Тема 15. Анимация

На это уроке мы узнаем, как можно добавить анимацию для простой картинки.

Шаг 1. Создаем внешний вид

Создаем layout main_layout.xml (src\main\res\layout\main_layout.xml) со следующим содержимым:

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/btn_name_one"
            android:id="@+id/button"
            android:layout_row="0"
            android:layout_column="0"
            android:textSize="16dp"
            android:padding="15dp"
            android:onClick="onAnimationImageOne" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/btn_name_two"
            android:id="@+id/button2"
            android:layout_row="0"
            android:layout_column="1"
            android:textSize="16dp"
            android:padding="15dp"
            android:layout_alignParentTop="true"
            android:layout_toRightOf="@+id/button"
            android:onClick="onAnimationImageTwo" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/btn_name_three"
            android:id="@+id/button3"
            android:layout_row="0"
            android:layout_column="2"
            android:textSize="16dp"
            android:padding="15dp"
            android:layout_alignParentTop="true"
            android:layout_alignParentRight="true"
            android:onClick="onAnimationImageThree" />

        <ImageView
```

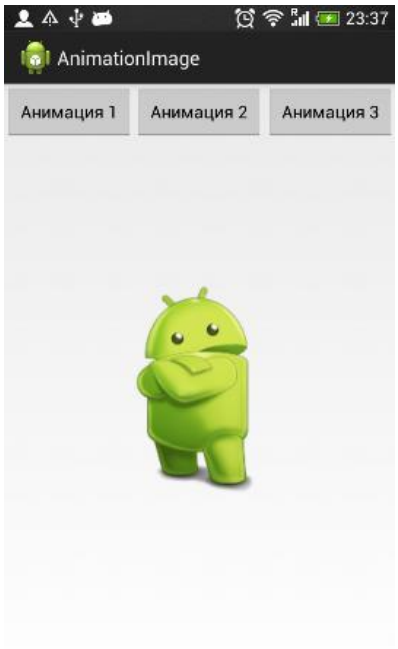
```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/imageView"
            android:layout_row="2"
            android:layout_column="1"
            android:src="@drawable/android_img"
            android:layout_centerVertical="true"
            android:layout_centerHorizontal="true" />
    </RelativeLayout>
```

И в string.xml добавим следующие строки:

```
    <resources>
        <string name="app_name">Example Animation Image</string>
        <string name="btn_name_one">Анимация 1</string>
        <string name="btn_name_two">Анимация 2</string>
        <string name="btn_name_three">Анимация 2</string>
    </resources>
```

И теперь в src\main\res\drawable добавляем картинку android_img.png:

В итоге наше приложение будет выглядеть так:



Шаг 2. Создаем Activity

Теперь давайте добавим Activity для нашего layout, для этого в \src\main\java\com\devcolibri**animationimage** создаем новое Activity **MainActivity.java** со следующим содержимым:

```
package com.devcolibri.animationimage;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;

public class MainActivity extends Activity {

   private ImageView imageView;

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

      imageView = (ImageView) findViewById(R.id.imageView);
   }

}
```

После этого ваш layout можно будет увидеть после запуска приложения, но незабывайте настроить activity в AndroidManifest.xml.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.devcolibri.animationimage"
   android:versionCode="1"
   android:versionName="1.0">

   <uses-sdk android:minSdkVersion="11" android:targetSdkVersion="17" />

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

</manifest>
```

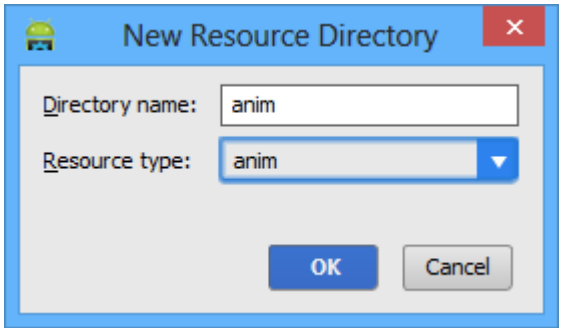
Теперь все будет работать

Шаг 3. Создаем анимации

В Android анимации представлены в виде анимационных ресурс-файлов, а точней в виде XML файлов, в которох описанно поведение элемента на который будет применена данная анимация.

