第5章 使用资源文件

查看 Android 项目文件夹，可以看到在第二级目录下有一个叫 res 的文件夹，该文件夹中存放的是 Android 应用中使用到的资源，包括字符串资源、颜色资源、数组资源、菜单资源等，在应用程序中可以直接对这些资源定义进行应用。

除了 res 目录下可以存放资源以外，assets 目录也可用于存放资源。一般情况下，应用无法直接访问的原生态资源将会被放到 assets 目录下，程序需要通过 AssetManager 以二进制流的形式来读取资源。而 res 目录下的资源，Android SDK 会在编译该应用时，自动在 R.java 文件中为这些资源创建索引，程序可以直接通过 res 清单类进行访问。

前面介绍的实例中大多都是直接将字符串值直接写在布局文件中或者 Java 程序代码中，其实这是一个不好的编程习惯，现在通过本章关于 Android 资源文件的学习后，应该在编程时将 Android 应用会使用到的资源放在 res 文件目录下并通过资源文件来管理，然后在布局文件或 Java 代码中采用 res 清单类进行访问。

5.1 资源的类型和存储方式

Android 资源包含了保存在 assets 目录下无法直接访问的原生资源以及保存在 res 目录下通过 R 清单类来访问的这两种类型的资源。

在 Android 项目文件夹的 res 目录下，不同的子目录存放着不同类型的应用资源，表 5.1 显示了 Android 不同资源在 res 目录下的存储方式。

表 5.1 res 目录下各资源存储方式

目　　录	存放的资源
/res/anim/	存放定义补间动画的 XML 文件
/res/color/	存放定义不同状态下颜色列表的 XML 文件
/res/drawable/	该目录下存放各种位图文件(如 *.png、*.9.png、*.jpg、*.gif)等，除此之外还可编译成如下各种 Drawable 对象的 XML 文件： BitmapDrawable NinePatchDrawable 对象 StateListDrawable 对象 ShapeDrawable 对象 AnimationDrawable 对象 Drawable 的其他各种子类的对象

续表

目　　录	存放的资源
/res/layout/	存放各种用户界面的布局文件
/res/menu/	存放为应用程序定义各种菜单的资源，包括选项菜单、子菜单、上下文菜单资源
/res/raw/	该目录下存放任意类型的原生资源。在 Java 代码中通过调用 Resource 对象的 openRawResource(int id)方法获取该资源的二进制输入流。 实际上，如果应用程序使用原生资源，推荐把这些原生资源保存到/assrts 目录下，然后在应用程序中使用 AssetManager 来访问这些资源。
/res/values/	存放各种简单的 XML 文件。这些简单值包括字符串值、整数值、颜色值、数组等。 字符串、整数值、颜色值、数组等各种值都是存放在该目录下，而且这些资源文件的根目录都是< resources.../>元素，若为该< resource.../>元素添加不同的子元素，则代表不同的资源，例如： string/integer/bool 子元素——代表添加一个字符串值/整数值/boolean 值 color 子元素——代表添加一个颜色值 array 子元素或 string-array、int-array 子元素——代表添加一个数组 style 子元素——代表添加一个样式 dimen——代表添加一个尺寸 由于各种简单值都可以定义在/res/values/目录下的资源文件中，如果在同一份资源文件中定义各种值，势必增加程序维护的难度。为此，Android 建议使用不同的文件来存放不同类型的值： arrays. xml——定义数组资源 colors. xml——定义颜色值资源 dimens. xml——定义尺寸值资源 strings. xml——定义字符串资源 styles. xml——定义样式资源
/res/xml/	任意的原生 XML 文件。这些 XML 文件可在 Java 代码中使用 Resources. getXML() 访问

将应用程序中的各种资源分别保存在 Android 项目文件夹下的 res 目录下后，就可以在 Java 代码中使用这些资源，也可以在 XML 文件中使用这些资源。

5.2 通过字体设置功能使用字符串、颜色、尺寸资源

本节所使用的实例需要用到的字符串资源、颜色资源、尺寸资源所对应的 XML 文件都将被保存在/res/values 目录下，它们默认的文件名以及在 R 类中对应的内部类如表 5.2 所示。

