CSC 481/581 - Milestone 4

Overview

Your task for this assignment is to add more functionality to your engine. You will submit the code for your engine and an individual game, along with a detailed write-up of your results, including screenshots. As always, you are expected to abide by the University's Academic Integrity Policy, which includes providing appropriate attribution for all external sources of information consulted.

The assignment is graded out of 100% and is broken down as follows:

- **Part 1: Engine Features:** 25%
- **Part 2: Individual Game Submission:** 25%
- **Part 3: Individual Write-up:** 50%

Part 1: Engine Features

This portion of the assignment focuses on the features built by your team for the core game engine.

1.1: Event Management System (12.5 pts)

You are tasked with adding an event management system to your game engine. You must queue events and allow for different priorities. You must provide solutions to the four design decisions of event management:

1. **Event Representation**
2. **Event Registration**
3. **Event Raising**
4. **Event Handling**

You will need to write an event manager class to keep track of which engine systems or game objects are registered to receive events of arbitrary types (registration), receive events after they have occurred (raising), and dispatch those events to the appropriate objects at the appropriate time (handling). You should use your Timeline class for creating timestamps for event priorities.

To demonstrate your system, create at least four types of events: **collision, death, spawn, and input**. These must be managed entirely by your new system, not SDL events.

1.2: Networked Event Management (12.5 pts)

Your task is to ensure the event handling can occur in a networked environment. You can choose a server-centric model, a distributed model, or a hybrid of the two. Events must be shared across the network. It is acceptable to serialize or reconstruct event data using a protocol of your choice (e.g., JSON, or a custom struct).

You must demonstrate at least one example of events being sent and handled across the network.

1.3: Replay System (6 pts, Required for 581, Optional for 481)

Build on your event system to implement a replay system. The replay system should expand on the event system adding new events which start recording the replay, ending the replay recording, as well as controlling replay playback.

Part 2: Individual Game Submission

For this part, you will individually create or modify a game that showcases the engine features developed in Part 1. Your submission will be graded on how effectively and creatively you use the event management system, networking, and (if applicable) input chords.

Your game must clearly demonstrate:

- The four required event types (collision, death, spawn, input) being used to create gameplay. (10)
- A networked interaction that relies on your event system. (10)
- A unique application of the engine that is distinct from your teammates' submissions. (5)
- (581 required) An implementation of the replay system which either uses an in-game trigger to begin the replay (such as player-controlled character death event) or manually start/stop recording and begin playback. (6)

Part 3: Individual Write-up (50%)

For 50% of your assignment credit, you will submit an individual write-up. This paper is a critical reflection on your design decisions and implementation. It is strongly suggested that you go beyond simply answering the questions and think deeply about your work.

Your write-up must include the following sections:

3.1: Engine Design and Rationale

- **Event Management:** How and why did you represent events the way you did? Discuss your solutions for event registration, raising, and handling. What design changes did you need to make to your existing engine? How does your design promote reuse and extensibility?

- **Network Architecture:** Explain your approach to networked event handling. Why did you choose a server-centric, distributed, or hybrid model? Discuss the interaction between your network architecture and your event manager.
- **Design Philosophy:** What were the guiding principles behind your team's design decisions? What future capabilities do these decisions enable?

3.2: Individual Game Implementation

- **Showcasing Features:** Describe how your individual game demonstrates each of the required engine features. Explain the specific gameplay mechanics you built around the event system.
- **Screenshots:** Include an appendix with relevant screenshots from your game. You must reference these screenshots in the body of your write-up to support your points (e.g., "Figure 1 shows the spawn event in action..."). Screenshots do not count towards the page requirement.

3.3: Reflection and Analysis

- **Successes and Challenges:** What worked the way you wanted, and what didn't? What was the most challenging part of this assignment and how did you overcome it?
- **Critical Evaluation:** If you could start over, what would you do differently? What are the limitations of your current system? Successful write-ups will contain evidence that you have thought deeply about your decisions and their consequences.

Submission

There are three deliverables for this assignment:

1. **Team Engine Submission:** A single submission for your team that includes the source code for the engine features from Part 1.
2. **Individual Game Submission:** Your personal game project that uses the engine, as described in Part 2. This is graded individually.
3. **Individual Write-up:** Your individual reflection paper, as detailed in Part 3.