## Experiment 6 - Write an application that draws basic graphical primitives on the screen

### FingerPath.java

```
package com.example.mypaint;

import android.graphics.Path;

public class FingerPath {
    public int color;
    public boolean emboss;
    public boolean blur;
    public int strokeWidth;
    public Path path;

    public FingerPath(int color, boolean emboss, boolean blur, int strokeWidth, Path
path) {
        this.color = color;
        this.emboss = emboss;
        this.blur = blur;
        this.strokeWidth = strokeWidth;
        this.path = path;
    }
}
```

### PaintView.java

```
package com.example.mypaint;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BlurMaskFilter;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.EmbossMaskFilter;
import android.graphics.MaskFilter;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.util.DisplayMetrics;
import android.view.MotionEvent;
import android.view.View;



import java.util.ArrayList;

public class PaintView extends View {
```

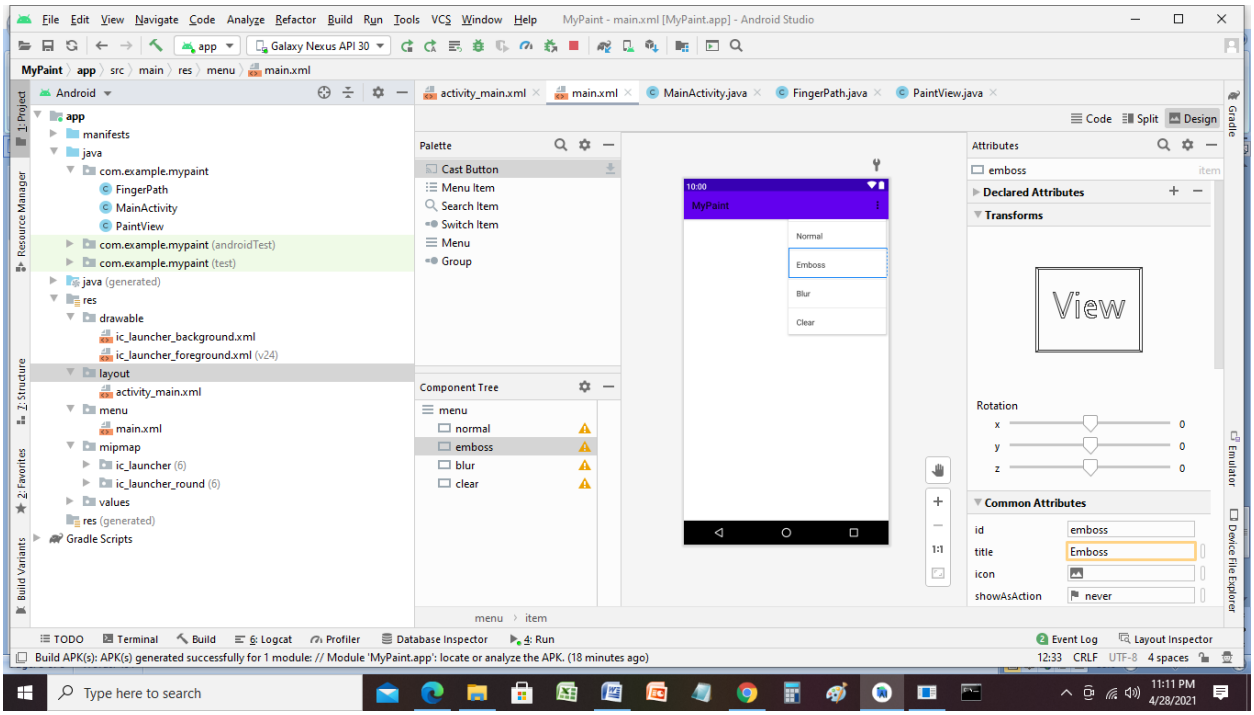
```
public static int BRUSH_SIZE = 20;
public static final int DEFAULT_COLOR = Color.BLUE ;
public static final int DEFAULT_BG_COLOR = Color.WHITE;
private static final float TOUCH_TOLERANCE = 4;
private float mX, mY;
private Path mPath;
private Paint mPaint;
private java.util.ArrayList<FingerPath>  paths = new ArrayList<>();