表 5.2　资源文件及其在 R 类中对应的内部类

资 源 类 型	资源文件默认文件名	对应 R 类中的内部类名称
字符串资源	/res/values/strings. xml	R. string
颜色资源	/res/values/colors. xml	R. color
尺寸资源	/res/values/dimens. xml	R. dimen

本节通过一个具有字体设置功能的实例来向读者介绍如何使用字符串、颜色、尺寸资源。首先，字符串资源文件 strings.xml 内容如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">ResourceDemo</string>
    <string name="text">我是文字</string>
    <string name="text_change">我是改变后的文字</string>
    <string name="btn_change_text">改变文本显示文字的内容</string>
    <string name="btn_change_color">改变文本显示文字的颜色</string>
    <string name="btn_change_size">改变文本显示文字的大小</string>
</resources>
```

从上面的字符串资源文件的 XML 代码可以看出，字符串资源文件的根元素是<resources>，该元素中每个<string>子元素定义一个字符串常量，其中<string>标签的 name 属性指定该常量的名称，<string>元素的开始标签和结束标签之间的内容定义了字符串值。如上面代码中的<string name="app_name">ResourceDemo</string>语句。

接着，我们一起来看看颜色资源文件 colors.xml 中的内容，如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#000000</color><!--黑色 -->
    <color name="red">#FF0000</color><!--红色 -->
    <color name="white">#FFFFFF</color><!--白色 -->
    <color name="yellow">#FFFF00</color><!--黄色 -->
</resources>
```

从上面的颜色资源文件的 XML 代码可以看出，颜色资源文件的根元素是<resources>，该元素中的每个<color>子元素定义了一个颜色值常量，其中<color>标签的 name 属性指定该颜色的名称，<color>元素的开始标签与结束标签之间的内容定义了颜色值。如上面代码中的<color name="red">#FF0000</color>语句。

下面再来看看尺寸资源文件 dimens.xml 中的内容，如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="text_size_16">16.0sp</dimen>
    <dimen name="text_size_20">16.0sp</dimen>
</resources>
```

从上面的尺寸资源文件的 XML 代码可以看出，尺寸资源文件的根元素是<resources>，该元素中的每个<dimen>子元素定义了一个颜色值常量，其中<dimen>标签的 name 属性指定该颜色的名称，<dimen>元素的开始标签与结束标签之间的内容定义了颜色值。如上面代码中的<dimen name="text_size_16">16.0sp</dimen>语句。

至此，我们已将实例中会使用到三种资源定义好了，包括字符串资源、颜色资源、尺寸资源。现在开始编写实例的布局的 XML 代码，如下：

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

```
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:gravity = "center"
    android:orientation = "vertical" >

    < TextView
        android:id = "@ + id/txt_show"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:paddingBottom = "5dp"
        android:text = "@string/text"
        android:textColor = "@color/white"
        android:textSize = "@dimen/text_size_16" />

    < Button
        android:id = "@ + id/btn_change_text"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "@string/btn_change_text" />

    < Button
        android:id = "@ + id/btn_change_color"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "@string/btn_change_color" />

    < Button
        android:id = "@ + id/btn_change_size"
        android:layout_width = "wrap_content"
        android:layout_height = "wrap_content"
        android:text = "@string/btn_change_size" />

</LinearLayout >
```

代码文件：codes\05\5.2\ResourceDemo\res\layout\activity_main.xml

以上布局文件比较简单，从上至下放置了一个 TextView，用于显示文字，三个 Button 按钮分别代表不用的功能，按下第一个 Button 按钮将会改变 TextView 所显示的文字，按下第二个 Button 按钮将会改变 TextView 显示文字的颜色，按下第三个 Button 按钮将会使 TextView 所显示的文字字体变大。

从以上布局文件中可以看出，在 XML 文件中使用布局文件遵循的语法格式为：@< resource_type >/< resource_name >，例如，使用字符串资源时用到@string/text，说明如下：

- < resource_type >——R 类中代表不同资源类型的内部类；
- < resource_name >——指定资源的名称，在这里是 XML 资源元素中由 android:name 属性所指定的名称，其实该资源名称也可以是无后缀的文件名，如放在 drawable-hdpi、drawable-ldpi、drawable-mdpi 这三个文件夹下的图片资源。

其实上面的语法格式并不是完整的语法格式，完整的语法格式为：@[< package_name >:]< resource_type >/< resource_name >，package_name 指定了资源类所在应用的包。如果所引用的资源和当前资源位于同一个包下，则< package_name >可以省略。

在学习了如何在 XML 文件里使用资源文件后，接下来介绍如何在 Java 代码中使用字

符串资源、颜色资源、尺寸资源。本实例的 MainActivity 的 Java 代码如下：

