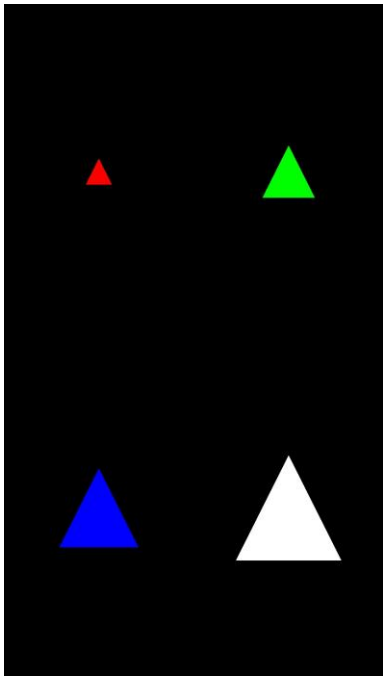
Android - OpenGL ES 2 - Tutorial 3

Disegnare Triangoli

Lo scopo di questo tutorial è di disegnare 4 triangoli come nel seguente screenshot:



Modifichiamo il progetto precedente nel seguente modo:

- nella classe "ObjTriangle", il metodo "DrawTriangle()" diventa:

```
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);
}
```

Mentre la variabile del colore:

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

viene sostituita con:

```
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 };
```

Qui abbiamo aggiunto la variabile "color" che passiamo in fase di chiamata per specificare il colore da utilizzare.

- il metodo "onDrawFrame()" diventa:

```
@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);
}
```

Notiamo il reset della matrice del modello prima di ogni disegno. Se non resettiamo la matrice, la traslazione o lo scalo verrà eseguito sulla base della precedente operazione.

Il codice:

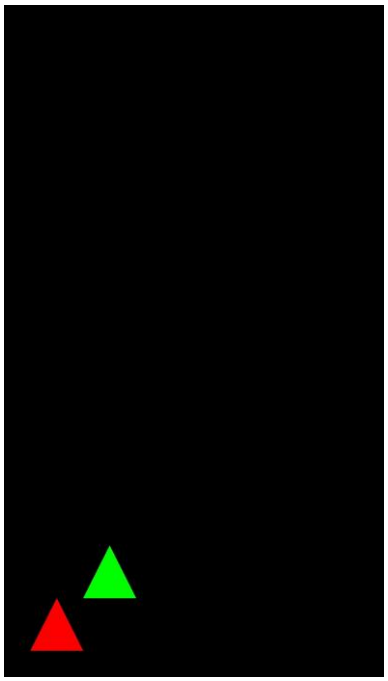
```
this.ResetModelMatrix();
Matrix.translateM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
Matrix.scaleM(this.modelMatrix, 0, 100.0f, 100.0f, 0.0f);
this.objtriangle.DrawTriangle(this.modelMatrix, 1);

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);
```

ed il codice:

```
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);
```

producono lo stesso risultato:



Nel primo codice, per ogni triangolo, resettiamo, trasliamo, scaliamo e disegniamo.

Nel secondo codice, resettiamo, scaliamo (ottenendo come unità di base "100"), trasliamo di 1 (ovvero 100), disegniamo, trasliamo nuovamente di 1 (ovvero altri 100) sulla base della precedente traslazione, disegniamo.