Intents

5.1 Introduction

Till now, you have learned to create and design the Activities. But rarely, a single Activity is enough for an Android application. An Android application is made up of multiple Activities, Services and Broadcast Receivers which are the core constituents of an Android application. It means there should be some kind of interaction between Activities. This is done through another important constituent of an Android app called Intent. Intent works like a glue between the different Activities to work together and gives the seamless performance to your application. Intents will be discussed thoroughly in this Section.

Intent: Intent is an object of Intent class that is used to start a new component or passing some data to another component.

App Component: An application component can be an Activity, Service or Broadcast Receiver.

URI: A URI (Uniform Resource Identifier) is string that is used to identify a resource such as web browser, phone dialler or an Activity of Another app.

Intent-filter: Intent filter stipulates the types of intents to which an Activity can respond. It specifies for the corresponding Activity to listen the specific types of actions of specific category.

Bundle class: A Bundle class object is used to wrap a set of strings in a parcel form.

Broadcast Receiver: A Broadcast Receiver is an Android application component which is used to respond to the broadcasted messages of other applications.

5.2 Intent

Intent is an object that is used to pass the data or request to another application component (such as an Activity, a Service, or a Broadcast Receiver). For example, it can help you to switch from one Activity to another by clicking on any UI component. An Intent object can be used:

- To start a new Activity.
- To pass the data to another Activity.
- To receive the result or data from another Activity.
- To start any background operation (i.e. Service).
- To bind a service with another component.
- To broadcast a message such as the ringing the phone or receiving an SMS (i.e. to start a Broadcast Receiver).

5.2.1 Types of Intents

Mainly, there are two types of intents: Explicit Intents and Implicit Intents.

Explicit Intents: Explicit Intents are used to call a specific component using fully qualified class name. It is used when you know which component you want to launch and don't want to give the free control to the user to choose a specific component to process the request. In abstract, it is used to start another component in your own application. Suppose you have an application that has 2 activities: Activity A and activity B. You want to launch the activity B from activity A. In this case you define an explicit Intent in Activity A to target the Activity B and then use it to call the Activity B directly.

Implicit Intents: It is used when you have an idea of what you want to do, but you do not know which component should be launched. If you want to give the user an option to choose between a list of components, then you must use Implicit Intents. Implicit Intents are sent to the Android system then it searches for all components which are registered for the specific action and the data type. If only one component is found, Android starts the component directly; for example, you have an application that uses the camera to take photos. One of the features of your application is that you give the user the possibility to send the photos he has taken. You do not know what kind of application a user has that can send photos, and you also want to give the user an option to choose which external application to be used if he / she has more than one application. In this case you should not use an explicit intent; instead of using the explicit intents, you should use an implicit Intent.

When you use an explicit Intent to start service or an activity, the system instantly starts the application component quantified in the object of Intent. But when you are using an implicit intent, then it is the responsibility of Android system to find the suitable component to start. The Android system does it by comparing the contents of the intent to the *Intent Filters*. Intent Filters are the expressions written under the <activity> element of the Manifest file to specify the type of Intent likely to be received by the component. The Android system starts the component if the intent matches an Intent Filter. If many Intent Filters are well-matched, the Android system shows a dialog on the screen from where a user can choose which application must be use. It is shown in Figure 5.1.

When an explicit intent calls its target component the intent filters are not checked, on the other hand implicit intent calls its target component only if it matches to any of the component's filters.

