



LẬP TRÌNH DI ĐỘNG

Bài 5: Intent và cơ chế trao đổi dữ liệu giữa các thành phần trong android



Nhắc lại bài trước

- Các điều khiển hữu ích: ProgressBar, ProgressDialog, AutoComplete TextView, TimePicker, TimePicker Dialog, DatePicker, DatePicker Dialog, ListView, Spinner, WebView
- Quá trình xây dựng giao diện:
 1. Thiết lập giao diện trong XML
 2. Cái nào không dùng XML được thì viết trong onCreate
 3. Thiết lập dữ liệu cho điều khiển
 4. Viết các hàm xử lý sự kiện cho điều khiển
- Vài kiểu viết mã xử lý sự kiện



Nội dung

1. Giới thiệu về intent
2. Sử dụng intent để trao đổi dữ liệu
3. Intent filter
4. Intent tường minh vs không tường minh
5. Các thành phần của intent
 - Action & Data
 - Category
 - Type
 - Component
 - Extras



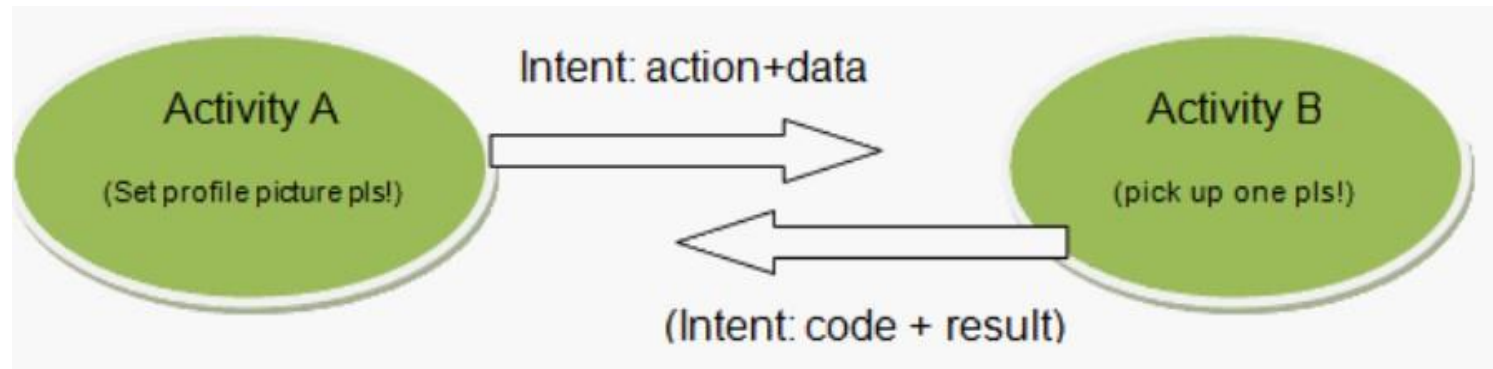
Phần 1

Giới thiệu về intent



Intent & Intent Service

- **Intent** là chuẩn giao tiếp giữa các thành phần trong Android OS (activity, service, provider, receiver)
- **Intent service** là dịch vụ hệ thống, vai trò như người đưa thư: chuyển intent tới thành phần nhận phù hợp nhất (chiều theo địa chỉ ghi trong intent)
- **Intent giống một lá thư**: các thông tin cần thiết được đóng gói bên trong một intent (địa chỉ + nội dung)





Ví dụ sử dụng Intent

```
Intent x = new Intent(this, Login.class);  
x.putExtra("loginname", "abcxyz");  
startActivity(x);
```

**mở activity
Login với dữ
liệu gửi kèm
loginname là
abcxyz**

```
Intent i = new Intent(Intent.ACTION_VIEW);  
i.setData(Uri.parse("http://dantri.com.vn"));  
startActivity(i);
```

mở trang dantri.com.vn

**mở activity quay số với
số 0946771280 điền sẵn**

```
Intent dialIntent = new  
Intent(Intent.ACTION_DIAL,  
startActivity(dialIntent);
```

```
Uri.parse("tel:0946771280"));
```



