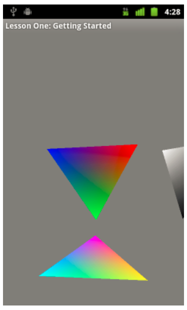
Learn OpenGL ES

Learn how to develop mobile graphics using OpenGL ES 2

Android Lesson One: Getting Started



This is the first tutorial on using OpenGL ES 2 on Android. In this lesson, we're going to go over the code step-by-step, and look at how to create an OpenGL ES 2 context and draw to the screen. We'll also take a look at what shaders are and how they work, as well as how matrices are used to transform the scene into the image you see on the screen. Finally, we'll look at what you need to add to the manifest to let the Android market know that you're using OpenGL ES 2.

PREREQUISITES

Before we begin, you'll want to make sure you have the following tools installed on your machine:

- Android SDK
- Eclipse
- ADT Plugin for Eclipse

~~Unfortunately, the Android emulator does not support OpenGL ES 2, so you'll need access to an actual Android device in order to run the tutorial.~~ The Android Emulator now supports OpenGL ES 2 in recent versions of the Android SDK.

Most recent devices should also support OpenGL ES 2 these days, so all you need to do is enable developer mode and hook the phone up to your machine.

ASSUMPTIONS

The reader should be familiar with Android and Java on a basic level. The Android Tutorials are a good place to start.

GETTING STARTED

We'll go over all of the code below and explain what each part does. You can copy the code on a segment by segment basis by creating your own project, or you can download the completed project at the end of the lesson. Once you have the tools installed, go ahead and create a new Android project in Eclipse. The names don't matter, but for the lesson I will be referring to the main activity as *LessonOneActivity*.

Let's take a look at the code:

```
/** Hold a reference to our GLSurfaceView */
private GLSurfaceView mGLSurfaceView;
```

The GLSurfaceView is a special view which manages OpenGL surfaces for us and draws it into the Android view system. It also adds a lot of features which make it easier to use OpenGL, including but not limited to:

- It provides a dedicated render thread for OpenGL so that the main thread is not stalled.
- It supports continuous or on-demand rendering.
- It takes care of the screen setup for you using EGL, the interface between OpenGL and the underlying window system.

The GLSurfaceView makes setting up and using OpenGL from Android relatively painless.

```
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);

    mGLSurfaceView = new GLSurfaceView(this);

    // Check if the system supports OpenGL ES 2.0.
    final ActivityManager activityManager = (ActivityManager) getSystemSer
    final ConfigurationInfo configurationInfo = activityManager.getDevice(
    final boolean supportsEs2 = configurationInfo.reqGlEsVersion >= 0x200

    if (supportsEs2)
    {
        // Request an OpenGL ES 2.0 compatible context.
        mGLSurfaceView.setEGLContextClientVersion(2);

        // Set the renderer to our demo renderer, defined below.
        mGLSurfaceView.setRenderer(new LessonOneRenderer());
    }
    else
    {
        // This is where you could create an OpenGL ES 1.x compatible
        // renderer if you wanted to support both ES 1 and ES 2.
        return;
    }

    setContentView(mGLSurfaceView);
}
```

The onCreate() of our activity is the important part where our OpenGL context gets created and where everything starts happening. In our onCreate(), the first thing we do after calling the superclass is creating our GLSurfaceView. We then need to figure out if the system supports OpenGL ES 2. To do this, we get an ActivityManager instance which lets us interact with the global system state. We can then use this to get the device configuration info, which will tell us if the device supports OpenGL ES 2.

Once we know if the device supports OpenGL ES 2 or not, we tell the surface view we want an OpenGL ES 2 compatible surface and then we pass in a custom renderer. This renderer will be called by the system whenever it's time to adjust the surface or draw a new frame. We can also support OpenGL Es 1.x by passing in a different renderer, though we would need to write different code as the APIs are different. For this lesson we'll only look at supporting OpenGL ES 2.

Finally, we set the content view to our GLSurfaceView, which tells Android that the activity's contents should be filled by our OpenGL surface. To get into OpenGL, it's as easy as that!

```
@Override
protected void onResume()
```

```
{
    // The activity must call the GL surface view's onResume() on activity
    super.onResume();
    mGLSurfaceView.onResume();
}

@Override
protected void onPause()
{
    // The activity must call the GL surface view's onPause() on activity
    super.onPause();
    mGLSurfaceView.onPause();
}
```

GLSurfaceView requires that we call its onResume() and onPause() methods whenever the parent Activity has its own onResume() and onPaused() called. We add in the calls here to round out our activity.

VISUALIZING A 3D WORLD

In this section, we'll start looking at how OpenGL ES 2 works and how we can start drawing stuff onto the screen. In the activity we passed in a custom GLSurfaceView.Renderer to the GLSurfaceView, which will be defined here. The renderer has three important methods which will be automatically called by the system whenever events happen:

public void onSurfaceCreated(GL10 glUnused, EGLConfig config)
This method is called when the surface is first created. It will also be called if we lose our surface context and it is later recreated by the system.

public void onSurfaceChanged(GL10 glUnused, int width, int height)
This is called whenever the surface changes; for example, when switching from portrait to landscape. It is also called after the surface has been created.

public void onDrawFrame(GL10 glUnused)
This is called whenever it's time to draw a new frame.

You may have noticed that the GL10 instance passed in is referred to as glUnused. We don't use this when drawing using OpenGL ES 2; instead, we use the static methods of the class GLES20. The GL10 parameter is only there because the same interface is used for OpenGL ES 1.x.

Before our renderer can display anything, we’ll need to have something to display. In OpenGL ES 2, we pass in stuff to display by specifying arrays of numbers. These numbers can represent positions, colors, or anything else we need them to. In this demo, we’ll display three triangles.

```
// New class members
/** Store our model data in a float buffer. */
private final FloatBuffer mTriangle1Vertices;
private final FloatBuffer mTriangle2Vertices;
private final FloatBuffer mTriangle3Vertices;

/** How many bytes per float. */
private final int mBytesPerFloat = 4;

/**
 * Initialize the model data.
 */
public LessonOneRenderer()
{
    // This triangle is red, green, and blue.
    final float[] triangle1VerticesData = {
            // X, Y, Z,
            // R, G, B, A
            -0.5f, -0.25f, 0.0f,
            1.0f, 0.0f, 0.0f, 1.0f,

            0.5f, -0.25f, 0.0f,
            0.0f, 0.0f, 1.0f, 1.0f,

            0.0f, 0.559016994f, 0.0f,
            0.0f, 1.0f, 0.0f, 1.0f};

    ...

    // Initialize the buffers.
    mTriangle1Vertices = ByteBuffer.allocateDirect(triangle1VerticesData.
    .order(ByteOrder.nativeOrder()).asFloatBuffer();

    ...

    mTriangle1Vertices.put(triangle1VerticesData).position(0);

    ...
}
```

So, what’s all this about? If you’ve ever used OpenGL 1, you might be used to doing things this way:

```
glBegin(GL_TRIANGLES);
glVertex3f(-0.5f, -0.25f, 0.0f);
glColor3f(1.0f, 0.0f, 0.0f);
...
glEnd();
```

Things don’t work that way in OpenGL ES 2. Instead of defining points via a bunch of method calls, we define an array instead. Let’s take a look at our array again:

```
final float[] triangle1VerticesData = {
                // X, Y, Z,
                // R, G, B, A
                -0.5f, -0.25f, 0.0f,
                1.0f, 0.0f, 0.0f, 1.0f,
                ...
```

This represents one point of the triangle. We’ve set things up so that the first three numbers represent the position (X, Y, and Z), and the last four numbers represent the color (red, green, blue, and alpha (transparency)). You don’t need to worry too much about how this array is defined; just remember that when we want to draw stuff in OpenGL ES 2, we need to pass it data in chunks instead of passing it in one at a time.

Understanding buffers

```
// Initialize the buffers.
        mTriangle1Vertices = ByteBuffer.allocateDirect(triangle1VerticesDa
        .order(ByteOrder.nativeOrder()).asFloatBuffer();
        ...
        mTriangle1Vertices.put(triangle1VerticesData).position(0);
```

We do our coding in Java on Android, but the underlying implementation of OpenGL ES 2 is actually written in C. Before we pass our data to OpenGL, we need to convert it into a form that it’s going to understand. Java and the native system might not store their bytes in the same order, so we use a special set of buffer classes and create a ByteBuffer large enough to hold our data, and tell it to store its data using the native byte order. We then convert it into a FloatBuffer so that we can use it to hold floating-point data. Finally, we copy our array into the buffer.

This buffer stuff might seem confusing (it was to me when I first came across it!), but just remember that it’s an extra step we need to do before passing our data to OpenGL. Our buffers are now ready to be used to pass data into OpenGL.

As a side note, float buffers are slow on Froyo and moderately faster on Gingerbread, so you probably don’t want to be changing them too often.

Understanding matrices

```
// New class definitions
 /**
 * Store the view matrix. This can be thought of as our camera. This matri
 * it positions things relative to our eye.
 */
private float[] mViewMatrix = new float[16];

@Override
public void onSurfaceCreated(GL10 glUnused, EGLConfig config)
{
    // Set the background clear color to gray.
    GLES20.glClearColor(0.5f, 0.5f, 0.5f, 0.5f);

    // Position the eye behind the origin.
    final float eyeX = 0.0f;
    final float eyeY = 0.0f;
    final float eyeZ = 1.5f;

    // We are looking toward the distance
    final float lookX = 0.0f;
    final float lookY = 0.0f;
    final float lookZ = -5.0f;

    // Set our up vector. This is where our head would be pointing were we
    final float upX = 0.0f;
    final float upY = 1.0f;
    final float upZ = 0.0f;

    // Set the view matrix. This matrix can be said to represent the camer
    // NOTE: In OpenGL 1, a ModelView matrix is used, which is a combinati
    // view matrix. In OpenGL 2, we can keep track of these matrices separ
    Matrix.setLookAtM(mViewMatrix, 0, eyeX, eyeY, eyeZ, lookX, lookY, look

    ...
```

Another ‘fun’ topic is matrices! These will become your best friends whenever you do 3D programming, so you’ll want to get to know them well.

When our surface is created, the first thing we do is set our clear color to gray. The alpha part has also been set to gray, but we’re not doing alpha blending in this lesson so this value is unused. We only need to set the clear color once since we will not be changing it later.

The second thing we do is setup our view matrix. There are several different kinds of matrices we use and they all do something important:

1. The model matrix. This matrix is used to place a model somewhere in the “world”. For example, if you have a model of a car and you want it located

1000 meters to the east, you will use the model matrix to do this.
2. The view matrix. This matrix represents the camera. If we want to view our car which is 1000 meters to the east, we'll have to move ourselves 1000 meters to the east as well (another way of thinking about it is that we remain stationary, and the rest of the world moves 1000 meters to the west). We use the view matrix to do this.
3. The projection matrix. Since our screens are flat, we need to do a final transformation to "project" our view onto our screen and get that nice 3D perspective. This is what the projection matrix is used for.

A good explanation of this can be found over at SongHo's OpenGL Tutorials. I recommend reading it a few times until you grasp the idea well; don't worry, it took me a few reads as well!

In OpenGL 1, the model and view matrices are combined and the camera is assumed to be at (0, 0, 0) and facing the -Z direction.

We don't need to construct these matrices by hand. Android has a Matrix helper class which can do the heavy lifting for us. Here, I create a view matrix for a camera which is positioned behind the origin and looking toward the distance.

Defining vertex and fragment shaders

```
final String vertexShader =
    "uniform mat4 u_MVPMatrix;      \n"     // A constant representing the

  + "attribute vec4 a_Position;     \n"     // Per-vertex position informa
  + "attribute vec4 a_Color;        \n"     // Per-vertex color informatio

  + "varying vec4 v_Color;          \n"     // This will be passed into th

  + "void main()                    \n"     // The entry point for our ver
  + "{                              \n"
  + "   v_Color = a_Color;          \n"     // Pass the color through to t
                                            // It will be interpolated acr
  + "   gl_Position = u_MVPMatrix   \n"     // gl_Position is a special va
  + "               * a_Position;   \n"     // Multiply the vertex by the
  + "}                              \n";    // normalized screen coordinat
```

In OpenGL ES 2, anything we want to display on the screen is first going to have to go through a vertex and fragment shader. The good thing is that these shaders are really not as complicated as they appear. Vertex shaders perform operations on each vertex, and the results of these operations are used in the fragment shaders which do additional calculations per pixel.

Each shader basically consists of input, output, and a program. First we define a uniform, which is a combined matrix containing all of our transformations. This is a constant across all vertices and is used to project them onto the screen. Then we define two attributes for position and color. These attributes will be read in from the buffer we defined earlier on, and specify the position and color of each vertex. We then define a varying, which interpolates values across the triangle and passes it on to the fragment shader. When it gets to the fragment shader, it will hold an interpolated value for each pixel.

Let’s say we defined a triangle with each point being red, green, and blue, and we sized it so that it will take up 10 pixels on the screen. When the fragment shader runs, it will contain a different varying color for each pixel. At one point, that varying will be red, but halfway between red and blue it may be a more purplish color.

Aside from setting the color, we also tell OpenGL what the final position of the vertex should be on the screen. Then we define the fragment shader:

```
final String fragmentShader =
    "precision mediump float;       \n"     // Set the default precision
                                            // precision in the fragment
  + "varying vec4 v_Color;          \n"     // This is the color from the
                                            // triangle per fragment.
  + "void main()                    \n"     // The entry point for our fr
  + "{                              \n"
  + "   gl_FragColor = v_Color;     \n"     // Pass the color directly th
  + "}                              \n";
```

This is the fragment shader which will actually put stuff on the screen. In this shader, we grab the varying color from the vertex shader, and just pass it straight through to OpenGL. The point is already interpolated per pixel since the fragment shader runs for each pixel that will be drawn.

More information can be found on the OpenGL ES 2 quick reference card.

Loading shaders into OpenGL

```
// Load in the vertex shader.
int vertexShaderHandle = GLES20.glCreateShader(GLES20.GL_VERTEX_SHADER);

if (vertexShaderHandle != 0)
{
    // Pass in the shader source.
    GLES20.glShaderSource(vertexShaderHandle, vertexShader);

    // Compile the shader.
    GLES20.glCompileShader(vertexShaderHandle);

    // Get the compilation status.
    final int[] compileStatus = new int[1];
    GLES20.glGetShaderiv(vertexShaderHandle, GLES20.GL_COMPILE_STATUS, con

    // If the compilation failed, delete the shader.
    if (compileStatus[0] == 0)
    {
        GLES20.glDeleteShader(vertexShaderHandle);
        vertexShaderHandle = 0;
    }
}

if (vertexShaderHandle == 0)
{
    throw new RuntimeException("Error creating vertex shader.");
}
```

First, we create the shader object. If this succeeded, we'll get a reference to the object. Then we use this reference to pass in the shader source code, and then we compile it. We can obtain the status from OpenGL and see if it compiled successfully. If there were errors, we can use *GLES20.glGetShaderInfoLog(shader)* to find out why. We follow the same steps to load the fragment shader.

Linking a vertex and fragment shader together into a program

```
// Create a program object and store the handle to it.
int programHandle = GLES20.glCreateProgram();

if (programHandle != 0)
{
    // Bind the vertex shader to the program.
    GLES20.glAttachShader(programHandle, vertexShaderHandle);

    // Bind the fragment shader to the program.
    GLES20.glAttachShader(programHandle, fragmentShaderHandle);

    // Bind attributes
    GLES20.glBindAttribLocation(programHandle, 0, "a_Position");
    GLES20.glBindAttribLocation(programHandle, 1, "a_Color");

    // Link the two shaders together into a program.
    GLES20.glLinkProgram(programHandle);
    // Get the link status.
    final int[] linkStatus = new int[1];
    GLES20.glGetProgramiv(programHandle, GLES20.GL_LINK_STATUS, linkStatus

    // If the link failed, delete the program.
    if (linkStatus[0] == 0)
    {
        GLES20.glDeleteProgram(programHandle);
        programHandle = 0;
    }
}

if (programHandle == 0)
{
    throw new RuntimeException("Error creating program.");
}
```

Before we can use our vertex and fragment shader, we need to bind them together into a program. This is what connects the output of the vertex shader with the input of the fragment shader. It’s also what lets us pass in input from our program and use the shader to draw our shapes.

We create a new program object, and if that succeeded, we then attach our shaders. We want to pass in the position and color as attributes, so we need to bind these attributes. We then link the shaders together.

```
//New class members
/** This will be used to pass in the transformation matrix. */
private int mMVPMatrixHandle;

/** This will be used to pass in model position information. */
private int mPositionHandle;

/** This will be used to pass in model color information. */
private int mColorHandle;

@Override
public void onSurfaceCreated(GL10 glUnused, EGLConfig config)
{
    ...

    // Set program handles. These will later be used to pass in values to
    mMVPMatrixHandle = GLES20.glGetUniformLocation(programHandle, "u_MVPMa
    mPositionHandle = GLES20.glGetAttribLocation(programHandle, "a_Positio
    mColorHandle = GLES20.glGetAttribLocation(programHandle, "a_Color");

    // Tell OpenGL to use this program when rendering.
    GLES20.glUseProgram(programHandle);
}
```

After we successfully linked our program, we finish up with a couple more tasks so we can actually use it. The first task is obtaining references so we can pass data into the program. Then we tell OpenGL to use this program when drawing. Since we only use one program in this lesson, we can put this in the onSurfaceCreated() instead of the onDrawFrame().

Setting the perspective projection

```
// New class members
/** Store the projection matrix. This is used to project the scene onto a
private float[] mProjectionMatrix = new float[16];

@Override
public void onSurfaceChanged(GL10 glUnused, int width, int height)
{
    // Set the OpenGL viewport to the same size as the surface.
    GLES20.glViewport(0, 0, width, height);

    // Create a new perspective projection matrix. The height will stay th
    // while the width will vary as per aspect ratio.
    final float ratio = (float) width / height;
    final float left = -ratio;
    final float right = ratio;
    final float bottom = -1.0f;
    final float top = 1.0f;
    final float near = 1.0f;
    final float far = 10.0f;

    Matrix.frustumM(mProjectionMatrix, 0, left, right, bottom, top, near,
}
```

Our onSurfaceChanged() is called at least once and also whenever our surface is changed. Since we only need to reset our projection matrix whenever the screen we're projecting onto has changed, onSurfaceChanged() is an ideal place to do it.

Drawing stuff to the screen!

```
    // New class members
    /**
     * Store the model matrix. This matrix is used to move models from obj
     * of being located at the center of the universe) to world space.
     */
    private float[] mModelMatrix = new float[16];

    @Override
    public void onDrawFrame(GL10 glUnused)
    {
        GLES20.glClear(GLES20.GL_DEPTH_BUFFER_BIT | GLES20.GL_COLOR_BUFFE

        // Do a complete rotation every 10 seconds.
        long time = SystemClock.uptimeMillis() % 10000L;
        float angleInDegrees = (360.0f / 10000.0f) * ((int) time);

        // Draw the triangle facing straight on.
        Matrix.setIdentityM(mModelMatrix, 0);
        Matrix.rotateM(mModelMatrix, 0, angleInDegrees, 0.0f, 0.0f, 1.0f)
        drawTriangle(mTriangle1Vertices);

        ...
}
```

This is where stuff actually get displayed on the screen. We clear the screen so we don't get any weird hall of mirror effects, and we want our triangles to animate smoothly so we rotate them using time. Whenever you animate something on the screen, it's usually better to use time instead of framerate.

