Parniuk Maryia

JUNIOR TOOLS PROGRAMMER

C++ & C# Developer | WPF | Qt | Game Dev Enthusiast

+456 40 93 93

Portfolio: maryia.be

Github: GitHub
parniuk.masha@gmail.com
Belguim

I am a passionate Junior Tools Programmer with hands-on experience in game development and desktop application development. I hold a degree in Game Development and recently completed an internship at Larian Studios, where I contributed to tools and optimized production pipelines and enhanced developer workflows. My technical expertise includes C++, C#, .Net, WPF and Qt.

SKILLS

- **Programming Languages: C++, C#, XAML, C++/CLI**
- **Frameworks & Technologies: .NET, WPF, Qt**
- **Databases: SQL, Relational Databases**
- **Version Control & CI/CD: Git, Perforce, Jenkins, Swarm**
- **Scripting: Lua, Python**
- **Game Engines: Unity, Unreal Engine**
- **Augmented Reality (AR): Unity AR Foundation, Niantic Tools**
- **3D Graphics & Rendering: OpenGL, Vulkan API, HLSL**
- **UI/UX Design: Figma**
- **Build & Deployment: CMake**
- **Management: Confluence, Trello, Jira**
- **IDEs: Visual Studio, JetBrains Rider**
- **Compilers: GCC, Clang, MSCV**

EXPERIENCE

February 2025 - Current

Intern Tool Programmer
Larian Studios

Developed and implemented new features for custom tools, enhancing the efficiency of the development pipeline. Fixed bugs to ensure smooth operation and stability of the tools. Collaborated with cross-functional teams to gather requirements and ensure tool functionality met team needs. Participated in regular communication and provided updates to improve tool performance and usability.

June 2022 – July 2023 **Freelance UI/UX Designer**

Designed and prototyped user interfaces for web and desktop applications using **Figma**.Created interactive wireframes, user flows, and visual designs focused on usability and user experience.Collaborated with clients to understand requirements, gather feedback, and iterate on design solutions.Ensured responsive design principles were applied for compatibility across devices.

EDUCATION

2022-2025 **Hogeschool West-Vlaanderen (Howest University of Applied Science)**
Digital Arts and Entertainment

Deep learning of C++ and C# , including data structures, algorithms, and object-oriented design.

COMMUNICATION

Strong collaboration skills through group projects and communication with cross-functional teams, ensuring alignment and effective problem-solving.

RECENT PROJECTS

Project no.1: **Custom Asset Manager**

Stack: **WPF, C#**

Project description: Developed a robust tool for efficient management of game assets, including 3D models, textures, and audio files.

Achievements: **Key Features:**

Database Integration: Implemented SQL-based storage for efficient data retrieval and asset version tracking.

- **Tagging, Sorting, and Searching:** Streamlined asset organization with advanced filtering and search capabilities.
- **Batch Processing:** Automated import, export, and processing of multiple assets simultaneously.
- **3D Preview & Validation:** Real-time visualization and validation of assets, checking for issues like missing materials or non-manifold geometry.
- **Multi-Engine Support:** Designed to integrate with Unity, Unreal Engine, and other DCC tools.
- **Pipeline Automation:** Automated parts of the 3DS Max to Unity workflow, reducing manual steps.

- **Plugin Support:** Enabled extensibility for third-party tools and additional file formats.
- **Version Tracking:** Displayed asset status to highlight outdated or modified files.

The **Game Asset Organizer/Manager** is a desktop application developed in **C# and WPF** that streamlines the management of game assets such as 3D models, textures, and audio files. Built with the **MVVM architecture**, it enables efficient tagging, sorting, and searching of assets, with real-time 3D previews for quick validation. The tool supports **multithreaded batch processing** for importing, exporting, and processing assets simultaneously, significantly speeding up workflow. Integration with **SQL databases** allows for efficient storage and version tracking, ensuring that assets remain up-to-date across projects. Additionally, it features **plugin support** for Unity, Unreal Engine, and DCC tools, making it highly extensible and adaptable to various development environments.

Project no.2: **Self-developed SDL2/glm modular engine**

Stack: C++, SLD2, SDL2_image, SDL2_mixer, SDL2_ttf, ImGui, GLM (OpenGL Mathematics), Git

Project description: The engine is a self-made modular framework built using SDL2 and glm, created for building 2D C++ game projects. Designed with user comfort in mind, this engine integrates key design patterns from "Game Programming Patterns" by Robert Nystrom to streamline game development.

Achievements: **Key Features:**

- **Designed and implemented a custom BoxCollider component, tailored to enhance collision detection and response in game environments**
- **Created a multithreaded audio system, enabling simultaneous audio processing and playback to reduce latency and enhance audio performance**
- **Applied the State Pattern to AI behavior, allowing for flexible and organized management of enemy states and actions Implemented the Observer Pattern, enabling efficient and scalable event notification systems across various game components Integrated the Command Pattern for control management**
- **Implemented the Component Pattern to enable modular and reusable game components Utilized the Service Locator pattern to streamline and centralize audio management**
- **Developed a custom input system to handle player inputs across various devices**
- **Applied Dirty Flag optimizations to improve the efficiency of rendering and updates by minimizing unnecessary computations and redraws**

- **Integrated Finite State Machines (FSM) for handling complex animation states**
- **Designed a custom event queue specifically for audio events, improving the organization and processing of sound-related actions.**

Project no.3: **Self-developed 3D cross-platform Vulkan engine**

Stack: C++, Vulkan, Vulkan Pipelines, GLFW, TinyOBJLoader, GLM (OpenGL Mathematics), SPIR-V, GLSL, stb, CMake, Git

Project description: The engine is a custom cross-platform engine using the Vulkan API, designed for 2D and 3D game projects. It supports full 2D and 3D rendering capabilities. It manages multiple textures simultaneously and renders with multiple pipelines using Vulkan semaphores for efficient operation.

Achievements: **Key Features:**

- Developed a fully cross-platform engine, ensuring compatibility across various operating systems and hardware configurations
- Implemented efficient model loading systems for importing and rendering complex 3D models
- Integrated Physically Based Rendering (PBR) shaders for realistic and high-quality visual effects
- Enabled multithreaded rendering and processing to improve performance and responsiveness
- Supported comprehensive 2D/3D rendering capabilities Utilized Vulkan's push constants feature to optimize shader performance and streamline data transfer
- Designed a custom camera system for flexible and dynamic control over camera views and perspectives Implemented a depth buffer for accurate depth testing and rendering of 3D scenes
- Used staging buffers for efficient data transfer between the CPU and GPU, enhancing rendering performance
- Created a timer class to handle time-based events and manage frame rates accurately Integrated multisampling techniques to reduce aliasing and improve visual quality in rendered scenes

LANGUAGES

English – Professional Working Proficiency (C1)

Dutch – Intermediate Working Proficiency (B1)