пример программы с различной логикой вариантов рисования на экране. Смотрите комментарии и меняйте значение mySurfaceView.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT  
В комментариях отписал, что и как вызывается и что мне не очень нравится в том или ином варианте.

MainActivity

**package** com.example.surfaceviewexample;

**import** android.os.Bundle;

**import** android.app.Activity;

**import** android.graphics.Canvas;

**import** android.view.Gravity;

**import** android.view.Menu;

**import** android.view.MenuItem;

**import** android.view.SubMenu;

**import** android.widget.Toast;

**public** **class** **MainActivity** **extends** **Activity** {

MySurfaceView mySurfaceView;

BackgroundThread backgroundThread;

**@Override**

**public** **void** **onCreate**(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

mySurfaceView = **new** MySurfaceView(**this**);

setContentView(mySurfaceView);

}

*//BackgroundThread knows about only MainActivity and call this.parent.draw*

*//but MainActivity knows about MySurfaceView and delegate draw to its "draw" method*

**public** **void** **draw**(**int** number) {

**switch**(mySurfaceView.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(2):

**case**(3):

**case**(5):

**case**(6):{

mySurfaceView.draw(number);

**break**;

}

**case**(4):{

**final** **int** i = number;

**new** Thread(**new** Runnable(){

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

mySurfaceView.post(**new** Runnable(){

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

mySurfaceView.draw(i);

}

});

}

}).start();

}

}

}

**@Override**

**protected** **void** **onStart**() {

**super**.onStart();

backgroundThread = **new** BackgroundThread(**this**);

backgroundThread.start();

}

**@Override**

**protected** **void** **onStop**() {

**super**.onStop();

**if**(backgroundThread!=**null**){

backgroundThread.stop();

backgroundThread = **null**;

}

}

**@Override**

**protected** **void** **onDestroy**() {

**super**.onDestroy();

**if**(backgroundThread!=**null**){

backgroundThread.destroy();

backgroundThread = **null**;

}

}

}

BackgroundThread

**package** com.example.surfaceviewexample;

**public** **class** **BackgroundThread** **extends** **Thread** {

*//It knows about only MainActivity and call this.parent.draw*

MainActivity parent;

**int** index = 1;

**public** **BackgroundThread**(MainActivity parent) {

**this**.parent = parent;

}

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

**try** {

Thread.sleep(1000);

} **catch** (InterruptedException e1) {

e1.printStackTrace();

}

**while** (**true**) {

**this**.parent.draw(index++);

**try** {

Thread.sleep(1000);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

}

MySurfaceView

**package** com.example.surfaceviewexample;

**import** java.util.ArrayList;

**import** android.content.Context;

**import** android.graphics.Canvas;

**import** android.graphics.Color;

**import** android.graphics.Paint;

**import** android.view.SurfaceHolder;

**import** android.view.SurfaceView;

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

DrawingThread drawingThread;

ArrayList<Integer> stack = **new** ArrayList<Integer>();

**public** **int** JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT = 6;*///1,2,3,4,5,6*

*//1 - start drawingThread.draw - drawingThread is only one - not work - canvas blinks*

*//1 - запуск в лоб drawingThread.draw - drawingThread один - экран мерцает четный/нечетный*

*//2 - start drawingThread.draw in new Thread - it may be very difficult for comprehension - drawingThread is only one - not work - canvas blinks*

*//2 - запуск в новой нити исполнения drawingThread.draw - это сложно для понимания, но кто-то так просто посоветовал попробовать - drawingThread один - экран мерцает четный/нечетный*

*//3 - create every time new DrawingThread and start him - drawingThread not declared and there are many DrawingThread objects - not work - canvas blinks*

*//3 - каждый раз создается и стартуется новый DrawingThread - drawingThread вообще не объявляется - экран мерцает четный/нечетный*

*//4 - using existing method "post" of SurfaceView of Runnable interface, see MySurfaceView.draw case 4 - экран мерцает четный/нечетный*

*//4 - используется существующий метод "post" суперкласса SurfaceView, см. MySurfaceView.draw вариант 4 - экран мерцает четный/нечетный*

*//5 - work well - there is one DrawingThread object and it have to store stack of all drawing Objects - I dont't like existing stack and idling process, not call on demand*

*//5 - работает по интерфейсу пользователя как надо - и drawingThread только один, и есть стек отображаемых объектов - Мне не нравится, что для того чтоб дорисовать объект, нужно хранить и перерисовывать весь стек, есть холостой ход фонового потока рисования, а хочется отрисовку объекта по требованию*

*//6 - work well - but there are more DrawingThread objects and it have to store stack of all drawing Objects - I dont't like existing stack. There is call on demand, although some people say that it is no good use multithread access to UI.*

*//6 - работает по интерфейсу пользователякак надо - каждый раз создается и стартуется новый DrawingThread, и есть стек всех отображаемых объектов - Мне не нравится, что для того чтоб дорисовать объект, нужно хранить и перерисовывать весь стек. Но реализована отрисовка по требованию, хотя многие разработчики предостеригают от создания множества нитей, используемых для рисования на одной канве.*

