第 3 章　Android 项目设计

- 创建新的 Android 项目
- 了解 Android 程序应用组件
- 掌握 Manifest 文件内容
- 修改 main.xml 文件

在 Eclipse IDE 开发环境中建立一个 Android 应用程序之前，首先要创建一个 Android 项目工程，并且建立一个启动配置，建立项目工程的目的就是为开发的应用程序搭建好运行环境需要的支持。在此之后你才可以开始编写、调试，以及运行应用程序。本章主要通过项目的建立，了解 Android 项目建立过程以及几个重要文件和文件内容配置。

3.1　开始第一个 Android 项目 Helloworld

1. 新建项目

打开 Eclipse，选择 File -> New -> Project -> Android Project，如图 3-1 所示。

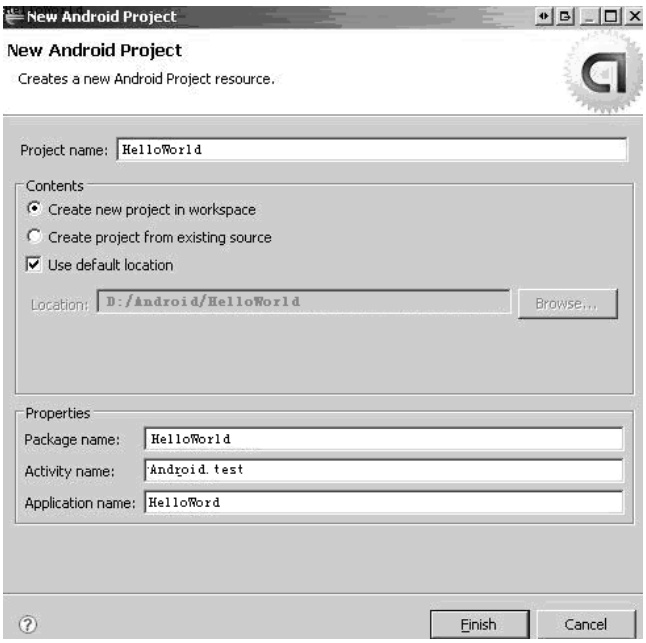

图 3-1

在图中需要用户填入和选择的项为：

- 输入工程名称（project name）：工程名称是应用程序在 Eclipse 环境中项目目录名称，如 HelloWorld。
- Content 信息栏：有几个选项，勾选 Create new project（建立新项目）和 Use default location（使用默认工作空间）。
- Properties 属性栏。在属性栏中包括：

（1）Package name：Activity 类的包名称，包名是应用程序命名空间（遵照 Java 语言命名原则定义），包名称在 Android 系统中必须唯一。

（2）Activity name 活动名称：创建 stub.java 文件等文件和程序名字。

（3）Application name 应用程序名称：应用程序名称是应用程序呈现给用户的名称，如 HelloWorld。

完成输入后单击 Finish 按钮。

2. 运行程序

当上面的过程完成后，ADT 插件会根据你的工程类型创建合适的文件和文件夹，用户可以运行应用程序，我们可以选择在模拟机中试运行程序。不过先看看 Android 工程文件结构，了解一下工程的各个组成部分以及它们的作用。

在 Eclipse 的 package Explorer 浏览器中选择刚才建立的 Helloworld 项目，可以看到根目录包含四个子目录，如图 3-2 所示。

图 3-2

- src/——Java 源代码存放目录，用于管理由 ADT（Android 开发工具）自动生成的 Activity 框架代码以及用户自己创建的代码，允许用户修改的 Java 文件都放置在这里，如上面创建的工程 HelloWorld。
- res/——资源文件夹：包含程序中所有的文件，在这个目录中存放应用将用到的资源。如 res 目录下有 res/drawable 用于放置图片资源，res/layout 用于放置布局用的 xml 文件，res/values 用于放置一些常量数据。

