Mobile App Development

Exercise 3

Q1. Develop a simple application to draw the basic graphical primitives on the screen

MainActivity.java:

```
package com.example.paint;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import com.pes.androidmaterialcolorpickerdialog.ColorPicker;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    ColorPicker cp;
    MyCanvas canvas;
    private static final int STORAGE_PERMISSION_CODE = 101;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        cp = new ColorPicker(MainActivity.this, 0, 0, 0);
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);
        canvas = new MyCanvas(this, cp, dm.widthPixels, dm.heightPixels);
        checkPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE,
STORAGE_PERMISSION_CODE);
        checkPermission(Manifest.permission.READ_EXTERNAL_STORAGE,
STORAGE_PERMISSION_CODE);
        setContentView(canvas);
    }

    public void checkPermission(String permission, int requestCode) {
        if (ContextCompat.checkSelfPermission(MainActivity.this, permission)
== PackageManager.PERMISSION_DENIED) {
            ActivityCompat.requestPermissions(MainActivity.this, new
String[]{permission}, requestCode);
        } else {
            Toast.makeText(MainActivity.this, "Permission already granted",
```

```
Toast.LENGTH_SHORT).show();
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        if (requestCode == STORAGE_PERMISSION_CODE) {
            if (grantResults.length > 0
                    && grantResults[0] == PackageManager.PERMISSION_GRANTED)
{
                Toast.makeText(MainActivity.this, "Storage Permission
Granted", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(MainActivity.this, "Storage Permission
Denied", Toast.LENGTH_SHORT).show();
            }
        }
    }


}
```

MyCanvas.java

```
package com.example.paint;

import android.annotation.SuppressLint;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Path;
import android.graphics.Typeface;
import android.view.MotionEvent;
import android.view.View;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import com.pes.androidmaterialcolorpickerdialog.ColorPicker;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Objects;
```

```
@SuppressLint("ViewConstructor")
public class MyCanvas extends View {


    ColorPicker cp;
    int height, width, height_percent, box_width, border, opr, touch_event,
color, curr shape, paint stroke, text size = 0;
    float touch_start_x, touch_start_y, touch_x, touch_y, paint_width,
box width float;
    List<List<Number>> buttons;
    List<List<Number>> history;

    public MyCanvas(Context context, ColorPicker atr_cp, int width, int
height) {
        super(context);
        Paint paint;
        paint = new Paint();
        paint.setAntiAlias(true);
        color = -16777216;
        paint.setColor(color);
        paint.setStrokeJoin(Paint.Join.ROUND);
        paint.setStrokeWidth(8f);
        cp = atr_cp;
        this.height = height;
        this.width = width;
        this.height percent = this.border = height / 100;
        this.box_width = (width - (border * 12)) / 6;
        box_width_float = ((float) width - (((float) height / 100) * 12)) /
6;
        text_size = 100;
        paint.setTypeface(Typeface.create("Arial", Typeface.BOLD));
        paint.setTextSize(text_size);
        while (paint.measureText("Clear") > (box_width - box_width * 0.1)) {
            text size = text size - 5;
            paint.setTextSize(text_size);
        }
        history = new ArrayList<>();
        buttons = new ArrayList<>();
        List<Number> shape;
        paint_stroke = 1;
        paint width = 5f;
        //Row 1
        shape = Arrays.asList(1, 0, 0, width, height percent * 10, -16777216,
paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, border, border, border + box_width, border +
height_percent * 8, -1, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 3) + box_width, border, (border *
3) + (box_width * 2), border + height_percent * 8, -1, paint_stroke,
paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 5) + (box_width * 2), border,
(border * 5) + (box_width * 3), border + height_percent * 8, -1,
paint stroke, paint width);
        buttons.add(shape);
```

```
        shape = Arrays.asList(1, (border * 7) + (box_width * 3), border,
(border * 7) + (box_width * 4), border + height_percent * 8, -1,
paint stroke, paint width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 9) + (box width * 4), border,
(border * 9) + (box width * 5), border + height percent * 8, -1,
paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 11) + (box_width * 5), border,
(border * 11) + (box width * 6), border + height percent * 8, -1,
paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, border, border, (border) + (box width / 2),
border + height_percent * 4, -65536, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border) + (box_width / 2), border, (border)