private int currentColor;
private int backgroundColor = DEFAULT_BG_COLOR;
private int strokeWidth;
private boolean emboss;
private boolean blur;
private android.graphics.MaskFilter  mEmboss;
private MaskFilter mBlur;
private Bitmap mBitmap;
private Canvas mCanvas;
private Paint mBitmapPaint = new Paint(Paint.DITHER_FLAG);

public PaintView(Context context) {
    this(context, null);
}

public PaintView(Context context, AttributeSet attrs) {
    super(context, attrs);
    mPaint = new Paint();
    mPaint.setAntiAlias(true);
    mPaint.setDither(true);
    mPaint.setColor(DEFAULT_COLOR);
    mPaint.setStyle(Paint.Style.STROKE);
    mPaint.setStrokeJoin(Paint.Join.ROUND);
    mPaint.setStrokeCap(Paint.Cap.ROUND);
    mPaint.setXfermode(null);
    mPaint.setAlpha(0xff);

    mEmboss = new EmbossMaskFilter(new float[] {1, 1, 1}, 0.4f, 6, 3.5f);
    mBlur = new BlurMaskFilter(5, BlurMaskFilter.Blur.NORMAL) ;
}

public void init(DisplayMetrics  metrics) {
    int height = metrics.heightPixels;
    int width = metrics.widthPixels;

    mBitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
    mCanvas = new Canvas(mBitmap);

    currentColor = DEFAULT_COLOR;
    strokeWidth = BRUSH_SIZE;
}

public void normal() {
    emboss = false;
    blur = false;
}
```

```
    public void emboss() {
        emboss = true;
        blur = false;
    }

    public void blur() {