- gen/——自动生成目录，顾名思义就是系统自动生成的一些文件存放于此，其中最重要的文件是 R.java 文件，由 ADT 自动生成，用于引用资源文件夹中的资源文件。ADT 会根据资源夹中的 xml 布局文件，图标文件和常量自动同步 R.java 文件，所以，应避免手工修改 R.java 文件内容。
- assets/——资产目录，此用于存放原式格式文件，例如音频文件、视频文件等二进制格式文件。此目录中的资源不能被 R.java 文件索引，一般情况下为空。

一个库文件：android.jar 库文件是 Android 程序应用的函数库文件，Android 支持的 API 都包含在这个文件里。

两个工程文件：

- AndroidManifest.xml——工程清单文件。应用程序中所有的功能都在此列出，相当于配置文件，配置用户程序名称、图标、Activtity、Service、Receiver 等，如果在程序中使用到了系统内置的功能（GPS、电话服务、互联网服务、短信服务等），需要在这个文件中声明使用权限。
- default.properties——项目环境信息，一般不修改此文件。

3.2 Android 应用程序构成

Android 应用由各种各样的组件构成。这些组件大部分都是松散连接的，准确地说可以把一个应用程序看成是组件的联合而非是一个单一的应用。Android 中一个核心的概念就是组件的重用，也就是说在一个应用程序中使用的组件在另外的应用程序中也可以使用。

在 Android 系统中，为了支持组件重用的功能，系统必须可以启动这些组件，因此 Android 系统中提供了一些必要的组件，帮助系统完成组件调用功能。

一个 Android 应用程序一般有四大组件，分别是 Activity（活动）、Service（服务）、Broadcast Content Providers（内容提供者）、Receiver（广播接收器）。Android 的每个请求都由特定的组件来处理。应用程序中用到的那些组件在文件 AndroidManifest.xm 工程清单中将会列出。

- 活动（Activity）：活动（Activity）就是一个有生命周期的对象。一个 Activity 就是完成某些工作的代码块，Activity 是为用户操作而展示的可视化用户界面，用户可以与程序进行交互的界面。典型地，你将会指定你的应用程序中的一个活动为整个程序的入口点。
- 服务（Service）：服务是运行在后台的一段代码，用户不可见。它可以运行在它自己的进程，也可以运行在其他应用程序进程的上下文（context）里面，这取决于自身的需要。其他的组件可以绑定到一个服务（Service）上面，通过远程过程调用（RPC）来调用这个方法。例如媒体播放器的服务，当用户退出媒体选择用户界面，他仍然希望音乐依然可以继续播放，这时就是由服务（Service）来保证当用户界面关闭时音乐继续播放的。
- 内容提供者（Content Providers）：内容提供者（Content Providers）提供对设备上数据进行访问的数据仓库。在 Android 中，内容提供者统一了数据访问方式。典型的例子就是使用内容提供者来访问联系人列表。应用程序可以使用其他程序通过内容提供者提供的数据，同时也可以定义你自己的内容提供者来向其他应用提供数据访

问服务。

内容提供者有一个基本的方法集合来帮助程序之间可以相互使用和存储数据，应用程序通过 ContentResolver 对象可以方便地调用内容提供者的数据内容。

- 广播接收者（Broadcast Receiver）：一个广播接收者就是可以接收广播并能作出反应的组件。在 Android 系统中有各种各样的广播，如反映电池电量变化的广播。一个应用程序可以通过广播接收来监听他关心的时间并作出反应。

  广播接收者不显示在用户界面上，它可以启动一个活动来反应某些信息。或者通过 Notification Manager 来提示用户，他们一般放一个图标在状态栏上，比如我们的短信信息状态栏。

3.3　Android 几个重要项目文件的讲解

我们在上面建立了一个程序 HelloWorld，我们看看 Android 自动建立的文件内容。

3.3.1　首先建立的 HelloWorld 类

当我们建立一个 Helloworld 应用程序时，在 src 文件夹中就会为其建立一个源代码文件 Helloword.java，打开文件我们可以看到里面的具体内容。

```
1: package com.android.Helloworld;
2: import android.app.Activity;
3: import android.os.Bundle;
4: public class HelloWorld extends Activity {
/*  当第一次创建时回调 Activity 方法  */
5:        @Override
6:       public void onCreate(Bundle savedInstanceState) {
7:              super.onCreate(savedInstanceState);
8:              setContentView(R.layout.main);
9:        }
10:          }
```

