Mel Aise

genericpb@gmail.com // https://github.com/yurapyon

p r o j e c t s

maru // OpenGL game/graphics framework // https://github.com/yurapyon/maru

maru is a graphics framework designed to make it easy to get up and running with motion graphics or game projects. A video made with an older version of the software can be found here: https://www.youtube.com/watch?v=loHvrW5EKBk

maru's graphics functionality generally aims to be a thin wrapper over OpenGL and GLFW. Users of the library are expected to have some knowledge of these libraries; the main goal is to encapsulate most of the boilerplate needed when starting new projects. Designing a library in this way provides the user less resistance in structuring their projects, compared to working with off-the-shelf game engines.

The library also includes an example of a 2D game framework, featuring 2D meshes, a spritebatch, bitmap fonts, and templates for shaders. A fair amount of functionality was inspired by LÖVE, after looking through the source and seeing that LÖVE is, in some ways, also a thin wrapper over OpenGL. A framework for 3D graphics is currently being implemented.

kasumi // realtime audio engine // https://github.com/yurapyon/kasumi

kasumi is a realtime audio engine for use in games and digital audio workstations. While it currently only implements an audio graph and sample playback, it's planned to implement functionality such as 3d audio, audio effects, and realtime sound synthesis.

The core data structure is a directed acyclic graph. Nodes in the graph represent audio generators or audio effects, and the user can chain these together for their processing needs. Ideally, kasumi could be used in some of the situations FMOD is used, such as interactive audio, however being able to use the library for music production is also a goal. To this end, kasumi is mostly concerned with high-level structure rather than lower level details like hardware audio I/O.

Still, there are some low-level constraints that need to be accounted for: namely, realtime audio threads must avoid any code that could block the thread for an unknown duration. Some examples of this are allocations and mutex locks. To get around these constraints, kasumi makes extensive use of lock-free queues. In the case of allocations, all of them are done in the main thread, and when finished, pointers to the new memory are atomically sent to the audio thread. These queues are also used for communication between threads, such as when events that happen in the main thread should affect the audio.

o ther information

I'm self-taught in programming and game development. The languages I use the most are Zig and Forth. I have experience with C, Scheme, Lua and Rust as well. I mostly work on video game related projects but I also have an interest in language design and microcontrollers.

I attended the Recurse Center Winter 2017 batch. This was a great chance to meet other programmers. I practiced public speaking and pair programming, learned functional programming concepts, and deepened my understanding of data structures and algorithms.