The actual drawing is done in drawTriangle:

```
// New class members
/** Allocate storage for the final combined matrix. This will be passed i
private float[] mMVPMatrix = new float[16];

/** How many elements per vertex. */
private final int mStrideBytes = 7 * mBytesPerFloat;

/** Offset of the position data. */
private final int mPositionOffset = 0;

/** Size of the position data in elements. */
private final int mPositionDataSize = 3;

/** Offset of the color data. */
private final int mColorOffset = 3;

/** Size of the color data in elements. */
private final int mColorDataSize = 4;

/**
 * Draws a triangle from the given vertex data.
 *
 * @param aTriangleBuffer The buffer containing the vertex data.
 */
private void drawTriangle(final FloatBuffer aTriangleBuffer)
{
    // Pass in the position information
    aTriangleBuffer.position(mPositionOffset);
    GLES20.glVertexAttribPointer(mPositionHandle, mPositionDataSize, GLES2
            mStrideBytes, aTriangleBuffer);

    GLES20.glEnableVertexAttribArray(mPositionHandle);

    // Pass in the color information
    aTriangleBuffer.position(mColorOffset);
    GLES20.glVertexAttribPointer(mColorHandle, mColorDataSize, GLES20.GL_
            mStrideBytes, aTriangleBuffer);
    GLES20.glEnableVertexAttribArray(mColorHandle);

    // This multiplies the view matrix by the model matrix, and stores the
    // (which currently contains model * view).
    Matrix.multiplyMM(mMVPMatrix, 0, mViewMatrix, 0, mModelMatrix, 0);

    // This multiplies the modelview matrix by the projection matrix, and
    // (which now contains model * view * projection).
    Matrix.multiplyMM(mMVPMatrix, 0, mProjectionMatrix, 0, mMVPMatrix, 0)

    GLES20.glUniformMatrix4fv(mMVPMatrixHandle, 1, false, mMVPMatrix, 0);
    GLES20.glDrawArrays(GLES20.GL_TRIANGLES, 0, 3);
}
```

Do you remember those buffers we defined when we originally created our renderer? We're finally going to be able to use them. We need to tell OpenGL how to use this data using GLES20.glVertexAttribPointer(). Let's look at the first call.

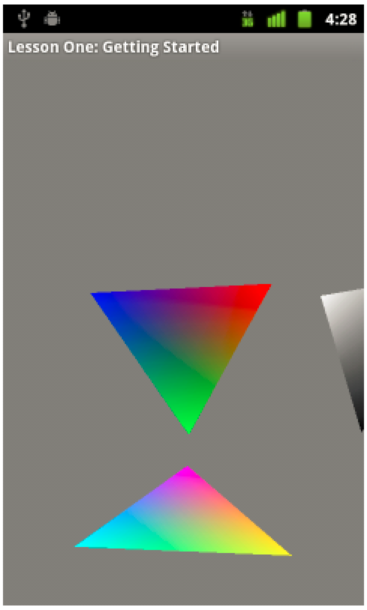
```
// Pass in the position information
aTriangleBuffer.position(mPositionOffset);
GLES20.glVertexAttribPointer(mPositionHandle, mPositionDataSize, GLES20.GL
        mStrideBytes, aTriangleBuffer);
GLES20.glEnableVertexAttribArray(mPositionHandle);
```

We set our buffer position to the position offset, which is at the beginning of the buffer. We then tell OpenGL to use this data and feed it into the vertex shader and apply it to our position attribute. We also need to tell OpenGL how many elements there are between each vertex, or the *stride*.

*Note: The stride needs to be defined in **bytes**. Although we have 7 elements (3 for the position, 4 for the color) between vertices, we actually have 28 bytes, since each floating point number takes up 4 bytes. Forgetting this step may not cause any errors, but you will be wondering why you don't see anything on the screen.*

Finally, we enable the vertex attribute and move on to the next attribute. A little bit further down we build a combined matrix to project points onto the screen. We could do this in the vertex shader, too, but since it only needs to be done once we may as well just cache the result. We pass in the final matrix to the vertex shader using GLES20.glUniformMatrix4fv() and GLES20.glDrawArrays() converts our points into a triangle and draws it on the screen.

RECAP



Whew! This was a big lesson, and kudos to you if you made it all the way through. We learned how to create our OpenGL context, pass shape data, load in a vertex and pixel shader, set up our transformation matrices, and finally bring it all together. If everything went well, you should see something similar to the screenshot on the right.

This lesson was a lot to digest and you may need to go over the steps a few times to understand it well. OpenGL ES 2 takes more setup work to get going, but once you've been through the process a few times you'll remember the flow by the back of your hand.

Publishing on Android Market

When developing apps, we wouldn't want people unable to run those apps to see them in the market, otherwise we could end up with a lot of bad reviews and ratings when the app crashes on their device. To prevent an OpenGL ES 2 app from appearing on a device which doesn't support it, you can add this to your manifest:

```
 <uses-feature
android:glEsVersion="0x00020000"
android:required="true" />
```

This tells the Market that your app requires OpenGL ES 2, and it will hide your app from devices which don't support it.

EXPLORING FURTHER

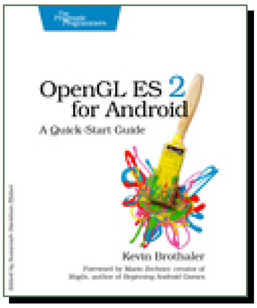
Try changing the animation speed, vertex points, or colors, and see what happens!

The full source code for this lesson can be downloaded from the project site on GitHub.

A compiled version of the lesson can also be downloaded directly from the Android Market.

I also recommend looking at the code in *ApiDemos*, which can be found under the Samples folder in your Android SDK. The code in there helped me out a lot when I was preparing this lesson.

Please don’t hesitate to ask any questions or offer feedback, and thanks for stopping by!



Buy Now

ABOUT THE BOOK

Android is booming like never before, with millions of devices shipping every day. In *OpenGL ES 2 for Android: A Quick-Start Guide*, you’ll learn all about shaders and the OpenGL pipeline, and discover the power of OpenGL ES 2.0, which is much more feature-rich than its predecessor.

It’s never been a better time to learn how to create your own 3D games and live wallpapers. If you can program in Java and you have a creative vision that you’d like to share with the world, then this is the book for you.

Share / Save …

Author: Admin

Kevin is the author of OpenGL ES 2 for Android: A Quick-Start Guide. He also has extensive experience in Android development. View all posts by Admin

Admin / May 29, 2011 / Android, Android Tutorials / camera, fragment shader, matrices, modelview matrix, programs, projection matrix, shaders, vertex shader

328 thoughts on “Android Lesson One: Getting Started”

miguel

June 13, 2011 at 3:27 am

Hola,

This is a very good intro to the OpenGL ES 2.0 on Android.
(at least the best I have seen so far on the net!!)

Thanks a lot and looking forward to read the next lesson…

🙂

Admin

June 13, 2011 at 11:42 pm

Thanks, Miguel, I appreciate the feedback! The next lesson will be up hopefully soon!

John Mark Isaac Madison

June 22, 2015 at 9:21 pm

I stumbled upon this via the android app store after suspecting the 2 tutorials I did
were failing because JellyBean did not support openGLES2.0.

It has been 2 days of research and I have done 3 triangle tutorials.
All of them failed. Was able to get a colored screen. But that is as far as I could get.
The checking of the compile state of the shader I had to learn from a lecture.
Which I then implemented to error check my tutorial code. But it didn't work.

This looks very promising and seems to cover everything I have found on my own
via lectures and tutorials.

AKA: This information looks dense and high quality.

-John Mark

Pingback: Android Lesson Two: Ambient and Diffuse Lighting | Learn OpenGL ES

Pingback: Android Lesson Three: Moving to Per-Fragment Lighting | Learn OpenGL ES

Pingback: WebGL Lesson One: Getting Started | Learn OpenGL ES

---

patrick

September 11, 2011 at 6:50 pm

This was very helpful – thanks for posting it.

Lawrence D'Oliveiro

September 26, 2011 at 7:47 am

Anybody looking for an intro to 3D geometry could start with this section of the "Blender 3D: Noob to Pro" WikiBook:

http://en.wikibooks.org/wiki/Blender_3D:_Noob_to_Pro/3D_Geometry

Admin

March 9, 2012 at 3:27 pm

Thanks for sharing this. 🙂

Pingback: Android Lesson Five: An Introduction to Blending | Learn OpenGL ES

Stof

October 3, 2011 at 8:32 pm

Lots of tutorials on OpenGL ES 1.0 on the net, but those on OpenGL ES 2.0 are hard to find … Yours is great, thanks a lot.

RuiLang

November 10, 2011 at 7:30 am

This was very helpful for me!

Thanks for posting it.

Pingback: Android Lesson Four: Introducing Basic Texturing | Learn OpenGL ES

John

December 21, 2011 at 7:28 pm

This is the most concise, detailed and best explained opengl es 2 tutorial for android beginners. I have been googling for days and there is almost nothing out there except the google tutorial that only part works and has warnings and intermittently crashes my phone.

The way you describe the camera matrix is so clear and has ended the confusion caused by the google tutorial.

Big thanks!!

Consider changing the article title because you cannot find this by searching with the expected terms. I only stumbled across it because I saw the demos on Android marketplace.

Admin

March 9, 2012 at 3:28 pm

We spoke a bit in email, but thanks again for the great feedback. I have started to promote the site more and add tags, so I hope that this makes it easier to find the tutorials in the future.

Stephan

December 24, 2011 at 8:30 pm

Nicely done!
Any specific reason why your clear color has an alpha of 0.5? (Set in onSurfaceCreated() with GLES20.glClearColor(0.5f, 0.5f, 0.5f, 0.5f))

Stephan

December 24, 2011 at 8:32 pm

Never mind, I see you never turn on alpha blending.

Admin

March 9, 2012 at 3:30 pm

Yep, there is no specific reason for it since no alpha blending being done. Even if it was turned on I guess it wouldn't matter unless we were using destination alpha.

wangjunyong

January 20, 2012 at 9:45 am

really helped me a lot

EEYUGH.

February 8, 2012 at 12:39 am

Holy shit. This is fucking stupid.

Why the fuck would anyone ever design an API so that you have to pass strings into a compiler, and then link things up AT FUCKING RUNTIME.

Honestly, are you people fucking assholes. You just want us all to be miserable failures, bang our heads against some shitty wall, don't you?

First of all there are better ways to pass character data through a Java VM. And second, this shit just shouldn't be happening at runtime. Plain and simple, end of story.

kangta

February 24, 2012 at 12:41 am

Sounds like someone hasn't ever programmed in Java or OpenGL for that matter.. Why else would you be reading this? Please elaborate on your better way to pass a character data besides a final String to the function glShaderSource(int handle, String src). Inline OpenGL source in a final String is the easiest way I can think of. It happens elsewhere, ever see Linux kernel source? There's lots of inline-asm. Oh and linking at runtime? Ever hear of DLLs, kernel modules, or drivers? It happens all the time.

OpenGL ES is ported to various platforms of ranging capabilities. OpenGL hides the complexities of various hardware accelerators and platforms to provide a unified interface. Thus the API and the inline rendering source code.

Stop your whining bitch and just learn, asshole

Admin

March 9, 2012 at 3:33 pm

Good points, kangta. It’s also possible to read in the shader source from a text file, and it’s even possible to store GPU-specific compiled binaries (see: http://www.khronos.org/opengles/sdk/docs/man/xhtml/glShaderBinary.xml) so I’m not sure I understand all the swearing and hate from the originator. 😉

Martin

August 26, 2012 at 3:16 pm

That’s interesting. How do I compile the shader on a computer? or shall I compile it once on the phone when the game is started first and store it afterwards in a binary file cos it’s GPU-specific.
I’m going to implement bump mapping and specular lighting, which will make my shaders very large. Precompiled shaders could make the loading times shorter.

Admin

August 27, 2012 at 2:46 pm

I don’t think every phone will support this, but for those that do you could just precompile on the phone itself on first launch.

John Mark Isaac Madison

June 22, 2015 at 9:25 pm

Lol. You seem to know what you are talking about.
As someone who has done both game and ui programming.
I see nothing wrong with openGL's approach here.

Initialization can be clunky and non-optimized.
So strings are acceptable.

And they even have handy methods for checking if the inlined strings have errors!
I wish other openGL tutorials would have told me about that!

-John Mark

Sam Tipton

December 7, 2015 at 9:20 pm

final strings are concatenated at compile time in java

LarsD

February 26, 2012 at 9:15 am

Thanks for the tutorial, it's very very helpful.

Annotations:

Unfortunely the Download is not complete ready for Eclipse. I had to edit the .classpath file, I inserted the lines, now it works (Eclipse Indigo/Android 3.2):

At Lesson One there is something wrong with the html version. The declaration of "fragmentShaderHandle" is missed. The download project do not contains this error.

Yours, Lars

Admin 👤

February 27, 2012 at 1:39 am

Hi Lars,

The comment did not contain the lines, (WordPress probably interpreted as HTML and filtered them out) but I updated the project, tried importing it as read-only from github and it seems to work now. Let me know if there are still issues.

Also not sure what you mean about the HTML version.

Let me know, and thanks! 🙂

Mohsen Navabi

September 3, 2012 at 2:55 pm

Thanks for your helpful tutorial

the only problem is that "fragmentShaderHandle" definition and codes is missed in the tutorial

you can just easily press CTRL+F when you are in browser opening the Lesson 1 tutorial page and then you will see that you use "fragmentShaderHandle" but u hadn't declare it before

Admin 👤

September 4, 2012 at 4:27 pm

Good catch, the steps for the fragment shader were left out, as it's the same sequence of code as for the vertex shader.

Pingback: Android Lesson Seven: An Introduction to Vertex Buffer Objects (VBOs) | Learn OpenGL ES

PDUBB

March 11, 2012 at 8:26 pm

I'm guessing, aside from touch or tilt listeners, this code is completely portable so you can plug in the verticies, surface, color, motion varibles, camera data… after it's been written once.

Because I'm new, I have to ask, will the tutorials end with standard code we can customize with geometry and textures from an obj file created with a CAD program?

BTW Great tutorial! Thanks for what must have been a tremendous effort.

Parksungsu

March 28, 2012 at 5:17 am

Hi, My name is Austin.
Currently, I practice the project of OpenGL ES 2.0 on Android ADT.
But It didn't operate on ADT. So, I have found more information.
Today, I find your information about the Android emulator does not support OpenGL ES 2.
It is good information to me. So, I am going to buy Galaxy Nexus.
At that time, I will make the project.

Admin

April 3, 2012 at 12:04 am

Hi Austin,

That’s right, unfortunately the emulator doesn’t support OpenGL ES 2. Most devices should have support for it now. Thanks for the comment! 🙂

Ned

April 17, 2012 at 11:28 pm

Great tutorial, just what i was looking for. Much appreciated

Ned

April 17, 2012 at 11:34 pm

=O i just went to your github and you have 7 lessons on there, THANK YOU!!!

Admin 👤

April 22, 2012 at 8:17 pm

No problem, hope you find it useful!

Oscar

April 22, 2012 at 4:19 am

Seriously, the best tutorial on the Internet for OpenGL ES 2.0.
I’m amazed.

Please don’t turn off this website.
I bookmarked some sites that went down.

Hopefully you guys keep this site alive and make more tutorials!

Admin 👤

April 22, 2012 at 8:17 pm

Thanks for the kind compliments, Oscar, and no worries, I'm in it for the long haul!

Gun

April 23, 2012 at 10:00 am

Hi, my name is Gun, its been a week i'm trying to figured it out how to do transformation in opengl es 2.0. Well some tutorial maker often did less or much for their "lesson one". Most of them missed the most commonly operation which is transformation, and they skip it or sometimes they put too much on the camera view thats makes the whole scene of the tutorial very confusing.

I wanted to thank to you for providing this tutorial, especially lesson one. its very clean and shows clearly the base we step on. Thank you again. =)

Admin

April 26, 2012 at 2:56 pm

Thanks, Gun, I'm glad that it helped out!

Hobbyist

April 25, 2012 at 6:56 pm

Before I start posting, first thing's first, great website!

I've recently got interested in android phone programming and tried out some 3D modelling and stumbled upon your website, manage to understand how opengl es 2.0 works in general and all that stuff!

But you'd guess it, since I'm posting, I must have hit a problem!

This tutorial works fine when I deployed it to a real android device, but it crashes immediately on the SDK emulator, I was wondering if you're aware of why it's happening, I've did my fair share of googling and could come to no conclusion =(

I have the lastest emulator/SDK installed, with GPU emulation enabled to support opengl es 2.0, yet this error message pops up the moment I click on the application:

! Sorry!
The application
LessonOneActivity(process
lesson.One) has stopped
unexpectedly. Please try again.

Force close

If there's anymore information you think would help with pinpointing the problem, I'll gladly provide it!

Thanks!

Admin

April 26, 2012 at 2:55 pm

Hi Hobbyist,

Thank you for the compliments! For the crash, please change the following line: final boolean supportsEs2 = configurationInfo.reqGlEsVersion >= 0x20000;

to : final boolean supportsEs2 = true;

This is a bug with the emulator, even if you have GPU emulation turned on!

Let me know if this helps.

Thanks!

otrotipomas

December 18, 2012 at 4:07 am

hey there, I'm not Hobbyist but tested what you said.
You are right.
changing that line to final boolean supportsEs2 = true;
on the activity of each lesson is enough to avoid the error.

Thanks ! !

btw, it would be awesome if you could do a lesson about 3d models that change their shape over time.

Brajo

December 28, 2012 at 6:13 pm

I had the same thing, GPU enabled, latest api, but that change did not fix it. It would stil stop. Even running the OpenGL ES API Demos would also crash like that. After some digging around it turned out there was a problem with my EGL configuration. For some reason the EGL was not being configured automatically. The log cat would give the following error "java.lang.IllegalArgumentException: No config chosen" as the cause in the trace. Anayway, what I had to do was add a call to set the configuration manually, for example:

```
class MyGLSurfaceView extends GLSurfaceView {

public MyGLSurfaceView(Context context){
super(context);
setEGLContextClientVersion(2);

———-> setEGLConfigChooser(8, 8, 8, 8, 16, 8);
setRenderer(new MyGL20Renderer());
}
}
```

carl

January 8, 2013 at 5:44 pm

i have the same problem: java.lang.IllegalArgumentException: No configs match ConfigSpec. is it because this line -> //final boolean supportsEs2 = configurationInfo.reqGlEsVersion >= 0x20000; was replaced w/ this: final boolean supportsEs2 = true;?

in any case could you be more specific and show the code explicitly that allowed the lesson to run in the emulator. seems u can just place the call (setEGLConfigChooser(8, 8, 8, 8, 16, 8);) in the true branch of the if(supportsEs2) statement. if you can provide the code for the function it would be very much appreciated. thanks

Hobbyist

April 27, 2012 at 4:35 am

Hey,

Found out about that bug the hard way, after spending some time tweaking the emulator, I decided to delete all the checks in the code, then made a separate app to check for the GK version, what a trek, lol.