第 1 行表示包的名称。

2、3 行代码导入 Android 的包，包中包含将用到的类的定义。

4～10 行是 HelloWorld 类的主体，HelloWorld 类继承自 Activity，Android 中所有的用户界面展示的类都直接或间接继承自 Activity。

其中 5～8 行是一个重要的函数，这个函数重写 Actitity 中的 onCreate，每一个继承自 Activity 的子类都要重写该方法来初始化界面，其中第 5 行中 Override 表示方法的“重写”，是 Java 的关键字，第 8 行设置了 HelloWorld 这个 Activity 要展示的用户界面的配置文件，R.layout.main 是一个资源的常量，这个资源是对 main.xml 的一个间接引用，当程序启动时将 main.xml 文件中的内容展示给用户。main.xml 就是放在 res 下，layout 下面的文件 xml 布局文件，我们可以直接使用 R.layout.main 进行直接地引用，我们要做的只是把这个 xml 文件的索引给 Android，它会自动帮我们找到并使用。

3.3.2 main.xml 布局文件内容

Android 中的 main.xml 布局文件内容主要是有关用户界面布局和设计的，利用 XML 语言描述用户界面，主要为应用程序中使用的组件进行定义、设置界面属性值，以及组件使用的资源参考设置等。例如，布局文件中为组件设置属性：定义组件的 ID 号：android:id="@+id/id 名称"；布局参数：android:属性= "属性值"等。

布局文件位于目录 res/layout 下，在此目录下找到 main.xml 文件，双击打开它，找到 main.xml 代码。

```
<?xml version="1.0" encoding="utf-8"?>   //xml 文件头，每个 xml 文件的声明语句，包括版本和编码方式
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
 android:orientation="vertical"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent"   >
<TextView     //文本组件描述
     android:layout_width="fill_parent"        //填满屏幕
     android:layout_height="wrap_content"    //填满组件控件
     android:text="@string/hello"                //应用资源文件
     />
</LinearLayout>
```

其中的黑体字符表示布局的关键字，主要表示组件在屏幕上的布局安排。

3.3.3 AndroidManifest.xml 内容

AndroidManifest.xml 是系统的控制文件，是整个项目的配置资源，它告诉系统如何处理你所创建的所有顶层组件（尤其是 Activitiy、服务、Intent 接收器和内容管理器）。应用程序在这里列出该程序实现的所有功能。举例来说，控制文件就是把你的活动（Activities）要接收的 Intents 连接在一起的连接带。

AndroidManifest.xml 存在于一个应用程序目录的根目录中，不能修改为其他名称。文件中包含了应用程序和消息对象用到的所有关系，只有在此文件中注册过的 API 才能被使用。在 Android 程序执行前，就会获取 AndroidManifest.xml 文件中的内容，如果找不到该文件或者文件内容有误，Android 系统将不能运行程序。图 3-3 是 AndroidManifest.xml 的内容结构。

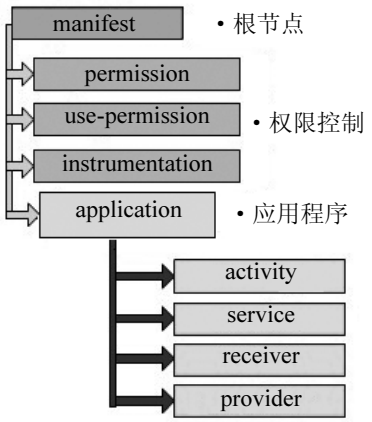

图 3-3

图 3-3 描述了 AndroidManifest.xml 文件的组成结构，<manifest> 文件根节点用来描述.apk 文件，其中 package 包属性必须给出，包括包名称、版本号等属性。permission 权限说明用来定义控制其他包对板报内的组件访问的权限对象。application 应用描述是应用组件的说明部分，包含

