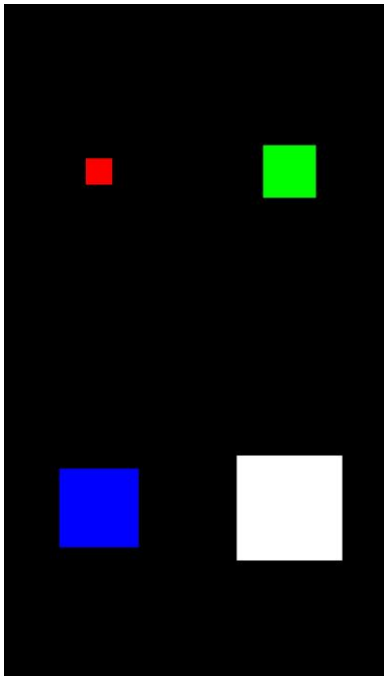
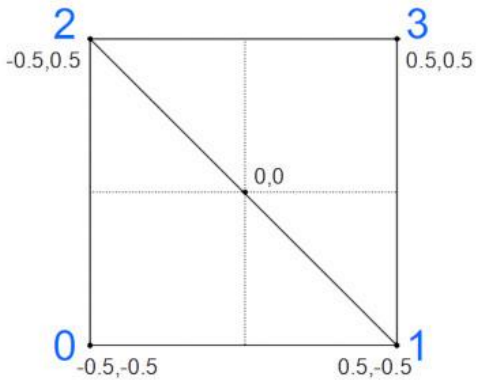
Android - OpenGL ES 1 - Tutorial 5

Draw Squares

In the same way as triangles, we draw squares as in the following screenshot:



Consider the square of the following image:



Create a class "ObjSquare" with the following code:

```
package opengles1.tutorial5;

import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.FloatBuffer;
import java.nio.ShortBuffer;

import javax.microedition.khronos.opengles.GL10;
```

```
public class ObjSquare {
    private FloatBuffer vertexBuffer;
    private ShortBuffer indexBuffer;

    private float[] vertices = {
            -0.5f, -0.5f,
            0.5f, -0.5f,
            -0.5f, 0.5f,
            0.5f, 0.5f
    };

    private short[] indices = {
            0, 1, 2,
            2, 1, 3
    };

    public ObjSquare() {
        ByteBuffer vbb = ByteBuffer.allocateDirect(this.vertices.length * 4);
        vbb.order(ByteOrder.nativeOrder());
        this.vertexBuffer = vbb.asFloatBuffer();
        this.vertexBuffer.put(this.vertices);
        this.vertexBuffer.position(0);

        ByteBuffer ibb = ByteBuffer.allocateDirect(this.indices.length * 2);
        ibb.order(ByteOrder.nativeOrder());
        this.indexBuffer = ibb.asShortBuffer();
        this.indexBuffer.put(this.indices);
        this.indexBuffer.position(0);
    }

    public void DrawSquare(GL10 gl, int color) {
        gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
        gl.glVertexPointer(2, GL10.GL_FLOAT, 0, this.vertexBuffer);
        if (color == 1) // Red.
            gl.glColor4f(1.0f, 0.0f, 0.0f, 1.0f);
        else if (color == 2) // Green.
            gl.glColor4f(0.0f, 1.0f, 0.0f, 1.0f);
        else if (color == 3) // Blue.
            gl.glColor4f(0.0f, 0.0f, 1.0f, 1.0f);
        else // White.
            gl.glColor4f(1.0f, 1.0f, 1.0f, 1.0f);
        gl.glDrawElements(GL10.GL_TRIANGLES, this.indices.length, GL10.GL_UNSIGNED_SHORT,
this.indexBuffer);
        gl.glDisableClientState(GL10.GL_VERTEX_ARRAY);
    }

}
```

Every shape we want to draw must be brought back into triangles, therefore OpenGL will draw the square with 2 triangles.

The "indices" variable specifies the vertices to be taken from "vertices" for each triangle (counterclockwise). For the first triangle, pair of coordinates x,y: 0, 1 and 2, for the second triangle, pair of coordinates x,y: 2, 1 and 3.

In the "MainRenderer" class we declare:

```
ObjSquare objsquare;
```

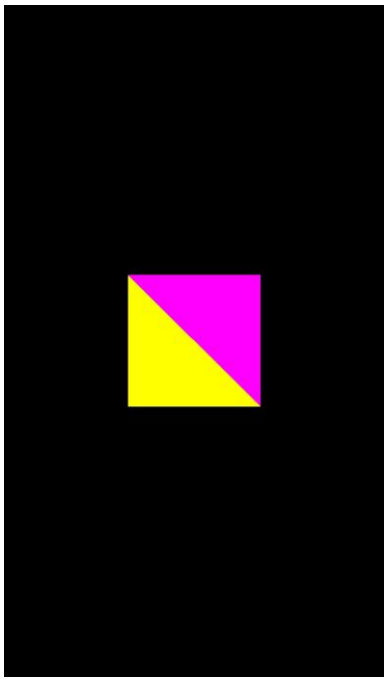
and initialize in the constructor:

```
public MainRenderer() {
    this.objsquare = new ObjSquare();
}
```