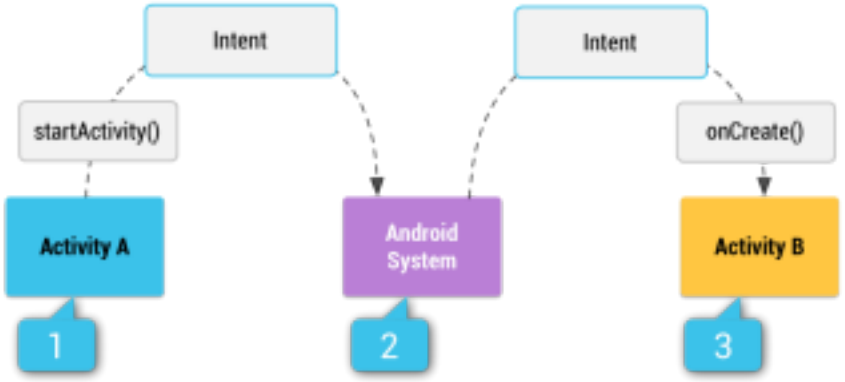

Figure 5.1: How an implicit intent is delivered through the system to start another activity: [1] Activity A creates an Intent with an action description and passes it to startActivity(). [2] The Android System searches all apps for an intent filter that matches the intent. When a match is found, [3] the system starts the matching activity (Activity B) by invoking its onCreate() method and passing it the Intent.
(Source: http://developer.android.com/guide/components/intents-filters.html)

5.2.2 Working with Intents

To work with the intents first you must understand the types of data that Intent can carry with it.

Information carried by Intents: An Intent object is received by the component as a bundle of information. The information can be used either to receive the Intent only or to take the decision by

Android system to decide which component should be start and what action to be performed. The information contained in the Intent object can be of the following types:

1. ***Component Name:*** It is the name of the target component. It can be a fully qualified path of the target component such as the Activity class name. The name of the component to start. This is optional, but it's the critical piece of information that makes an intent explicit, meaning that the intent should be delivered only to the app component defined by the component name. Without a component name, the intent is implicit, and the system decides which component should receive the intent based on the other intent information (such as the action, data, and category—described below). If you need to start a specific component in your app, you should specify the component name.
2. ***Action:*** An action is a string that is used by an Activity to specify an action to be perform. The Intent class has several action constants corresponding to different intents. Some of the common actions performed are ACTION_VIEW, ACTION_MAIN, ACTION_EDIT, etc. You can specify the action when creating the Intent object.
3. ***Data:*** It is a URI (Uniform Resource Locator) that references the data to be worked on. A URI is sent in the pairing of action constant. The Data (URI) and the action pair defines the operation to be performed. For example, if the action is ACTION_DIAL then the URI "tel:123" is used to display the phone dialler with the given number 123 filled into it.
4. ***Category:*** It is a string that contains the extra information about the type of component that will handle the intent. For example, category CATEGORY_LAUNCHER is used to specify that the Activity is the launcher Activity, i.e. it is the first Activity to be displayed on the screen when the application is launched.
5. ***Extras:*** When you want to bind additional data (extra data) to the Intent object, putExtra() method is used. The data is wrapped in the object of the Bundle class in the form of key-value pairs.

For more details of the type of the information that can be contained by an Intent, explore the following link:

http://developer.android.com/reference/android/content/Intent.html

Reading

To understand the working of the Intents, first learn to link the Activities using Explicit Intents and Implicit Intents respectively.

5.2.2.1 Linking the Activities using Intents

As you know an Intent can be used either to start another component (such as an Activity) or to pass the data to another component. Generally, Activities are linked to each other with the help of an Intents. To understand this, use the following examples.

1. **Explicit Intent to start another Activity:** For example, first create two activities in two different XML layout files; one is having a Button view, and another is having a TextView view.

```
<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:orientation="vertical"
tools:context="com.nikhil.explicit_intent.MainActivity">
<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Start Activity"/>
</LinearLayout>
```

activity_main.xml

The output screen of the activity_main.xml file is shown in Figure 5.2.1.

```
<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"
tools:context="com.nikhil.explicit_intent.SecondActivity">

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="You are welcomed in Second Activity."/>

</LinearLayout>
```

second.xml

The output screen of the activity_main.xml file is shown in Figure 5.2.2.

