

PHÁT TRIỂN ỨNG DỤNG TRÊN THIẾT BỊ DI ĐỘNG

Hồ Văn Tú

Bộ môn Tin học ứng dụng
Khoa CNTT và truyền thông

hvtu@ctu.edu.vn

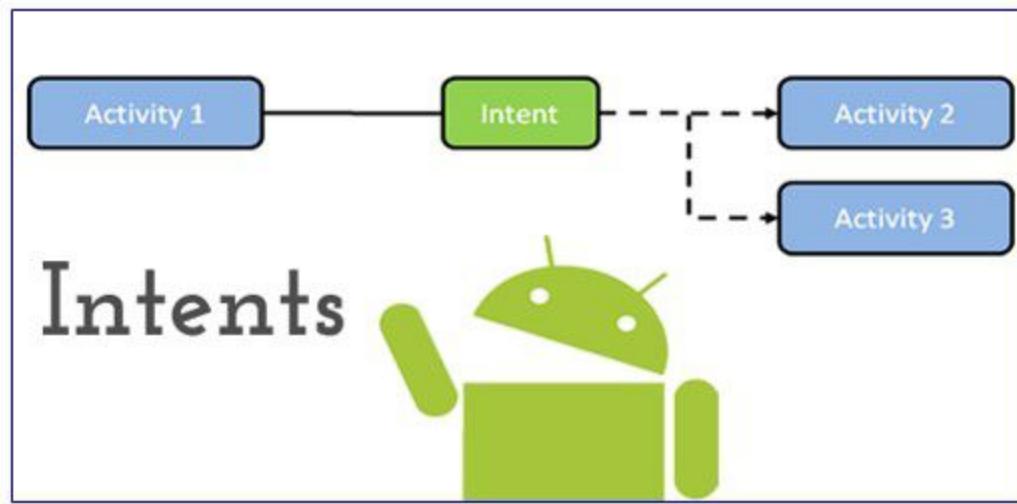
Chương 4

Intents, Services

và Broadcast Receiver

NỘI DUNG

- Intents
 - Giới thiệu
 - Explicit Intents
 - Implicit Intents
 - Intent-Filter
- Services
 - Khái niệm
 - Tạo và sử dụng
- Broadcast Receiver
 - Khái niệm
 - Tạo và sử dụng



Intents_Giới thiệu

- Thuộc lớp android.content.Intent
- Lưu trữ và cung cấp thông tin, dữ liệu giúp xác định công việc cần thực hiện
- Là cơ cấu cho phép truyền thông điệp giữa các thành phần của ứng dụng và giữa các ứng dụng
- Được sử dụng để gửi yêu cầu khởi tạo Activity, Service, hay Broadcast Receiver

VD: Gửi Intent để khởi tạo Activity hiển thị danh sách khách hàng, hay hiển thị hình ảnh, hay mở một trang Web

- Gồm 2 loại: Explicit Intents (Intent tường minh) và Implicit Intents (Intent không tường minh)

Intents_Thuộc tính (1)

- **Action:** là hành động được thực hiện
- **Data:** là dữ liệu sẽ được xử lý trong Action, thường được diễn tả là một Uri (Uniform Resource Identifier)

VD: Built-in Actions và Data kèm theo

ACTION_VIEW content://contacts/people/1 → Hiển thị thông tin trong danh bạ với mã danh 1

ACTION_DIAL content://contacts/people/1 → Gọi đến người với mã danh 1

ACTION_DIAL tel:0908000999 → Gọi đến số 0908000999

Intents_Thuộc tính (2)

- Component: chỉ rõ thành phần sẽ nhận và xử lý Intent. Khi thuộc tính này được xác định thì các thuộc tính khác sẽ trở thành thuộc tính phụ
- Category: bổ sung thêm thông tin cho Action của Intent
- Type: xác định kiểu của Data
- Extras: cho phép truyền dữ liệu theo Intent
- Flags: yêu cầu riêng khi thực thi công việc

