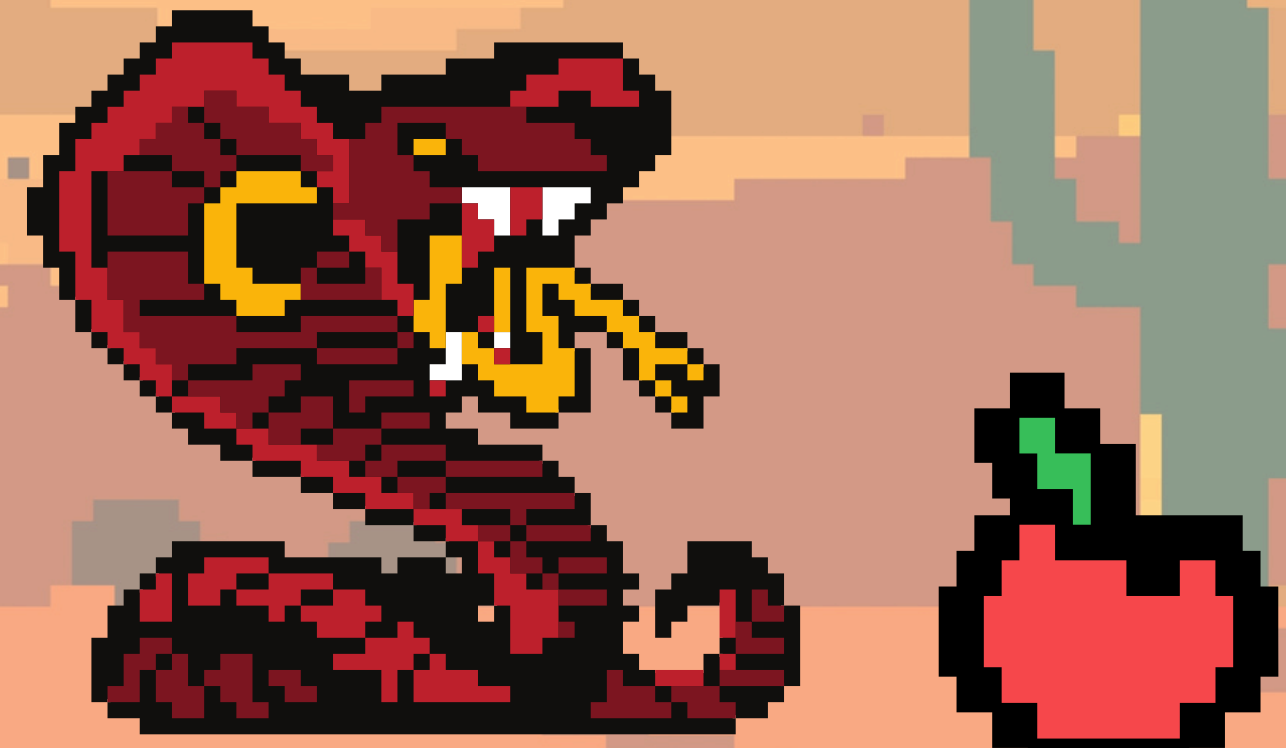
SNAKE

Manual y juego elaborado por:
Derek Revilla
Diseño y maquetación:
Evelyn Paez



Manual del Juego "SnakeX"

Introducción

El juego SnakeX es una versión moderna del clásico “Snake” o “La Serpiente”, desarrollado completamente en HTML, CSS y JavaScript.

En este juego, el jugador controla una serpiente que se mueve por un tablero en busca de comida.

Cada vez que come, su cuerpo crece y su puntuación aumenta.

El juego termina cuando la serpiente choca contra las paredes o contra sí misma.

Este manual detalla cada sección del código, explicando su utilidad y funcionamiento.

Documentos utilizados

Este proyecto está formado por un solo documento HTML que incluye el código HTML, CSS (diseño visual) y JavaScript (lógica del juego) dentro del mismo archivo:

1. HTML: estructura del juego y elementos visibles.
2. CSS: estilos visuales, colores, tamaños, animaciones y diseño.
3. JavaScript: controla el movimiento de la serpiente, la comida, la puntuación y las reglas del juego.

