先读后写操作，观察效果）。

注：若手机上不存在该文件，当我们向该文件中写入内容时，系统会自动地为我们创建该文件，而未写先读时，读出的内容为空，系统会生成files文件夹，但并不会生成该文件。只有在写的时候才会生成该文件。实际上，此时系统会在控制台打印出警告信息，但程序不会强制退出。

(2) 不同操作模式的区别，当我们多次执行写入操作时，文件里的内容是被覆盖还是不断地在文件末尾附加新数据？

具体做法：修改openFileOutput()方法的第二个参数。

注：若采用附加模式，则多次写入时，会在文件末尾进行添加新写的内容，不会覆盖以前的内容，如果改成其他模式，则会进行覆盖。

7.1.2 读写 SD 卡上的文件

前面我们学习了如何读取手机内存中的文件，对于手机而言，内存是非常宝贵的，相对而言也是比较小的。内存的空间直接会影响到手机的运行速度，通常不建议将数据保存到手机内存中，特别是一些比较大的资源如图片、音频、视频等。那么这些资源存放在哪里呢？存放在外存上，几乎所有的Android设备，都会配有外存设备，最常见的就是SD卡。

读取SD卡上的文件和读取手机上的文件类似，都是通过文件操作流的方式进行读取的，Android中没有提供单独的SD卡文件操作类，直接使用Java中的文件操作即可，关键是如何确定文件的位置。因为SD卡的可移动性，因此，在访问之前，我们需要验

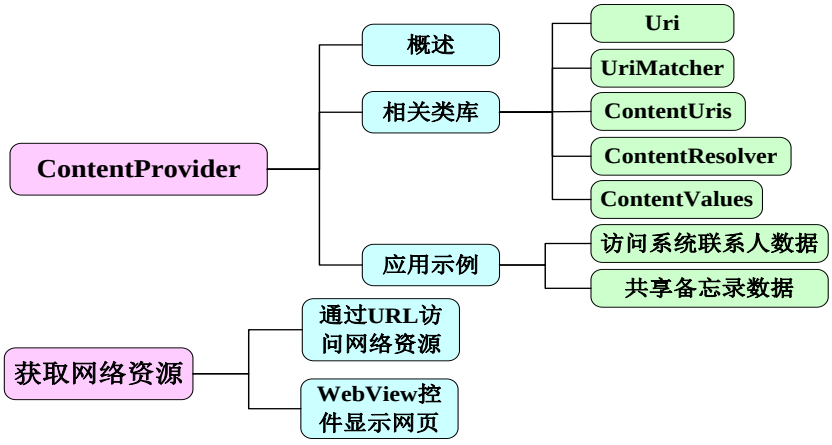
“p136~155所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 8 章 Android 内容提供者(ContentProvider)应用

本章要点

- 统一内容提供者——ContentProvider 类及其应用
- 网络资源的读取

本章知识结构图



本章示例

随着我们手机上应用的增多，往往在不同的应用之间需要共享数据，比如现在有一个短信群发的应用，用户需要选择收件人，一个个手机号码输入当然可以达到目的，但是比较麻烦，并且很少有人会记住所有联系人的号码。这时候就需要获取联系人应用的数据，然后从中选择收件人即可。对于应用之间数据的共享，我们可以在一个应用中直接操作另一个应用所记录的数据，比如上一章中所学的文件、SharedPreferences或数据库等。这不仅需要应用程序提供相应的权限，而且还必须知道应用程序中数据存储的细节，不同应用程序记录数据的方式差别也很大，不利于数据的交换。为此，Android提供了ContentProvider，用统一的方法实现不同应用程序间共享数据。

8.1 ContentProvider 简介

ContentProvider是不同应用程序之间进行数据交换的标准API，为存储和读取数据提供了统一的接口；通过ContentProvider，应用程序可以实现数据共享；Android内置的许多应用都使用ContentProvider向外提供数据，供开发者调用(如视频、音频、图片、通讯录等)，其中最典型的应用就是通讯录。

那么ContentProvider是如何对外提供数据的呢，又是如何实现这一机制的呢？ContentProvider以某种URI的形式对外提供数据，数据以类似数据库中表的方式暴露，允许其他应用访问或修改数据，其他应用程序使用ContentResolver根据URI去访问操作指定的数据。URI是通用资源标识符，即每个ContentProvider都有一个唯一标识的URI，其他应用程序的ContentResolver根据URI就知道具体解析的是哪个ContentProvider，然后调用相应的操作方法，而ContentResolver的方法内部实际上是调用该ContentProvider的对应方法，而ContentProvider方法内部是如何实现的，其他应用程序是不知道具体细节的。只是知道有哪一个方法。这就达到了统一接口的目的。对于不同的数据的存储方式，该方法内部的实现是不同的，而外部访问方法都是一致的。

ContentProvider也是Android四大组件之一，如果要开发自己的ContentProvider必须实现Android系统提供的ContentProvider基类，并且需要在AndroidManifest.xml文件中进行配置。

ContentProvider基类的常用方法简介：

- public **abstract boolean onCreate()**：该方法在 ContentProvider 创建后调用，当其他应用程序第一次访问 ContentProvider 时，ContentProvider 会被创建，并立即调用该方法；
- public **abstract Cursor query**(Uri uri, String[] projection, String selection, String[] selectionArgs, String sortOrder)：根据 Uri 查询符合条件的全部记录，其中 projection 是所需要获取的数据列；
- public **abstract int update**(Uri uri, ContentValues values, String select, String[] selectArgs)：根据 Uri 修改 select 条件所匹配的全部记录；
- public **abstract int delete**(Uri uri, String selection, String[] selectionArgs)：根据 Uri 删除符合条件的全部记录；
- public **abstract Uri insert**(Uri uri, ContentValues values)：根据 Uri 插入 values 对

应的数据，ContentValues 类似于 map，存放的是键值对；

- public **abstract String getType**(Uri uri)：该方法返回当前 Uri 所代表的数据的 MIME 类型。如果该 Uri 对应的数据包含多条记录，则 MIME 类型字符串应该以 vnd.android.curor.dir/开头，如果该 Uri 对应的数据只包含一条记录，则 MIME 类型字符串应该以 vnd.android.cursor.item/开头。

上面几个方法都是抽象方法，开发自己的ContentProvider时，必须重写这些方法，然后在AndroidManifest.xml文件中配置该ContentProvider，为了能让其他应用找到该ContentProvider，ContentProvider 采用了authorities（主机名/域名）对它进行唯一标志，你可以把ContentProvider看作是一个网站，authorities就是它的域名，只需在<application.../>元素内添加以下代码即可。

```
1  <provider android:name=".MyProvider"                          →指定ContentProvider类
2        android:authorities="iet.jxufe.cn.android.provider.myprovider">   →域名
3  </provider>
```

注意：authorities是必备属性，如果没有authorities属性会报错。

一旦某个应用程序通过ContentProvider暴露了自己的数据操作接口，那么不管该应用程序是否启动，其他应用程序都可通过该接口来操作该应用程序的内部数据。

8.2 ContentProvider 操作常用类

上节介绍ContentProvider时涉及几个知识点：Uri、ContentResolver、ContentValues，本节将详细介绍这几个类的作用和用法。

8.2.1 Uri 基础

Uri代表了要操作的数据，Uri主要包含了两部分信息：

- 需要操作的 ContentProvider；
- 对 ContentProvider 中的什么数据进行操作。一个 Uri 组成如图 8-1 所示：

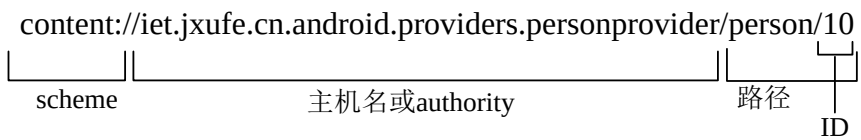


图 8-1　Uri 的组成部分

scheme：ContentProvider（内容提供者）的 scheme 已经由 Android 所规定为：content://；

主机名（或Authority）：用于唯一标志这个ContentProvider，外部调用者可以根据这个标志来找到它。

路径（或资源）：用于确定我们要操作该ContentProvider中的什么数据，一个

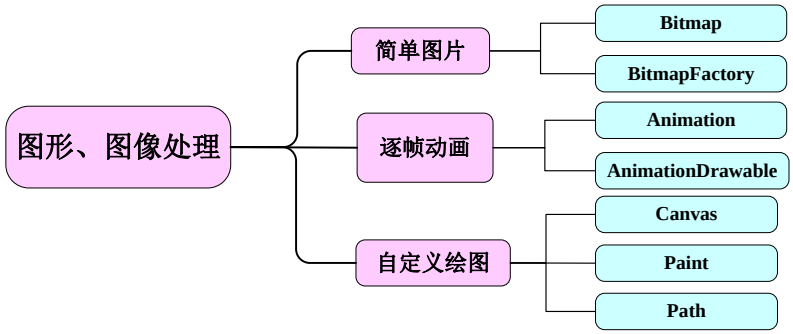
“p159~171所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 9 章　Android 图形图像处理

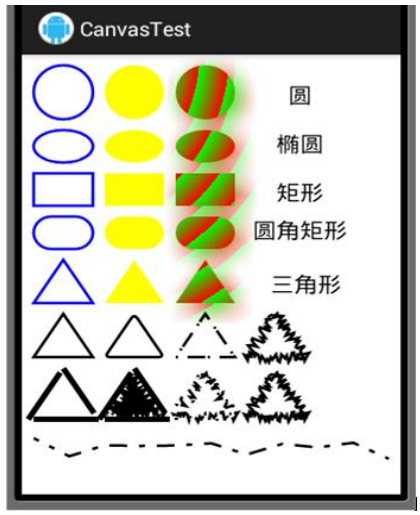
本章要点

- 图形处理基础
- Bitmap 与 BitmapFactory
- 逐帧动画
- 自定义 View 进行绘图
- Canvas 与 Paint
- 使用 Path 绘制路径
- 使用 Shader 进行渲染
- 使用 PathEffect 改变路径效果