VD: FLAG_ACTIVITY_NEW_TASK → thực thi Activity trong một Task mới

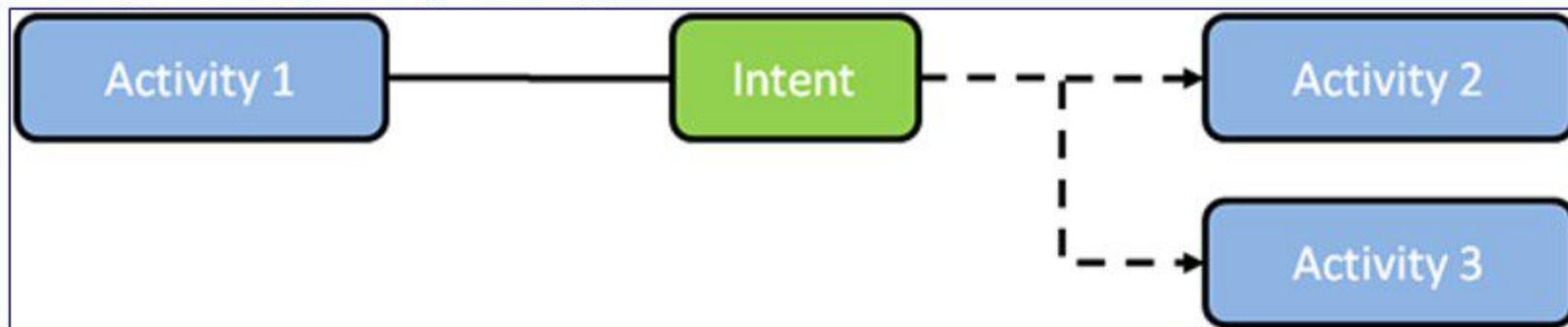
Intents _ Phương thức khởi tạo

- Intent intent = new Intent(Context, Class<?>)
- Intent intent = new Intent(Action)
- Intent intent = new Intent(Action, Uri)

```
Intent intent1 = new Intent(this, DaLat.class);  
Intent intent2 = new Intent(ACTION_VIEW);  
Uri webpage = Uri.parse("https://www.android.com");  
Intent intent3 = new Intent(Intent.ACTION_VIEW, webpage);
```

Explicit Intents (1)

- Intent tường minh
- Xác định được thuộc tính Component
- Thông thường không cần bổ sung thêm các thuộc tính khác như Action, Data
- Thường được sử dụng để thực thi các Activity/ Service trong cùng ứng dụng



```
Intent intent = new Intent(this, Activity2.class);  
startActivity(intent);
```

Explicit Intents (2)

- Tạo và thực thi Intent đơn giản

VD: Khai báo và yêu cầu khởi tạo Activity theo ngữ cảnh

- + Trong ngữ cảnh (Context) của Activity QuanLy, khai báo và yêu cầu khởi tạo Activity DaLat.

```
Intent intent = new Intent(this, DaLat.class);  
startActivity(intent);
```

- + Trong ngữ cảnh (Context) của Fragment GioiThieu, khai báo và yêu cầu khởi tạo Activity KhuyenMai.

```
Intent intent = new Intent(getActivity(), KhuyenMai.class);  
startActivity(intent);
```

Explicit Intents (3)

- Tạo và thực thi Intent có gởi kèm dữ liệu
 - Hỗ trợ trao đổi dữ liệu giữa các Activity
 - Phương thức thông dụng
 - + `bundle.putXXX(key, value)`: thêm giá trị (value) vào gói, từ khóa nhận dạng (key), kiểu dữ liệu XXX (Int, Long, Double, String, ...)
 - + `intent.putExtras(bundle)`: thêm gói cần gởi vào intent.
 - + `intent.getExtras()`: trả về gói đã gởi từ intent.
 - + `bundle.getXXX(key, default_value)`: nhận giá trị trong gói, từ khóa nhận dạng (key) phải giống bên gởi, nếu không có thì trả về giá trị mặc định (default _value).

Explicit Intents (4)

- Tạo và thực thi Intent có gởi kèm dữ liệu thông thường

VD: Activity DaLat gởi dữ liệu, Activity DatTour nhận dữ liệu.

- + Trong Activity DaLat: gởi dữ liệu kèm theo

```
int intSoNgay = 5;
String strDiaDiem = "Đà Lạt";
Bundle bundleGoi = new Bundle();
bundleGoi.putInt("SoNgay", intSoNgay);
bundleGoi.putString("DiaDiem", strDiaDiem);
Intent intent = new Intent(this, DatTour.class);
intent.putExtras(bundleGoi);
startActivity(intent);
```

Explicit Intents (5)

- Tạo và thực thi Intent có gởi kèm dữ liệu thông thường

VD: Activity DaLat gởi dữ liệu, Activity DatTour nhận dữ liệu.

- + Trong Activity DatTour: nhận dữ liệu

```
Intent intent = this.getIntent();
Bundle bundleNhan = intent.getExtras();
int intNgay = bundleNhan.getInt("SoNgay", 0);
String strTour = bundleNhan.getString("DiaDiem", "");
```

Explicit Intents (6)

- Tạo và thực thi Intent có gởi kèm dữ liệu đối tượng: đối tượng phải được implements Serializable

```
public class KhachHang implements Serializable {  
    private int id;  
    private String name;  
    private String email;  
  
    public KhachHang(int id, String name, String email) {  
        this.id = id;  
        this.name = name;  
        this.email = email;  
    }  
    public String getName() {  
        return name;  
    }  
    ...  
}
```

Explicit Intents (7)

- Tạo và thực thi Intent có gởi kèm dữ liệu đối tượng

VD: Activity DatTour gởi dữ liệu dạng đối tượng KhachHang, Activity QuanLy nhận dữ liệu.

