Chen Xin Lin

Modding Halo Custom Edition Culmination Reflection

Initially, my culmination project was about creating an FPS game in Unity, but it wasn't interesting after some time of working on it. It feels like a task that I have to complete for a grade. And it wasn't what I'm looking for when I joined game design. In fact, I thought that I could pick a game of my choice, in this case, Halo CE, to work on: dissecting, learning the programming language it uses, and visual level design. In early October, I pitched to my technical advisor that I would like to change my culmination project to modding Halo CE. Then he told me that if I could show him enough progress after one week, he would let me change my culmination project to modding Halo CE. And so that's what I did, I showed him enough progress the next week and therefore, he agreed for me to switch. With the go-ahead, I quickly did a lot of research and brought myself up to speed with the programs that are used in modding Halo CE. I found out that the programs that I needed were Sapien (visual representation of the level), Guerilla (creating AI properties, and other things with limited documentation), Tool.exe (used for compiling the final file: .map file and other behind-the-scenes resources). At first, I was ambitious about creating my own giant model of a map in 3DS Max, but, unfortunately, Tool.exe wasn't creating the shaders for the map terrains. With that, I had no choice, and with the limited time I had, but to customize an existing map with AI bots and custom vehicles. Who would have thought, the map that I was customizing already had a huge file size? I later found that the multiplayer map has a file size limit of 128 MB. Tool.exe wouldn't compile into a map file if it is over 128 MB. I even researched how to hex edit Tool.exe so that I could compile over that file limit. However, even if I managed to trick Tool.exe into creating the file, the game won't load into the map. In fact, the game outright crashes (gathering exception data). Once again, I decided on the map, DeathIsland, which was 15x smaller than the initial map I was going to customize. During modding, I noticed that vehicles won't spawn in the final .map file. Once again, I went back to the internet and looked for answers, and then I found an old post that helped me solve the vehicle spawning problem. But there was a catch, as I noticed a few days after, they spawned all in one place when I clicked on "restart" in-game. I went back to the internet, and this time, I found the right script for what I'm doing. In mid-November, my technical advisor told me to try to create a scenery object and imported it into the game, but once again, Tool.exe won't create the shaders for it. Therefore, it was another failed attempt. All of those processes took some time and there is no official documentation of how to do it, it is just bits and pieces that I have to put together and make sense.

That's one of the reasons why I signed up for game design, a game of the student's choice to dissect and learn things about that game including, but not limited to programming language, visual representation of the level design, etc as a starting point. In that way, it is easier to relate back to the game the students had played, and therefore, easier for the student to understand this is that or that is this, or why it is this way as they themselves had played it and loved it. At least that's how I would absorb the information much quicker. When I started learning Unity, it took me 1 year to figure out what "prefab" even is. But, in modding Halo CE, I know exactly what "Tags" means, in less than 5 minutes, because I had played the game and can refer back to it. Again, in Unity, I have no such reference in my mind and therefore it was hard for me to grasp what it looks like. Moving on, as a kid, I was always wondering and interested in how they made those levels in the original Halo Combat Evolved for the Xbox. When I was modding Halo CE, I

felt like I was a real developer, dictating every AI's fate and what team they're going to be, and who they're battling. For the past 2 years in game design, the only game engine they had taught me was Unity which I was disappointed with because Unity isn't what I'm signing up for in the first place. I want to learn game design from a game of my choice, in this case, Halo CE, because existing games build on the fundamentals of game design; tearing something down is much easier to understand than building it from the ground up. And then you have to keep in mind what to do and what not to do. What I mean by this is that learning about Halo CE first would massively help me understand other game engines and their jargon. As I mentioned before, it took me so long to know what exactly "prefab" is and other words and their meaning. In other words, there's not one way of learning things, but the way they had set up to only learn Unity in game design makes me disappointed. Anyway with modding Halo CE, you got to learn how to 3D model the maps and everything else in the game, add bitmaps, learn to use the command prompt (in a very specific way), edit/create properties, animations, rig a character, and much more. This is all the stuff I want to learn about Halo CE. And as you can see, it has almost all the elements in Halo CE that are also in Unity as well. But the difference is that Unity requires much more scripting in order to work which is the part I hate. Halo CE scripting is what I call light scripting as you don't need to worry about collision detection scripts or AI scripting. In the end, I solely enjoyed modding Halo CE to the fullest. Despite the fact that it had so little documentation, I still felt that it is my biggest accomplishment so far in those 2 years that I'm in game design as the past Unity projects, to me, felt more like a chore that I need to finish to get a grade. In the future, I want to create a single-player map with triggers and enemy encounters, and a little scripting.