Анимация на Android с помощью SurfaceView

**Среда программирования:**

Android Studio

В уроке 2 [Анимация на Android](http://grafika.me/node/454) показано, как делать анимацию с помощью **View**и функции **invalidate()**. Но у такого метода есть недостаток – метод **invalidate()** вызывает **onDraw()** не мгновенно, а когда решит операционная система. И если анимация требует тяжелых расчетов, она может заметно тормозить.  
Для тяжелых случаев есть другой метод рисования через **SurfaceView**.

Класс **SurfaceView**предоставляет объект **Surface**, который поддерживает рисование в фоновом потоке и дает возможность использовать OpenGL для трехмерной графики. Это отличный вариант для насыщенных графикой элементов, которые нуждаются в частых обновлениях или должны отображать сложную графическую информацию, как в случае с играми и трехмерной визуализацией.

Основной плюс **SurfaceView** в том, что он работает в паре с объектом класса **Thread**, что позволяет выполнять тяжелые графические расчеты в отдельном потоке.

Создадим в нашем проекте еще два класса: **MySurfaceView**и **MyThread**.

**MySurfaceView:**

**package** me.graphica.canvasanimation;

**import** android.content.Context;

**import** android.view.SurfaceHolder;

**import** android.view.SurfaceView;

**public** **class** MySurfaceView **extends** SurfaceView **implements** SurfaceHolder.Callback {

**private** MyThread mMyThread; *//наш поток прорисовки*

**public** MySurfaceView([Context](http://www.google.com/search?hl=en&q=allinurl%3Acontext+java.sun.com&btnI=I%27m%20Feeling%20Lucky) context) {

**super**(context);

getHolder().addCallback(**this**);

}

@Override

**public** **void** surfaceCreated(SurfaceHolder holder) { *//вызывается, когда surfaceView появляется на экране*

mMyThread = **new** MyThread(getHolder());

mMyThread.setRunning(**true**);

mMyThread.start(); *//запускает процесс в отдельном потоке*

}

@Override

**public** **void** surfaceChanged(SurfaceHolder holder, **int** format, **int** width, **int** height) {

*//когда view меняет свой размер*

}

@Override

**public** **void** surfaceDestroyed(SurfaceHolder holder) { *//когда view исчезает из поля зрения*

**boolean** retry = **true**;

mMyThread.setRunning(**false**); *//останавливает процесс*

**while**(retry) {

**try** {

mMyThread.join(); *//ждет окончательной остановки процесса*

retry = **false**;

}

**catch** ([InterruptedException](http://www.google.com/search?hl=en&q=allinurl%3Ainterruptedexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) e) {

*//не более чем формальность*

}

}

}

}

**MyThread:**

**package** me.graphica.canvasanimation;

**import** android.animation.ArgbEvaluator;

**import** android.graphics.Canvas;

**import** android.graphics.Color;

**import** android.graphics.Paint;

**import** android.util.Log;

**import** android.view.SurfaceHolder;

***/\*\****

***\* Created by Sex\_predator on 27.03.2016.***

***\*/***

**public** **class** MyThread **extends** [Thread](http://www.google.com/search?hl=en&q=allinurl%3Athread+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {

**private** **final** **int** REDRAW\_TIME = 10; *//частота обновления экрана - 10 мс*

**private** **final** **int** ANIMATION\_TIME = 1\_500; *//анимация - 1,5 сек*

**private** **final** SurfaceHolder mSurfaceHolder; *//нужен, для получения canvas*

**private** **boolean** mRunning; *//запущен ли процесс*

**private** **long** mStartTime; *//время начала анимации*

**private** **long** mPrevRedrawTime; *//предыдущее время перерисовки*

**private** [Paint](http://www.google.com/search?hl=en&q=allinurl%3Apaint+java.sun.com&btnI=I%27m%20Feeling%20Lucky) mPaint;

