Integrantes: Kevin Sinchi

Rafael Serrano

Materia: PROGRAMACIÓN Y PLATAFORMAS WEB

Juego Duelo de Dados

Enlace

https://github.com/elsebasdev1/web-dice-duel

Código Java

```
const rulesDialog = document.getElementById('rules-dialog');
const cube = document.querySelector('.cube');
const time = 1.85; // Duración total de la animación
const rollButton = document.getElementById('roll-dice');
const turnText = document.getElementById('player-turn');
const roundText = document.getElementById('game-round');
const player1 = document.getElementById("player1");
const player2 = document.getElementById("player2");
const players = { "Player 1": 0, "Player 2": 0 };
const restartButton = document.getElementById('restart-game');
const settingsButton = document.getElementById('game-settings');
const settingsDialog = document.getElementById('settings-dialog');
const closeSettingsButton = document.getElementById('close-settings');
const settingsForm = document.getElementById('settings-form');
const diceColorPlayer1 = document.getElementById('dice-color-player1');
const diceColorPlayer2 = document.getElementById('dice-color-player2');
const player1NameInput = document.getElementById('player1-name');
const player2NameInput = document.getElementById('player2-name');
const tblPlayersPlayer1 = document.getElementById('tbl-players-player1');
const tblPlayersPlayer2 = document.getElementById('tbl-players-player2');
const tblHistoryPlayer1 = document.getElementById('tbl-history-player1');
const tblHistoryPlayer2 = document.getElementById('tbl-history-player2');
let roundScores = { "Player 1": 0, "Player 2": 0 };
const historyPopup = document.getElementById("history-popup");
const openHistoryButton = document.getElementById("open-history");
const closeHistoryButton = document.getElementById("close-history");
```

Se definen las variables y elementos del DOM que se utilizarán en el juego, incluyendo botones, cuadros de texto, configuraciones de los jugadores, tablas de historial y ventanas emergentes.

```
openHistoryButton.addEventListener("click", () => {
    loadFinalScores();
    historyPopup.showModal();
});

closeHistoryButton.addEventListener("click", () => {
    historyPopup.close();
});
```

Se agregan eventos a los botones para abrir y cerrar la ventana emergente (popup) que muestra el historial del juego. Antes de abrirlo, se cargan los puntajes finales almacenados.

```
function changeBackgroundColor() {
    const color = turn === 1 ? colorPlayer1 : colorPlayer2;
    document.body.style.background = `linear-gradient(0deg, #ddd, ${color} )`;
}

diceColorPlayer1.addEventListener('input', (event) => {
    colorPlayer1 = event.target.value;
    if (turn === 1) {
        changeBackgroundColor();
    }
});

diceColorPlayer2.addEventListener('input', (event) => {
    colorPlayer2 = event.target.value;
    if (turn === 2) {
        changeBackgroundColor();
    }
});
```

Cambia dinámicamente el fondo del juego cuando se modifica el color de los dados en la configuración. Se actualiza el color correspondiente al turno del jugador activo.

```
settingsForm.addEventListener('submit', (event) => {
    event.preventDefault();
    colorPlayer1 = diceColorPlayer1.value;
    colorPlayer2 = diceColorPlayer2.value;
    player1Name = player1NameInput.value;
    player2Name = player2NameInput.value;
    turnText.textContent = player1Name;
    tblPlayersPlayer1.textContent = player1Name;
    tblPlayersPlayer2.textContent = player2Name;
    tblHistoryPlayer1.textContent = player1Name;
    tblHistoryPlayer2.textContent = player2Name;
    changeBackgroundColor();
    settingsDialog.close();
});
```

