第5章 Android组件通信

杨刚
中国人民大学

本章学习目标

- 了解使用Intent进行组件通信的原理
- 掌握使用Intent启动Activity的方法
- 掌握获取Activity返回值的方法
- 了解Intent过滤器的原理与匹配机制

5.1 Intent简介

- 在一个Android应用中，主要是由四种组件组成的
  - Activity, Intent Receiver, Service, Content Provider
  - 这四种组件相互独立、可以互相调用，协调工作，最终组成一个真正的Android应用。
  - 在这些组件之间的通讯中，主要是由Intent协助完成的
- 应用程序内部、应用程序之间数据通信

5.1 Intent简介

- Intent：一种轻量级的消息传递机制
  - Intent：“意图，目的”， 可以理解为不同组件之间通信的“媒介”“信使” What do you want to do?
  - 它是一个动作的完整描述，包含了动作的产生组件、接收组件和传递的数据信息
  - Intent的目的：用于组件之间数据交换
    - Activity、Service和BroadcastReceiver之间的数据交互
    - 启动Activity和Service
  - 发送广播消息

5.1 Intent简介

- Intent的优点
  - 使用Intent来传播动作，利于组件之间的分离，允许无缝地替换应用程序元素。
  - 提供了一个简单的用于扩展应用程序功能的模型的基础架构

5.1 Intent简介

- 5.1.1 启动Activity
  - 应用程序一般都有多个Activity，Intent可以实现不同Activity之间的切换和数据传递
  - 启动Activity方式
    - 显式启动
      - 必须在Intent中指明启动的Activity所在的类
    - 隐式启动
      - 根据Intent的动作和数据来决定启动哪一个Activity
      - 选择权由Android系统和最终用户来决定

5.1 Intent简介

- 不同类型的组件有不同的传递Intent方式：
  - 激活一个新的Activity，或者让一个现有的Activity做新的操作，可以通过调用Context.startActivity()或者Activity.startActivityForResult()方法；
  - 启动一个新的Service，或者向一个已有的Service传递新的指令，调用Context.startService()方法或者调用Context.bindService()方法将调用此方法的上下文对象与Service绑定；

5.1 Intent简介

- 不同类型的组件有不同的传递Intent方式：
  - Context.sendBroadcast()、Context.sendOrderBroadcast()、Context.sendStickBroadcast()这三个方法可以发送Broadcast Intent。发送之后，所有已注册的并且拥有与之相匹配IntentFilter的BroadcastReceiver就会被激活

5.1 Intent简介

- 5.1.1 启动Activity
  - 显式启动
    - 创建一个Intent
    - 指定当前的应用程序上下文，以及要启动的Activity
    - 把创建好的Intent作为参数传递给startActivity()方法

```
1. Intent intent = new Intent(IntentDemo.this, ActivityToStart.class);
2. startActivity(intent);
```

    - 主要用于同一个应用程序中的Activity切换。在同一个应用程序内，一般来说，我们通常都知道要启动的 Activity 具体是哪一个，因此常用显式的 Intent 来实现，简单直接。

5.1 Intent简介

- 5.1.1 启动Activity
  - 显式启动
    - 指定了component属性的Intent(调用setComponent(ComponentName)或者setClass(Context, Class)来指定)。通过指定具体的组件类，通知应用启动对应的组件
    - 务必在AndroidManifest.xml文件中注册这两个Activity

发起端

```
button.setOnClickListener(new Button.OnClickListener() {
    @Override
    public void onClick(View arg0) {
        // TODO Auto-generated method stub
        Intent intent = new Intent();
        ComponentName com = new ComponentName(
                "com.example.testcomponent",
                "com.example.testcomponent.MyActivity");
        intent.setComponent(com);
        String s = "This is a Test.";
        intent.putExtra("id", s);
        startActivity(intent);
    }
});
```