了包中的所有应用组件的属性描述，如果应用中使用的组件没有在此说明注册，那么系统在运行此应用程序时无法找到这个应用组件。

下面是 AndroidManifest.xml 的内容例子：

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest
      xmlns:android=http://schemas.android.com/apk/res/android
3 package="com.android.helloworld "> //定义应用程序的包名称
4 <application
      android:icon="@drawable/icon">
5 <activity                          //定义活动的内容
6    android:name="test"        //定义活动的名称
7    android:label="@string/app_name"> //定义 Android 应用程序的标签名称
8    <intent-filter>
9     <action android:name="android.intent.action.MAIN" />
10     <category android:name="android.intent.category.LAUNCHER" />
11     </intent-filter>
12 </activity>
13 </application>
14 </manifest>
```

第 2 行<manifest>元素是文件的根元素，是必须有的。xmlns:android 定义了 Android 的命名空间，使用指向 http://schemas.android.com/apk/res/android 文件。

第 3 行 package 属性是指定 Android 应用所在的包。

第 4 行<application>元素，这是非常重要的一个元素，开发的组件都会在这个元素下定义。这里是 Android 应用程序的图标，其中@drawable/icon 是一种资源引用方式，表示资源类型是图像，资源名称为 icon，对应的资源文件为 res/drawable 目录下的 icon.png。

第 5～12 行定义<activity>应用程序组件，这个元素的作用是注册一个 Activity 信息，Activity 在 Android 中属于组件，需要在功能清单文件中进行配置。

在前面章节中，当我们创建一个新的项目的时候，有一个多选框提示是否 create Activity, 指定了一个指定的活动的名称（NameActivity），系统自动创建了一个名为 NameActivity.java 的文件。

其中第 8～11 行是 Android Intent 定义，intent-filter 描述了此 Activity 启动的位置和时间。每当一个 Activity（或者操作系统）要执行一个操作时，它将创建出一个 Intent 的对象，这个 Intent 对象能承载的信息可描述你想做什么、你想处理什么数据、数据的类型以及一些其他信息。而 Android 则会和每个 Application 提供的 intent-filter 的数据进行比较，找到最合适的 Activity 来处理调用者所指定的数据和操作，具体指示我们在后面章节介绍。

3.3.4　其他的文件

1. string.xml 文件

res 目录下有一个 value 子目录，其下有一个 string.xml 文件，这个文件是用来存放所有文本信息和数值的。

使用 string.xml 文件主要有两个目的：一是为了程序国际化，程序的开发如果面向的用户

不同，界面文字的显示信息就不同，比如面向中国的用户，我们的屏幕显示就应该使用中文信息，而使用其他语言的用户就需要本国的语言提示，所以用一个 string.xml 文件把所有屏幕中出现的文字信息都集中存储，当需要改变程序的文字显示时，只需修改 sting.xml 文件内容，无需修改主程序。其二，主要是为了减少文字的重复使用，降低数据冗余，当一个提示信息需要在程序中使用多次时，就可以放在 string.xml 文件中，需要的时候只需要引用就可。

string.xml 文件详解：

```
<?xml version="1.0"encoding="utf-8"?>
<resources>
<string name="hello">HelloWorld,HelloActivity!</string>
<string name="app_name">Android,你好！ </string>
</resources>
```

每个 string 标签声明了一个字符串，name 属性指定引用名。标注：value 文件下可有多个 xml 文件，不同的类别可以用不同名字的 xml 文件写，但根元素必须都是<resources>，只有这样才能识别调用资源。例如：arrays.xml，colors.xml，dimens.xml 等。

2. R.java 文件

Android 应用程序目录 gen 下保存的是项目的所有包及源文件（.java），gen 目录下包含了项目中的所有资源。

而.java 格式的文件是在建立项目时自动生成的，这个文件是只读模式，不能更改。R.java 文件是定义该项目所有资源的索引文件。

