JavaScript + HTML5

**BlockBreaker Supreme!**

Recreating the classic blockbreaker game using JavaScript and HTML5

Devin Piner

May 29, 2018

**Introduction**

In this report, I will describe the development process

**Project Analysis and Discussion**

Architecture

Early in the development of BlockBreaker, when the systems in place were relatively simple, I was able to get away with a sequential-style architecture. That is, almost everything I had implemented was in one file. As the project grew, I quickly realized that components needed to be abstracted to easily maintain and grow the game. One of the abstractions I needed to make was , controlled from the main loop of. This allowed me to quickly and easily add new sections to the game without changing large portions of the code in the main loop. I will discuss these abstractions further next.

GAME\_AREA

The (GA) is the main driver of the entire game. It is initiated by the function which is itself called on the event of the body. The main objective of the GA is to perform the initial setup of the game, such as giving game scenes access to canvas width and height, as well as defining the canvas boundaries and whether the game ball has collided with them. Furthermore, the GA provides event listeners for useful game-related action from the user, such as clicking and button presses. The event listeners forward the event details to the currently active scene defined by where it can be handled appropriately (we will discuss this further later). Most importantly, the GA defines the canvas refresh rate (fps) – the number of times the canvas is redrawn every second. The fps interval implements the main game loop in the form of a function named called *fps* times per second.

Scenes

These objects contain all information needed for each different section of the game. I implemented a scene for each menu – the main menu, the pause menu, level selection, etc. – and a scene where you play the game. Each scene contained several generic functions that could be controlled from the main game loop.