Justin Smith

Before this class, my level of game engine usage was pretty limited. I used to work with Radiant (A game engine for Call of Duty Custom Zombies), but otherwise, the only other engines were either Unity or Unreal Engine. I also took EECS 498 last semester to gain some more time within both engines academically. The three shmups at the beginning of the year feel like forever ago at this point, but I had the most fun being the only Clickteam Fusion developer (hahaha). I felt that I had the most difficulty using Clickteam because of the outdated tutorials and very interesting layout. Although it was still pretty straightforward to get what I wanted! I still felt that Unreal Engine and MonoGame felt more intuitive than what I am used to, but I think it was a great way to introduce myself to engines that I would never have touched outside on my own.

During week three, this was the first time where it began to take off into a more standardized EECS 281 project. I felt the experience was not as stressful outside of navigating the setup process for each OS. I also faced an issue with WSL and networks due to the public firewall. This meant that I was unable to upgrade or access Gitlab through my computer when developing. The only solution was to go on campus or through a Virtual Machine (which was kind of annoying). I think as the weeks went on, I could figure out any errors and linker/dll problems more quickly. In general week three was a more relaxing process that did not take too much time out of my week. I think hands down the toughest week was homework 6. The issue was not the introduction of spatial hashing or any performance concepts, but to get to a *point* where Profiling and optimization would be necessary. I, unfortunately, fell under a loop where I would fix one test case off by a one-pixel error, but another test case would fail. I REALLY found the idea mentioned in the discord with outlined red boxes to be helpful (even necessary) since it helped me realign or visualize anything that went astray during the process. I hope future generations are offered this advice (or even make it a necessary debugging feature) since this eventually got me out of that bubble (although I was up until 4 AM D:). My favorite week was the introduction of LuaBridge. It was so satisfying to delete all the ugly code I wrote from homework 6 since it felt rushed and stressful to debug. My weekly development process would usually start on the Thursday of every week. This meant I usually spent about 4 hours a day up until the deadline on Monday. This did bite me a few times (homework 6 and 7), but I think the payoff was worth it in the end.

My code was for the most part DRY, but when it became a problem of due dates, I will be honest, there was lots of duplicate spaghetti code just so I could go to sleep. I would then in the following week either reflect on my life decisions, delete it, or it becomes irrelevant that it would be removed entirely. As I progressed into homework 7, the circular dependency issue became less prevalent and I was more organized in class structure. Every time I return to my engine, I have the least trouble navigating my code, but I do believe my main.cpp needs a rework for initializing. The quality of my code is documented pretty well. At times, if I was stuck and needed AI assistance, I made sure to provide explanations so that I could always refer back to anything that I may not understand later on. I believe AI was a very helpful tool, especially for homework 7. It knew how to answer lots of my questions regarding LuaBridge, metatables, and overall navigating from C++ to the Lua realm. It also was a great tool sometimes when the specifications did not click for me, so I asked it to reword or provide a more descriptive understanding of what was being asked. UMGPT was the best resource since it was very knowledgeable of any weird issues that Stack Overflow could not resolve.

My Custom feature was the integration of OpenGL and OpenVR. I was able to get all haptic and positional features for the VR headset and VR controllers to work, and as I transitioned to OpenGL, there was a large jump in difficulty due to my limited background in 3D graphics. So I was only able to finish a small subset of what I wanted to accomplish in two weeks, but I really want to keep developing on this in the Summer in my free time. The reason I wanted to explore this feature was because of YouTube videos in my feed of people developing cool contraptions with their custom VR tools in their game engines. I assumed that the task was possible, so I wanted to try and get a start on it as my custom game feature. I think alongside OpenGL and OpenVR, I want to make a UI with my game engine, change over to YAML, use Cmake, and provide some multiplayer to it. I think there is much you can learn from these features and would be a great way to branch out. I was able to meet performance requirements every week (except homework 6). I think this meant that I feel that my knowledge in this subject has peaked since I started and I am very proud of what I have been able to accomplish. I think my lack of office hours and community made it a bit more difficult because the exam room felt very weird seeing everyone for the first time. I also need to focus on limiting how much time I invest one day on a feature. I think there is no schedule or idea of a stopping point when I develop. Because of this, it can feel a bit overwhelming or unhealthy to develop at times. I think this engine should be pretty useful for my portfolio since it is a very rewarding accomplishment. There is so much to talk about (and eventually prove) as long as I spend time this summer also building games on it as well! I have already discussed this game engine with recruiters and they love how I integrated Unity Lifecycle functions into my engine. It was fun talking about how the underlying processes work now that I had a chance to write these functionalities myself.

My confidence in C++ is much higher than it was before. I believe that many concepts in OOP resurfaced and I felt more fluent navigating and solving any errors and performance issues with ease in comparison to large projects I have done before in this language. I feel that I have developed a larger respect for the language that I have taken for granted when using a less verbose language such as Python. My biggest takeaway from the last 3-4 months is that there is so much more you can touch when it comes to game engines. There are so many features and upgrades you can make, and this class only scratched the surface. I hope that I can continue to work on the engine to reach a more professional level that I can show off to my friends and family one day!

I think a development blog would be a cool way to gauge more on how I progressed each week. I think discussions could be a great way for people to slowly develop this as a required process. I would also like for the specifications to be slightly refined for legibility purposes. There were times I needed to ask AI to reiterate.

EECS440 was a very lightweight and enjoyable class and I feel that it was a great transition into a more coding-intensive class as this one. I think my background using UE and Unreal Engine was a great introduction to game engines, which made this class more intriguing to take from an interest standpoint.

Thank you all for a wonderful semester! (If you read this Rahmy Salman said hi)