Al guardar la configuración, se actualizan los nombres de los jugadores, los colores y se refleja en la tabla de jugadores e historial. Se cierra el cuadro de diálogo de configuración.

```
function rollDice() {
    if (isRolling) {
        return;
    }

    isRolling = true;
    rollCount++;
    audio.currentTime = 0;
    audio.play();

    cube.style.transition = '';
    cube.style.animation = '';

    // Simular rebote inicial
    cube.style.transition = `transform 0.3s cubic-bezier(0.3, 0.5, 0.5, 1)`;
    cube.style.transform = `translateX(200px) rotateX(720deg) rotateY(720deg) rotateZ(720deg)`;

    setTimeout(() => {
        // Bajada del dado con rebote
        cube.style.transform = `translateY(100px) rotateX(360deg) rotateY(360deg) rotateZ(360deg)`;

        setTimeout(() => {
            // Rotación final aleatoria
            cube.style.transition = `transform ${time}s ease-out`;
            const randomValue = Math.floor(Math.random() * 6) + 1;
            console.log(`randomValue: ${randomValue}`);
            roundScores[`Player ${turn}`] = randomValue;

            switch(randomValue) {
                case 1: cube.style.transform = `translateY(200px) rotateX(3600deg) rotateY(3600deg) rotateZ(3600deg)`; break;
                case 2: cube.style.transform = `translateY(200px) rotateX(4410deg) rotateY(3600deg) rotateZ(3600deg)`; break;
                case 3: cube.style.transform = `translateY(200px) rotateX(3600deg) rotateY(4410deg) rotateZ(3600deg)`; break;
                case 4: cube.style.transform = `translateY(200px) rotateX(3600deg) rotateY(2430deg) rotateZ(3600deg)`; break;
                case 5: cube.style.transform = `translateY(200px) rotateX(2430deg) rotateY(3600deg) rotateZ(3600deg)`; break;
                case 6: cube.style.transform = `translateY(200px) rotateX(3600deg) rotateY(1980deg) rotateZ(3600deg)`; break;
```

Esta función maneja el lanzamiento del dado, aplicando animaciones y generando un número aleatorio. Luego, actualiza el puntaje del jugador activo.

```
function saveToHistory() {
    const historyBody = document.getElementById("history-body");
    const row = document.createElement("tr");

    const roundCell = document.createElement("td");
    const player1Cell = document.createElement("td");
    const player2Cell = document.createElement("td");

    roundCell.textContent = round;
    player1Cell.textContent = `${roundScores["Player 1"]} pts`;
    player2Cell.textContent = `${roundScores["Player 2"]} pts`;

    row.appendChild(roundCell);
    row.appendChild(player1Cell);
    row.appendChild(player2Cell);

    historyBody.appendChild(row);
}
```

Cada vez que ambos jugadores terminan su turno en una ronda, sus puntajes se registran en la tabla de historial.

```
function saveFinalScore() {
    const finalScores = {
        date: new Date().toLocaleString(),
        player1: players["Player 1"],
        player2: players["Player 2"]
    };

    let savedScores = JSON.parse(localStorage.getItem("gameScores")) || [];
    savedScores.push(finalScores);
    localStorage.setItem("gameScores", JSON.stringify(savedScores));

    loadFinalScores(); // Cargar en la tabla
}

function loadFinalScores() {
    const savedScores = JSON.parse(localStorage.getItem("gameScores")) || [];
    const historyBody1 = document.getElementById("history-body1");

    //historyBody1.innerHTML = ""; // Limpiar tabla antes de cargar

    savedScores.forEach(score => {
        const row = document.createElement("tr");

        const dateCell = document.createElement("td");
        const player1Cell = document.createElement("td");
        const player2Cell = document.createElement("td");

        dateCell.textContent = score.date;
        player1Cell.textContent = `${score.player1} pts`;
        player2Cell.textContent = `${score.player2} pts`;

        row.appendChild(dateCell);
        row.appendChild(player1Cell);
        row.appendChild(player2Cell);
        historyBody1.appendChild(row);
    });
}
```

Guardamos los datos de la suma total por los 2 jugadores en un localstorage, para después mostrarla en una tabla como historial de puntajes.

```
function restartGame(){
    if (confirm("Restart the game?")) {
        turnText.textContent = `Player 1`;
        roundText.textContent = ` 1`;
        players[`Player 1`] = 0;
        players[`Player 2`] = 0;
        player1.textContent = `${players["Player 1"]} pts`;
        player2.textContent = `${players["Player 2"]} pts`;
        turn = 1; //Player 1 starts
        round = 1;
        player1Name = "Player 1";
        player2Name = "Player 2";
        tblPlayersPlayer1.textContent = player1Name;
        tblPlayersPlayer2.textContent = player2Name;
        tblHistoryPlayer1.textContent = player1Name;
        tblHistoryPlayer2.textContent = player2Name;
        rollCount = 0;
        isRolling = false;
        const historyBody = document.getElementById("history-body");
        historyBody.innerHTML = "";
        rollButton.disabled = false;
    }
}
```

Después de haber confirmado el reinicio del juego, inicializamos todos los datos desde 0 para que así no haya ningún error.

Código HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dice Duel</title>
    <link rel="stylesheet" href="style.css">
    <link rel="shortcut icon" href="https://cdn-icons-png.flaticon.com/512/5052/5052121.png" type="image/x-icon">
    <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined&display=swap">
```

Definimos la estructura base del documento HTML, incluyendo la codificación de caracteres, la configuración de la vista en dispositivos móviles y la carga de estilos CSS, el ícono de la pestaña y los iconos de Google Fonts.