本章知识结构图



本章示例



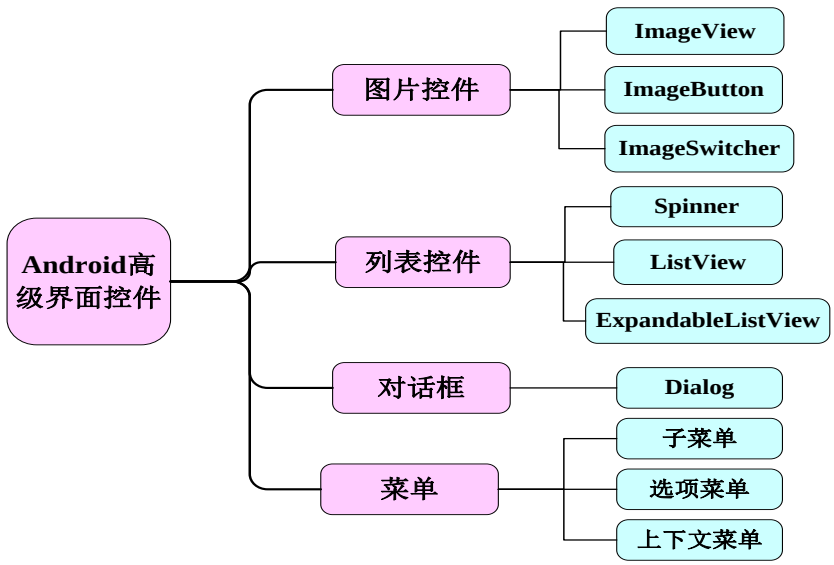
“p173~187所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 10 章 Android 界面设计进阶

本章要点

- ImageView 图片视图
- Spinner 下拉列表
- ListView 列表
- ExpandableListView 扩展下拉列表
- Dialog 对话框
- 菜单

本章知识结构图



本章示例

前面我们学习了Android中一些简单界面控件，以及如何使这些控件按我们的需求排列在界面上，能够设计出一些简单的界面效果。除此之外，通过继承View类，重写里面的方法，我们可以根据我们的需求设计界面控件。然而要想设计出一些界面复杂、功能强大的控件，还是存在一些困难。Android为我们提供了一些常用的、功能强大的高级控件，如图片控件、列表控件、对话框控件以及菜单。本章我们将集中讲解之。

在此基础上，大家有可能设计相对比较复杂的一些界面，还有一些更复杂的界面设计，本书作为一本入门书，就不作过分深入的介绍。读者若有兴趣，可参考《Android编程经典案例剖析》（清华大学出版社，2015年1月版，高成珍、钟元生主编）一书。

10.1 图片控件

一些好的界面设计，少不了图片的运用，Android为我们提供了多种图片控件，最为常用的有ImageView、ImageButton、ImageSwitcher、Gallery等。

10.1.1 ImageView 图片视图

ImageView（图片视图）的作用与TextView类似，TextView用于显示文字，ImageView则用于显示图片，既然是显示图片，那就要设置图片的来源，ImageView中有一个src属性用于指定图片的来源。显示图片还存在另外一个问题，就是当我们的图片比ImageView的区域大的时候如何显示呢？在ImageView中有一个常用并且重要的属性scaleType，用于设置图片的缩放类型。该属性值主要包含以下几个：

fitCenter：保持纵横比缩放图片，直到该图片能完全显示在ImageView中，缩放完成后将该图片放在ImageView的中央。

fitXY：对图片横向、纵向独立缩放，使得该图片完全适应于该ImageView，图片的纵横比可能会改变。

centerCrop：保持纵横比缩放图片，以使得图片能完全覆盖ImageView。

以一个简单的示例演示各种属性值对应的效果，现在假设有一个ImageView的宽和高分别是200和300，而当前的图片的大小是600*600。可计算宽度的缩放比为600/200=3，高度的缩放比为600/300=2。当采用fitCenter时，由于需要保持纵横比，且图片能够完整的在ImageView中显示，所以宽和高都缩放比较大的比例即3倍，缩放后的图片大小为：200*200，显示效果如图9-1所示；当采用centerCrop时，要使得图片能够完全覆盖ImageView，因此，宽和高都缩放比较小的比例即2倍，缩放后的图片大小为：300*300，但是超过ImageView的部分将不会显示，效果如图9-2所示，宽度有部分未显示出来；当采用fitXY时，宽和高按各自的比例进行缩放，缩放后的图片大小为200*300，此时图片已经发生了变形，如图10-3所示。

图 10- 1 fitCenter 效果　图 10- 2 centerCrop 效果　图 10- 3 fitXY 效果

当图片的纵横比与ImageView的纵横比一致时，三种值对应的效果将会完全一样。

10.1.2 ImageButton 图片按钮

ImageButton的作用与Button的作用类似，主要是用于添加单击事件处理。Button类从TextView继承而来，相应的ImageButton从ImageView继承而来，主要区别是，Button按钮上显示的是文字，而ImageButton按钮上显示的是图片，需要注意的是在ImageView、ImageButton上是无法显示文字的，即使在XML文件中为ImageButton添加android:text属性，虽然程序运行时不会报错，但运行结果仍无法显示文字。

如果想在按钮上既显示文字又显示图片，应该怎么办呢？一种方法是直接将图片和文字设计成一张图片，然后将其作为ImageButton的src属性的值，但这种方法不够灵活，当我们需要改变文字或图片时，需重新设计整张图片；另一种方式是直接将图片作为Button的背景，并为Button按钮添加android:text属性，这种情况下，图片和文字是分离的，可以单独进行设置，灵活性较好，但缺点就是图片作为背景时可能会变形，以适应Button的大小。

在ImageButton中，既可以设置background属性也可以设置src属性，这两个属性的值都可以指向一张图片，**那么这两个属性有什么区别呢**？

src属性表示的是图标，background属性表示的是背景。图标是中间的一块区域，而背景是我们所能看到的控件范围。简单来说，一个是前景图(src)，一个是背景图(background），这两个属性最大的区别是：用src属性时是原图显示，不会改变图片的大小；用background属性时，会按照ImageButton的大小来放大或者缩小图片。举例来说如果ImageButton的宽和高是100x100，而原图片的大小是80x80。如果用src属性来引用该图片，则图片会按80x80的大小居中显示在ImageButton上。如果用background属性来引用该图片，则图片会被拉伸成100x100。

在Android中图片的格式除了常规的JPG、PNG、GIF格式外，还可以用XML文件进行定义。下面我们定义一个XML文件，该文件设定在不同的状态下，引用不同的图片。程序运行效果如图10-4、10-5、10-6所示，在界面中包含一个ImageView和两个ImageButton。其中，ImageView和第一个ImageButton都引用了bg.xml文件作为图片源，该文件设置了在按下和未按下两种状态下显示的图片不同，第二个ImageButton则只是引用了blue.png作为图片，然后通过单击事件处理得到改变图片的效果。当我们按下ImageView时，图片并不会发生改变，这是因为ImageView不能处理事件，只是用于显

示图片；按下中间的ImageButton时，图片的颜色发生了变化；单击下面的ImageButton后，按钮图片发生变化，这是通过为ImageButton添加单击事件处理来实现的。

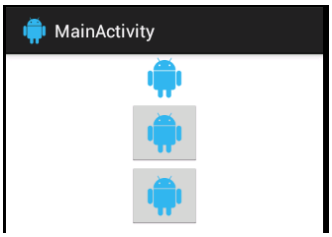


图 10- 4 初始效果

图 10- 5 按下中间按钮的效果

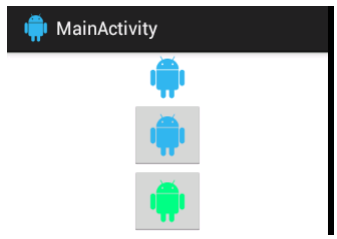


图 10- 6 单击下面按钮的效果

首先，查看bg.xml文件的代码，如下所示。

程序清单：codes\chapter10\ ImageTest\res\drawable\bg.xml

```
1  <selector xmlns:android="http://schemas.android.com/apk/res/android" >
2  <item android:state_pressed="false" android:drawable="@drawable/blue"></item>     →未按下为蓝色
3  <item android:state_pressed="true" android:drawable="@drawable/green"></item>     →按下为绿色
4  </selector>
```

界面布局代码如下。

程序清单：codes\ chapter 10\ ImageTest\res\layout\activity_main.xml

```
1    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2          xmlns:tools="http://schemas.android.com/tools"
3          android:layout_width="match_parent"                    →宽度填充父容器
4          android:gravity="center_horizontal"                    →线性布局中的内容水平居中
5          android:orientation="vertical"                         →垂直线性布局
6          android:layout_height="match_parent" >                 →高度为填充父容器
7         <ImageView
8              android:layout_width="wrap_content"
9              android:layout_height="wrap_content"
10             android:src="@drawable/bg"/>                       →图片源为bg.xml
11        <ImageButton
12             android:layout_width="wrap_content"
13             android:layout_height="wrap_content"
14             android:src="@drawable/bg"/>                       →图片源为bg.xml
15        <ImageButton
16             android:id="@+id/myImg"                            →为ImageButton添加ID属性
17             android:layout_width="wrap_content"
18             android:layout_height="wrap_content"
19             android:src="@drawable/blue"/>                     →图片源为blue.png
20   </LinearLayout>
```