**private** ArgbEvaluator mArgbEvaluator;

**public** MyThread(SurfaceHolder holder) {

mSurfaceHolder = holder;

mRunning = **false**;

mPaint = **new** [Paint](http://www.google.com/search?hl=en&q=allinurl%3Apaint+java.sun.com&btnI=I%27m%20Feeling%20Lucky)();

mPaint.setAntiAlias(**true**);

mPaint.setStyle([Paint](http://www.google.com/search?hl=en&q=allinurl%3Apaint+java.sun.com&btnI=I%27m%20Feeling%20Lucky).[Style](http://www.google.com/search?hl=en&q=allinurl%3Astyle+java.sun.com&btnI=I%27m%20Feeling%20Lucky).FILL);

mArgbEvaluator = **new** ArgbEvaluator();

}

**public** **void** setRunning(**boolean** running) { *//запускает и останавливает процесс*

mRunning = running;

mPrevRedrawTime = getTime();

}

**public** **long** getTime() {

**return** [System](http://www.google.com/search?hl=en&q=allinurl%3Asystem+java.sun.com&btnI=I%27m%20Feeling%20Lucky).nanoTime() / 1\_000\_000;

}

@Override

**public** **void** run() {

[Canvas](http://www.google.com/search?hl=en&q=allinurl%3Acanvas+java.sun.com&btnI=I%27m%20Feeling%20Lucky) canvas;

mStartTime = getTime();

**while** (mRunning) {

**long** curTime = getTime();

**long** elapsedTime = curTime - mPrevRedrawTime;

**if** (elapsedTime < REDRAW\_TIME) *//проверяет, прошло ли 10 мс*

**continue**;

*//если прошло, перерисовываем картинку*

canvas = **null**;

**try** {

canvas = mSurfaceHolder.lockCanvas(); *//получаем canvas*

**synchronized** (mSurfaceHolder) {

draw(canvas); *//функция рисования*

}

}

**catch** ([NullPointerException](http://www.google.com/search?hl=en&q=allinurl%3Anullpointerexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) e) {*/\*если canvas не доступен\*/*}

**finally** {

**if** (canvas != **null**)

mSurfaceHolder.unlockCanvasAndPost(canvas); *//освобождаем canvas*

}

mPrevRedrawTime = curTime;

}

}

**private** **void** draw([Canvas](http://www.google.com/search?hl=en&q=allinurl%3Acanvas+java.sun.com&btnI=I%27m%20Feeling%20Lucky) canvas) {

**long** curTime = getTime() - mStartTime;

**int** width = canvas.getWidth();

**int** height = canvas.getHeight();

canvas.drawColor([Color](http://www.google.com/search?hl=en&q=allinurl%3Acolor+java.sun.com&btnI=I%27m%20Feeling%20Lucky).BLACK);

**int** centerX = width / 2;

**int** centerY = height / 2;

**float** maxSize = [Math](http://www.google.com/search?hl=en&q=allinurl%3Amath+java.sun.com&btnI=I%27m%20Feeling%20Lucky).min(width, height) / 2;

**float** fraction = (**float**) (curTime % ANIMATION\_TIME) / ANIMATION\_TIME;

**int** color = (**int**) mArgbEvaluator.evaluate(fraction, [Color](http://www.google.com/search?hl=en&q=allinurl%3Acolor+java.sun.com&btnI=I%27m%20Feeling%20Lucky).RED, [Color](http://www.google.com/search?hl=en&q=allinurl%3Acolor+java.sun.com&btnI=I%27m%20Feeling%20Lucky).BLACK);

mPaint.setColor(color);

canvas.drawCircle(centerX, centerY, maxSize \* fraction, mPaint);

}

}

Не рекомендуется делать **REDRAW\_TIME**слишком маленьким, это может сильно нагружать систему. 10 мс вполне достаточно.

Последний штрих: вернемся в **MyActivity**и сменим  
setContentView(**new** MyView(**this**));  
на  
setContentView(**new** MySurfaceView(**this**));

Приложение должно выглядеть так:  