*//7 шутки ради*

**public** **MySurfaceView**(Context context){

**super**(context);

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

}

**public** **void** **surfaceChanged**(SurfaceHolder holder, **int** format, **int** width,

**int** height) {

}

*//BackgroundThread knows about only MainActivity and call this.parent.draw*

*//but MainActivity knows about MySurfaceView and delegate draw to its "draw" method*

*//but MySurfaceView knows how to draw object well, but delegates draw to "DrawingThread.draw" method*

**public** **void** **draw**(**int** index){

**switch**(**this**.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(4):{

drawingThread.draw(index);

**break**;

}

**case**(2):{

**final** **int** i = index;

**new** Thread(**new** Runnable(){

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

drawingThread.draw(i);

}

}).start();

**break**;

}

**case**(3):{

(**new** DrawingThread(getHolder(),**this**,index)).start();

**break**;

}

**case**(5):{

stack.add(index);

**break**;

}

**case**(6):{

stack.add(index);

(**new** DrawingThread(getHolder(),**this**)).start();

**break**;

}

}

}

**public** **void** **draw**(Canvas canvas, **int** number){

*//super.onDraw(canvas);*

**switch**(**this**.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(2):

**case**(3):

**case**(4):{

Paint p = **new** Paint();

p.setTextSize(canvas.getHeight()/50);

p.setColor(Color.RED);

canvas.drawText(""+number, p.getTextSize(), p.getTextSize()\*(number), p);

**break**;

}

**case**(5):

**case**(6):{

*//in case 5 variable "number" is not using - this is stack*

Paint p = **new** Paint();

p.setTextSize(canvas.getHeight()/50);

p.setColor(Color.RED);

**for**(Integer int\_item : stack){

canvas.drawText(""+int\_item, p.getTextSize(), p.getTextSize()\*(int\_item), p);

}

**break**;

}

}

}

**public** **void** **surfaceCreated**(SurfaceHolder holder) {

**switch**(**this**.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(2):

**case**(4):

**case**(5):{

drawingThread = **new** DrawingThread(getHolder(),**this**);

drawingThread.start();

**break**;

}

**case**(3):

**case**(6):{

*//Nothing else matter*

**break**;

}

}

}

**public** **void** **surfaceDestroyed**(SurfaceHolder holder) {

**switch**(**this**.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(2):

**case**(4):

**case**(5):{

drawingThread.destroy();

**break**;

}

**case**(3):

**case**(6):{

*//Nothing else matter*

**break**;

}

}

}

}

DrawingThread

**package** com.example.surfaceviewexample;

**import** android.graphics.Canvas;

**import** android.graphics.Color;

**import** android.graphics.Paint;

**import** android.view.SurfaceHolder;

**public** **class** **DrawingThread** **extends** **Thread** {

SurfaceHolder surfaceHolder;

MySurfaceView mySurfaceView;

**int** number = 0;

**public** **DrawingThread**(SurfaceHolder surfaceHolder,MySurfaceView mySurfaceView){

**super**();

**this**.surfaceHolder = surfaceHolder;

**this**.mySurfaceView = mySurfaceView;

}

**public** **DrawingThread**(SurfaceHolder surfaceHolder,MySurfaceView mySurfaceView,**int** number){

**super**();

**this**.surfaceHolder = surfaceHolder;

**this**.mySurfaceView = mySurfaceView;

**this**.number = number;

}

*//BackgroundThread knows about only MainActivity and call this.parent.draw*

*//but MainActivity knows about MySurfaceView and delegate draw to its "draw" method*

*//but MySurfaceView knows how to draw object well, but delegates draw to "DrawingThread.draw" method*

*//DrawingThread dose not know how to draw object well, it blocks canvas and call necessary MySurfaceView.draw(Canvas canvas, int number)*

**public** **void** **draw**(**int** number){

**this**.number = number;

**this**.draw();

}

**protected** **void** **draw**(){

Canvas canvas = **this**.surfaceHolder.lockCanvas();

**try**{

**synchronized** (surfaceHolder) {

*//in case 5 or 6 variable "this.number" is not using - this is stack*

mySurfaceView.draw(canvas, **this**.number);

}

}

**finally**{

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

surfaceHolder.unlockCanvasAndPost(canvas);

}

}

}

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

**switch**(mySurfaceView.JUST\_CHANGE\_DEFAULT\_DRAWING\_LOGIC\_VARIANT){

**case**(1):

**case**(2):{

**while**(**true**){}

}

**case**(3):{

**this**.draw();

**break**;

}

**case**(4):{

*//Nothing else matter*

**break**;

}

**case**(5):{

*//Nothing else matter*

**while**(**true**){**this**.draw();}

}

**case**(6):{

*//Nothing else matter*

**this**.draw();

**break**;

}

}

}

}

Если кто-то все же будет разгребать и у него не пойдет, пишите в личку, скину исходник архивом.