接下来在MainActivity中为下面的ImageButton添加事件处理。

程序清单：codes\ chapter10\ImageTest\src\iet\jxufe\cn\android\MainActivity.java

```
1    private boolean flag=true;                                   →定义一个标志变量
2          private ImageButton myBtn;
3          public void onCreate(Bundle savedInstanceState) {
4              super.onCreate(savedInstanceState);
5              setContentView(R.layout.activity_main);          →设置界面布局文件
```

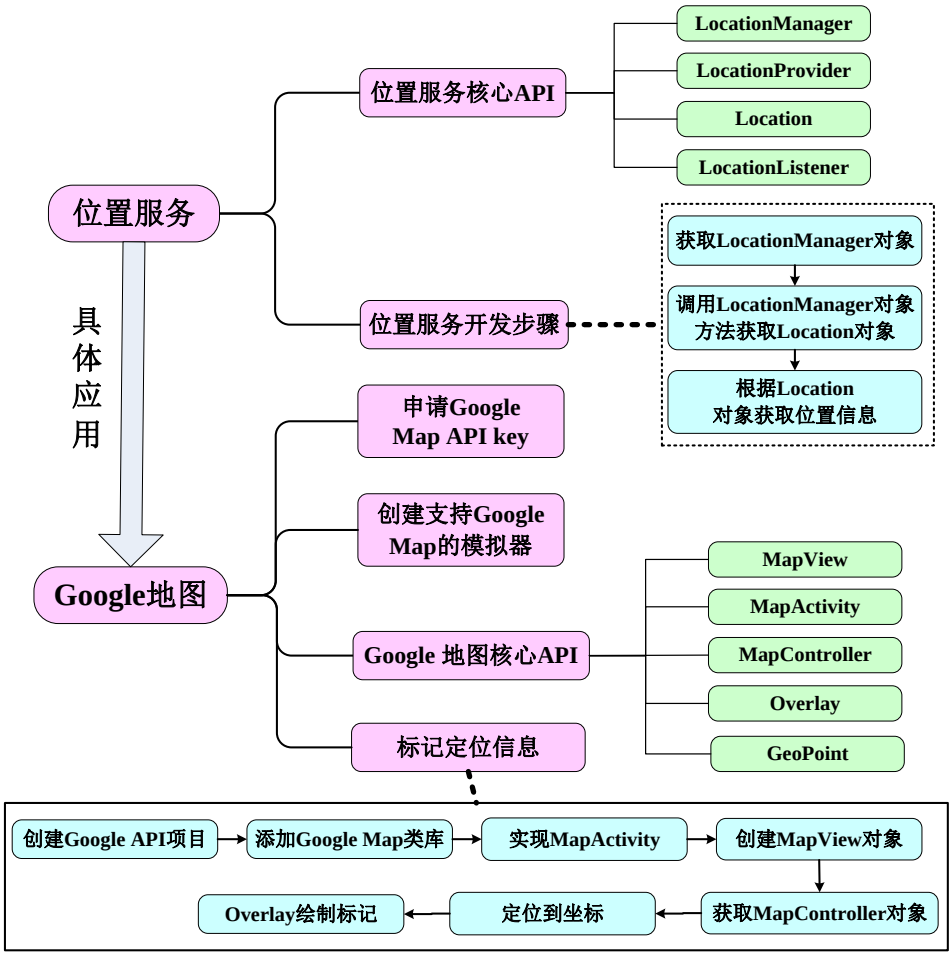
“p192~218所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 11 章 Android GPS 位置服务与地图编程

本章要点

- Android 中支持位置服务的核心 API
- 通过 LocationListener 监听位置信息
- Android 中临近警告
- Google Map Key 申请
- Google 插件下载
- Google 地图核心 API
- 在 Google 地图上标记位置

本章知识结构图



手机相对于个人电脑来说，除了携带方便,最重要的特点就是具有可移动性,如果我们能够时刻获取到手机的地理位置，进而开发出与位置相关的应用,将会给用户带来更好的体验，提供更贴切的服务。最典型的应用就是根据位置，查找周边的建筑物以及交通情况。GPS等位置应用是现在大多数智能移动终端设备的标准配置。

Android平台支持提供位置服务的API，可以利用GPS（Global Positioning System，全球定位系统）和Network Location Provider(网络位置提供器)来获得用户的位置。GPS相对来说更精确，但它只能在户外工作，很费电，并且不能像用户期望的那样立即就能返回位置信息。而Android的网络位置提供器是使用手机发射塔和Wi-Fi信号来判断用户位置，在室内室外都能工作、响应速度快，并且更加省电。如果想在应用程序中获得用户的位置，你可以同时使用GPS和网络位置提供器，或者其中一种。通过定位服务可以获取当前设备的地理位置，应用程序可以定时请求更新设备当前的地理定位信息，从而达到实时监测的功能。例如以经纬度和半径划定一个区域，一旦设备出入该区域时，发出提醒信息。

监测位置变化仅仅是其中的一部分，Google还提供了相应的API来管理Google地图数据，通过MapView来显示地图、MapController来操作地图，要使用这些工具，首先我们要获取Map API Key。一旦获取了Key，就可以放大或缩小地图、查找任何一个位置、甚至可以在地图上添加自己的标记。

本章将详细讲解与位置服务相关的API，获取定位信息，然后结合Google地图开发出比较实用的应用。

11.1 GPS 位置服务编程

位置服务（Location-Based Services，LBS）又称定位服务或基于位置的服务，融合了GPS定位、移动通信、导航等多种技术，提供了与空间位置相关的综合应用服务。

11.1.1 支持位置服务的核心 API

Android为支持位置服务，提供了android.location包，该包中包含了与位置信息密切相关的类和接口，主要有：**LocationManager**、**LocationProvider**、**Location**、**LocationListener**。

LocationManager（定位管理者）类是访问Android系统位置服务的入口，所有定位相关的服务、对象都将由该类的对象来产生。和其他服务一样，程序不能直接创建LocationManager对象，而是通过Context的getSystemService()方法来获取，代码如下。

```
1    LocationManager locMg=getSystemService(Context.LOCATION_SERVICE);
```

一旦得到了LocationManager对象，即可调用LocationManager类的方法获取定位相关的服务和对象，例如获取最佳定位提供者、实现临近警报功能等，该类的常用方法如下。

- public String getBestProvider (Criteria criteria, boolean enabledOnly)：根据指定条件返回最优的 LocationProvider；criteria 表示过滤条件，enabledOnly 表示是否

要求处于启用状态。

- public Location getLastKnownLocation (String provider)：根据 LocationProvider 获取最近一次已知的 Location，provider 表示提供上次位置的 LocationProvider 名称。
- public LocationProvider getProvider (String name)：根据名称返回 LocationProvider。
- public List<String> getProviders (boolean enabledOnly)：获取所有可用的 LocationProvider。
- public void addProximityAlert (double latitude, double longitude, float radius, long expiration, PendingIntent intent)：添加一个临近警告，即不断监听手机的位置，当手机与固定点的距离小于指定范围时，系统将会触发相应事件，进行处理。latitude 指定中心点的经度，longitude 指定中心点的纬度，radius 指定一个半径长度，expiration 指定经过多少毫秒后该临近警告就会过期失效，-1 指定永不过期，intent 指定临近该固定点时触发该 intent 对应的组件。
- public void requestLocationUpdates (String provider, long minTime, float minDistance, PendingIntent intent)：通过指定的 LocationProvider 周期性地获取定位信息，并通过 intent 启动相应的组件，进行事件处理，provider 表示 LocationProvider 的名称，mimTime 表示每次更新的时间间隔，单位为毫秒，minDistance 表示更新的最近位置，单位为米，intent 每次更新时启动的组件。
- public void requestLocationUpdates (String provider, long minTime, float minDistance, LocationListener listener)：通过指定的 LocationProvider 周期性地获取定位信息，并触发 listener 所对应的触发器。

LocationProvider（定位提供者）类是对定位组件的抽象表示，用来提供定位信息，能够周期性的报告设备的地理位置，Android中支持多种LocationProvider，它们以不同的技术提供设备的当前位置，区别在于定位的精度、速度和成本等方面。常用的LocationProvider主要有以下两种。

network：由LocationManager.NETWORK_PROVIDER常量表示，代表通过网络获取定位信息的Location Provider对象；

gps：由LocationManager.GPS_PROVIDER常量表示，代表通过GPS获取定位信息的LocationProvider对象。

GPS相对来说精度更高，但它只能在户外工作，很费电，并且不能像用户期望的那样立即就能返回位置信息，而网络位置提供器使用手机发射塔或Wi-Fi信号来判断用户位置，在室内室外都能工作、响应速度快，并且更加省电。

LocationProvider类的常用方法如下。

- int getAccuracy()：返回该 LocationProvider 的精度；
- String getName()：返回该 LocationProvider 的名称；
- boolean hasMonetaryCost()：返回该 LocationProvider 是收费的还是免费的；
- boolean supportsAltitude()：判断该 LocationProvider 是否支持高度信息；
- boolean supportsBearing()：判断该 LocationProvider 是否支持方向信息；

- boolean supportsSpeed()：判断该 LocationProvider 是否支持速度信息；

Location类就是代表位置信息的抽象类，通过Location可获取定位信息的精度、高度、方向、纬度、经度、速度以及该位置的LocationProvider等信息。

LocationListener接口用于监听定位信息的监听器，必须在定位管理器中注册该对象，这样在位置发生变化的时候就会触发相应的方法进行事件处理，该监听器包含的方法如下。

- public abstract void onLocationChanged (Location location)：位置发生改变时回调该方法；
- public abstract void onProviderDisabled (String provider)：Provider 禁用时回调该方法；
- public abstract void onProviderEnabled (String provider)：Provider 启用时回调该方法；
- public abstract void onStatusChanged (String provider, int status, Bundle extras)：当 Provider 状态发生变化时回调该方法；