- + Trong Activity DatTour: gởi dữ liệu kèm theo

```
KhachHang khGoi = new KhachHang(1, "Tran Xuan Tung",
                                  "txtung@gmail.com");
Bundle bundle = new Bundle();
bundle.putSerializable("KhachDatTour", khGoi);
Intent intent = new Intent(this, DatTour.class);
intent.putExtras(bundle);
startActivity(intent);
```

Explicit Intents (8)

- Tạo và thực thi Intent có gởi kèm dữ liệu đối tượng

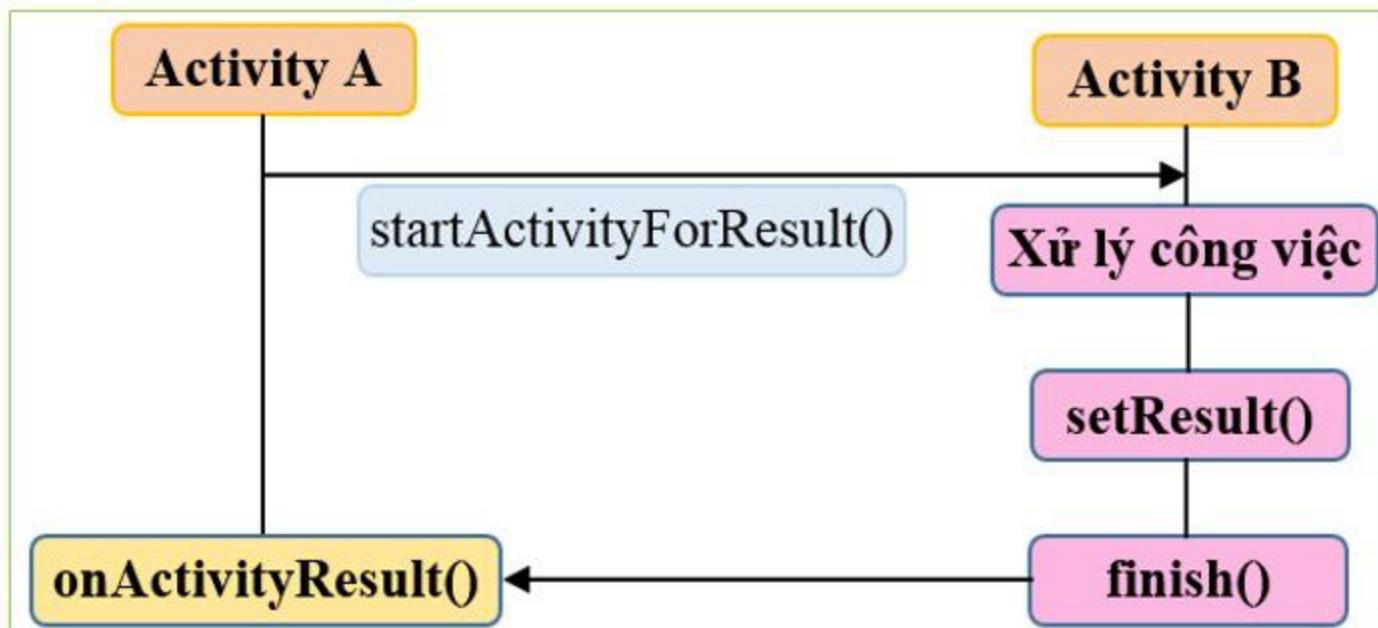
VD: Activity DatTour gởi dữ liệu dạng đối tượng KhachHang, Activity QuanLy nhận dữ liệu.

- + Trong Activity QuanLy: nhận dữ liệu

```
Intent intent = this.getIntent();
Bundle bundle = intent.getExtras();
KhachHang khNhan = (KhachHang)
    bundle.getSerializable("KhachDatTour");
String strName = khNhan.getName();
```

Explicit Intents (9)

- Tạo và thực thi Intent có yêu cầu kết quả trả về
 - Cần kết quả trả về từ Activity con (Sub-Activity)
 - Khi nhận kết quả từ Activity con gửi trả, Activity sẽ xử lý tiếp công việc trong phương thức ghi đè onActivityResult



Explicit Intents (10)

- Tạo và thực thi Intent có yêu cầu kết quả trả về: phương thức thông dụng
 - startActivityForResult(intent, requestCode): thực thi
 - setResult(resultCode): gán kết quả trả về (thường sử dụng hằng **RESULT_OK** hoặc **RESULT_CANCELED**)
 - setResult(resultCode, data): gán Intent lưu dữ liệu kèm theo kết quả trả về
 - onActivityResult(requestCode, resultCode, data): phương thức ghi đè sau khi nhận được kết quả

Explicit Intents (11)

- Tạo và thực thi Intent có yêu cầu kết quả trả về

VD: Activity KhachHang yêu cầu khởi tạo Activity DatHang có nhận kết quả trả về.

