Intent

Learning Objective

After studying this unit, the student should be able to

- Understand the structure of an Intent
- Perform operations on Intent
- List the types of Intent
- Know what is Intent Resolution?
- Give example of Intent
- List standard Activity Actions
- List standard Broadcast Actions

Introduction

Intent is an abstract description of an operation to be performed. It can be used to launch an Activity, broadcastIntent to send it to any interested BroadcastReceiver components, and to communicate with a background Service.

Intent provides a facility for performing late runtime binding between the codes in different applications. Its most significant use is in the launching of activities, where it can be thought of as the glue between activities. It is basically a passive data structure holding an abstract description of an action to be performed.

Intent Structure

The intent has primary attributes which are mandatory and secondary attributes which are optional.

Primary Attributes

Primary Attributes: The primary pieces of information in intent are:

1. **Action:** The general action to be performed, such as ACTION_VIEW, ACTION_EDIT, ACTION_MAIN, etc.
2. **Data:** The data to operate on, such as a person record in the contacts database, expressed as an Uri.

Some examples of action/data pairs are:

- ACTION_VIEW ***content://contacts/people/9*** : Display information about the person whose identifier is "9".
- ACTION_DIAL ***content://contacts/people/9*** : Display the phone dialer with the person filled in.
- ACTION_VIEW ***tel:123*** : Display the phone dialer with the given number filled in. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.
- ACTION_DIAL ***tel:123*** : Display the phone dialer with the given number filled in.
- ACTION_EDIT ***content://contacts/people/9*** : Edit information about the person whose identifier is "9".
- ACTION_VIEW ***content://contacts/people/*** : Display a list of people, which the user can browse through. This example is a typical top-level entry into the Contacts application, showing you the list of people. Selecting a particular person to view would result in a new intent {ACTION_VIEWcontent://contacts/people/N } being used to start an activity to display that person.

Secondary Attributes

In addition to these primary attributes, there are a number of secondary attributes that you can also include with intent:

- **Category:** Gives additional information about the action to execute. For example, CATEGORY_LAUNCHERmeans it should appear in the Launcher as a top-level application, while CATEGORY_ALTERNATIVE means it should be included in a list of alternative actions the user can perform on a piece of data.
- **Type:** Specifies an explicit type (a MIME type) of the intent data. Normally the type is inferred from the data itself. By setting this attribute, you disable that evaluation and force an explicit type.

- **Component:** Specifies an explicit name of a component class to use for the intent. Normally this is determined by looking at the other information in the intent (the action, data/type, and categories) and matching that with a component that can handle it. If this attribute is set then none of the evaluation is performed, and this component is used exactly as is. By specifying this attribute, all of the other Intent attributes become optional.

- **Extras:** This is a Bundle of any additional information. This can be used to provide extended information to the component. For example, if we have a action to send an e-mail message, we could also include extra pieces of data here to supply a subject, body, etc.

Other Operations on Intent

Here are some examples of other operations you can specify as intents using these additional parameters:

- **ACTION_MAIN with category CATEGORY_HOME:** Launch the home screen.
- **ACTION_GET_CONTENT with MIME type vnd.android.cursor.item/phone:** Display the list of people's phone numbers, allowing the user to browse through them and pick one and return it to the parent activity.
- **ACTION_GET_CONTENT with MIME type */* and category CATEGORY_OPENABLE**: Display all pickers for data that can be opened and allowing the user to pick one of them and then some data inside of it and returning the resulting URI to the caller. This can be used, for example, in an e-mail application to allow the user to pick some data to include as an attachment.

There are a variety of standard Intent action and category constants defined in the Intent class, but applications can also define their own, for example, the standard ACTION_VIEW is called "android.intent.action.VIEW".

Types of Intent

There are two primary forms of intents you will use.

- **Explicit Intents** have specified a component which provides the exact class to be run. Often these will not include any other information, simply being a way for an application to launch various internal activities it has as the user interacts with the application.
- **Implicit Intents** have not specified a component; instead, they must include enough information for the system to determine which of the available components is best to run for that intent.

