LẬP TRÌNH ANDROID NÂNG CAO

BÀI 5 :

- GOOGLE MAP

LẬP TRÌNH ANDROID NÂNG CAO

PHẦN I :
SERVICE

6.3.1 SERVICES – KHÁI NIỆM

- **Service (dịch vụ)** là 1 trong bốn component cơ bản của android (services, activities, content providers, broadcast receivers). Service chạy ẩn ở dưới để thực hiện các thao tác dài hạn mà không cần tương tác với người dùng.
- Ví dụ: service có thể mở một bản nhạc trong khi người dùng đang sử dụng ứng dụng khác, hoặc là tải dữ liệu thông qua mạng mà không ảnh hưởng gì đến các hoạt động của người dùng. Một service sẽ có 2 loại:
  - Started service
  - Bound service

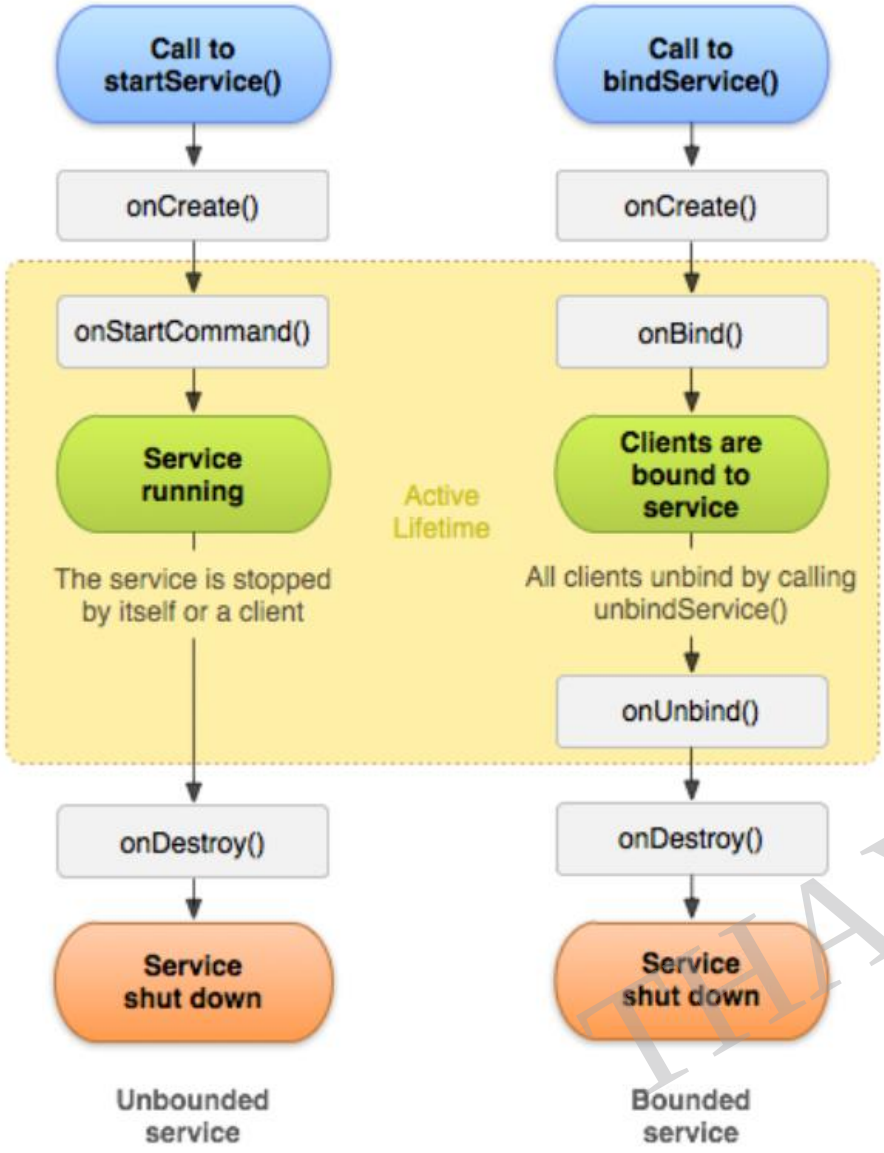
6.3.1.2 Các loại Service

1. **Started Service (Unbound service)**
   - Thực hiện một hành động đơn lẻ và không trả về kết quả cho đối tượng gọi.
   - Hoạt động giống như activities.
   - Gọi hàm startService() để bắt đầu 1 service
   - Ví dụ, nó có thể thực hiện tải hoặc upload file thông qua mạng và khi thực hiện xong thì nó tự đông dừng lại.

