

PENGEMBANGAN APLIKASI PERANGKAT BERGERAK (MOBILE)

Intent

K Candra Brata



andra.course@gmail.com



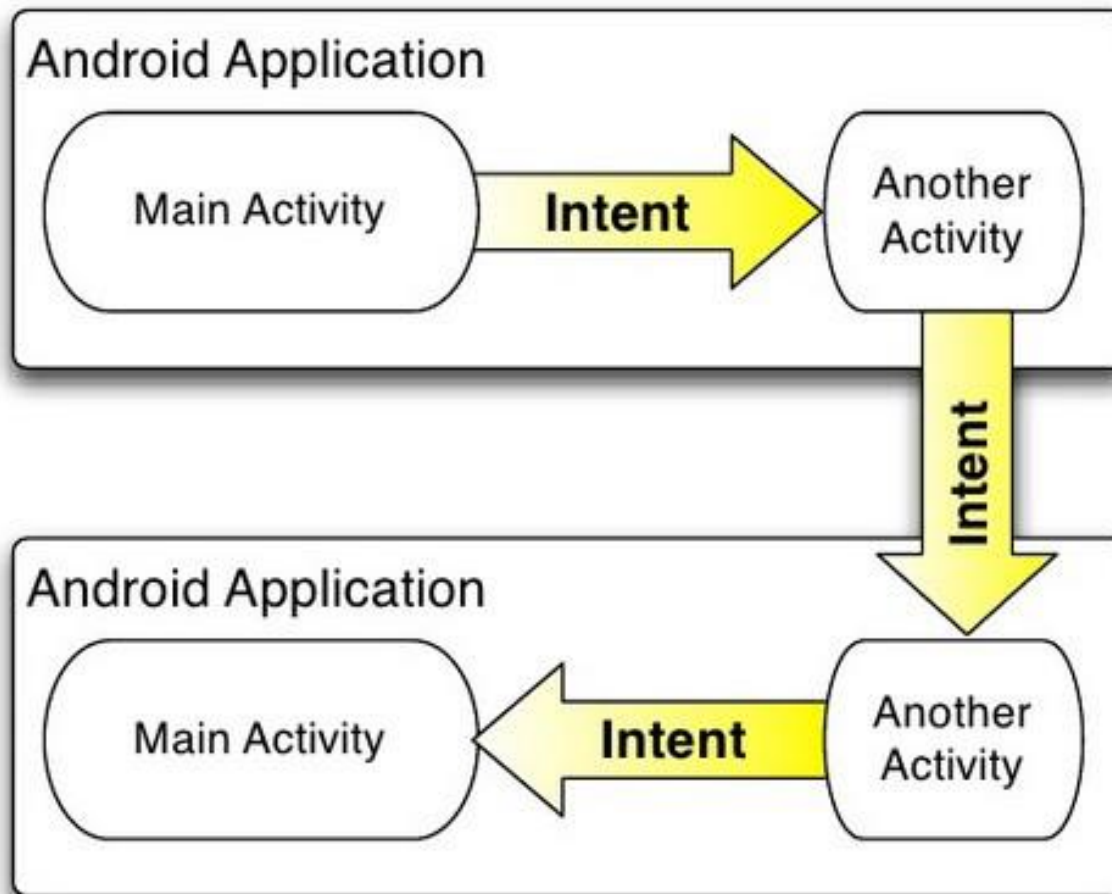
Intent

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

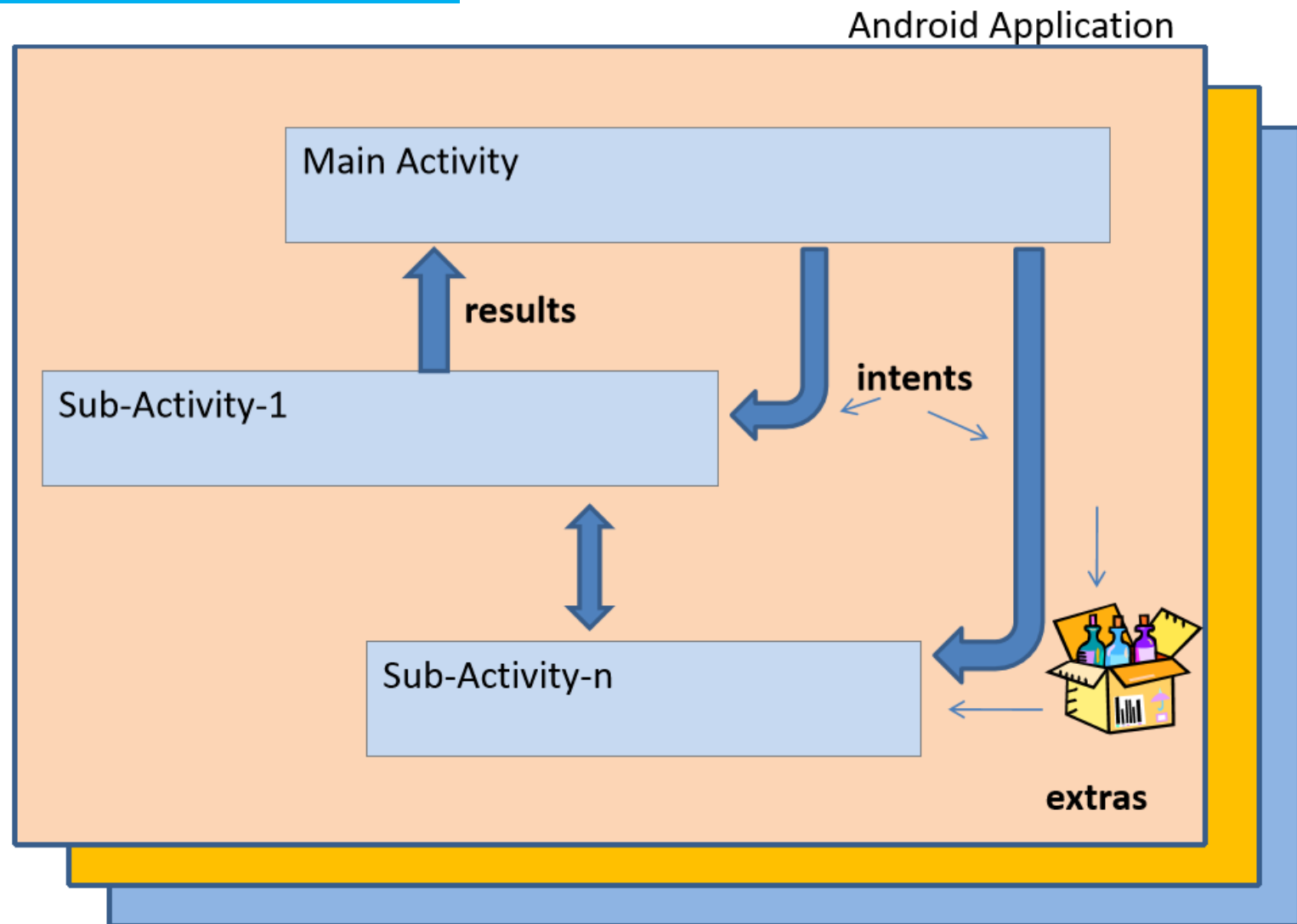
Definition

- ❑ An Android application could include any number of activities.
- ❑ Activities are independent of each other; however they usually cooperate exchanging data and actions.
- ❑ Typically, one of the activities is designated as the first one (main) that should be presented to the user when the application is launched.
- ❑ *Moving from **one activity** to **another** is accomplished by asking the current activity to execute an **intent**.*
- ❑ Activities interact with each other in an asynchronous mode.

Intent



Intent



Intent

- Intents are invoked using the following options

<i>startActivity (intent)</i>	launches an <i>Activity</i>
<i>sendBroadcast (intent)</i>	sends an intent to any interested <i>BroadcastReceiver</i> components
<i>startService(intent)</i> or <i>bindService(intent, ...)</i>	communicate with a background Service.

