Desenvolvimento de uma Variação do Jogo 'Snake' Utilizando Tecnologias Web: HTML, CSS e JavaScript

Thiago Minuzzi Batista[1], Tandson Alves de Azevedo Filho [2], Elaine da Cunha Silva Paz[3]

[1]Estudante do Curso Técnico em Informática para Internet Integrado ao Ensino Médio – IFTO. e-mail: thiago.batista4@estudante.ifto.edu.br

[2]Estudante do Curso Técnico em Informática para Internet Integrado ao Ensino Médio – IFTO. e-mail: tandson.filho@estudante.ifto.edu.br

[3]Docente de Química – IFTO. Orientador(a) e-mail: elaine@ifto.edu.br;

1 INTRODUÇÃO

A evolução das tecnologias de desenvolvimento web tem permitido a criação de jogos diretamente nos navegadores, proporcionando experiências interativas e visuais inovadoras. O jogo da cobrinha, um clássico dos jogos eletrônicos, é amplamente reconhecido por sua simplicidade e jogabilidade envolvente. Este projeto tem como objetivo desenvolver uma versão moderna do jogo da cobrinha que incorpora elementos gráficos tridimensionais para enriquecer a experiência do usuário, utilizando tecnologias web padrão, como HTML5, CSS e JavaScript. A adição de um fundo animado em 3D, criado com a biblioteca Three.js, visa tornar o jogo mais atraente visualmente, proporcionando um ambiente dinâmico e interativo.

2 JUSTIFICATIVA

A implementação de jogos clássicos com recursos gráficos modernos é uma tendência que visa aumentar o engajamento e a satisfação dos usuários. Integrar um fundo animado em 3D a um jogo de interface tradicional contribui para uma experiência mais imersiva, aumentando o interesse dos jogadores e destacando-se entre os jogos convencionais baseados em navegador. Além disso, este projeto demonstra como utilizar técnicas avançadas de renderização gráfica com bibliotecas JavaScript como Three.js, promovendo o aprendizado e a aplicação prática de conceitos de desenvolvimento de jogos e gráficos computacionais na web.

3 CONCEITO TÉCNICO

A aplicação é composta por três componentes principais: o HTML e CSS, que definem a estrutura e o estilo da interface do usuário, incluindo os elementos para o jogo e a lista de pontuações; o JavaScript, que controla a lógica do jogo da cobrinha, como o movimento da cobra, a detecção de colisões e o sistema de pontuação, manipulando diretamente o elemento canvas para desenhar o estado do jogo em cada iteração; e a biblioteca Three.js, utilizada para criar um fundo dinâmico em 3D, renderizado em um segundo elemento canvas, utilizando objetos tridimensionais básicos e animações de rotação, com o objetivo de criar um ambiente visualmente interessante e interativo.

4 CÓDIGOS FONTES

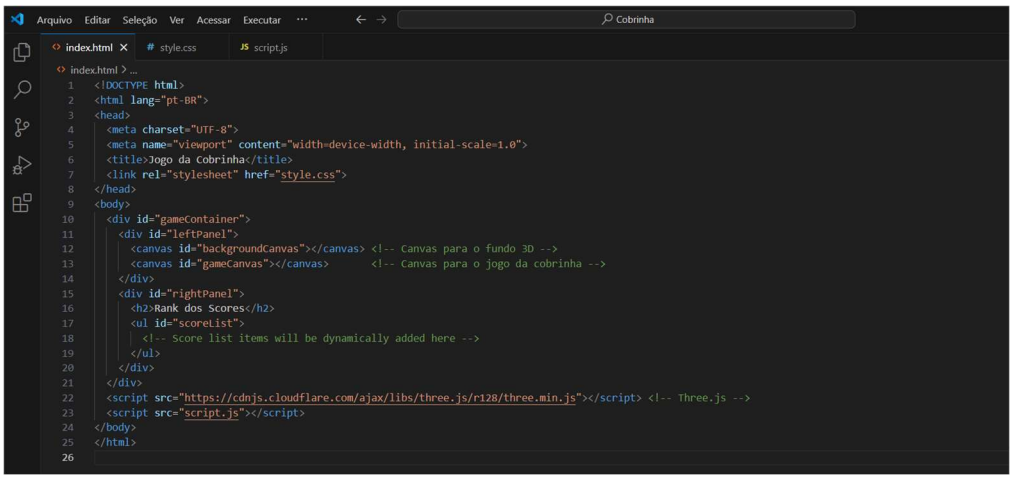
O código começa criando a estrutura básica do jogo utilizando HTML e CSS. O HTML define dois elementos canvas: um para a área de jogo (gameCanvas) e outro para o fundo animado em 3D (backgroundCanvas). O layout é dividido em dois painéis: o painel esquerdo contém os elementos canvas para o jogo e a animação de fundo, enquanto o painel direito exibe a lista de pontuações. O CSS estiliza esses elementos, proporcionando uma aparência moderna, com um esquema de cores escuras e detalhes em verde neon para destacar o jogo e a lista de pontuações, garantindo que a interface seja visualmente atraente e funcional.