6.3.1.2 Các loại Service

2. **Bound Service**:
   - Một service được ràng buộc cho phép components tương tác với service, gửi yêu cầu, nhận kết quả trả về.
   - Thông qua gọi hàm bindService().
   - Service ràng buộc thường là kiểu giao diện client-server.
   - Một service ràng buộc có thể chạy với nhiều components ràng buộc đến nó, khi tất cả components không còn ràng buộc nữa thì service sẽ bị hệ thống hủy

6.3.1.3. VÒNG ĐỜI CỦA SERVICE



6.3.1.4 Các phương thức của Service

- onStartCommand(): sẽ được gọi khi một component khác hoặc một activity yêu cầu bắt đầu service bằng cách gọi hàm startService().
- onBind(): hệ thống sẽ gọi hàm này khi một component muốn ràng buộc với service bằng cách gọi hàm bindService().
- onUnbind(): Khi service bị ngắt kết nối
- onRebind(): Có 1 kết nối lại với service
- onCreate(): hệ thống gọi hàm này khi lần đầu tiên service chạy trước khi chạy hàm onStartCommand() và onBind().
- onDestroy(): khi service không được dùng nữa và ta cần giải phóng tài nguyên liên quan đến service ở đây

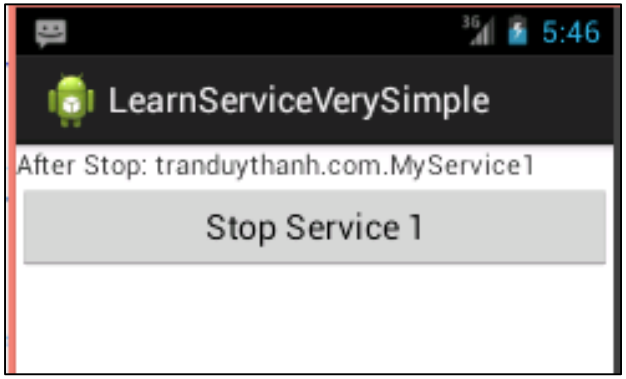
6.3.1.5 KHAI BÁO SERVICE TRONG MANIFEST

- Sử dụng thẻ <service android:name="tên service"/>

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="cis493.demos" android:versionCode="1" android:versionName="1.0.0">
    <uses-sdk android:minSdkVersion="4"></uses-sdk>

    <application android:icon="@drawable/icon" android:label="@string/app_name">

        <activity android:name=".MyServiceDriver2">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <service android:name="MyService2" />

        <receiver android:name="MyBroadcastReceiver">
            <intent-filter>
                <action android:name = "matos.action.GOSERVICE2" />
            </intent-filter>
        </receiver>

    </application>
</manifest>
```

6.3.1.6 Ví dụ 1 – Simple service

- Chương trình chạy 1 service và sevice này sẽ in ra ngoài cửa sổ DDMS Logcat. Service chạy cho đến khi bấm vào nút stop thì service mới dừng lại




```
<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <activity..
    <service android:name="tranduythanh.com.MyService1"></service>
</application>
```

6.3.1.6 Ví dụ 1

```
public class MainActivity extends Activity {
    Button btnstop;
    Intent intentMyService;
    TextView txtmsg;
    ComponentName service;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtmsg=(TextView) findViewById(R.id.textView1);
        intentMyService=new Intent(this, MyService1.class);
        service=startService(intentMyService);
        txtmsg.setText("Begin: "+service.getClassName());
        btnstop=(Button) findViewById(R.id.btnstop);
        btnstop.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                stopService(intentMyService);
                txtmsg.setText("After Stop: "+
                        service.getClassName() );
            }
        });
    }
}
```

6.3.1.6 Ví dụ 1