```html
<body>
    <main>
        <h1>Welcome to Dice Duel</h1>

        <div class="table-styled">
            <table>
                <thead>
                    <tr>
                        <th>Turn</th><td id="player-turn">Player 1</td>
                    </tr>
                </thead>
                <thead>
                    <tr>
                        <th>Round</th><td id="game-round">1</td>
                    </tr>
                </thead>

            </table>
        </div>
```

Muestra el título del juego y una tabla con información básica del estado actual: turno del jugador y número de ronda.

```html
<div class="table-players">
    <table>
        <thead>
            <tr>
                <th id="tbl-players-player1">Player 1</th><th id="tbl-players-player2">Player 2</th>
            </tr>
        </thead>
        <tr>
            <td id="player1">0 pts</td><td id="player2">0 pts</td>
        </tr>
    </table>
</div>
```

Esta tabla muestra los nombres y puntajes actuales de ambos jugadores.

```html
<div class="table-history">
    <table>
        <thead>
            <tr>
                <th>Turno</th><th id="tbl-history-player1">Player 1</th><th id="tbl-history-player2">Player 2</th>
            </tr>
        </thead>
        <tbody id="history-body"></tbody>
    </table>
```

Tabla donde se registran los puntajes de cada ronda a medida que avanza el juego.

```html
<dialog id="history-popup">
    <div class="popup-content">
        <h3>Historial de Juegos</h3>
        <table>
            <thead>
                <tr>
                    <th>Fecha</th>
                    <th>Player 1</th>
                    <th>Player 2</th>
                </tr>
            </thead>
            <tbody id="history-body1"></tbody>
        </table>
        <button id="close-history">Cerrar</button>
    </div>
</dialog>
```

Muestra un historial de juegos previos dentro de un cuadro de diálogo emergente.

```
<div class="container">
    <div class="cube">
        <div class="cube-face front">
            <div class="inside">
                <span class="dot"></span>
            </div>
        </div>
        <div class="cube-face back">
            <div class="inside">
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
            </div>
        </div>
        <div class="cube-face left">
            <div class="inside">
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
            </div>
        </div>
        <div class="cube-face right">
            <div class="inside">
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
                <span class="dot"></span>
            </div>
        </div>
```

Define la estructura del dado en 3D con sus seis caras, cada una representando un número con puntos.

```
<button id="roll-dice">
    <span class="material-symbols-outlined">casino</span>
</button>
ain>
```

Botón que el usuario presiona para lanzar el dado.

```
<div class="tooltip-container rules-btn">
    <button id="open-popup">
        <span class="material-symbols-outlined">help</span>
    </button>
    <span class="tooltip-text">Rules</span>
</div>

<div class="tooltip-container restart-btn">
    <button id="restart-game">
        <span class="material-symbols-outlined">restart_alt</span>
    </button>
    <span class="tooltip-text">Restart</span>
</div>

<div class="tooltip-container setting-btn">
    <button id="game-settings">
        <span class="material-symbols-outlined">settings</span>
    </button>
    <span class="tooltip-text">Settings</span>
</div>

<div class="tooltip-container history-btn">
    <button id="open-history">
        <span class="material-symbols-outlined">history</span>
    </button>
    <span class="tooltip-text">History</span>
</div>
```

Botones con iconos que permiten al usuario ver reglas, reiniciar el juego, cambiar configuraciones y revisar el historial.

```
<dialog id="settings-dialog">
    <h2>Game's Configuration</h2>
    <form id="settings-form">
        <label for="player1-name">Player 1:</label>
        <input type="text" id="player1-name" name="player1-name" value="Player 1">
        <label for="dice-color">Color:</label>
        <input type="color" id="dice-color-player1" name="dice-color-player1" value="#90caf9">
        <label for="player2-name">Player 2:</label>
        <input type="text" id="player2-name" name="player2-name" value="Player 2">
        <label for="dice-color">Color:</label>
        <input type="color" id="dice-color-player2" name="dice-color-player2" value="#ffbb80">
        <br>
        <button type="submit">Save</button>
        <button type="button" id="close-settings">Close</button>
    </form>
</dialog>
```

Permite cambiar nombres y colores de los jugadores.

Código CSS

```css
body{
    margin: 0;
    font-family: "Poppins", sans-serif;
    background: linear-gradient(0deg, #ddd, #90caf9);
}
```

Define los estilos generales del cuerpo de la página, eliminando los márgenes y estableciendo una fuente personalizada con un fondo en degradado.