- + Trong Activity KhachHang, khai báo khởi tạo Activity DatHang có yêu cầu kết quả trả về.

```
Intent intent = new Intent(this, DatHang.class);
```

```
startActivityForResult(intent, 1);
```

- + Trong Activity DatHang, gửi kết quả trả về sau khi thực hiện công việc.

```
setResult(RESULT_OK); // Nếu xử lý thành công
```

```
finish();
```

Explicit Intents (12)

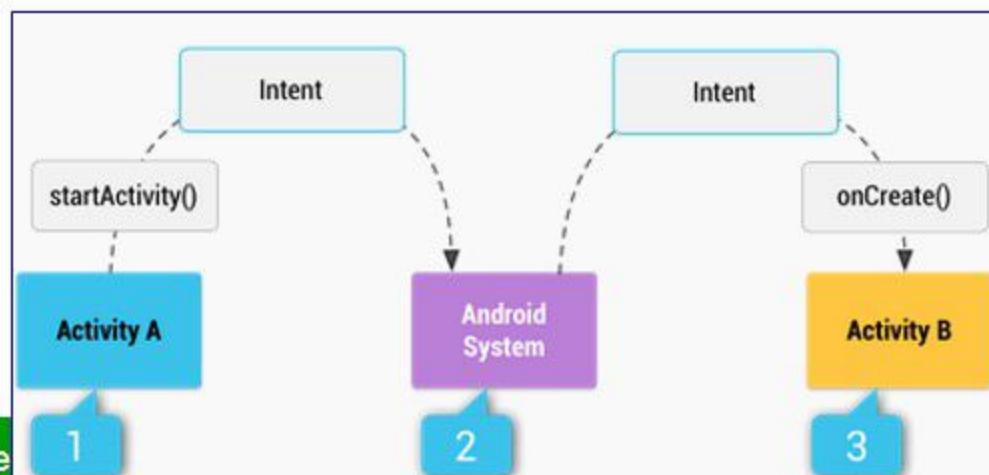
- Tạo và thực thi Intent có yêu cầu kết quả trả về
 - + Trong Activity KhachHang, ghi đè phương thức onActivityResult xử lý kết quả nhận được.

```
@Override  
protected void onActivityResult(int requestCode,  
                                int resultCode, Intent data) {  
    if (requestCode == 1) {  
        if (resultCode == RESULT_OK) {  
            // Mã lệnh nếu thành công  
        }  
        else {  
            // Mã lệnh nếu không thành công  
        }  
    }  
}
```

Implicit Intents (1)

- Intent không tường minh
- Không chỉ rõ Component xử lý, thường sử dụng thông tin khác trong các thuộc tính (như Action, Data)
- Khi Intent được gửi đi, hệ thống sẽ dựa vào những thông tin này và Intent-Filter để quyết định Component thích hợp

VD: ACTION_DIAL tel:0908000999 được giao cho Phone Dialer xử lý



Implicit Intents (2)

- Một số Built-in Action thường sử dụng

ACTION_ANSWER	ACTION_SEND
ACTION_CALL	ACTION_SENDTO
ACTION_DELETE	ACTION_VIEW
ACTION_DIAL	ACTION_MAIN
ACTION_EDIT	

Implicit Intents (3)

- Khai báo và sử dụng

VD1: Thực thi Intent yêu cầu hiển thị Website. Lưu ý trong AndroidManifest phải khai báo quyền truy cập Internet.

```
Uri webpage = Uri.parse("https://www.android.com");
Intent intent = new Intent(Intent.ACTION_VIEW, webpage);
startActivity(intent);
```

VD2: Thực thi Intent yêu cầu xem người thứ nhất trong danh bạ.

```
Uri contact = Uri.parse("content://contacts/people/1");
Intent intent = new Intent(Intent.ACTION_VIEW, contact);
startActivity(intent);
```

VD3: Thực thi Intent yêu cầu gọi điện thoại đến số máy cụ thể.

```
Uri dial = Uri.parse("tel:0908000999");
Intent intent = new Intent(Intent.ACTION_DIAL, dial);
startActivity(intent);
```

Intent-Filter (1)

- Bộ lọc Intent, thuộc lớp android.content.IntentFilter, mô tả khả năng thực hiện của các thành phần
- Activity, Service và BroadCast Receiver sử dụng Intent Filter để thông báo cho hệ thống biết các Implicit Intent nó có thể xử lý
- Thường được khai báo trong tập tin AndroidManifest.xml
- Dựa vào Intent và Intent Filter, hệ thống sẽ quyết định khởi chạy ứng dụng thích hợp nhất để xử lý Intent bắt được
- Nếu có nhiều ứng dụng thích hợp, người dùng sẽ được lựa chọn ứng dụng mình muốn sử dụng

Intent-Filter (2)

- Khai báo Intent-Filter trong tập tin AndroidManifest.xml

VD1: Khai báo Activity thực thi đầu tiên trong ứng dụng ở mức trên (top-level), mỗi ứng dụng chỉ có một Activity thực thi đầu tiên

```
<activity
    android:name=".DuLichDaLat"
    android:label="@string/dulich_dalat"
    android:theme="@style/Theme.AppCompat.Light">
    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category
            android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
```