+ (box_width), border + height_percent * 4, -16711936, paint_stroke,
paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, border, border + height percent * 4,
(border) + (box_width), border + height_percent * 8, -256, paint_stroke,
paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border) + (box_width / 2), border +
height percent * 4, (border) + (box width), border + height percent * 8, -
16776961, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, border * 3 + box_width + box_width / 10,
border + height_percent, border * 3 + box_width * 2 - box_width / 10, border
+ height_percent * 7, -16777216, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(2, border * 5 + box_width * 2 + box_width / 2,
border + height_percent * 4, (int) ((height_percent * 3.5 + (float)
(box_width - box_width / 5) / 2) / 2), -16777216, paint_stroke, paint_width);
        buttons.add(shape);

        //Row 2
        shape = Arrays.asList(1, 0, height_percent * 10, width,
height_percent * 20, -16777216, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, border, border + height_percent * 10, border
+ box width, border + height percent * 18, -1, paint stroke, paint width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 3) + box width, border +
height_percent * 10, (border * 3) + (box_width * 2), border + height_percent
* 18, -1, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 5) + (box_width * 2), border +
height percent * 10, (border * 5) + (box width * 3), border + height percent
* 18, -1, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 7) + (box_width * 3), border +
height_percent * 10, (border * 7) + (box_width * 4), border + height_percent
* 18, -1, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 9) + (box width * 4), border +
height_percent * 10, (border * 9) + (box_width * 5), border + height_percent
```

```
* 18, -26880, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(1, (border * 11) + (box_width * 5), border +
height_percent * 10, (border * 11) + (box_width * 6), border + height_percent
* 18, -1, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(3, border + box_width / 10, border +
height_percent * 11, border + box_width - box_width / 10, border +
height_percent * 17, -16777216, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(4, (border * 3) + box_width + box_width / 10,
border + height_percent * 12, (border * 3) + (box_width * 2) - box_width /
10, border + height_percent * 16, -16777216, paint_stroke, paint_width);
        buttons.add(shape);
        shape = Arrays.asList(5, (border * 5) + (box_width * 2) + box_width /
10, border + height_percent * 14, (border * 5) + (box_width * 3) - box_width
/ 10, border + height_percent * 14, -16777216, paint_stroke, paint_width);
        buttons.add(shape);
    }

    @SuppressLint("DrawAllocation")
    protected void onDraw(Canvas canvas) {
        List<Number> shape;
        super.onDraw(canvas);
        int save_flag = 0;
        switch (touch_event) {
            case 1:
                if (touch_y > height_percent * 10) {
                    touch_start_x = touch_x;
                    touch_start_y = touch_y;
                }
                break;
            case 2:
                if (touch_y > height_percent * 10) {
                    if (opr == 0) {
                        if (touch_start_y > height_percent * 20) {
                            shape = Arrays.asList(0, touch_start_x,
touch_start_y, touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(shape);
                        }
                        touch_start_x = touch_x;
                        touch_start_y = touch_y;
                    } else if (opr == 1) {
                        try {
                            history.get(curr_shape).set(3, touch_x);
                            history.get(curr_shape).set(4, touch_y);
                        } catch (Exception e) {
                            shape = Arrays.asList(1, touch_x, touch_y,
touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(curr_shape, shape);
                        }
                    } else if (opr == 2) {
                        float radius = ((float)
Math.sqrt(Math.pow(touch_start_x - touch_x, 2) + Math.pow(touch_start_y -
touch_y, 2))) / 2;
                        try {
                            history.get(curr_shape).set(1, (touch_x +
```

```
touch_start_x) / 2);
                            history.get(curr_shape).set(2, (touch_y +
touch start y) / 2);
                            history.get(curr_shape).set(3, radius);
                        } catch (Exception e) {
                            shape = Arrays.asList(2, (touch x + touch x) / 2,
(touch_y + touch_y) / 2, radius, color, paint_stroke, paint_width);
                            history.add(curr shape, shape);
                        }