接收端

```
text = (TextView)findViewById(R.id.text);

Intent intent = getIntent();
String str = intent.getStringExtra("id");
ComponentName com = intent.getComponent();
String pkgName = com.getPackageName();
String className = com.getClassName();
text.setText(str + "\n" + pkgName + "\n" + className);
```

5.1 Intent简介

- 5.1.1 启动Activity
  - 隐式启动
    - 即Intent的发送者不指定接收者，很可能不知道也不关心接收者是谁，而由Android框架去寻找最匹配的接收者
    - 有利于降低组件之间的耦合度
    - 隐式启动的原理：
      - 选择隐式启动Activity，Android系统会在程序运行时解析Intent，并根据一定的规则对Intent和Activity进行匹配，使Intent上的动作、数据与Activity完全吻合
      - 匹配的组件可以是程序本身的Activity，也可以是Android系统内置的Activity，还可以是第三方应用程序提供的Activity
      - 这种方式强调了Android组件的可复用性

5.1 Intent简介

- 5.1.1 启动Activity
  - 隐式启动
    - 使用Intent隐式启动浏览器，查看指定的网页内容，
      - 可将“浏览动作”和“Web地址”作为参数传递给Intent
      - Android系统则通过匹配动作和数据格式，找到最适合于此动作和数据格式的组件
        - 在匹配Intent时，根据动作Intent.ACTION_VIEW，得知需要启动具备浏览功能的Activity，但具体是浏览电话号码还是浏览网页，还需要根据URI的数据类型来做最后判断
        - 因为数据提供的是Web地址"http://www.google.com"，所以最终可以判定Intent需要启动具有网页浏览功能的Activity
        - 在缺省情况下，Android系统会调用内置的Web浏览器

```
1. Intent intent = new Intent(Intent.ACTION_VIEW,
   Uri.parse("http://www.google.com"));
2. startActivity(intent);
```

5.1 Intent简介

- Intent中传递的数据分为：
  - 操作（Action）
  - 数据（Data）
  - 数据类型（Type）
  - 操作类别（Category）
  - 附加信息（Extras）
  - 组件（Component）
  - 标志（Flag）

5.1 Intent简介

- 系统常用的Action

调动Intent过滤器匹配的Activity

Action	描述
ACTION_ANSWER	打开接听电话的Activity，默认为Android内置的拨号界面
ACTION_CALL	打开拨号盘界面并拨打电话，使用Uri中的数字部分作为电话号码
ACTION_DELETE	打开一个Activity，对所提供的数据进行删除操作
ACTION_DIAL	打开内置拨号界面，显示Uri中提供的电话号码
ACTION_EDIT	打开一个Activity，对所提供的数据进行编辑操作
ACTION_INSERT	打开一个Activity，在提供数据的当前位置插入新项
ACTION_PICK	启动一个子Activity，从提供的数据列表中选取一项
ACTION_SEARCH	启动一个Activity，执行搜索动作
ACTION_SENDTO	启动一个Activity，向数据提供的联系人发送信息
ACTION_SEND	启动一个可以发送数据的Activity
ACTION_VIEW	最常用的动作，对以Uri方式传送的数据，根据Uri协议部分以最佳方式启动相应的Activity进行处理。对于http:address将打开浏览器查看；对于tel:address将打开拨号界面并呼叫指定的电话号码
ACTION_WEB_SEARCH	打开一个Activity，对提供的数据进行Web搜索

5.1 Intent简介

- 常用的Data
  - 用Android.net.Uri类进行设置

Data（Uri）格式	描述
http://网页地址	浏览网页
tel:电话号码	拨打电话
smsto:短信接收人号码	发送短信
file://sdcard/文件或目录	查找sd卡文件
geo:坐标，坐标	显示地图

- 常用的Type
  - 指定要传送数据的MIME类型，用setType()方法进行设置

MIME类型	描述
vdn.android-dir/mms-sms	发送短信
Image/png	设置图片
text/plain	普通文本
audio/mp3	设置音乐