Intent-Filter (3)

- Khai báo Intent-Filter trong tập tin AndroidManifest.xml

VD2: Khai báo Activity ThongBao có thể nhận dữ liệu được chia sẻ.

```
<activity android:name=".ThongBao" >  
    <intent-filter>  
        <action android:name="android.intent.action.SEND"/>  
        <category android:name="android.intent.category.DEFAULT"/>  
        <data android:mimeType="text/plain" />  
    </intent-filter>
```

Services (1)

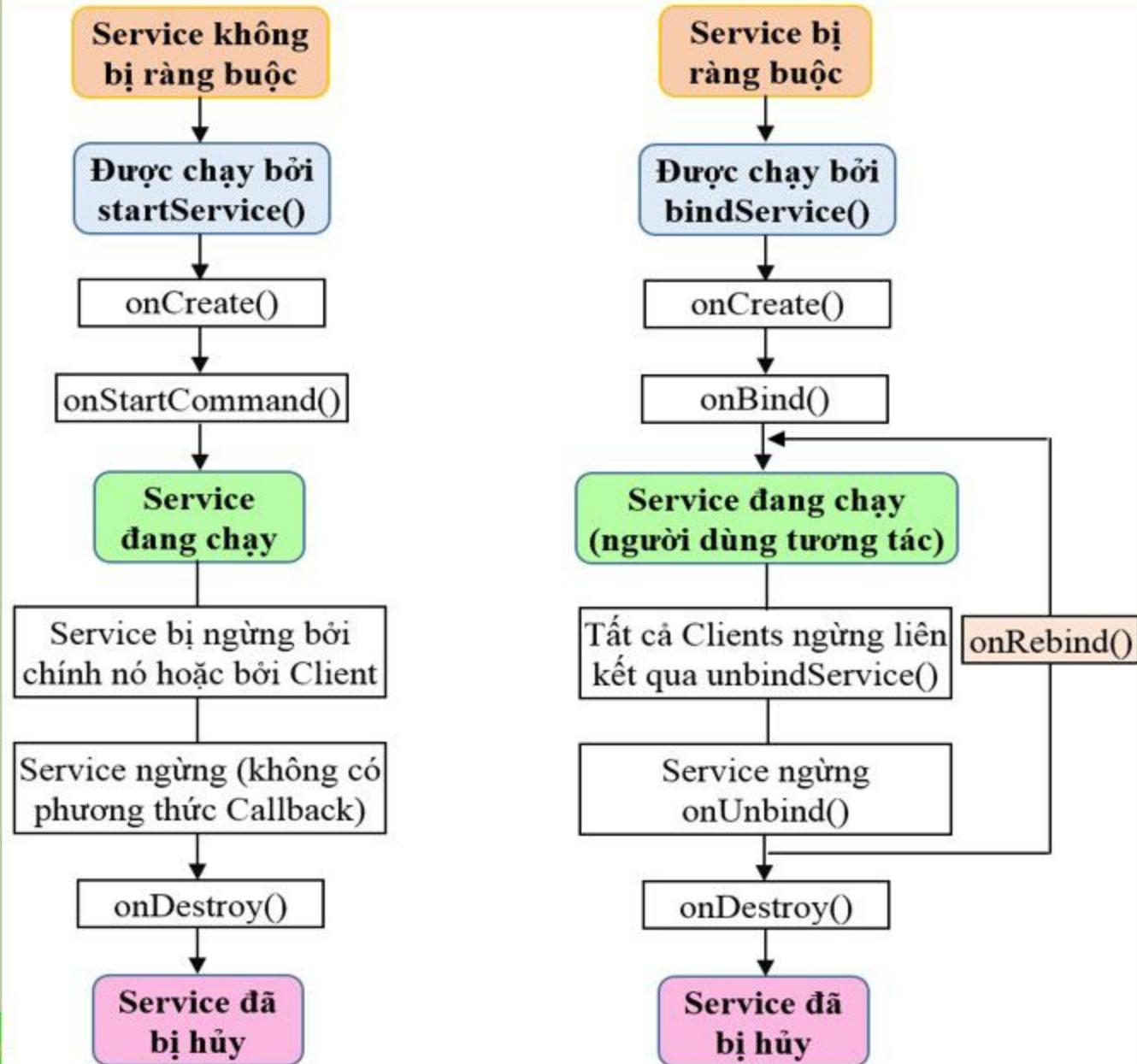
- Thuộc lớp android.app.Service, là thành phần chạy ngầm thực hiện các công việc không cần giao diện
VD: ứng dụng đồng hồ báo thức, ứng dụng nghe nhạc, download hay upload dữ liệu
- Thường được gọi từ thành phần khác, Service cũng chạy tại Thread chính của tiến trình chủ
- Service có vòng đời riêng, có thể chạy cho đến khi người dùng dừng/ hoặc tự dừng/ hoặc hệ thống dừng
- Có thể tiếp tục thực hiện khi thành phần gọi nó đã bị hủy
- Phải có khai báo <service> tại AndroidManifest.xml