Dùng Intent và nhận dữ liệu trả về

```
private static int TAKE_PICTURE = 1;
private Uri outputFileUri;
private void TakePhoto() {
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    File file = new File(Environment.getExternalStorageDirectory(), "test.jpg");

    outputFileUri = Uri.fromFile(file);
    intent.putExtra(MediaStore.EXTRA_OUTPUT, outputFileUri);
    startActivityForResult(intent, TAKE_PICTURE);
}
```

**bật ứng dụng camera chụp
ảnh và trả về ảnh chụp**

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

    if (requestCode == TAKE_PICTURE){
        ShowMessage(outputFileUri.toString());
    }

}
```



Intent explicit

- Di chuyển giữa 2 màn hình
 - `Intent i = new Intent(Mactivity.this, Sactivity.class)`
 - `startActivity(i);`
- Gửi dữ liệu đi kèm
 - `i.putExtra(key, value);`
 - Sử dụng sau new intent và trước startActivity
- Màn hình thứ 2 nhận dữ liệu
 - `Intent i = getIntent();`
 - `String noidung=i.getStringExtra(key);`



Intent explicit

- Gửi dữ liệu dạng số
 - `i.putExtra(key, value);`
 - Sử dụng sau new intent và trước startActivity
- Màn hình thứ 2 nhận dữ liệu
 - `Intent i = getIntent();`
 - `int noidung=i.getIntExtra(key, số mặc định);`
- Các trường hợp nhận số mặc định
 - Gửi sai kiểu dữ liệu;
 - Nhận sai key



Intent explicit

- Gửi dữ liệu dạng mảng
 - `String[]`
`array={"android","iOS","PHP","NodeJS","Unity"};`
 - `i.putExtra(key, array);`
- Màn hình thứ 2 nhận dữ liệu
 - `Intent i = getIntent();`
 - `String[] noidung=i.getStringArrayExtra(key);`



Intent explicit

- Gửi dữ liệu đối tượng
 - Đối tượng cần gửi phải implements Serializable
 - `i.putExtra(key, đối tượng gửi);`
- Màn hình thứ 2 nhận dữ liệu
 - `Intent i = getIntent();`
 - Đối tượng `dt=i.getSerializableExtra(key);`



Intent explicit

- Gửi dữ liệu dạng Bundle
 - Dữ liệu được đóng gói trong bundle và gửi kèm intent
 - `Bundle bundle = new Bundle();`
 - `bundle.putExtra(key, dữ liệu số);`
 - `bundle.putExtra(key, dữ liệu chuỗi);`
 - ...
 - `i.putExtra(key, bundle);`
- Màn hình thứ 2 nhận dữ liệu
 - `Intent i = getIntent();`
 - `Bundle b = i.getBundleExtra(key);`
 - `String str = b.getString(key);`



Phần 2

Sử dụng intent để trao đổi dữ liệu

Trao đổi intent giữa các activity



- Dữ liệu được đóng gói vào intent
 - Lựa chọn dữ liệu đưa vào intent phù hợp
 - Có chuẩn trao đổi dữ liệu chung để dễ xử lý
- Sau đó gọi activity phù hợp để xử lý
 - `startActivity(myIntent);`
- Nếu muốn nhận kết quả trả về thì dùng
 - `startActivityForResult(myIntent, CODE);`
 - Trong đó CODE là một số nguyên, dùng để phân biệt kết quả trả về giữa các activity khác nhau
 - Xử lý bằng cách viết lại hàm `onActivityResult`

Trao đổi intent giữa các activity



- Về phía bên activity nhận, lấy intent gửi cho mình bằng `getIntent()`
- Nếu cần trả về kết quả nào đó cho activity trước
 - Thông báo thất bại: `setResult(RESULT_CANCELED);`
 - Thông báo thành công: `setResult(RESULT_OK, x);`
 - Với x là intent mà sẽ trả ngược lại cho activity gọi
 - Cần đưa dữ liệu vào x trước khi setResult
 - Việc đưa dữ liệu sử dụng các hàm putExtra)
 - Chú ý trường hợp dữ liệu phức tạp: putSerializable



