Functional Programming and Secure Web Development: A Case Study in Game Design

John Orland
CPSC 490 Project Proposal
Advisor: Ruzica Piskac

Overview

Many servers, databases and native software packages are developed with functional programming languages; they offer superior runtime safety, strong support for concurrency, and stateless design patterns that can reduce code complexity and enhance software security. However, game developers cling to C++, deferring to imperative design standards and existing software infrastructures. Similarly, popular web frameworks favor imperative code and HTTP architectures, and accept low standards for performance and user experience. By designing and implementing a multiplayer game using a functional programming language and web sockets, I will research the advantages of functional programming to both game and web development while exploring “single page app” performance boundaries. I hope to reveal ways that functional programming can enhance performance and simplify a programming genre that is notorious for unmanageable state.

Objectives

1. Demonstrate that game development in a functional style is feasible by discovering methods and data structures which rival imperative counterparts in performance
2. Ascertain usefulness of functional programming in managing software and network security risks associated with asynchronous interactivity
3. Promote openness toward different paradigms and technologies with an application example that is exciting to a broad audience

Proposal

I will design and produce server code for a multiplayer roleplaying game using a modern functional language. This will involve reconsidering the classic imperative game loop which acts upon numerous mutable objects (players, non-player characters, the environment, etc.) in favor of functional methods and immutable records.
My goal is that this alternative will naturally support asynchronous event handling (input from many users on the network), and be capable of supporting low latency interactions between players. The bulk of my work will comprise this design and optimization process, rather than fleshing out advanced gameplay features. Therefore, my project will be learning and research intensive; the game will be a framework, not a complete entertainment product. It is unclear how effectively my client/server will respond to latency, packet loss, and other performance variables associated with web sockets, but this is an area that I will explore during the optimization and testing phase.

The game will allow players to navigate and interact within a 2D environment. At minimum, gameplay will consist of simple combat between players and non-players in the style of the classic Atari “Asteroids” game. The client will be simple and imperative; a JavaScript front end that receives state updates via web sockets, and renders sprite-based graphics with HTML5. The multiplayer server should be able to run constantly, with clients able to connect and disconnect without interrupting gameplay.

**Summary of Deliverables:**

1. F# server code and JavaScript client (git repository and test deployment)
2. Blog posts describing the development process
3. Final Report

**Execution Strategy:**

Stage 1 – February 19th

- Develop skeletal server and client code to handle connections, test WebSockets, and to confirm the stability of a hosting solution
- Explore F# language features and research existing applications of F# in web and game development
- Conduct general research of game design principles, understand multiplayer network challenges

Stage 1 – March 26th (soft), April 16th

- Fully prototype game code, including interactivity and network fault tolerance, by soft deadline.
- Optimize performance, take more implementation risks in project branches
- Improve client and user experience, add extras if there is time (ex. add user login, chat, record player stats in database, investigate scaling capability)
Stage 3 – May 5th

- Produce final report
- Produce final blog posts
- Continue testing
- Launch the game in an open source community