                    } else if (opr == 3) {
                        try {
                            history.get(curr_shape).set(3, touch_x);
                            history.get(curr shape).set(4, touch y);
                        } catch (Exception e) {
                            shape = Arrays.asList(3, touch x, touch y,
touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(curr_shape, shape);
                        }

                    } else if (opr == 4) {
                        try {
                            history.get(curr_shape).set(3, touch_x);
                            history.get(curr_shape).set(4, touch_y);
                        } catch (Exception e) {
                            shape = Arrays.asList(4, touch x, touch y,
touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(curr shape, shape);
                        }

                    } else if (opr == 5) {
                        try {
                            history.get(curr_shape).set(3, touch_x);
                            history.get(curr_shape).set(4, touch_y);
                        } catch (Exception e) {
                            shape = Arrays.asList(5, touch x, touch y,
touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(curr_shape, shape);
                        }

                    } else if (opr == 6) {
                        try {
                            history.get(curr shape).set(3, touch x);
                            history.get(curr_shape).set(4, touch_y);
                        } catch (Exception e) {
                            shape = Arrays.asList(6, touch_x, touch_y,
touch_x, touch_y, color, paint_stroke, paint_width);
                            history.add(curr_shape, shape);
                        }

                    }

                }
                break;
            case 3:
                if (touch_y < height_percent * 10) {
                    if (touch x < (box width + border * 2)) {
                        cp.show();
```

```
                        cp.setCallback(clr -> {
                            color = clr;
                            cp.dismiss();
                        });
                        Toast.makeText(getContext(), "Color Picker",
Toast.LENGTH_SHORT).show();
                    } else if (touch_x > (box_width + border * 2) && touch_x
< (box_width * 2 + border * 4)) {
                        if (opr == 1) {
                            opr = 0;
                            buttons.get(2).set(5, -1);
                        } else {
                            opr = 1;
                            Toast.makeText(getContext(), "Rectangle",
Toast.LENGTH_SHORT).show();
                            curr_shape = history.size();
                            buttons.get(2).set(5, -26880);
                        }
                        buttons.get(14).set(5, -1);
                        buttons.get(15).set(5, -1);
                        buttons.get(3).set(5, -1);
                        buttons.get(16).set(5, -1);
                        buttons.get(17).set(5, -1);
                    } else if (touch_x > (box_width * 2 + border * 4) &&
touch_x < (box_width * 3 + border * 6)) {
                        if (opr == 2) {
                            opr = 0;
                            buttons.get(3).set(5, -1);
                        } else {
                            opr = 2;
                            Toast.makeText(getContext(), "Circle",
Toast.LENGTH_SHORT).show();
                            curr_shape = history.size();

                            buttons.get(3).set(5, -26880);
                        }
                        buttons.get(14).set(5, -1);
                        buttons.get(15).set(5, -1);
                        buttons.get(2).set(5, -1);
                        buttons.get(16).set(5, -1);
                        buttons.get(17).set(5, -1);
                    } else if (touch_x > (box_width * 3 + border * 6) &&
touch_x < (box_width * 4 + border * 8)) {
                        try {
                            history.remove(history.size() - 1);
                            if (opr == 1 || opr == 2) {
                                curr_shape--;
                            }
                        } catch (Exception ignored) {
                        }
                        Toast.makeText(getContext(), "Undo",
Toast.LENGTH_SHORT).show();
                    } else if (touch_x > (box_width * 4 + border * 2) &&
touch_x < (box_width * 5 + border * 10)) {
                        save_flag = 1;
                    } else if (touch_x > (box_width * 5 + border * 2)) {
                        Toast.makeText(getContext(), "Clear",
```

```
Toast.LENGTH_SHORT).show();
                        curr_shape = 0;
                        history = new ArrayList<>();
                    }

                } else if (touch_y < height_percent * 20) {