Intent

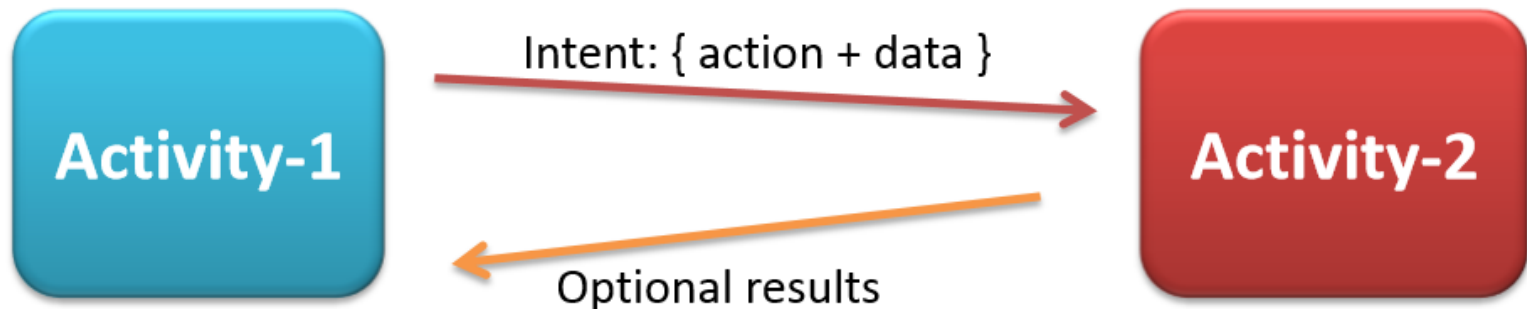
Intent dibagi 2 yaitu:

- ❑ **Implicit intent** adalah intent yang memanggil fungsi activity yang sudah ada di fungsi internal android (Built-in) seperti Dial Number, Open Browser , Gallery, Music Player dan lainnya.
- ❑ **Explicit Intent** yang memanggil Activity lain yang masih dalam 1 project ataupun beda project.

Intent

The main arguments of an Intent are

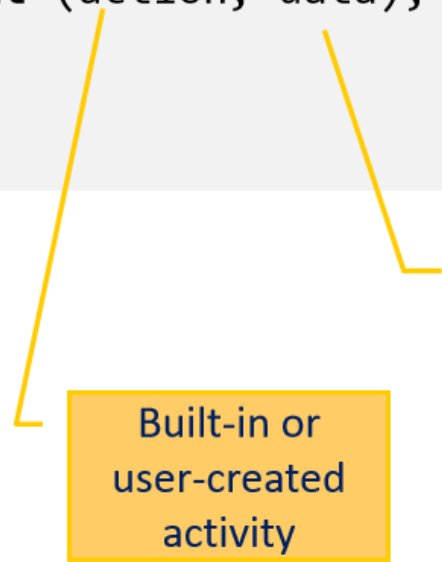
1. **Action**, The built-in action to be performed, such as `ACTION_VIEW`, `ACTION_EDIT`, `ACTION_MAIN`, ... or user-created-activity.
2. **Data**, The primary data to operate on, such as a phone number to be called (expressed as a **Uri**).



Intent

Typically an intent is called as follows:

```
Intent myActivity = new Intent (action, data);  
startActivity (myActivity);
```



Built-in or
user-created
activity

Primary data (as an URI)
tel://
http://
sendto://

Implicit Intent

Built-in Standard Actions

List of standard actions that Intents can use for launching activities (usually through *startActivity(Intent)*).

ACTION_MAIN

ACTION_VIEW

ACTION_ATTACH_DATA

ACTION_EDIT

ACTION_PICK

ACTION_CHOOSER

ACTION_GET_CONTENT

ACTION_DIAL

ACTION_CALL

ACTION_SEND

ACTION_SENDTO

ACTION_ANSWER

ACTION_INSERT

ACTION_DELETE

ACTION_RUN

ACTION_SYNC

ACTION_PICK_ACTIVITY

ACTION_SEARCH

ACTION_WEB_SEARCH

ACTION_FACTORY_TEST

Implicit Intent

Examples of action/data pairs are:

ACTION_DIAL , **tel:123**

Display the phone dialer with the given number filled in.

ACTION_VIEW , **http://www.google.com**

Show Google page in a browser view. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.

ACTION_EDIT , **content://contacts/people/2**

Edit information about the person whose identifier is "2".

