Lecture 9: Intent and Broadcast

Jianjun Chen (Jianjun.Chen@xjtlu.edu.cn)

Intent

Explicit, Implicit, Intent-filter

Types of Intents

- Explicit intents:
  - “Specify which application will satisfy the intent, by supplying either the target app's package name or a fully-qualified component class name.”
    - Intent intent = new Intent(Activity1.this, Activity2.class);
    - https://stackoverflow.com/questions/14658469/android-intent-context-confusing.
- Implicit intents:
  - Declare a general action to perform.
  - Android system finds the appropriate component by comparing this implicit intent to **intent filters**.

Properties of an Intent

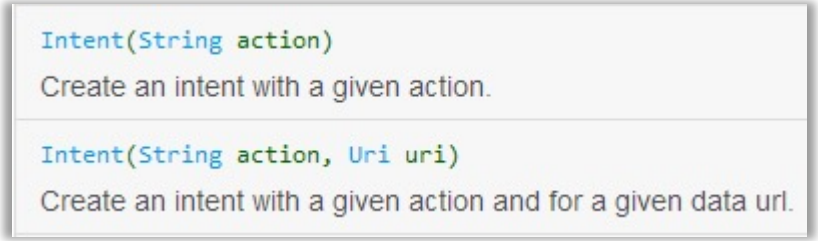
- To support implicit and explicit intents, 6 properties are defined:
  - Action, Data, Category, Type, Component, Extras
- These properties are used to facilitate communications between components. Three of the most fundamental communications are:
  - Starting an activity.
  - Starting a service.
  - Broadcasting.

Components of an Intent

- Implicit intents:
  - Use **Action**, **Data**, **Category** and **Type**.
- Explicit intents:
  - Use **Component**.
- To pass data (Bundle) between activities (or other system components): **Extra**.

Intent: Action

- An **Action** in intent is basically a string.
- You can set the **action** of an intent using
  - `intent.setAction(String)`
  - Intent constructors also allow you to set actions.



- Android has its own set of standard **action**s.
  - ACTION_MAIN, ACTION_VIEW …
  - Check the API document for intent.

Intent: Data

- Data is an **uniform resource identifier (URI)**. URI is a formatted piece of text that is an immutable one-to-one mapping to a resource or data.
  - `<scheme>://<host>:<port>/<path>`
  - For example, the URI of a contact with identifier 1 is content://contacts/1
  - ftp://ftp.is.co.za/rfc/rfc1808.txt
- It makes more sense if you combine **data** and **action** together:

```
ACTION_VIEW    content://contacts/people/1
```

- This asks android to display information about the person whose identifier is "1".

Intent: Data

- To set the **data** of an intent, use:

```
intent.setData(Uri data)
```

- To get an Uri object of something, use

```
Uri.parse()
```

- For example:

```
Intent intent = new Intent(Intent.ACTION_DIAL);
intent.setData(Uri.parse("tel:1771778888"));
startActivity(intent);
```

URI spec: http://www.faqs.org/rfcs/rfc2396.html.

Intent: Category

- Gives additional information about the **action** to execute.
- To set the **category** of an intent:

```
intent.addCategory("CATEGORY_NAME");
```

- You can add multiple categories to an intent.

Intent: Type

- This property provides the data type (MIME) enforcement to the target.
  - Such as what kind of files I want to open in the file browser
- Normally the **type** is inferred from the **data** itself. By setting this attribute, you disable that evaluation and force an explicit type.
- A MIME type is a string indicating the type of a file:
  - A sound can be labelled as “audio/ogg” or “audio/mp3”
  - A image can be labelled as “image/png” or “image/jpeg”

Intent: Type

- To set MIME data type of an Intent:

```
intent.setType(String)
```

- This will unset any **data** that was previously set.

- You can set **data** and **type** in one go:

```
setDataAndType(Uri data, String type)
```

Example

```
static final int REQUEST_IMAGE_GET = 1;

public void selectImage() {
    Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
    intent.setType("image/*");
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(intent, REQUEST_IMAGE_GET);
    }
}

protected void onActivityResult
            (int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == REQUEST_IMAGE_GET && resultCode == RESULT_OK) {
        Bitmap thumbnail = data.getExtras().getParcelable("data");
        Uri fullPhotoUri = data.getData();
        // Do work with photo saved at fullPhotoUri
    }
}
```

Intent: Component

- Specifies an **explicit** name of a component class to use for the intent.

```
Intent i2 = new Intent(getApplicationContext(), ImageShow.class);
startActivity(i2);
```

- `ImageShow.class` is the component.

Intent: Extras

- This is a **`Bundle`** of any additional information. Extras can be used to provide extended information to the component.

- Check our previous slides if you have no idea what a `Bundle` is.

