6 January 2025

Project 9 Submission: "Bricks Breaker"

Names and Emails:

Ohad Cohen, ohad.cohen02@post.runi.ac.il
Raz Bouganim, raz.bouganim@post.runi.ac.il

Concept / Idea of the game:

"Bricks Breaker" is a classic arcade-style game which challenges players to control a paddle at the bottom of the screen, using it to bounce a ball and break a set of bricks arranged at the top. The objective is to clear all the bricks by hitting them with the ball while preventing the ball from falling off the screen.

Motivation:

Our motivation to build "Bricks Breaker" comes from our passion for gaming and ambition to explore the gaming industry. Developing this game was an opportunity to combine technical skills with our love for gaming, resulting in a project that is not only fun to play but also challenging to build. Our goal was to create something we could genuinely be proud of, and even to showcase on our resumes as a testament to our dedication and creativity.

Google Drive Link:

https://drive.google.com/file/d/1yZ-fY6JsIiFqAl8IBtel_3Cp57XJfCBq/view?usp=sharing

6 January 2025

Architecture:

1. Main.jack:

   Purpose: Acts as the entry point and game manager. It initializes the game components, handles the main game loop, and manages game states (win/lose conditions).

   Main Logic:

   Initializes Paddle, Ball, and BrickLine objects.

   Implements the game loop to:

   - Update the position of the paddle and ball.
   - Detect collisions between the ball and the paddle, bricks, and screen boundaries.
   - Check for win or lose conditions and handle corresponding game states.

   Displays the score.

   Handles user input for restarting the game.

2. Paddle.jack:

   Purpose: Represents the paddle controlled by the player.

   Main Logic:

   Maintains the paddle's position and dimensions (x, y, width, height).

   Updates its position based on user input (left and right arrow keys).

   Draws and erases itself on the screen using the Screen class.

3. Ball.jack:

   Purpose: Represents the ball in the game.

   Main Logic:

   Maintains the ball's position (x, y) and movement direction (dx, dy).

   Moves the ball based on its current direction, bouncing off screen boundaries by reversing direction as needed.

   Detects if the ball has fallen off the screen (lose condition).

   Handles drawing and erasing itself on the screen.

4. Brick.jack:

   Purpose: Represents an individual brick in the game.

   Main Logic:

   Maintains its position (x, y), dimensions (width, height), and visibility status.

   Draws and erases itself on the screen.

   Changes visibility to false when hit by the ball, effectively "destroying" the brick.

5. BrickLines.jack:

   Purpose: Manages a grid of bricks.

   Main Logic:

   Creates a grid of Brick objects based on specified rows and columns.

   Draws all visible bricks on the screen.

   Provides methods to retrieve individual bricks and check if all bricks have been destroyed (win condition).