```
public void btnNhap(View v) {
    Intent i = new Intent(this, NhapSoActivity.class);
    startActivityForResult(i, 1001);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == 1001) {
        if (resultCode == RESULT_OK) {
            TextView t1 = (TextView) findViewById(R.id.textView1);
            TextView t2 = (TextView) findViewById(R.id.textView2);
            TextView t3 = (TextView) findViewById(R.id.textView3);
            String a = data.getStringExtra("SoA");
            String b = data.getStringExtra("SoB");
            t1.setText("A = " + a);
            t2.setText("B = " + b);
            t3.setText("Tổng = " + (Integer.parseInt(a) + Integer.parseInt(b)));
            Toast.makeText(this, "Trả về thành công", Toast.LENGTH_SHORT).show();
        }
        else
            Toast.makeText(this, "Trả về thất bại", Toast.LENGTH_SHORT).show();
    }
    else
        super.onActivityResult(requestCode, resultCode, data);
}
```

Đoạn mã minh họa việc gọi activity nhập liệu và xử lý kết quả trả về

`onActivityResult`: được tự động gọi khi activity nhập liệu kết thúc



```
public class NhapSoActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_nhap_so);
    }

    public void btnCancel(View v) {
        setResult(RESULT_CANCELED);
        finish();
    }

    public void btnOK(View v) {
        Intent i = new Intent();
        EditText t1 = (EditText) findViewById(R.id.editText1);
        EditText t2 = (EditText) findViewById(R.id.editText2);
        i.putExtra("SoA", t1.getText().toString());
        i.putExtra("SoB", t2.getText().toString());
        setResult(RESULT_OK, i);
        finish();
    }
}
```

Xử lý ở phía activity nhập liệu

Dùng `setResult` để thiết lập dữ liệu trả về cho activity cha



Phần 3

Intent Implicit



Intent filter (bộ lọc intent)

- Activity, Service và Broadcast receiver sử dụng intent filter để thông báo cho hệ thống biết các dạng intent mà nó có thể xử lý
- Phân giải intent (intent resolution): khi nhận được một intent, hệ thống tiến hành chọn activity phù hợp nhất với intent đó theo ưu tiên sau
 - Action trong intent
 - Chuỗi tham số (URI trong phần data)
 - Category của intent



Intent filter (bộ lọc intent)

```
<activity android:name=".Dialer"
           android:label="@string/app_name">
  <intent-filter android:priority="100" >
    <action android:name="android.intent.action.DIAL" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:scheme="tel"/>
  </intent-filter>
</activity>
```

Đoạn mã XML trên đăng kí với hệ thống một activity “lắng nghe” việc người dùng muốn thực hiện cuộc gọi bằng cách chỉ định thuộc tính **action** trong intent-filter





Intent Implicit

- Mở web với ACTION_VIEW
 - `Intent intent= new Intent ();`
 - `intent.setAction(Intent.ACTION_VIEW);`
 - `intent.setData(Uri.parse("http://utt.edu.vn"));`
 - `startActivity(intent);`
- Mở màn hình tin nhắn với ACTION_SENDTO
 - `Intent intent= new Intent ();`
 - `intent.setAction(Intent.ACTION_SENDTO);`
 - `Intent.putExtra("sms_body","Chào bạn...");`
 - `intent.setData(Uri.parse("sms:0969880912"));`
 - `startActivity(intent);`



Intent Implicit

- Mở màn hình gọi với ACTION_CALL
 - `Intent intent= new Intent ();`
 - `intent.setAction(Intent.ACTION_CALL);`
 - `intent.setData(Uri.parse("tel:0969880912"));`
 - `startActivity(intent);`
- Mở camera và đổ hình vào ứng dụng
 - `Intent intent= new Intent`
`(MediaStore.ACTION_IMAGE_CAPTURE);`
 - `startActivityForResult(intent,REQ_CODE);`



Intent Implicit

- Nhận dữ liệu ảnh từ camera gửi về hàm `onActivityResult(requestCode, resultCode, data)`
 - `Bitmap bitmap=data.getExtras().get("data");//mặc định`
 - `imgHinh.setImageBitmap(bitmap);`
 - Ps: `resultCode==RESULT_OK(-1)`



Intent Implicit

- Mở gallery chọn hình và đổ hình vào ứng dụng
 - `Intent intent= new Intent (Intent.ACTION_PICK);`
 - `intent.setType("image/*");`
 - `startActivityForResult(intent,REQ_CODE);`



Intent Implicit

- Nhận dữ liệu ảnh từ gallery trả về hàm `onActivityResult(requestCode, resultCode, data)`
 - `Bitmap bitmap=data.getExtras().get("data");//mặc định`
 - `imgHinh.setImageBitmap(bitmap);`
 - **Ps:** `resultCode==RESULT_OK(-1)`



Intent Runtime Permission API 23

- Xác nhận hỏi quyền trước khi chạy
 - `ActivityCompat.requestPermissions(this, new String[] { Manifest.permission.CAMERA }, REQ_CODE_CAMERA);`
- Kết quả trả về hàm:
 - `onRequestPermissionsResult(requestCode, String[] permissions, int[] grantResults)`
- Xác nhận:
 - `requestCode==REQ_CODE_CAMERA`
 - `grantResults.length>0` (Có câu trả lời)
 - `grantResults[0]==PackageManager.PERMISSION_GRANTED`