```
package com.android.helloworld;
public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040001;
        public static final int hello=0x7f040000;
    }
}
```

在上述代码中定义了很多常量，并且这些常量的名字都与 res 文件夹中的文件名相同，这是因为.java 文件中所存储的是该项目所有资源的索引。有了这个文件，在程序中使用资源将变得更加方便，可以很快地找到要使用的资源。由于这个文件不能被手动编辑，所以当在项目中加入新的资源时，只需刷新一下该项目，.java 文件便自动生成了所有资源的索引。

3.4 在模拟器上运行项目

在开发环境中运行 Android 项目，单击工具栏的“运行”按钮，或选择菜单 Run->Run，

或右键 HelloWorld 项目文件夹，会弹出 Run As 对话框，选择 Android Application，单击 OK 按钮。运行效果如图 3-4 所示。

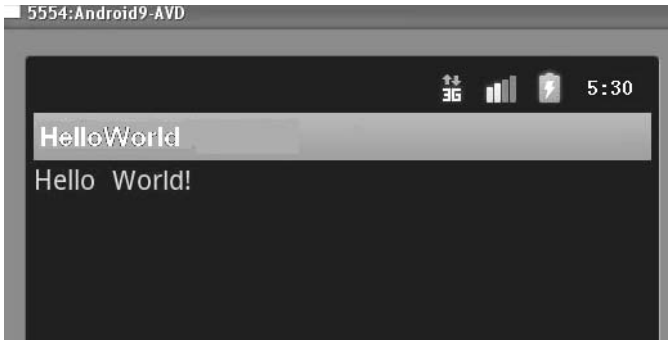

图 3-4

3.5　打包 Android 程序

当一个 Android 程序编写完成后，可以将程序打包，方便在移动手机上运行，Android 操作系统可以方便地实现打包程序，不会因为手机厂商不同实现不同的打包程序，这样可以方便开发人员进行程序打包。

用户在 Eclipse 中可以方便地实现程序的打包工作。Android 应用程序使用 Java 做为开发语言。aapt工具把编译后的 Java 代码连同其他应用程序需要的数据和资源文件一起打包到一个 Android 包文件中，文件使用.apk 做为扩展名，它是分发应用程序并安装到移动设备的媒介，用户只需下载并安装此文件到他们的设备。单一.apk 文件中的所有代码被认为是一个应用程序。

在本章中，我们介绍了在 Android 系统中建立一个程序项目的过程，应用程序工程包含的主要目录和文件。应用程序工程模拟测试以及程序打包应用的过程，通过本章学习，对 Android 程序设计流程有一个初步的了解。

1．Android 应用程序组件有哪些？
2．Manifest 文件的主要作用？
3．Android 四大组件是什么？

第 4 章　Android Activity 介绍

- Activity 作用
- Activity 程序的基本组成
- Activity 与 Android 项目中主要组成部分
- Android 文件结构

4.1　Activity 介绍

在 Android 系统中，Activity 是一个可见的、人机交互的界面，主要处理前端事务，接收用户的动作指令。在 Activity 中存放各个显示控件，这些控件可以是显示菜单、下拉菜单、文本输入等。程序的界面风格就是 Activity 设计风格。Activity 是 Android 的基本组成部分。

Activity 是提供视图（View）控件绘制的环境，我们可以认为一个 Activity 是视图组件的一个容器。通常在应用程序中一个 Activity 是单独的一个屏幕，一个 Android 程序将由多个 Activity 程序组成。通常一个应用程序中第一个呈现给用户的 Activity（活动）称为主活动（Main Activity）。

Activity 负责处理自己当前屏幕的内容，包括界面、菜单、填出菜单、程序执行等。当我们从一个屏幕画面切换到另一个屏幕画面的时候，就涉及了 Activity 之间的切换。

当 Activity 进行切换时，Activity 切换活动有两种类型，以是否需要与其他 Activity 交换资料分为“独立”Activity 与“相依”的 Activity。不同类型的 Activity，其动作也不尽相同：

1. 独立 Activity 活动

独立的 Activity 是不需要从其他地方取得信息的 Activity。只是单纯的从一个屏幕跳到下个屏幕，不涉及信息的交换。从一个独立的 Activity 呼叫另一个独立的 Activity 时，我们只要填好意图（Intent，一种携带参数信息的 Android 组件）的内容和动作，使用 startActivity() 函数调用，即可建立起独立的 Activity。

2. 相依的 Activity 活动

相依的 Activity 是需要与其他 Activity 交换信息的一种 Activity。相依的 Activity 又可再分为单向与双向；从一个屏幕跳到下一个屏幕时，将会把一些参数提供给下一个屏幕（Activity）使用，就是单向相依的 Activity；要在两个屏幕之间切换，屏幕上的信息会因另一个屏幕的操作而改变的，就是双向相依的 Activity。

创建一个 Activity

一个 Activity 通常显示为一个窗口，这个窗口会占满整个屏幕，不过，这个窗口的大小也

是可以调整的。

一个活动的窗口中有很多可见内容，我们称之为控件或者 View，这些 View 一般具有层次关系，都继承 View 类。用户通过操作这些 View 或控件提交要求获得需要的信息，因此 Activity 为用户和 View 之间的交互提供了场地。Android 中也为程序开发者提供了许多可以直接使用的 View 控件，包括 buttons（按钮）、text fields（文本输入）、menu items（菜单选项）、check boxes（选项）等组件。

1. 创建 Activity

创建 Activity 本身很简单，需要的步骤如下：

（1）继承 Activity 类。创建一个 Activity，必须继承 Activity 类或其子类，并重写相关的回调函数，其中最重要的两个函数是 onCreate()和 onPause()函数。

（2）实现 Activity 的回调方法：比如 onCreate()方法，一定要记得在里面回调父类的 onCreate 方法和 setContentView 方法，由父类去布局界面。改写 onCreate 方法，代码实例如下：