1. Estructura HTML (Interfaz del juego)

El código HTML define la estructura del juego y sus elementos visuales.

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>SnakeX</title>
```

- <!DOCTYPE html> indica que el documento está en formato HTML5.
- <html lang="es"> define el idioma español.
- <meta charset="UTF-8"> permite usar caracteres especiales (acentos, ñ, etc.).
- <meta name="viewport"...> hace que el juego sea visible correctamente en móviles.
- <title> define el nombre que aparece en la pestaña del navegador.

Cuerpo del juego

```html
<body>
    <div id="game-container">
        <h1>SnakeX</h1>
        <canvas id="game" width="400" height="400"></canvas>
        <div id="score-board">
            Puntaje: <span id="score">0</span>
        </div>
        <div class="message" id="message"></div>
        <button class="button" onclick="resetGame()">Jugar de nuevo</button>
    </div>
</body>
```

- : Contenedor principal del juego.
- <h1>: Título “SnakeX”.
- <canvas>: Lienzo donde se dibuja la serpiente y la comida.
- : Muestra el puntaje actual.
- : Se actualiza dinámicamente con JavaScript.
- : Muestra mensajes como “¡Juego terminado!”.
- <button>: Reinicia la partida llamando a la función resetGame().

2. Código CSS (Diseño y estilo visual)

El bloque <style> contiene todo el diseño gráfico del juego.

```css
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
```

Sirve para: Eliminar márgenes y paddings predeterminados, y hacer que los tamaños sean más precisos.

```css
body {
    background: linear-gradient(135deg, #f0f4f8, #d1e8e2);
    font-family: 'Arial', sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    overflow: hidden;
}
```

Sirve para:

- Crear un fondo con degradado suave.
- Centrar el juego en el medio de la pantalla.
- Ocupar toda la altura del navegador.

```css
#game-container {
    background-color: #3c3c3c;
    border-radius: 20px;
    padding: 30px;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);
    width: 450px;
    color: white;
    text-align: center;
}
```

Sirve para:

- Dar forma al recuadro del juego.
- Aplicar bordes redondeados y sombra.
- Centrar el contenido y establecer un ancho fijo.

```
canvas {
    background-color: #222;
    border: 5px solid #ffffff;
    border-radius: 15px;
}
```

Sirve para:

- Definir el área donde se dibuja la serpiente.
- Añadir un borde blanco y esquinas redondeadas.

```
.button {
    background-color: #4CAF50;
    color: white;
    border: none;
    padding: 15px 35px;
    font-size: 18px;
    border-radius: 10px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.2s ease;
}
.button:hover {
    background-color: #45a049;
    transform: scale(1.1);
}
```

Sirve para:

- Estilizar el botón de “Jugar de nuevo”.
- Cambia de color y se agranda al pasar el mouse.

```
@keyframes snake-animation {
    0% { transform: scale(1); }
    50% { transform: scale(1.1); }
    100% { transform: scale(1); }
}
.snake {
    animation: snake-animation 0.5s ease infinite;
}
```

Sirve para:

Crear una animación que hace que la serpiente parezca moverse o latir.

3.Código JavaScript (Lógica del juego)

Esta sección controla el funcionamiento del juego, el movimiento de la serpiente y la comida.

Variables principales

```
let canvas = document.getElementById('game');
let ctx = canvas.getContext('2d');
let scoreElement = document.getElementById('score');
let messageElement = document.getElementById('message');
const scale = 20;
const rows = canvas.height / scale;
const columns = canvas.width / scale;
let snake, food, score, dx, dy, changingDirection, gameInterval;
```

- canvas y ctx: Acceden al lienzo donde se dibuja.
- scoreElement: Muestra el puntaje actual.
- messageElement: Muestra los mensajes del juego.
- scale: Tamaño de cada "cuadro" del tablero.
- snake: Arreglo con las coordenadas del cuerpo de la serpiente.
- food: Guarda la posición de la comida.
- dx, dy: Dirección en la que se mueve la serpiente.
- gameInterval: Controla la velocidad del juego.

Función principal de inicio

