**Національний університет “Львівська політехніка”**

**Інститут комп'ютерних наук та інформаційних технологій**



**Звіт до лабораторної роботи №5:**

**«Створення графічного мобільного додатку»**

з курсу “Програмування мобільних пристроїв”

Виконав:

студент групи КН-314

Юнусов Ю. М.

Прийняв:

Григорович В.Г.

#### ЛЬВІВ – 2019

**Мета роботи:** створити власний графічний мобільний додаток.

**Лабораторне завдання:**

1. Розібратися з проектом у прикладі
2. Розібратися з класами для відображення графіки (myGlSurfaceView, Renderer)
3. Знайти у проекті:
4. вершини фігури, яка відображена;
5. шейдери, які використовуються;
6. матриця виду;
7. позиція глядача;
8. код, який виводить фігуру на екран.
9. Розібратися, яким чином формується фігура з постідовності вершин (https://en.wikipedia.org/wiki/Triangle\_strip)
10. Адаптувати приклад згідно варіанту завдання.
11. Написати звіт.

**Файли програми:**

**Клас, який рендерить об’єкт**

public class MainRenderer implements GLSurfaceView.Renderer {

Triangle triangle;

@Override

public void onSurfaceCreated(GL10 gl, EGLConfig config) {

triangle = new Triangle();

}

@Override

public void onSurfaceChanged(GL10 gl, int width, int height) {

}

@Override

public void onDrawFrame(GL10 gl) {

GLES20.glClearColor(0.8f, 0.5f, 0.5f, 1.0f);

GLES20.glClear(GLES20.GL\_COLOR\_BUFFER\_BIT | GLES20.GL\_DEPTH\_BUFFER\_BIT);

triangle.draw();

}

}

**Клас, який перевіряє підтримку технологій GLES2.0 на мобільному присторої**

public class MainSurfaceView extends GLSurfaceView {

public MainSurfaceView(Context context) {

super(context);

}

}

**Клас, в якому формується об’єкт для рендерингу. В ньому прописуються:**

1. **вершини фігури, яка відображаються;**
2. **шейдери, які використовуються;**
3. **матриця виду;**

public class Triangle {

private FloatBuffer vertexBuffer;

private float vertices[]= {

-0.3f, 0.3f, 0.0f,

-0.2f, 0.3f, 0.0f,

-0.2f, 0.2f, 0.0f,

-0.2f, 0.3f, 0.0f,

-0.2f, 0.2f, 0.0f,

-0.1f, 0.2f, 0.0f,

-0.2f, 0.2f, 0.0f,

-0.1f, 0.2f, 0.0f,

-0.1f, 0.1f, 0.0f,

-0.1f, 0.2f, 0.0f,

0.0f, 0.1f, 0.0f,

-0.1f, 0.1f, 0.0f,

0.0f, 0.1f, 0.0f,

0.1f, 0.1f, 0.0f,

0.1f, 0.2f, 0.0f,

0.1f, 0.1f, 0.0f,

0.1f, 0.2f, 0.0f,

0.2f, 0.2f, 0.0f,

0.1f, 0.2f, 0.0f,

0.2f, 0.2f, 0.0f,

0.2f, 0.3f, 0.0f,

0.2f, 0.2f, 0.0f,

0.2f, 0.3f, 0.0f,

0.3f, 0.3f, 0.0f,

//----------------

-0.1f, 0.1f, 0.0f,

0.0f, 0.1f, 0.0f,

0.0f, 0.0f, 0.0f,

0.0f, 0.0f, 0.0f,

0.0f, 0.1f, 0.0f,

0.1f, 0.1f, 0.0f,

-0.1f, 0.1f, 0.0f,

-0.1f, 0.0f, 0.0f,

0.0f, 0.0f, 0.0f,

0.0f, 0.0f, 0.0f,

0.1f, 0.1f, 0.0f,

0.1f, 0.0f, 0.0f,

-0.1f, 0.0f, 0.0f,

0.0f, 0.0f, 0.0f,

-0.1f, -0.1f, 0.0f,

0.0f, 0.0f, 0.0f,

-0.1f, -0.1f, 0.0f,

0.0f, -0.1f, 0.0f,

0.0f, -0.1f, 0.0f,

0.0f, 0.0f, 0.0f,

0.1f, 0.0f, 0.0f,

0.0f, -0.1f, 0.0f,

0.1f, -0.1f, 0.0f,

0.1f, 0.0f, 0.0f,

};

private float color[] = new float[] {0.0f, 0.8f, 0.0f, 1.0f};

private final String vertexShaderCode =

"attribute vec4 vPosition;" +

"void main() {" +

" gl\_Position = vPosition;" +

"}";

private final String fragmentShaderCode =

"precision mediump float;" +

"uniform vec4 vColor;" +

"void main() {" +

" gl\_FragColor = vColor;" +

"}";

private int shaderProgram;

public static int loadShader(int type, String shaderCode){

int shader = GLES20.glCreateShader(type);

GLES20.glShaderSource(shader, shaderCode);

GLES20.glCompileShader(shader);

return shader;

}

public Triangle(){

for(int i = 0 ; i < vertices.length; i++){

vertices[i] \*= 2;

}

ByteBuffer bb = ByteBuffer.allocateDirect(vertices.length \* 4);

bb.order(ByteOrder.nativeOrder());

vertexBuffer = bb.asFloatBuffer();

vertexBuffer.put(vertices);

vertexBuffer.position(0);

int vertexShader = loadShader(GLES20.GL\_VERTEX\_SHADER, vertexShaderCode);

int fragmentShader = loadShader(GLES20.GL\_FRAGMENT\_SHADER, fragmentShaderCode);

shaderProgram = GLES20.glCreateProgram();

GLES20.glAttachShader(shaderProgram, vertexShader);

GLES20.glAttachShader(shaderProgram, fragmentShader);

GLES20.glLinkProgram(shaderProgram );

}

public void draw(){

GLES20.glUseProgram(shaderProgram);

int positionAttrib = GLES20.glGetAttribLocation(shaderProgram, "vPosition");

GLES20.glEnableVertexAttribArray(positionAttrib);

GLES20.glVertexAttribPointer(positionAttrib, 3, GLES20.GL\_FLOAT, false, 0, vertexBuffer);

int colorUniform = GLES20.glGetUniformLocation(shaderProgram, "vColor");

GLES20.glUniform4fv(colorUniform, 1, color, 0);

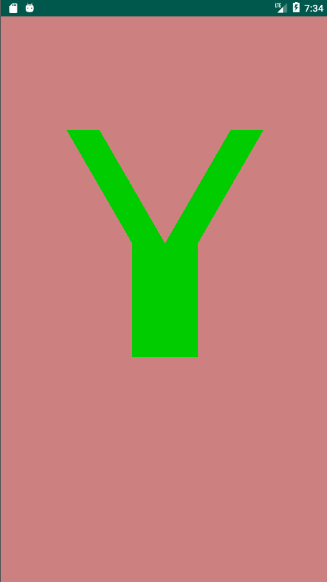
GLES20.glDrawArrays(GLES20.GL\_TRIANGLES, 0, vertices.length/3);

GLES20.glDisableVertexAttribArray(positionAttrib);

}

}

**Результат виконання роботи:**



**Висновок:**

На лабораторній роботі, я навчився розробляти простий інтерфейс програми, вивчив засоби розмітки FrameLayout, LinearLayout, вивчити елементи інтерфейсу EditText, ImageView, ScrollView, Button, навчився динамічно створювати елементи інтерфейсу з файлів розмітки, обробляти події, виконувати переходи між елементами програми.