```
package com.TestDemo.activity;
import android.app.Activity;
import android.os.Bundle;
public class MyActivity01Activity extends Activity {
        /* 当第一次创建时回调 Activity 方法 */
        @Override
        public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

（3）创建 Activity 里面具体的 View 组件，并设置它们的属性和布局方式。例如添加 TextView 控件，打开 Layout 文件下的 main.xml，所有的控件都要在此文件中注册。

```
<?xml version="1.0" encoding="utf-8"?>
    <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:orientation="horizontal" >
        <TextView
        android:id="@+id/text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, world" />
    </LinearLayout>
```

（4）具体的绘制界面的工作。

这样一个 Activity 就创建好了。创建好的活动 Activity 要能正常使用，需要完成下面一些工作。

2. 使用 Activity

要使用 Activity 也比较简单，Activity 的使用有两种方式，一是作为程序启动后展现的界

面；第二就是从另外的地方调用。

不管是哪种使用方式，任何 Activity 都必须要在 AndroidManifest.xml 文件中进行配置，也就是注册过程，注册过的 Activity 才能使用。

在 AndroidMainfest.xml 中注册如下：

```
……
<activity
android:name=".MyActivity01Activity" //Activity 类名
android:label="@string/app_name">  //Activity 标签
  <intent-filter>    //该 Activity 的意向过滤信息，指明该 Activity 允许的动作和分类
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
</activity>
```

3. *启动* Activity

运行程序如 4-1 所示

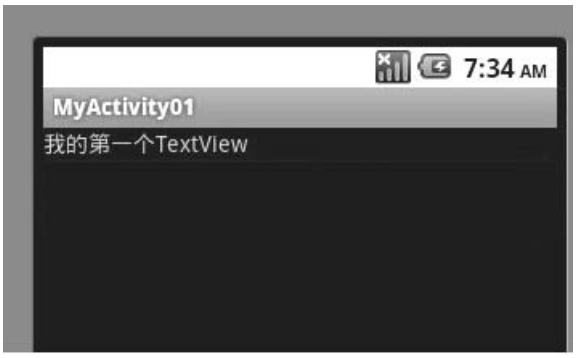

图 4-1

4. *关闭* Activity

Activity 可以通过调用它自己的方法 finish()来关闭，同时一个 Activity 可以通过方法 finishActivity()来关闭另一个 Activity。

