**ObjectAnimator Анимация шариков**

На эмуляторе есть приложение "API Demos", котором представлен пример анимации с шариками. Изучая пример, решил оставить его себе на память (с небольшими изменениями).

Для начала создадим вспомогательный класс **ShapeHolder**, позволяющий задать тип фигуры, цвет, позицию, градиент и прочие параметры.

package ru.alexanderklimov.testnofragment;

import android.graphics.Paint;

import android.graphics.RadialGradient;

import android.graphics.drawable.ShapeDrawable;

import android.graphics.drawable.shapes.Shape;

public class ShapeHolder {

private float mX = 0, mY = 0;

private ShapeDrawable mShape;

private int mColor;

private RadialGradient mRadialGradient;

private float mAlpha = 1f;

private Paint mPaint;

public void setPaint(Paint paint) {

mPaint = paint;

}

public Paint getPaint() {

return mPaint;

}

public void setX(float value) {

mX = value;

}

public float getX() {

return mX;

}

public void setY(float value) {

mY = value;

}

public float getY() {

return mY;

}

public void setShape(ShapeDrawable shape) {

mShape = shape;

}

public ShapeDrawable getShape() {

return mShape;

}

public int getColor() {

return mColor;

}

public void setColor(int color) {

mShape.getPaint().setColor(color);

mColor = color;

}

public void setGradient(RadialGradient gradient) {

mRadialGradient = gradient;

}

public RadialGradient getGradient() {

return mRadialGradient;

}

public void setAlpha(float alpha) {

this.mAlpha = alpha;

mShape.setAlpha((int) ((alpha \* 255f) + .5f));

}

public float getWidth() {

return mShape.getShape().getWidth();

}

public void setWidth(float width) {

Shape s = mShape.getShape();

s.resize(width, s.getHeight());

}

public float getHeight() {

return mShape.getShape().getHeight();

}

public void setHeight(float height) {

Shape s = mShape.getShape();

s.resize(s.getWidth(), height);

}

public ShapeHolder(ShapeDrawable drawable) {

mShape = drawable;

}

}

Далее создадим новый класс **BallView**, в котором зададим производство четырёх шариков с разными градиентами. У каждого шарика будет своя анимация.

package ru.alexanderklimov.testnofragment;

import android.animation.Animator;

import android.animation.AnimatorSet;

import android.animation.Keyframe;

import android.animation.ObjectAnimator;

import android.animation.PropertyValuesHolder;

import android.animation.ValueAnimator;

import android.content.Context;

import android.graphics.Canvas;

import android.graphics.Paint;

import android.graphics.RadialGradient;

import android.graphics.Shader;

import android.graphics.drawable.ShapeDrawable;

import android.graphics.drawable.shapes.OvalShape;

import android.view.View;

import android.view.animation.AccelerateInterpolator;

import android.view.animation.BounceInterpolator;

import java.util.ArrayList;

public class BallView extends View implements ValueAnimator.AnimatorUpdateListener {

private static final float BALL\_SIZE = 100f;

private static final int DURATION = 1500;

public final ArrayList<ShapeHolder> balls = new ArrayList<ShapeHolder>();

private Animator mBounceAnimator = null;

public BallView(Context context) {

super(context);

addBall(50, 0);

addBall(160, 0);

addBall(270, 0);

addBall(380, 0);

}

@Override

public void onAnimationUpdate(ValueAnimator animation) {

invalidate();

}

private void createAnimation() {

if (mBounceAnimator == null) {

ShapeHolder ball;

// первый шарик

ball = balls.get(0);

ObjectAnimator yBouncer = ObjectAnimator.ofFloat(ball, "y",

ball.getY(), getHeight() - BALL\_SIZE).setDuration(DURATION);

yBouncer.setInterpolator(new BounceInterpolator());

yBouncer.addUpdateListener(this);

// второй шарик

ball = balls.get(1);

PropertyValuesHolder pvhY = PropertyValuesHolder.ofFloat("y", ball.getY(),

getHeight() - BALL\_SIZE);

PropertyValuesHolder pvhAlpha = PropertyValuesHolder.ofFloat("alpha", 1.0f, 0f);