```
package cn.edu.hstc.resourcedemo.activity;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.TextView;

public class MainActivity extends Activity {
    private TextView show;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        show = (TextView) findViewById(R.id.txt_show);
        initButton();
    }

    private void initButton() {
        findViewById(R.id.btn_change_text).setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                //改变文本显示内容,将字符串资源文件对应的字符串赋予文本显示控件
                show.setText(R.string.text_change);
            }
        });

        findViewById(R.id.btn_change_color).setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                //使用颜色资源,改变文本字体颜色
                show.setTextColor(getResources().getColor(R.color.red));
            }
        });

        findViewById(R.id.btn_change_size).setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                //使用尺寸资源,改变文本字体大小
                show.setTextSize(getResources().getDimensionPixelOffset(R.dimen.text_size_20));
            }
        });
    }
}
```

代码文件：codes\05\5.2\ResourceDemo\cn\edu\hstc\activity\MainActivity.java

从上面的 Java 代码可以看出，在 Java 代码中按如下语法格式使用资源：[< package_name >]. R. < resource_type >. < resource_name >，与在 XML 中使用资源的语法格式类似，如果所使用的资源和当前资源位于同一个包下，则< package_name >可以省略。

将上面的程序部署在 Android 模拟器上，运行效果如图 5.1 所示。

单击第一个按钮，可以看到如图 5.2 的运行效果。

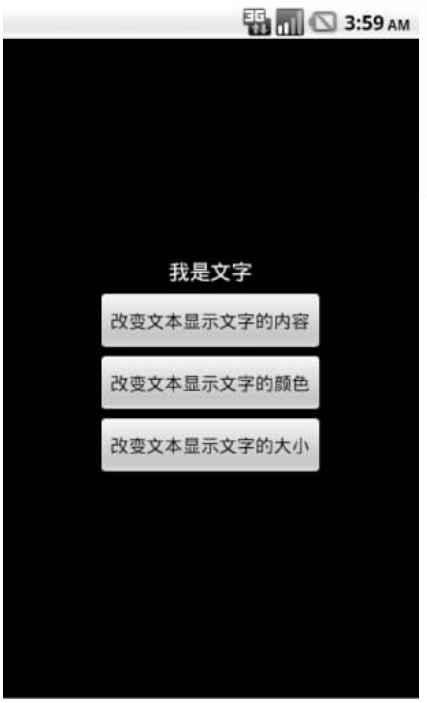

图 5.1 在 XML 布局文件中使用各种资源

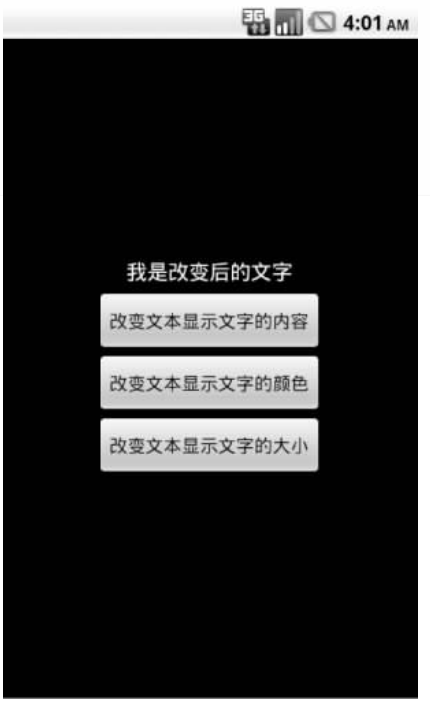

图 5.2 在 Java 代码中使用字符串资源改变文字内容

单击第二个按钮,可以看到如图 5.3 的运行效果。

单击第三个按钮,可以看到如图 5.4 的运行效果。

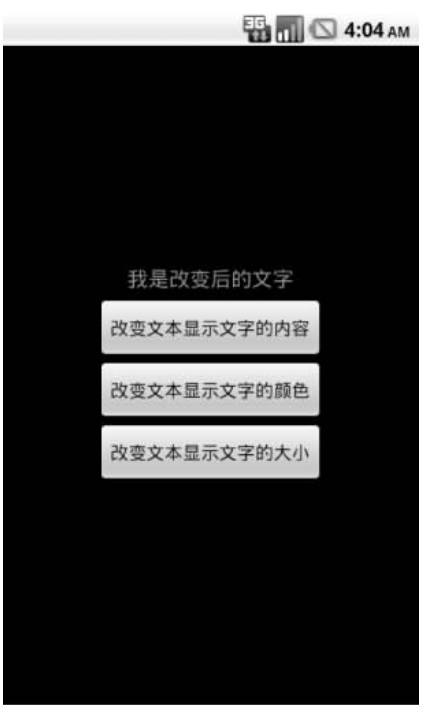

图 5.3 在 Java 代码中使用颜色资源改变文字颜色

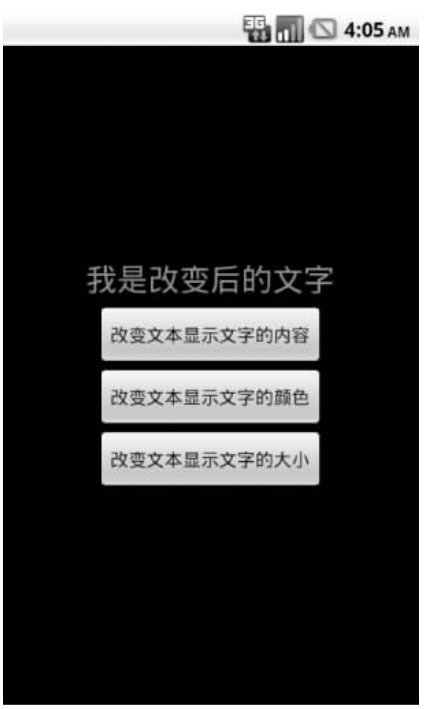

图 5.4 在 Java 代码中使用尺寸资源改变文字大小

5.3 使用图片资源

图片资源是最简单的一种 Drawable 资源，只需要把 *.png、*.jpg、*.gif 等格式的图片放入/res/drawable-xxx 目录下，Android SDK 就会在编译应用时自动加载该图片，并在 R 资源清单类中生成该资源的索引。

这里需要注意的是，Android 不允许图片资源的文件名中出现大写字母，且不能以数字开头，否则 Android SDK 无法为该图片在 R 类中生成资源索引。

使用图片资源也非常简单，在 Java 代码中可以使用[< package >.]R.drawable.< file_name >的语法格式来访问该图片资源，而在 XML 代码中可以使用@[< package_name >:]drawable/file_name 的语法格式来访问图片资源。

为了在程序中获得实际的 Drawable 对象，Resources 提供了 Drawable getDrawable(int id)方法，该方法可根据 Drawable 资源在 R 清单类中的 ID 来获取实际的 Drawable 对象。

本节通过一个实例来演示如何在 XML 代码中使用图片资源以及在 Java 代码中使用图片资源。实例的布局文件比较简单，只是在界面底部放置一个 Button 按钮，然后在该按钮的上方放置一个图片显示控件，该 Button 按钮的功能是将图片容器里的图片替换成另一张图片。界面布局代码如下：

