ATARI BREAKOUT GAME Development

This blog documents the process of developing a replica of famous ATARI Breakout Game. It is a simple 2-D game built using HTML 5 Canvas, JavaScript and Howler JS. The development process was divided into different sections which made the project easier to complete.

Styles.css

This file defines the basic layout of the game. It adds features like positioning, hovering, etc.

JavaScript Files.

a) View.js

This file defines most of the graphics of the game.

- Ball() and Paddle()

We first set the canvas height and width which would be the playing space. Then, we set the position of the ball and paddle. The position of the ball was set halfway through the x axis and 50 px above the bottom of the canvas. Similarly, the radius of the ball was set to 7, and the dx and dy represent the shift of the ball along the x-axis and y-axis. For the paddle, we fix the height and width and then set the x-axis position.

- generateBricks()

Once the ball and paddle are made, we first declare some block scope variables, and give them values. We then make an empty array called bricks, and then declare a function, generateBricks(). The function creates a two-dimensional bricks array and sets the status to 1. When the status is 1, the function will generate bricks, and when the status is 0, the bricks will not be generated.

- drawbricks()

  This function defines the formula to set the bricks on the x-axis and the y-axis. Once the position of the bricks is set, we create rectangles with bricks on x-axis, y-axis and also set their width and height on the new path we started. We also fill the color of the bricks and close the path.

- drawScore()

  This function fills the canvas with text at the top right corner. The text gives us the current score of the player.

- collisionDetection()

  This function checks if the ball hits one of the bricks. It checks if the position of the ball in the x-direction is in between the width of the brick. At the same time, it also checks if the position of the ball in the y-direction is in between height of the brick. If

the condition is satisfied, we change the direction of the ball, and then set the brick status to 0, which will remove the brick from the canvas.

b) controls.js

This file describes the controls of the user in the game.

- keyDownHandler(e) and keyUpHandler(e)
  We first create two variables rightPressed and leftPressed which represent our arrow keys. These variables are initially set to false. Then we add event listener which will listen for key down and triggers our functions keyDownHandler() and keyUpHandler(). If the left and right keys are pressed, the keyDownHandler() will set the variables to true and if the keys are released or not pressed, the keyUpHandler() will again set the variables to false.

- movepaddle()
  This function defines how the paddle will move. If we press the right or left keys, the paddle is moved by 7 pixels. Similarly, we don't want our paddle to move more than the width of the canvas, so we limit its x-position to canvas width.

c) data.js

This file describes how the score and high score data are saved in the game.

- resetScore()
  This method will reset the score of the game if the ball touches the bottom. If the combined radius of the

d) model.js

- bounce()
  This function defines how the ball will bounce off the paddle. If the ball is in the same x and y position as the paddle, the direction of the ball on the y-axis will be inverted.

- levelUp()
  This function defines the condition for changing the level in the game. If the score reaches a multiple of 15, we will regenerate the bricks to the canvas. We will also increase the speed by 2.5 for every time a player's score reaches a multiple of 15.

- play()
  The play function simply calls the various other functions we created in different files so that they all can work together to play the game.

e) Howler.js Library

To add music for the game, an online JavaScript library called Howler JS was used. Howler enables the use of simple prebuilt functions that can be easily followed to add sound effects into your website.