For the layout file second.xml, create the SecondActivity.java file.

```
package com.nikhil.explicit_intent;
import android.app.Activity;
import android.os.Bundle;

public class SecondActivity extends Activity {
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.second);
        }
}
```

SecondActivity.java

For the main layout file activity_main.xml create the following java file ActivityMain.java. This java

file has an Intent object that is used to start the activity SecondActivity which has a welcome message.

```
package com.nikhil.explicit_intent;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

publicclass MainActivity extends Activity {
        Button b1;
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                b1=(Button) findViewById(R.id.button1);
                b1.setOnClickListener(new OnClickListener() {
                        public void onClick(View v) {
                                Intent i = new Intent(MainActivity.this, SecondActivity.class);
                                 startActivity(i);
                        }
                });
        }
}
```

MainActivity.java

Whenever a new Activity is created, it needs to be registered in the Manifest file of the application.

```
<?xml version="1.0"encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.nikhil.explicit_intent"
android:versionCode="1"
android:versionName="1.0">

<uses-sdk
android:minSdkVersion="19"
android:targetSdkVersion="23"/>
<application
android:allowBackup="true"
android:icon="@drawable/ic_launcher"
android:label="@string/app_name"
android:theme="@style/AppTheme">
<activity
android:name=".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>
<activity android:name=".SecondActivity"></activity>
</application>
</manifest>
```

AndroidManifest.xml

Figure 5.2.1: Main Activity

Figure 5.2.2: Second Activity

2. **Implicit Intent to start a web browser:** To understand the Implicit Intents, you will use the following code to start a built-in browser to open the specified web page. Remember, if there are more than one web browser in your device, then Android system will ask you to choose one.

The layout file of the Activity having only one button to start a web browser will be as follows:

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/LinearLayout1"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

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

</LinearLayout>
```

activity_main.xml

Java code for the Activity will be as follows:

```
package com.nikhil.implicitintent;

import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;

public class MainActivity extends Activity {
        @Override
        protectedvoid onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                Button b1 = (Button) findViewById(R.id.button1);
                b1.setOnClickListener(new OnClickListener(){
                        @Override
                        public void onClick(View argo) {
                                Intent i = new
Intent(Intent.ACTION_VIEW,Uri.parse("https://www.facebook.com/dcaugepv")
                                ); startActivity(i);
                        }
                });
        }
}
```

MainActivity.java

There will not be any changes in the Manifest file because you didn't include a new Activity in the application.

When you will run the above code, the output will be as shown in Figure 5.3.1 and Figure 5.3.2.

Figure 5.3.1: Main Activity

Figure 5.3.2: Web Browser showing a web page

5.2.2.2 Intent Filters

Intent Filter is a very important construct that you must learn to understand the working of Intents more profoundly. If an Intent is sent to the Android system, it will determine suitable applications for this Intent. If several components have been registered for this type of Intents, Android offers the user a choice to open one of them. This is done due to the Intent Filters. Intent filter stipulates the types of intents to which an Activity can respond. It specifies for the corresponding Activity to listen the specific types of actions of specific category. An Intent Filter declares the capabilities of a component. It specifies what an activity or service do and what types of broadcasts a Receiver can handle. It allows the corresponding component to receive Intents of the declared type. Intent Filters are typically defined via the AndroidManifest.xml file. For Broadcast Receiver it is also possible to define them in coding. An Intent Filters is defined by its category, action, and data filters. It can also contain additional metadata.

If a component does not define an Intent filter, it can only be called by explicit Intents. There are two ways of defining Intent Filters; you can define Intent Filters in either your Manifest file or Broadcast Receiver. If you register the intent in Manifest file, then Android registers the filter when your application gets installed. If you want to receive an intent at run time, then use it for the Broadcast Receiver. To use the intent for Broadcast Receiver, you must define it at run time programmatically.

To know more about the Intent-Filters, follow the following link:

https://developer.android.com/guide/components/intents-common

---