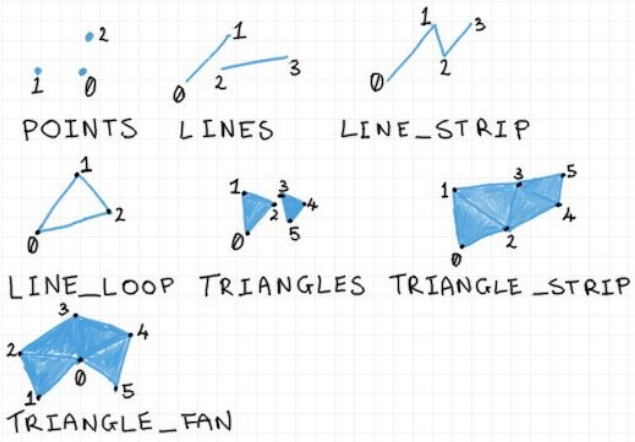
1 – GLSL vem da sigla OpenGL Shading Language, que como o nome sugere, é uma linguagem de programação usada para desenvolvimento de shaders dentro do pipeline do OpenGL.

Vertex shader: processa cada vértice separadamente. Seus inputs são os vértices do triângulo codificados com base no plano cartesiano e seu output são os vértices transformados exibidos na tela. Esse shader não pode criar novos vértices, apenas modificar os vértices existentes.

Fragment Shader: processa cada fragmento separademente, no que diz respeito as cores de cada pixel dos fragmentos que compõe o triângulo. No OpenGL as cores são formadas pela combinação das cores vermelho, verde e azul (RGB) + fator alfa (que determina opacidade do pixel), escalados de 0 – 1 de acordo com o sistema decimal.

2 – Primitivas gráficas são as instruções/comandos que fornecemos ao software para transmitir o que fazer com as coordenadas fornecida. Por exemplo:



Os dados são armazenados (e especialmente enviados para a GPU) através de buffers. No caso dos vértices, é o vertex buffer object (VBO).

3 – VBO: é o conjunto de dados (de vértices) que compõe um objeto. Na prática, um conjunto de coordenadas no plano cartesiano e/ou as cores de seus pixels.

VAO (vertex array objetc): de maneira muito simplificada, é um conjunto composto por diferentes VBO. Por exemplo: um VAO pode conter um VBO de posições e outro VBO de cores. Dessa forma, um VAO é o que de fato se precisa para formar uma imagem.

EBO (Element buffer object): tem funcionamento similar a um VBO, mas armazena índices, em vez de vértices. De maneira geral, é um recurso para simplificar o uso de vértices. Como o OpenGL funciona a base de triângulos, muitos vértices acabam sendo sobrepostos para gerar imagens complexas. Com o EBO, é utilizado o número mínimo de vértices para formar a figura desejada.