Turns out that there's some other incompatibilities with the new emulation too, it conflicts with certain firewalls like ZoneAlarm and will not run unless it's completely shut-downed.

Thanks for the great tutorials again! Hope to make something interesting soon!

Bobpantsspongesquare

May 1, 2012 at 5:49 pm

When LEARNING something new, the reader should not be forced to debug the tutorial code in order to get it to compile. Too many minor issues in this code that can cause those just getting started to get too frustrated to continue.

Admin

May 2, 2012 at 1:04 pm

Hi Bob,

Care to elaborate? I can't fix said minor issues if you don't point them out.

Thanks.

John Mark Isaac Madison

June 23, 2015 at 8:12 pm

This is my 4th OpenGL tutorial that has not worked.
Perhaps you've learned something since you posted this?
I really need help. I am feeling really stupid.
This is my 3rd day of focusing ONLY on getting a triangle on the screen.

-John Mark

leor8a

April 13, 2016 at 11:12 am

The debugging is as most as useful as the main content. It's important to understand it, if you start debugging at the start, in the future is useful if you're developing a more complex app.

Lawrence Shi

May 12, 2012 at 5:20 pm

Is it possible that admin write the tutorial in native layer (c)?
That would be great.
But thanks for the posting, really.

Admin

May 14, 2012 at 3:39 am

NDK code? That would be interesting. I should do it sometime. 🙂

HB

May 16, 2012 at 10:41 am

+1

Pingback: OpenGL ES Android Confusion

KP

May 30, 2012 at 9:07 am

I know this might be a one in thousand messages thanking you, but for me its a HUGE help!!!Thanks so much for awesome tutorials bud, keep up…

Admin

June 6, 2012 at 2:19 am

No problem, I appreciate each and every comment. 🙂

Andre

June 11, 2012 at 10:09 am

This looks like a very good introduction to OpenGL on Android.

I did stumble upon a compile error with line …

```
mGLSurfaceView.setEGLContextClientVersion(2);
```

I created the project for API version 7, "setEGLContextClientVersion" appears in API version 8.

Just thought I'd mention that. Luckily, it probably doesn't happen to many developers.

Andre

June 11, 2012 at 10:27 am

So, the code becomes

```
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);

mGLSurfaceView = new GLSurfaceView(this);
mGLSurfaceView.setRenderer(new LessonOneRenderer());
setContentView(mGLSurfaceView);
}
```

Andre

June 11, 2012 at 10:34 am

P.S. I do realise that OpenGL ES 2.0 is from version 8+. I am just using 7 at the moment. A good tutorial for OpenGL 1.x users.

---

afuhrtrumpet

June 12, 2012 at 1:11 pm

I wrote out everything from this lesson in Eclipse, yet when I try to run it, I get errors for the overridden methods in LessonOneRenderer saying that they must override a superclass. It must not be recognizing the renderer interface because when I remove the @Override, the methods are simply not called. What is going on here?
Stack Overflow Link

Admin

June 13, 2012 at 11:12 pm

I believe the errors stem from using Java 1.5 instead of Java 1.6. You can't use @Override on interface methods in 1.5 IIRC.

Try changing your project properties to use Java 1.6. The methods should definitely get called so long as you call gLSurfaceView.setRenderer(new YourRenderer()); from your activity.

afuhrtrumpet

June 26, 2012 at 12:41 am

I tried setting the compliance to 1.6, but still the compiler complains about the @Override, and getting rid of it still ensured that the methods are not called.

Admin

June 26, 2012 at 2:45 pm

That is so strange… are you testing on an emulator or device? Could you publish the smallest set of code possible (including activity, manifest, and renderer) to GitHub or StackExchange? I'm curious.

Demios

June 13, 2012 at 1:34 pm

Hi. Nice Tutorial.

Is there a specific reason that you use the modifier “final” for many of your variables? Even the ones that are in a short-lived scope. Is there an advantage to it?

Admin

June 13, 2012 at 11:09 pm

Hi Demios,

It’s mainly a style issue — it helps you avoid inadvertently changing the value of a variable where you didn’t want to do so. It can also be used to ensure that you assign a value to a variable through all of your code paths. I might have overused it in these early lessons. 😉

Anon

July 21, 2012 at 6:24 am

Hai.

I just want to say thanks for the tutorial.

THANK YOU ! 🙂

Admin

July 27, 2012 at 2:30 pm

No problem, and thank you for stopping by 🙂

Anon

July 21, 2012 at 6:46 am

I am the Anon above who say thanks.
Btw, people should check this site too for good OpenGL ES 2.x tutorial. 🙂

http://db-in.com/blog/2011/01/all-about-opengl-es-2-x-part-13/

Admin

July 27, 2012 at 2:30 pm

Yes, these are good tutorials as well.

Raghav

December 29, 2015 at 5:09 pm

The link above is dead i guess. Getting a 404 in firefox

sankar

July 26, 2012 at 9:44 am

hi,

I am getting the below error

the method setLookAtM is undefined for the type Matrix. when i cam creating anndroid project i have selected android 4.1 and minsdk as 2.3.3 version.

Admin

July 27, 2012 at 2:30 pm

Hi Sankar,

You'll probably want to check you imported android.opengl.Matrix and not android.graphics.Matrix.

sankar

July 28, 2012 at 10:13 am

ya it worked fine thank you

i have copied the code for lesson one and it working fine. I have understood the code on a high level, but there are some conceptual points that i am not quite clear on.

when you defined triagle1vertices date, where is origin located in terms of mobile screen (is it center of the screen or top right or top left). same question when you defined camera with setlookatM method

thanks
sankar

sankar

July 28, 2012 at 10:36 am

Just to add what i have been trying.. i have commented out the code for triangle2 and 3, also commented the code for rotation of trainle1. so what i have finally is a static triangle at the middle of my mobile screen. now i was trying to change the value for below parameters to understand the effect of the camera position on final display of triangle. but not quite understanding the effect.

```
// Position the eye behind the origin.
final float eyeX = 0.0f;
final float eyeY = 0.0f;
final float eyeZ = 1.5f;

// We are looking toward the distance
final float lookX = 0.0f;
final float lookY = 0.0f;
final float lookZ = -5.0f;
```

// Set our up vector. This is where our head would be pointing were we holding the camera.
final float upX = 0.0f;
final float upY = 1.0f;
final float upZ = 0.0f;

sankar

July 28, 2012 at 5:12 pm

sorry for troubling you with so many questions..i have debugged the program and saw that
mViewMatrix has values below

1 0 -0 0

-0 1 -0 0

0 0 1 0

0 0 -1.5 1

I would like to know how this matrix is generated

sankar

July 29, 2012 at 11:09 am

sorry the matrix was actually
1 0 -0 0
-0 1 -0 0
0 0 1 0
-1 -1 -1.5 1.

i understood the calculation from site http://webglfactory.blogspot.in/2011/06/how-to-create-view-matrix.html. also i have got projection matrix calculation from http://www.songho.ca/opengl/gl_projectionmatrix.html. so at least i am clear about how view and projection matrices gets generated :). but still i have below questions

– In the tutorial we have set the model matrix to identity matrix. is there is any reason for it? can we use any random 4×4 matrix for it?

– Also when a_position is filled from triangle vertices data which has 3 coordinates for each vertex (ignore the color). when we multiply a_position with MVP matrix which 4×4 in the shader, is one gets appended as 4th coordinate for each vertex (otherwise matrix multiplication is not possible right)?

-the values that we assign for x,y,z coordinates for vertices, are they influenced by view and projection matrices. what i mean to ask is, based on the matrices that gets generated, can we decide in which value ranges x, y and z values of vertices should lie in order them to be visible on device screen?

thanks,
sankar.

Admin

July 29, 2012 at 9:16 pm

Hi Sankar,

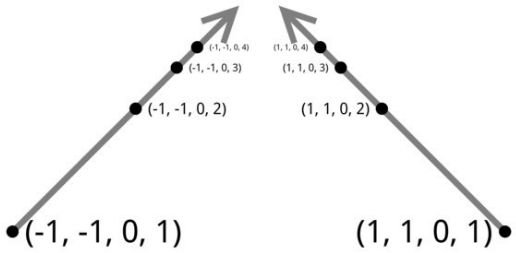
I would have pointed you over to those sites as they have a great description of how the matrices themselves get generated. I've learned quite a bit, myself, since I wrote these first programs. 😉

So at a high level, what's happening is that in OpenGL, normally -1, -1 is the bottom left corner and +1, +1, is the top right corner. These are known as normalized device coordinates.

To create the illusion of 3D, we use a projection matrix. The matrix itself doesn't create the 3D effect; what it does is it maps all of our Z positions onto a special coordinate component known as "W", and OpenGL will later divide X, Y, Z by their Ws.

This PDF goes into more details:
http://www.terathon.com/gdc07_lengyel.pdf

This image shows how the same coordinate gets closer to the center of the screen as the W value increases:



All a projection matrix will do is something like this. Say you have two source positions of the following:

(3, 3, -3)
(3, 3, -6)

The second point is a little bit further, or more “into the screen” than the first. The projection matrix will convert the coordinates into something like this:

(3, 3, -3, 3)
(3, 3, -6, 6)

That last component is W. Now, OpenGL will divide everything by W, so you get something like this:

(1, 1, -1)
(0.5, 0.5, -0.5)

Remembering that these are in normalized device coordinates, where -1, -1 is the lower left corner of the screen and +1, +1 is the upper right corner.

Model and view matrices

Alright, so once we have a projection matrix setup, we need to move stuff in and out of this viewspace so that we can actually see it.

This is where we use a model and a view matrix. The main purpose of these matrices is to translate and rotate objects so that they end up within our view frustum, and we can see them on the screen. They both have the same function — the view matrix does the same ting that the model matrix does. The difference is that the view matrix applies to all objects at the same time, so it's like moving around the entire world. The model matrix, on the other hand, is applied to a single object at a time, so we use this to take the definition of the object, and place it somewhere in our world. We can take the same data, render it once at 0, 0, 0, render it a second time at 1, 1, 1, and so on.

So, to come back and answer your questions:

"In the tutorial we have set the model matrix to identity matrix. is there is any reason for it? can we use any random 4×4 matrix for it?"

We reset a matrix to identity because an identity matrix, if multipled with any vector, returns the same vector. It's like multiplying any number by 1 — we get back the same number. This is to get a matrix that doesn't have any translations, rotations, or whatever applied to it.

We could use any matrix we want here, just as long as we understand what the results on the object will be.

"Also when a_position is filled from triangle vertices data which has 3 coordinates for each vertex (ignore the color). when we multiply a_position with MVP matrix which 4×4 in the shader, is one gets appended as 4th coordinate for each vertex (otherwise matrix multiplication is not possible right)?"

That's correct. When using attributes, OpenGL assigns a default of 1 to the 4th component if it's not specified.

"-the values that we assign for x,y,z coordinates for vertices, are they influenced by view and projection matrices. what i mean to ask is, based on the matrices that gets generated, can we decide in which value ranges x, y and z values of vertices should lie in order them to be visible on device screen?"

Yep that's pretty much how it works. The projection matrix influences how near/far you can see, and the view matrix influences what you see, by moving around and transforming the entire scene. If you wanted everything to be 10x smaller, you could use a view matrix that scaled down all coordinates by 10 before multiplying them with the projection matrix.

This comment is a little bit convoluted, but hopefully a few things make a bit more sense. If I were to redo the tutorials I'd take a different approach now, as I think all of the matrix stuff can be unnecessarily confusing in the very beginning.

Admin

July 29, 2012 at 9:44 pm

Another way of understanding the matrix code:

Column-Major order

The numbers are all written in column-major order, which means the array offsets are specified like this:

```
0  4   8  12
1  5   9  13
2  6  10  14
3  7  11  15
```

The view matrix

Meaning the view matrix actually looks like this:

1, 0, 0, 0,

0, 1, 0, 0,

0, 0, 1, -1.5

0, 0, 0, 1

This matrix is almost an identity matrix, except for that "-1.5". This will end up translating the Z by that amount, which has the effect of pushing everything -1.5 units into the screen. We could remove this matrix if we applied the same translation to the model matrix, instead.

To verify this, try it out with a matrix calculator:

http://www.math.ubc.ca/~israel/applet/mcalc/matcalc.html

Projection matrix

As for the projection matrix, here are the rounded values:

```
1.5, 0,  0,    0,
0,   1,  0,    0,
0,   0, -1.2, -2.2,
0,   0, -1,    0
```

This will transform coordinates as follows (with default W of 1 in the source coordinates):

```
(1, 1, -1, 1) --> (1.5, 1,  -1, 1)
(1, 1, -2, 1) --> (1.5, 1, 0.2, 2)
(2, 2, -2, 1) --> (3,   2, 0.2, 2)
```

After division by W, we get this:

```
(1.5, 1,  -1, 1) --> (1.5, 1, -1)
(1.5, 1, 0.2, 2) --> (0.75, 0.5, 0.1)
```

```
(3,   2, 0.2, 2) --> (1.5, 1, 0.1)
```

Notice that the projection matrix just sets up the W, and it’s the actual divide by OpenGL that does the perspective effect.

Admin

July 31, 2012 at 3:39 am

No worries, this is how I learn myself 🙂

Admin

July 29, 2012 at 8:58 pm

Hi Sankar,

Looking back at the code, I believe that 0, 0 is the center of the screen.

sankar

July 30, 2012 at 7:59 am

hi,

You are amazing.. 🙂 it is so nice of you to take time and answer all of my basic questions. thank you so much. keep up the good work

thanks
sankar

james

October 2, 2012 at 4:20 pm

Hi, I am confuesd that how
(1, 1, -1, 1)
(1, 1, -2, 1)
(2, 2, -2, 1)
multiply
1.5, 0, 0, 0,
0, 1, 0, 0,
0, 0, -1.2, -2.2,
0, 0, -1, 0
to get
–> (1.5, 1, -1, 1)
–> (1.5, 1, 0.2, 2)
–> (3, 2, 0.2, 2)
if you want to get the answer i think the projection matrix should be
1.5, 0, 0, 0,
0, 1, 0, 0,
0, 0, -1.2, -1,
0, 0, -0.2, 0

Admin

October 2, 2012 at 5:52 pm

Hi James,

It seemed to work fine with the original matrix when I tried with an online matrix multiplier. Please check you're not confusing rows with columns; this is common as OpenGL uses Column Major order. That is, m[0] – m[3] refer to the first column, not the first row.

---

james

October 3, 2012 at 6:27 am

Aha, I am confused with rows.
Btw, I am a new one to OpenGLES, I want to know how to set the value of

view matrix, model matrix, and projection matrix. Is there any rules? Thanks.

Admin

October 3, 2012 at 1:25 pm

Hi James,

I'm gonna post a new article on this topic. Please keep an eye out. 🙂

Admin

October 3, 2012 at 3:06 pm

Here it is! http://www.learnopengles.com/understanding-opengls-matrices/

Pingback: ガラパゴスブログ » Blog Archive » WebGL レッスン 1: 導入

popo

July 28, 2012 at 6:16 am

hi!
thank's for your great tutorial, I'm very new to android and opengl es 🙂
one thing I'd like to ask, I'm trying to draw a cube in android which its width is equal to screen width. I did it but its width didn't equal to screen width.

So If I set GLU.gluPerspective(gl, 45.0f, (float)width/height, 1.0f, 100.0f), camera is on (0,0,0), and I don't use any glTranslatef or glRotatef, how should I set the vertices (especially the X axis) so the width of the cube will be the same as screen width?

thank you so much for your help 🙂

Admin

July 29, 2012 at 8:57 pm

Hi popo,

What would be the purpose of this cube? The most straightforward way could be to skip the matrix stuff entirely and just work in normalized device coordinates, which range from -1, -1 on the bottom left corner to 1, 1 on the top right corner. In your shader you'd remove "gl_Position = matrix * a_Position" and instead you'd put "gl_Position = a_Position".

popo

July 30, 2012 at 4:13 am

hello, thank's for the answer. Actually I'm going to draw some cubes on the top of some map tile images (just like google map 3D, but in a simple version). On a certain zoom level, one of my object on that map has its width equals to screen width, so my cube had to have its width equal to screen width too. So I tried to draw that cube but later I found that my cube didn't have a screen-width width >.< This seems useless to build that app, but I really want to learn how to make it 😃
If you have any advice, I appreciate it so much 🙂

Admin

July 31, 2012 at 3:27 am

Hi popo,

Hmm, one way to do it is to see what kind of projection matrix you're ending up with. If you're using frustumM I think you can just take the right, left, top, bottom, and near plane as your coordinates!

For example:

```
(left, top) --------- (right, top)
|                                |
|                                |
|                                |
|                                |
(left, bottom) --- (right, bottom)
```

Setting the Z for all of those to whatever you had set as the near plane, inverted. For example, if your near plane is 1 put -1 as the Z. I think this should work, but you might need to use a very small offset (-0.999) if there are artifacts.

Admin

July 31, 2012 at 3:28 am

P.S. These would be coordinates after view/model takes effect, so this is feeding right into the projection matrix.