ACTION_VIEW , **content://contacts/people/2**

Used to start an activity to display 2-nd person.

ACTION_VIEW , **content://contacts/people/**

Display a list of people, which the user can browse through. Selecting a particular person to view would result in a new intent

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical">
```

<Button

```
    android:id="@+id/btn1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:onClick="btn1Click"
    android:text="Button 1"></Button>
```

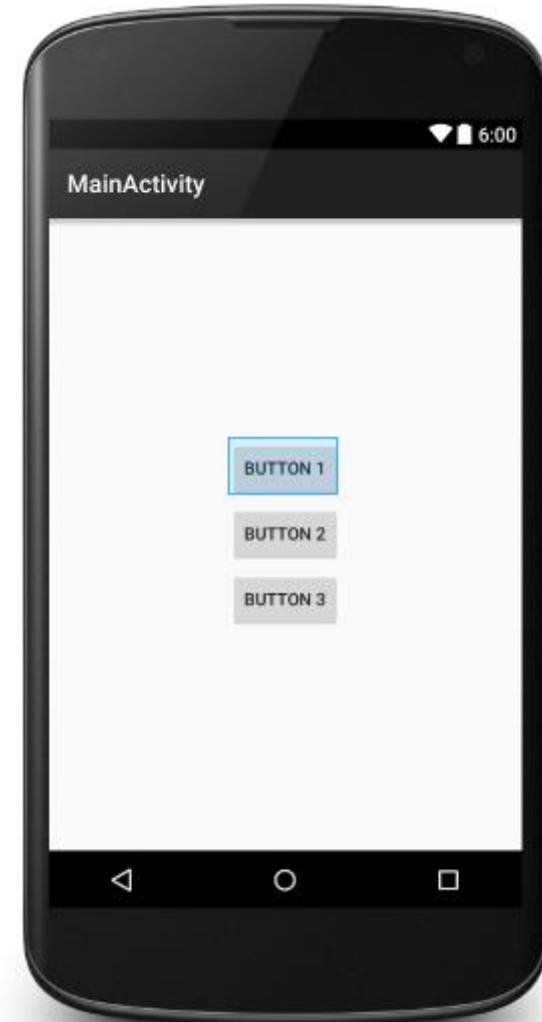
<Button

```
    android:id="@+id/btn2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_margin="5dp"
    android:onClick="btn2Click"
    android:text="Button 2"></Button>
```

<Button

```
    android:id="@+id/btn3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:onClick="btn3Click"
    android:text="Button 3"></Button>
```

```
</LinearLayout>
```



MainActivity.Java

```
public class MainActivity extends AppCompatActivity {

    private Button btn1,btn2,btn3;

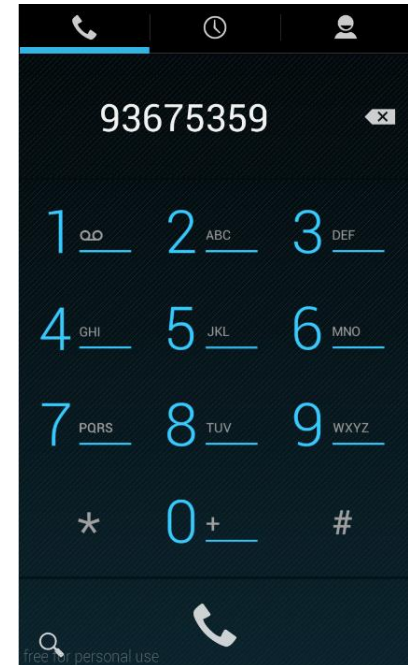
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        btn1 = (Button) findViewById(R.id.btn1);
        btn2 = (Button) findViewById(R.id.btn2);
        btn3 = (Button) findViewById(R.id.btn3);
    }

    public void btn1Click(View view) {
        Intent tlp = new Intent (Intent.ACTION_DIAL, Uri.parse("tel:93675359"));
        startActivity(tlp);  }

    public void btn2Click(View view) {
        Intent setting = new Intent( android.provider.Settings.ACTION_SETTINGS);
        startActivity(setting);
        Toast.makeText(this, "you have Pressed : " + btn2.getText() , Toast.LENGTH_LONG).show();
    }