```
public class MyService1 extends Service {
    public IBinder onBind(Intent arg0) {
        return null;
    }
    public void onCreate() {
        super.onCreate();
        Log.i("<MyServices1 - onCreate>", " I am alive 1");
    }
    public int onStartCommand
        (Intent intent, int flags, int startId) {
        Log.i("<MyServices1 - onStartCommand>",
                " I am onStartCommand");
        return super.onStartCommand
                (intent, flags, startId);
    }
    public void onStart(Intent intent, int startId) {
        Log.i("<MyServices1 - onStart>", " I am onStart");
    }
    public void onDestroy() {
        super.onDestroy();
        Log.i("<MyServices1 - onDestroy>", " I am onDestroy");
    }
}
```

6.3.1.6 Ví dụ 2 – Service & Broadcast Receiver

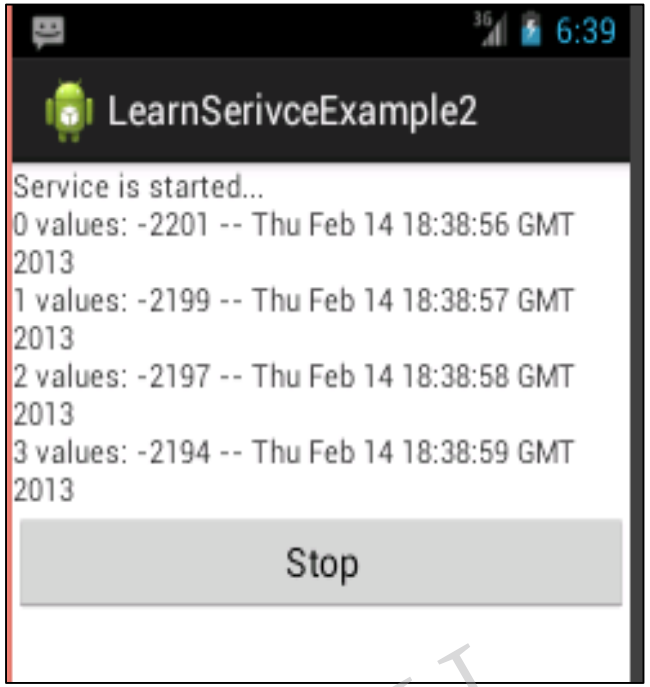
- Main Activity sẽ start service
- Service sẽ chạy nền bên dưới bằng việc dùng Thread
- Tại 1 thời điểm định sẵn, Service sẽ gửi 1 thông báo đến 1 Intent
- Intent được Broadcast với bộ lọc Filter

  matos.action.GOSERVICE3
- 1 Broadcast Receiver (định nghĩa bên trong MainActivity) sử dụng bộ lọc để bắt thông báo và hiển thị trong layout main.
- Có thể dừng service bằng việc bấm nút stop

6.3.1.6 Ví dụ 2 – Service & Broadcast Receiver

XML Layout

```
<LinearLayout xmlns:android="http://schemas.andro
    xmlns:tools="http://schemas.android.com/tools
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >
    <TextView
        android:id="@+id/textView1"
        android:layout_width="319dp"
        android:layout_height="wrap_content" />
    <Button
        android:id="@+id/btnstop"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Stop" />
</LinearLayout>
```



Ví dụ 2 – Service & Broadcast Receiver

```
public class MyService3 extends Service {
    boolean isRunning=true;
    public IBinder onBind(Intent arg0) {
    public void onCreate() {
    public void onDestroy() {
        super.onDestroy();isRunning=false;  }
    public void onStart(Intent intent, int startId) {
        Thread th=new Thread(new Runnable() {
            long startingtime=SystemClock.currentThreadTimeMillis();
            long tics=0;
            public void run() {
                for(int i=0;i<120 && isRunning;i++){
                    tics=SystemClock.currentThreadTimeMillis()-startingtime;
                    Intent myfilter=new Intent("matos.action.GOSERVICE3");
                    String msg=i+" values: "+tics;
                    myfilter.putExtra("servicedata", msg);
                    sendBroadcast(myfilter);
                    SystemClock.sleep(1000);
                }
            }
        });
        th.start();
    }
}
```

Ví dụ 2 – Service & Broadcast Receiver