                    if (touch_x < (box_width + border * 2)) {
                        if (opr == 3) {
                            opr = 0;
                            buttons.get(14).set(5, -1);
                        } else {
                            opr = 3;
                            Toast.makeText(getContext(), "Round Rectangle",
Toast.LENGTH_SHORT).show();
                            curr_shape = history.size();
                            buttons.get(14).set(5, -26880);
                        }
                        buttons.get(2).set(5, -1);
                        buttons.get(3).set(5, -1);
                        buttons.get(15).set(5, -1);
                        buttons.get(16).set(5, -1);
                        buttons.get(17).set(5, -1);
                    } else if (touch_x > (box_width + border * 2) && touch_x
< (box_width * 2 + border * 4)) {
                        if (opr == 4) {
                            opr = 0;
                            buttons.get(15).set(5, -1);
                        } else {
                            opr = 4;
                            Toast.makeText(getContext(), "Oval",
Toast.LENGTH_SHORT).show();
                            curr_shape = history.size();
                            buttons.get(15).set(5, -26880);
                        }
                        buttons.get(2).set(5, -1);
                        buttons.get(3).set(5, -1);
                        buttons.get(14).set(5, -1);
                        buttons.get(16).set(5, -1);
                        buttons.get(17).set(5, -1);
                    } else if (touch_x > (box_width * 2 + border * 4) &&
touch_x < (box_width * 3 + border * 6)) {
                        if (opr == 5) {
                            opr = 0;
                            buttons.get(16).set(5, -1);
                        } else {
                            opr = 5;
                            Toast.makeText(getContext(), "Line",
Toast.LENGTH_SHORT).show();
                            curr_shape = history.size();

                            buttons.get(16).set(5, -26880);
                        }
                        buttons.get(14).set(5, -1);
                        buttons.get(15).set(5, -1);
                        buttons.get(17).set(5, -1);
                        buttons.get(2).set(5, -1);
                        buttons.get(3).set(5, -1);
```

```
                    } else if (touch_x > (box_width * 3 + border * 6) &&
touch_x < (box_width * 4 + border * 8)) {
                        if (opr == 6) {
                            opr = 0;
                            buttons.get(17).set(5, -1);
                        } else {
                            opr = 6;
                            Toast.makeText(getContext(), "Star",
Toast.LENGTH_SHORT).show();
                            curr shape = history.size();

                            buttons.get(17).set(5, -26880);
                        }
                        buttons.get(14).set(5, -1);
                        buttons.get(15).set(5, -1);
                        buttons.get(16).set(5, -1);
                        buttons.get(2).set(5, -1);
                        buttons.get(3).set(5, -1);
                    } else if (touch_x > (box_width * 4 + border * 8) &&
touch x < (box width * 5 + border * 10)) {
                        if (paint_stroke == 1) {
                            paint_stroke = 0;
                            buttons.get(18).set(5, -1);
                        } else {
                            paint stroke = 1;
                            Toast.makeText(getContext(), "Fill Shapes",
Toast.LENGTH SHORT).show();
                            buttons.get(18).set(5, -26880);
                        }
                    } else if (touch_x > (box_width * 5 + border * 10) &&
touch_x < (box_width * 6 + border * 12)) {
                        final AlertDialog.Builder popDialog = new
AlertDialog.Builder(getContext());
                        final SeekBar seek = new SeekBar(getContext());
                        seek.setMax(100);
                        seek.setProgress((int) paint_width);
                        popDialog.setIcon(android.R.drawable.ic_menu_edit);
                        popDialog.setTitle("Select Brush Size : 1-100 ");
                        popDialog.setView(seek);
                        seek.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
                            public void onProgressChanged(SeekBar seekBar,
int progress, boolean fromUser) {
                                paint width = progress;
                            }

                            public void onStartTrackingTouch(SeekBar arg0) {
                            }

                            public void onStopTrackingTouch(SeekBar seekBar)
{
                                Toast.makeText(getContext(), "Size :