```css
main{
    min-height: 100vh;
    display: grid;
    place-items: center;
    grid-template-rows: 0 1fr auto; /*Se divide en tres filas*/
}
```

Establece el área principal de la página con una altura mínima del 100% de la ventana, centrando su contenido con grid y dividiendo el espacio en tres filas.

```css
h1{
    align-self: self-start;
    font-size: 3em;
    font-weight: bold;
    color: #222;
    margin-top: 0.25rem;
}
```

Define los estilos de los títulos principales con un tamaño grande, peso en negrita, color oscuro y un pequeño margen superior.

```css
#open-popup,
#restart-game,
#roll-dice,
#game-settings,
#open-history{
    align-self: end; /* Abajo */
    justify-self: start; /* Izquierda */
    font-size: 1rem;
    background: #00acee;
    border: none;
    color: white;
    cursor: pointer;
    border-radius: 5px;
    padding: 0.75em 1em;
    margin-left: 1rem;
    margin-bottom: 1rem;
}
```

Aplica estilos generales a los botones, estableciendo su posición, color de fondo azul, texto blanco, bordes redondeados y un cursor de puntero al pasar el ratón.

```css
dialog {
    box-sizing: border-box;
    width: 100%;
    max-width: 525px;
    min-width: 300px;
    border: 1px solid #ddd;
    border-radius: 0.5em;
    text-align: center;
    padding: 0.8em 1.5em 1em;
}
```

Define el diseño de los cuadros de diálogo, con un ancho máximo de 525px, bordes redondeados y un texto centrado.

```css
.tooltip-container .tooltip-text {
    visibility: hidden;
    width: 80px;
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 6px;
    border-radius: 6px;
    position: absolute;
    bottom: 120%; /* Para que aparezca arriba del botón */
    left: 70%;
    transform: translateX(-55%);
    font-size: 12px;
    white-space: nowrap;
}
```

Estiliza los tooltips, que aparecen al pasar el cursor sobre ciertos elementos. Se ocultan inicialmente y tienen un fondo oscuro, texto blanco y bordes redondeados.

```css
.container {
    width: 200px;
    height: 200px;
    margin: auto auto;
    position: absolute;
    perspective: 800px;
    top: -50px;
}
```

Define el contenedor del dado 3D, estableciendo su tamaño y agregando perspective para permitir efectos tridimensionales.

```css
.cube {
    transform: translateY(200px) rotateX(0deg) rotateY(0deg) rotateZ(0deg);
    width: 100%;
    height: 100%;
    transform-style: preserve-3d;
    margin-top: 4.5em;
}
```

Configura el dado 3D con un estilo tridimensional (preserve-3d) y una posición inicial en el eje Y.

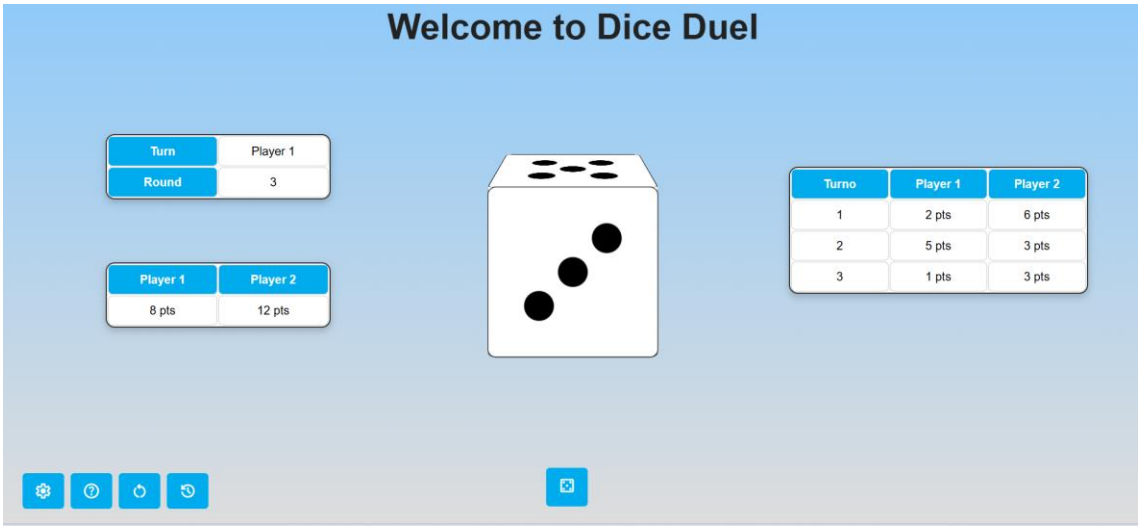
```css
.table-styled, .table-players, .table-history {
    position: absolute;
    border: 2px solid #333;
    background: #fff;
    border-collapse: separate;
    border-spacing: 0;
    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
    border-radius: 12px;
    overflow: hidden;
    text-align: left;
    width: 300px; /* Valor por defecto */
}
```

Aplica estilos a las tablas del juego, dándoles bordes redondeados, un fondo blanco, una sombra y un tamaño predefinido.