5.1 Intent简介

- 常用的操作类别（Category）
  - 可以用addCategory()发给对Action设置多个类别

Category名称	描述
CATEGORY_LAUNCHER	程序显示在应用程序列表中
CATEGORY_HOME	显示主桌面，开机时的第一个界面
CATEGORY_PREFERENCE	运行后将出现一个选择面板
CATEGORY_DEFAULT	设置ACTION的默认执行
CATEGORY_OPENABLE	当Action设置为GET_CONTENT时用于打开指定的Uri

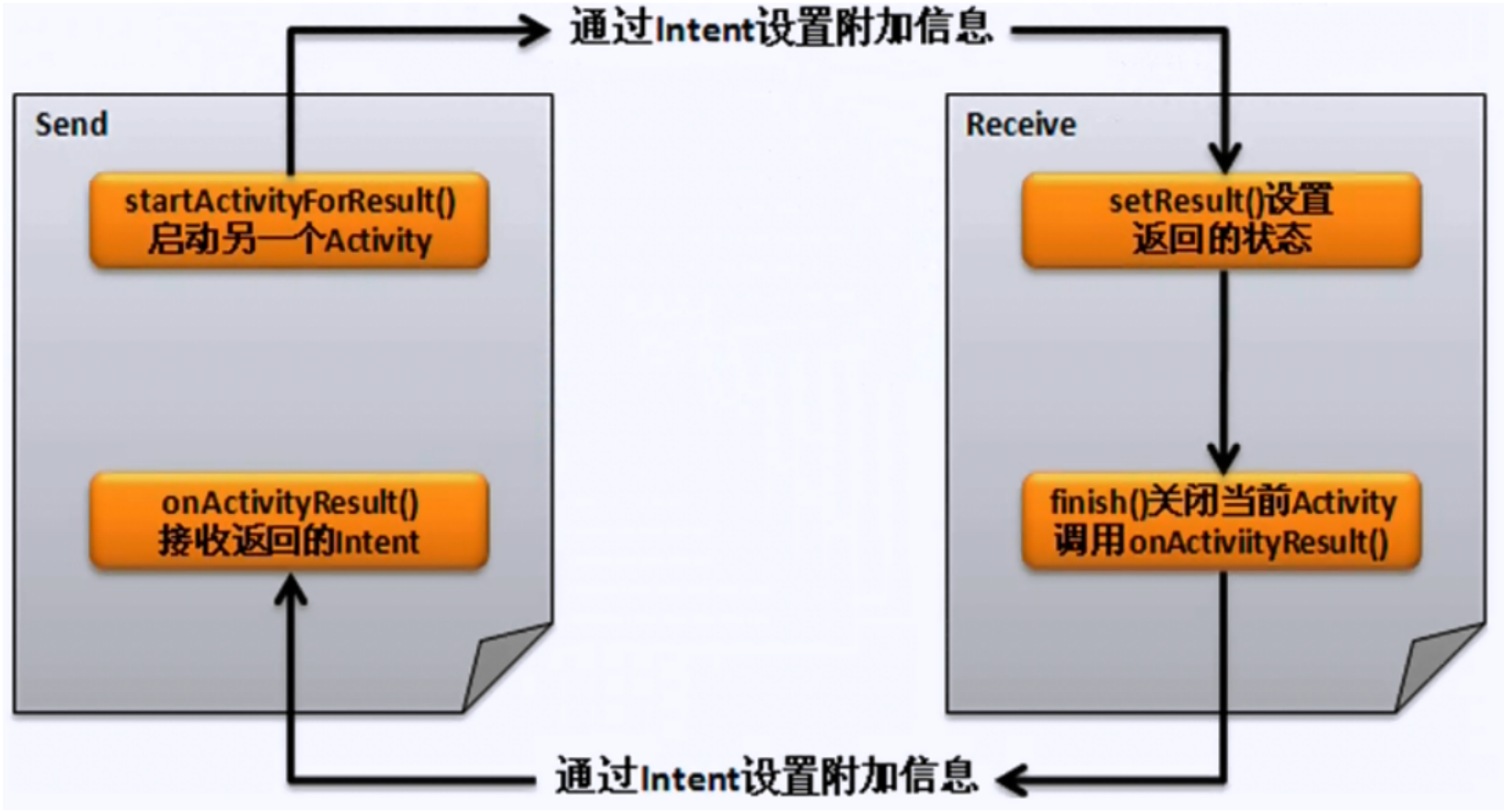
5.1 Intent简介

- 附加信息（Extras）
  - 传递一组键值对，用putExtra()方法设置
  - 传递数据（uri）所需要的一些额外的操作信息
- 组件（Component）
  - 指明将要处理的Activity程序，所有的组件信息都被封装在一个ComponentName对象中，这些组件都必须在Manifest文件的<application>中注册
- 标志（Flag）
  - 用于指示Android系统如何加载并运行一个操作，可通过addFlags()方法进行增加

5.1 Intent简介

- 5.1.2 获取Activity返回值
  - 获取子Activity的返回值，一般分为三个步骤：
    - 以Sub-Activity的方式启动子Activity ；
    - 设置子Activity的返回值；
    - 在父Activity中获取返回值；

5.1 Intent简介



- 如果Receiver需要回传给Sender数据，则不能使用startActivity()方法，必须使用startActivityForResult()

5.1 Intent简介

- Activity支持的Intent操作方法

No.	方法	描述
1	startActivity()	启动一个Activity，并通过Intent传送数据
2	startActivityForResult()	启动并接受另外一个Activity程序回传的数据
3	getIntent()	返回启动当前Activity程序的Intent
4	onActivityResults()	当需要接收Intent回传数据时覆写此方法对回传操作进行处理
5	finish()	调用此方法会返回之前的Activity程序，并自动调用onActivityResult()方法