        emboss = false;
        blur = true;
    }

    public void clear() {
        backgroundColor = DEFAULT_BG_COLOR;
        paths.clear();
        normal();
        invalidate();
    }

    @Override
    protected void onDraw(Canvas canvas) {

        canvas.save();
        mCanvas.drawColor(backgroundColor);

        for (FingerPath fp : paths) {
            mPaint.setColor(fp.color);
            mPaint.setStrokeWidth(fp.strokeWidth);
            mPaint.setMaskFilter(null);

            if (fp.emboss)
                mPaint.setMaskFilter(mEmboss);
            else if (fp.blur)
                mPaint.setMaskFilter(mBlur);

            mCanvas.drawPath(fp.path, mPaint);

        }

        canvas.drawBitmap(mBitmap, 0, 0, mBitmapPaint);
        canvas.restore();
    }

    private void touchStart(float x, float y) {
        mPath = new Path();
        FingerPath fp = new FingerPath(currentColor, emboss, blur, strokeWidth,
mPath);
        paths.add(fp);

        mPath.reset();
        mPath.moveTo(x, y);
        mX = x;
        mY = y;
    }

    private void touchMove(float x, float y) {
```

```
    float dx = Math.abs(x - mX);
    float dy = Math.abs(y - mY);

    if (dx >= TOUCH_TOLERANCE || dy >= TOUCH_TOLERANCE)
    {
        mPath.quadTo(mX, mY, (x + mX) / 2, (y + mY) / 2);
        mX = x;
        mY = y;
    }
}

private void touchUp() {
    mPath.lineTo(mX, mY);