Intent Resolution

Intent filters, Matching Process

Implicit Intent & Intent Filters

- Intent is a request message sent by a source components.
- To be able to receive an **implicit intent** request, the target component must first register itself in the Android System.
- The registration is achieved by providing `<intent-filter>` blocks in the manifest XML.
- https://developer.android.google.cn/guide/components/intents-filters?hl=en

Intent Filters

AndroidManifest.xml

- An intent filter is an expression in an app's manifest file that specifies the type of (implicit) intents that the component would like to receive.

```
<activity android:name=".IntentTest">
   <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
<activity android:name=".ImageShow">
   <intent-filter>
      <action android:name="cn.edu.xjtlu.TestApp.SHOW_SLUG" />
      <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
```

Intent Resolution

- When an Intent is sent to Android, the system will check it against **all** registered intent filters.
- Items compared:

Intent-filter	intent
<action android:name=…>	intent action
<data …>	intent MIME type and Data
<category android: name=…>	intent category

- Each <intent-filter> block can have several action, data and category components defined.

Intent Resolution

- If a component is designed to receive multiple types of intents, it will need several `<intent-filter>` blocks.

- To successfully send an intent to a component, the intent will be must pass the **action**, **type** and **category** check of any filter block of that component.

Action & Action Test

- Example:

```
<action android:name="action_name" />
```

- The action specified in the Intent must match one of the **action** block inside the filter to pass the test.
  - An intent filter with zero action declared will never receive any intent.

Category & Category Test

- Example:

```
<category android:name="category_name" />
```

- Every **category** in the Intent must match a **category** in the filter. The reverse is not necessary.
  - An intent with zero category will always pass the test.
  - An intent with categories A, B matches a filter with categories A, B, C.

Category & Category Test

- Android automatically applies the `CATEGORY_DEFAULT` category to all implicit intents passed to `startActivity()` and `startActivityForResult()`.

- An activity's category should always have `android.intent.category.DEFAULT` in its intent filter.

Data & Data Test

- <data> element:

```
<data android:scheme="string"
      android:host="string"
      android:port="string"
      android:path="string"
      android:pathPattern="string"
      android:pathPrefix="string"
      android:mimeType="string" />
```

Read More:

https://developer.android.com/guide/topics/manifest/data-element

https://developer.android.com/guide/components/intents-filters#DataTest

```
<scheme>://<host>:<port>[<path>|<pathPrefix>|<pathPattern>]
```

- Data test compares both the URI and the MIME type in the intent to a URI and MIME type specified in the filter.

Intent: Example 1

- The code below tries to start the calculator in the system.
  - works on Xiaomi tablet, but not on AVD android system.

```
Intent intent = new Intent();
intent.setAction(Intent.ACTION_MAIN);
intent.addCategory(Intent.CATEGORY_APP_CALCULATOR);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(intent);
```

Intent: Example 2

- The code below starts the browser and redirects the user to our university's homepage

```
Uri webpage = Uri.parse("https://www.xjtlu.edu.cn");
Intent intent = new Intent(Intent.ACTION_VIEW, webpage);
if (intent.resolveActivity(getPackageManager()) != null) {
    startActivity(intent);
}
```

Checks whether a component can handle this request

- More examples: https://developer.android.google.cn/guide/components/intents-common?hl=en

Intent: Example 3

- But how to use your own intent filters?
- They are just string values:

```
Intent intent = new Intent();
intent.setAction("hello_hello_helloooo");
startActivity(intent);
```

```
<activity
   android:name=".MainActivity2"
   android:label="@string/title_activity_main2"
   android:theme="@style/AppTheme.NoActionBar">
   <intent-filter>
      <action android:name="hello_hello_helloooo" />
      <category android:name="android.intent.category.DEFAULT" />
   </intent-filter>
</activity>
```

Broadcasts

Mechanism and Example

Creating BroadcastReceivers

- Like activities, a BR can be registered dynamically via code or statically using the manifest file.

- In the next few slides, we aim to create an activity that let the user monitor the battery level of his phone.
  - A toggle button can switch on/off the monitoring.
  - Dynamical method will be used.

Step 1: UI

```
<ToggleButton
   android:id="@+id/monitorBtn"
   ......
   android:textOn="Monitoring"
   android:textOff="Zzz" />


<TextView
   android:id="@+id/BatteryLevelText"

   ......
   android:text="Monitoring is off."
   android:textSize="50dp" />
```

Step 2: Activity

```
public class MonitorMain extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ToggleButton monitorBtn = findViewById(R.id.monitorBtn);
        final TextView batteryLvl = findViewById(R.id.BatteryLevelText);
        ...
```