"+paint_width, Toast.LENGTH_SHORT).show();
                            }
                        });
                        popDialog.setPositiveButton("OK", new
DialogInterface.OnClickListener() {
```

```
                            public void onClick(DialogInterface dialog, int
which) {
                                dialog.dismiss();
                                Toast.makeText(getContext(), "Size :
"+paint width, Toast.LENGTH SHORT).show();
                            }
                        });
                        popDialog.create();
                        popDialog.show();

                    }
                } else {
                    if (opr != 0) {
                        curr_shape++;
                    }
                }
                break;
        }

        //Canvasing
        List<List<Number>> main = new ArrayList<>();
        main.addAll(history);
        main.addAll(buttons);

        for (int i = 0; i < main.size(); i++) {
            Paint paint = new Paint();
            shape = main.get(i);
            if ((int) shape.get(0) == 1) {
                paint.setColor((int) shape.get(5));
                paint.setStrokeWidth((float) shape.get(7));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                try {
                    canvas.drawRect((float) shape.get(1), (float)
shape.get(2), (float) shape.get(3), (float) shape.get(4), paint);
                } catch (ClassCastException e) {
                    canvas.drawRect((int) shape.get(1), (int) shape.get(2),
(int) shape.get(3), (int) shape.get(4), paint);
                }
            }

            if ((int) shape.get(0) == 2) {
                paint.setColor((int) shape.get(4));
                paint.setStrokeWidth((float) shape.get(6));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(5) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                try {
                    canvas.drawCircle((float) shape.get(1), (float)
shape.get(2), (float) shape.get(3), paint);
```

```
                } catch (ClassCastException e) {
                    canvas.drawCircle((int) shape.get(1), (int) shape.get(2),
(int) shape.get(3), paint);
                }
            }

            if ((int) shape.get(0) == 0) {
                paint.setColor((int) shape.get(5));
                paint.setStrokeWidth((float) shape.get(7));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                canvas.drawLine((float) shape.get(1), (float) shape.get(2),
(float) shape.get(3), (float) shape.get(4), paint);
            }
            if ((int) shape.get(0) == 3) {
                paint.setColor((int) shape.get(5));
                paint.setStrokeWidth((float) shape.get(7));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                try {
                    float xr = Math.abs((float) shape.get(3) - (float)
shape.get(1)) / 10;
                    float yr = Math.abs((float) shape.get(4) - (float)
shape.get(2)) / 10;
                    canvas.drawRoundRect((float) shape.get(1), (float)
shape.get(2), (float) shape.get(3), (float) shape.get(4), xr, yr, paint);
                } catch (ClassCastException e) {
                    int xr = Math.abs((int) shape.get(3) - (int)
shape.get(1)) / 10;
                    int yr = Math.abs((int) shape.get(4) - (int)
shape.get(2)) / 10;
                    canvas.drawRoundRect((int) shape.get(1), (int)
shape.get(2), (int) shape.get(3), (int) shape.get(4), xr, yr, paint);
                }
            }
            if ((int) shape.get(0) == 4) {
                paint.setColor((int) shape.get(5));
                paint.setStrokeWidth((float) shape.get(7));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                try {
                    canvas.drawOval((float) shape.get(1), (float)
shape.get(2), (float) shape.get(3), (float) shape.get(4), paint);
                } catch (ClassCastException e) {
                    canvas.drawOval((int) shape.get(1), (int) shape.get(2),
```

```
(int) shape.get(3), (int) shape.get(4), paint);
                }
            }
            if ((int) shape.get(0) == 5) {
                paint.setColor((int) shape.get(5));
                paint.setStrokeWidth((float) shape.get(7));
                paint.setStrokeJoin(Paint.Join.ROUND);
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                } else {
                    paint.setStyle(Paint.Style.FILL);
                }
                try {
                    canvas.drawLine((float) shape.get(1), (float)
shape.get(2), (float) shape.get(3), (float) shape.get(4), paint);
                } catch (ClassCastException e) {
                    canvas.drawLine((int) shape.get(1), (int) shape.get(2),
(int) shape.get(3), (int) shape.get(4), paint);
                }
            }
            if ((int) shape.get(0) == 6) {
                paint.setColor((int) shape.get(5));