}

@Override
public boolean dispatchTouchEvent(MotionEvent event) {
    float x = event.getX();
    float y = event.getY();

    switch(event.getAction()) {
        case MotionEvent.ACTION_DOWN :
            touchStart(x, y);
            invalidate();
            break;
        case MotionEvent.ACTION_MOVE :
            touchMove(x, y);
            invalidate();
            break;
        case MotionEvent.ACTION_UP :
            touchUp();
            invalidate();
            break;
    }

    return true;
}


}
```

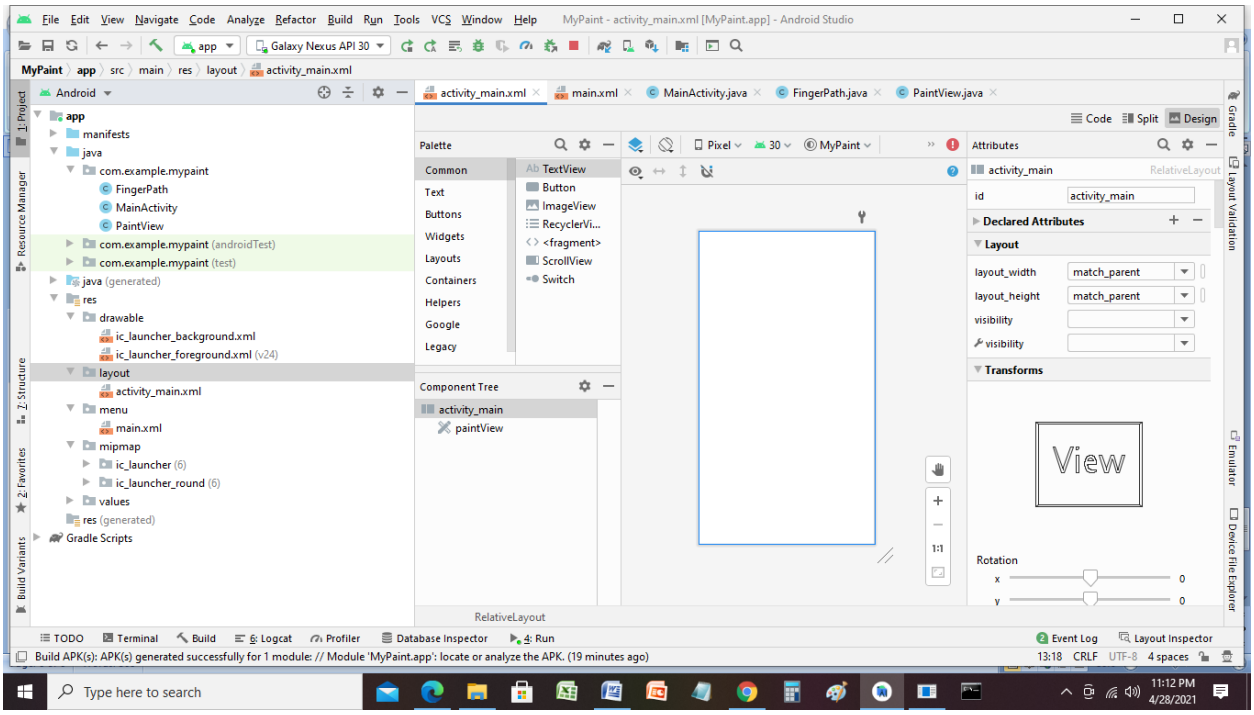
# main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto" >

    <item
        android:id="@+id/normal"
        app:showAsAction="never"
        android:title="Normal"
        />
    <item
        android:id="@+id/emboss"
        app:showAsAction="never"
        android:title="Emboss"
        />
    <item
        android:id="@+id/blur"
        app:showAsAction="never"
        android:title="Blur"
        />
    <item
        android:id="@+id/clear"
        app:showAsAction="never"
        android:title="Clear"
        />
</menu>
```



# activity_main.xml

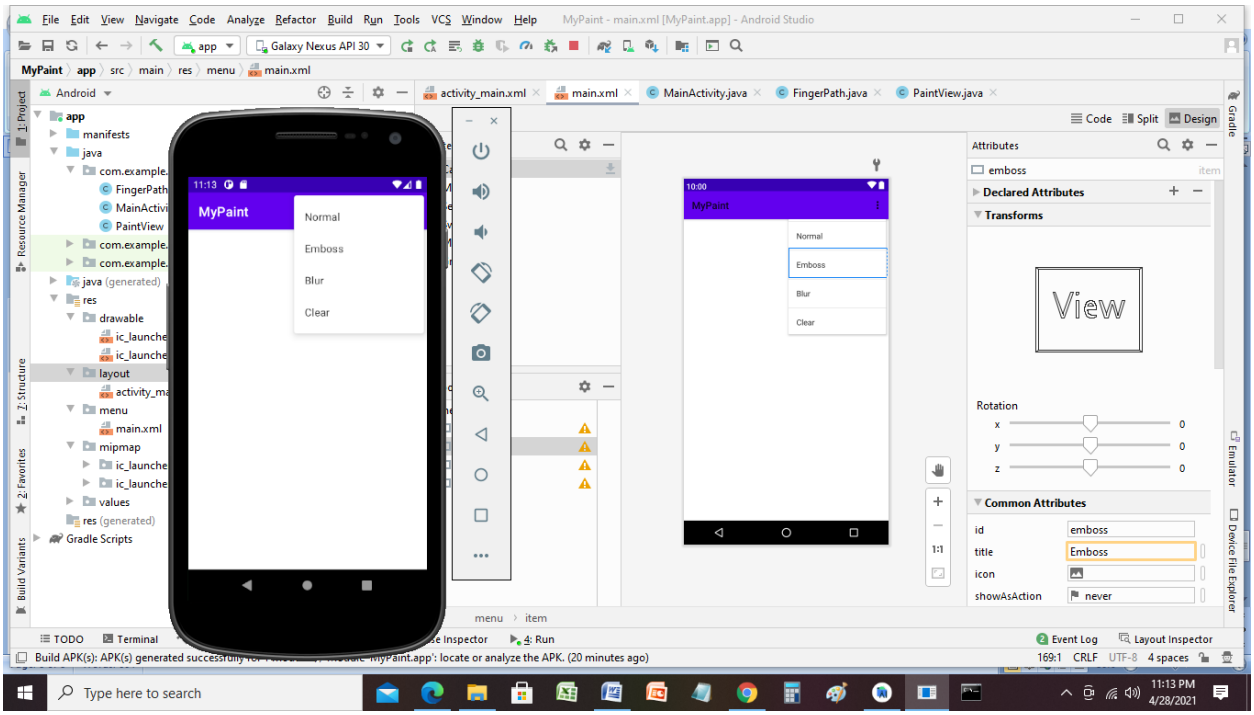
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.example.mypaint.PaintView
        android:id="@+id/paintView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</RelativeLayout>
```

