Vortex Engine 2.0 Programming Log:

May 13, 2019:

-created the package hierarchy, will contain core package to house all core utilities the entirety of the engine could not possibly run without, as well as eventually the resources package to house structures like 3D models and animations etc. when they are eventually implemented

-started in core, created a window package to hold everything to do with displaying the window to the screen, am using lwjgl library to create the window

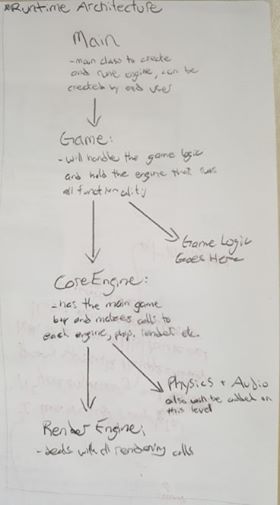
-window class for now will be created as a static class for a single window as I am not sure how I will want to implement window management, this implementation will possibly change in the future

May 14, 2019:

-after looking into component-based architecture and rendering I have decided to stick with the node/component based system used in the previous vortex engine 1.0 where everything to be used in the game is a game object (a type of parent node capable of calling all its methods on all its children) capable of containing many components that will determine how the game object interacts in the world.

-also decided to stick with the overall engine architecture (well mostly similar if not identical) where the game class holds all the instances of both game logic and the engine, the engine will be broken down into two major stages the core engine which will run the core game loop as well as timers and anything that any of the dedicated engines could need, it will also make calls to the subsidiary engines such as the graphics, physics and audio engines.

-above described implementation will look something like this (very similar to vortex engine 1.0)



-decided a proper naming for the Game method should be Root since it is more descriptive of the classes function

May 15, 2019:

-implemented the same maths library from vortex engine 1.0 along with a few extra methods for a more completed functionality

-move some methods around in the maths library to places I felt they would be better suited

-implementing the same model architecture as vortex engine 1.0 where a model has a material and a mesh, and the mesh is an ordered list of vertices as at the moment it seems like the best implementation.

-bitangents are no longer being stored in the vertex class as they can be calculated in real time and in the past, I haven’t found much need for them so storing them would only be a waste of memory

-the vao for the time being will not be implemented in the model class as I may put it somewhere I feel it will better fit, still deciding, otherwise the model implementation has remained unchanged

June 10, 2019:

-completing implementing the obj loader, used the same format as vortex engine 1.0 since after some googling it turned out id already used a highly optimized obj loader so until I can find a better one I will be sticking to the old version

-implemented the same shader from vortex engine 1.0 the render implementation could be finished, took all the info from renderInfo class in vortexengine 1.0 and implemented it directly into the renerer class instead since the renderInfo class served only as a container for two other classes and I thought it made more sence and looked cleaner if they were both implemented directly in the renderer

-implemented a simpler version of the vortex engine 1.0 camera without the buggy free camera movement, will attempt to re implement free camera movement when a better method can be found

-almost all prerequisites to rendering are in place, will next attempt to insert the recently added functionality into the render engine loop and attempt to render something to the screen