UNIVERSIDAD DE MEDELLÍN

INGENIEIRA DE SISTEMAS

DESARROLLO DE APLICACIONES WEB

JOSÉ JULIÁN ZAPATA ARBELÁEZ

BONIFICACION JAVA SCRIPT – JUEGO

SANTIAGO VIEIRA CEBALLOS

2025-1

Explicación y estructura del juego:

- Estructura del HTML usé las clases container y player para una mejor edición en el css y un llamado más fácil a las funciones en el javascript.

index.html

app > html > index.html > ...

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Juego de Pelea</title>
    <link rel="stylesheet" href="./../css/styles.css">
</head>
<body>
    <div class="container">
        <div id="player1-box" class="player">
            <h2 id="player1-name">Johnvy</h2>
            <p>Vida: <span id="player1-life">50</span></p>
            <button id="player1" onclick="fight(player1)">Atacar</button>
        </div>
        <div id="player2-box" class="player">
            <h2 id="player2-name">El santi</h2>
            <p>Vida: <span id="player2-life">50</span></p>
            <button id="player2" onclick="fight(player2)">Atacar</button>
        </div>
    </div>

    <script src="./../scripts/script.js"></script>
</body>
</html>
```

- En la estructura del HTML resalta una etiqueta como lo es el . Que funciona para mostrar dinámicamente la vida de cada jugador en la interfaz. Es un contenedor en línea que permite modificar solo esa parte del texto sin afectar el resto del contenido.

```
<div id="player2-box" class="player">
    <h2 id="player2-name">El santi</h2>
    <p>Vida: <span id="player2-life">50</span></p>
    <button id="player2" onclick="fight(player2)">Atacar</button>
</div>
```

```
<div id="player1-box" class="player">
    <h2 id="player1-name">Johnvy</h2>
    <p>Vida: <span id="player1-life">50</span></p>
    <button id="player1" onclick="fight(player1)">Atacar</button>
```

- Estructura del funcionamiento con JS.

```
app > scripts > JS script.js > fight
let player1 = { name: 'Johnvy', life: 50 };
let player2 = { name: 'El santi', life: 50 };

const randomShoot = () => Math.floor(Math.random() * 10) + 1;

const updateLife = () => {
    document.getElementById("player1-life").textContent = player1.life;
    document.getElementById("player2-life").textContent = player2.life;
};

const fightWithDamage = (player, damage) => {
    player.life -= damage;
    updateLife();
};

const endGame = player => {
    if (player.life <= 0) {
        document.getElementById(player.name === 'Johnvy' ? "player1" : "player2").disabled = true;
        document.getElementById(player.name === 'Johnvy' ? "player1-box" : "player2-box").innerHTML += `<p><strong>${player
    }
};

const fight = player => {
    if (player.life <= 0) return;
    let damage = randomShoot();
    if (player === player1) {
        fightWithDamage(player2, damage);
        endGame(player2);
    } else {
        fightWithDamage(player1, damage);
        endGame(player1);
    }
```

- Este código fue realizado en clase, pero tuvo una modificación especial en la función endGame, que consta de un innerHTML

  Usa += para agregar contenido dentro del div, sin borrar lo que ya está.
  En este caso, agrega un con el mensaje:
  Johnvy ha sido derrotado.

  Esto permite que el mensaje se muestre en pantalla cuando un jugador pierda.

```
document.getElementById(player.name === 'Johnvy' ? "player1-box" : "player2-box").innerHTML += `<p><strong>${player.name}
```

En resumidas cuentas, el funcionamiento de este es de la siguiente forma:

Muestra un mensaje cuando un jugador pierde.

Usa innerHTML para agregarlo dentro de un div.

Elige dinámicamente el div correcto usando un operador ternario.

- Estructuras del CSS.

app > css > # styles.css > .container

```css
body {
    background: #0a0a0a;
    color: white;
    font-family: Arial, sans-serif;
    text-align: center;
    margin: 0;
    height: 100vh;
    display: flex;
    align-items: center;
    justify-content: center;
}

.container {
    display: flex;
    gap: 50px;
    padding: 20px;
}

.player {
    display: inline-block;
    padding: 20px;
    background: rgba(50, 0, 0, 0.8);
    border: 2px solid red;
    border-radius: 10px;
    box-shadow: 0 0 10px red;
}
```

```css
button {
    padding: 10px 20px;
    font-size: 16px;
    cursor: pointer;
    background: red;
    color: white;
    border: none;
    border-radius: 5px;
    transition: 0.3s;
}

button:hover {
    background: darkred;
}
```

- Como dije inicialmente me ayudé de las clases para poder realizar un organizado archivo CSS y tener una mejor experiencia al jugar. El resultado final es este.