Новые анимации активностей в Android 4.1 (Jelly Bean)

В Android 4.1 (Jelly Bean) появился новый вспомогательный класс **ActivityOptions** с тремя методами для анимации активности:

* makeCustomAnimation()
* makeScaleUpAnimation()
* makeThumbnailScaleUpAnimation()

Давайте рассмотрим эти методы на примерах. На эмуляторе всё происходит очень быстро и увидеть красоту анимации не удалось, остаётся надеяться, что на реальном устройстве анимация будет выглядеть лучше.

Для первого метода **makeCustomAnimation()** (создатьСобственнуюАнимацию) нам понадобятся два XML-файла, которые необходимо разместить в папке res/anim. Анимация будет простой и хорошо нам знакомой - изменение прозрачности.

fade\_in.xml

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

android:duration="@android:integer/config\_longAnimTime"

android:fromAlpha="0.0"

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

android:toAlpha="1.0" />

fade\_out.xml

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

android:duration="@android:integer/config\_longAnimTime"

android:fromAlpha="1.0"

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

android:toAlpha="0.0" />

Теперь разместим в главной активности три кнопки для демонстрации каждой анимации. В атрибуте **onClick** зададим имена обработчиков:

<LinearLayout 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"

android:orientation="vertical" >

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:onClick="fadeAnimation"

android:text="Fade Animation" />

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:onClick="scaleupAnimation"

android:text="Scaleup Animation" />

<Button

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:onClick="thumbNailScaleAnimation"

android:text="Thumbnail Scaleup Animation" />

</LinearLayout>

Далее создадим разметку для второй активности (activity\_second.xml). Здесь важно использовать ImageView, который будет играть роль превьюшки для анимации:

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical" >

<ImageView

android:id="@+id/imageView1"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_centerInParent="true"

android:layout\_marginRight="44dp"

android:layout\_marginTop="54dp"

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

</RelativeLayout>

Код для второй активности будет минимален - просто загрузить созданную разметку:

package ru.alexanderklimov.jellyactivityanimation;

import android.app.Activity;

import android.os.Bundle;

public class SecondActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_second);

}

}

Ну и наконец, код для главной активности. Для каждой кнопки мы создаём собственный обработчик, который просто запускает активность:

package ru.alexanderklimov.jellyactivityanimation;

import android.os.Bundle;

import android.app.Activity;

import android.app.ActivityOptions;

import android.content.Intent;

import android.graphics.Bitmap;

import android.view.Menu;

import android.view.View;

public class MainActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}

public void scaleupAnimation(View view) {

ActivityOptions opts = ActivityOptions.makeScaleUpAnimation(view, 0, 0,

view.getWidth(), view.getHeight());

startActivity(new Intent(MainActivity.this, SecondActivity.class),

opts.toBundle());

}

public void thumbNailScaleAnimation(View view) {

view.setDrawingCacheEnabled(true);

view.setPressed(false);

view.refreshDrawableState();

Bitmap bitmap = view.getDrawingCache();

ActivityOptions opts = ActivityOptions.makeThumbnailScaleUpAnimation(

view, bitmap, 0, 0);

startActivity(new Intent(MainActivity.this, SecondActivity.class),

opts.toBundle());

view.setDrawingCacheEnabled(false);

}

public void fadeAnimation(View view) {

ActivityOptions opts = ActivityOptions.makeCustomAnimation(

MainActivity.this, R.anim.fade\_in, R.anim.fade\_out);

startActivity(new Intent(MainActivity.this, SecondActivity.class),

opts.toBundle());

}

}

Запускаем проект и смотрим на результат.

Для метода **makeCustomAnimation()** также попробуйте эффект сдвига в виде слайд-шоу:

public void slideAnimation(View view) {

ActivityOptions opts = ActivityOptions.makeCustomAnimation(

MainActivity.this, R.anim.slide\_in\_left, R.anim.slide\_out\_left);

startActivity(new Intent(MainActivity.this, SecondActivity.class),

opts.toBundle());

}

К этому примеру требуются файлы анимации в папке **res/anim**:

slide\_in\_left

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

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

android:fromXDelta="100%p" android:toXDelta="0"

android:duration="500" />

slide\_out\_left

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

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

android:fromXDelta="0" android:toXDelta="-50%p"

android:duration="500" />

А чтобы при закрытии второй активности через кнопку Back также воспроизводилась анимация слайд-шоу только в обратном порядке, то добавьте в метод **finish()** второй активности следующий код:

@Override

public void finish() {

super.finish();

overridePendingTransition(R.anim.slide\_in\_right, R.anim.slide\_out\_right);

}

Соответственно, для этой анимации требуются свои файлы анимации:

slide\_in\_right.xml

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

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

android:fromXDelta="-50%p" android:toXDelta="0"

android:duration="500" />

slide\_out\_right.xml

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

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

android:fromXDelta="0" android:toXDelta="100%p"

android:duration="500" />

**Метод makeThumbnailScaleUpAnimation()**

Разработчики из Google выложили интересный пример применения метода **makeThumbnailScaleUpAnimation()**. Экран разбит на клетки (GridView) и щелчок на любой клетке вызывает новую активность. Посмотрите ролик, а также можно скачать [пример](http://developer.android.com/shareables/devbytes/WindowZoomAnimations.zip).