Admin

July 31, 2012 at 3:38 am

Also, I haven't tested this, but if you don't have the right, left, top, down, because you're using a helper function, try this:

Take the matrix value at m[0], and set X range = 1 / m[0]; Then you can use minus X range to plus X Range as the screen width. Something similar should work for the height at m[5]. Again I haven't tested this but let me know, as it makes good research material. 😉

popo

August 5, 2012 at 4:45 pm

hi! finally I don't use any matrix modification as my mistake was because I got wrong to pick half-height of near plane to be my half-width of near plane. Seems like it's very simple matter, but maybe it could help somebody someday 😃

I set GLU.gluPerspective(gl, 45.0f, (float)width / (float) height, 1.0f, 100.0f), so basically my near plane will be on -1.0f, and maximal half-width of my near plane was calculated : zNear * tan(fovy / 2) * aspectRatio = 0.230f –> my near plane half-width.
and that's all 😃
thank's alot for your help and sorry for bothering you with my question >.<

Admin

August 8, 2012 at 6:37 pm

Thanks for sharing this! 🙂

sankar

July 30, 2012 at 6:02 pm

Hi,

i am playing with the view matrix generated by setlookatM method. i have made the projection matrix as in identity matrix and model matrix is set to identity in your code. so only matrix that affects the vertices is view matrix. when i execute the program i do not see anything on the screen. why is that so?
view matrix is
1, 0, 0, 0,
0, 1, 0, 0,
0, 0, 1, -1.5
0, 0, 0, 1

these are vertices defined in th eprogram

// X, Y, Z,
// R, G, B, A
-0.5f, -0.25f, 0.0f,
1.0f, 0.0f, 0.0f, 1.0f,

0.5f, -0.25f, 0.0f,
0.0f, 0.0f, 1.0f, 1.0f,

0.0f, 0.559016994f, 0.0f,
0.0f, 1.0f, 0.0f, 1.0f};

the view matrix should only push the z coordinate right?

thanks
sankar

Admin

July 31, 2012 at 3:13 am

Hi Sankar,

If you are only using a view matrix, you'll want to be sure that the final coordinates end up in the range [-1, 1] for X, Y, Z, and W. They will need to be in this range to be visible on the screen. With the -1.5 it seems that they would not be in this range.

sankar

July 31, 2012 at 12:05 pm

hi

the view matrix in the example is
1, 0, 0, 0,
0, 1, 0, 0,

0, 0, 1, -1.5
0, 0, 0, 1

projection matrix is

1.508 0.000 0.000 0.000
0.000 1.000 0.000 0.000
0.000 0.000 -1.220 0.000
0.000 0.000 -1.000 0.000

final MVP matrix is
1.508 0.000 0.000 0.000
0.000 1.000 0.000 0.000
0.000 0.000 0.278 0.000
0.000 0.000 -1.000 0.000

model matrix is set to identity matrix and i did multiplication from right to left (V multiplied by P and the result is multiplied by M). hope i am right.

now with the above matrix i have multiplied each of the vertices (arranged column wise)

-0.500 0.500 0.000
-0.250 -0.250 0.560
0.000 0.000 0.000
1.000 1.000 1.000

the result coordinates are

-0.754 0.754 0.000
-0.250 -0.250 0.560
0.000 0.000 0.000
0.000 0.000 0.000

apparently W in this case is 0 an divide with 0 is not allowed(by the way i can see the triangle on the screen) so how this situation is handled. am i missing something?

thanks
sankar

Admin

August 1, 2012 at 4:21 pm

Hi Sankar,

You should really put the Z coordinates as something like -2 so that they end up in the range between -1 and 1 after multiplied with the matrix.

I'm surprised that things still show up for you, since clip space is defined as -W to +W. I guess it also says somewhere in the specs that the graphics card isn't supposed to blow up if W is zero. 😉 I couldn't find a clear answer myself but would be interested if someone has one.

Admin

August 1, 2012 at 4:29 pm

Also it seems a bit odd that the last column of your projection matrix is all zeroes. What are you passing in to frustumM ?

sankar

August 1, 2012 at 5:37 pm

hi,

I took the values from your code. in fact i haven't changed any values in any of the functions. i have commented out rotations part of code. i debugged it and those are matrices i see.

thanks
sankar

Admin

August 1, 2012 at 6:00 pm

Interesting, I get a matrix that looks more like this:

```
1.5, 0,  0,    0,
0,   1,  0,    0,
0,   0, -1.2, -2.2,
0,   0, -1,    0
```

I think there should be something in that last column…

Admin

August 1, 2012 at 6:00 pm

What happens when your code gets to Matrix.frustumM(mProjectionMatrix, 0, left, right, bottom, top, near, far); ? What are the actual values?

sankar

August 2, 2012 at 5:25 am

sorry i messed up last time.

this is proj matrix after the frustrumM code

1.508 0.000 0.000 0.000
0.000 1.000 0.000 0.000
0.000 0.000 -1.220 -2.200
0.000 0.000 -1.000 0.000

after multiplication with viewmatrix result is

1.508 0.000 0.000 0.000
0.000 1.000 0.000 0.000
0.000 0.000 -1.220 -0.370
0.000 0.000 -1.000 1.500

sorry for the confusion

Admin

August 8, 2012 at 6:36 pm

Ah that makes more sense then! Then it's probably just a question of making sure that you're pushing the right amount into the Z axis so that it appears on the screen. You can try with a matrix calculator (here is one: http://www.math.ubc.ca/~israel/applet/mcalc/matcalc.html) with that matrix, and see which values give you a Z inbetween -1 and W.

It seems like a point of (1, 1, -3, 1) should show up.

sankar

August 10, 2012 at 10:53 am

Hi,

Do you know any good tutorials for collision detection in 3D with open gl ES 2.0?

thanks

Admin

August 10, 2012 at 6:31 pm

I've heard good things about http://code.google.com/p/bullet/downloads/list and it seems that some Android games are using it!

http://bulletphysics.org/wordpress/ has a forum where you can ask questions and get more help.

Admin

August 10, 2012 at 6:34 pm

You could also try http://jbullet.advel.cz/ for a Java port, so no need to use the NDK!

kamal

August 10, 2012 at 1:54 pm

it's a great tutorial for beginner. Really helped me in getting, how openGl works..I tried to draw a point only when i touch the screen, i tried to play with vertices array, but didn't get expected result. Can you help with some sample or tutorial?

Admin

August 10, 2012 at 6:32 pm

You could try posting a question to StackOverflow with some sample code. Let me know and I'll see if I can help out, too.

Plasty Grove

September 7, 2012 at 8:05 am

First off, thanks so much for this amazing tutorial series. I'm sure it took a lot of hardwork and dedication to explain everything so clearly. The code is also well commented and documented.

I'm trying to develop a 3D game in android, probably not a first-person-shooter or anything, but a simple 2D game with 3D models. Will I be able to port the models to android using OpenGLES? I don't want to go with the standard mobile game frameworks and engines that are available since this is more fun! 🙂

Admin

September 11, 2012 at 4:38 pm

Thanks for the compliments, Plasty! We actually had a couple of community members working on an OBJ loader; will have to see how they're coming along.

Martin

September 12, 2012 at 7:29 pm

I'm so sorry I haven't said anything to this topic for weeks, but here is a short Code Snippet:
Basic Aspects of the .Obj FileFormat can be looked up here(http://en.wikipedia.org/wiki/Wavefront_.obj_file)

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;

import android.content.Context;

public class QREOBJFile_Parser {
public static float[][] qreparseObject(Context mActivityContext, int
resourceId)
{
final InputStream inputStream =
mActivityContext.getResources().openRawResource(
resourceId);
final InputStreamReader inputStreamReader = new InputStreamReader(
inputStream);
final BufferedReader bufferedReader = new BufferedReader(
inputStreamReader);

String nextLine;

String[] vector = new String[2];
String texvector;
String[] indexdata = new String[2];

List unindexedVectors = new ArrayList();
List unindexedNormals = new ArrayList();
List unindexedtexVectors = new ArrayList();
List Vectorslist= new ArrayList();
List Normalslist= new ArrayList();
List texVectorslist= new ArrayList();

try
{

while ((nextLine = bufferedReader.readLine()) != null)
{
if(nextLine.substring(0, 2).equals(“vn”))
{
vector = nextLine.substring(3).split(” “);
for(int j=0;j<3;j++)
{
unindexedNormals.add(Float.parseFloat(vector[j]+"f"));
}
}else if(nextLine.substring(0, 2).equals("vt"))
{
texvector = nextLine.substring(3);
unindexedtexVectors.add(Float.parseFloat(texvector.substring(0,8)+"f"));
unindexedtexVectors.add(Float.parseFloat(texvector.substring(9)+"f"));
}else if(nextLine.substring(0, 2).equals("v "))
{
vector = nextLine.substring(2).split(" ");
for(int j=0;j<3;j++)
{
unindexedVectors.add((float)
(Float.parseFloat(vector[j].substring(0,5)+"f")));
}
}else if(nextLine.substring(0, 1).equals("f"))
{
vector = nextLine.substring(2).split(" ");
int k;
for(k=0;k<3;k++){
indexdata =vector[k].split("/");
for(int l=0;l<3;l++){
Vectorslist.add(unindexedVectors.get((Integer.parseInt(indexdata[0])-1)*3+l
));

Normalslist.add(unindexedNormals.get((Integer.parseInt(indexdata[2])-1)*
3+l));

}
texVectorslist.add(unindexedtexVectors.get((Integer.parseInt(indexdata[1])-
1)*2));
texVectorslist.add(unindexedtexVectors.get((Integer.parseInt(indexdata[1])-
1)*2+1));
}
}
}
}

catch (IOException e)
{
return null;
}
System.gc();
final float[][] returndata=
{toPrimitive(Vectorslist),toPrimitive(texVectorslist),toPrimitive(Normalslist
)};
return returndata;
```

```
}
private static final float[] toPrimitive(List floatList){
final float[] floatArray = new float[floatList.size()];

for (int i = 0; i < floatList.size(); i++) {
floatArray[i] = floatList.get(i).floatValue(); // Or whatever default you want.
}
return floatArray;
}
}
```

Plasty Grove

September 14, 2012 at 4:18 am

Thanks so much for the code Martin. I'll have to meditate on it for a while though 🙂

Admin

September 14, 2012 at 3:21 pm

Thanks for sharing this, Martin! Do you have a GitHub or similar link as well? I could share that in an upcoming post. We had one other member who was working on an implementation, so we could share them both.

Angus Cheng

September 13, 2012 at 9:55 am

I think this tutorial is correct, but I think it's a bit silly to expect people to write 200+ lines of code and get it to run the first time.

Admin

September 14, 2012 at 3:17 pm

That’s why you have GitHub. Download the source if need be. 🙂

Hank

September 14, 2012 at 3:20 am

Hi, thank you for a great tutorial!
I am creating a map using OpenGL, my objects are getting loaded into an ArrayList from a Spatialite database in my activity class.
My main question is: how would I get my ArrayList to the renderer?

Admin

September 14, 2012 at 3:20 pm

You’ll need to use ByteBuffers and FloatBuffers to get your data into the native heap using syntax like the following:

```
ByteBuffer.allocateDirect(LENGTH_IN_BYTES)
.order(ByteOrder.nativeOrder())
.asFloatBuffer()
.put(ArrayList.toArray(new Type[]))
.position(0);
```

Once your data is there you can follow the techniques in these tutorials to use either glVertexAttribPointer with this vertex array, or upload it to the GPU as a vertex buffer object.

Admin

September 14, 2012 at 3:24 pm

Forgot to add, you’ll probably also need to cherrypick and serialize the data out of your array, so a simple put will not work.

Instead you can extract all of your data into a float[], short[] or byte[] array, depending on your needs, and then use a FloatBuffer, ShortBuffer, or

ByteBuffer as your view. You can then call .put with that array.

Hank

September 15, 2012 at 4:25 am

I must admit this is a little confusing for me. So basically the data in my ArrayList is held as a Geometry, its know that it is a polygon, point or linestring, I need to process each one accordingly to get node coordinates (x and y's) and create a float array. At the same time as creating the individual float arrays, I need to create a FloatBuffer. I will then need to keep both the float array and FloatBuffer in an array of their own. Is that logic of mine correct?

Some thing like:

```
//layer.geometries is a list of geometries in each table
ListIterator GeometryList_it = layer.geometries.listIterator();
while (GeometryList_it.hasNext()) {
Geometry geometry = GeometryList_it.next();

float[] shapeVerticeData;
FloatBuffer mShapeVertices;

if (geometry.getType() == SHP_POINT){

float cX = geometry.getX;
float cY = geometry.getY;

shapeVerticeData = {
// X, Y, Z,
// R, G, B, A
cX, cY, 0.0f,
0.0f, 0.0f, 0.0f, 1.0f};

// Initialize the buffers.
mShapeVertices = ByteBuffer.allocateDirect(shapeVerticeData.length *
mBytesPerFloat)
.order(ByteOrder.nativeOrder()).asFloatBuffer();
mShapeVertices.put(shapeVerticeData).position(0);

//Add both to an encompassing array
}
else if (geometry.getType() == SHP_LINESTRING){

}
else if (geometry.getType() == SHP_POLYGON){

}
}
```

Admin

September 16, 2012 at 3:10 pm

Hi Hank,

Yes, you'll need to extract node coordinates from your ArrayList and create a float array. You could start out by organizing it like this:

One array for all points.
One array for all lines.
One array for all polygons (they will need to be converted into triangles).

So something like this

```
ArrayList pointData;

for (Geometry geometry : layer.geometries) {
if (geometry.getType() == SHP_POINT) {
pointData.add(geometry.getX);
pointData.add(geometry.getY);
}
}
```

FloatBuffer pointBuffer = ByteBuffer.allocateDirect(pointData.size() * 4)
.order(ByteOrder.nativeOrder()).asFloatBuffer();

pointBuffer.put(pointData.toArray(new float[pointData.size()]));

// You can let pointData go out of scope now, you only need to keep around the FloatBuffer

If you want separate colors per point or stuff like that you can add that to the float buffer; otherwise, you could just use a uniform.

Admin

September 16, 2012 at 3:15 pm

Using one array works if you draw everything as GL_POINTS, GL_LINES, or GL_TRIANGLES because OpenGL will treat each as disjoint. You'll probably need to convert line strings as well, breaking into individual lines.

Hank

September 17, 2012 at 2:25 am

Thanks! So just a couple more general questions.
What if you wanted to select a line and have it highlight? Can you have that kind of interactivity?
Second question, can you have layers? On a map as you zoom in and out different layers should be visible? Maybe that can be controlled in what you do and don't draw, think I just answered that one.
However having an interactive map would make for a nice feeling map. I know I could capture where the user taps on screen then create a buffer of sorts to query what is at that location, highlighting an object though whether it is a point, line or polygon? If lines were broken up that makes it a little more tricky and can I alter the style or color in the buffer later maybe?

Admin

September 18, 2012 at 2:46 am

Yeah, you could for sure. You would just need to have a Java-side meta-data object controlling what's highlighted or not, and associate it with the offsets for the vertices associated with that object. For example, if you know that floats 10 – 20 correspond to one line loop, and there are two floats per position, then positions 5 – 10 correspond to the same line loop. Then if you want to highlight it, you just need to do this (pseudocode):

```
setUniform(Red);
drawArrays(LINES, start at 5, 5 positions);
```

To have layers, you're exactly right: just control what you draw and don't draw. you could use more than one vertex array and control the layers at that level. Alternatively you could use meta-data for this as well, and not draw the positions that correspond with a given layer. That could get a bit crazy though and less efficient because you'd have to break down one array into a bunch of draw calls. Better to keep it efficient and group each layer together.

For touch feedback it's a bit trickier. You'd need to map the touched coordinate to the appropriate source object. I haven't done too much research on this, so would actually be interested in what you come up with here! 😉

Hank

September 18, 2012 at 7:03 am

Hi Admin,

one more small question about the following line.
pointBuffer.put(pointData.toArray(new float[pointData.size()]));

I am getting the following:
"The method toArray(Object[]) in the type ArrayList is not applicable for the arguments (float[])"

If I do this:
pointBuffer.put(pointData.toArray(new Float[pointData.size()]));

I get:
"The method put(float) in the type FloatBuffer is not applicable for the arguments (Object[])"

Any ideas?

asaf

September 17, 2012 at 1:49 pm

I am trying to run the app samples (and they req the gpu emul)
but when i add gpu emulation to be trun on
the emulator is crashing on start up

I lookt it up on google and from what iv being reading it was a bug but it got fixt at v19 and I have v20

i have win7 64 gtx 8800

u happan to run in to this problem and know how to fix it?

Admin

September 18, 2012 at 2:36 am

Let me know if this post helps: http://www.learnopengles.com/android-emulator-now-supports-native-opengl-es2-0/

asaf

September 18, 2012 at 8:39 am

thats what i did
remove the supported check set the boolean to true
the problem is when i set the "GPU Emulation" to true (yes) it crash on start up
;/

Admin

September 18, 2012 at 8:35 pm

Could it be an issue with the emulator image? I was able to get it to work with Arm V7 4.0.3. I don't know if it works in other images. Unfortunately the support is quite buggy and might not work for all hardware configurations.

asaf

September 19, 2012 at 7:13 pm

yep just test it on my laptop and it works fine ,so this like you sayd hardware configuration not supported ;/
thx for the help sorry to bug you with this silly problem 🙂

Admin

September 20, 2012 at 1:28 am

It's too bad they haven't fixed these bugs yet. The Nexus 7 tablet isn't too expensive at least, so if you're looking for a hardware solution you could always go for that. 😉

Hank

September 20, 2012 at 7:28 pm

Hi,
So I’ve got my data in the buffers, I am going to try having line strings in one buffer to start with hopefully this won’t be a problem. I have created a drawLine method for them in my renderer and it seems to be working without error.

So I process as follows:
setContentView(mGLView);
I have my own CustomGLView that sets the renderer called mGLRenderer
then it calls
LoadMapData(); Which starts a new thread to show a progress wheel as it processes the geometry records.
After it is done it calls
mGLView.mGLRenderer.initDrawLayers();

```
public void initDrawLayers(){
GLES20.glClear(GLES20.GL_DEPTH_BUFFER_BIT |
GLES20.GL_COLOR_BUFFER_BIT);

ListIterator orgNonAssetCatLayersList_it =
default_settings.orgNonAssetCatMappableLayers.listIterator();
while (orgNonAssetCatLayersList_it.hasNext()) {
mapLayer MapLayer = orgNonAssetCatLayersList_it.next();

ListIterator mapLayerObjectList_it = MapLayer.objFloatBuffer.listIterator();
ListIterator mapLayerObjectTypeList_it = MapLayer.objTypeArray.listIterator();
while (mapLayerObjectTypeList_it.hasNext()) {

switch (mapLayerObjectTypeList_it.next()) {
case PointObject:
break;
case LineStringObject:
Matrix.setIdentityM(mModelMatrix, 0);
Matrix.rotateM(mModelMatrix, 0, 0, 0.0f, 0.0f, 1.0f);
drawLineString(mapLayerObjectList_it.next(),
MapLayer.lineStringObjColorBuffer);
break;
case PolygonObject:
```

```
break;
}
}
}
}

private void drawLineString(final FloatBuffer geometryBuffer, final FloatBuffer colorBuffer)
{
//Log.d(“”,”Drawing”);
// Pass in the position information
geometryBuffer.position(mPositionOffset);
GLES20.glVertexAttribPointer(mPositionHandle, mPositionDataSize,
GLES20.GL_FLOAT, false, mFloatStrideBytes, geometryBuffer);

GLES20.glEnableVertexAttribArray(mPositionHandle);

// Pass in the color information
colorBuffer.position(mColorOffset);
GLES20.glVertexAttribPointer(mColorHandle, mColorDataSize,
GLES20.GL_FLOAT, false, mFloatStrideBytes, colorBuffer);

GLES20.glEnableVertexAttribArray(mColorHandle);

// This multiplies the view matrix by the model matrix, and stores the result in the MVP matrix
// (which currently contains model * view).
Matrix.multiplyMM(mMVPMatrix, 0, mViewMatrix, 0, mModelMatrix, 0);

// This multiplies the modelview matrix by the projection matrix, and stores the result in the MVP matrix
// (which now contains model * view * projection).
Matrix.multiplyMM(mMVPMatrix, 0, mProjectionMatrix, 0, mMVPMatrix, 0);