A lógica principal do jogo da cobrinha é implementada em JavaScript. A cobra é representada como um array de objetos, onde cada objeto corresponde a um segmento da cobra. A função gameLoop é executada a cada 100 milissegundos, atualizando a posição da cobra com base na direção atual e redesenhando a tela. A comida é gerada em posições aleatórias no grid e, quando a cobra a consome, a pontuação aumenta, um som é tocado, e a comida é reposicionada aleatoriamente. Caso a cobra colida com as bordas do canvas ou com seu próprio corpo, o jogo termina, e o jogador é solicitado a inserir seu nome, que é adicionado à lista de pontuações classificadas. Esse gerenciamento de estado permite uma jogabilidade fluida e interativa.

Para o fundo animado em 3D, o código utiliza a biblioteca Three.js, que cria uma cena tridimensional com um cubo rotativo. A cena é inicializada com a função init3DBackground, que configura a câmera, o renderizador e o objeto 3D (cubo). A animação do fundo é gerada por meio de um loop de animação contínuo na função animate3DBackground, que rotaciona o cubo e renderiza a cena a cada frame. Esta animação de fundo adiciona uma camada visual dinâmica ao jogo, tornando-o mais envolvente e atraente para o jogador.

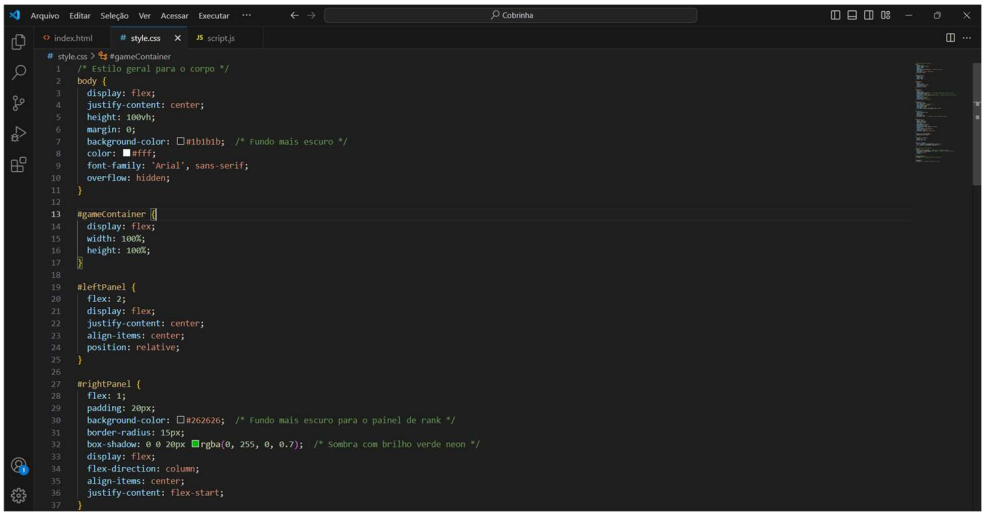
5 RESULTADOS E DISCUSSÃO

Figura 1 – Código index.html



Fonte: Autores

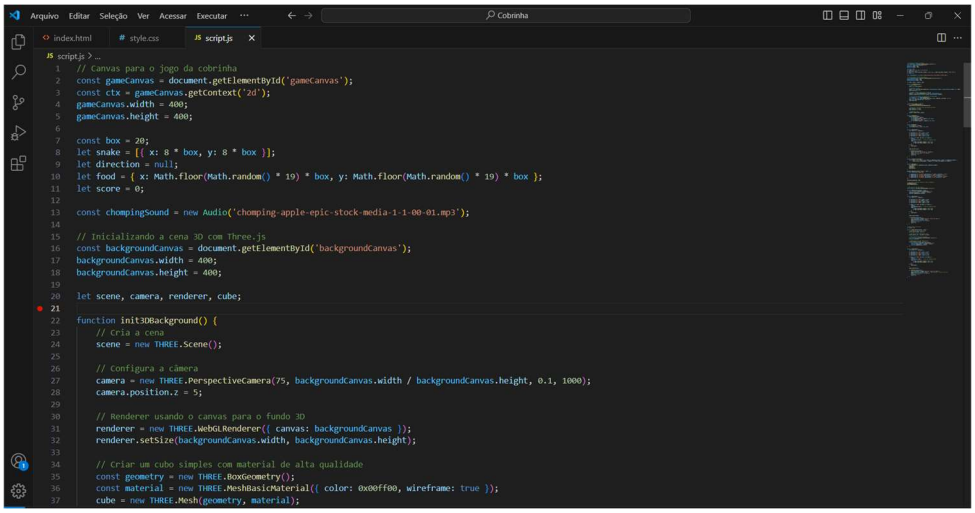
Figura 2 – Código style.css



```css
/* Estilo geral para o corpo */
body {
  display: flex;
  justify-content: center;
  height: 100vh;
  margin: 0;
  background-color: #1b1b1b;  /* Fundo mais escuro */
  color: #fff;
  font-family: 'Arial', sans-serif;
  overflow: hidden;
}

#gameContainer {
  display: flex;
  width: 100%;
  height: 100%;
}

#leftPanel {
  flex: 2;
  display: flex;
  justify-content: center;
  align-items: center;
  position: relative;
}

#rightPanel {
  flex: 1;
  padding: 20px;
  background-color: #262626;  /* Fundo mais escuro para o painel de rank */
  border-radius: 15px;
  box-shadow: 0 0 20px rgba(0, 255, 0, 0.7);  /* Sombra com brilho verde neon */
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: flex-start;
}
```

Fontes: Autores

Figura 3 – Código script.js



```javascript
// Canvas para o jogo da cobrinha
const gameCanvas = document.getElementById('gameCanvas');
const ctx = gameCanvas.getContext('2d');