11.1.2 简单位置服务应用

前面我们学习了Android位置服务的核心API，那么它们之间是如何协作来完成，位置服务功能呢？使用他们来获取位置信息的通用步骤为：

1）获取系统的LocationManager对象；

2）使用Locationmanager，通过指定LocationProvider来获取定位信息，定位信息由Location来对象来表示；

3）从Location对象中获取定位信息。

下面我们以一个简单的例子来演示如何获取位置信息，并进行相应的判断，程序运行后，在DDMS视图下的Location Controls面板中，模拟位置的变化，发送经纬度，如图11-1所示，当位置发生变化后，程序能够及时捕捉到该变化，并显示当前的位置信息，如图11-2所示，得到位置信息后，与南昌的坐标位置相比较，如果在在这个范围内，则显示你进入南昌，如果离开了这个范围，则显示你离开了南昌，效果如图11-3和图11-4所示。

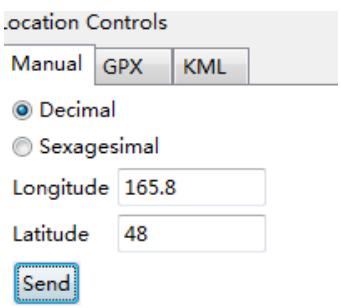


图 11- 1 模拟位置信息变化

图 11- 2 获取当前位置信息

图 11-3 进入南昌效果图

图 11-4 离开南昌效果图

程序界面布局相对简单，只有两个文本显示框，用于显示经纬度信息，在此不再列出，获取位置信息，添加临近警告的代码如下。

程序清单：codes\chapter11\LocationService\src\iet\jxufe\cn\android\MainActivity.java

```
1  public class MainActivity extends Activity {
2      private LocationManager locMg;
3      private Location location;
4      private TextView tv;                                          →显示经纬度信息
5      public void onCreate(Bundle savedInstanceState) {             的文本显示框
6          super.onCreate(savedInstanceState);
7          setContentView(R.layout.activity_main);
8          locMg = (LocationManager)                                 →获取位置服务
   getSystemService(Context.LOCATION_ SERVICE);
9          location = locMg.getLastKnownLocation(LocationManager.GPS_ →得到上次的位置
   PROVIDER);
10         tv=(TextView)findViewById(R.id.myLoc);
11         showInfo(location);                                       →显示位置信息
12         locMg.requestLocationUpdates(LocationManager.GPS_PROVIDER, 3000, 8,
13                 new LocationListener() {                          →注册监听器，每
14                 public void onStatusChanged(String provider, int status,  个3秒获取位置信
15                         Bundle extras) {                          息
16                 }                                                 →当LocationProv
17                 public void onProviderEnabled(String provider) {  ider状态发生变化
18                 showInfo(locMg.getLastKnownLocation(provider));   时，触发该方法
19                 }                                                 →LocationProvider
20                 public void onProviderDisabled(String provider) { 启用时调用该方法
21                     showInfo(null);
22                 }                                                 →LocationProvider
```

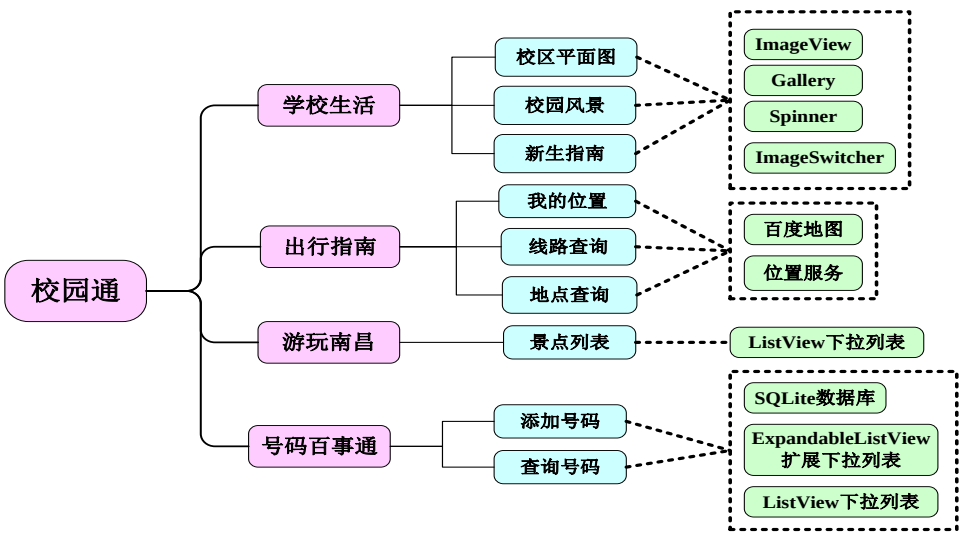
“p224~235所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

第 12 章 Android 编程综合案例

本章要点

- 综合案例----校园通需求概述
- 综合案例程序结构
- 综合案例功能模块分析
- 界面设计
- 关键代码解释
- 注意事项

本章知识结构图



本章示例

通过前面章节的学习，我们已经掌握了android开发的基础知识，本章我们将通过一个综合案例将前面所学的知识串联起来，共同开发一个实用的应用程序。本章以江西财经大学导航为例，带着大家从零开始开发一个“校园通”应用程序，该应用主要是为在校学生服务，方便学生迅速查找信息，包括学校生活、出行指南、游玩南昌、号码百事通四个模块。读者可以根据自己所在学校或社区，模仿本用例开发自己的校园通或者社区通。

本案例所涉及的知识包括：基本界面控件的使用，如TextView、Button、ImageView等、高级界面控件的使用，如：Spinner、Gallery、ListView、ExpandableListView、菜单等、Activity之间的跳转与数据的传递、事件处理，如单击事件、触摸事件、选择事件等、SQLite数据库的使用、位置服务与百度地图。

通过本章的学习，读者将对这些知识有更深入的了解，并且能够自主开发一些小的应用。

12.1 “校园通”概述

“校园通”应用软件主要是为江西财经大学的新生、老生、老师以及对江西财经大学感兴趣的人服务的，提供一个信息服务平台，方便他们迅速查找相应信息，主要包括学校生活、出行信息、游玩南昌、号码百事通四个模块。

学校生活主要介绍学校的基本情况，包括校区平面图、校园风景、新生指南等，对于即将入学的新生以及校外人士了解财大情况非常有帮助；

出行信息包括我的位置、公交线路查询、位置查询等；

游玩南昌包括南昌主要景点介绍；号码百事通包括号码的查询和添加。

“校园通”应用程序的功能模块图如图12-1所示。

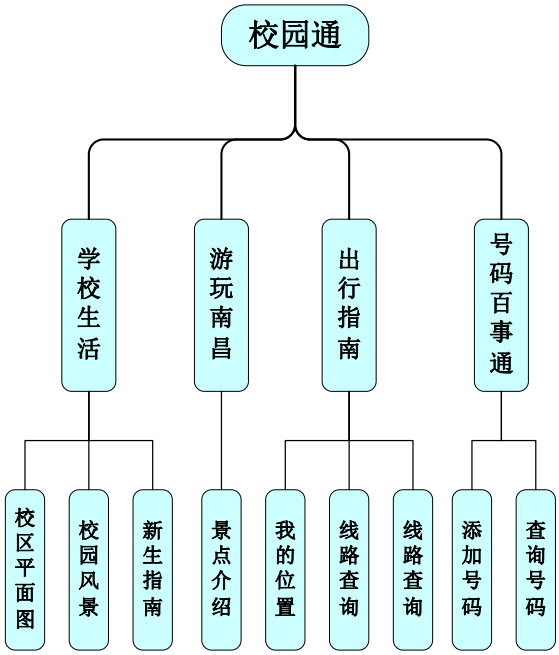


图 12-1 校园通功能结构图

12.2 “校园通”应用程序结构

“校园通”应用程序涉及的功能和界面较多，为了方便对代码的管理，为每个功能模块单独建立一个包，将功能相关的Activity放在同一包下，“校园通”应用程序的程序结构如图12-2所示。

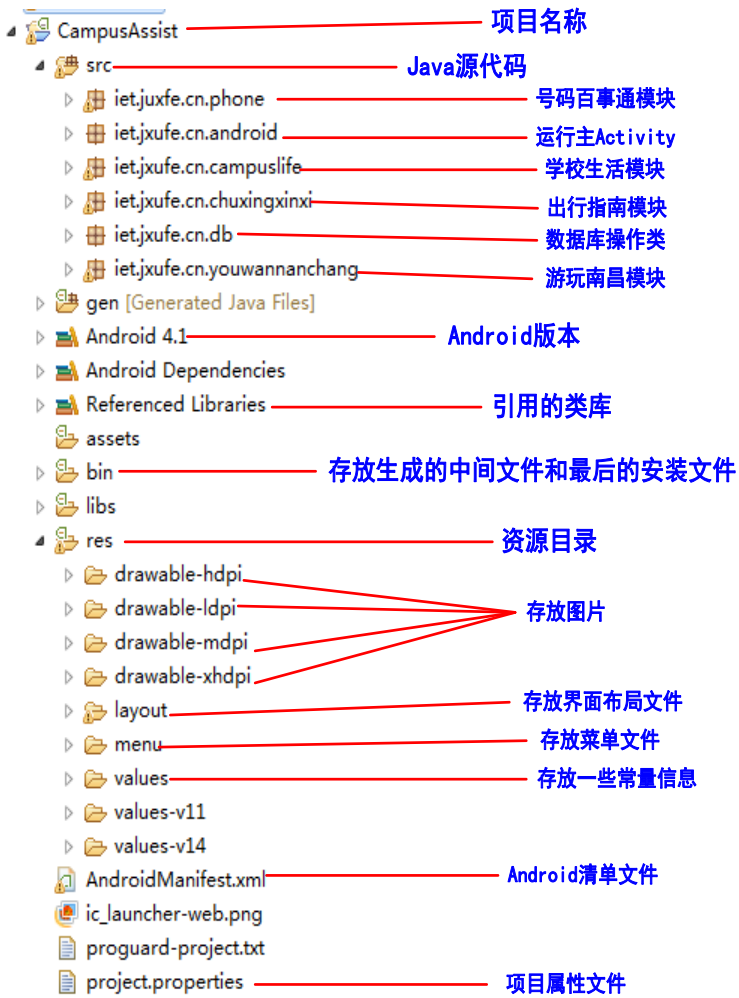


图 12-2 校园通应用程序结构图

12.3 “校园通”应用程序功能模块

“校园通”应用程序，主要包含四个模块，程序运行的主界面及界面分析如图12-3所示。单击学校生活、出行指南、游玩南昌、号码百事通等按钮后，能够跳转到相应的功能模块。

