**1**

**Intro self and title**

**2**

**Break down**

**Intro – what it is – does- versions**

**How to code**

**No live demo**

**3**

**Stands for**

**ES types**

**Android and IOS**

**2D 3D renders**

**4**

**GLES subset of GL**

**Gl desktops**

**GLES – cheaper simpler hardware**

**main difference – code- gl: brackets**

**BUT both render 2D and 3D graphics**

**5**

**Simple graphics**

**Complex apps**

**Popular game**

**To show gles capability**

**Vulkan**

**‘problem’ of gles**

**6**

**Many versions**

**Very first 1.x**

**Overhaul v2.0 – shaders and pipeline**

**Improving on 2.0**

**People use 2.0 – because FPS**

**7**

**Graphics pipeline**

**Data – image**

**8 Surface View**

**Before coding - need view container for graphics**

**Gles has activities – uses GLSurfaceView**

**Canvas - can draw and manipulate objects**

**9 Surface View Code**

**Code for surface view –**

**SHOW activity and class**

**Needs renderer code – determines gles version- used for next step**

**10 Renderer Class**

**Controls what is drawn on SurfaceView**

**Contains 3 V.imp methods**

1. **onSurfaceCreated() - Called once - when creating the [GLSurfaceView](https://developer.android.com/reference/android/opengl/GLSurfaceView).**

**sets OpenGL environment parameters**

**OR initializing OpenGL graphic objects.**

1. **onDrawFrame()- Called for each redraw of the view.**
2. **onSurfaceChanged()- Called if the geometry of the view changes, for example when the device's screen orientation changes.**

**11 code**

**SHOW methods**

**Will be added to later**

**Creates EMPTY black background**

**12 defining shapes**

**We set up raw data**

**Triangle IMP**

**Vertices**

**Winding – Culling**

**Overlapping – simplifications**

**13 code**

**SHOW coordinates**

**Programmer decides CCW**

**SHOW floating array**

[**ByteBuffer**](https://developer.android.com/reference/java/nio/ByteBuffer) **is passed to pipeline**

**Culling**

**OnSurfaceCreated calls triangle creation method**

**14 shaders**

**V.Imp for drawing**

**Shaders –> Code that render**

**Vertex shader -> renders vertices**

**Fragment shader -> renders shape face/colour/texture**

**Program -> contains shaders**

**DLSL – compile before using GLES environ**

**15 code**

**Ex of triangle shaders**

**To compile this code, create a utility method in your renderer class**

**16**

**Add shader code to OpenGL ES program object**

**Link to program.**

**This is done in the drawn object’s constructor, so it is only done once.**

**17 draw shape**

**Call draw() – sets shaders, position and colour – executes draw fn**

**Shape classes contain own logic**

**Call draw() from onDrawFrame()**

**18 code**

**Read comments**

**Call draw()**

**Complex shapes**

**19 result**

**Describe result**

**But**

**Rotate**

**Stretch**

**20 Mapping**

**How gles maps coords to screen**

**Uniform coord system**

**Assumes perfect square**

**(-1,-1) -> (1,1)**

**21 proj and cam view**

**Transform coords - from OUR eyes**

**22 pics**

**SHOW cam proj**

**SHOW cam view cone**

**SHOW frustum – describe use**

**23 Defn Proj**

**Adjusts coords based on W&H of GLSurfaceView**

**Both data for AND proj transf are calculated in onSurfaceChanged()**

**Matrix.frustumM() populates proj transf Matrix**

**24 code**

**SHOW frustum()**

**25 Defn Cam View**

**Adjusts coord based on VIRTUAL cam postn**

**Cam view changed – user or application**

**Is calc using Matrix.setLookAtM()**

**Combined with proj matrix**

**26 code**

**SHOW Matrix.setLookAtM()**

**SHOW combined proj and view**

**27 Applying proj**

**Matrix variable MUST be added to VERTEX shader to combine view and proj**

**Draw() is modified to accept transf and apply to shape**

**28 code**

**Vertex shader code – added MatrixHandle**

**Draw() added MatrixHandle**

**29**

**Correct result**

**30 can have motion and touch sensitivity**

**31 conclude**

**Opengl es is interesting to use**

**not the easiest to use for complex graphics**

**Thank you**