6	managedQuery	处理返回的Cursor结果集

5.1 Intent简介

- 5.1.2 获取Activity返回值
  - 以Sub-Activity的方式启动子Activity
    - 显式启动子Activity的代码如下

```
1. int SUBACTIVITY1 = 1;
2. Intent intent = new Intent(this, SubActivity1.class);
3. startActivityForResult(intent, SUBACTIVITY1);
```

    - 隐式启动子Activity的代码如下

```
1. int SUBACTIVITY2 = 2;
2. Uri uri = Uri.parse("content://contacts/people");
3. Intent intent = new Intent(Intent.ACTION_PICK, uri);
4. startActivityForResult(intent, SUBACTIVITY2);
```

startActivityForResult(Intent, requestCode)函数，参数Intent用于决定启动哪个Activity，参数requestCode是请求码。因为所有子Activity返回时，父Activity都调用相同的处理函数，因此父Activity使用requestCode来确定数据是哪一个子Activity返回的

5.2 Intent过滤器

- Intent解析
  - Android系统一定存在一种**匹配机制**，使Android系统能够根据Intent中的数据信息，找到需要启动的组件
  - 这种匹配机制是依靠Android系统中的**Intent过滤器**（Intent Filter）来实现的

5.2 Intent过滤器

- Intent解析
  - Intent过滤器
    - 是一种根据Intent中的动作（Action）、类别（Categorie）和数据（Data）等内容，对适合接收该Intent的组件进行匹配和筛选的机制
    - 可以匹配数据类型、路径和协议，还可以确定多个匹配项顺序的优先级（Priority）
  - 注册Intent过滤器的组件
    - Activity
    - Service
    - BroadcastReceiver

