CS 499 Computer Science Capstone

Daniel J. Vidmar

Southern New Hampshire University

CS499: Enhancement One Narrative

Gene Bryant

Artifact Description

The artifact I selected for Enhancement 1 is a Java-based 2D survival game built with the libGDX framework. It originates from my earlier CS 330 project, a C++ OpenGL visualization tool designed to render 3D objects and camera controls. For CS 499, I enhanced and repurposed this artifact by migrating the project into Java and evolving it into a functional 2D tile-based survival game engine. The current version includes a modular game loop, player movement, rendering, and a robust system for procedural terrain generation with biome classification.

The core feature added in this enhancement is a noise-based terrain generator. This generator uses OpenSimplex noise to assign terrain tiles (e.g., grass, water, forest, mountain) based on height values and biome thresholds. Biomes are determined using noise blending and configurable parameters for moisture and temperature. The game world is chunked and scrolls dynamically as the player moves, simulating an infinite environment.

Justification for Inclusion in ePortfolio

This artifact represents a well-rounded demonstration of software engineering in the context of game development. I selected it for my ePortfolio because it transitions from a static academic visualization to an actively playable and extendable product that integrates real-time systems, modular design, and algorithmic complexity.

Key features that highlight my skillset include:

- Procedural Terrain Engine: Implements OpenSimplex noise to generate heightmaps and classify tile types by biome procedurally.

- Modular Architecture: Each system—input, rendering, world generation, camera—is isolated and testable.
- Chunked World System: World data is generated and rendered in memory-efficient chunks, preparing the game for future scaling and multiplayer support.
- Code Readability: Thoughtful method naming, in-line documentation and reusable helper functions reflect professional-level design patterns.

The artifact was significantly improved over the original by introducing the procedural world generation system, converting the logic from C++ to idiomatic Java, and designing the project for extendibility rather than hardcoded behavior.

Outcome Alignment

This enhancement supports the following Computer Science program outcomes:

- Outcome 1: The game's modular design facilitates collaboration by separating systems (e.g., camera, tile manager, input) cleanly across packages and classes.
- Outcome 2: I produced technical documentation (comments and structure) to ensure the code is understandable and maintainable.
- Outcome 3: Procedural generation and biome logic use algorithmic principles such as noise functions, conditional thresholds, and runtime efficiency management.
- Outcome 4: I demonstrate proficiency with professional tools and libraries, such as libGDX, Gradle, IntelliJ, and Java Collections.

At this point, I am on track with all planned initial outcomes. This enhancement provided full coverage of Outcomes 1, 2, 3, and 4. Outcome 5 will be addressed in the next phase, where player data persistence and database security will be added using MongoDB.

Reflection on the Enhancement Process

Enhancing this artifact taught me how to think about systems at scale. Transitioning from a hardcoded 3D demo to a procedurally generated, infinite world involved significant architectural planning. I learned how to apply noise algorithms to simulate organic terrain and how to layer additional features like biome types over raw height data.

I faced early challenges adapting from C++'s immediate-mode OpenGL model to Java's object-oriented and framework-driven style. Working with libGDX also required learning how to batch render efficiently and manage camera views in a 2D environment. Another challenge was balancing performance with flexibility, particularly when generating and caching terrain on the fly.

Through trial and refactoring, I implemented a chunk system that balances performance (by only generating visible terrain) with realism (via seamless biome blending). I also learned the value of separating rendering from logic to enable unit testing and easier bug tracing.

Final Thoughts

This enhancement stands as a milestone in my academic and professional development. It bridges graphics, procedural algorithms, and engine architecture—all within a context that is both creative and technically rigorous. I'm proud of the transformation this project has undergone, and it now serves as a strong software engineering example in my ePortfolio.

As I move into the next enhancement, I plan to build on this structure by integrating MongoDB for saving player progress and analytics and adding gameplay features such as crafting, health systems, and AI.