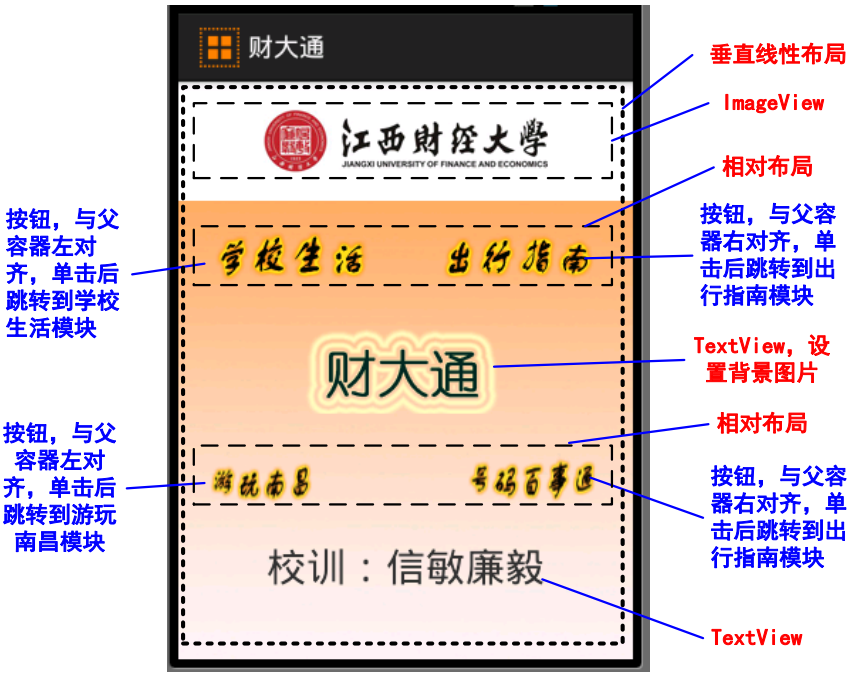


图 12-3 校园通应用程序主界面分析

主界面的布局文件关键代码如下。

程序清单：codes\ chapter12\ CampusAssist\res\layout\activity_main.xml

```
1  <LinearLayout...>                                          →总体是垂直线性布局
2      <ImageView.../>                                        →空白ImageView
3      <ImageView                                             →显示财大标记的ImageView
4          android:src="@drawable/logo".../>
5      <ImageView.../>                                        →空白ImageView
6      <RelativeLayout...>                                    →相对布局
7          <Button android:layout_alignParentLeft="true".../> →学校生活按钮,与父容器左对齐
8          <Button android:layout_alignParentRight="true".../> →出行指南按钮,与父容器右对齐
9      </RelativeLayout>
10     <TextView.../>                                         →显示中间的财大通文字
11     <RelativeLayout...>                                    →相对布局
12         <Button android:layout_alignParentLeft="true".../> →游玩南昌按钮,与父容器左对齐
13         <Button android:layout_alignParentRight="true"/>  →号码百事通按钮与父容器右对齐
14     </RelativeLayout>
15     <TextView.../>                                         →显示校训文字
16 </LinearLayout>
```

单击各个按钮后能够跳转到相应的功能模块，事件处理的关键代码如下。

程序清单：codes\ chapter12\ CampusAssist\src\iet\jxufe\cn\android\MainActivity.java

```
1  public class MainActivity extends Activity {
2      private Button phoneAssist,trafficAssist,campusLife,scenery;        →声明Button类型变量
3      public void onCreate(Bundle savedInstanceState) {
4          super.onCreate(savedInstanceState);
5          setContentView(R.layout.activity_main);                          →设置界面布局
6          phoneAssist=(Button)findViewById(R.id.phoneAssist);              →根据ID找到控件
7          trafficAssist=(Button)findViewById(R.id.trafficAssist);
8          campusLife=(Button)findViewById(R.id.campusLife);
9          scenery=(Button)findViewById(R.id.scenery);
10         MyOnClickListener myOnClickListener=new                          →创建监听器对象
               MyOnClickListener();
11         phoneAssist.setOnClickListener(myOnClickListener);               →为按钮添加事件监听器
12         trafficAssist.setOnClickListener(myOnClickListener);
13         campusLife.setOnClickListener(myOnClickListener);
14         scenery.setOnClickListener(myOnClickListener);
15     }
16     public class MyOnClickListener implements OnClickListener{          →内部类实现事件监听器
17         Intent intent=null;
18         public void onClick(View v) {                                    →单击事件处理方法
19           switch (v.getId()) {                                           →判断事件源
20               case R.id.phoneAssist:
21                   intent=new Intent                                      →跳转到号码百事通
                         (MainActivity.this,PhoneListActivity.class);
22                   break;
23               case R.id.trafficAssist:
24                   intent=new Intent                                      →跳转到出行指南
                       (MainActivity.this,ChuxingxinxiActivity.class);
25                   break;
26               case R.id.campusLife:
27                   intent =new Intent                                     →跳转到校园生活
                       (MainActivity.this,CampusLifeActivity.class);
28                   break;
29               case R.id.scenery:
30                   intent=new Intent                                      →跳转到游玩南昌
                       (MainActivity.this,SceneryActivity.class);
31                   break;
32               default:break;
33               }
34               startActivity(intent);
35           }
36     }
37 }
```

在主界面中，有四个按钮都需要进行单击事件处理，因此可创建一个内部类来实现事件监听器，所有的按钮注册到同一个事件监听器上，单击事件发生后，通过判断事件源从而进行不同的处理。下面分别对各个模块的功能进行详细分析与介绍。

12.3.1 学校生活模块

学校生活主要介绍学校的基本情况，包括校区平面图、校园风景以及新生指南三部分。程序结构图以及各个Activity之间的跳转关系如图12-4所示。

图 12-4 学校生活模块程序结构

学校生活模块运行主界面及分析如图12-5所示。

图 12-5 学校生活主界面分析

界面布局相对简单，总体使用垂直线性布局，设置背景图片，对齐方式为垂直居中并右对齐android:gravity="center_vertical|right"，并设置右边距为20dp，即paddingRight = 20 dp。按钮的事件处理与前面主界面上的按钮事件处理类似，在此不给出相应代码。可查看codes\chapter12\CampusAssist\src\iet\jxufe\cn\android\ CampusLifeActivity.java。

单击校区平面图按钮后，跳转到 CampusBuildActivity，界面运行效果如图 12-6、12-7 所示。该界面中包含一个 Spinner 下拉列表、一个 ImageView 以及一个返回按钮，选择 Spinner 中的某一项后能在 ImageView 中显示对应的图片，由于图片比较大，因此为图片添加了触摸事件，能够拖动以查看图片其他部分。

图 12- 6 校区平面图运行界面

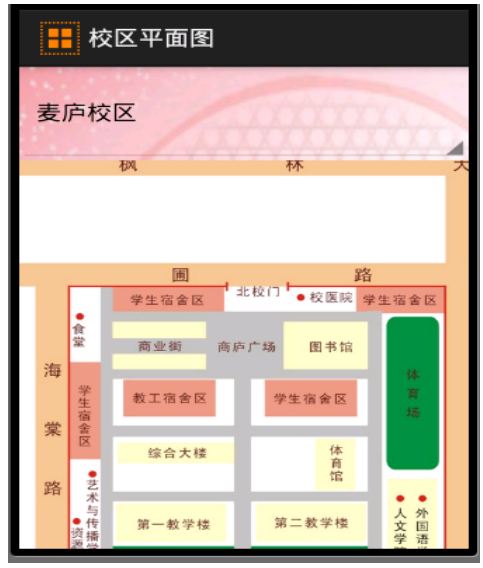


图 12- 7 麦庐校区显示效果

界面整体采用垂直线性布局，相对比较简单，在此不列出代码，在此界面中，存在三种事件处理，一种是下拉列表的选择事件，一种是图片的触摸事件还有一种是返回按钮的单击事件。关键代码如下所示。