4.2 Activity 生命周期

所谓周期就是从开始到结束的一个轮回，Activity 的主要功能是与用户打交道，Activity 的生命周期就是从建立一个 Activity 到这个 Activity 活动结束。下面的图显示了 Activity 的重要状态转换，矩形框表明 Activity 在状态转换之间的回调接口，开发人员可以重载实现以便执行相关代码，椭圆形表明 Activity 所处的状态。

在图 4-2 中的 Activity 七个生命周期函数：

```
void onCreate(Bundle savedInstanceState)
void onStart()
void onRestart()
void onResume()
```

```
void onPause()
void onStop()
void onDestroy()
```

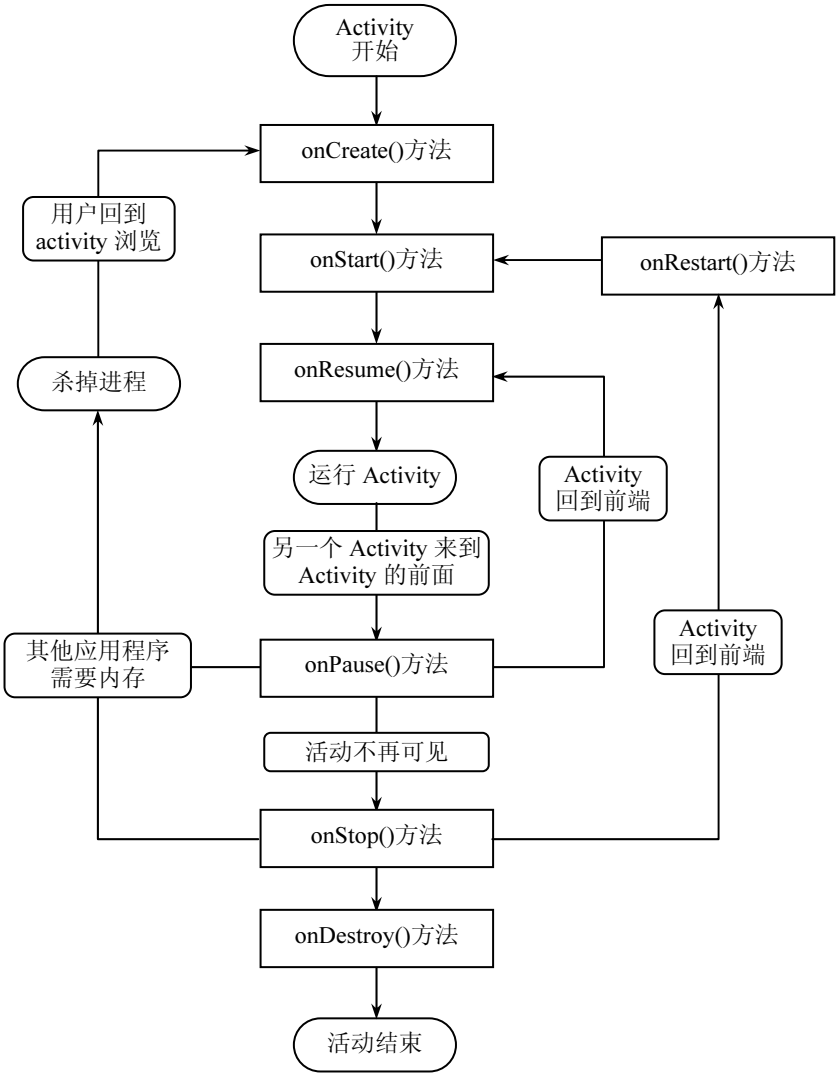

图 4-2

从图 4-2 我们知道，一个 Activity 主要有 4 个状态：

（1）运行状态。当 Activity 在屏幕前台时（位于当前任务堆栈的顶部），它是活跃或运行的状态。此时它对应用户操作的 Activity，比如向用户提供信息、捕获捕获用户动作等。