```
<?xml version = "1.0" encoding = "utf - 8"?>
< RelativeLayout xmlns:android = "http://schemas.android.com/apk/res/android"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent" >

    < Button
        android:id = "@ + id/button"
        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
        android:text = "换一张图片"
        android:layout_alignParentBottom = "true" />

    < ImageView
        android:id = "@ + id/image"
        android:layout_width = "fill_parent"
        android:layout_height = "fill_parent"
        android:layout_above = "@id/button"
        android:background = "@drawable/a"
        android:scaleType = "fitXY" />

</RelativeLayout >
```

代码文件：codes\05\5.3\ImageResourceDemo\res\layout\activity_main.xml

上面的程序演示了在 XML 代码中如何使用图片资源。

接下来看看在 Java 代码中如何使用图片资源，代码如下：

```
package cn.edu.hstc.imageresourcedemo.activity;

import com.example.imageresourcedemo.R;
```

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;

public class MainActivity extends Activity {
    ImageView image = null;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        image = (ImageView) findViewById(R.id.image);
        findViewById(R.id.button).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //在 Java 代码中使用图片资源
                image.setImageDrawable(getResources().getDrawable(R.drawable.b));
            }
        });
    }
}
```

代码文件：codes\05\5.3\ImageResourceDemo\cn\edu\hstc\activity\MainActivity.java

上面的程序演示了如何在 Java 代码中使用图片资源。

将程序部署在模拟器上，运行效果如图 5.5 所示。

单击布局最下面的“换一张图片”按钮，会看到界面上的图片被替换成另外一张，运行效果如图 5.6 所示。

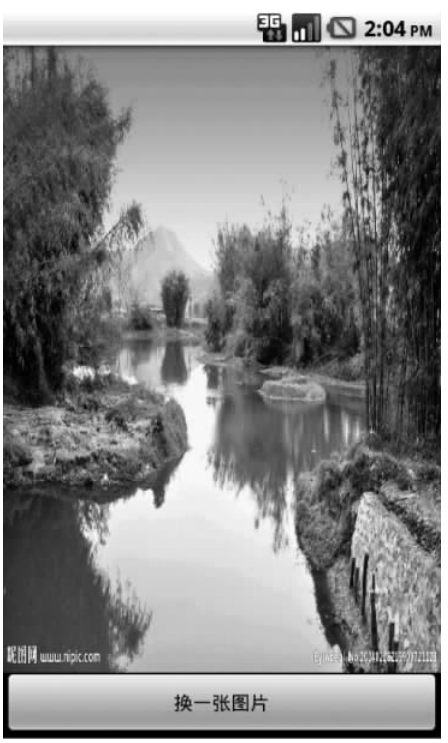

图 5.5　在 XML 界面布局中直接使用图片资源

图 5.6　在 Java 代码中使用图片资源

5.4 通过声音播放功能使用样式资源、主题资源和原始资源

假如经常需要对某个类型的组件设置大致相似的属性和对应的属性值，也就是说，该类型的组件在该应用中保持一样的长相，例如字体、颜色、背景色等。如果每次都对该 View 组件进行重复的属性的指定，这明显是不科学的，也不利于后期项目的维护。

类似于 Web 开发中定义一个一个的 css 文件，文件中的一块一块的 css 代码，Android 提供了样式资源的概念：一个样式等于一组格式的集合。为一个组件设置了某个样式后，该样式所包含的全部格式将会应用于该组件。这就是样式资源。

主题资源与样式资源非常类似，主要区别在于：

- 主题资源不能作用于单个 View 组件，主题应用对整个应用中的所有的 Activity 起作用或对指定的 Activity 起作用。
- 主题资源定义的格式应该是改变窗口外观的格式，例如，窗口标题、窗口边框等。

原始资源，指的是声音资源等 Android 应用会用到的大量的原生态的资源。类似声音文件以及其他各种类型的文件，只要 Android 没有为其提供专门的支持，这种资源都被称为原始资源。Android 的原始资源被存放在/res/raw 目录下以及/assets 目录下。只是 Android SDK 会在 R 清单类中为/res/raw 目录下的资源生成一个索引，而/assets 目录下的资源会是更彻底的原始资源，Android 应用需要通过 AssetManager 来管理该目录下的原始资源。

本节通过一个简单的声音播放功能来介绍如何使用样式资源、主题资源和原始资源。实例需要用到的样式资源以及主题资源文件 my_style.xml 代码如下：

```
<?xml version="1.0" encoding="utf-8"?>
<resources xmlns:android="http://schemas.android.com/apk/res/android">
    <!-- 定义一个样式,指定字号大小以及字体颜色 -->
    <style name="style1">
        <item name="android:textSize">20sp</item>
        <item name="android:textColor">#FF0000</item>
    </style>

    <!-- 定义一个样式,继承前一个样式 -->
    <style name="style2" parent="@style/style1">
        <item name="android:background">#FFFF00</item>
        <item name="android:padding">8dp</item>
        <!-- 覆盖父样式中指定的属性 -->
        <item name="android:textColor">#000000</item>
    </style>

    <!-- 定义主题资源 -->
    <style name="mytheme">
        <item name="android:windowNoTitle">true</item>
        <item name="android:windowFullscreen">true</item>
        <item name="android:windowBackground">@drawable/ic_launcher</item>
    </style>
```

```
</resources>
```