Services (2)

- Service được chia làm ba loại chính
 - UnBounded Service (Service không bị ràng buộc/bắt đầu): thực hiện nhiệm vụ dài và lặp lại
 - Bounded Service (Service bị ràng buộc): ràng buộc với thành phần giao diện gọi sử dụng
 - Intent Service: thực hiện nhiệm vụ một lần và tự hủy khi hoàn thành

Services (3)

- Vòng đời



Services (4)

- Tạo và sử dụng UnBound Services
- **Tạo Service không bị ràng buộc:** tạo Service mới và thực hiện
 - + Ghi đè phương thức onCreate(): khởi tạo Service.
 - + Ghi đè phương thức onStartCommand(): bắt đầu Service.
 - + Ghi đè phương thức onDestroy(): dọn rác khi Service bị hủy.
 - + Ghi đè phương thức onBind(): mặc định phải có, không cần thay đổi mã lệnh trong Service không bị ràng buộc.
- **Sử dụng Service không bị ràng buộc**
 - + Phương thức startService(Intent): yêu cầu khởi tạo Service (tương tự Activity).
 - + Phương thức stopService(Intent): yêu cầu hủy bỏ Service.

Services (5)

- Tạo và sử dụng UnBound Services

```
@Override  
public void onCreate() {  
    super.onCreate();  
    mediaPlayer = MediaPlayer.create(getApplicationContext(),  
        R.raw.huongluamiennam);  
    mediaPlayer.setLooping(true);  
}  
  
@Override  
public int onStartCommand(Intent intent, int flags, int startId){  
    mediaPlayer.start();  
    return START_STICKY;  
}  
  
@Override  
public void onDestroy() {  
    mediaPlayer.release();  
    super.onDestroy();  
}
```

Services (6)

- Tạo và sử dụng UnBound Services



```
public void ThucThiservice(View view) {  
    Intent intent = new Intent(this, Media_UnBoundService.class);  
    startService(intent);  
}  
  
public void KetThucService(View view) {  
    Intent intent = new Intent(this, Media_UnBoundService.class);  
    stopService(intent);  
}
```

Services (7)

- Tạo và sử dụng Bound Services
 - **Tạo Service bị ràng buộc:** tạo Service mới và thực hiện
 - + Khai báo lớp MyLocalBinder kế thừa Binder trả về đối tượng Service đang định nghĩa.
 - + Khai báo biến myBinder là đối tượng MyLocalBinder lưu giá trị làm kết quả trả về trong phương thức onBind().
 - + Ghi đè phương thức onCreate(): khởi tạo Service.
 - + Ghi đè phương thức onBind(): gắn kết Service với Clients sử dụng nó, trả về giá trị của biến myBinder.
 - + Ghi đè phương thức onUnBind(): hủy bỏ gắn kết Service với Clients sử dụng nó.
 - + Tạo phương thức public thực hiện công việc theo yêu cầu.
 - + Ghi đè phương thức onDestroy(): dọn rác khi Service bị hủy.

Services (8)

- Tạo và sử dụng Bound Services

- Sử dụng Service bị ràng buộc

- + Khai báo biến myService là đối tượng Service vừa tạo và biến isBound xác định trạng thái gắn kết của Service với Clients.
 - + Khai báo biến myConnection là đối tượng ServiceConnection cho phép kết nối hoặc ngắt kết nối với Service vừa tạo.
 - + Phương thức bindService(Intent): yêu cầu khởi tạo Service bị ràng buộc.
 - + Phương thức unbindService(ServiceConnection): yêu cầu ngắt kết nối với Service.
 - + Ghi đè phương thức onStop(): ngắt kết nối và hủy bỏ gắn kết Service với Clients sử dụng nó.

Services (9)

- Tạo và sử dụng Bound Services

```
public class Factorial_BoundService extends Service {  
    private IBinder myBinder = new MyLocalBinder();  
  
    public class MyLocalBinder extends Binder {  
        public Factorial_BoundService getService() {  
            return Factorial_BoundService.this;  
        }  
    }  
  
    @Override  
    public IBinder onBind(Intent intent) {  
        return myBinder;  
    }  
  
    public long getFactorial(int n) {  
        long ketqua = 1;  
        for(int i=2; i<=n; i++)  
            ketqua = ketqua * i;  
        return ketqua;  
    }  
}
```

Services (10)

- Tạo và sử dụng Bound Services



Services (10)

- Tạo và sử dụng Bound Services

```
public class UsingFactorialService extends AppCompatActivity {  
    private Factorial_BoundService myService;  
    private boolean isBound = false;  
    private ServiceConnection myConnection =  
        new ServiceConnection() {  
            public void onServiceConnected(  
                ComponentName className, IBinder service) {  
                Factorial_BoundService.MyLocalBinder binder =  
                    (Factorial_BoundService.MyLocalBinder) service;  
                myService = binder.getService();  
                isBound = true;  
            }  
            public void onServiceDisconnected(ComponentName arg0) {  
                isBound = false;  
            }  
        };
```