GLES20.glUniformMatrix4fv(mMVPMatrixHandle, 1, false, mMVPMatrix, 0);
```

```
GLES20.glDrawArrays(GLES20.GL_LINES, 0, geometryBuffer.capacity());
}
```

The bounding rectangle for these lines is approx:
Top Left 152.068, 26.458
Bottom Right 152.769 27.565

What do I need to do to view it? Is it the view matrix that I need to manipulate. I will eventually add in zoom, pan, possibly rotation, functionality so I am also interested in what I would need manipulate for these as well.

Also I set setRenderMode(GLSurfaceView.RENDERMODE_WHEN_DIRTY); so it is not constantly running, how do you programmically tell the renderer to redraw.

Kind Regards Hank

Admin

September 21, 2012 at 1:38 pm

Hi Hank,

To tell it to redraw, you can call requestRender () on your GlSurfaceView.

You are quite right in that you'll need a view matrix to zoom, pan, rotate and so on. What you can do is something like this:

Projection Matrix: stores your overall projection
View Matrix: Stores global rotations, pans, whatever.
Model Matrix: Used for transforming individual objects in a scene, if needed.

You can prebake Projection * View into a matrix, then if you also need a model matrix, just multiply ProjectionView * Model and pass that into the shader. If your models are already stored in global coordinates, then you don't need a model matrix and can use the ProjectionView everywhere.

Regarding threading, please keep in mind that all calls that touch OpenGL *must* occur on the OpenGL thread. You can dispatch anything you need to this thread by calling GlSurfaceView.queueEvent(). Anything which touches Android views needs to happen on the main thread (the default thread that calls onCreate() and so forth, as well as your listener callbacks).

Hope this helps!

Anders

September 21, 2012 at 3:48 pm

Hey. It seems there is a bug with glGetShaderInfoLog() if the compilation process actually fails.

http://code.google.com/p/android/issues/detail?id=9953

I have seen some suggestions to workarounds, but I did not understand any of them (there is one in the comments of the GoogleCode link I posted)

Do you know how to get glGetShaderInfoLog() to actually return some information? Debugging shader code is virtually impossible without compiler output.

Admin

September 21, 2012 at 4:20 pm

Yikes. You could try the LIBGDX bindings to see if they work (instructions in this post: http://www.learnopengles.com/android-lesson-seven-an-introduction-to-vertex-buffer-objects-vbos/)

Anders

September 21, 2012 at 3:53 pm

And by the way. Thanks for the great tutorial 🙂

Admin

September 21, 2012 at 4:20 pm

You are welcome, thanks for stopping by 🙂

Hank

September 24, 2012 at 2:55 pm

Hi,

so I tried shifting your flat triangle (No.1) to the following:

(Basically the same size just a different location)

```
final float[] triangle1VerticesData = {
152.0f, -27.25f, 0.0f,
1.0f, 0.0f, 0.0f, 1.0f,
153f, -27.25f, 0.0f,
0.0f, 0.0f, 1.0f, 1.0f,
152.5f, -26.404508497f, 0.0f,
0.0f, 1.0f, 0.0f, 1.0f};

// Position the eye behind the origin.
final float eyeX = 152.5f;
final float eyeY = -27.0f;
final float eyeZ = 1.5f;
// We are looking toward the distance
final float lookX = 152.5f;
final float lookY = -27.0f;
final float lookZ = 0.0f;

// Set our up vector. This is where our head would be pointing were we holding the camera.
final float upX = 152.5f;
final float upY = -26.0f;
final float upZ = 0.0f;

Matrix.frustumM(mProjectionMatrix, 0, 152.0f, 153f, -27.25f, -26.404508497f,
1.5, 10);

Matrix.setIdentityM(mModelMatrix, 0);
Matrix.rotateM(mModelMatrix, 0, 0, 152.5f, -27.0f, 1.0f);
drawTriangle(mTriangle1Vertices);
```

This does not work it just comes back with a grey screen, I am obviously missing something here, can you see where I am going wrong?

Hank

September 24, 2012 at 3:06 pm

Whoops ignore the rotateM x and y should be set to 0.0f there.

Hank

September 24, 2012 at 4:50 pm

What I was attempting to do was shift both the triangle and camera to get the same picture from a different spot.

John Mark Isaac Madison

June 23, 2015 at 8:11 pm

Is perhaps the problem in your stride?
Which needs to be 7 * Float.SIZE
Which I am thinking Float.SIZE could be 4 or 8.
(The values are measured in bytes).
4 for 32 bit systems.
8 for 64 bit systems maybe?

I am also a bit lost. I’ve done a few openGL examples now and none of them work.
This seemed to be the most promising example… But this one just crashed on me.

If you have learned anything since posting this, I’d like to know.

-John Mark

Hank

September 25, 2012 at 8:56 am

Hi, Nevermind, tried shifting it a smaller amount and found that you don’t alter the Up vector or Perspective as those two are relational to the camera.

Hank

September 25, 2012 at 7:22 pm

How can I display my different layer objects with differing colors?

One is defined like:

```
final float[] pointObjColor = {1.0f, 0.2f, 0.0f, 1.0f};
final float[] lineStringObjColor = {1.0f, 0.3f, 0.0f, 1.0f};
final float[] polygonObjColor = {1.0f, 0.4f, 0.0f, 1.0f};
```

another is:

```
final float[] pointObjColor = {0.0f, 1.0f, 0.5f, 1.0f};
final float[] lineStringObjColor = {0.0f, 1.0f, 0.6f, 1.0f};
final float[] polygonObjColor = {0.0f, 1.0f, 0.7f, 1.0f};
```

On each layer the color is going to be the same for each geometry type. I figured that I would do it this way to save a little memory and it doesn’t need to be defined for each vertex, but obviously it needs to be applied to each vertex, if I’m understanding how the vertex shader works correctly.

Any ideas on how I could accomplish this?
Kind regards

Admin

September 25, 2012 at 9:16 pm

Hi Hank,

You could do this by using a uniform. An OpenGL uniform keeps its value until changed. Follow the code in this tutorial to see how to use a uniform.

The general gist will be:

Fragment shader:

uniform vec4 u_Color;
…
gl_FragColor = u_Color;

Java code:

Do this once, after building the shaders:
colorUniformLocation = GLES20.glGetUniformLocation(programHandle, “u_Color”);
…

Do this whenever you need to change the color:
GLES20.glUniform4f(colorUniformLocation, 0f, 1f, 0f, 1f);

Unlike with attributes, all elements of a uniform must be specified, so if you use a vec4, you have to use a corresponding Uniform4f. Hope this helps!

Hank

September 28, 2012 at 7:14 am

Hi,

that helps a lot thankyou!

I was also curious about two things:

– If I am receiving polygons from the database, which at the moment I am just rendering as GL_LINE_LOOPs, if I want to create a fill color, it will be a uniform color still but different from the border, I need to create an extra buffer that holds the indexing of my triangles, then I create a GL_TRIANGLE_STRIP, correct? What would be good practice to form my triangle index dynamically, for I won't know the number of sides?

– My other question is about text, I will be wanting to label certain objects on the map, one of the layers contains road center-line data and I will be wanting to place the road name along it. How would this be accomplished?

Regards Hank

Admin

October 1, 2012 at 5:06 pm

Hi Hank,

Indeed, you'd need to draw the interior differently. For convex shapes and some concave shapes, I think you could get away with drawing it as a triangle fan. Otherwise, you'd need to use an algorithm to prepare triangles out of the polygon. You'd have to do a bit more research on this. It could definitely make things easier to make for you if you store all the vertices once and refer to them with an index buffer.

For text, it's also going to be a bit tricky. There are many different ways of doing this, including overlaying a canvas on top of your OpenGL view and drawing to it using Android's text methods! You could try to start with http://stackoverflow.com/questions/8847899/opengl-how-to-draw-text-using-only-opengl-methods.

This seems like it will be an interesting project once it’s done!

Pingback: Android terrain test « Ghoshehsoft's Blog

Vu Phung

November 30, 2012 at 9:38 am

Hi all,

I try to create new project width android target is 2.3.3 then I copy the LessonOneActivity.java & LessonOneRenderer.java from the AndroidOpenGLESLesson project(I down from learnopengles).

I build and load into real android device 2.3.4 but I just got the error message: “Unfortunately Opengles20demo has stopped” although I tried to set supportsEs2 = true too.

I try run on emulator but the result is the same.

I saw in the AndroidOpenGLESLesson project there’re many files like the library such as:AndroidGL20.java… and in the libs folder are armeabi & armeabi-v7a… in my project not has.

Anyone could help me fix this bug?

Thanks and Best regards,
Vu Phung

Admin

December 6, 2012 at 4:01 am

Hmm, what happens if you try to run the code from the site by itself? You can avoid using the libs stuff if you’re targeting API 9 or higher.

Rajesh M D

December 6, 2012 at 3:38 pm

hi Admin..
i just read the lesson one getting started. wow.. really the best tutorial for OpenGL Android.
Thanks for the best tutorial.. 🙂
All the best for your future tutorials..:)

---

datdev

December 7, 2012 at 1:39 pm

This is the best source for OpenglES2 tutorials, i updated my knowledge from openGL ES 1.x to 2 in a couple of days by learning from this tutorials and porting my own game, thank you thank you thank you!!

Aaron

December 7, 2012 at 9:45 pm

This is the best site for OpenGL ES 2.0 that I have found to date….Keep up the awesome work.

Thank you so much.

Raymond

December 8, 2012 at 4:33 am

Hi y'all,

I'm getting a runtime error, actually a caught exception. The shaders fail to compile, and I have used the shaders included with your code as well as one from the android website.

Also, I am getting a “called unimplemented Open GL ES API” error.

I have tried so may websites and tutorials trying to get a GLSL program working to no avail. I have created working android OpenGLES programs, but as soon as I throw shaders into the mix, my programs break.

I’m starting with a straight up copy of your code from github before I create my own based on your tutorial, I just want to make sure it works first. So far, Im still frustrated. :/

Please help!

---

tranthuan

December 23, 2012 at 4:42 pm

Hello! Admin
The firt thing, I want to say thanks to your tutorials. It’s very very nice.
I have a question for you. (But it is’n about above subject).
I have an image (called img1).
+
I have a frame
I want to combine with them and result is an image (call img3)
(img2 have background is a part of img1 and size = frame)
following link to image detail:

img1

img2

Admin

December 27, 2012 at 8:34 pm

Hmm, will that image always remain the same? I would do it with a paint program. If it should move then one way you can do this is with alpha blending. What you do is you have a black & white image that is the outline of your shape above, with the black areas the parts you want to exclude and the white areas the parts you want to keep. You then blend that with the texture of the apples to get the part that you want to keep, like in the second image. I have a quick intro to multiplicative blending at http://www.learnopengles.com/android-lesson-five-an-introduction-to-blending/, and for more info, I'd recommend to search for "Alpha Masking" or "Alpha Blending".

Caleb Daniels

January 3, 2013 at 9:44 am

Well, im very new to the entirety of opengl and have been fiddling with it to try and realize an idea I had. Anyway, Im a tad confused on what exactly the up vector does in the display. I cant seem to find anywhere else that makes use of it except in your code. I was wondering if all that the up vector changed was the rotation of the camera, so to speak?

Sorry if my question is difficult to understand.

---

carl

January 4, 2013 at 4:36 am

hi, i downloaded the source and created android project from existing code.
edited
final boolean supportsEs2 = true; and ran as android application on emulator. it loaded on my emulator (android api17) and i selected the icon. lesson 1 thru 4 appeared. i tried each one but they each returned …

"Unfortunately Opengles20 tutorials has stopped
————————————————
ok"

running eclipse 4.2.1 fully updated on vista 32bit.

any suggestions on how to get it to work?

thanks

Nickless

January 12, 2013 at 9:56 am

Hi.

Hope you're still active, and thanks for some great tutorials!

But, can you explain a little thing:
I'm looking into ray-picking so I can determine when a user clicks on the triangle. To do so I need: modelview matrix and projection matrix. But what is the modelview matrix in this example?
As far as I understand each object that is drawn on the screen has its own model matrix, so I should not use that one as the modelview matrix. When I check older versions of openGL the modelview matrix can easily be fetched by calling glGetFloatv (GL_MODELVIEW_MATRIX, matrix); But can't find a function for doing this in OpenGL ES 2.0?

Thanks for any help!

Admin

January 15, 2013 at 2:33 am

In OpenGL ES 2 you keep track of the matrices yourself, so you always have access to them. For this example we have a mViewMatrix and a mModelMatrix, so we could get the modelview matrix by multiplying them together like this:

multiplyMM(mModelViewMatrix, 0, mViewMatrix, 0, mModelMatrix, 0);

You're right in that each object has its own model matrix, but since the model matrix is used mainly to put stuff into the "world", maybe what they mean is that you should use the "view" part of the viewmodel matrix, which represents the camera. I did a bit of object picking and I actually used the view matrix and the projection matrix, and created an inverted one using code like this:

```
multiplyMM(viewProjectionMatrix, 0, projectionMatrix, 0, viewMatrix, 0);
invertM(invertedViewProjectionMatrix, 0, viewProjectionMatrix, 0);
```

This will help you convert a 2D touched point into a 3D ray, with code like this:

```
 private Ray convertNormalized2DPointToRay(float normalizedX, float
normalizedY) {
final float[] nearPointNdc = {normalizedX, normalizedY, -1, 1};
final float[] farPointNdc = {normalizedX, normalizedY, 1, 1};
final float[] nearPointWorld = new float[4];
final float[] farPointWorld = new float[4];
multiplyMV(nearPointWorld, 0, invertedViewProjectionMatrix, 0, nearPointNdc,
0);
multiplyMV(farPointWorld, 0, invertedViewProjectionMatrix, 0, farPointNdc, 0);



divideByW(nearPointWorld);
divideByW(farPointWorld);

Point nearPointRay = new Point(nearPointWorld[0], nearPointWorld[1],
nearPointWorld[2]);
Point farPointRay = new Point(farPointWorld[0], farPointWorld[1],
farPointWorld[2]);

return new Ray(nearPointRay, Geometry.vectorBetween(nearPointRay,
farPointRay));
}
```

```
 private void divideByW(float[] vector) {
vector[0] /= vector[3];
vector[1] /= vector[3];
vector[2] /= vector[3];
}
```

I need to write up a tutorial to share more info, but hopefully that gives you a bit of a start with the resources that you're looking at!

dekana

January 18, 2013 at 6:50 pm

Hi,
my question is: you are sending position of vertices as 3 dimensional vector, but in vertex shader it is defined as 4 dimensional vector. I think the fourth dimension is w, but I don´t know where it is added.
best regards

Admin

January 18, 2013 at 7:48 pm

With an "attribute" the unspecified parameters are implicit, with the first 3 defaulting to 0 and the 4th defaulting to 1, so in this case W is defaulting to 1.

dlam

January 29, 2013 at 10:13 pm

hi,
i'm trying to find the source code on the github. is it under the webgl of android breifcase. I found a lesson 1 from webgl but it does not work/seem like its the same from this lesson and I looked through all the folders and I can not find another file for the life of me. Can you help direct me please?

Admin

January 30, 2013 at 3:04 pm

For the WebGL you may need a few additional steps to get it to work, this post has a few more details: http://www.learnopengles.com/how-to-embed-webgl-into-a-wordpress-post/

Jimmy

January 31, 2013 at 8:42 am

Hello Admin, It is really a good tutorial to know how the 3D shapes(triangles, cubes) can be drawn over GLSurfaceView. Now I am able to draw and rotate those shapes. I want to apply pinch/zoom effects to those shapes simply with fingers. I studied how to apply zoom effects for images but couldn't able to do the same for objects over GLSurfaceView.

Could you give me some demo example for that… If you want I can post my code over here..
Thanks in advance

Admin

January 31, 2013 at 8:23 pm

Hi Jimmy,

For a GLSurfaceView, you'll probably want to adjust the perspective or the view matrix in response to a pinch/zoom; for example, you could use a zoom in to decrease the FOV, and a zoom out to increase it, or you could use zoom in / out to move the camera in and out by adjusting the view matrix. I have no code off hand but I imagine you could use http://www.learnopengles.com/rotating-an-object-with-touch-events/ as a base.

Rui

June 4, 2013 at 3:09 pm

The best tutorial on openGl for android that I've found!

Damn it is a long tutorial, but keep them coming! Looking forward for more stuff!

Android App Development

June 6, 2013 at 10:16 am

That was very informative information about the Android. It something gives brief information on OpenGL. I have learn these OpenGL in C,C++ Programming languages. It was hard for me. But as far I have done more practice , its now easy for me.

Pierre

June 6, 2013 at 6:09 pm

Nice tutorial !

Suggestion: Next version should include some basic panning, zooming, selecting item, camera view movement……processing event in general That's what I found weak in your demo …lack of interaction 🙂

But….the demo build nicely ! Good job….thanks!

Alan

June 16, 2013 at 12:46 am

I pretty much copy and pasted the code that you have into my project, but when I try to run it on my actual device it displays a blank white screen, no triangles or anything. I've done other opengl tutorials and this problem happens to me everytime. Afterwards, eclipse says "source not found". Is it because my device doesn't support openGL es 2.0, or maybe because I forgot to import some jar file? Thank you in advance.

Admin

June 21, 2013 at 2:35 pm

Hi Alan,

What happens if you download the project from GitHub and import that into Eclipse?

Harold

June 27, 2013 at 1:13 am

Hi, I was wondering about the level of understanding one needs about Shader Languages when doing game programming. Do we need to be fluent with it? If so, do you have any introductory reference online/ebook?

P.S.
Awesome tutorials, by the way! The best I could find online! Although, there's one minor thing I would suggest (or maybe it's just me), though — and that is to add more diagrams. 🙂

Admin

July 8, 2013 at 10:07 pm

Hi Harold,

It would really depend on the complexity of the graphics within your game. For mobile-level graphics, I think you can get away with a limited understanding especially if it's a 2D game! For more complex graphics, you'll need to know more details. I haven't checked out any books specifically for shaders; when I was first learning myself I checked out "OpenGL ES 2.0 Programming Guide" which is more of a reference work. You can adapt the shaders there right onto Android.

Hank

July 4, 2013 at 3:39 pm

Hi, it's been a while since my last post, just getting back into things.

With your examples you place the 3 dimensions into FloatBuffers.

With my data, coordinates are kept as doubles, 2 dimensions only. So the fastest way to get this data out for me is by placing it into a DoubleBuffer with the 2 dimensions.

So what I am wondering is, is it possible to actually use DoubleBuffer? I have tried altering the following:

```
private final int mBytesPerFloat = 8;
private final int mPositionDataSize = 2;

private void drawPolygon(final DoubleBuffer geometryBuffer, final float[]
colorArray)
{
....
GLES20.glDrawArrays(GLES20.GL_LINE_LOOP, 0,
geometryBuffer.capacity()/2);
}
```

Although it is not throw any errors it is not drawing.

And if it is possible to only two dimensions then does the z just default to zero?

Hank

July 10, 2013 at 4:19 am

Hi Admin,

I have a FloatBuffer fbb. This holds about 17000 line loops, which is displaying fine, however I have just added a touch event handler to pan the map, only pan no zoom or rotate. onDrawFrame loops through each of the FloatBuffer's
I have found that it pans just fine but it is not what you would call a smooth panning experience.

Is there any way to make this more efficient or alternatives to my interactivity that you can see? Would appreciate your feedback!

I have placed an example of my code on to stack exchange: http://stackoverflow.com/questions/17562193/how-to-speed-up-rendering-with-opengl-es-2-android

Admin

July 10, 2013 at 7:27 pm

Hi Hank,

I saw you had some pretty good answers on Stack Overflow! From my experience, if your vertex shaders are simple then the mobile GPUs can handle very high polygon counts, so I wouldn't expect a slowdown to come from that. However, it's important that you batch the operations. It seems that you're drawing things by individual polygon and line, and this will be very slow. I'm not sure how fast line loop drawing is compared to polygon drawing, so that is also something you may need to address later on, by pre-rendering the lines into a texture and drawing a texture, instead.

It's hard to time stuff accurately, OpenGL buffers the calls and even the Android tracer gives inaccurate results, from my experience. What you can do is try removing a bit of code completely and time between runs, see how things change then.

Try removing Thread.currentThread().setPriority(Thread.MIN_PRIORITY);, and re-engineer the app to put your data into two vertex buffers: one for the polygons and one for the lines. Calling OpenGL thousands of times per second is most likely the source of poor performance. Draw those in two draw calls. If you need to modify the lists you can do that when needed, at the beginning or at the end of onDrawFrame().

eggmatters

July 11, 2013 at 5:20 am

Wow! I remember feeling kind of like EEYUGH when I dusted off an old openGL project and realized that the standard (non ES) 3.x version of open GL no longer supported immediate rendering. Wow. Instead of flaming the closest person I could find, I looked into it and researched the history of shaders (They were first developed by Pixar.) the graphics pipeline and the awesomeness of delegating all of the heavy lifting to the GPU. My kludgy scene which had about 10K polygons, 2 textures, lighting and fog took about 1.5 seconds to render a frame in immediate mode. It rendered flawlessly with shaders.
So, EEYUGH, ever heard of a dll in a linux kernel? Me either – get your facts straight if your'e going to bitch and preach.
Why have the overhead of a device driver to send data to a device driver?
Why or rather, how can dynamic rendering NOT occur at runtime? Jesus Christ you're a moron.
BTW, the java VM has nothing to do with this. The strings are actually "programs" which are written in a language called GLSL, not java. Open your eyes. Do those strings look like java to you?
If you're so offended by strings, write them as plaintext files and just stream the output. Christ you're dense.
Google GLSL and the rendering pipeline:
http://www.lighthouse3d.com/tutorials/glsl-core-tutorial/glsl-core-tutorial-index/ and see how far your ridiculous flame gets you there.
Pull your righteous head out of your ass and do some work. It's people like you who write shitty code so people like me (and the other eager participants on this site.) have to clean up.

Hank

July 11, 2013 at 6:38 am

Thanks appreciate the feedback.

I decided to optimize what I've got before hitting LOD's etc.
These two layers (parcel and roads) are not going to be changing. So I'll use the GL_STATIC_DRAW

I have placed all the individual FloatBuffers into one large FloatBuffer for each dataset.

There must be something else I need to do to the drawPolygon function, as it is throwing errors. I am now wondering how I go about adjusting my drawPolygon function to suit the batch style?

Also where the best place to set GLES20.glBufferData(target, size, data, usage) is?

Hank

Hank

July 11, 2013 at 8:34 am

I should have gone into more detail, the polygons and lines are drawing but they are now one continuous polygon and one continuous line. It is a bit of a mess to look at. However at this point it is now very fast and panning the map is smooth. Whether this changes by breaking up them up remains to be seen. Do I need to send GLES an array of start positions or something along those lines for each individual buffer?

Admin

July 11, 2013 at 12:57 pm

If you just want to set the data once you can call glBufferData from onSurfaceCreated. Can you find a way to generate the line data so that you can draw the lines individually instead of as line loops/strips? That would let you batch all the lines as well. This may increase the size of your data, but saving on thousands of draw calls may be worth it. If you use index buffers, the increase in size will be limited to the indices which can take up a lot less space than the vertices.

To continue to use different colors, you can add the color as an attribute of the vertex data instead of as a uniform. You can also calculate and pass in the matrix uniform once per overall draw instead of per line. No need to calculate it per individual line/polygon unless you want to move them around relative to each other.

Hank

July 11, 2013 at 1:34 pm

Yes I believe I can generate the lines that way, I'll read up on lesson eight before I start 🙂

Hank

July 12, 2013 at 12:38 am

Just thinking if I have this 17000 polygons some of them have 4 sides a lot of them have more. So if we say at a minimum 17000*4 = 68000 points.
If I create an index buffer and these can be a byte or short value, then won't I hit issues with values over (32,767 * 2)?

Hank

July 12, 2013 at 7:54 am

Yeap if I create a ShortBuffer and try to stick my indexing for each point I get some pretty interesting results. For the polygons there were a total of 250625 points, averaging about 15 points per polygon. Obviously this won't go into a java short which can only go to 32767 even if it were able to handle unsigned it would only go to 65535. What do you think could resolve this?

Admin

July 12, 2013 at 3:19 pm

You can go up to 65535 by casting the short as follows: (short) 65535; It will "just work" even if Java thinks it's -32768.

What if you just use a single vertex buffer for now without using an index buffer? That will reduce the complexity of the code. Later on you can figure out a way to LOD the data to get to or under 65536 unique vertices, or you can render using multiple index/vertex buffers.

Pingback: [Calling OpenGL from C on Android, Using the NDK | Learn OpenGL ES]

stety

July 16, 2013 at 6:59 pm

Great guide, but it is a bit complicated.
Can you please explain what does the following line TriangleBuffer.position (mColorOffset);
And why is it set to 3?
aTriangleBuffer.position (mPositionOffset) and here's to 0?

Perhaps you could have one bit of advice what exactly does GLES20.glEnableVertexAttribArray (mPositionHandle);?

Admin

July 16, 2013 at 10:53 pm

The vertex data is laid out like this: position 1, color 1, position 2, color 2, position 3, color 3, etc....
Since the position takes up 3 floats, we want to skip over those 3 floats when we start reading the colors. This is why we set the position to 3 before we read in the colors.

GLES20.glEnableVertexAttribArray (mPositionHandle) tells OpenGL to enable using the position data in the shaders. We read in mPositionHandle before with this line: mPositionHandle = GLES20.glGetAttribLocation(programHandle, "a_Position"); We have to enable the attribute array before using it in the shaders.

Let me know if you have any more questions! 🙂

thanos

June 15, 2016 at 9:47 am

Thanks for the great tutorial,

am I correct that mColorOffset and mPositionDataSize should be equal?
or is it that mPositionDataSize should always be >= mColorOffset ?

Thanks

---

stety

July 17, 2013 at 1:42 pm

Thank you.

---

stety

July 17, 2013 at 9:07 pm

I have another problem:
When you adjust the position of the first triangle:
final float[] triangle1VerticesData = {
-0.5f, 0.25f, 0.0f // zde
1.0f, 0.0f, 0.0f, 1.0f,

0.5f, 00.25f, 0.0f, / / here
0.0f, 0.0f, 1.0f, 1.0f,

0.0f, 1.0f, 0.0f, / / here
0.0f, 1.0f, 0.0f, 1.0f};
I have a problem with pivot. (triangle1 pivot)
Because pivot is outside the triangle. (not in the center)
So how can I set the pivot at your location?
I tried to do it using Matrix.rotateM(mModelMatrix, 0, angleInDegrees, 0.0f, 0.0f, 1.0f); howto but I did not.

Admin

July 19, 2013 at 2:47 pm

Hi stety,

Please post a question on stack overflow and add some example code there, then post the link here — I will check it out. 🙂

Hank

July 18, 2013 at 8:18 am

Hi Admin,
Have got the buffers working really well now, so thank you for your help on that. I have a new issue that has to do with zooming in and panning.
I see possibly three different ways to zoom.
– Projection – make the projection area smaller.
– View – eye gets closer to the model.
– Model – model is scaled larger.

Initially I tried using the first which worked well up until it gets to a certain level then the panning starts jumping. After testing I found that it was because the floating point value of the eye, could not cope with such a small shift in position. I keep my x and y eye values in doubles so it continues to calculate shifting positions, then when calling setLookAtM() I convert them to floats.

You've seen the image of the map and this is quite a large area. Panning starts to jump noticeably when it gets to a level of about 50m away. I would preferably like to get this to a sub-metre level.

Now I was going to try using one of the other two methods, probably scaling the model larger first. I think I might run into the same issues if I use the eye getting closer. If I scale the model larger then the scale of the eye shift shouldn't have to change… I think. How's my logic so far?

Tried to do a scale test but it didn't work. Not even sure if it should work. I just tried this test in my eye shift function, to see if it would at least scale.

```
public void setEye(double x, double y){
eyeX -= (x / screen_vs_map_horz_ratio);
lookX = eyeX;
eyeY += (y / screen_vs_map_vert_ratio);
lookY = eyeY;

Matrix.scaleM(mModelMatrix, 0, 0.5f, 0.5f, 1f);

Matrix.setLookAtM(mViewMatrix, 0, (float)eyeX, (float)eyeY, eyeZ, (float)lookX,
(float)lookY, lookZ, upX, upY, upZ);
}
```

Surely appreciate your advice on this.
Thanks Hank

Admin

July 19, 2013 at 2:55 pm

Hi Hank,

When working with floating point doing things like eyeX -= and eyeY += will cause the inaccuracies to build up over time, so you may want to work with absolute assignment instead of relative additions and subtractions.

When you said it didn't work, what happened? It didn't scale or you just don't see anything anymore? Hard to say from looking at this code snippet. It would all depend on where you're calling scaleM as well — maybe let's try another StackOverflow question with some example code? 😉

stety

July 18, 2013 at 12:57 pm

I already knew about. Pivot does not move.
When I want to rotate the triangle at the center of, I place it in the center triangle1VerticesData.
A move him through Matrix.translateM.
Or mile? it somehow move the pivot?

---

stety

July 19, 2013 at 8:23 pm

I'm sorry, I wrote another post a few lines below. Here it is:
I already knew about. Pivot does not move.
When I want to rotate the triangle at the center of, I place it in the center triangle1VerticesData.
A move him through Matrix.translateM.
Or mile? it somehow move the pivot?

---

stety

July 19, 2013 at 8:28 pm

I do not know why but when I answer always give a brand new post.
I'm sorry, I wrote another post a few lines below. Here it is:
I already knew about. Pivot does not move.
When I want to rotate the triangle at the center of, I place it in the center triangle1VerticesData.
A move him through Matrix.translateM.
Or mile? it somehow move the pivot?

Hank

July 20, 2013 at 6:33 am

Hi Admin, I opened a question on StackOverflow, http://stackoverflow.com/questions/17759346/zooming-in-map-with-opengl-es2-android.
I have just posted my code with original zoom by projection and pan by scaled

eye movements. My attempts so far have been unsuccessful, it didn't scale, it didn't disappear or anything, so I thought I would start with a clean slate for the question.

Pingback: Loading a PNG into Memory and Displaying It as a Texture with OpenGL ES 2, Using (Almost) the Same Code on iOS, Android, and Emscripten | Learn OpenGL ES

Hank

July 24, 2013 at 4:52 am

Hi Admin, any luck with looking over my scaling issue?

Admin

July 24, 2013 at 12:00 pm

Hi Hank,

Nothing jumps out to me as I look at the code. Something you can try is to temporarily disable the zoom code based on touch, and just zoom things yourself based on a variable you change each frame. For example, you could do like this:

scaleFactor = 1;

In draw:

scaleFactor *= 1.001;
…
draw

Just to see what happens. If this way is tricky (you said things move off to the side; that's because the scale is done from (0,0) and not from the center of the screen) you may want to go back to the way you were doing before that was working, and try to fix the jumping around.

Admin

July 24, 2013 at 12:06 pm

Also you can try scaling the view matrix instead, just for fun? I'm curious as to what will happen.

Lula

July 29, 2013 at 6:47 am

Hey , thanks. it is well articulated lesson. No errors But I cant see the triangle.

Can tell me why?

Thxs in advance 🙂

Admin

July 30, 2013 at 1:34 am

You can compare against the source here: https://github.com/learnopengles/Learn-OpenGLES-Tutorials Sometimes working backwards from working code can help to find the missing spot!

Hank

August 2, 2013 at 7:33 am

Hi Admin,

I have the zoom working by scaling the model and translating to the correct point. Panning is just by translating the model. However the same issue occurs with the panning but in the opposite way, the values for are too large for the floating point, at a close zoom.

As mentioned I keep all my values in doubles and cast them last second into floats for OpenGL.

What I was thinking was I am scaling and translating the model right now, I have stored the difference in accuracy like so.

```
public void pan(double x, double y){
modelXShift = modelXShift + x;
modelYShift = modelYShift – y;

diffXShift = modelXShift – (float)modelXShift;
diffYShift = modelYShift – (float)modelYShift;
}
```

Then use the difference to make slight adjustments in one of the following the eye, view or projection.
Either shifting the eye, translate the view or translate the projection.

However I don’t think this is quite as simple as its logic. The following, only once zoomed in, is jumping around. I tried by using setLookAtM() but it was a worse reaction, one side of the image would disappear. I have tried the following with the projection matrix as well with the same results.

```
Matrix.translateM(mModelMatrix, 0, (float)modelXShift, (float)modelYShift, 0f);
Matrix.scaleM(mModelMatrix, 0, (float)mScaleFactor, (float)mScaleFactor, 1.0f);
Matrix.translateM(mViewMatrix, 0, (float)diffXShift, (float)diffYShift, 0f);
```

My question is and hopefully explained it clearly enough, how does my logic seem? The fact that my image is jumping around maybe the amount I’m adjusting by or something I’m not adding to the equation.

Admin

August 3, 2013 at 11:35 pm

Hi Hank,

Off hand I don’t know a simple solution to this problem. There must be a solution since Google Maps etc… have found a solution. Just as a test, what happens if you blew up all of the source data (that is, before it touches any matrix transforms) by say, 1000x, truncated that to a reasonable area visible on the screen and then just scrolled that around as normal without any further zooming in on your part. Would that still jump around? I think the key will be to keep the floating point numbers that pass through your transforms from getting too large or too small. Unfortunately I can’t tell you exactly how to get there, that will take more research on your part. 😉 Perhaps http://gamedev.stackexchange.com/ would have some interesting ideas for you.

Pingback: Туториали за OpenGLES и Android | Georgi Neykov's Blog

Suflet

August 20, 2013 at 9:33 am

I have one big problem. Nothing gets displayed , neither in the emulator nor in my Galaxy S3 … why would this happen ? I don’t have any errors in Eclipse

Suflet

August 20, 2013 at 10:44 am

Later observation : I copied the source code from the LessonOneActivity LessonOneRenderer into Eclipse , modified the Manifest file to use open GL 2.0 , and still the app stops , reason : Unfortunately NAME has stopped. Why is this happening ? I am using API 17 so there shouldn’t be a problem … Hope you are still there to answer me!

Suflet

August 20, 2013 at 11:02 am

Log Cat output:

```
08-20 10:57:28.731: E/AndroidRuntime(784): FATAL EXCEPTION: GLThread 75
08-20 10:57:28.731: E/AndroidRuntime(784): java.lang.IllegalArgumentException: No config chosen
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$BaseConfigChooser.chooseConfig(GLSurfaceView.java:874)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$EglHelper.start(GLSurfaceView.java:1024)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$GLThread.guardedRun(GLSurfaceView.java:1401)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$GLThread.run(GLSurfaceView.java:1240)
```

Admin

August 20, 2013 at 1:24 pm

Hi Suflet,

Try adding this to your Activity before calling setRenderer:

```
glSurfaceView.setEGLConfigChooser(8 , 8, 8, 8, 16, 0);
```

That might fix the issue.

Suflet

August 20, 2013 at 1:27 pm

Nope , didn’t do it . Tried it a few hours ago … now i get this :

```
08-20 10:57:28.731: E/AndroidRuntime(784): FATAL EXCEPTION: GLThread 75
08-20 10:57:28.731: E/AndroidRuntime(784): java.lang.IllegalArgumentException: No config chosen
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$BaseConfigChooser.chooseConfig(GLSurfaceView.java:874)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$EglHelper.start(GLSurfaceView.java:1024)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$GLThread.guardedRun(GLSurfaceView.java:1401)
08-20 10:57:28.731: E/AndroidRuntime(784): at android.opengl.GLSurfaceView$GLThread.run(GLSurfaceView.java:1240)
```

Admin

August 20, 2013 at 1:30 pm

Does the emulator have “Use host GPU” enabled? What happens if you download the code samples from https://github.com/learnopengles/Learn-OpenGLES-Tutorials or http://pragprog.com/titles/kbogla/source_code and run them on the device (not the emulator) without changing any code?

There’s also an APK you can download here: https://market.android.com/details?id=com.learnopengles.android

Suflet

August 20, 2013 at 1:32 pm

Fixed that too , now this is happening :

```
08-20 13:29:44.341: E/AndroidRuntime(1355): FATAL EXCEPTION: GLThread 90
08-20 13:29:44.341: E/AndroidRuntime(1355): java.lang.RuntimeException: Error creating vertex shader.
```