Анимация №1

Для начала создадим самую простую анимацию для первой кнопки. Она будет смещать наже изображение влево, потом возвращать на исходное положение. Для того, чтобы создать анимацию в папке res создайте новую ресурсную папку anim:



После этого в папке anim создаем ***Animation resource file*** с именем ***animation_one.xml*** и содержимым:

```
<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android"
      android:interpolator="@android:anim/decelerate_interpolator">

      <translate android:fromXDelta="0"
        android:toXDelta="-20%p"
        android:duration="200"/>

</set>
```

Как видите все начинается с тега **set,** он является стандартным тегом для всех анимационных русурс файлов, а вот тег **translate** – это уже наша анимация, где:

android:toXDelta – это смежение по оси X в лево;

android:duration – это продожытельность перемещения, чем больше duration, тем медленей будет двигаться изображение.

Анимация №2

Теперь создадим в тойже папке anim новый ресурс файл с именем ***anumation_two.xml*** и содержимым:

```
<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android"
   android:interpolator="@android:anim/decelerate_interpolator">
```

```
    <rotate android:pivotX="50%"
        android:pivotY="50%"
        android:fromDegrees="0"
        android:toDegrees="360"
        android:duration="1000"/>

</set>
```

Как видите в этой анимации мы используем тег **rotate** – он позволит нам заставить наше изображение сделать поворот, в нашем случае на 360 градусов.

Давайте рассмотрим детальней атрибуты свойства rotate:

android:pivotX – это говорит о том, что точка поворота будет в центре изображения, поэтому оно просто сделает круговой поворот;
android:fromDegrees – тут мы указываем начальное угловое положение в градусах;
android:toDegrees – а тут конечное угловое положение в градусах;
android:duration – это продолжительность анимации, чем больше оно будет тем медленей будет анимация.

Анимация №3

Теперь рассмотрим немного сложней анимацию. Для этого создаем 3-ю анимацию и называем её ***animation_three.xml***:

```
<?xml version="1.0" encoding="utf-8"?>

<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false" >

    <scale
        android:duration="1250"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="2.0"
        android:toYScale="2.0" />

    <rotate
        android:duration="2500"
        android:fromDegrees="0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toDegrees="360" />

    <scale
        android:duration="1250"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:pivotX="50%"
```

```
        android:pivotY="50%"
        android:startOffset="1250"
        android:toXScale="0.5"
        android:toYScale="0.5" />

    </set>
```

Тут мы уже видим несколько анимационных свойств объеденены в одну анимацию. Вы уже, наверное, заметели, что мы сново используем анимацию ***rotate***, но помимо её тут есть и новая анимация ***scale***.

Scale позволяет сделать динамическое изменение размера компонента, в нашем случае картинки.

В данном сценарии мы сначало выполняем увеличение изображения, и в этот же момент поворачиваем его на 360 градусов и на половине поворота изображение перестает увеличиваться, и начинает уменьшаться.

Атрибуты scale:

android:duration – продолжительность анимации;
android:fromXScale – начальная позиция смещения анимации по оси X;
android:fromYScale – начальная позиция смещения анимации по оси Y;
android:pivotX – определяет начальную точку изменения размера по оси X;
android:pivotY – определяет начальную точку изменения размера по оси Y;
android:startOffset – количество миллисекунд задержки анимации после запуска;
android:toXScale – размер смещения по оси X;
android:toYScale – размер смещения по оси Y.

Шаг 4. Подключение анимаций

Теперь давайте подключим анимации на картинку, но они будут срабатывать после клика по кнопки, где каждая кнопка будет отвечать за определёную анимацию.

Вот полный код MainActivity:

```
package com.devcolibri.animationimage;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;

public class MainActivity extends Activity {

    // Создаем экземпляры для наших анимаций
    private Animation animOne, animTwo, animThree;
```

```
    private ImageView imageView;

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

        imageView = (ImageView) findViewById(R.id.imageView);

        // Подгружаем все анимации
        animOne = AnimationUtils.loadAnimation(this, R.anim.animation_one);
        animTwo = AnimationUtils.loadAnimation(this, R.anim.anumation_two);
        animThree = AnimationUtils.loadAnimation(this, R.anim.animation_three);
    }

    // Анимация №1
    public void onAnimationImageOne(View v){
        imageView.startAnimation(animOne);
    }

    // Анимация №2
    public void onAnimationImageTwo(View v){
        imageView.startAnimation(animTwo);
    }

    // Анимация №3
    public void onAnimationImageThree(View v){
        imageView.startAnimation(animThree);
    }

}
```

Методы onAnimationImageOne, onAnimationImageTwo, onAnimationImageThree вызываются в ***main_layout*** на каждой кнопке в свойстве android:onClick.

Вот структура проекта