代码文件：codes\05\5.4\AudioDemo\res\values\my_style.xml

从上面的代码可以看出，样式以及主题资源文件的根元素是< resources >，该元素内可包含多个< style >子元素，每个< style >元素定义一个样式或主题，< style >元素指定如下两个属性：

- name——指定样式或主题的名称；
- parent——指定该样式或主题所继承的父样式或父主题。当继承某个父样式或主题，该样式或主题将会获得父样式或主题中定义的全部资源，当然，当前样式或主题也可以对父样式或主题的属性进行重新设置。

例如，在上面的代码中，定义了两个样式，第二个样式继承了第一个样式并对第一个样式的 android:textColor 属性进行重新设定。

接下来，看一下在布局 XML 文件中如何使用上面所定义的样式和主题，实例的布局文件代码如下：

```
<?xml version = "1.0" encoding = "utf - 8"?>
< LinearLayout xmlns:android = "http://schemas.android.com/apk/res/android"
    android:layout_width = "match_parent"
    android:layout_height = "match_parent"
    android:orientation = "vertical"
    android:padding = "5dp" >

    < LinearLayout
        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
        android:orientation = "horizontal" >

        < TextView
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:text = "声源 1: " />

        < EditText
            android:layout_width = "200dp"
            android:layout_height = "wrap_content"
            android:text = "test1.mp3"
            style = "@style/style1" />

        < Button
            android:id = "@ + id/button1"
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:text = "播放" />
    </LinearLayout >

    < LinearLayout
        android:layout_width = "fill_parent"
        android:layout_height = "wrap_content"
```

```
        android:orientation = "horizontal" >

        < TextView
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:text = "声源 1: " />

        < EditText
            android:layout_width = "200dp"
            android:layout_height = "wrap_content"
            android:text = "test2.mp3"
            style = "@style/style2" />

        < Button
            android:id = "@ + id/button2"
            android:layout_width = "wrap_content"
            android:layout_height = "wrap_content"
            android:text = "播放" />
    </LinearLayout >

</LinearLayout >
```

代码文件：codes\05\5.4\AudioDemo\res\layout\activity_main.xml

从上面的代码可以看出，在 XML 中使用样式资源的语法格式为@[< package_name >:] style/file_name。接着再看看如何使用主题资源，一般使用主题资源是通过 Android 全局配置文件 AndroidManifest.xml 来指定 Activity 或整个应用所使用到的主题。代码如下：

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

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

    <! -- 在全局配置文件中使用主题资源 -- >
    < application
        android:allowBackup = "true"
        android:icon = "@drawable/ic_launcher"
        android:label = "@string/app_name"
        android:theme = "@style/mytheme" >
        < activity
            android:name = "cn.edu.hstc.audiodemo.activity.MainActivity"
            android:label = "@string/app_name" >
            < intent - filter >
                < action android:name = "android.intent.action.MAIN" />

                < category android:name = "android.intent.category.LAUNCHER" />
            </intent - filter >
        </activity >
```

```
    </application>

</manifest>
```

代码文件：codes\05\5.4\AudioDemo\ AndroidManifest.xml

上面的代码在< application >标签指定了 android:theme 的属性值为 my_style.xml 中对应的主题资源，即可让该主题资源作用于整个应用的 Activity，下面的每个< Activity >标签也可以单独指定该属性，覆盖< application >标签中指定的主题资源。从代码可以看出，在 XML 文件中使用主题资源的语法格式与使用样式资源一样。

接下来，看一下布局文件所对应的 Activity Java 代码，在该 Activity 文件中，将对一开始放在/res/raw 目录下的 test1.mp3 文件以及/assets 目录下的 test2.mp3 文件这两个原生资源进行访问。主程序代码如下：

