First Problem - Which Emulator?

- To figure out which emulator to continue development on required a meeting with Andrew Borman at the Strong Museum of Play about what the game preservation space needed to prioritize in terms of emulation. The initial choice was between Yuzu and Xemu. Yuzu is the Nintendo Switch emulator and due to it's popularity it has a very close full rendition of the Nintendo Switch experience. Xemu on the other hand has been known to be much more hacked together and only more recently (within the last year or so) has there been a push to make games more compatible with the emulator through its developer community. So the clear choice, while more difficult, was Xemu because it still had a ways to go in terms of reaching a good level of game preservation standard that the Museum could potentially learn from.

Second Problem - How does Xemu work? What is Qemu?

- A large portion of the early part of the research process was understanding how the Xbox emulator Xemu worked.
- From there more questions and problems arose with the introduction of its underlying emulation from Qemu.
    - A state diagram was created on how the Qemu and Xemu states worked respectively.
- The solution for this research question is that a general knowledge of how Xemu works was found, but specifics are only known for whatever part development is needed on. A whole curriculum encompasses how this software works in its process to emulate hardware.

Third Problem - How to best build/run Xemu?

- According to Xemu documentation the build process heavily favors the Linux side of things and makes use of build systems such as ninja and meson.
    - Getting a proper build working was probably one of the longer parts of the research process as there were many roadblocks to getting a stable build
- This meant that Windows based build wasn't possible and the recommended route was using WSL (2.0) and docker to create a linux-based build environment.
    - WSL is the Windows Subsystem Environment and allows a real-time environment of Linux directly on top of the Windows System without dual-booting or having a virtual machine present, which reduces overhead and improves performance.
        - Problems with WSL
            - Seemed like a performant integration with Windows, but had some intricacies that made it hard to work with since the environment wasn't its own individual linux shell.
            - Docker had some issues as well with the PWD of the linux shell as well as requiring sudo permissions which had to be approved through Windows Powershell first which slowed things down and made it tedious.

- Having no real development experience with Mac OS this operating system was skipped
- Linux was the next step for building the xemu project. Unfortunately I had to operate under the overhead of a virtual machine so my Linux builds and testing were significantly slower than the Windows counterpart as it was not the base system.
    - Some steps were taken to try and improve this performance loss. Increased cores and memory allocation, and a Pro Version of VMWare was attempted which gave slightly more stable performance, but it had issues with it's networking toolset and was unreliable in connecting to the internet
    - Certain parts of the configure path for the project couldn't build on their own making it hard to configure files
    - Linux Emulated CPUs don't work nearly as efficiently as they would have if the environment was the host.
        - CPU threading isn't properly distributed
        - CPU utilization is run from the host system to quest causing slowdown
    - VirtualBox ran from anywhere between 5-30 frames per second for titles depending on the stress on the system
    - VMWare Pro ran anywhere from 5-40 frames per second depending on system stress
- The ideal solution for this problem is to run development on either a solely Linux system or dual boot whatever device is being used. Because Xemu and it's Qemu interior is very CPU intensive having the full system's power is the best way to go about the emulator development process.

Fourth Problem - How to Research?

- Being new to research it was clear that my inexperience led me off on the wrong track numerous times and that I wasn't stepping properly through the process
- The solution for this was from Professor Pai's advice, take the problem down to the fundamentals:
    - Find a problem
    - Replicate said problem
    - Analyze what you see (find data etc.)
    - Produce a plan to solve the problem
    - Try the solution
    - Either problem solved or iterate again through the steps
- I often went ahead of the process and became overwhelmed by the scope of what I didn't know that I forgot to slow down which created more problems than removed.

Fifth Problem - Project Gotham Shadows & Audio issues found in Xemu

- It was reported that Project Gotham started but was nearly unplayable due to graphical and auditory errors
- From testing its confirmed that various graphics assets do not render correctly and the audio is not streamlined but instead playing from many sources at once which can be seen from its 100% utilization within Xemu's audio debugger.
- (Need to get data out of the app to be able to analyze)
    - Log gives significant info, need to link this to trace-events in order to get information out
- Possible Solution?...
- Try Solution…