                float paint_stroke_width = (float) shape.get(7);
                paint.setStrokeWidth(paint_stroke_width);
                paint.setStrokeJoin(Paint.Join.ROUND);
                Path star;
                float x, y, star width;
                x = Math.min((float) shape.get(1), (float) shape.get(3));
                star_width = Math.abs((float) shape.get(3) - (float)
shape.get(1));
                y = Math.min((float) shape.get(2), (float) shape.get(4));
                if ((int) shape.get(6) == 0) {
                    paint.setStyle(Paint.Style.STROKE);
                    star = createStarBySize(x, y, star_width, (int)
paint stroke width, 0);
                } else {
                    star = createStarBySize(x, y, star_width, (int)
paint_stroke_width, 1);
                    paint.setStyle(Paint.Style.FILL);
                }
                canvas.drawPath(star, paint);
            }

        }

        Paint paint = new Paint();
        //eraser
        paint.setColor(-16777216);
        paint.setStrokeJoin(Paint.Join.ROUND);
        paint.setTextSize(text_size);
        int text_start = (int) ((border * 7) + (box_width * 3) + ((box_width
/ 2) - (paint.measureText("Undo") / 2)));
        canvas.drawText("Undo", text_start, (float) (border + height_percent
* 4.5), paint);
        //size
        text start = (int) ((border * 9) + (box width * 4) + ((box width / 2)
- (paint.measureText("Save") / 2)));
```

```
        canvas.drawText("Save", text_start, (float) (border + height_percent
* 4.5), paint);
        //clear
        text_start = (int) ((border * 11) + (box_width * 5) + ((box_width /
2) - (paint.measureText("Clear") / 2)));
        canvas.drawText("Clear", text_start, (float) (border + height_percent
* 4.5), paint);
        text_start = (int) ((border * 11) + (box_width * 5) + ((box_width /
2) - (paint.measureText("Size") / 2)));
        canvas.drawText("Size", text_start, (float) (border + height_percent
* 14.5), paint);
        paint.setColor(color);
        text_start = (int) ((border * 9) + (box_width * 4) + ((box_width / 2)
- (paint.measureText("Fill") / 2)));
        canvas.drawText("Fill", text_start, (float) (border + height_percent
* 14.5), paint);
        paint.setColor(color);
        Path star = createStarBySize((border * 7) + (box_width * 3) +
(box_width_float / 10), (float) (border + height_percent * 10) +
(box_width_float / 10), (box_width_float * 8 / 10), 5, 1);
        canvas.drawPath(star, paint);

        if (save_flag == 1) {
            Bitmap bitmap = Bitmap.createBitmap(width, height,
Bitmap.Config.ARGB_8888);
            canvas.drawBitmap(bitmap, 0, height_percent * 10, paint);
            FileOutputStream fos;
            try {
                File file = new
File("/storage/emulated/0/paints/download.jpeg");
                Objects.requireNonNull(file.getParentFile());
                fos = new FileOutputStream(file);
                bitmap.compress(Bitmap.CompressFormat.JPEG, 100, fos);
                fos.flush();
                fos.close();
                Toast.makeText(getContext(), "PNG Saved",
Toast.LENGTH_SHORT).show();
            } catch (IOException e) {
                Toast.makeText(getContext(), "Permission Error",
Toast.LENGTH_SHORT).show();
            }
        }

    }

    @SuppressLint("ClickableViewAccessibility")
    public boolean onTouchEvent(MotionEvent event) {
        touch_x = event.getX();
        touch_y = event.getY();
        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
                touch_event = 1;
                break;
            case MotionEvent.ACTION_MOVE:
                touch_event = 2;
                break;
            case MotionEvent.ACTION_UP:
```

```
                touch_event = 3;
                break;
        }
        invalidate();
        return true;
    }

    private Path createStarBySize(float width, float height, float
star_width, int steps, int stroke) {
        float halfWidth = star width / 2.0F;
        float radius = halfWidth / 2.0F;
        float degreesPerStep = (float) Math.toRadians(360.0F / (float)
steps);
        float halfDegreesPerStep = degreesPerStep / 2.0F;
        Path star = new Path();
        if (stroke == 1) {
            star.setFillType(Path.FillType.EVEN_ODD);
        }
        float max = (float) (2.0F * Math.PI);
        star.moveTo(star width + width, halfWidth + height);
        for (double step = 0; step < max; step += degreesPerStep) {
            star.lineTo((float) (halfWidth + halfWidth * Math.cos(step)) +
width, (float) (halfWidth + halfWidth * Math.sin(step)) + height);
            star.lineTo((float) (halfWidth + radius * Math.cos(step +
halfDegreesPerStep)) + width, (float) (halfWidth + radius * Math.sin(step +
halfDegreesPerStep)) + height);
        }
        star.close();
        return star;
    }
}
```

Screenshot