```
08-20 13:29:44.341: E/AndroidRuntime(1355): at com.example.incercarigl.primaIncercareRenderer.onSurfaceCreated(primaIncercareRenderer.java:127)
08-20 13:29:44.341: E/AndroidRuntime(1355): at android.opengl.GLSurfaceView$GLThread.guardedRun(GLSurfaceView.java:1494)
08-20 13:29:44.341: E/AndroidRuntime(1355): at android.opengl.GLSurfaceView$GLThread.run(GLSurfaceView.java:1240)
```

it's hard writing 300 lines of code line by line to try and understand it and then to get errors , to much lines of code , i've spent the last 4 hours trying to figure out what happened because there is no syntax error , so algorithm error is the main problem…

Suflet

August 20, 2013 at 1:34 pm

I downloaded the APK works great , as for the example I copied the entire code from the manifest file , the main java file and the renderer file . Same error messages …

Admin

August 20, 2013 at 2:25 pm

Hi Suflet,

Instead of copying the code, what if you just import the project directly into Eclipse and run it as is, without copying? If the APK works great, then the code is fine — something is probably just getting lost when you copy code. From the above error, it looks like you might not have copied the vertex shader source.

I would recommend work with the APK code as a base, get it to run in Eclipse, and then modify that code step-by-step to suit your needs. Then it will be much easier to see when things go wrong, and what caused it.

Suflet

August 20, 2013 at 3:16 pm

What happens to the downloaded source codes : after installing the apk and try to run it , i get a static image with a red square on top and a blue one on the bottom , but no action , nothing moves. Just static display.

Admin

August 20, 2013 at 5:51 pm

That sounds like the air hockey example — that is what you are supposed to see from the early examples. 🙂 Everything sounds good.

Suflet

August 20, 2013 at 6:28 pm

Sorry , I was a bit looney today , lots o' stuff on my mind. Yeah I figured that AirHockey1 was supposed to do that , i didn't get the chance to try other apk's , and in the earlier comment i was talking about the apk on the market , the one with all the lessons in it.

Well , to make the long story short , after (i lost count how many) hours , i finally got this lesson working. I didn't figure out what the problem was , maybe next time it will pop in my mind. But for now all is good. Thanks for the quick replies. Didn't think you were still around after 2 years from posting this tutorial ! Now up to the next Lesson !!!

Vasya

August 31, 2013 at 7:35 am

good tutorial. there is no fragment shader compile code, but anyway its good, got my rotated triagle, K.K.107

September 9, 2013 at 7:17 am

We are an Android App developer at Japan.
Here, only few material related to Open GL ES 2 exist. On Android environment, in local language we now only one book. Despite in english "Learn Open GL ES 2" is the best one. It is organized according to japanese learning style. We had translated it to native language for internal (office) use. Many people outside the office interested on this material. We intend to put it in our Web Site (107.co.jp). This site is free and open for public access. Certainly, We will mention all the necessary credit information about the source, but no else more! No payment of any kind will be paid to you or your company. We appreciate your comment about this possibility.

Congratulation for written premium quality tutorial. We hope continuous success in all future publications.

Sincerely, K.K.107
obana@107.co.jp

Admin

September 12, 2013 at 4:06 pm

Hello,

If you are referring to this website, then the content here is licensed CC-BY-SA (http://creativecommons.org/licenses/by-sa/3.0/), so you are completely free to translate and make the material available on your own website! You only have to give me attribution and release your translated material also under the CC-BY-SA.

If you are referring to the book, that is under copyright by The Pragmatic Bookshelf and the best way to inquire about translation rights is to contact them here: http://pragprog.com/contact

Thanks!

pbs6

September 11, 2013 at 9:51 am

Great tutorial – really helps – going to take a look at your book and probably buy it soon!

I did have a problem, however, which was the same as Suflet's. I kept getting the "Error creating vertex shader" problem. The reason for this is that when I copied the code from the website to my Eclipse project file, there were some weird Unicode characters (which weren't visible to the eye !) which got copied across as well.

This didn't hinder compilation, since the problem lie within the strings, which of course Java wouldn't care about. However, clearly the shader compiler picked up the problem, which gave the error.

There are two solutions to this: (1) completely delete the "final String vertexShader = ' … '" line and type it by hand, or (2) copy it directly from the downloadable project files.

Either way, you don't end up with those unwanted and invisible Unicode characters!

Great tutorial – the best on the net.

Admin

September 12, 2013 at 4:12 pm

This happened when copying the code from this blog post? Ouch, that's not good! Thanks for letting me know about this; maybe the better solution is to switch to GitHub Gists or something similar.

Eric

September 14, 2013 at 12:58 am

Sorry to bother you, but I had a strange issue. I copied the code line for line so that I could understand and get a feel for what's going on, and then installed the package on my phone and had the strangest behavior.

Once the app loaded, I had a black screen. Nothing happened. No animation. However, once I hit the home button, I could, for a second or two, see the three triangles spinning before my phone returned to the home screen.

Very strange. Is there any possible explanation for such behavior?

Admin

September 16, 2013 at 2:27 pm

Hi Eric,

I'm just wondering if that happens with the downloadable APK or with the code from GitHub?

pbs6

September 14, 2013 at 4:58 pm

Yes, I was select/copy/pasting across from the blog post. I should have picked up the error earlier really because the problem wasn't restricted to the text inside the strings, but all the code in the copy block. I suspect its probably a tabbing Unicode that's to blame!

raticide

September 17, 2013 at 7:48 am

Hello Kevin,

That’s really a great job you’ve done here. I’m a beginner with openGL and this article really worth the read.

I have a question though. I noticed that when rotating the screen the full process of creation is performed : creation of GLSurfaceView, GLRenderer as well as the creation of the objects (Triangle). So basicaly, it’s like it restarts the application nearly from scratch.
This is easy to observe by adding traces in onCreate, onPause, OnResume of the Activity and adding other traces in GLRenderer (onSurfaceChanged, onSurfaceCreated).

My code does create a pretty complex object, so it takes time (20 seconds) to create it. So if the user rotates the screen I don’t want my object to be recreated from scratch as it means a lot of time which shall not be necessary as the object already exists.

So far, I’ve :
. declared my object as static to not loose it. Indeed, that works but when I rotate the screen i just does not work at all.(note that I’ve added the piece of code that prevents from recreating the object if it is not null).
I also tried to create the Renderer as static but no luck.

But it does not work at all.

I was wondering what would be the modification to bring to your OpenGLES20Complete example code so that there is no recreation of the object when rotating the screen.

Well, that would really help me a lot.

Admin

September 17, 2013 at 3:34 pm

The easiest way I can think of is to just add something like this to your AndroidManifest.xml:

Your activity can still be restarted so you should still handle anything you need to persist in onPause() / onResume(), it just won’t restart in as many places.

Admin

September 17, 2013 at 3:37 pm

Whoops, WordPress filtered out the tags. That “android:configChanges” should be added to your activity tag in AndroidManifest.xml.

raticide

September 18, 2013 at 6:18 am

Ooooh man, what a lovely solution. Thanks so much. I was spending so much time on dealing with the EGL context being lost. There’s a new function with GLSurfaceView (GLSurfaceView.setPreserveEGLContextOnPause(booelan) to prevent it from being destroyed but that assumes that your GPU is capable of dealing with multiple context which does not seem to be the case for mine.

A huge THANK YOU Kevin.

Admin

September 18, 2013 at 3:47 pm

Glad that adding that stuff to the manifest helped out. 🙂 When you go on to more complex apps, I think GLSurfaceView.setPreserveEGLContextOnPause can still be useful, as it will help you out when onPause() and onResume() are called, i.e. if the user receives a call or presses the home key. In those cases the activity is not destroyed, but you could still lose the GL context so it’s still worth keeping that call around.

raticide

September 19, 2013 at 12:28 pm

Yes, that worked on your example. I then tried to implement in my own code but it's a bit more complex because I have to change my layout depending on the orientation of the screen (portrait or landscape). I noticed that as soon as I load an other layout the EGLcontext is lost. That's so annoying.
I will still investigate a bit.

Admin

September 20, 2013 at 3:05 pm

It's a bit more complicated but you can hide/show views instead of reloading the layout… or you can get fancy and reload the layout, but when you do so in onConfigurationChanged(), you can replace the GLSurfaceView in the new layout with the one you already have instantiated (viewGroup.removeView, viewGroup.addView). In any case, just some ideas. If it's possible a layout that works in both portrait and landscape will be easier to deal with.

raticide

September 20, 2013 at 4:23 pm

Thanks for the answer. I was about to write that I had decided to hide/show views finaly This solution works fine even if I don't like it a lot.

I've performed many tests with changing the layout in onConfigurationChange (XML file). I also replaced dynamically the parent of my GLSurfaceView view as suggested (remove/add) but then … I noticed that the simple fact of replacing the layout with SetContent just made the EGL context being lost.
This is weird.

Admin

September 20, 2013 at 5:14 pm

I believe that is because changing the layout removes the old GLSurfaceView from the hierarchy, which causes it to destroy its GL thread which destroys the EGL context. To keep the same context, you have to keep the same GLSurfaceView.

raticide

September 21, 2013 at 11:09 am

Hi Kevin. What you suggest is pretty convincing. I don't have my dev environment with me during the weekend (family time) but I'll give it a go on monday and let you know. I can't remember if I was loading the new layout before or after removing the GLSurfaceView's parent. I'm just sure that I was keeping the same GLSurfaceView.
What I will try in sequence : 1) remove GLSurfaceView's parent. 2) Load the new layout (setcontent) 3) Add the view to its new parent.

Rahul

September 21, 2013 at 3:00 pm

Nice Tutorial But When i run my application in emulator error Unfortunately, the Android emulator does not support OpenGL ES 2 Please Help Me How to run OpenGL Application my emulator

Admin

September 23, 2013 at 8:39 pm

Please let me know if this post helps:
http://www.learnopengles.com/android-emulator-now-supports-native-opengl-es2-0/

raticide

September 23, 2013 at 6:59 am

Just performed the test as follow in onConfigurationChanged : 1) removeLSurfaceView's parent 2) SetContent(new xml layout) 3) find the target

LinearLayout per ID 4) add GLSurfaceView to the target LinearLayout Unfortunately, the context is lost. I confirm that GLSurfaceView was kept the same. If you wish I can post the little modified source somewhere. But that's what I discovered. It seems like SetContent(layout) destroys every existing EGL context.

Admin

September 23, 2013 at 8:38 pm

Indeed, thinking back on the implementation I remember now that the GL context gets destroyed as soon as you remove the GLSurfaceView from the view hierarchy, even if you use the same GLSurfaceView later. So, I think the only other options may be to use a shared EGL context (not sure how well these work on Android), or to use one XML for both portrait and landscape. You might be able to do that by toggling visibility between GONE and VISIBLE for your other UI elements.

dotaness

November 3, 2013 at 5:12 am

Hey,

Can you explain why when i change the eyeZ to 1.0f the triangle is not showing up?

Like this

// Position the eye behind the origin.
final float eyeX = 0.0f;
final float eyeY = 0.0f;
final float eyeZ = 1.0f;

Whats wrong with 1.0f?

Admin

November 4, 2013 at 5:29 pm

I'm guessing that this is because the projection matrix is setting the near plane to 1.0f, so the stuff that gets drawn ends up 1.0f distance away from the eye and gets clipped by the near plane. Try changing the numbers for the projection matrix and you'll probably see a difference.

skadush

November 4, 2013 at 9:59 am

This is much more advance than of the book. The method you use is very very different.

BTW can you tell me why float[16]?

Admin

November 4, 2013 at 5:51 pm

Hi Skadush,

The float[16] is a raw buffer to hold the data for a 4×4 matrix, for example, the projection matrix. This post is much older than the book, so it is more unrefined. 😉

skadush

November 5, 2013 at 10:03 am

So ah…… I should follow the book? Because its much more advance? and I havent seen all the content of the book yet but does the book contains all the tutorial in here as well with better coding atleast?

Admin

November 5, 2013 at 6:17 pm

I definitely recommend to follow the book as it goes a lot more in-depth, explains things in more detail, and assumes less knowledge on the part of the reader. Once you've read the book you'll have more of a knowledge base to continue your education on this site and on other sites.

Jacky

November 14, 2013 at 1:26 am

Hello,

I am following your tutorial and it begin to make more sense when i read it a couple times. However, I try to make the triangle move to the right but it does not work. Can you please tell me how to do that using this existing code skeleton?

Thank you!

Admin

November 19, 2013 at 3:00 pm

Hi Jacky, for the first triangle that gets drawn like this:

```
// Draw the triangle facing straight on.
Matrix.setIdentityM(mModelMatrix, 0);
Matrix.rotateM(mModelMatrix, 0, angleInDegrees, 0.0f, 0.0f, 1.0f);
drawTriangle(mTriangle1Vertices);
```

You could move it to the right by translating it, like this:

```
Matrix.setIdentityM(mModelMatrix, 0);
Matrix.translateM(mModelMatrix, 0, 1.0f, 0.0f, 0.0f);
```

Matrix.rotateM(mModelMatrix, 0, angleInDegrees, 0.0f, 0.0f, 1.0f);
drawTriangle(mTriangle1Vertices);

You can read up more about transformation matrices here: http://www.mathplanet.com/education/geometry/transformations/transformation-using-matrices and https://www.khanacademy.org/math/linear-algebra/matrix_transformations

gamedevNoobie

November 14, 2013 at 2:57 am

WAAAIT!.. XD

What is the difference of attribute and uniform?? Cant I just use uniform? Is uniform only for constants?

Admin

November 19, 2013 at 2:13 pm

An attribute is something that you would set for each vertex, while a uniform is something that can apply to many vertices and that you're only able to change between draw calls. So, while a uniform isn't only for constants (you can still change them between draw calls), it can make sense to use them in that way.

Jair Humberto

November 17, 2013 at 1:01 am

This tutorial is a work of art, and is very, very helpful, thank you a lot!

But I still don't understand some things like: why to multiply matrices to transform a object. In my unsuitable concept, to move an object, I should sum a number to it's position. So, I not understand, the "because" of the use of matrices for this and why multiply it instead.

Another thing I don’t understand is why fragment shader is a programmable step since it does a so single thing.

Well, again, thank you for this nice tutorial!

🙂

Admin

November 19, 2013 at 2:33 pm

When it comes to translation it’s actually kind of a hack. 😉 By multiplying a matrix with a translation matrix, what actually happens is that the translation amount is added to the X, Y, and Z parts of the matrix. You can learn more here:
http://www.arcsynthesis.org/gltut/Positioning/Tut06%20Translation.html

However, normally we’re doing a lot more than just translation. We’re also projecting to the screen, and we might be rotating and doing other transformations as well. That’s why transformation code will usually premultiply these matrices together, and just pass the final constructed matrix into the shaders.

In OpenGL ES 1.0, the fragment shader wasn’t programmable. This made it easier to do simple renderings like the one we do in the lesson. On the other hand, it was a lot more complicated to do more advanced effects. In OpenGL ES 2.0, having a programmable fragment shader adds more initial overhead to simple scenes like this, but the flexibility really pays off later on.

Jair Humberto

November 17, 2013 at 1:38 am

Another thing I didn’t understand is why set colors on vertex? I always thing about colors on pixels not on vertex

Admin

November 19, 2013 at 2:29 pm

Hi Jair, indeed colours are set on pixels in the fragment shader. By setting a colour on each vertex, the fragment shader can interpolate the colours between the 3 neighbouring vertices for the current fragment. In a real app or game, it might be combining the colours from several different textures and lighting effects before applying to each fragment.

Chamika

December 20, 2013 at 6:03 am

Hi
First thank you very much for the great tutorial.

I need a little help with orthogonal projection. This is what I do in the onDrawFrame method

```
onDrawFrame() {

setPerspectiveProjection(); // code took from onSurfaceChanged method

// trangle drawing code in the original example
…
// set matrix to orthogonal
Matrix.orthoM(mProjectionMatrix, 0, 0, screenWidth, 0, screenHeight, 0, 2);

// Draw the triangle facing straight on.
Matrix.setIdentityM(mModelMatrix, 0);
drawTriangle(mTriangle1Vertices);

}
```

But I cannot see the trangle drawn in 2D

// reset the perspective project

Sergey

January 20, 2014 at 2:59 pm

Hi! I have a question. Why doesn't use GLES20.glDisableVertexAttribArray function after draw?

Julian Motture

February 7, 2014 at 3:43 pm

Hi, really good lesson. I followed the link to download the source code but I can't see it on the website. I'm prob being blind but can you please tell me which link on repository will give me the source code? Many thanks.

Sam

February 28, 2014 at 2:54 am

I do not mean to be rude, and please forgive my noobieness, but i also was caught on this and without this comment, it would have rendered my experience on this site pretty much useless.

Sorry if i already should have known but it said for beginners, and im starting here from scratch into openGL es.

S

April 1, 2014 at 1:32 pm

Very helpful website.

I am a newbie and please forgive me for asking this simplistic question, it's stuck in my head.
The triangles have been defined to be in the xy plane with z = 0.0

And the near and far clipping planes are z = 1.0 and z = 10.0 respectively. So the triangles lie outside the view frustum, right? I do not understand how they are visible. What makes me even more curious is that the triangles vanish when I set their z coordinates to 1.0, which should be included in the view frustum. I searched and searched on Google but I could not find the answer or the correct keywords to use. Somebody please answer, or this curiosity will kill me.

Admin

April 1, 2014 at 2:49 pm

Hi S,

The key here is that the triangle coordinates get multiplied by the view matrix before the projection matrix, and the view matrix moves the triangles into the right range. It might be easier to understand it if you debug your program, copy the actual values into a calculator, like this one: http://www.calcul.com/show/calculator/matrix-multiplication

Enter the values like this

0 4 8 12
1 5 9 13
2 6 10 14
3 7 11 15

With the number corresponding to that position in the array. Then you can multiply the matrices, or matrix * vector, just as you see it in the code, and also compare the results to what you see in the code.

ramsey

April 7, 2014 at 11:09 am

Hey 🙂

It doesn't work on my samsung galaxy s3 :/ it only shows a black screen. Do you what the problem is?

Admin

April 7, 2014 at 4:08 pm

Does it happen with the downloadable app? You could try it here: https://market.android.com/details?id=com.learnopengles.android

Ogen

June 26, 2014 at 4:42 am

Hello,

Firstly, thanks for the android it's very helpful. I just have a question.

In the code, your triangle data is x, y, z then r, g, b, a and it repeats like that. The way I've set up my code has all of the coordinates and then the colors, also, it is in 2D. So my triangle data is something like {x, y, x, y, x, y, r, g, b, a, r, g, b, a, r, g, b, a}

Because of this change, I don't know how to implement your drawTriangle method, the variables for the offsets and such have to change but I don't know how to set it up.

Any help is appreciated.

Admin

June 26, 2014 at 1:57 pm

Hi Ogen,

You can do this too, it's just a matter of changing the calls to glVertexAttribPointer. For example, you can do it like this:

```
/** Offset of the position data. */
private final int mPositionOffset = 0;

/** Offset of the color data. */
private final int mColorOffset = 3; // FIXME: Instead of 3, this should be the start position of your colour data.

// Pass in the position information
aTriangleBuffer.position(mPositionOffset);
GLES20.glVertexAttribPointer(mPositionHandle, mPositionDataSize,
GLES20.GL_FLOAT, false,
0, aTriangleBuffer); // NOTE: A stride of 0 since the data is packed.

GLES20.glEnableVertexAttribArray(mPositionHandle);

// Pass in the color information
aTriangleBuffer.position(mColorOffset);
GLES20.glVertexAttribPointer(mColorHandle, mColorDataSize,
GLES20.GL_FLOAT, false,
0, aTriangleBuffer); // NOTE: A stride of 0 since the data is packed.

GLES20.glEnableVertexAttribArray(mColorHandle);
```

I didn’t compile and try this out, but I believe that this should work. 🙂

Ogen

June 26, 2014 at 2:52 pm

Thank you SO much, that worked like a charm!

Ogen

June 26, 2014 at 3:52 pm

Wait a minute, it’s not T.T I though it was…

What do you set mPositionDataSize and mColorDataSize to? I set them to 2 and 4 respectively. Is that right?

Admin

June 28, 2014 at 3:23 pm

Yeah that would make sense if it was X, Y, and R, G, B, A

Ogen

June 26, 2014 at 4:18 pm

Here is the code for my draw method.

```
protected void draw(GL10 gl) {
// Add program to OpenGL ES environment
GLES20.glUseProgram(mProgram);

// get handles
mPositionHandle = GLES20.glGetAttribLocation(mProgram, “vPosition”);
mColorHandle = GLES20.glGetUniformLocation(mProgram, “vColor”);
InGameActivity.checkGlError(“glGetAttribLocation”);
InGameActivity.checkGlError(“glGetUniformLocation”);

vertexBuffer.position(0);
GLES20.glVertexAttribPointer(mPositionHandle, 2, GLES20.GL_FLOAT, false,
0, vertexBuffer); // NOTE: A stride of 0 since the data is packed.