（2）暂停状态。当一个 Activity 失去焦点但仍然对用户可见时，它处于暂停状态。也就是被另外一个 Activity 取代。这个 Activity 也许是透明的，或者未能完全遮蔽全屏，所以被暂停，但是这个 Activity 仍对用户可见。暂停的 Activity 仍然是存活状态（它保留着所有的状态和成员信息并连接至窗口管理器），但当系统处于极低内存的情况下，仍然可以杀死这个 Activity。

（3）停止状态。如果一个 Activity 完全被另一个 Activity 覆盖，那它处于停止状态。它仍然保留所有的状态和成员信息。然而它不再被用户可见，它的窗口也将被隐藏，如果其他地方需要内存，则系统经常会杀死这个 Activity。

（4）kill 状态。如果一个 Activity 处于暂停或停止状态，系统可以通过要求它结束（调用它的 finish()方法）或直接杀死它的进程来将它驱出内存。当它再次为用户可见的时候，它只能完全重新启动并恢复至以前的状态。

Android 管理 Activity 通过叫栈的技术处理，如图 4-3 所示。

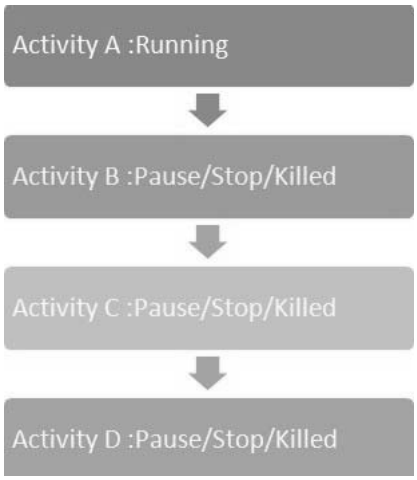

图 4-3

正在运行的 Activity 处在栈的最顶端，它是运行状态的；当有新 Activity 进入屏幕最上端时，原来的 Activity 就会被压入第二层，如果它的屏幕没有被完全遮盖，那么它处于 Pause 状态，如果被遮盖，那么它处于 Stop 状态。但是不管程序当前处于哪一层，都可能在系统觉得资源不足时被强行关闭，处在栈底的程序最先被关闭。

【例 1】通过一个 Activity 讲解生命周期过程。

第一步，运行状态。

如图 4-4 所示，当开始启动应用程序时，Activity 生命周期中执行的方法是 OnCreate()、onStart()与 Onresume。

图 4-4

第二步，暂停状态。

输入文字后，按 Home 键，程序会在后台运行，执行的方法会是：onPause()，onStop()。当再次启动该程序，执行的方法会是：onRestart()，onStart()，onResume()，并且刚才输入的文字会显示在文本框里，如图 4-5 所示。

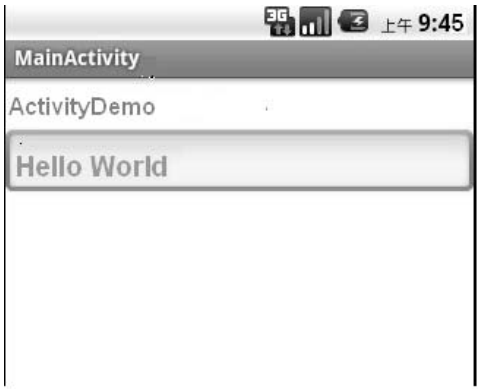

图 4-5

第三步，停止状态。

按回退键，会退出程序，执行的方法是：onPause()，onStop()，onDestroy()。

再次执行该程序，会执行的方法：onStart()与 Onresume，活动重新被创建的。

本章小结

本章主要介绍了 Android 系统中四大组件之一 Activity 活动的主要知识，介绍了 Activity 的生命周期过程、概念以及方法。

习题 4

1．Activity 的主要功能是什么？

2．Activity 的生命周期包括几个部分？

3．简述 Activity 的几个生存状态。

4．简述 Activity 的建立过程。