```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if(requestCode==REQ_CODE_CAMERA&&grantResults.length>0&&grantResults[0]==PackageManager.PERMISSION_GRANTED) {
        Intent intent=new Intent();
        intent.setAction(MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(intent,REQ_CODE_CAMERA);
    }else if(requestCode==REQ_CODE_CALL&&grantResults.length>0&&grantResults[0]==PackageManager.PERMISSION_GRANTED) {
        Intent intent=new Intent();
        intent.setAction(Intent.ACTION_CALL);
        intent.setData(Uri.parse("tel:0988888888"));
        startActivity(intent);
    }
    else {
        Toast.makeText(context: this, text: "Bạn không cho phép mở camera hoặc call!", Toast.LENGTH_LONG).show();
    }
}
```



Download Image from Internet

- `doInBackground(String... strings)`
 - `URL url=new URL(strings[0]);`
 - `URLConnection urlCon=url.openConnection();`
 - `InputStream is=urlCon.getInputStream();`
 - `bitmap=BitmapFactory.decodeStream(is);`
 - **`return`** `bitmap;`
- `onPostExecute(Bitmap bitmap)`
 - **`imgHinhAnh.setImageBitmap(bitmap);`**



Phần 4

Intent tường minh vs ngầm định



Intent tường minh (explicit)

- Sử dụng thành phần component để chỉ định rõ đối tượng sẽ thực thi
- Sử dụng phương thức:
 - `setComponent(ComponentName)`
 - `setClass(Context, Class)`
 - `setClassName(Context, String)`
 - `setClassName(string, string)`
- Chỉ được dùng để gọi các activity trong cùng một ứng dụng
- Dữ liệu trao đổi nên chuyển vào phần extras



Intent tường minh (explicit)

```
Intent intent = new Intent();
```

```
intent.setClassName("ten_package", "ten_class");
```

```
// chuẩn bị dữ liệu trước khi gửi
```

```
... startActivity(intent);
```

Hoặc `Intent i = new Intent(this, Activity2.class);`