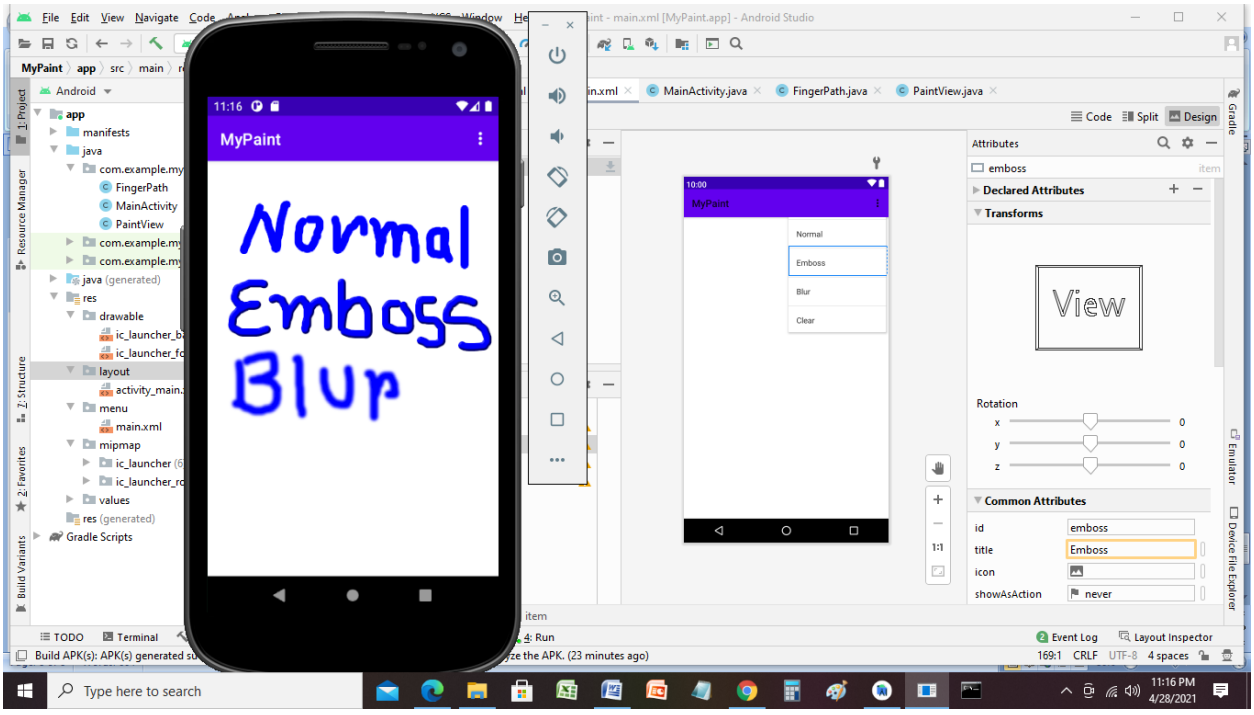


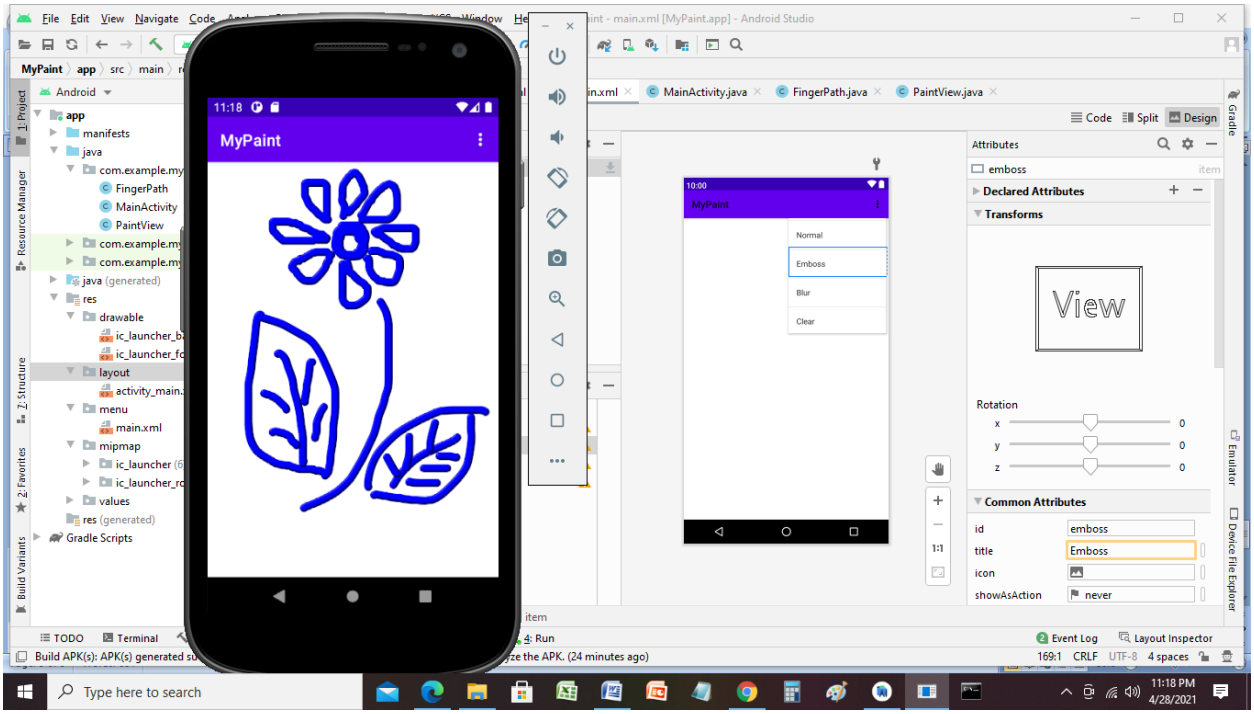
## MainActivity.java

```
package com.example.mypaint;

import android.annotation.SuppressLint;
import android.os.Bundle;

import android.util.DisplayMetrics;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private PaintView paintView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        paintView = (PaintView) findViewById(R.id.paintView);
        DisplayMetrics metrics = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(metrics);
        paintView.init(metrics);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.main, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @SuppressLint("NonConstantResourceId")
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch(item.getItemId()) {
            case R.id.normal:
                paintView.normal();

                return true;
            case R.id.emboss:
                paintView.emboss();
                return true;
            case R.id.blur:
                paintView.blur();
                return true;
            case R.id.clear:
                paintView.clear();
                return true;
        }
```

```
        return super.onOptionsItemSelected(item);
    }
}
```




MyPaint
Normal
Emboss
Blur
Clear
View
Declared Attributes
Transforms
Common Attributes
Rotation
id
title
icon
showAsAction
emboss
Emboss
never