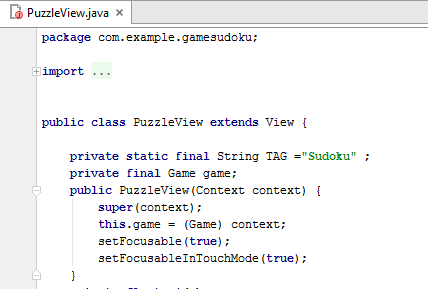
**Họ và tên: Tăng Xuân Phượng Hoàng**

**Bài tập lớn: Xây dựng game Sodoku trên Android**

**Phần: Xử lí chạm cho game Sodoku lập trình trên Android.**

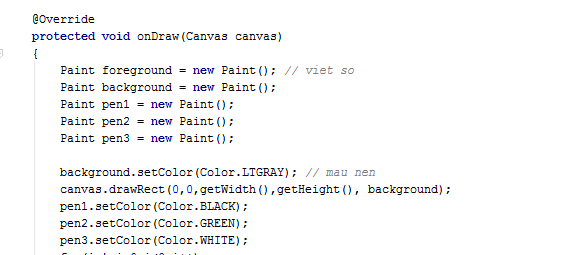
**-** onTouch là khái niệm nâng cao trong lập trình Android (Lập trình cảm ứng đa điểm), cho phép chúng ta điều khiển các cử chỉ chạm vào màn hình như phóng to-thu nhỏ và nhiều thao tác khác liên quan đến cảm ứng.

-Trong file PuzzleView.java, chúng ta tạo một class PuzzleView kề thừa lớp View

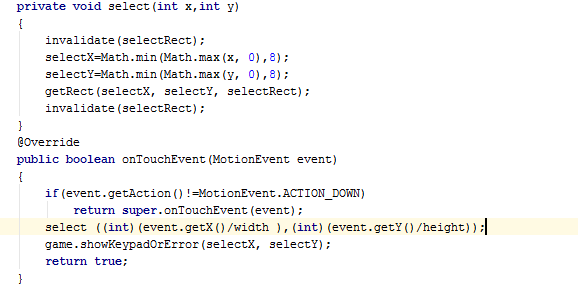


-Sau đó ta gọi hàm onDraw có chức năng vẽ các điểm đánh dấu khi chạm nên màn hình:

+ta khai báo các hàm Paint là pen1,pen2,pen3 (Hàm này vẽ ra các điểm đánh dấu khi chậm nên màn hình)



-Tiếp theo ta gọi hàm onTouchEvent. Hàm này sẽ nấy ra tất cả các sự kiện cảm ứng khi ta chạm nên màn hình.



+ int x:tọa độ x khi chạm tay

+int y: tọa độ y khi chạm tay

+ACTION\_DOWN: sự bắt đầu của cảm ứng

+getAction() : Lấy một giá trị int cho các loại hành động

+GetX(): Lấy một giá trị int cho vị trí tọa độ X của cảm ứng

+GetY():Lấy một giá trị int cho vị trí tọa độ Y của cảm ứng

Sau đó ta sẽ ghi đè lên hàm onTouchEvent. Đối tượng event sẽ có hàm getX và getY cho ta lấy vị trí tọa độ được chạm

+Trong chương trình khi ta chạm bất kì vị trí nào trong bàn cờ (9,9) game sodoku thì dữ liệu được hiện thị ra từ vị trí đó.

-Toàn bộ code PuzzleView.java như sau:

**package** com.example.gamesudoku;  
  
**import** android.R.color;  
**import** android.content.Context;  
**import** android.graphics.Canvas;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.Rect;  
**import** android.util.Log;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
**import** android.graphics.Paint.FontMetrics;  
**import** android.graphics.Paint.Style;  
  
  
**public class** PuzzleView **extends** View {  
  
 **private static final** String TAG =**"Sudoku"** ;  
 **private final** Game game;   
 **public** PuzzleView(Context context) {  
 **super**(context);  
 **this**.game = (Game) context;  
 setFocusable(**true**);  
 setFocusableInTouchMode(**true**);  
 }  
 **private float** width;  
 **private float** height;  
 **private int** selectX;  
 **private int** selectY;  
 **private final** Rect selectRect = **new** Rect();  
   
