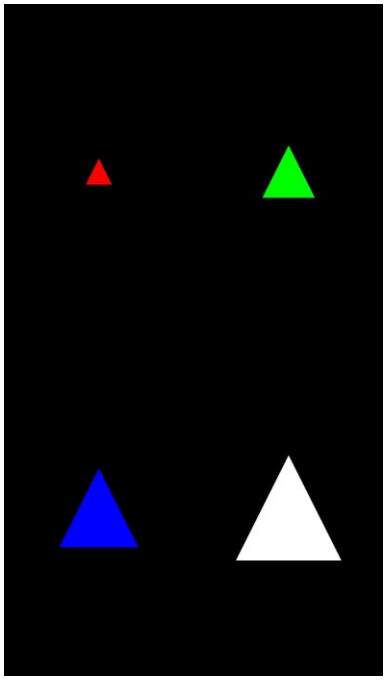
Android - OpenGL ES 2 - Tutorial 3

Draw Triangles

The purpose of this tutorial is to draw 4 triangles as in the following screenshot:



Modify the previous project in the following way:

- in the "ObjTriangle" class, the "DrawTriangle()" method becomes:

```
public void DrawTriangle(float[] modelMatrix, int color) {
    GLES20.glUseProgram(this.mProgram);

    this.positionHandle = GLES20.glGetAttribLocation(this.mProgram, "vPosition");
    GLES20.glEnableVertexAttribArray(this.positionHandle);
    GLES20.glVertexAttribPointer(this.positionHandle, COORDS_PER_VERTEX, GLES20.GL_FLOAT,
false, this.vertexStride, this.vertexBuffer);

    this.colorHandle = GLES20.glGetUniformLocation(this.mProgram, "vColor");
    if (color == 1) // Red.
        GLES20.glUniform4fv(this.colorHandle, 1, this.color_red, 0);
    else if (color == 2) // Green.
        GLES20.glUniform4fv(this.colorHandle, 1, this.color_green, 0);
    else if (color == 3) // Blue.
        GLES20.glUniform4fv(this.colorHandle, 1, this.color_blue, 0);
    else // White.
        GLES20.glUniform4fv(this.colorHandle, 1, this.color_white, 0);

    this.modelHandle = GLES20.glGetUniformLocation(this.mProgram, "uMVPMatrix");
    GLES20.glUniformMatrix4fv(this.modelHandle, 1, false, modelMatrix, 0);

    GLES20.glDrawElements(GL10.GL_TRIANGLES, this.indices.length, GL10.GL_UNSIGNED_SHORT,
this.indexBuffer);

    GLES20.glDisableVertexAttribArray(this.positionHandle);
}
```

While the color variable:

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

is replaced with:

```
private float[] color_red = { 1.0f, 0.0f, 0.0f, 1.0f };
private float[] color_green = { 0.0f, 1.0f, 0.0f, 1.0f };
private float[] color_blue = { 0.0f, 0.0f, 1.0f, 1.0f };
private float[] color_white = { 1.0f, 1.0f, 1.0f, 1.0f };
```

Here we have added the "color" variable that we pass during the call to specify the color to be used.

- the "onDrawFrame()" method becomes:

```
@Override
public void onDrawFrame(GL10 unused) {
    GLES20.glClear(GLES20.GL_COLOR_BUFFER_BIT);

    this.ResetModelMatrix();
    Matrix.translateM(this.modelMatrix, 0, this.width / 4.0f, this.height / 4.0f * 3.0f,
0.0f);
    Matrix.scaleM(this.modelMatrix, 0, 50.0f, 50.0f, 0.0f);
    this.objtriangle.DrawTriangle(this.modelMatrix, 1);

    this.ResetModelMatrix();
    Matrix.translateM(this.modelMatrix, 0, this.width / 4.0f * 3.0f, this.height / 4.0f *
3.0f, 0.0f);
    Matrix.scaleM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
    this.objtriangle.DrawTriangle(this.modelMatrix, 2);

    this.ResetModelMatrix();
    Matrix.translateM(this.modelMatrix, 0, this.width / 4.0f, this.height / 4.0f, 0.0f);
    Matrix.scaleM(this.modelMatrix, 0, 150.0f, 150.0f, 0.0f);
    this.objtriangle.DrawTriangle(this.modelMatrix, 3);

    this.ResetModelMatrix();
    Matrix.translateM(this.modelMatrix, 0, this.width / 4.0f * 3.0f, this.height / 4.0f,
0.0f);
    Matrix.scaleM(this.modelMatrix, 0, 200.0f, 200.0f, 0.0f);
    this.objtriangle.DrawTriangle(this.modelMatrix, 0);
}
```

Note the reset of the model matrix before each drawing. If we do not reset the matrix, translate or scale will be performed on the basis of the previous operation.

The code:

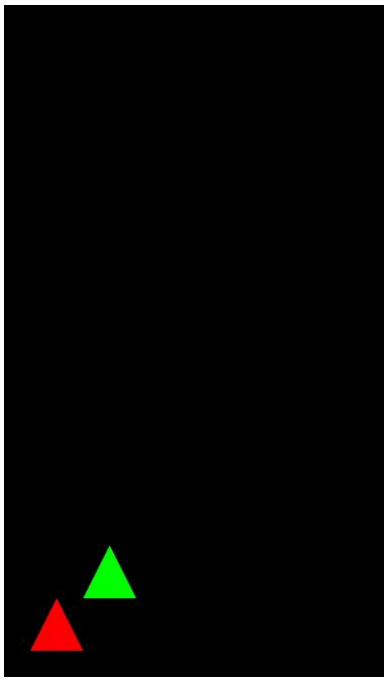
```
this.ResetModelMatrix();
Matrix.translateM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
Matrix.scaleM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
this.objtriangle.DrawTriangle(this.modelMatrix, 1);

this.ResetModelMatrix();
Matrix.translateM(this.modelMatrix, 0, 200.0f, 200.0f, 0.0f);
Matrix.scaleM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
this.objtriangle.DrawTriangle(this.modelMatrix, 2);
```

and the code:

```
this.ResetModelMatrix();
Matrix.scaleM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
Matrix.translateM(this.modelMatrix, 0, 1.0f, 1.0f, 0.0f);
this.objtriangle.DrawTriangle(this.modelMatrix, 1);
Matrix.translateM(this.modelMatrix, 0, 1.0f, 1.0f, 0.0f);
this.objtriangle.DrawTriangle(this.modelMatrix, 2);
```

produce the same result:



In the first code, for each triangle, we reset, translate, scale and draw.

In the second code, we reset, scale (obtaining a basic unit of "100"), translate by 1 (that is 100), draw, translate again by 1 (that is an other 100) on the previous translate, draw.