**Translate**

Рассмотрим пример перемещения объекта. Создадим проект, в котором элемент **ImageView** будет двигаться сверху вниз (падать). В папке **res/anim** создадим файл *falling.xml*:

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

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

android:interpolator="@android:anim/linear\_interpolator" >

<translate

android:duration="2000"

android:fromYDelta="-100%"

android:toYDelta="700" />

</set>

Разместите на форме кнопку и **ImageView** и добавьте код:

package ru.alexanderklimov.testapp;

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.Button;

import android.widget.ImageView;

public class TestAppActivity extends Activity {

private ImageView mImageView;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

mImageView = (ImageView) findViewById(R.id.imageView1);

Button startFallingButton = (Button) findViewById(R.id.starttranslate);

final Animation fallingAnimation = AnimationUtils.loadAnimation(this,

R.anim.falling);

startFallingButton.setOnClickListener(new Button.OnClickListener(){

@Override

public void onClick(View arg0) {

// TODO Auto-generated method stub

mImageView.startAnimation(fallingAnimation);

}});

}

}

Запускайте проект и любуйтесь эффектом.

Усложним проект и заставим падать несколько объектов по очереди. Создадим два новых файла анимации:

**falling\_down.xml**

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

<translate xmlns:android="http://schemas.android.com/apk/res/android"

android:duration="1000"

android:fromYDelta="-50%"

android:interpolator="@android:anim/accelerate\_interpolator"

android:toYDelta="0.0" />

**falling\_up.xml**

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

<translate xmlns:android="http://schemas.android.com/apk/res/android"

android:duration="1000"

android:fromYDelta="0.0"

android:interpolator="@android:anim/accelerate\_interpolator"

android:toYDelta="50%" />

Подготовим разметку из трёх невидимых **ImageView**:

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="horizontal" >

<ImageView

android:id="@+id/image1"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:src="@drawable/ic\_launcher"

android:visibility="invisible" />

<ImageView

android:id="@+id/image2"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:src="@drawable/ic\_launcher"

android:visibility="invisible" />

<ImageView

android:id="@+id/image3"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:src="@drawable/ic\_launcher"

android:visibility="invisible" />

</LinearLayout>

Напишем код:

package ru.alexanderklimov.falling;

import ...

public class MainActivity extends Activity {

private ImageView mImageView1, mImageView2, mImageView3;

private Animation fallingAnimation, upAnimation;

private ImageView curFallingImageView;

@Override

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.activity\_main);

mImageView1 = (ImageView) findViewById(R.id.image1);

mImageView2 = (ImageView) findViewById(R.id.image2);

mImageView3 = (ImageView) findViewById(R.id.image3);

fallingAnimation = AnimationUtils.loadAnimation(this,

R.anim.falling\_down);

upAnimation = AnimationUtils.loadAnimation(this, R.anim.falling\_up);

fallingAnimation.setAnimationListener(animationFallingListener);

upAnimation.setAnimationListener(animationUpListener);

curFallingImageView = mImageView1;

mImageView1.startAnimation(fallingAnimation);

mImageView1.setVisibility(View.VISIBLE);

}

@Override

protected void onPause() {

// TODO Auto-generated method stub

super.onPause();

mImageView1.clearAnimation();

mImageView2.clearAnimation();

mImageView3.clearAnimation();

}

AnimationListener animationFallingListener = new AnimationListener() {

@Override

public void onAnimationEnd(Animation arg0) {

// TODO Auto-generated method stub

if (curFallingImageView == mImageView1) {

mImageView1.startAnimation(upAnimation);

} else if (curFallingImageView == mImageView2) {

mImageView2.startAnimation(upAnimation);

} else if (curFallingImageView == mImageView3) {

mImageView3.startAnimation(upAnimation);

}

}

@Override

public void onAnimationRepeat(Animation animation) {

// TODO Auto-generated method stub

}

@Override

public void onAnimationStart(Animation animation) {

// TODO Auto-generated method stub

}

};

AnimationListener animationUpListener = new AnimationListener() {

@Override

public void onAnimationEnd(Animation animation) {

// TODO Auto-generated method stub

if (curFallingImageView == mImageView1) {

curFallingImageView = mImageView2;

mImageView2.startAnimation(fallingAnimation);

mImageView1.setVisibility(View.INVISIBLE);

mImageView2.setVisibility(View.VISIBLE);

mImageView3.setVisibility(View.INVISIBLE);

} else if (curFallingImageView == mImageView2) {

curFallingImageView = mImageView3;

mImageView3.startAnimation(fallingAnimation);

mImageView1.setVisibility(View.INVISIBLE);

mImageView2.setVisibility(View.INVISIBLE);

mImageView3.setVisibility(View.VISIBLE);

} else if (curFallingImageView == mImageView3) {

curFallingImageView = mImageView1;

mImageView1.startAnimation(fallingAnimation);

mImageView1.setVisibility(View.VISIBLE);

mImageView2.setVisibility(View.INVISIBLE);

mImageView3.setVisibility(View.INVISIBLE);

}

}

@Override

public void onAnimationRepeat(Animation animation) {

// TODO Auto-generated method stub

}

@Override

public void onAnimationStart(Animation animation) {

// TODO Auto-generated method stub

}

};

}

Запустив проект, вы увидите, как сверху будут сыпаться картинки.

**Движение по диагонали**

По такому же принципу можно организовать движение по диагонали, изменяя одновременно значения X и Y. Создадим ещё один файл **res/anim/slide\_diagonal.xml**:

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

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

<translate

android:fromYDelta="0"

android:toYDelta="100%p"/>

<translate

android:fromXDelta="0"

android:toXDelta="100%p"/>

</set>

Хотя эта запись избыточна, можно и так:

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

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

<translate

android:fromXDelta="0"

android:fromYDelta="0"

android:toXDelta="100%p"

android:toYDelta="100%p" />

</set>

Разместите в углу экрана **ImageView** внутри контейнера **LinearLayout**, чтобы у него была возможность двигаться из угла в угол. Напишем код.

package ru.alexanderklimov.test;

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 mImageView;

private Animation mDiagonalAnimation;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

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

mDiagonalAnimation = AnimationUtils.loadAnimation(this, R.anim.slide\_diagonal);

mDiagonalAnimation.setDuration(3000);

mDiagonalAnimation.setAnimationListener(animationListener);

mImageView.startAnimation(mDiagonalAnimation);

}

Animation.AnimationListener animationListener = new Animation.AnimationListener() {

@Override

public void onAnimationStart(Animation animation) {

}

@Override

public void onAnimationEnd(Animation animation) {

mImageView.startAnimation(mDiagonalAnimation);

}

@Override

public void onAnimationRepeat(Animation animation) {

}

};

}

**Смотри также**

[Класс TranslateAnimation](http://developer.alexanderklimov.ru/android/animation/translateanimation.php)