 **private void** select(**int** x,**int** y)  
 {  
 invalidate(selectRect);  
 selectX=Math.min(Math.max(x, 0),8);  
 selectY=Math.min(Math.max(y, 0),8);  
 getRect(selectX, selectY, selectRect);  
 invalidate(selectRect);  
 }  
 @Override  
 **public boolean** onTouchEvent(MotionEvent event)   
 {  
 **if**(event.getAction()!=MotionEvent.ACTION\_DOWN)  
 **return super**.onTouchEvent(event);  
 select ((**int**)(event.getX()/width ),(**int**)(event.getY()/height));  
 game.showKeypadOrError(selectX, selectY);  
 **return true**;  
 }  
   
 @Override   
 **protected void** onSizeChanged(**int** w,**int** h, **int** oldw,**int** oldh)  
 {  
 width = w/9f;  
 height=h/9f;  
 getRect(selectX, selectY, selectRect);  
 **super**.onSizeChanged(w, h, oldw, oldh);  
   
 }  
 **public void** setSelectedTile(**int** tile) {  
 **if** (game.setTileIfValid(selectX, selectY, tile)) {  
 invalidate();  
 }   
 **else** {  
 Log.d(TAG, **"setSelectedTile : invalid "** + tile );  
 }  
  
 }  
 **private void** getRect(**int** x,**int** y , Rect rect)  
 {  
 rect.set((**int**)(x\*width), (**int**)(y\*height),(**int**)(x\*width + width),(**int**) (y\*height + height));  
 }  
  
 @Override  
 **protected void** onDraw(Canvas canvas)  
 {  
 Paint foreground = **new** Paint(); *// viet so* Paint background = **new** Paint();  
 Paint pen1 = **new** Paint();  
 Paint pen2 = **new** Paint();  
 Paint pen3 = **new** Paint();  
   
 background.setColor(Color.LTGRAY); *// mau nen* canvas.drawRect(0,0,getWidth(),getHeight(), background);  
 pen1.setColor(Color.BLACK);   
 pen2.setColor(Color.GREEN);  
 pen3.setColor(Color.WHITE);  
 **for**(**int** i=0;i<9;i++)  
 {  
 canvas.drawLine(0, i\*height, getWidth(), i\*height, pen3);  
 canvas.drawLine(0, i\*height+1, getWidth(), i\*height+1, pen2);  
 canvas.drawLine(i\*width, 0, i\*width,getHeight(), pen3);  
 canvas.drawLine(i\*width+1, 0, i\*width+1,getHeight(), pen2);  
 }  
 pen1.setStrokeWidth(5);   
 **for**(**int** i=0;i<9;i++)  
 {  
 **if**(i%3!=0) **continue**;  
 canvas.drawLine(0, i\*height, getWidth(), i\*height, pen1);  
 canvas.drawLine(0, i\*height+1, getWidth(), i\*height+1, pen2);  
 canvas.drawLine(i\*width, 0, i\*width,getHeight(), pen1);  
 canvas.drawLine(i\*width+1, 0, i\*width+1,getHeight(), pen2);  
 }  
 foreground = **new** Paint(Paint.ANTI\_ALIAS\_FLAG);  
 foreground.setTextSize(70);  
 foreground.setColor(Color.BLUE);  
 foreground.setStyle(Style.FILL);  
 foreground.setTextAlign(Paint.Align.CENTER);  
 FontMetrics fm = foreground.getFontMetrics();  
 **float** x = width / 2;  
 **float** y = height / 2 - (fm.ascent + fm.descent) / 2;  
 **for** (**int** i = 0; i < 9; i++) {  
 **for** (**int** j = 0; j < 9; j++) {  
 canvas.drawText(**this**.game.getTitleString(i, j), i \* width + x,  
 j \* height + y, foreground);  
 }  
 }  
   
 foreground = **new** Paint(Paint.ANTI\_ALIAS\_FLAG);  
 foreground.setTextSize(70);  
 foreground.setColor(Color.BLACK);  
 foreground.setStyle(Style.FILL);  
 foreground.setTextAlign(Paint.Align.CENTER);  
   
 Paint select = **new** Paint();  
 select.setColor(Color.argb(64, 255, 80, 0));  
 canvas.drawRect(selectRect, select);  
 }  
}