Step 3: BroadcastReceiver

- Still inside `onCreate()`

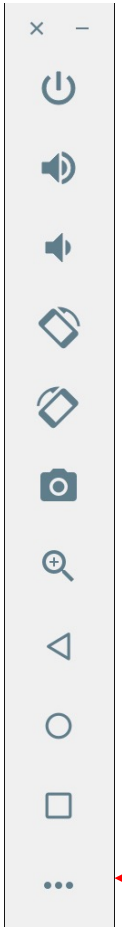
```
final BroadcastReceiver receiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        if (intent.getAction().equals(intent.ACTION_BATTERY_CHANGED)) {
            int current = intent.getExtras().getInt("level");
            int max = intent.getExtras().getInt("scale");
            int value = current * 100 / max;
            batteryLvl.setText("Battery Level is: " + value + "%");
        }
    }
};
```

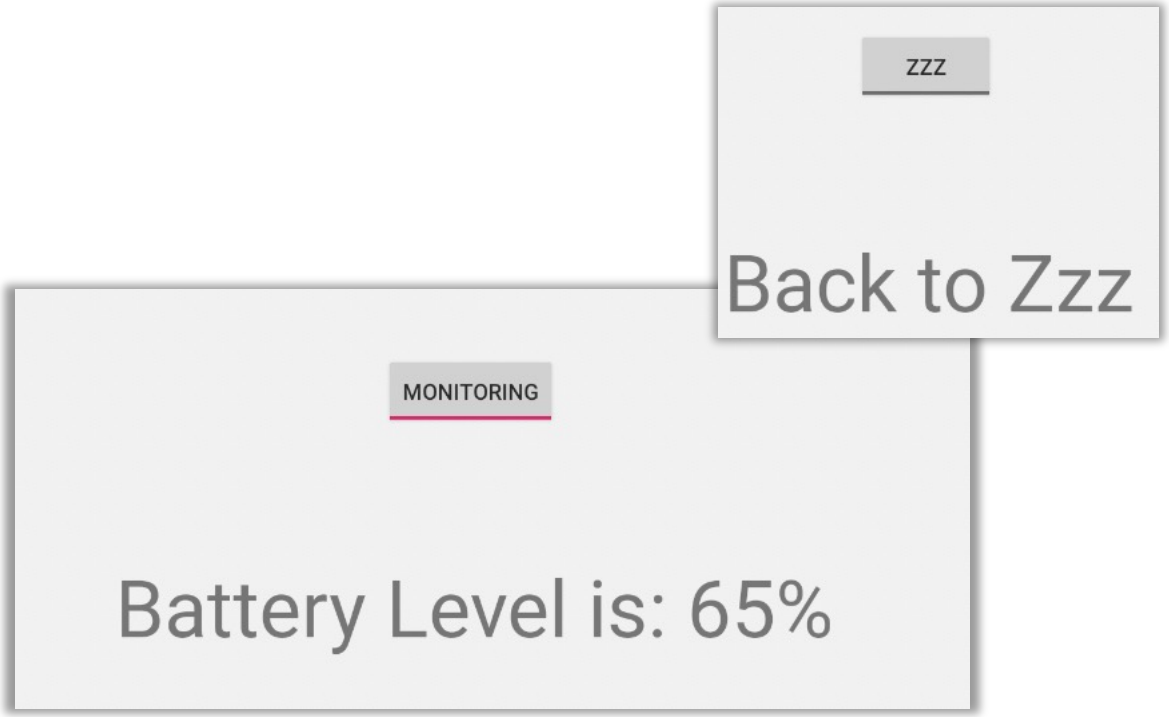
Step 4: Button Event

- Still inside `onCreate()`

```
monitorBtn.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(CompoundButton buttonView,
                                 boolean isChecked) {
        if (isChecked) {
            IntentFilter filter = new IntentFilter(Intent.ACTION_BATTERY_CHANGED);
            registerReceiver(receiver, filter);
        } else {
            unregisterReceiver(receiver);
            batteryLvl.setText("Back to Zzz");
        }
    }
});
```

Running and Testing





Change battery level here in AVD

Create Your Own Broadcasts

- The `Context` class provide two methods:

```
public abstract void sendBroadcast (Intent intent,
                String receiverPermission)
```

```
public abstract void sendOrderedBroadcast (Intent intent,
                String receiverPermission)
```

- What is the difference? Find it out by yourself
  - https://developer.android.google.cn/guide/components/broadcasts?hl=en

Lab Tasks:

- Read about the **data** check in the intent-filter block.

- ~~Find out how to re-create the battery monitor using the static way, using XMLs.~~
  - You can't do that: https://developer.android.com/reference/android/content/Intent.html#ACTION_BATTERY_CHANGED
- Write an app that warns the user when the battery level is too low.
  - find out how to automatically start this app when the system starts. (After learning lecture 10)

- 这里要改了，比如你想创建一个receiver，用来接收battery level change. 如果你是通过manifest里面的 intent-filter来尝试接收，收不到的。
- 又比如，你想自己创建一个 action 匹配 receiver, 用manifest文件注册这个intent-filter也不行，必须手动地创建intentfilter，通过java代码的方式来注册。
- 但是，bootup broadcast却没有以上的问题，这个机制是怎么区别的呢？