GLES20.glEnableVertexAttribArray(mPositionHandle);

// Pass in the color information
vertexBuffer.position(6);
GLES20.glVertexAttribPointer(mColorHandle, 4, GLES20.GL_FLOAT, false,
0, vertexBuffer); // NOTE: A stride of 0 since the data is packed.

GLES20.glEnableVertexAttribArray(mColorHandle);
```

GLES20.glDrawArrays(GLES20.GL_TRIANGLES, 0, 3);
}

Admin

June 28, 2014 at 3:22 pm

Hi Ogen,

I don't see anything wrong in particular with that code — would you be able to paste a self-contained sample to a GitHub Gist? That way I could compile it on my side and see.

Ogen

June 29, 2014 at 2:35 am

Okay I made one, here it is:
https://gist.github.com/ogenodisho/58fb2f32ce2115dd9195

Thanks for taking the time 🙂

Admin

July 1, 2014 at 8:13 pm

Hi Ogen,

Np, I will make some time to check it out tomorrow.

Admin

July 2, 2014 at 4:01 pm

Hi Ogen,

I was able to get it working. I'll give you some hints to try and hit the solution on your own (as you'll learn much more that way ;):

1) To read in array data in the vertex shader, declare it as an attribute, and bind it with glVertexAttribPointer. Uniforms are only for "constant" data that is shared across many vertices.
2) To access color from the fragment shader, output it as a varying from the vertex shader and read it as a varying in the fragment shader.

So the main changes you need to make are with the shaders. If you're still stuck after a couple of days, don't hesitate to get back in touch and I'll send over the solution. At the risk of being overly self-promoting, I also wrote a book that is just for beginners, since I was once in the same shoes as you: OpenGL ES 2 for Android: A Quick-Start Guide. Hope this helps out!

John Mark Isaac Madison

June 23, 2015 at 8:20 pm

Hay there! I've been working on OpenGL for 3 days now with no luck.
I've done 4 tutorials so far and watched lectures.
Feel like I am getting closer.
I need some contacts who are also serious about OpenGL.
From your git-hub post, I am taking you as serious.
My email is J4M4I5M7 AT hotmail.com
-John Mark

Jax

July 1, 2014 at 2:58 am

Thank you so much for looking intro to OPENGL 2.0,

Ogen

July 3, 2014 at 3:53 am

I changed my shaders to this and it’s still not working:

```
private final String vertexShaderCode =
// This matrix member variable provides a hook to manipulate
// the coordinates of the objects that use this vertex shader
“uniform mat4 uMVPMatrix;” +
“attribute vec4 a_Position;” +
“attribute vec4 a_Color;” +
“varying vec4 v_Color;” +
“void main() {” +
” v_Color = a_Color;” +
// the matrix must be included as a modifier of gl_Position
” gl_Position = uMVPMatrix * a_Position;” +
“}”;

private final String fragmentShaderCode =
“precision mediump float;” +
“varying vec4 v_Color;” +
“void main() {” +
” gl_FragColor = v_Color;” +
“}”;
```

From my point of view, it looks like it’s right. I have outputted the color as a varying from the vertex shader and I’ve read it in as a varying from the fragment shader.

Ogen

July 3, 2014 at 2:30 pm

Nevermind I just made a silly mistake, I got it working. Thanks for not doing it all for me, I did actually learn more!

Admin

July 3, 2014 at 2:59 pm

Nice 🙂 Glad that it worked.

Ogen

July 5, 2014 at 5:46 pm

Hello,

Sorry to bother you again but I'm ready to publish the opengl es 2.0 app I've been working on (thanks to your help) but I just have one last problem.

I have an activity where some triangles are being drawn etc and at one point, I want to quit the activity and then later, I want to start it again fresh. I am calling the finish() method and it exits the activity but then when I start it again, some of my triangles are still there eventhough I called finish(). I get the feeling that it must be the mGLView and the renderer and the GLSurfaceView objects that didn't get deleted or something.

How can I just get rid the entire activity? I have tried using System.exit(0) but I read that it is bad practice and it was causing a lot of buggy behavior so I decided to just try to finish the activity normally but I've been really struggling.

Any help with this final hurdle would be greatly appreciated.
Also, I don't want you to think I'm handing this problem on to you, I am actively trying to find a solution but nothing is working.

Admin

July 5, 2014 at 7:24 pm

Hi Ogen,

Is this when pressing the home button or the back button? Normally you should never need to call finish() manually, and especially not System.exit(0). Is it possible that you're using static variables to hold onto stuff and thus that stuff doesn't get cleared? I would avoid statics except when used as constant data, or as some sort of in-memory cache.

Admin

July 5, 2014 at 7:27 pm

I think I found the line in your code that’s doing it, actually. Are there any static variables that you keep concatenating to, over and over? I’m not sure cause I didn’t trace it, but check it out. 😉

Ogen

July 6, 2014 at 1:50 am

Hi,

All the of triangle data is not static, but my main activity does have a lot of static variables. I have a bunch of Timers, Tasks, ArrayList that are all static.

I ‘didn’t know static variables stick around. I’ll try to make them all non-static and see if it works.

Thanks for the reply

Ogen

July 6, 2014 at 1:51 am

Also, its not when pressing the home button OR the back button. When the game finishes, an alert dialog comes out that has a button called “Play Again” and “Back to main menu”. I want to kill the activity when they click “Back to main menu”

Ogen

July 6, 2014 at 1:57 am

The reason I have so many static variables is because of the openGL stuff.

I have this class: public static class MyGLRenderer implements GLSurfaceView.Renderer

It’s static so everything inside it needs to be static otherwise I get an error. Am I allowed to remove the static from MyGLRenderer? Or does it need to be static.

Ogen

July 6, 2014 at 2:32 am

Hello,

I got it working, I just removed everything that is static and now my finish() and recreate() (to restart the game) are working.

Thanks.

Ogen

July 6, 2014 at 3:21 am

You don’t mind if I put you in the acknowledgements for my app right. I’d still be stuck on the shaders if it weren’t for you 🙂

Ogen

July 6, 2014 at 1:03 pm

Here is a link to my app.

Thanks for all the help.

https://play.google.com/store/apps/details?id=ogen.tri.morph&hl=en

---

Admin 👤

July 7, 2014 at 1:13 pm

Hi Ogen,

This looks pretty neat! I'll link to it in my next roundup and help to promote.

I had a few observations I wanted to share with you:

1) It wasn't obvious to me at first that I had to move the corners of the triangle. It might be helpful to have a short tutorial the first time you start the game just showing how to move the corners to dodge the enemy triangles.

2) At first I wasn't able to play because my Google Play Services was out of date. If it isn't mandatory (it seems to be for the leaderboard?) it might be worth making the UI flow for this optional.

3) There's a performance issue right now where the framerate starts to slow down after 15 seconds or so. I'm not sure why this is. I would trace over the onDrawFrame and what it's doing in there to check nothing in there is slowing it down too much.

4) I saw that you disabled the back button, probably to avoid restarting the game when the user returns to the app. One way you can have your cake and eat it too is by using onSaveInstanceState / onCreate to save and restore the game state.

5) The "Rate" option wasn't working for me.

This is pretty neat, and it probably required you to solve several different problems, like detecting and reacting to the touch events, calculating triangle overlap, and so on.

Ogen

July 8, 2014 at 5:01 am

Thanks for the advice, I'll try to fix all the stuff you said.

The main reason I disabled the back button was because I felt that people could abuse the system by just pausing it when it gets to a hard position, then restarting it again. I wanted one game to happen in one hit, similar to flappy bird where you can't go back.

As for the rate button, It works for me and I just copied code from a stackoverflow question that starts up the playstore so I don't know how to debug that because I never get any errors.

The main thing that bothers me is the quality of the game how it degrades over time like you said. This is my onDrawFrame().

```
public void onDrawFrame(GL10 gl) {
// Redraw background color
GLES20.glClear(GLES20.GL_COLOR_BUFFER_BIT);

mTriangle.draw();

if (!stopDrawing) {
Triangle current;

for (int i = 0; i < numActiveTriangles; i++) {
current = fallingTrianglesArray[i];

if (current != null) {
calculateHits(current);

if (!current.fell) {
current.draw();
}
if (current.isGarrisonGoldTriangle()) {
mTriangle.draw();
}
}
}
} else {
triangleThatHitYou.draw();
}
}
```

And this is after I optimized it a lot so it is probably still doing too much work or something. The calculateHits method is pretty long. And it does a lot of work which is probably the cause.

Admin

July 8, 2014 at 12:32 pm

Hi Ogen,

Is calculateHits ignoring the fallen triangles? It might be worth keeping all the triangle data in one vertex buffer, and simply appending to that data when you add new triangles, and swapping them out when you remove. You can preallocate a buffer big enough for say, 200 triangles, or whatever is the upper limit of triangles you expect to have on the screen at the same time. This reduces your draw calls to 1 for all of the triangles.

To get an idea of how this could be implemented, check out the code for the Particles chapter here: http://pragprog.com/titles/kbogla/source_code

Ogen

July 9, 2014 at 5:45 am

Wow I never thought of that. That’s a really good idea and it’ll make the app run much more smoothly.

Currently, every triangle object has a boolean called hasFallen. And the draw method is only drawing triangles from an arraylist that contains non-fallen triangles. So I am only taking the triangles on the screen into account.

I'm going to try to put all of the coordinates of the on screen triangle into one buffer and draw it.

Thanks for the tip 🙂

Kervin

August 6, 2014 at 2:43 pm

Hi, I'm new to OpenGL and Eclipse, but want to learn fast.
I downloaded the sample project but when run and press any lesson button it says "Unfortuantely, Learn OpenGL ES Tutorials has stopped.". I don't know what did I wrong, I just open the project with Eclipse and run. Any ideas?

Admin

September 15, 2014 at 1:26 pm

Hi Kervin,

If this happens only in the emulator you could try the tips on this page: http://www.learnopengles.com/android-emulator-now-supports-native-opengl-es2-0/. What is the error you see in your logcat?

Hami

August 15, 2014 at 12:53 am

Thnx for the tutorial.
How can I change triangle to a square or a star?

Admin

September 15, 2014 at 1:28 pm

Hi Hami,

The simplest way would be by drawing additional triangles. For example, you can draw a square out of 2 triangles, and a star out of 5 triangles that slightly overlap. I would recommend to experiment and try it! 🙂

Andrew

August 25, 2014 at 11:57 pm

Thank you for these helpful tutorials! Really great, especially when the Android OpenGL reference lists all the methods but not what they do, and you have to hope that khronos' c++ documentation is sufficient for the java version. o_o.

Might pay to update the page – some emulators can support OpenGL ES 2.0 now.

Also, would you pretty please be able to draw a diagram of the app lifecycle, and when the OpenGL context is destroyed and recreated? Every time I lock my phone and unlock it, the VBOs are no longer valid. It would be helpful to have an image to show exactly when to expect this so I can reload all GL content.

Cheers,
Andrew

Admin

September 15, 2014 at 1:22 pm

Hi Andrew, normally everything will get destroyed in GLSurfaceView's onPause() method, but you can avoid this by calling GLSurfaceView.setPreserveEGLContextOnPause(boolean preserveOnPause). This doesn't work on all devices but should work on most newer ones.

Andrew

September 16, 2014 at 9:52 pm

Is there any easy way to check if the context was preserved? At the moment, all I can think of is creating something trivial, like a 2×2 texture, and checking if it can be bound in onResume().

Thanks again for this website. I can't say enough about how much it helped me to understand OpenGL.

Admin

September 17, 2014 at 4:42 pm

Hi Andrew,

If your onSurfaceCreated() gets called again, then you know you lost the context. 😉 In addition to the preserve EGL call, if you're only using GLSurfaceView and don't have a complicated layout with other views or fragments, then you can ask Android not to restart your activity on rotation with this line in your AndroidManifest.xml:

android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|uiMode|screenSize|smallestScreenSize"

This is what AdMob is using, for the same reason of not wanting to restart the activity. You can remove any of those if you want Android to restart your activity when they change.

Thank you for the kind compliments, glad that the site helped. 🙂

Yousif

November 4, 2014 at 3:33 pm

Great tutorial.
One question. The opengl.org page on fragment shaders has no mention to gl_FragColor… how come?

https://www.opengl.org/wiki/Fragment_Shader

Admin

November 12, 2014 at 6:36 pm

For OpenGL ES stuff you can use the resources here directly for 2.0 and 3.0/3.1:

https://www.khronos.org/files/opengles20-reference-card.pdf
https://www.khronos.org/files/opengles31-quick-reference-card.pdf
https://www.khronos.org/opengles/sdk/docs/man31/index.php

diego

November 19, 2014 at 6:31 am

Why we need allocatedirect() for attributes and isn't neccesary for uniforms?

Admin

January 5, 2015 at 4:08 pm

Hi Diego,

A uniform is usually a small amount of data represented by 3 or 4 floats, or up to 16 in the case of a matrix, so it can be sent directly to OpenGL through one of the method calls. An attribute, on the other hand, represents an array which can be hundreds of thousands of elements or larger, so for this reason, a pointer to the data is passed to OpenGL rather than the data itself. AllocateDirect allows OpenGL to safely access this data by making sure that the VM won't move the data or perform other shenanigans that the native code would not expect.

Will

November 28, 2014 at 10:04 am

Very nice Tutorial
Easy to understand if you have basic knowledge of rendering pipeline.

royanon

December 10, 2014 at 11:19 am

Best comment I've ever read in this website. Cheer!

link96

January 31, 2015 at 2:57 pm

Raymond, take a look on this (this is how I get the surfaceview)

GLSurfaceView glmapsurface = (GLSurfaceView) findViewById (R.id.map_gl);
glmapsurface.setEGLContextClientVersion(2); // <- THIS IS THE IMPORTANT LINE
glmapsurface.setZOrderMediaOverlay(true);
glmapsurface.setRenderer(glmapview.mRenderer);

Also don't forget to put ''
in the manifest.

Pingback: এন্ড্রয়েড গেম ডেভেলপমেন্ট টিউটোরিয়াল – AndEngine + Android Studio – জাকিরের টেক ডায়েরী – জাকির হোসাইন

Danny

March 2, 2015 at 2:23 am

I'm learning so much from your book.

Thank you.

Pingback: এন্ড্রয়েড গেম ডেভেলপমেন্ট টিউটোরিয়াল – AndEngine + Android Studio | Techtunes | টেকটিউনস

doug

April 13, 2015 at 11:57 pm

This doesn’t work on current Android emulators

doug

April 14, 2015 at 12:00 am

Nevermind, this is an emulator problem

kevin chen

May 17, 2015 at 6:44 am

Did OpenGL es working on Android Studio?

Danny S

June 23, 2015 at 1:55 pm

The only issue I’ve had coding OGL in Android Studio is that the plugin places some kind of comment in the first line of the glsl. I get an error when I compile, unless I delete that line. Other than that, I haven’t seen any issues.

I test to a connected device. I don’t use the emulator, so I can’t comment on that.

Pingback: এন্ড্রয়েড গেম ডেভেলপমেন্ট টিউটোরিয়াল – AndEngine + Android Studio – টেক ম্যাগজি

John Mark Isaac Madison

July 2, 2015 at 6:25 pm

Is:
// Bind attributes
GLES20.glBindAttribLocation(programHandle, 0, "a_Position");
GLES20.glBindAttribLocation(programHandle, 1, "a_Color");

Combinded with:
mPositionHandle = GLES20.glGetAttribLocation(programHandle, "a_Position");
mColorHandle = GLES20.glGetAttribLocation(programHandle, "a_Color");

A redundancy?
I Thought you used one or the other.
glBindAttribLocation or glGetAttribLocation

Terry

July 19, 2015 at 1:54 am

I used Android Studio. Created a blank activity, and after the project was built, created another class file. Deleted the contents of both files except for the first line. Then went to GitHub and copied the contents of both LessonOneActivity.java and LessonOneRenderer.java, except the first line of each, and pasted them into may files of the same names. Compiled and ran on an emulator, first try.

Thank you for a great tutorial

Randall Eskildsen

July 29, 2015 at 6:11 pm

Ok so how do i create a 2d and 3d air hockey game using Android Studio and *NOT* use Eclipse since it is now unsupported.

Migrating from Eclipse to Android Studio is very confusing to me.

Admin

September 29, 2015 at 7:17 pm

Codewise it's basically the same but IDE-wise some steps will be different. If you're building a new project from scratch, Android Studio is easy to get up and running with.

http://www.raywenderlich.com/78574/android-tutorial-for-beginners-part-1
http://code.tutsplus.com/tutorials/getting-started-with-android-studio–mobile-22958

Reza

September 8, 2015 at 6:21 am

Thanks for the great tutorial. I have used desktop OpenGL a few years ago. What comes to my mind is this OpenGL ES is very low level, for a very simple and basic task, many things MUST be implemented manually which personally I do not like it. I prefer a little more high level interface (such min3D) to do my things.
Anyway, for those who will refer to this page and read your helpful tutorial, the following link may be helpful too:
http://androidblog.reindustries.com/opengl-es-2-0-2d-shaders-series-001-basic-shaders/

Utkarsh

October 10, 2015 at 4:23 am

Thanks a lot for this amazing tutorial. I have just 1 question. I tried multiplying the mViewMatrix with mProjectionMatrix and then multiplying the result with mRotationMatrix and it still worked. Now I see that you've multiplied the rotation matrix with view matrix and the result with projection matrix. I tried it your way and it worked too. But from what I understand of 3d mathematics is

that matrix multiplication is commutative and they both should not give the same result.
P.S.- I can't thank you enough for the way you've explained everything. It cleared almost all of my doubts.

Admin

October 16, 2015 at 2:15 pm

Hi Utkarsh, this might help: http://www.learnopengles.com/understanding-opengls-matrices/

Pingback: [Android]OpenGL Study | Dion's Note

Garry

January 22, 2016 at 3:42 pm

Hi guys I just wanted to make the page longer, thanks.

kishorramani

February 25, 2016 at 11:37 am

Hi Admin,
This is nice Demo for beginner.
I read sankar ,Hank and your reply…
now i want to draw star instead of triangle what can i do?
which site is helpful for this creation?

herve terrolle

March 11, 2016 at 2:38 pm

Hi,
First ! thank you for sharing your work.

I am new in opengl and android. And when i tried to run the lesson7 in found out that one package was missing.
import com.learnopengles.android.R;

It is possible to get it ?

thanks.

John

March 16, 2016 at 7:38 pm

Has anybody got a working OBJ loader that I can use? I am looking to build a simple game but I realise I am going to need more complex shapes in order to do this. I am following Kevin Brothalers book on Opengl ES and have built a basic Air Hockey table. This book does not cover an OBJ loader which maybe a suggestion for the second ed!

Regards John

Валерий

July 6, 2016 at 3:17 pm

Большое спасибо за прекрасный урок!
Таких мало в интернете.
Здоровья Вам и всего самого хорошего!

Pingback: [OPEN GL ES 2 Android |]()

Learn OpenGL ES / Proudly powered by WordPress