```
// chuẩn bị dữ liệu trước khi gửi
```

```
... startActivity(intent);
```



Intent ngầm định (implicit)

- Dùng các thành phần action, category,...
- Hệ thống tự động xác định đối tượng phù hợp nhất để đáp ứng với Intent đó (theo nguyên tắc “phân giải intent” đã trình bày ở slide 14)
- Dùng để giao tiếp với các dịch vụ hệ thống hoặc dịch vụ do bên thứ ba cung cấp:
 - Gọi activity: `startActivity` / `startActivityForResult`
 - Gọi service: `startService` / `bindService`
 - Gửi broadcast: `sendBroadcast` / `sendOrderedBroadcast` / `setStickyBroadcast`



Intent ngầm định (implicit)

Một số trường hợp sử dụng implicit intent

Định dạng	Action	Mô tả
tel:phone_number	ACTION_VIEW	Mở Dial form (chưa gọi)
tel:phone_number	ACTION_CALL	Thực hiện gọi tới số phone
http://web_address https://web_address	ACTION_VIEW	Mở trình duyệt web với địa chỉ được cấp
"some_words" (string) http://web_address https://web_address	ACTION_WEB_SEARCH	Thực hiện search
sms://	ACTION_SENDTO	Gửi tin nhắn
geo:latitude,longitude geo:latitude,longitude?z=zoom geo:0,0?q=my+street+address geo:0,0?q=business+near+city	ACTION_VIEW	Mở ứng dụng Maps và chỉ tới vị trí được xác định



Các action được định nghĩa sẵn

Built-in Standard Actions	
<u>ACTION_MAIN</u> <u>ACTION_VIEW</u> <u>ACTION_ATTACH_DATA</u> <u>ACTION_EDIT</u> <u>ACTION_PICK</u> <u>ACTION_CHOOSER</u> <u>ACTION_GET_CONTENT</u> <u>ACTION_DIAL</u> <u>ACTION_CALL</u> <u>ACTION_SEND</u>	<u>ACTION_ANSWER</u> <u>ACTION_INSERT</u> <u>ACTION_DELETE</u> <u>ACTION_RUN</u> <u>ACTION_SYNC</u> <u>ACTION_PICK_ACTIVITY</u> <u>ACTION_SEARCH</u> <u>ACTION_WEB_SEARCH</u> <u>ACTION_FACTORY_TEST</u> <u>ACTION_SENDTO</u>
Built-in Standard Broadcast Actions	
<u>ACTION_TIME_TICK</u> <u>ACTION_TIME_CHANGED</u> <u>ACTION_TIMEZONE_CHANGED</u> <u>ACTION_BOOT_COMPLETED</u> <u>ACTION_PACKAGE_ADDED</u> <u>ACTION_PACKAGE_CHANGED</u> <u>ACTION_PACKAGE_REMOVED</u>	<u>ACTION_PACKAGE_RESTARTED</u> <u>ACTION_PACKAGE_DATA_CLEARED</u> <u>ACTION_UID_REMOVED</u> <u>ACTION_BATTERY_CHANGED</u> <u>ACTION_POWER_CONNECTED</u> <u>ACTION_POWER_DISCONNECTED</u> <u>ACTION_SHUTDOWN</u>



Phần 5

Các thành phần của intent



Các thành phần của Intent

Thuộc tính chính	Thuộc tính phụ
action <ul style="list-style-type: none">-tên (string) của action mà Intent sẽ yêu cầu thực hiện-có thể là action được Android định nghĩa sẵn (built-in standard action) hoặc do người lập trình tự định nghĩa	category <ul style="list-style-type: none">-thông tin về nhóm của action type <ul style="list-style-type: none">-định dạng kiểu dữ liệu (chuẩn MIME)-thường được tự động xác định
data <ul style="list-style-type: none">-dữ liệu mà Activity được gọi sẽ xử lý-định dạng Uri (thông qua hàm Uri.parse(data))	component <ul style="list-style-type: none">-chỉ định cụ thể lớp sẽ thực thi Activity-khi được xác định, các thuộc tính khác trở thành không bắt buộc (optional) extras <ul style="list-style-type: none">-chứa tất cả các cặp (key,value) do ứng dụng thêm vào để truyền qua Intent (cấu trúc Bundle)
http://developer.android.com/reference/android/content/Intent.html	



Các thành phần của Intent

- **Component name**: tên class xử lý intent (ví dụ: “com.example.project.app.MyActivity1”)
- **Action**: tên các hành động mà intent yêu cầu thực hiện (ví dụ: action_view, action_call,...)
- **Data**: dữ liệu yêu cầu được xử lý, dữ liệu này thường được biểu diễn dưới dạng URI (ví dụ: "tel:216-555-1234", "<http://dantri.com.vn>",...)
 - Trường hợp dữ liệu phức tạp hoặc không cố định, người ta thường đẩy vào phần extras



Các thành phần của Intent

- **Type**: định dạng kiểu dữ liệu của data (dùng chuẩn MIME), thường được tự xác định bởi hệ thống
- **Category**: bổ sung thông tin cho các action của intent (ví dụ: nếu một activity có thuộc tính category là `CATEGORY_LAUNCHER` nghĩa là activity đó có thể khởi chạy cấp ứng dụng)
- **Extras**: dữ liệu bổ sung nếu vùng Data là chưa đủ, extras sử dụng cấu trúc bundle gồm các cặp (key, value)