Devon Raymond Murray

(203) 893-6507 - devonrm98@gmail.com – Seattle, WA 98199

Career Objective . . . .

Work as a game developer, pushing the boundaries of what makes a game fun through making unique mechanics, intricate sounds, and robust systems. Whether I'm programming gameplay, audio, or tools, what matters most is that I'm passionate and proud about any project I work on.

Qualifications . . . .

- ○ C++ programmer skilled in real-time systems, software architecture, and linear algebra
- ○ Demonstrated analytical and quantitative problem-solving skills
- ○ Capable of handling large workloads under intense pressure and deadlines

Work Experience . . . .

Dark Catt Studios – Gameplay Engineer Intern | Saint Charles, IL | Jan. 2019 – June 2019

Credited on *Djinni & Thaco: Trial By Spire.* (Game Page Link)

- ○ A VR Tower Defense game made using Unreal Engine 4 Blueprints. Fend off enemies by pounding the ground, casting spells, and building towers.
  - ▪ Debugging and programming enemies, audio systems, and controls.

Education . . . .

B. S. in Computer Science – *DePaul University* | Chicago, IL | Sep. 2016 – Mar. 2021

Concentration in Game Systems

Relevant Projects & Coursework . . . .

Modest Engine (GAM 372/GAM 377) [C++, FMOD] (Click for Videos)

Created a Game Engine, during my degree, to help with developing a game. Consists of several core systems as well as Audio research.

- ○ Modest Sound is an audio API made with FMOD Core.
  - ▪ Features modifiable effects in real-time, creation/destruction of managed sounds, and 3D audio.
  - ▪ Simplifies resource management for the user.
- ○ Game Object System
  - ▪ A polymorphic system, which a game object can derive from several different sub-types: Inputable, Collideable, Updateable, Drawable, and Alarmable.
  - ▪ All objects are registered and deregistered for each of the types listed above.
- ○ Collision System
  - ▪ Tiered collision volume testing and template classes allow for any object with a collider to be passed in and checked efficiently.
  - ▪ Volumes: BSphere, OBB, AABB, and Ray Casting.
- ○ Scenes
  - ▪ Management of Scene switching, creation, and termination.
  - ▪ Keeps track of objects, draws, updates and processes commands.
- ○ Terrain System
  - ▪ Players/Enemies move along the terrain using Barycentric Coordinates.
  - ▪ Terrain is a grid of AABB volumes, staircasing with the heightmap.
  - ▪ Terrain is created with a texture and heightmap through OpenGL.

- Math Tools
    - Static math functions that can be called from anywhere.
- Alarm System
- Input System
- Sprites, Fonts, & Screen Logging
    - Updateable debug text that is displayed to the user's liking.
    - 2D Sprites for UI, HUD, and more.
    - A system for Font management, so the User can load different fonts.
- Resource Management
    - User chooses to Load in assets (fonts, textures, terrain, sprites, sounds) with one file, calling static functions from each manager in each system.

Vox Tanx (GAM 372/GAM 377) [C++, Modest Engine] (Click for Video)

A simple tank game demoing the features of the Modest Engine (see above).

Optimized C++ Multithreading (CSC 362) [C++, WaveOut]

- Jetsons
    - A system that reads multiple audio files, writes to, and stiches together twenty 2K-byte audio buffers, playing them seamlessly, as if it's one file.
    - Simultaneously loads, writes to buffers, and coordinates which/when audio is played. Using Circular Queues, Futures/Promises, and waiting.
- Multithreaded Maze Solver
    - An exercise in making a DFS maze solver faster through parallelization.

Bumper Ship Showdown (GAM 392) [C#, Unity] – Lead Programmer

A 3D online brawler where players partake in a dangerous sport, that is essentially bumper cars in space with more objectives. This was my capstone project.

- Gameplay, Audio, and Network Programming.
    - Events, Game Modes, UI, Optimization, Debugging
- Music Composition and Sound Design

Real-Time Multithreaded Architecture (CSC 488) [C++, XAudio2]

Created a Multithreaded Audio API using basic multithreading techniques and software architecture strategies.

Game Performance Optimization / Optimized C++ (CSC 361)

- Reworked structures to improve memory usage and data caching.
- Enhanced Vector and Matrix math library using SIMD instruction set.
- Developed heap-based memory system for improved data management.
- Created a fast read/write data storing system using OS-native functions.
- Final Project: Refactored 15K particles system for class competition.
    - Implemented SIMD, modified main loop and math systems.

Technical Summary

Programming Languages:	C / C++, C#, Python, Java
Frameworks/ Libraries:	STL, Unreal Engine 4, Unity, DirectX11, FMOD Core, XAudio2
Agile Platforms:	Taiga, Trello
Platforms & IDEs:	Windows, Visual Studio
Version Control:	Perforce, GitHub, Tortoise SVN