Project Report
Web Coding
Minesweeper Cricket

Mayank Kumar

June 14, 2023

Contents

1 Introduction

In this report, I will describe the code implementation for the Minesweeper Cricket game. The game combines elements of Minesweeper and Cricket, where the player needs to reveal blocks on a grid and avoid fielders. The objective is to score runs by revealing safe blocks. The implementation involves HTML, CSS, and JavaScript to create the game logic and user interface.

2 Design and Implementation

The game is designed using HTML and CSS for the user interface, and JavaScript for the game logic. The main HTML structure includes a title, game settings, score display, game board, and script reference.

The CSS code is responsible for styling the game elements, including the background image, font, colors, and grid layout.

The JavaScript code handles the game functionality. It initializes variables, creates the game board, distributes fielders and power-ups randomly, and handles block clicks to reveal the content. It also manages the scoring and game-over conditions.

3 Customization

To enhance the game experience, several customizations can be implemented. These include:

- Allowing the player to choose different grid sizes. The player can choose whether to play on a 6*6 board or a 7*7 board.
- Points Galore PowerUp1 - If the player clicks on this power up block any random points between 1 to 3 are added to the total score.
- Who is there? PowerUp2 - On clicking this power up, all the blocks of the game board are revealed for a time period of 1.5 seconds.

4 Code Overview

4.1 HTML Code

The HTML code represents the structure and layout of the Minesweeper Cricket game interface. Let's break down the different components:

1. `<!DOCTYPE html>`: This is the document type declaration, specifying that the document is an HTML5 document.
2. `<html>`: The root element of the HTML document.
3. `<head>`: Contains meta-information about the document, such as the title and external CSS file link.
4. `<link rel="stylesheet" type="text/css" href="style.css">`: This line links the external CSS file named "style.css" to the HTML document, allowing the styling rules defined in the CSS file to be applied to the HTML elements.
5. `<body>`: Contains the visible content of the web page.
6. `<h1>`: Displays the heading "Minesweeper Cricket" at the top of the page.

7. `<div class="settings">`: Represents a container for the game settings elements.
    - `<label for="gridSize">`: Displays the label "Grid Size:".
    - `<select id="gridSize">`: Creates a dropdown menu for selecting the grid size.
    - `<option value="6">6x6</option>`: Represents an option in the dropdown menu with a value of 6, indicating a 6x6 grid size.
    - `<option value="7">7x7</option>`: Represents an option in the dropdown menu with a value of 7, indicating a 7x7 grid size.
    - `<button id="startBtn">Start Game</button>`: Creates a button with the ID "startBtn" to start the game.
8. `<div id="scoreboard">`: Displays the current score of the player.

    - "Score: ": Text indicating the purpose of the element. - ‘¡span id="score"¿0¡/span¿‘: Represents a placeholder for the score value. Initially set to 0.
9. `<div id="gameboard"></div>`: Represents the game board container where the blocks and gameplay will be displayed.
10. `<script src="script.js"></script>`: Loads the external JavaScript file named "script.js" into the HTML document, which contains the game logic and functionality.

This HTML code sets up the basic structure and elements of the Minesweeper Cricket game, including the game title, game settings, score display, and the game board container. The CSS file is linked to provide styling rules for the elements, while the JavaScript file is included to handle the game logic and user interactions.

4.2 Java Script Code

1. Event Listener:

    document.addEventListener("DOMContentLoaded", function () ...);: Listens for the "DOMContentLoaded" event, which indicates that the HTML document has been completely loaded and parsed. This ensures that the JavaScript code executes after the HTML content is ready.
2. Variable Declarations:
    - const gameboard = document.getElementById("gameboard");: Retrieves the gameboard element from the HTML document.
    - const scoreElement = document.getElementById("score");: Retrieves the score element from the HTML document.
    - const startBtn = document.getElementById("startBtn");: Retrieves the start button element from the HTML document.
    - const gridSizeSelect = document.getElementById("gridSize");: Retrieves the grid size select element from the HTML document.
    - let gridSize = parseInt(gridSizeSelect.value);: Initializes the gridSize variable with the selected grid size value converted to an integer.
    - let fielderCount = 11;: Initializes the fielderCount variable with the number of fielders.
    - let score = 0;: Initializes the score variable to keep track of the player's score.
3. Function Declarations:
    - createGameboard(): Creates the gameboard grid dynamically based on the selected grid size.

- getRandomInt(max): Generates a random integer between 0 and the specified max value.
- distributeFielders(): Randomly distributes the fielders on the gameboard.
- distributePowerUp(): Randomly distributes Points Galore power up on the gameboard.
- revealBlock(row, col): Handles the block click event and reveals the clicked block based on its content (fielder, power-ups, or empty block).
- revealAllBlocks(): Reveals all the blocks on the gameboard, highlighting the fielders and power-ups.
- undoReveal() : To hide the blocks which have been revealed by the Who is there? power up after a specified interval.
- endGame(): Displays a game over message with the final score and resets the game.
- resetGame(): Resets the game by resetting the score, creating a new gameboard, and distributing fielders and power-ups.