gameCanvas.width = 400;
gameCanvas.height = 400;

const box = 20;
let snake = [{ x: 8 * box, y: 8 * box }];
let direction = null;
let food = { x: Math.floor(Math.random() * 19) * box, y: Math.floor(Math.random() * 19) * box };
let score = 0;

const chompingSound = new Audio('chomping-apple-epic-stock-media-1-1-00-01.mp3');

// Inicializando a cena 3D com Three.js
const backgroundCanvas = document.getElementById('backgroundCanvas');
backgroundCanvas.width = 400;
backgroundCanvas.height = 400;

let scene, camera, renderer, cube;

function init3DBackground() {
    // Cria a cena
    scene = new THREE.Scene();

    // Configura a câmera
    camera = new THREE.PerspectiveCamera(75, backgroundCanvas.width / backgroundCanvas.height, 0.1, 1000);
    camera.position.z = 5;

    // Renderer usando o canvas para o fundo 3D
    renderer = new THREE.WebGLRenderer({ canvas: backgroundCanvas });
    renderer.setSize(backgroundCanvas.width, backgroundCanvas.height);

    // Criar um cubo simples com material de alta qualidade
    const geometry = new THREE.BoxGeometry();
    const material = new THREE.MeshBasicMaterial({ color: 0x00ff00, wireframe: true });
    cube = new THREE.Mesh(geometry, material);
```

Fonte: Autores

Figura 4 – Código funcionando

Fonte: Autores

6 CONSIDERAÇÕES FINAIS

Este projeto demonstra como diferentes tecnologias web podem ser combinadas para criar uma aplicação rica em recursos e visualmente atrativa. Ao integrar HTML, CSS, JavaScript e Three.js, a aplicação não só oferece uma experiência de jogo envolvente, mas também explora o potencial das tecnologias web para criar interfaces dinâmicas e interativas.

AGRADECIMENTOS

Agradecemos ao CNPq e ao IFTO pelo fomento e apoio para a execução do projeto que possibilitou a realização desta pesquisa.

REFERÊNCIAS

Mozilla Developer Network (MDN). "Canvas API." Disponível em: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API. Acesso em: 8 de setembro de 2024.

Three.js Documentation. "Introduction to Three.js." Disponível em: https://threejs.org/docs/. Acesso em: 8 de setembro de 2024.

W3Schools. "HTML5 Canvas." Disponível em: https://www.w3schools.com/html/html5_canvas.asp. Acesso em: 8 de setembro de 2024.

Khronos Group. "WebGL Overview." Disponível em: https://www.khronos.org/webgl/. Acesso em: 8 de setembro de 2024.