```
function setup() {
    snake = [
        { x: 9 * scale, y: 10 * scale },
        { x: 8 * scale, y: 10 * scale },
        { x: 7 * scale, y: 10 * scale }
    ];
    food = generateFood();
    score = 0;
    dx = scale * 1;
    dy = 0;
    changingDirection = false;
    messageElement.textContent = '';
    updateScore();
    document.addEventListener('keydown', changeDirection);
    gameInterval = setInterval(main, 150);
}
```

Sirve para:

Iniciar una nueva partida configurando la serpiente, comida, puntuación y velocidad.

Ciclo principal del juego

```
function main() {
    if (gameOver()) {
        clearInterval(gameInterval);
        return;
    }
    changingDirection = false;
    clearCanvas();
    drawFood();
    moveSnake();
    drawSnake();
    updateScore();
}
```

Sirve para:

Actualizar constantemente el juego: borra el lienzo, mueve la serpiente, dibuja la comida y actualiza el puntaje.

Funciones complementarias

```
function clearCanvas() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
}
```

Limpia el tablero antes de redibujar.

```
function drawFood() {
    ctx.fillStyle = '#FF5733';
    ctx.fillRect(food.x, food.y, scale, scale);
}
```

Dibuja la comida en una posición aleatoria.

Movimiento y crecimiento

```
function moveSnake() {
    const head = { x: snake[0].x + dx, y: snake[0].y + dy };
    snake.unshift(head);
    if (head.x === food.x && head.y === food.y) {
        score += 10;
        food = generateFood();
    } else {
        snake.pop();
    }
}
```

Sirve para:

- Crear una nueva cabeza en la dirección actual.
- Si come comida, aumenta la puntuación.
- Si no, elimina el último segmento (para simular movimiento).

Dibujo de la serpiente

```
function drawSnake() {
    ctx.fillStyle = '#66BB6A';
    snake.forEach(segment => {
        ctx.fillRect(segment.x, segment.y, scale, scale);
    });
}
```

Pinta cada parte del cuerpo de la serpiente.

Generar comida aleatoria

```
function generateFood() {
    let x = Math.floor(Math.random() * rows) * scale;
    let y = Math.floor(Math.random() * columns) * scale;
    return { x, y };
}
```

Crea posiciones aleatorias dentro del tablero.

Detección de fin del juego

```
function gameOver() {
    const head = snake[0];
    if (head.x < 0 || head.x >= canvas.width || head.y < 0 || head.y >= canvas.height)
        return mostrarFin();

    for (let i = 1; i < snake.length; i++) {
        if (head.x === snake[i].x && head.y === snake[i].y)
            return mostrarFin();
    }
    return false;
}
```

Detecta si la serpiente choca contra los bordes o consigo misma.

Control de dirección

```
function changeDirection(event) {
    if (changingDirection) return;
    changingDirection = true;
    const LEFT = 37, RIGHT = 39, UP = 38, DOWN = 40;

    if (event.keyCode === LEFT && dx === 0) { dx = -scale; dy = 0; }
    if (event.keyCode === RIGHT && dx === 0) { dx = scale; dy = 0; }
    if (event.keyCode === UP && dy === 0) { dx = 0; dy = -scale; }
    if (event.keyCode === DOWN && dy === 0) { dx = 0; dy = scale; }
}
```

Permite mover la serpiente con las flechas del teclado.

Actualización del puntaje

```
function updateScore() {
    scoreElement.textContent = score;
}
```

Muestra la puntuación actual en pantalla.

Reiniciar el juego

```
function resetGame() {
    score = 0;
    messageElement.textContent = '';
    clearInterval(gameInterval);
    setup();
}
```

Permite empezar una nueva partida desde cero.

Conclusión

El código de SnakeX combina:

- HTML para estructurar el tablero y la interfaz.
- CSS para dar estilo, color y animaciones.
- JavaScript para la lógica del movimiento, colisiones, puntuación y reinicio.

Gracias a la función setInterval(), el juego se actualiza constantemente, creando la ilusión de movimiento.
Este juego es una excelente muestra de cómo se puede usar la programación web para desarrollar experiencias interactivas con solo tres lenguajes base.