```
package cn.edu.hstc.audiodemo.activity;

import com.example.audiodemo.R;

import android.app.Activity;
import android.content.res.AssetFileDescriptor;
import android.content.res.AssetManager;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends Activity {
    MediaPlayer mediaPlayer1 = null;
    MediaPlayer mediaPlayer2 = null;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //直接根据声音文件的 ID 来创建 MediaPlayer
        mediaPlayer1 = MediaPlayer.create(this, R.raw.test1);
        //获取应用的 AssetManager
        AssetManager am = getAssets();
        try {
            //获取指定文件对应的 AssetFileDescriptor
            AssetFileDescriptor afd = am.openFd("test2.mp3");
            mediaPlayer2 = new MediaPlayer();
            //使用 MediaPlayer 加载指定的声音文件
            mediaPlayer2.setDataSource(afd.getFileDescriptor());
            mediaPlayer2.prepare();
        } catch (Exception e) {
            e.printStackTrace();
        }

        //获取第一个按钮并为其绑定事件监听器
        findViewById(R.id.button1).setOnClickListener(new View.OnClickListener() {
            @Override
```

```
            public void onClick(View v) {
                //播放声音
                mediaPlayer1.start();
            }
        });
        //获取第二个按钮并为其绑定事件监听器
        findViewById(R.id.button2).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //播放声音
                mediaPlayer2.start();
            }
        });
    }
}
```

代码文件：codes\05\5.4\AudioDemo\cn\edu\hstc\activity\MainActivity.java

上面的代码分别获取/res/raw 和/assets 目录下的资源作为两个 MediaPlayer 的声源文件，并赋予两个 Button 控件不同的事件监听器来播放声音。从上面的 Java 代码可以看出，使用/res/raw 目录下的原始资源的语法格式为[< package >.]R. raw. < file_name >，当然，由以往的经验可以知道，在 XML 代码中访问/res/raw 目录下的原始资源的语法格式为：@[< package_name >:]raw/file_name。

而获取/assets 目录下的原始资源则需要通过一个专门管理/assets 目录下的原始资源的管理器类 AssetManager，本示例通过该类的 AssetFileDescriptor openFd(String fileName)方法，根据文件名来获取原始资源对应的 AssetFileDescriptor，再通过 AssetFileDescriptor 的 getFileDescriptor()方法来获取对应的原始资源。

事实上，还有另外的一种获取/assets 目录下的资源的方法，就是通过 InputStream open(String fileName)方法，根据文件名来获取原始资源对应的输入流。

至此，对如何使用样式资源、主题资源以及原始资源已经全部介绍完毕，接下来，把应用部署在 Android 模拟器上，将会看到如图 5.7 所示的运行效果。

分别单击界面上的两个 Button 按钮，可以听到不同的音乐。从运行效果也可以看出，两个文本框控件的样式受到了 my_style. xml 文件中定义的样式资源 sytle1 以及 style2 的影响，分别呈现出不同的效果，而整个窗体则受到了 my_style. xml 文件中的定义的主题资源的影响，呈现了没有状态栏、全屏、背景为 Android Robot 的效果。

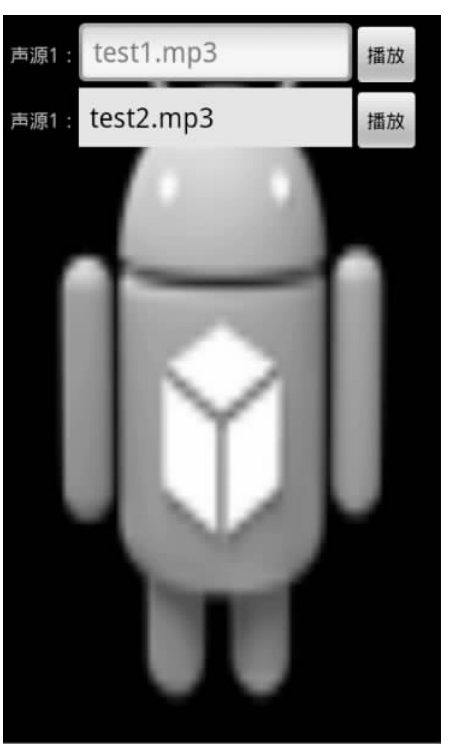

图 5.7　使用样式、主题、原始资源

5.5 本章小结

本章主要介绍了 Android 应用资源的使用，通过使用各种资源文件，Android 应用可以把各种字符串、图片、颜色、界面布局等交给 XML 文件配置管理，形成了一种高度解耦的设计模式。各种 Android 资源的存储方式以及如何使用是本章的学习重点。当然，除了本章所介绍的几种资源以外，还有很多其他类型的资源，在日后的学习工作中，读者可以慢慢进行了解。