```
public class MainActivity extends Activity {
    TextView txtmsg;
    Button btnstop;
    ComponentName service;
    Intent intentMyService;
    MyMainLocalReceiver receiver;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        txtmsg=(TextView) findViewById(R.id.textView1);
        intentMyService=new Intent(this, MyService3.class);
        service=startService(intentMyService);
        txtmsg.setText("Service is started...");
        IntentFilter mainFilter=new
                IntentFilter("matos.action.GOSERVICE3");
        receiver=new MyMainLocalReceiver();
        registerReceiver(receiver, mainFilter);
        btnstop=(Button) findViewById(R.id.btnstop);
        btnstop.setOnClickListener(new View.OnClickListener() {
            public void onClick(View arg0) {
                stopService(intentMyService);
                txtmsg.append("Service is stopped");
            }
        });
    }
```

Ví dụ 2 – Service & Broadcast Receiver

```
    protected void onDestroy() {
        super.onDestroy();
        stopService(intentMyService);
        unregisterReceiver(receiver);
    }
    public class MyMainLocalReceiver extends BroadcastReceiver {
        public void onReceive(Context arg0, Intent arg1) {
            String servicesData=arg1.getStringExtra("servicedata");
            Log.e("MAIN>>", servicesData +" - receiving data "
                    +SystemClock.elapsedRealtime());
            String now="\n"+servicesData+" -- "+new Date().toString();
            txtmsg.append(now);
        }
    }
}
```

LẬP TRÌNH ANDROID NÂNG CAO

PHẦN II : ANIMATION

7.1 ANIMATION

- **Animation** là những chuyển động hay tập hợp những chuyển động trên các đối tượng giúp chúng chuyển động mượt mà hơn.
- Thực chất thì Animation trong Android chỉ là sự thay đổi những thuộc tính của View. Và khi có những sự thay đổi thì Android sẽ tiến hành vẽ lại View (gọi lại phương thức onDraw của class View)
- Để làm việc với animation chúng ta cần **xách định được những tham số** sau đây:
  - Xác định được đối tượng và thuộc tính của đối tượng cần animate.
  - Giá trị bắt đầu và kết thúc của thuộc tính
  - Khoảng thời gian mà chúng ta muốn animate

7.1 ANIMATION

- **Animation** có thể thực hiện theo nhiều cách khác nhau: java code hoặc trong file xml, tuy nhiên phổ biến nhất là trong file xml gọi là Tween Animation.
- Tween Animation thực hiện các chuyển động bằng các thuộc tính và được android thực thi bằng lớp Animation.
- Cú pháp:

```
Animation animation = AnimationUtils.loadAnimation(
                    getApplicationContext(), R.anim.myanimation);
[ImageView].startAnimation(animation);
```

7.2 Các loại Animation

- Đây là các hiệu ứng thông dụng trong Animation:

Fade In	Zoom Out	Slide Down
Fade Out	Rotate	Sequential
Blink	Move	Together
Zoom In	Slide Up	

- Đây là các hiệu ứng thông dụng trong Animation:

7.2 Các loại Animation

- Đây là các hiệu ứng thông dụng trong Animation:

Thuộc tính	Miêu tả
android:duration	Thời gian hoàn thành
android:startOffset	Thời gian chờ trước khi một animaiton bắt đầu và thường được sử dụng khi có nhiều animation
android:repeatMode	Chế độ lặp lại animation (reverse)
android:repeatCount	Xác định số lần lặp lại animation. Nếu giá trị này là infinite thì animation sẽ lặp lại lần vô hạn
android:interpolator	Tỷ lệ thay đổi animation
android:fillAfter	Xác định liệu có áp dụng việc chuyển đổi đối tượng về trạng thái ban đầu sau khi một animation đã hoàn thành hay không.

7.3.1 FADE IN – FADE OUT

- Đây là các hiệu ứng ẩn hiện thông qua thuộc tính Alpha (giá trị từ 0 – 1):