Intent Resolution

When using implicit intents, given such an arbitrary intent we need to know what to do with it. This is handled by the process of Intent resolution, which maps an Intent to an Activity, BroadcastReceiver, or Service that can handle it.

The intent resolution mechanism basically revolves around matching Intent against all of the <intent-filter> descriptions in the installed application packages.

There are three pieces of information in the Intent that are used for resolution: the action, type, and category. Using this information, a query is done on the PackageManager for a component that can handle the intent. The appropriate component is determined based on the intent information supplied in the AndroidManifest.xml file as follows:

- The action, if given, must be listed by the component as one it handles.
- The type is retrieved from the Intent's data, if not already supplied in the Intent. Like the action, if a type is included in the intent (either explicitly or implicitly in its data), then this must be listed by the component as one it handles.
- For data that is not a content: URI and where no explicit type is included in the Intent, instead the scheme of the intent data (such as http: or mailto:) is considered. Again like the action, if we are matching a scheme it must be listed by the component as one it can handle.
- The categories, if supplied, must all be listed by the activity as categories it handles. That is, if you include the categories CATEGORY_LAUNCHER and

CATEGORY_ALTERNATIVE, then you will only resolve to components with an intent that lists both of those categories. Activities will very often need to support the CATEGORY_DEFAULT so that they can be found by startActivity.

Example of Intent

For example, consider the Note Pad sample application that allows a user to browse through a list of notes data and view details about individual items.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="in.edu.baou.notepad">
  <application android:icon="@drawable/app_notes"  android:label="@string/app_name">
        <provider class=".NotePadProvider" android:authorities="in.edu.baou.provider.NotePad" />

        <activity class=".NotesList" android:label="@string/title_notes_list">
                <intent-filter>
                        <action android:name="android.intent.action.MAIN" />
                        <category android:name="android.intent.category.LAUNCHER" />
                </intent-filter>
                <intent-filter>
                        <action android:name="android.intent.action.VIEW" />
                        <action android:name="android.intent.action.EDIT" />
                        <action android:name="android.intent.action.PICK" />
                        <category android:name="android.intent.category.DEFAULT" />
                        <data android:mimeType="vnd.android.cursor.dir/vnd.google.note" />
                </intent-filter>
                <intent-filter>
                        <action android:name="android.intent.action.GET_CONTENT" />
                        <category android:name="android.intent.category.DEFAULT" />
                        <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
                </intent-filter>
      </activity>

      <activity class=".NoteEditor" android:label="@string/title_note">
                <intent-filter android:label="@string/resolve_edit">
                        <action android:name="android.intent.action.VIEW" />
                        <action android:name="android.intent.action.EDIT" />
                        <category android:name="android.intent.category.DEFAULT" />
                        <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
                </intent-filter>
```

```
            <intent-filter>
                <action android:name="android.intent.action.INSERT" />
                <category android:name="android.intent.category.DEFAULT" />
                <data android:mimeType="vnd.android.cursor.dir/vnd.google.note" />
            </intent-filter>
        </activity>
        <activity class=".TitleEditor" android:label="@string/title_edit_title"
            android:theme="@android:style/Theme.Dialog">
            <intent-filter android:label="@string/resolve_title">
                <action android:name="com.android.notepad.action.EDIT_TITLE" />
                <category android:name="android.intent.category.DEFAULT" />
                <category android:name="android.intent.category.ALTERNATIVE" />
                <category android:name="android.intent.category.SELECTED_ALTERNATIVE" />
                <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Explanation of Example

In above example, the first activity, in.edu.baou.provider.notepad,NotesList, serves as our main entry into the app. It can do three things as described by its three intent templates:

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

This provides a top-level entry into the NotePad application: the standard MAIN action is a main entry point (not requiring any other information in the Intent), and the LAUNCHER category says that this entry point should be listed in the application launcher.

```
 <intent-filter>
     <action android:name="android.intent.action.VIEW" />
     <action android:name="android.intent.action.EDIT" />
     <action android:name="android.intent.action.PICK" />
     <category android:name="android.intent.category.DEFAULT" />
```

```
    <data android:mimeType="vnd.android.cursor.dir/vnd.google.note" />
</intent-filter>
```

This declares the things that the activity can do on a directory of notes. The type being supported is given with the <type> tag, where vnd.android.cursor.dir/vnd.google.note is a URI from which a Cursor of zero or more items (vnd.android.cursor.dir) can be retrieved which holds our note pad data (vnd.google.note). The activity allows the user to view or edit the directory of data (via the VIEW and EDIT actions), or to pick a particular note and return it to the caller (via the PICK action). Note also the DEFAULT category supplied here: this is *required* for the startActivity method to resolve your activity when its component name is not explicitly specified.

```
<intent-filter>
    <action android:name="android.intent.action.GET_CONTENT" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
</intent-filter>
```

This filter describes the ability to return to the caller a note selected by the user without needing to know where it came from. The data type vnd.android.cursor.item/vnd.google.note is a URI from which a Cursor of exactly one (vnd.android.cursor.item) item can be retrieved which contains our note pad data (vnd.google.note). The GET_CONTENT action is similar to the PICK action, where the activity will return to its caller a piece of data selected by the user. Here, however, the caller specifies the type of data they desire instead of the type of data the user will be picking from.

Given these capabilities, the following intents will resolve to the NotesList activity:

- **{ action=android.app.action.MAIN }** matches all of the activities that can be used as top-level entry points into an application.
- **{ action=android.app.action.MAIN, category=android.app.category.LAUNCHER }** is the actual intent used by the Launcher to populate its top-level list.
- **{ action=android.intent.action.VIEW data=content://com.google.provider.NotePad/notes }** displays a list of all the

notes under "content://com.google.provider.NotePad/notes", which the user can browse through and see the details on.

- **{ action=android.app.action.PICK data=content://com.google.provider.NotePad/notes }** provides a list of the notes under "content://com.google.provider.NotePad/notes", from which the user can pick a note whose data URL is returned back to the caller.
- **{ action=android.app.action.GET_CONTENT type=vnd.android.cursor.item/vnd.google.note }** is similar to the pick action, but allows the caller to specify the kind of data they want back so that the system can find the appropriate activity to pick something of that data type.
- The second activity, in.edu.baou.notepad.NoteEditor, shows the user a single note entry and allows them to edit it. It can do two things as described by its two intent templates:

```
<intent-filter android:label="@string/resolve_edit">
    <action android:name="android.intent.action.VIEW" />
    <action android:name="android.intent.action.EDIT" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
</intent-filter>
```

The first, primary, purpose of this activity is to let the user interact with a single note, as described by the MIME type vnd.android.cursor.item/vnd.google.note. The activity can either VIEW a note or allow the user to EDIT it. Again we support the DEFAULT category to allow the activity to be launched without explicitly specifying its component.

```
<intent-filter>
    <action android:name="android.intent.action.INSERT" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:mimeType="vnd.android.cursor.dir/vnd.google.note" />
</intent-filter>
```

The secondary use of this activity is to insert a new note entry into an existing directory of notes. This is used when the user creates a new note: the INSERT

action is executed on the directory of notes, causing this activity to run and have the user create the new note data which it then adds to the content provider.