In the "onDrawFrame()" method we draw our squares:

```
@Override
public void onDrawFrame(GL10 gl) {
    gl.glClear(GL10.GL_COLOR_BUFFER_BIT);

    gl.glLoadIdentity();
    gl.glTranslatef(this.width / 4.0f, this.height / 4.0f * 3.0f, 0.0f);
    gl.glScalef(50.0f, 50.0f, 0.0f);
    this.objsquare.DrawSquare(gl, 1);

    gl.glLoadIdentity();
    gl.glTranslatef(this.width / 4.0f * 3.0f, this.height / 4.0f * 3.0f, 0.0f);
    gl.glScalef(100.0f, 100.0f, 0.0f);
    this.objsquare.DrawSquare(gl, 2);

    gl.glLoadIdentity();
    gl.glTranslatef(this.width / 4.0f, this.height / 4.0f, 0.0f);
    gl.glScalef(150.0f, 150.0f, 0.0f);
    this.objsquare.DrawSquare(gl, 3);

    gl.glLoadIdentity();
    gl.glTranslatef(this.width / 4.0f * 3.0f, this.height / 4.0f, 0.0f);
    gl.glScalef(200.0f, 200.0f, 0.0f);
    this.objsquare.DrawSquare(gl, 0);
}
```

What if we wanted to draw a square like this?



Obviously we can individually draw two triangles adjacent to each other, but we create a dedicated class "ObjSquare2" with the following code:

```
package opengles1.tutorial5b;

import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.FloatBuffer;
import java.nio.ShortBuffer;

import javax.microedition.khronos.opengles.GL10;

public class ObjSquare2 {
    private FloatBuffer vertexBuffer;
    private ShortBuffer indexBuffer;
    private FloatBuffer colorBuffer;

    private float[] vertices = {
            -0.5f, -0.5f,
            0.5f, -0.5f,
            -0.5f, 0.5f,
            0.5f, -0.5f,
            -0.5f, 0.5f,
            0.5f, 0.5f
    };

    private short[] indices = {
            0, 1, 2,
            3, 4, 5
    };

    private float[] colors = {
            1.0f, 1.0f, 0.0f, 1.0f, // Vertex 0 triangle 1 (Red+Green).
            1.0f, 1.0f, 0.0f, 1.0f, // Vertex 1 triangle 1 (Red+Green).
            1.0f, 1.0f, 0.0f, 1.0f, // Vertex 2 triangle 1 (Red+Green).
            1.0f, 0.0f, 1.0f, 1.0f, // Vertex 2 triangle 2 (Red+Blue).
            1.0f, 0.0f, 1.0f, 1.0f, // Vertex 1 triangle 2 (Red+Blue).
            1.0f, 0.0f, 1.0f, 1.0f // Vertex 3 triangle 2 (Red+Blue).
    };

    public ObjSquare2() {
        ByteBuffer vbb = ByteBuffer.allocateDirect(this.vertices.length * 4);
        vbb.order(ByteOrder.nativeOrder());
        this.vertexBuffer = vbb.asFloatBuffer();
        this.vertexBuffer.put(this.vertices);
        this.vertexBuffer.position(0);

        ByteBuffer ibb = ByteBuffer.allocateDirect(this.indices.length * 2);
        ibb.order(ByteOrder.nativeOrder());
        this.indexBuffer = ibb.asShortBuffer();
        this.indexBuffer.put(this.indices);
        this.indexBuffer.position(0);

        ByteBuffer vcc = ByteBuffer.allocateDirect(this.colors.length * 4);
        vcc.order(ByteOrder.nativeOrder());
        this.colorBuffer = vcc.asFloatBuffer();
        this.colorBuffer.put(this.colors);
        this.colorBuffer.position(0);
    }
```

```
    public void DrawSquare(GL10 gl) {
        gl.glEnableClientState(GL10.GL_VERTEX_ARRAY);
        gl.glVertexPointer(2, GL10.GL_FLOAT, 0, this.vertexBuffer);
        gl.glEnableClientState(GL10.GL_COLOR_ARRAY);
        gl.glColorPointer(4, GL10.GL_FLOAT, 0, this.colorBuffer);
        gl.glDrawElements(GL10.GL_TRIANGLES, this.indices.length, GL10.GL_UNSIGNED_SHORT,
this.indexBuffer);
        gl.glDisableClientState(GL10.GL_VERTEX_ARRAY);
        gl.glDisableClientState(GL10.GL_COLOR_ARRAY);
    }

}
```

In this case, the "indices" variable is used to indicate, in addition to the vertices of the triangles, also the respective colors to be taken from "colors". The colors of the two triangles are not in common, so we are forced to specify them separately and consequently to repeat in "vertices" the coordinates.

For the first triangle, pair of coordinates x,y: 0, 1 and 2 with colors in position 0, 1 and 2, for the second triangle, pair of coordinates x,y: 3, 4 and 5 with colors in position 3, 4 and 5.

In the "MainRenderer" class we declare:

```
ObjSquare2 objsquare2;
```

and initialize in the constructor:

```
public MainRenderer() {
    this.objsquare2 = new ObjSquare2();
}
```

Than, in the "onDrawFrame()" method:

```
@Override
public void onDrawFrame(GL10 gl) {
    gl.glClear(GL10.GL_COLOR_BUFFER_BIT);

    gl.glLoadIdentity();
    gl.glTranslatef(this.width / 2.0f, this.height / 2.0f, 0.0f);
    gl.glScalef(250.0f, 250.0f, 0.0f);
    this.objsquare2.DrawSquare(gl);
}
```