```
<set xmlns:android="http://schemas.android.com/apk/res/android" >
        <alpha
        android:fromAlpha="0"
        android:toAlpha="1"
        android:duration="2000" >
        </alpha>
</set>
```

Fade in

```
<set xmlns:android="http://schemas.android.com/apk/res/android" >
        <alpha
        android:fromAlpha="1"
        android:toAlpha="0"
        android:duration="2000" >
        </alpha>
</set>
```

Fade out

7.3.2 BLINK

- Đây là các hiệu ứng nhấp nháy cũng thông qua thuộc tính Alpha (giá trị từ 0 – 1):

```
<set xmlns:android="http://schemas.android.com/android">
   <alpha android:fromAlpha="0.0"
      android:toAlpha="1.0"
      android:duration="600"
      android:repeatMode="reverse"
      android:repeatCount="infinite"/>
</set>
```

Blink

7.3.3 ZOOM IN – ZOOM OUT

- Đây là các hiệu ứng phóng to / thu nhỏ thông qua thuộc tính Scale:

```
<scale
 android:duration="1000"
 android:fromXScale="1"
 android:fromYScale="1"
 android:pivotX="50%"
 android:pivotY="50%"
 android:toXScale="3"
 android:toYScale="3" >
</scale>
```

Zoom in

```
<scale
 android:duration="1000"
 android:fromXScale="1"
 android:fromYScale="1"
 android:pivotX="50%"
 android:pivotY="50%"
 android:toXScale="0.5"
 android:toYScale="0.5" >
</scale>
```

Zoom out

7.3.4 ROTATE

- Đây là các hiệu ứng xoay thông qua thuộc tính Rotate :

```
<rotate android:fromDegrees="0"
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="600"
        android:repeatMode="restart"
        android:repeatCount="infinite"
"/>
```

Xoay theo chiều kim đồng hồ

7.3.5 Move

- Đây là các hiệu ứng di chuyển thông qua thuộc tính Translate :

```
<translate
        android:fromXDelta="0%p"
        android:fromYDelta="0%p"
        android:toXDelta="75%p"
        android:toXDelta="75%p"
        android:duration="800" />
```

Di chuyển xéo

7.3.6 SLIDE UP – SLIDE DOWN

- Đây là các hiệu ứng trượt lên / xuống thông qua thuộc tính Scale:

```
<scale
android:duration="500"
android:fromXScale="1.0"
android:fromYScale="1.0"
android:toXScale="1.0"
android:toYScale="0.0" />
```

Slide up

```
<scale
android:duration="500"
android:fromXScale="1.0"
android:fromYScale="0.0"
android:toXScale="1.0"
android:toYScale="1.0" />
```

Slide down

7.3.7 SEQUENTIAL

- Đây là các hiệu ứng thực hiện tuần tự:
  - Sử dụng **android:startOffset** để thiết lập thời gian trễ.
  - Cách tính: cộng giá trị **android:duration** của hiệu ứng hiện tại với giá trị **android:startOffset** của hiệu ứng trước

1. Di chuyển

```
<translate
    android:duration="800"
    android:fillAfter="true"
    android:fromXDelta="0%p"
    android:startOffset="300"
    android:toXDelta="75%p" />
<translate
    android:duration="800"
    android:fillAfter="true"
    android:fromYDelta="0%p"
    android:startOffset="1100"
    android:toYDelta="70%p" />
```

```
<translate
    android:duration="800"
    android:fillAfter="true"
    android:fromXDelta="0%p"
    android:startOffset="1900"
    android:toXDelta="-75%p" />
<translate
    android:duration="800"
    android:fillAfter="true"
    android:fromYDelta="0%p"
    android:startOffset="2700"
    android:toYDelta="-70%p" />
```

7.3.7 SEQUENTIAL (TT)

2. Xoay

```
<rotate
        android:duration="1000"
        android:fromDegrees="0"
        android:interpolator="@android:anim/cycle_interpolator"
        android:pivotX="50%"
        android:pivotY="50%"
        android:startOffset="3700"
        android:repeatCount="infinite"
        android:repeatMode="restart"
        android:toDegrees="360" />
```

7.3.8 TOGETHER ANIMATION

- Đây là các hiệu ứng thực hiện đồng thời tại cùng 1 thời điểm:
  - Viết tất cả các hiệu ứng mà không chỉ định **android:startOffset**

1. Di chuyển