下拉列表选择事件关键代码如下。

程序清单：codes\chapter12\ CampusAssist\src\iet\jxufe\cn\ android\ CampusBuildActivity.java

```
1   int [] imageIds=new int[]{R.drawable.jiaotong,R.drawable.jiaoqiaoxiaoqu,
2         R.drawable.mailuxiaoqu,R.drawable.fenglinxiaoqu};          →图片ID数组
3   String[] xiaoqu=new String[]{"交通示意图","蛟桥校区","麦庐校区","          →列表内容
        枫林校区"};
4   protected void onCreate(Bundle savedInstanceState) {
5         super.onCreate(savedInstanceState);
6         setContentView(R.layout.campus_build);
7         mySpinner = (Spinner)findViewById(R.id.spinner);          →根据ID找到下拉列表
8         myImage=(ImageView)findViewById(R.id.myImage);          →根据ID找到ImageView
9         ArrayAdapter<String> adapter=new ArrayAdapter<String>(this,
10              android.R.layout.simple_dropdown_item_1line,xiaoqu);          →设置样式和内容
11        mySpinner.setAdapter(adapter);          →为Spinner设置Adapter
12        mySpinner.setOnItemSelectedListener(new          →选中事件监听器
          OnItemSelectedListener() {
13          public void onItemSelected(AdapterView<?> arg0, View arg1,int position, long id) {
14              myImage.setImageResource(imageIds[position]);          →根据选择显示图片
15          }
16        public void onNothingSelected(AdapterView<?> arg0) {
```

```
17                  myImage.setImageResource(imageIds[0]);                    →默认显示第一张图片
18              }
19      });
```

图片的触摸事件处理代码如下。

```
1       myImage.setOnTouchListener(new OnTouchListener() {              →匿名内部类实现触摸事件
2               public boolean onTouch(View v, MotionEvent event) {     监听器
3                       float curX,curY;                                →触摸事件发生的坐标
4                       switch (event.getAction()) {
5                       case MotionEvent.ACTION_DOWN:
6                               mx=event.getX();
7                               my=event.getY();
8                               break;
9                       case MotionEvent.ACTION_MOVE:
10                              curX=event.getX();
11                              curY=event.getY();
12                              myImage.scrollBy((int)(mx-curX), (int)(my-curY));
13                              mx=curX;
14                              my=curY;
15                              break;
16                      case MotionEvent.ACTION_UP:
17                              curX=event.getX();
18                              curY=event.getY();
19                              myImage.scrollBy((int)(mx-curX), (int)(my-curY));
20                              break;
21                      default:
22                              break;
23                      }
24                      return true;
25              }
26      });
```

触摸事件的原理是：记录按下时的坐标以及移动后的坐标，根据这两个坐标的距离来移动整张图片。在触摸事件中存在三种状态：按下、移动、松开，因此需对触摸事件的状态进行判断，然后再做具体的操作。

单击校园风景按钮后，跳转到CampusSceneryActivity，界面运行效果如图12-8所示。在此界面中包含三种控件：Gallery、ImageSwitcher以及按钮，整体采用垂直线性布局，布局中对齐方式为水平居中。ImageSwitcher图片选择器，主要用于显示图片，相对于ImageView而言，它在图片切换时能添加一些动态效果。

Gallery是画廊，是存放图片的列表，选择某一张图片是时，该图片能够突出显示，而其他为选择的图片则以半透明的形式显示，开发者可自由设置未选中图片的透明度以及图片间的间距等。由于Gallery没选中一张图片时，会单独创建一个ImageView对象，内存消耗比较大，因此逐渐被淘汰了，不推荐使用，可用HorizontalScrollView或PageView代替。在此只是为了显示效果，对内存要求不高，所以选择Gallery。

图 12-8 校园风景运行效果图

该界面中单击Gallery中的某一张图片时，ImageSwitcher中的图片就会相应的进行变化。关键代码如下。

codes\ chapter12\CampusAssist\src\iet\jxufe\cn\android\CampusSceneryActivity.java

```
1   switcher.setFactory(new ViewFactory(){                          →为Switcher创建
2       public View makeView(){                                     图片，并设置效果
3           ImageView imageView = new ImageView(CampusSceneryActivity.this);
4           imageView.setScaleType(ImageView.ScaleType.FIT_CENTER);
5           imageView.setLayoutParams(new ImageSwitcher.LayoutParams(
6               LayoutParams.WRAP_CONTENT, LayoutParams.WRAP_CONTENT));
7           return imageView;
8       }
    });
9   switcher.setInAnimation(AnimationUtils.loadAnimation(this,android.R      →设置图片切换的
    .anim.fade_in));                                                        动画：淡入淡出
10  switcher.setOutAnimation(AnimationUtils.loadAnimation(this,android.R.anim.fade_out));
11  BaseAdapter adapter = new BaseAdapter(){                    →下拉列表对应的内容信息
12      public int getCount(){→获取下拉列表的选项数
13          return Integer.MAX_VALUE;                                   →设置最大值，从
14      }                                                               而循环显示图片
15      public Object getItem(int position){
16          return position;                                            →返回选项所对应
17      }                                                               的位置
18      public long getItemId(int position){
19          return position;
```

```
        }
        public View getView(int position, View convertView, ViewGroup parent){
                ImageView imageView = new ImageView(CampusSceneryActivity.this);
                imageView.setImageResource(images[position % images.length]);
                imageView.setScaleType(ImageView.ScaleType.FIT_XY);
                imageView.setLayoutParams(new Gallery.LayoutParams(75, 100));
                TypedArray typedArray = obtainStyledAttributes(R.styleable.Gallery);
                imageView.setBackgroundResource(typedArray.getResourceId(
                                R.styleable.Gallery_android_galleryItemBackground, 0));
                return imageView;
        }
};
gallery.setAdapter(adapter);
gallery.setOnItemSelectedListener(new OnItemSelectedListener() {
public void onItemSelected(AdapterView<?> arg0, View arg1,int position, long id) {
                switcher.setImageResource(images[position%images.length]);        →选择事件处理
        }
        public void onNothingSelected(AdapterView<?> arg0) {
        }
});
```

单击新生指南按钮后，跳转到FreshAssistActivity，界面运行效果如图12-9所示。该界面采用垂直线性布局，包含八个按钮，居中显示，单击某一按钮后跳转到DetailInfoActivity，显示详细信息，如图12-10所示。

图 12-9 新生指南主界面

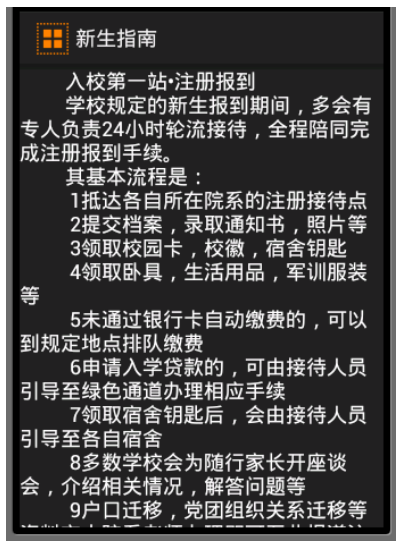


图 12-10 新生指南详细信息

新生指南主界面中包含8个按钮，每个按钮都需要添加事件处理，一个个添加比较麻烦，此处采用数组存储每个按钮的ID，然后循环遍历数组，取出ID，并根据ID找到对应的按钮，并为其注册事件监听器。事件处理后显示的信息虽然不同，但是结构一

致，因此，此处采取数组存储数据，将需要显示的内容存储到Intent中，从而达到动态改变的效果，而不用为每部分单独建立一个Activity显示内容，大大减少了Activity的数量，关键代码如下。

程序清单：codes\ chapter 12\ CampusAssist\src\iet\jxufe\cn\android\ FreshAssistActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.fresh_assist);
    int[] btnIds = new int[] { R.id.woshi, R.id.xuezhang, R.id.zhengli,R.id.dida,
            R.id.jiejiao, R.id.qinlian, R.id.shenghuo,R.id.ruxiao };    →定义按钮对应的ID数组
    Button[] btns = new Button[btnIds.length];                          →创建对应长度的按钮数组
    myOnClickListener myListener = new myOnClickListener();             →创建事件监听器对象
        for (int i = 0; i < btns.length; i++) {
            btns[i] = (Button) findViewById(btnIds[i]);                 →遍历数组根据ID得到按钮
            btns[i].setOnClickListener(myListener);                     →为每个按钮注册事件监听
        }                                                               器
    }
```

注意：上面代码中的btnIds定义必须放在setContentView(R.layout.fresh_assist);之后，因为只有加载了fresh_assist.xml文件之后，才会有对应的按钮ID。

单击事件监听器的实现类关键代码如下。