ObjectAnimator yAlphaBouncer = ObjectAnimator.ofPropertyValuesHolder(ball,

pvhY, pvhAlpha).setDuration(DURATION / 2);

yAlphaBouncer.setInterpolator(new AccelerateInterpolator());

yAlphaBouncer.setRepeatCount(1);

yAlphaBouncer.setRepeatMode(ValueAnimator.REVERSE);

// третий шарик

ball = balls.get(2);

PropertyValuesHolder pvhW = PropertyValuesHolder.ofFloat("width", ball.getWidth(),

ball.getWidth() \* 2);

PropertyValuesHolder pvhH = PropertyValuesHolder.ofFloat("height", ball.getHeight(),

ball.getHeight() \* 2);

PropertyValuesHolder pvTX = PropertyValuesHolder.ofFloat("x", ball.getX(),

ball.getX() - BALL\_SIZE / 2f);

PropertyValuesHolder pvTY = PropertyValuesHolder.ofFloat("y", ball.getY(),

ball.getY() - BALL\_SIZE / 2f);

ObjectAnimator whxyBouncer = ObjectAnimator.ofPropertyValuesHolder(ball, pvhW, pvhH,

pvTX, pvTY).setDuration(DURATION / 2);

whxyBouncer.setRepeatCount(1);

whxyBouncer.setRepeatMode(ValueAnimator.REVERSE);

// четвертый шарик

ball = balls.get(3);

pvhY = PropertyValuesHolder.ofFloat("y", ball.getY(), getHeight() - BALL\_SIZE);

float ballX = ball.getX();

Keyframe kf0 = Keyframe.ofFloat(0f, ballX);

Keyframe kf1 = Keyframe.ofFloat(.5f, ballX + 100f);

Keyframe kf2 = Keyframe.ofFloat(1f, ballX + 50f);

PropertyValuesHolder pvhX = PropertyValuesHolder.ofKeyframe("x", kf0, kf1, kf2);

ObjectAnimator yxBouncer = ObjectAnimator.ofPropertyValuesHolder(ball, pvhY,

pvhX).setDuration(DURATION / 2);

yxBouncer.setRepeatCount(1);

yxBouncer.setRepeatMode(ValueAnimator.REVERSE);

mBounceAnimator = new AnimatorSet();

((AnimatorSet) mBounceAnimator).playTogether(yBouncer, yAlphaBouncer, whxyBouncer,

yxBouncer);

}

}

public void startAnimation() {

createAnimation();

mBounceAnimator.start();

}

private ShapeHolder addBall(float x, float y) {

OvalShape circle = new OvalShape();

circle.resize(BALL\_SIZE, BALL\_SIZE);

ShapeDrawable drawable = new ShapeDrawable(circle);

ShapeHolder shapeHolder = new ShapeHolder(drawable);

shapeHolder.setX(x);

shapeHolder.setY(y);

int red = (int)(100 + Math.random() \* 55);

int green = (int)(100 + Math.random() \* 105);

int blue = (int)(100 + Math.random() \* 205);

int color = 0xff000000 | red << 16 | green << 8 | blue;

Paint paint = drawable.getPaint();

int darkColor = 0xff000000 | blue/4 << 16 | green/6 << 8 | red/4;

RadialGradient gradient = new RadialGradient(37.5f, 12.5f,

50f, color, darkColor, Shader.TileMode.CLAMP);

paint.setShader(gradient);

shapeHolder.setPaint(paint);

balls.add(shapeHolder);

return shapeHolder;

}

@Override

protected void onDraw(Canvas canvas) {

for (ShapeHolder ball : balls) {

canvas.translate(ball.getX(), ball.getY());

ball.getShape().draw(canvas);

canvas.translate(-ball.getX(), -ball.getY());

}

}

}

Предварительная работа закончена. Создайте экран на основе **LinearLayout** с одной кнопкой, шарики будут добавляться динамически через код.

package ru.alexanderklimov.test;

import android.app.Activity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.LinearLayout;

public class MainActivity extends Activity {

private BallView mBallView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

LinearLayout container = (LinearLayout) findViewById(R.id.container);

mBallView = new BallView(this);

container.addView(mBallView);

}

public void onClick(View view) {

mBallView.startAnimation();

}

}