    public void btn3Click(View view) {
        // DO YOUR METHOD HERE !!
    }
}
```



Intent

Secondary Attributes

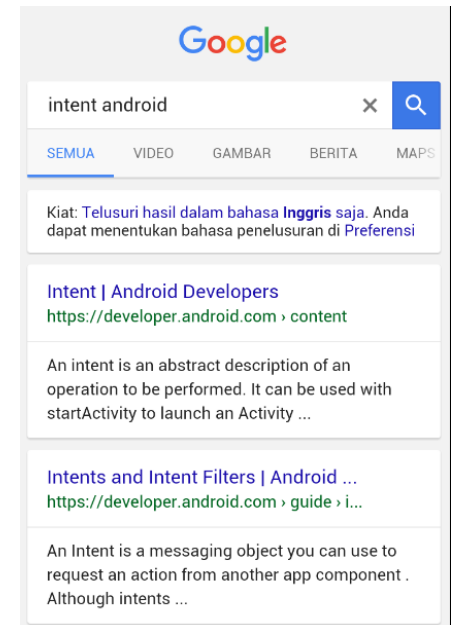
In addition to the primary action/data attributes, there are a number of ***secondary attributes*** that you can also include with an intent, such as:

1. **Category**
2. **Type**
3. **Extras**

Example: Doing a Google search looking for intent android.

Modify Button 3 Click !!!

```
Intent search = new Intent(Intent.ACTION_WEB_SEARCH );  
search.putExtra(SearchManager.QUERY, "intent android");  
startActivity(search);
```



Implicit Intent

Secondary Attributes

```
Intent myIntent= new Intent();  
myIntent.setType("audio/mp3");  
myIntent.setAction(Intent.ACTION_GET_CONTENT);  
startActivity(myIntent);
```

Explicit Intent

Secondary Attributes

Sending Activity

```
Intent intent = new Intent(SendingActivity.this, RecievingActivity.class);  
// pass your values and retrieve them in the other Activity using keyName  
intent.putExtra("keyName", value);  
startActivity(intent);
```

Recieving Activity

```
Bundle extras = intent.getExtras();  
if(extras != null)  
String data = extras.getString("keyName"); // retrieve the data using keyName
```

OR