```
private class myOnClickListener implements OnClickListener {
    Intent intent = new Intent(FreshAssistActivity.this,DetailInfoActivity.class);
    public void onClick(View v) {
        switch (v.getId()) {
        case R.id.zhengli:
            intent.putExtra("info", info[0]);
            break;
        case R.id.dida:
            intent.putExtra("info", info[1]);
            break;
        ……                                                  →其他匹配项，在此不再列出
        default:
            break;
        }
        startActivity(intent);
    }
```

代码中Info是一个字符串数组，用于存放需要传递的字符串信息。对于不同的按钮，传递的数据不同，但接收数据的Activity是一致的，所以在switch语句外面创建Intent对象。

12.3.2 出行指南模块

出行指南主要包括获取当前的位置信息、查找公交路线信息，以及搜索一些关键地点的位置。程序结构图以及各个Activity之间的跳转关系如图12-11所示。

图 12- 11 出行指南模块程序结构

出行指南模块运行主界面如图12-12所示。单击线路查询按钮后，跳转到GongjiaoluxianActivity，调用百度地图API，在界面中输入某一公交路线，会在地图上显示出该公交路线的站点信息，如图12-13所示，显示南昌的232路公交路线，单击缩小和放大可以缩放地图；单击我的位置按钮后，调用百度地图API，并能够在地图上用一个图片标记当前位置，如图12-14所示；单击位置查询后，调用百度地图，并在地图上标记出与之相关的位置信息，如图12-15所示，标记的是南昌与财大相关的位置信息。

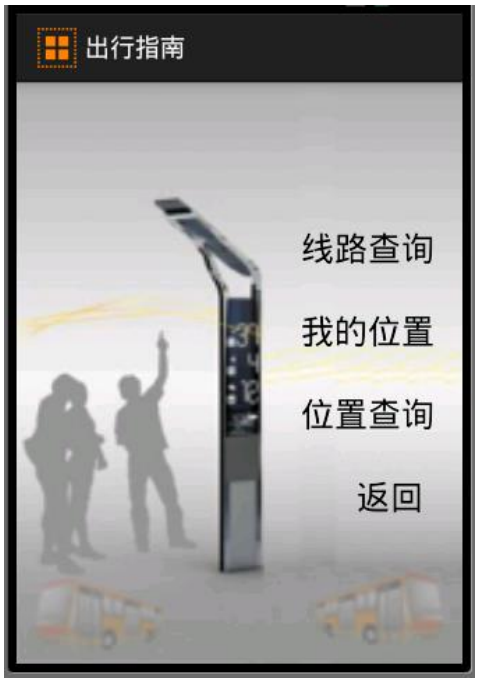


图 12- 12 出行指南模块程序结构图

图 12- 13 公交路线查询结果

图 12- 14 我的位置显示图

图 12- 15 百度地图搜索财大的结果

以上功能模块中都涉及百度地图，先对其进行简要介绍。首先在网上下载百度地图的相关API，并申请使用API的Key。百度地图网址：http://dev.baidu.com/wiki/imap/index.php，进入网页后选择android，如图12-16所示。

图 12- 16 百度地图首页截图

选择Android平台后，可下载相关jar包、技术文档，申请使用API的Key，查看开发

流程等，如图12-17所示。

图 12- 17 android 版百度地图

单击Key申请进入百度地图API Key申请页面。如图12-18所示。

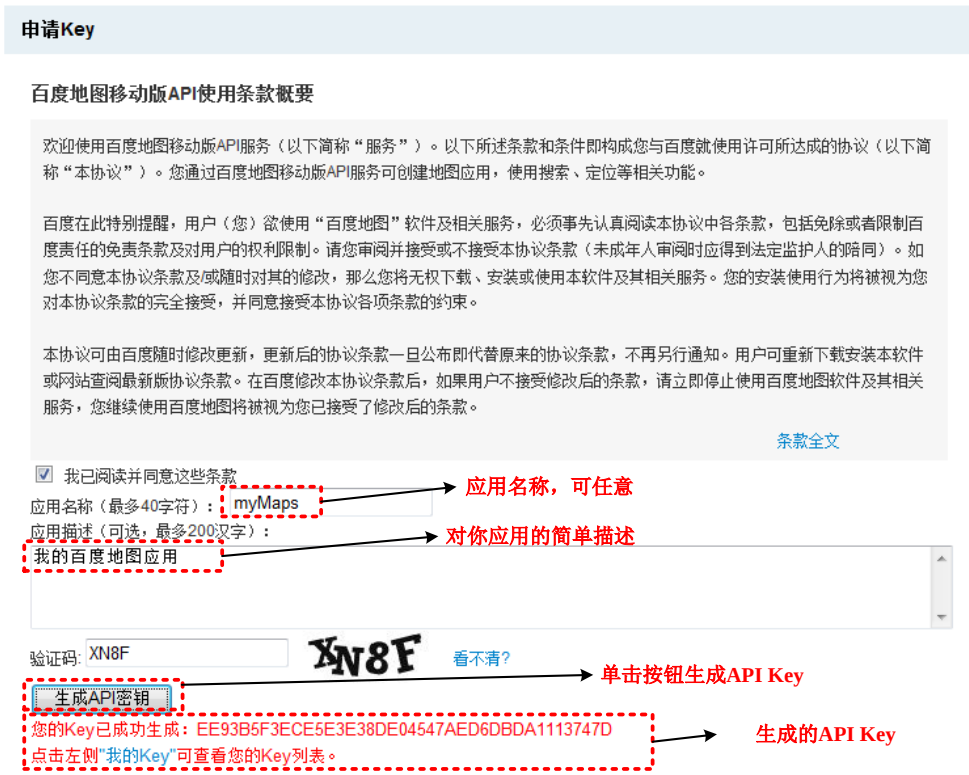


图 12- 18 申请百度地图 API 的 Key

注意：获取API密钥时，前提是你已经登录，所以若没有百度账号，需注册一个账号，然后再进行申请。生成的API Key需要进行保存，在后面开发百度地图相关应用中需要用到。

有了百度地图的API Key和相关Jar包，我们就可以开发自己的应用了，开发步骤如下：

(1) 在项目中添加相关的Jar包，将下载的jar包中的baidumapapi.jar放在项目中的libs目录下，然后在libs目录下，建立一个armeabi文件夹，然后将libBMapApiEngine_v1_3_3.so复制到该工程目录下；

(2) 由于要调用百度地图的相关数据，因此需要添加相应的权限，那么究竟需要添加哪些权限？我们可以通过查看下载的百度地图的示例文件，从它的AndroidManifest.xml中进行拷贝即可，或者运行时根据提示信息一个个进行添加；

(3) 在布局文件中添加地图控件；

```
1   <com.baidu.mapapi.MapView                    →百度地图提供的地图控件，完整的包名+类名
2         android:id="@+id/bmapView"
3         android:layout_width="fill_parent"
4         android:layout_height="fill_parent"
5         android:clickable="true" />
```

这样我们的准备工作就完成了，下面就需要针对具体要实现的功能调用相应的API。先简要介绍下，百度地图中几个比较核心的类库。类的名称与Google 地图的相似。

MapActivity：该类是用于显示地图的Activity类，是一个抽象类，任何想要显示MapView的Activity都需要派生自MapActivity，并且在onCreate()中，都要创建一个MapView实例。

MapView：用于显示地图的View控件。它派生自ViewGroup，它必须和MapActivity配合使用，而且只能被MapActivity创建，这是因为MapView需要通过后台的线程来连接网络或文件系统，这些线程要由MapActivity来管理。当MapView获取的焦点时，它将捕捉按键和触摸手势，自动地平移和缩放地图，还可以在地图上绘制许多Overlay类型标记。

BMapManager: 地图引擎管理类，用于初始化、开启和停止地图。

MapController：用于控制地图的移动、缩放等的工具类。

Overlay：是一个可显示在地图之上的可绘制的对象，常用于绘制标记。如果需要在地图上标注一些图标文字等信息，就需要使用Overlay。添加一个overlay时，从这个基类派生出一个子类，创建一个实例，然后把它加入到一个列表中。这个列表通过调用MapView.getOverlays()得到。

GeoPoint：表示一个地理坐标点，存放经度和纬度，以微度的整数形式存储。（1微度=10-6度）

MkSearch：用于位置检索、周边检索、范围检索、公交检索、驾乘检索、步行检索。

开发百度地图应用的一些基本步骤如下。

程序清单：codes\chapter12\ CampusAssist\src\iet\jxufe\cn\android\ FreshAssistActivity.java

```
1   public class GongjiaoluxianActivity extends MapActivity {
2       private MapView mapView;                                    →MapView用于显示地图
3       private BMapManager bMapManager;                            →地图管理类
4       private MapController mc;                                   →地图控制类
5       private MKSearch mkSearch;                                  →用于位置检索、周边检索、范围检索、公交检索、步行检索等
6       private String keyString = "
    "EE93B5F3ECE5E3E38DE04547AED6DBDA1113747D";                     →申请的API Key(字符串中的Key根据各人申请的代码修改)
7       private EditText bus,city;                                  →城市和线路信息
8       private Button search;                                      →搜索按钮
9       public void onCreate(Bundle savedInstanceState) {
10          super.onCreate(savedInstanceState);
11          setContentView(R.layout.gongjiaoluxian);                →布局文件中只有一个MapView控件
12          bus=(EditText)findViewById(R.id.bus);                   →获取要查询的线路
13          search=(Button)findViewById(R.id.search);
14          mapView = (MapView) findViewById(R.id.bmapView);
15          city=(EditText)findViewById(R.id.city);                 →获取查询的城市
16          bMapManager = new BMapManager(this);                    →创建地图管理类的对象
17          bMapManager.init(keyString, new MKGeneralListener() {   →初始化地图管理类
18              public void onGetPermissionState(int arg0) {
19                  if (result == 300) {                            →返回授权验证错误,300表示验证失败
20                      Toast.makeText(GongjiaoluxianActivity.this, "验证不通过，请重新输入！
    ",
21                              Toast.LENGTH_SHORT).show();
22                  }
23              }
24              public void onGetNetworkState(int result) {         →返回网络错误
25              }
26          });
27          initMapActivity(bMapManager);                           →初始化MapActivity
28          mapView.setBuiltInZoomControls(true);                   →设置地图可放大、缩小
29          mc=mapView.getController();                             →获取地图控制对象
30          mc.setZoom(15);                                         →设置缩放级别为15
31  }
```

在上述代码中，bMapManager对象初始化时，需要传递两个参数，第一个参数为申请的授权验证码，即我们申请的API Key，第二个参数为注册回调事件，用于获取验证错误信息或网络错误信息。通过上述这段代码，我们才可以在模拟器中显示地图信息

了，并可以进行缩放、移动，默认是以北京天安门为中心显示信息。需要注意的是，还必须要在androidManifest.xml文件中添加相应的权限信息。

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

下面我们实现查询公交路线功能，为搜索按钮添加事件处理。主要是调用MKSearch类来实现公交检索，关键代码如下。

