

# MODULE 6

# INTENT


# TABLE OF CONTENT

- Message-Based Communication
- What is an Intent?
- Explicit or Implicit Intent
- Launching an Activity
- Activity Result
- Extra Data Associated to the Intent
- Implicit Intent
- Webography

# Message-Based Communication

- ▶ Communication in Android is based on messages
  - Sending
  - Receiving
- ▶ Messages to express an action intended to do
  - Abstract description of operations to perform

# What is an Intent?

- ▶ Message-passing mechanism
  - Notifies the application of occurrence of certain events
  - Represents the action to execute in response to the event
- ▶ Structure describing operations to perform 
  - Action to perform
  - Data to operate on

# What is an Intent?

- ▶ Used to communicate with
  - Activities
  - Services
  - Other applications

# What is an Intent?

## ▶ Services

- Perform long-running operations in the background
  - Even if the user switches to another application
- Do not provide a user interface
  - Services do not return a result to the caller
- Usually, a started service performs a single operation
  - When the operation is done, the service should stop itself
- E.g, to handle network transactions, to play music, to perform file I/O, ...

# What is an Intent?

- ▶ Through android.content.**Intent** class
  - Containing all information needed to realise the intended action
  - *E.g, to start an activity or service*

# Explicit or Implicit Intent

## ► Explicit intent

- The target component is explicitly designated
  - Only limited to be used within an application
  - *E.g, starting of an activity designated by its class name*

## ► Implicit intent

- The target component is not specified
  - Just declare the intent and leave the system to find the most suitable component
    - The component will be chosen by the system at the execution time, not at compilation
  - Usually to activate components of other applications
  - *E.g, dialing a phone number, making a Web search*



# Launching an Activity

- ▶ Create a new **child** activity

- Create new layout (View)
  - In res/layout
- Create new activity class
  - Subclass of Activity class
  - E.g,

```
public class ChildActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_child);  
    }  
}
```

→ Activity class

→ Activity layout

- Declare the activity in the AndroidManifest.XML
  - *If not defined in AndroidManifest, the activity cannot be launched through intent*
  - E.g,

```
<activity android:name=".ChildActivity" android:label="@string/app_name"></activity>
```

# Launching an Activity

- ▶ Using **explicit** intent in **parent** activity
- ▶ First, create an object of the Intent class

- Parameters :
  - Current application context
  - Class name of the activity to launch

◦ E.g, `new Intent (MainActivity.this, ChildActivity.class);`

Current activity

Activity to launch

- ▶ Then, launch the activity using the intent
  - Without testing the result of the launched activity
  - Method : **startActivity**
    - Parameter : an intent
  - E.g, `startActivity(new Intent (MainActivity.this, ChildActivity.class));`

# Activity Result


- ▶ To test the result of a launched activity
- ▶ In **parent** activity,
  - Give a code identifying the child activity to launch
    - Will be used to identify which activity has produced a returned value
  - Method : **startActivityForResult**
    - Parameters :
      - An intent
      - Code (Id) of the activity to launch (int)
  - E.g,

```
startActivityForResult(new Intent(MainActivity.this, ChildActivity.class), 1);
```

Id of the activity

# Activity Result

## ► In **child** activity

- To set result
  - Method : **setResult** 
    - Parameter : result code (int)

## ◦ To finish the activity

- Method : **finish**

↳ Automatically call of the **onActivityResult** method of the parent activity

## ◦ E.g,

```
@Override
public void onClick(View arg0) {
    setResult(1);
    finish();
}}
```

# Activity Result

- ▶ In **parent** activity

- To test the result of the launched activity
- Method : **onActivityResult**

↪ *Automatically called when finish( ) is called in child activity*

- Parameters :
  - Id of the launched activity (int)
  - Result code returned by the child activity (int)
  - The intent (Intent)

# Activity Result

- E.g,

Id of the launched activity

Result code returned by the child activity

```
@Override
public void onActivityResult (int requestCode, int resultCode, Intent intent)
{
    super.onActivityResult(requestCode, resultCode, intent);
    if (requestCode == 1)
    {
        switch (resultCode)
        {
            case 0 : Toast.makeText(MainActivity.this, "Result code = 0", Toast.LENGTH_LONG).show();
                    break;
            case 1 : Toast.makeText(MainActivity.this, "Result code = 1", Toast.LENGTH_LONG).show();
                    break;
        }
    }
}
```

# Extra Data Associated to the Intent

- ▶ Additional data can be embedded into the intent
- ▶ In **parent** activity
  - To add data or message to the intent
  - Method : **putExtra**
    - Parameters :
      - Id of the message (String)
      - Value of the message
    - E.g,

```
Intent intent = new Intent(MainActivity.this, ChildActivity.class);  
intent.putExtra("infoId", "Message sent by the parent");
```

Id

Message value



# Extra Data Associated to the Intent

## ► In **child** activity

- To get data or message embeded into the intent
  - Get the data bundle of the intent
    - Through **getExtras ( )** method
    - Returns an object of the android.os.**Bundle** class
  - Use the right get method on the bundle according to the type of message
    - getString, getInt, getDouble, ...
- E.g,

```
Bundle bundle = this.getIntent().getExtras();  
String messageFromParent = bundle.getString("infoId");
```



Getter for String message



# Implicit Intent

- ▶ The target component is not specified
  - The component will be chosen by the system at the execution time
  - E.g, dialing a phone number or making a Web search
- ▶ Declare the permissions the application needs
  - Add uses-permission in the AndroidManifest.xml
  - E.g,

```
<uses-permission android:name="android.permission.CALL_PHONE"/>  
<uses-permission android:name="android.permission.INTERNET"/>
```


→ To call a phone number

→ To make web search

# Implicit Intent


## ► Create implicit intent

◦ E.g,



```
android.net.Uri uri = Uri.parse("tel:0497123654");  
Intent intent = new Intent(Intent.ACTION_DIAL, uri);  
startActivity(intent);
```

→ To dial a phone number



```
android.net.Uri uri = Uri.parse("http://www.google.com/#q=henallux");  
Intent intent = new Intent(Intent.ACTION_VIEW, uri);  
startActivity(intent);
```

↓  
To visualize the element identified by the URI

# Webography

- ▶ <http://developer.android.com/guide/components/activities.html>
- ▶ <http://developer.android.com/guide/components/intents-filters.html>