The functions which are used to reveal or hide a block make use of the classList by including or excluding a particular event for which the graphics have been included in the CSS file. The number of power ups in each game is also random because the function which distributes the second power up can also allot the second power up to the blocks on which the first power up has been already alloted.

4. Event Listeners:
   - startBtn.addEventListener("click", function () ...);: Listens for the click event on the start button and triggers the resetGame function.
   - block.addEventListener("click", function () ...);: Listens for the click event on each block element and triggers the revealBlock function.
5. Initialization:
   - createGameboard(): Initializes the gameboard by calling the createGameboard function.
   - distributeFielders(): Distributes the fielders on the gameboard by calling the distributeFielders function.
   - distributePowerUp(): Distributes the power-ups on the gameboard by calling the distributePowerUp function.

4.3 CSS Code

The CSS code defines the styles and visual appearance of the Minesweeper Cricket game interface. Let's break down the different CSS rules:

1. body: Applies styling rules to the ‘¡body¿‘ element.
   - `background-image: url('BackImage2.jpg');`: Sets the background image of the body to "BackImage2.jpg".
   - ‘background-repeat: no-repeat;‘: Specifies that the background image should not repeat.
   - ‘background-size: 100
   - ‘background-color: ;‘: Sets the background color of the body to a light gray.
   - ‘font-family: Arial, sans-serif;‘: Specifies the font family to be used for the text content.
2. ‘h1‘: Applies styling rules to the ‘¡h1¿‘ element.

3. ‘.settings‘: Applies styling rules to the elements with the class ”settings”.
    - text-align: center;: Centers the text within the container.
    - margin-bottom: 20px; : Adds a bottom margin of 20 pixels to create spacing.
4. ‘scoreboard‘: Applies styling rules to the element with the ID ”scoreboard”.
    - ‘text-align: center;‘: Centers the text within the element.
    - ‘font-size: 24px;‘: Sets the font size to 24 pixels.
    - ‘margin-bottom: 20px;‘: Adds a bottom margin of 20 pixels to create spacing.
5. ‘gameboard‘: Applies styling rules to the element with the ID ”gameboard”.
    - ‘width: 400px;‘: Sets the width of the game board container to 400 pixels.
    - ‘margin: 0 auto;‘: Centers the game board horizontally within its parent container.
    - ‘display: grid;‘: Uses CSS Grid layout to arrange the blocks within the game board.
6. ‘.block‘: Applies styling rules to elements with the class ”block”.
    - ‘border: 1px solid ;‘: Sets a 1-pixel solid border with a black color.
    - ‘padding: 10px;‘: Adds a 10-pixel padding around the content within the block.
    - ‘text-align: center;‘: Centers the text within the block.
    - ‘font-size: 18px;‘: Sets the font size to 18 pixels.
    - ‘cursor: pointer;‘: Changes the mouse cursor to a pointer when hovering over the block.
    - ‘background-color: ;‘: Sets the background color of the block to white.
    - ‘transition: background-color 0.2s;‘: Specifies a smooth transition effect when changing the background color.
7. .block.className: Applies styling rules to blocks with the class ”className”.

    background-color: ;: Sets the background color of the clicked block to specified colour

5 Results and Discussion

Upon running the code, the Minesweeper Cricket game is displayed with the specified customizations. The player can interact with the game board by clicking on the blocks to reveal their content. The score is updated accordingly, and the game ends when a fielder is revealed.

During testing, I observed that the game provides an engaging experience, combining the strategic element of Minesweeper with the excitement of Cricket. The customization options allow for different gameplay variations and increased replay value.

6 Conclusion

The Minesweeper Cricket game implementation successfully combines the mechanics of Minesweeper and Cricket, creating an enjoyable and interactive experience. The HTML, CSS, and JavaScript code work together to provide the game logic and user interface. The customizations further enhance the gameplay, adding variety and challenges. Future enhancements could include additional features and game modes to further enrich the player experience.

References

[Aro] Nick Arocho. Minesweeper. https://www.nickarocho.com/projects-index/minesweeper.

[Ell] Danielle Ellis. Waitfunction.js. https://blog.hubspot.com/website/javascript-wait.

[Far] Faraz. Minesweeper. https://www.codewithfaraz.com/content/134/learn-how-to-create-a-minesweeper-game-with-html-css-and-javascript.

[LT] LaTeX-Tutorial. Bibliography. https://latex-tutorial.com/tutorials/bibtex/.

[oL] Royal Holloway University of London. Bibliography. https://libguides.rhul.ac.uk/referencing/latex.

[T] The Modern Java Script T. Js. https://javascript.info/.

[T] [Far] [Aro] [Ell] [oL] [LT]