Given these capabilities, the following intents will resolve to the NoteEditor activity:

- **{action=android.intent.action.VIEW data=content://in.edu.baou.provider.NotePad/notes/*{ID}*}** shows the user the content of note ***{ID}***.
- **{ action=android.app.action.EDIT data=content:// in.edu.baou.provider.NotePad/notes/*{ID}* }** allows the user to edit the content of note ***{ID}***.
- **{ action=android.app.action.INSERT data=content:// in.edu.baou.provider.NotePad/notes }** creates a new, empty note in the notes list at "content://com.google.provider.NotePad/notes" and allows the user to edit it. If they keep their changes, the URI of the newly created note is returned to the caller.
- The last activity, com.android.notepad.TitleEditor, allows the user to edit the title of a note. This could be implemented as a class that the application directly invokes (by explicitly setting its component in the Intent), but here we show a way you can publish alternative operations on existing data:

```
<intent-filter android:label="@string/resolve_title">
    <action android:name="com.android.notepad.action.EDIT_TITLE" />
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.ALTERNATIVE" />
    <category android:name="android.intent.category.SELECTED_ALTERNATIVE"
/>
    <data android:mimeType="vnd.android.cursor.item/vnd.google.note" />
</intent-filter>
```

- In the single intent template here, we have created our own private action calledcom.android.notepad.action.EDIT_TITLE which means to edit the title of a note. It must be invoked on a specific note like the previous view and edit actions, but here displays and edits the title contained in the note data.
- In addition to supporting the default category as usual, our title editor also supports two other standard categories: ALTERNATIVE and SELECTED_ALTERNATIVE. Implementing these categories allows others to find the special action it provides

without directly knowing about it, through the PackageManager.queryIntentActivityOptions(ComponentName, Intent[], Intent, int) method, or more often to build dynamic menu items with Menu.addIntentOptions(int, int, int, ComponentName, Intent[], Intent, int, MenuItem[]). Note that in the intent template here was also supply an explicit name for the template (via android:label="@string/resolve_title") to better control what the user sees when presented with this activity as an alternative action to the data they are viewing.

- Given these capabilities, the following intent will resolve to the TitleEditor activity:
- {action=com.android.notepad.action.EDIT_TITLE data=content://com.google.provider.NotePad/notes/*{ID}*} displays and allows the user to edit the title associated with note *{ID}*.

Standard Activity Actions

These are the current standard actions that Intent defines for launching activities (usually through Context#startActivity. The most important, and by far most frequently used, are ACTION_MAIN and ACTION_EDIT.

ACTION_MAIN	ACTION_DIAL	ACTION_RUN
ACTION_VIEW	ACTION_CALL	ACTION_SYNC
ACTION_ATTACH_DATA	ACTION_SEND	ACTION_PICK_ACTIVITY
ACTION_EDIT	ACTION_SENDTO	ACTION_SEARCH
ACTION_PICK	ACTION_ANSWER	ACTION_WEB_SEARCH
ACTION_CHOOSER	ACTION_INSERT	ACTION_FACTORY_TEST
ACTION_GET_CONTENT	ACTION_DELETE	

Standard Broadcast Actions

These are the current standard actions that Intent defines for receiving broadcasts (usually through registerReceiver or a <receiver> tag in a manifest).

ACTION_TIME_TICK	ACTION_PACKAGE_DATA_CLEARED
ACTION_TIME_CHANGED	ACTION_PACKAGES_SUSPENDED
ACTION_TIMEZONE_CHANGED	ACTION_PACKAGES_UNSUSPENDED
ACTION_BOOT_COMPLETED	ACTION_UID_REMOVED
ACTION_PACKAGE_ADDED	ACTION_BATTERY_CHANGED

ACTION_PACKAGE_CHANGED	ACTION_POWER_CONNECTED
ACTION_PACKAGE_REMOVED	ACTION_POWER_DISCONNECTED
ACTION_PACKAGE_RESTARTED	ACTION_SHUTDOWN

Let us sum up

In this unit we have learned about the structure of Intent, what type of operation can be performed on Intent, types of Intent, Intent Resolution, take example of Intent, discussed standard Activity actions and standard broadcast actions.

Further Reading

- https://developer.android.com/reference/android/content/Intent

Acknowledgement: “The content in this module is modifications based on work created and shared by the Android Open-Source Project and used according to terms described in the Creative Commons 2.5 Attribution License.”