SSE3052: Embedded Systems Practice

Jinkyu Jeong
jinkyu@skku.edu
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Android Application

- Activity Lifecycle
- Intents
- Intent Filters
Creating Activity

• `onCreate()`
  – Must implement
  – System calls this when creating activity
  – Initializes essential components of activity
    • **Ex)** `setContentViewById()` to define layout of user interface

• `onPause()`
  – Indication that user is leaving activity
  – Usually, commit any changes that should be persisted
Activity State

- Resumed (= running)
  - Foreground
  - Has user focus
- Paused
  - Another activity is in foreground
  - Still visible (completely alive)
- Stopped
  - Completely obscured by another activity
  - No longer visible (but still alive)
- Killed
public class ExampleActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // The activity is being created.
    }
    @Override
    protected void onStart() {
        super.onStart();
        // The activity is about to become visible.
    }
    @Override
    protected void onResume() {
        super.onResume();
        // The activity has become visible (it is now "resumed").
    }
    @Override
    protected void onPause() {
        super.onPause();
        // Another activity is taking focus (this activity is about to be "paused").
    }
    @Override
    protected void onStop() {
        super.onStop();
        // The activity is no longer visible (it is now "stopped")
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        // The activity is about to be destroyed.
    }
}
Intent

• Facilitates communication between components
• 3 fundamental use cases:
  – Starting an activity
  – Starting a service
  – Delivering a broadcast
Starting Activities

- **Use** `startActivity()` **method**

**Ex)**

```java
Intent intent = new Intent(this, ActivityB.class);
startActivity(intent);
```
Intent Types

• **Explicit** – specify the component name
  – Used to start a component in the same application
  – `Intent intent = new Intent(this, ActivityB.class);`

• **Implicit** – do not name a specific component
  – Instead, declare a general action to perform
  – `Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://csl.skku.edu"));`
Building Intent

• **Constructors:**
  - `Intent(String action, Uri uri)`
  - `Intent(Context packageContext, Class<?> cls)`

• **Information contained in Intent:**
  - **Component name:** Class name of target component
  - **Action:** Generic action to perform
    - e.g. `ACTION_VIEW`, `ACTION_SEND`
  - **Data:** URI that references data to be acted on
  - (Category, Type, and Extras)
Action/Data Pair

• Examples:
  – ACTION_VIEW content://contacts/people/1
  – ACTION_DIAL content://contacts/people/1
  – ACTION_VIEW tel:123
  – ACTION_DIAL tel:123
  – ...

https://developer.android.com/reference/android/content/Intent.html
Data Transfer to Target

• An intent can contain data via a Bundle

• Add data directly with putExtra() methods
  – 2 parameters: key-value pair
  – Key is always String
  – Values can be int, float, String, Bundle, Parcelable, etc.
  – Ex) Intent.putExtra("Val1", "This is for ActivityB");

• Or, create a Bundle object with all extra data, then pass the object with putExtras()
Example: Building Intent

Explicit

Intent downloadIntent = new Intent(this,
DownloadService.class);
downloadIntent.setData(Uri.parse(fileUrl));
startService(downloadIntent);

Implicit

Intent sendIntent = new Intent();
sendIntent.setAction(Intent.ACTION_SEND);
sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
sendIntent.setType("text/plain");
if (sendIntent.resolveActivity(getPackageManager()) != null)
{
    startActivity(sendIntent);
}
Receiving Data

• getIntent() - to retrieve Intent object
  – getAction()
  – getData()
  – getExtras() - to retrieve Bundle object (extra data)

Bundle extras = getIntent().getExtras();
if (extras == null) {
    return;
}
// get data via the key
String value1 = extras.getString("Val1");
String value2 = extras.getString(Intent.EXTRA_TEXT);
Retrieving Result Data

Intent resolution by the Android system

One activity is started

Intent + resultCode provided by called activity

onActivityResult(requestCode, resultCode, intent)

RequestCode provided by Android to identify which activity type was started
Retrieving Result Data

- **Instead of** `startActivity()` **use** `startActivityForResult()` **method**
- **Once sub-activity ends**, `onActivityResult()` **is called**
- **Sub-activity uses** `finish()` **method to go back to caller**
- **Sets a result using** `setResult()` **method.**
Caller Activity

Intent i = new Intent(this, ActivityTwo.class);
i.putExtra("Value1", "This value one for ActivityTwo ");
i.putExtra("Value2", "This value two ActivityTwo");
// set the request code to any code you like,
// you can identify the callback via this code
startActivityForResult(i, REQUEST_CODE);

Callee Activity

public void finish() {
    // Prepare data intent
    Intent data = new Intent();
data.putExtra("returnKey1", "Swinging on a star. ");
data.putExtra("returnKey2", "You could be better then you are. ");
    // Activity finished ok, return the data
    setResult(RESULT_OK, data);
    super.finish();
}
Caller Activity

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {
        if (data.hasExtra("returnKey1")) {

        }
    }
}
Intent Filter

• When system receives an implicit intent, it searches for appropriate components

• Intent filter specifies the types of intents to which a component can respond
  – Defined by its category, action, and data filters.
  – Registered in AndroidManifest.xml

Manifest Example

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.skku.csl.helloworld">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```
Exercise