5.2 Intent过滤器

- Intent解析
  - AndroidManifest.xml文件代码：

```
1.  <activity android:name=".IntentResolutionDemo"
2.          android:label="@string/app_name">
3.          <intent-filter>
4.                  <action android:name="android.intent.action.MAIN" />
5.                  <category
    android:name="android.intent.category.LAUNCHER" />
6.          </intent-filter>
7.  </activity>
8.
9.  <activity android:name=".ActivityToStart"
10.         android:label="@string/app_name">
11.         <intent-filter>
12.                 <action android:name="android.intent.action.VIEW" />
13.                 <category
    android:name="android.intent.category.DEFAULT" />
14.                 <data android:scheme="schemodemo"
    android:host=“com.example" />
15.         </intent-filter>
16. </activity>
```

5.2 Intent过滤器

- Intent解析
  - Intent启动Activity代码：

```
1. Intent intent = new Intent(Intent.ACTION_VIEW,
   Uri.parse("schemodemo://com.example/path"));
2.  startActivity(intent);
```

5.2 Intent过滤器

- Intent解析
  - 组件注册Intent过滤器，在AndroidManifest.xml文件的各个组件下定义<intent-filter>节点，然后在<intent-filter>节点中声明该组件所支持的动作、执行的环境和数据格式等信息
  - <intent-filter>节点中
    - <action>标签：Intent过滤器的“动作”
    - <category>标签：Intent过滤器的“类别”
    - <data>标签：Intent过滤器的“数据”

5.2 Intent过滤器

<intent-filter>节点支持的标签和属性说明：

标签	属性	说明
<action>	android:name	指定组件所能响应的动作，用字符串表示，通常由Java类名和包的完全限定名构成
<category>	android:category	指定以何种方式去服务Intent请求的动作
<data>	Android:host	指定一个有效的主机名
	android:mimetype	指定组件能处理的数据类型
	android:path	有效的URI路径名
	android:port	主机的有效端口号
	android:scheme	所需要的特定协议

5.2 Intent过滤器

- Intent解析
  - <category>标签用来指定Intent过滤器的服务方式
  - Android系统提供的类别：

值	说明
ALTERNATIVE	Intent数据默认动作的一个可替换的执行方法
SELECTED_ALTERNATIVE	和ALTERNATIVE类似，但替换的执行方法不是指定的，而是被解析出来的
BROWSABLE	声明Activity可以由浏览器启动
DEFAULT	为Intent过滤器中定义的数据提供默认动作
HOME	设备启动后显示的第一个Activity
LAUNCHER	在应用程序启动时首先被显示