Services (11)

- Tạo và sử dụng Bound Services

```
public void ThucThiservice(View view) {  
    Intent intent = new Intent(this, Factorial_BoundService.class);  
    bindService(intent, myConnection, Context.BIND_AUTO_CREATE);  
}  
  
public void TinhGiaiThua(View view) {  
    int n = 0;  
    if (isBound) {  
        if(edtSoNguyen.getText().length() > 0)  
            n = Integer.parseInt(edtSoNguyen.getText().toString());  
        long ketqua = myService.getFactorial(n);  
        edtGiaitThua.setText("'" + ketqua);  
    }  
    else  
        edtGiaitThua.setText("Service chưa thực thi!");  
}
```

Services (12)

- Tạo và sử dụng Bound Services

```
public void NgatKetNoi() {  
    if (isBound) {  
        unbindService(myConnection);  
        isBound = false;  
    }  
}  
  
public void KetThucService(View view) {  
    NgatKetNoi();  
}
```

Services (13)

- Tạo và sử dụng Intent Services
 - **Tạo Intent Service:** tạo Service (IntentService) mới, không chọn hỗ trợ tạo các phương thức khởi tạo Service và thực hiện
 - + Điều chỉnh mã lệnh cho phương thức onHandleIntent() thực hiện nhiệm vụ của Service.
 - + Ghi đè phương thức onCreate(): khởi tạo Service.
 - + Ghi đè phương thức onStartCommand (): bắt đầu Service.
 - + Ghi đè phương thức onDestroy(): dọn rác khi Service bị hủy.
 - **Sử dụng Intent Service:** sử dụng phương thức startService(Intent): yêu cầu khởi tạo Service (tương tự Activity và UnBound Service).

Services (14)

- Tạo và sử dụng Intent Services

```
public class RandomNumber_IntentService extends IntentService {  
  
    @Override  
    protected void onHandleIntent(Intent intent) {  
        if (intent != null) {  
            Random myRandom = new Random();  
            int n = myRandom.nextInt(100);  
            Log.i("Intent Service", "Số ngẫu nhiên: " + n);  
        }  
    }  
  
    @Override  
    public int onStartCommand(@Nullable Intent intent,  
                             int flags, int startId) {  
        return super.onStartCommand(intent, flags, startId);  
    }  
}
```

Services (15)

- Tạo và sử dụng Intent Services

THỰC THI SERVICE

ĐÓNG ACTIVITY

```
public class UsingIntentService extends AppCompatActivity {

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

    public void ThucThiService(View view) {
        Intent intent = new Intent(this,
                RandomNumber_IntentService.class);
        startService(intent);
    }
}
```

Notifications (1)

- Đưa ra các thông báo mà không làm cho các Activity phải ngừng hoạt động

Ví dụ: thông báo có tin nhắn, Email mới; cập nhật ứng dụng; ...

- Tạo Notification: NotificationCompat.Builder thuộc lớp android.support.v4.app.NotificationCompat
- Phát Notification: NotificationManager thuộc lớp android.app.NotificationManager

Notifications (2)

- Tạo Notification
 - Tạo đối tượng `NotificationCompat.Builder`
 - Gán các thuộc tính cơ bản: `setTicker`, `setIcon`, `setTitle`, `setContent`, ...
 - Thêm Intent thực hiện (nếu có): `setContentIntent`
- Phát Notification
 - Tạo đối tượng `NotificationManager`
 - Thực hiện phát Notification

Notifications (3)

- VD

```
NotificationCompat.Builder mBuilder =  
    new NotificationCompat.Builder(context,  
        "LUCKY_NUMBER");  
mBuilder.setSmallIcon(R.mipmap.luckyseven)  
    .setContentTitle("Lucky seven")  
    .setContentText("Good luck to you!");  
NotificationManager mNotificationManager =  
    (NotificationManager) context.getSystemService(  
        Context.NOTIFICATION_SERVICE);  
mNotificationManager.notify(1, mBuilder.build());
```

Broadcast Receiver (1)

- Thuộc lớp android.content.BroadcastReceiver, có chức năng lắng nghe, nhận (Receiver), và hồi đáp các thông báo quảng bá trên toàn hệ thống
- Thường được sử dụng ở hai dạng
 - Khởi nguồn từ hệ thống: như thông báo màn hình đã tắt, pin yếu, hoặc một bức ảnh được chụp
 - Khởi nguồn từ ứng dụng (cục bộ): ứng dụng cũng có thể khởi tạo quảng bá, như cho biết một phần dữ liệu đã được tải xuống hay thông báo về sự kiện sắp tới

Broadcast Receiver (2)

