Miguel Saavedra 13826904

Postmortem Overview - Butter Cat and Breakfast

Overview
What Went Right,

Sprite Spawning
This was relatively easy to use with the SDL framework as I had already done it during the tutorial activities and it didn't pose a challenge initially. This was when I was using static sprites as placeholders while looking for animated sprite sheets.

Fmod sound manipulation
Fmod was relatively easy to install and use to utilize for the game. I could create and set sounds to play on a queue without any hitches. however, I was not able to release the sounds on quit.

SDL TTF
This also was easy to install with the help of the internet and information provided by one of the student's posts. However, I did not find a modular way of drawing the text on the screen without repeating unneeded code at times.

What Went Wrong
Animation
At first, it spawned the whole sprite map, soon after realizing while reviewing the lecture slides that I had to create a whole new function to cut the sprite map and render it. This then posed a problem with custom made sprites for my game, as I had to reposition them accurately so that part of the previous frames sprite is not seen in the next ones.

IniParser
This caused the most amount of pain as reading in the file and parsing its body did not work. Then trying a different approach by reading it in as a stream finally worked for me. Although there were many hiccups with reading in sprite locations, as I thought that it was the parsers fault but after resaving the png from Photoshop it fixed the problem.

Memory Leaks
There still is some memory leaks however quite small within the game at the moment. In the beginning iterating over a vector and erasing the elements within it, after some googling, I found that you would have to return pointer after erasing it which solved my problem.

Repeated Code and Bad Design Decisions
As I was not sure of many things such as in what way would some middle wear have to be implemented and used within the game and how to do animatedSprites, I ended with a lot of repeated code and dirty code in some places. I also did not end up follow coding standards with

variables and class naming conventions with half being camel casing and the other the Microsoft standard.

Lessons Learnt,
Memory leaks
- Check memory leaks early
- Fix them Early

Repeated Code and Bad Design Decisions
- Take more time to think about and design to loosely couple classes from each other
- Design more general functions which do autonomous behavior

Collision
- Implementing and testing collision first and polishing it ensures good gameplay

Conclusion.
This project taught me more about the quirks of c++ compared to java as memory management is a new thing to me. This has also proven to be a good lesson with the design of classes, loose coupling and object life cycles.
If I were to do this again I would design based on the core features first keeping more in mind modularity and consistency with coding standards.