```
search.setOnClickListener(new OnClickListener() {                    →为搜索按钮添加事件监听器
    public void onClick(View v) {
        mkSearch = new MKSearch();                                   →创建MKSearch对象
        mkSearch.init(bMapManager,new MyMKSearchListener());         →初始化MKSearch对象
        mkSearch.poiSearchInCity(city.getText().toString().trim(),   →传入两个参数，城市和公交路线
bus.getText().toString().trim());
    }
});
```

MkSearch类对象初始化时传入两个参数，地图管理对象以及MKSearchListener监听器，该监听器用于返回poi搜索（**位置、关键点搜索**）、公交搜索、驾乘路线和步行路线结果。poiSearchInCity()方法用于城市poi搜索，将会自动调用MKSearchListener中的onGetPoiResult()，在该方法中会返回所有的poi信息，然后我们获取公交路线的poi信息，再调用MKSearch的busLineSearch()根据poi信息来搜索公交的详细信息，并绘制路线标记，关键代码如下所示。

```
public class MyMKSearchListener implements MKSearchListener{
    public void onGetWalkingRouteResult(MKWalkingRouteResult result,
            int arg1) {                                              →返回步行路线搜索结果
    }
    public void onGetTransitRouteResult(MKTransitRouteResult result,
            int arg1) {                                              →返回公交搜索结果
    }
public void onGetSuggestionResult(MKSuggestionResult arg0, int arg1) {   →返回搜索结果
    }
public void onGetPoiResult(MKPoiResult result, int type, int iError) {   →返回poi搜索结果
        if (result == null || iError != 0) {                         result - 搜索结果iError - 错误号，0表示正确返回
        Toast.makeText(GongjiaoluxianActivity.this,
                "对不起，没有相应结果",
1000).show();
            return;
        }
        MKPoiInfo mkPoiInfo=null;                                    →搜索的Poi信息
        int mkPoiNum=result.getNumPois();                            →得到的Poi信息个数
        for(int i=0;i<mkPoiNum;i++){                                 →循环获取所有的Poi
```

```
19                              mkPoiInfo=result.getPoi(i);                          信息
20                              if(mkPoiInfo.ePoiType==2){                           →2表示公交路线
21                                  break;
22                              }
23                          }
24   mkSearch.busLineSearch(city.getText().toString().trim(), mkPoiInfo.uid);      → 搜索公交详细信息
25              }
26              public void onGetDrivingRouteResult(MKDrivingRouteResult result,int arg1) {
27              }                                                                  → 返回驾乘路线搜索结果
28              public void onGetBusDetailResult(MKBusLineResult result,           → 返回公交车详情信息搜索结果
    int iError) {
29                  if (result == null || iError != 0) {
30                      Toast.makeText(GongjiaoluxianActivity.this,
31  "对不起，没有相应结果",1000).show();
32                      return;
33                  }
34              RouteOverlay routeOverlay=new RouteOverlay(GongjiaoluxianActivity.this, mapView);
35                  routeOverlay.setData(result.getBusRoute());                    →为标记设置数据
36                  mapView.getOverlays().clear();                                 →清空原有标记
37  mapView.getOverlays().add(routeOverlay);                                       →添加路线标记
38                  mapView.invalidate();                                          →刷新地图
39          mapView.getController().animateTo(result.getBusRoute().getStart());   →定位到起点
40              }
41              public void onGetAddrResult(MKAddrInfo arg0, int arg1) {           →返回地址信息搜索结果
42              }
43          }
```

主要流程是：查询城市的所有poi信息，然后对poi信息进行遍历，获取公交路线poi信息，最后根据搜索公交路线的详细信息。整个流程如图12-19所示。

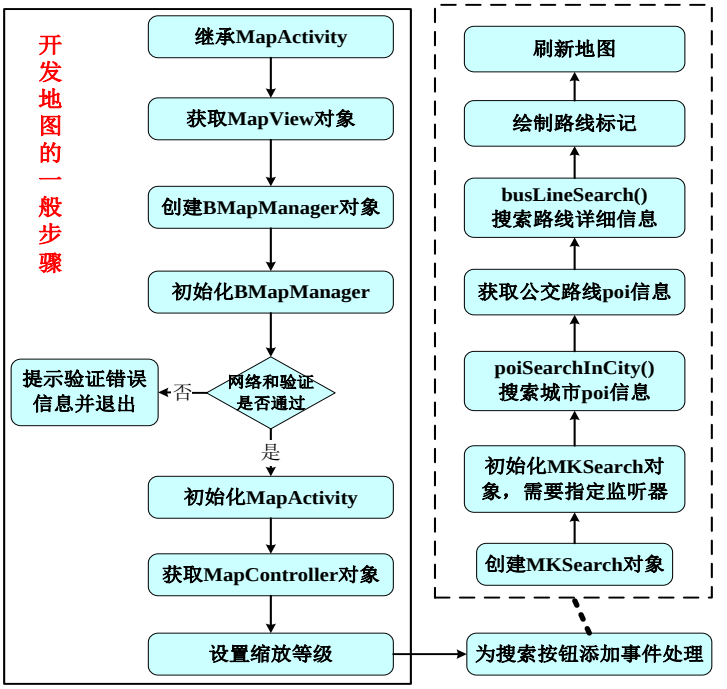


图 12- 19 百度地图开发的一般流程

由于地图比较消耗资源与流量，因此建议离开界面时应停止地图；恢复时重新启动地图；退出时销毁地图。主要是通过Activity的状态来对地图进行控制。关键代码如下。

```
1    protected void onResume() {                          →恢复时重新启动地图
2            super.onResume();
3            if (bMapManager != null) {
4                bMapManager.start();
5            }
6        }
7        protected void onPause() {                       →暂停时，停止地图
8            super.onPause();
9            if (bMapManager != null) {
10   bMapManager.stop();
11           }
12       }
13       protected void onDestroy() {                     →退出时，销毁地图
14           super.onDestroy();
15           if (bMapManager != null) {
16               bMapManager.destroy();
17               bMapManager = null;
18           }
19       }
```

位置搜索原理与公交路线类似，只是获取信息和处理方式有所差别，关键代码如下。

```
1    public void onGetPoiResult(MKPoiResult result, int type, int iError) {    → 返回poi搜索结果 result
```

```
2           if (result == null || iError != 0) {                    ——搜索结果iError 为错误号，0表示正确返回
3   Toast.makeText(GuanjiandianActivity.this, "对不起，没有相应结果",
    Toast.LENGTH_LONG).show();
4               return;
5   }
6           mapView.getOverlays().clear();                          →清除地图上已有的覆盖物
7           PoiOverlay poioverlay = new PoiOverlay (GuanjiandianAct → PoiOverlay是baidu map api提供的用于显示POI的Overlay
    ivity.this, mapView);
8           poioverlay.setData(result.getAllPoi());                 →设置搜索到的POI数据
9           mapView.getOverlays().add(poioverlay);                  →在地图上显示PoiOverlay
10          if(result.getNumPois() > 0) {                           （将搜索到的兴趣点标注在地图上）
11              MKPoiInfo poiInfo = result.getPoi(0);               →设置其中一个搜索结果所
12              mc.setCenter(poiInfo.pt);                           在地理坐标为地图的中心
13          }
14  }
```

“我的位置”功能则结合了位置服务和百度地图功能，首先根据位置服务定位自己的坐标，然后在百度地图上对应的坐标处绘制标记，关键代码如下。

```
1   locMg = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
    →获取位置服务
2   updateLocation(location);                                       →默认的位置标记
3   locMg.requestLocationUpdates(LocationManager.GPS_PROVIDER, 3000, 10,
4           new android.location.LocationListener() {               →位置监听器
5           public void onStatusChanged(String provider, int status,Bundle extras) {
6           }
7           public void onProviderEnabled(String provider) {
8               updateLocation(locMg.getLastKnownLocation(provider));
9           }
10          public void onProviderDisabled(String provider) {
11              updateLocation(null);
12          }
13          public void onLocationChanged(Location location) {
14              updateLocation(location);                           →位置发生变化时，
15          }                                                       更新地图上的显示
16  });
```

显示地图上当前位置的标记方法代码如下。

```
1   public void updateLocation(Location location) {
2           if (location == null) {
3           geoPoint = new GeoPoint((int) (28.73 * 1E6), (int) (115.81 * 1E6));  →默认位置
4               } else {
```

“p256~275所涉内容，请参见清华大学出版社正式出版《Android编程》（钟元生 高成珍主编），2015年版”

App开发案例教程

一、最佳的快速入门教程：《Android编程》， 钟元生 高成珍 编著

详细讲解Android编程基础知识，包括Android界面设计、事件处理、四大组件以及数据存储等。着重分析Android的基本语法、基本技术与基本应用，强调可运行案例的实现，帮助读者理论联系实践，寓教于练、寓教于用，非常适合大学课堂教学。

提供包括手把手教学视频、教学大纲、课件、源代码、实验指导、网络测试、MOOC教学和网络辅导社区等全方位教学资源，同时将提供Android实习就业方面的网络增值服务。

二、最适合的练习案例：《Android编程经典案例解析》， 高成珍 钟元生 等 编著

详细分析17个典型的Android实用案例，如简易计算器、天气预报、音乐播放器等。这些案例稍加修改即可直接用于自己的项目中。

图解分析、代码展示、通俗易懂；案例贴近生活、实用，实践性强；内容由浅入深，知识融会贯通，提升综合运用能力；涉及知识面广，可参考性高。

还总结了常见Android错误与程序调试方法、编程测试题库。既可作为Android入门者的自测和提高练习指南，又可作为移动开发者的参考帮手。

三、最全而巧的实战案例：《App开发案例教程》，钟元生 曹权 编著

围绕一个完整的移动应用APP的开发，系统地介绍了Android客户端、IOS客户端、PC端以及Web服务器端的开发技术和开发流程。读者可边学习边搭建系统，完整地开发一个移动应用软件。

通过实践、体验和思考，从而体会移动开发的细节，尽快达到熟练水平。本书既可作为入门者开发一个完整移动应用的指南，又可作为高年级学生软件项目实践课程或毕业设计的参考。

清华大学出版社数字出版网站
WQBook 书文启泉
www.wqbook.com

定价：34.50元