- Có thể hoạt động khi ứng dụng bị tắt, không giao diện
- Tạo và sử dụng Broadcast Receiver
 - Tạo lớp con kế thừa lớp BroadcastReceiver và ghi đè phương thức onReceive()
 - Sử dụng
 - *Broadcast Receiver hệ thống*: đăng ký quyền truy cập và đăng ký lắng nghe sự kiện
 - *Broadcast Receiver cục bộ*: gọi thực thi với sendBroadcast

Broadcast Receiver (3)

- Tạo và sử dụng Broadcast Receiver hệ thống

Sự kiện thông dụng cho Broadcast Receiver hệ thống

Sự kiện	Mô tả
android.intent.action.BATTERY_CHANGED	Trạng thái nạp, mức độ, và thông tin khác về pin
android.intent.action.BATTERY_LOW	Trạng thái pin yếu
android.intent.action.BATTERY_OKAY	Trạng thái pin tốt
android.intent.action.BOOT_COMPLETED	Trạng thái hoàn thành khởi động
android.intent.action.DATE_CHANGED	Trạng thái ngày thay đổi
android.intent.action.PHONE_STATE	Trạng thái cuộc gọi
android.net.conn.CONNECTIVITY_CHANGE	Trạng thái kết nối mạng
android.provider.Telephony.SMS_RECEIVED	Trạng thái có tin nhắn mới

Broadcast Receiver (4)

- Tạo và sử dụng Broadcast Receiver hệ thống

VD: Tạo Broadcast Receiver cho phép nhận sự kiện khi thiết bị hoàn thành khởi động sẽ tự động thực thi Media_UnBoundService (Ví dụ 4.18).

- + **Bước 1:** tạo Broadcast Receiver với tên ReceiverBootCompleted, thêm mã lệnh cho phép thực thi Media_UnBoundService.

```
public class ReceiverBootCompleted extends BroadcastReceiver{  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        Intent myIntent = new Intent(context,  
                                     Media_UnBoundService.class);  
        context.startService(myIntent);  
    }  
}
```

Broadcast Receiver (5)

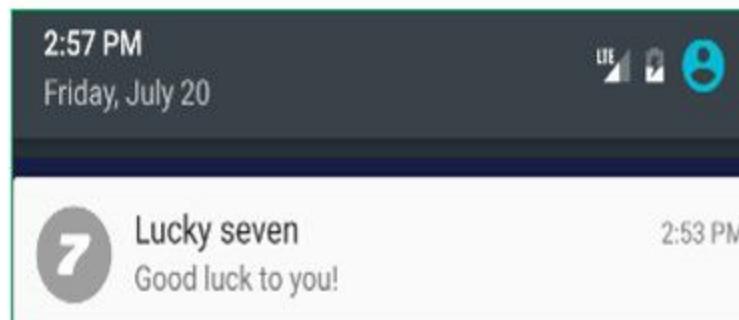
- Tạo và sử dụng Broadcast Receiver hệ thống
 - + **Bước 2:** trong AndroidManifest, đăng ký quyền nhận tín hiệu thiết bị hoàn thành khởi động và đăng ký Intent-Filter cho Broadcast Receiver. Để tránh các ứng dụng khác sử dụng, chọn exported="false".

```
<uses-permission  
    android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>  
  
<receiver  
    android:name=".chapter4.ReceiverBootCompleted"  
    android:enabled="true"  
    android:exported="false">  
    <intent-filter>  
        <action android:name="android.intent.action.BOOT_COMPLETED"/>  
    </intent-filter>  
</receiver>
```

Broadcast Receiver (6)

- Tạo và sử dụng Broadcast Receiver cục bộ

VD: Tạo và sử dụng Broadcast Receiver “Số may mắn”: khi số nhập vào là 7 thì yêu cầu thực hiện gởi Broadcast Receiver cục bộ.



Broadcast Receiver (7)

- Tạo và sử dụng Broadcast Receiver cục bộ

- + **Bước 1:** tạo Broadcast Receiver với tên SoMayMan, thêm mã lệnh tạo và hiển thị Notification.

```
public class SoMayMan extends BroadcastReceiver {  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        NotificationCompat.Builder mBuilder =  
            new NotificationCompat.Builder(context,  
                "LUCKY_NUMBER");  
        mBuilder.setSmallIcon(R.mipmap.luckyseven)  
            .setContentTitle("Lucky seven")  
            .setContentText("Good luck to you!");  
        NotificationManager mNotificationManager =  
            (NotificationManager) context.getSystemService(  
                Context.NOTIFICATION_SERVICE);  
        mNotificationManager.notify(1, mBuilder.build());  
    }  
}
```

Broadcast Receiver (8)

- Tạo và sử dụng Broadcast Receiver cục bộ
 - + **Bước 2:** trong Activity UsingFactorialService, thêm mã lệnh trong phương thức TinhGiaiThua, nếu số cần tính giai thừa là 7 thì thực thi Broadcast Receiver.

```
if(n == 7) {  
    Intent myIntent = new Intent(this, SoMayMan.class);  
    sendBroadcast(myIntent);  
}
```