**Milestone 3**

**Briefly describe the artifact. What is it? When was it created?**

This project was created as an introduction to OpenGL Graphics API programming by referencing the LearnOpenGL site (LearnOpenGL, n.d.). The initial code for the project involved building a 3D scene of our choosing based on everyday objects.

My scene was built to depict a candle, 2 vases, and some stuffed pumpkins my wife had for Fall décor. I believe I built the scene last August?

The code enhancement for the artifact looks to streamline the rendering processes, by removing redundant code in the Primitive Meshes (Cube, Cylinder, Pyramid, Plane, Sphere) and moving it to a base Mesh class. Using inheritance, the primitives would be able to reference these methods, without me needing to change all meshes when I make 1 change. It also improves the Data Structure to allow for the code to be further expanded to allow for new features at a later date.

Another change made by adding this parent class methodology, is now meshes can manage their own linked textures and tell the Shader about their textures. Before the change, if I added a new object, I had to manually assign textures in OpenGL’s state machine, in the proper order, and notify the shader of the objects’ attributes. Now all of this is handled in the object’s Draw() method.

Simplifying the draw logic for these meshes also improved runtime to a point where I believe is more acceptable. I was seeing 5-10 second loads at the beginning, and now its down to about 3 seconds. Pre-compiling shaders would probably be another enhancement that would pay dividends.

**Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?**

I chose this artifact because GPU programming is a valuable skill in the modern age. Its used in computational heavy projects, like Scientific calculations, simulations, cryptography, and games. Being able to show that I understand and can work my way around GPU programming and Shaders would be a boon to my portfolio.

As stated previously, I simplified the primitive mesh classes, and removed the redundant and repeated code, and moved it to a shared base Mesh class. This simplifies any changes that affect all meshes, since its managed from the single base class.

This also provides a gateway to further expand the functionality, which I have already done by making meshes the owners of their Textures, instead of the programmer having to add them everytime a new mesh is added. When a Mesh calls its Draw method, it will now notify the GPU of its textures its using and it will notify the shader of any attributes it has. This has greatly reduced the complexity of the Render loop, and saw a speed improvement at runtime.

**Did you meet the course objectives you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?**

I believe I did. I did have to reduce scope a bit with the improving the rendering algorithm, but even then, I still hit that pretty well by reducing the complexity of the loop and improving runtime.

The changes I have made also improved modularity, and scalability in the long run. The meshes (and by extension the textures) can now be expanded to have far more functionality, like adding a Material class to manage textures and their individual attributes, or opening the way for complex meshes to be managed by a greater class, or even adding in Mesh loading from popular tools such as Blender (Blender, n.d.).

**Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?**

I learned of the hubris of man, and the insurmountable amount of spaghetti code one man can produce in a caffeine-induced rush. I also learned how to sift through said code, and fix a good amount of it without breaking it, especially after not touching the code for nearly a year.

This project was one I really enjoyed when I started wrapping my head around it during the course (it took many weeks of anguish before I hit that point). I was torn when the course ended and had no reason to touch it anymore. Then I heard my Capstone would have me choosing projects to enhance, and I instantly knew I wanted to work on this one.

Unfortunately, I did not realize how much of a mess the codebase was. This project was my first true foray into CPP programming so that probably tells you a lot.

The enhancement project was a lot like looking at code a novice, or a madman wrote, and took quite a bit of deciphering to pull apart and produce code that wouldn’t fully break the project. There is still an issue of World Space versus Local Space matrix modification that did cause me to have to hack a fix for the pumpkins in the scene, but for the most part, I feel I grew as a developer by improving this project, and making it far more modular and scalable than my original attempt.

**References**

LearnOpenGL (n.d) *LearnOpenGL – Graphics Programming*. <https://www.learnopengl.com>

Blender (n.d) *Blender – a 3D modelling and rendering package*. https://www.blender.org