5.2 Intent过滤器

- Intent解析
  - Intent到Intent过滤器的映射
  - 匹配规则：

（1） Android系统把所有应用程序包中的Intent过滤器集合在一起，形成一个完整的Intent过滤器列表

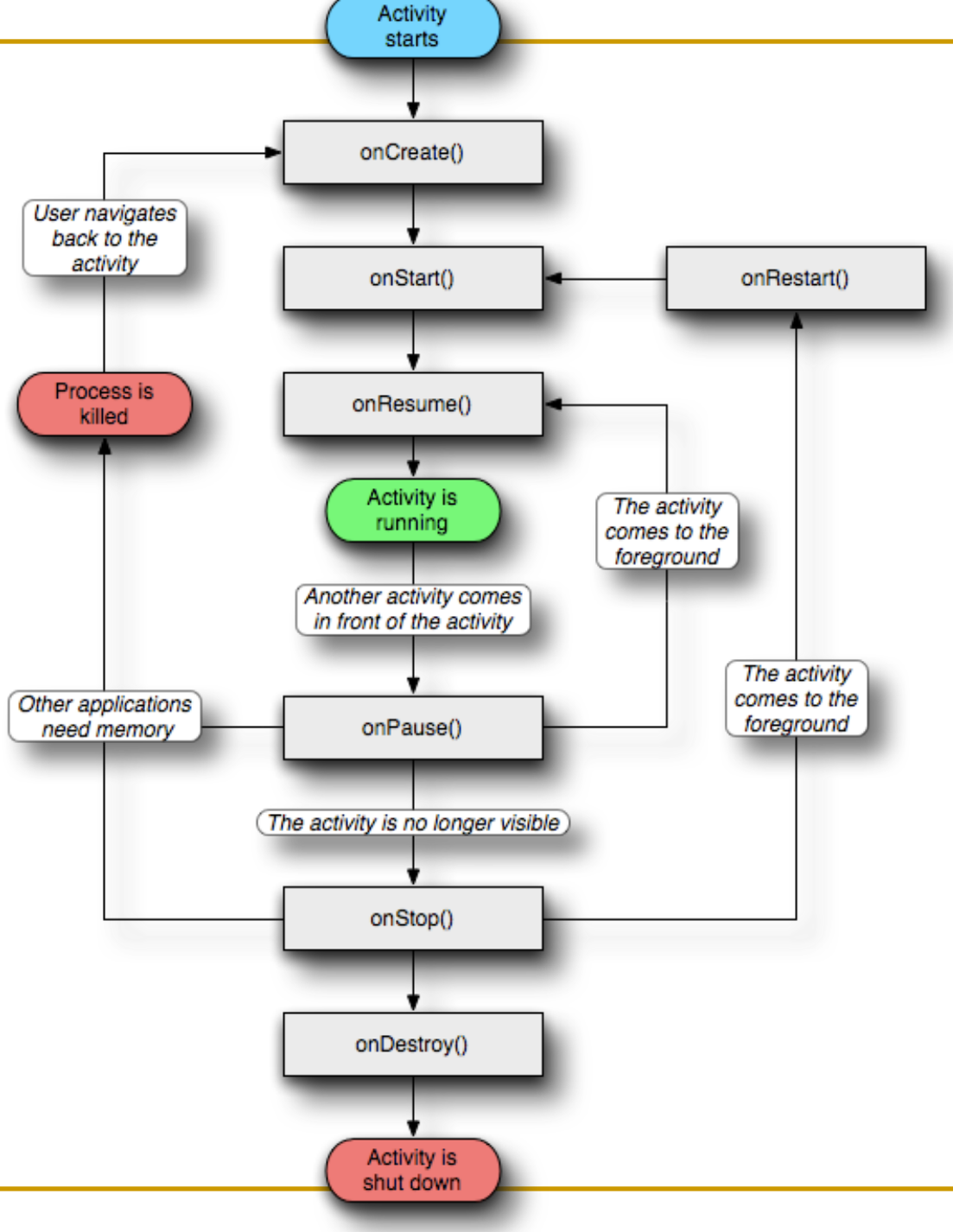
（2） 在Intent与Intent过滤器进行匹配时，Android系统会将列表中所有Intent过滤器的“动作”和“类别”与Intent进行匹配，任何不匹配的Intent过滤器都将被过滤掉。没有指定“动作”的Intent过滤器可以匹配任何的Intent，但是没有指定“类别”的Intent过滤器只能匹配没有“类别”的Intent

5.2 Intent过滤器

- Intent解析
  - Intent到Intent过滤器的映射
  - 匹配规则：

（3） 把Intent数据Uri的每个子部与Intent过滤器的<data>标签中的属性进行匹配，如果<data>标签指定了协议、主机名、路径名或MIME类型，那么这些属性都要与Intent的Uri数据部分进行匹配，任何不匹配的Intent过滤器均被过滤掉

（4） 如果Intent过滤器的匹配结果多于一个，则可以根据在<intent-filter>标签中定义的优先级标签来对Intent过滤器进行排序，优先级最高的Intent过滤器将被选择


Activity starts
onCreate()
User navigates back to the activity
onStart()
onRestart()
Process is killed
onResume()
Activity is running
The activity comes to the foreground
Another activity comes in front of the activity
Other applications need memory
onPause()
The activity comes to the foreground
The activity is no longer visible
onStop()
onDestroy()
Activity is shut down