```css
@keyframes glowing {
    0% {
        background-position: 0 0;
    }
    50% {
        background-position: 400% 0;
    }
    100% {
        background-position: 0 0;
    }
}
```

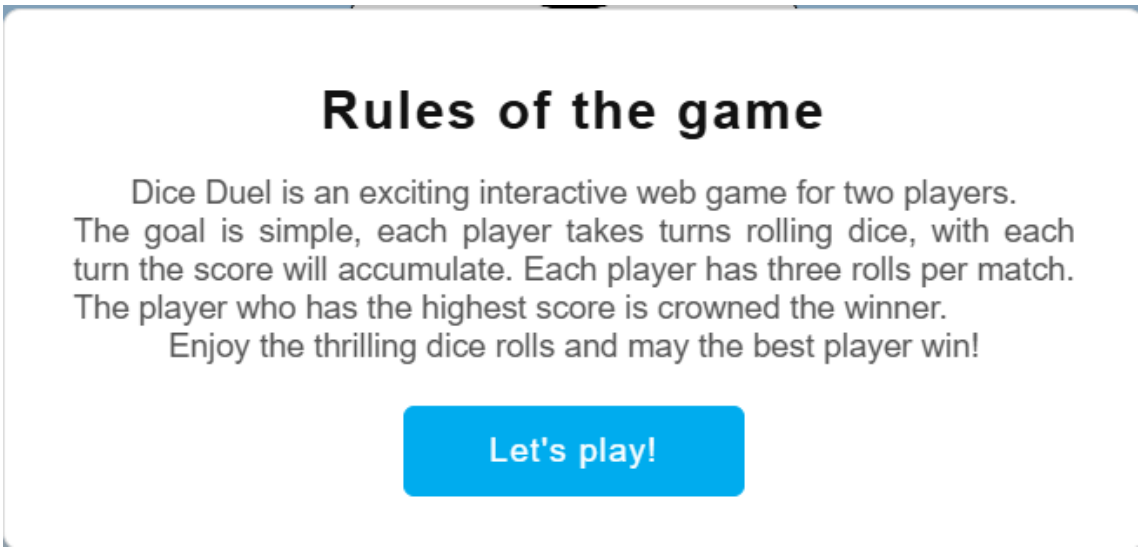
Define una animación que crea un efecto de brillo en los botones, moviendo el fondo en un ciclo continuo.

Parte Grafica



Esta es la forma final que tiene nuestro juego de duelo de dados. Tenemos 3 tablas que nos muestran a que jugador le toca su turno, la ronda en la que están, la segunda tabla muestra la suma del número aleatorio que sacaron. Y la tercera tabla es el numero que sacaron en cada ronda. Adicionalmente los botones de lanzar dado, configuración, reglas, reiniciar e historial.

Aquí podemos cambiar los nombres de los jugadores que pusimos predeterminados y color que deseen escoger.



Al iniciar la página automáticamente aparecen las reglas del juego pero también existe un botón en caso de que los jugadores deseen volver a leerlas.

Historial de Juegos

Fecha	Player 1	Player 2
2/4/2025, 9:47:47 p. m.	9 pts	13 pts
2/4/2025, 9:52:45 p. m.	17 pts	12 pts
2/4/2025, 9:53:44 p. m.	7 pts	9 pts
2/4/2025, 10:05:21 p. m.	3 pts	9 pts
2/4/2025, 10:14:14 p. m.	8 pts	15 pts
2/4/2025, 10:27:29 p. m.	7 pts	8 pts
2/4/2025, 10:28:53 p. m.	8 pts	12 pts
2/4/2025, 10:29:45 p. m.	12 pts	9 pts
2/4/2025, 10:30:33 p. m.	8 pts	9 pts
2/4/2025, 10:31:02 p. m.	5 pts	11 pts
2/4/2025, 10:34:26 p. m.	10 pts	10 pts
2/4/2025, 10:52:06 p. m.	16 pts	11 pts

La tabla de historial nos muestra la fecha y la sumatoria de puntajes obtenidos por cada jugador.