```
<scale
    android:duration="4000"
    android:fromXScale="1"
    android:fromYScale="1"
    android:pivotX="50%"
    android:pivotY="50%"
    android:toXScale="4"
    android:toYScale="4" >

</scale>
```

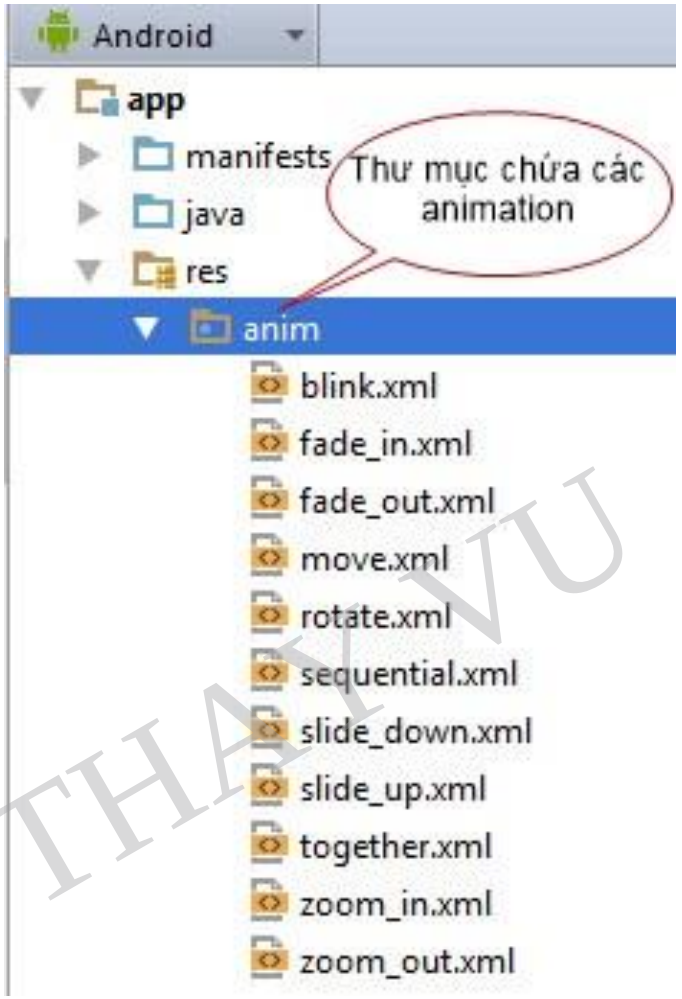
2. Xoay

```
<!-- Rotate 180 degrees -->
<rotate
    android:duration="500"
    android:fromDegrees="0"
    android:pivotX="50%"
    android:pivotY="50%"
    android:repeatCount="infinite"
    android:repeatMode="restart"
    android:toDegrees="360" />
```

7.4 CÁCH THỨC THỰC HIỆN

- **Bước 1: Tạo tập tin xml định nghĩa animation (**Tập tin này được đặt trong thư mục **anim** dưới thư mục **res** (res ⇒ anim ⇒ xml))

1/ Tạo thư mục **anim** bằng cách chuột phải thư mục **res ->** chọn **New ->** chọn **Directory**
2/ Tạo tập tin xml bằng cách chuột phải thư mục **anim ->** chọn **New ->** chọn **Animation resource file**



7.4 CÁCH THỨC THỰC HIỆN

❑ **Bước 2: Nạp animation**

Tại activity tạo một đối tượng của lớp Animation và nạp tập tin xml sử dụng phương thức **loadAnimation()** của lớp **AnimationUtils**

❑ **Bước 3: Thiết lập sự kiện (Tuỳ chọn)**

Nếu có xử lý các sự kiện như start, end và repeat, phải cài đặt giao diện AnimationListener cho activity.

❑ **Bước 4: Bắt đầu animation**

gọi phương thức **startAnimation()** cho đối tượng cần thực thi animation