```
// shortest way to recieve data..  
String data = getIntent().getExtras().getString("keyName");
```


Explicit Intent

Secondary Attributes



Intent

- **Starting Activities and Getting Results**
- The `startActivity(Intent)` method is used to start a new activity, which will be placed at the ***top*** of the activity stack.
- Sometimes you want to get a ***result back*** from the called sub-activity when it ends.

For example, you may start an activity that let the user pick a image from galery; when it ends, it returns the image that was selected.

Intent

- In order to get results back from the called activity we use the method

- **startActivityForResult (Intent, requestCodeID)** 

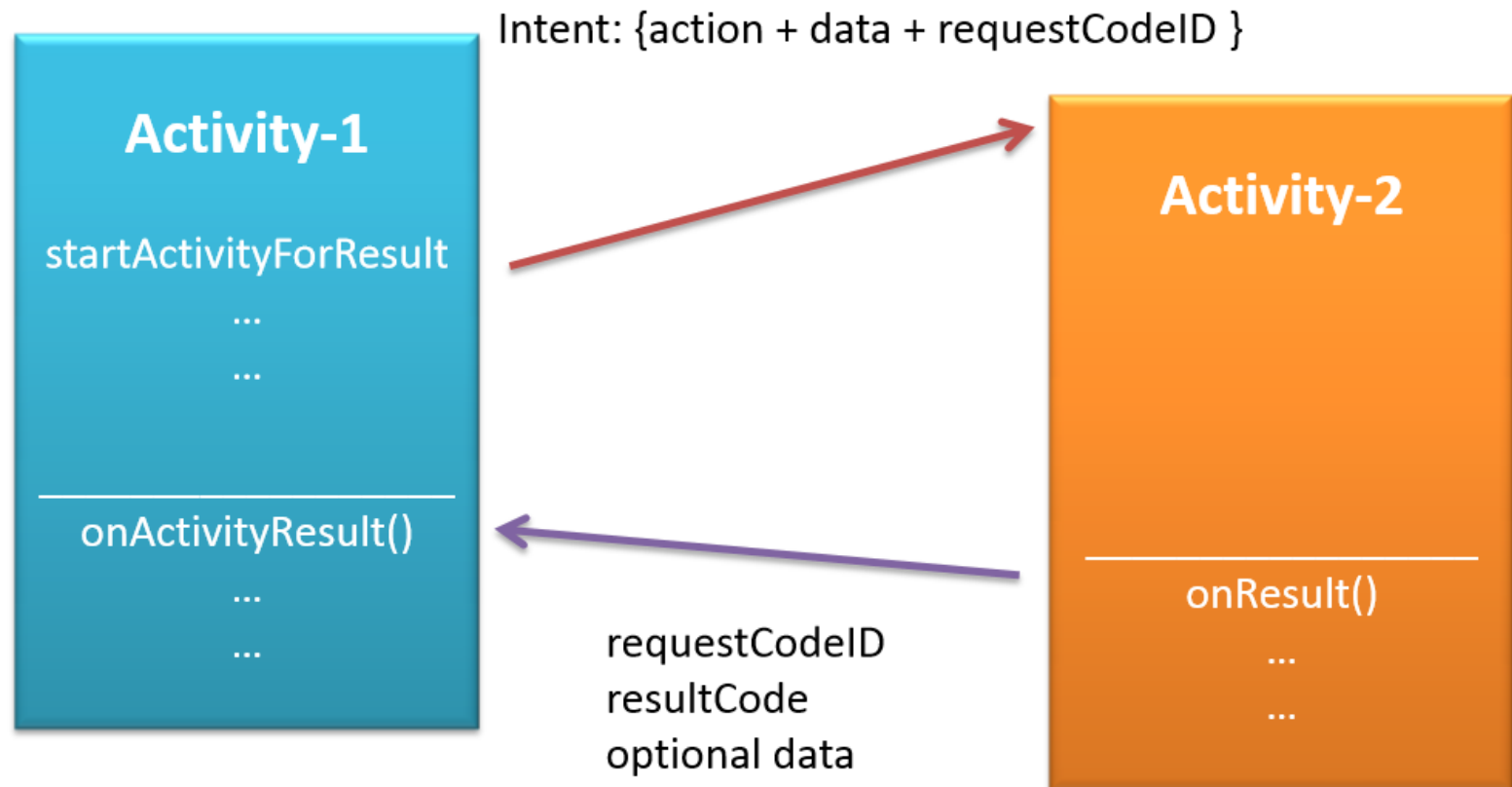
- Where the second (requestCodeID) parameter identifies the call.
- The result sent by the sub-activity could be picked up through the asynchronous method.

- **onActivityResult (requestCodeID, resultCode, Intent)** 

Intent

- Before an activity exits, it can call `setResult(resultCode)` to return a termination signal back to its parent.
- Always supply a result code, which can be the standard results **`Activity.RESULT_CANCELED`**, **`Activity.RESULT_OK`**, or any custom values.
- All of this information can be capture back on the parent's **`onActivityResult (int requestCodeID, int resultCode, Intent data)`** along with the integer identifier it originally supplied.
- If a child activity fails for any reason (such as crashing), the parent activity will receive a result with the code **`RESULT_CANCELED`**.

Intent



Modify Your Code !!!!

**Showing Pictures and Video,
Calling a sub-activity, receiving results.**

```
public void btn3Click(View view) {
    Intent myIntent= new Intent();
    myIntent.setType("video/*, images/*");
    myIntent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(myIntent, 0);

}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent intent) {
    super.onActivityResult(requestCode, resultCode, intent);
    if((requestCode== 0) && (resultCode == Activity.RESULT_OK)) {
        String selectedImage= intent.getDataString();
        Toast.makeText(this, selectedImage, Toast.LENGTH_LONG).show();

        // show a 'nice' screen with the selected image
        Intent myAct3 = new Intent(Intent.ACTION_VIEW, Uri.parse(selectedImage));
        startActivity(myAct3);
    }
}
```

Thanks!

QUESTIONS?



JOIN !!

