Task3

to.js

const board = document.getElementById('board');

const statusDiv = document.getElementById('status');

const restartBtn = document.getElementById('restart');

const scoreX = document.getElementById('score-x');

const scoreO = document.getElementById('score-o');

const scoreDraw = document.getElementById('score-draw');

let cells = [];

let gameState = Array(9).fill('');

let currentPlayer = 'X';

let gameActive = true;

let scores = { X: 0, O: 0, Draw: 0 };

const winCombos = [

[0,1,2], [3,4,5], [6,7,8], // rows

[0,3,6], [1,4,7], [2,5,8], // columns

[0,4,8], [2,4,6] // diagonals

];

// Create cells dynamically

function createBoard() {

board.innerHTML = '';

cells = [];

for (let i = 0; i < 9; i++) {

const cell = document.createElement('div');

cell.className = 'cell';

cell.dataset.index = i;

cell.tabIndex = 0; // for keyboard accessibility

cell.addEventListener('click', handleCellClick);

// Hover preview

cell.addEventListener('mouseenter', handleCellMouseEnter);

cell.addEventListener('mouseleave', handleCellMouseLeave);

// Keyboard support: Space/Enter to play

cell.addEventListener('keydown', function(e) {

if (!gameActive || gameState[i]) return;

if (e.key === " " || e.key === "Enter") {

handleCellClick({target: cell});

}

});

board.appendChild(cell);

cells.push(cell);

}

}

// Hover preview logic

function handleCellMouseEnter(e) {

const idx = e.target.dataset.index;

if (!gameActive || gameState[idx]) return;

e.target.textContent = currentPlayer;

e.target.classList.add('preview');

}

function handleCellMouseLeave(e) {

const idx = e.target.dataset.index;

if (!gameActive || gameState[idx]) return;

e.target.textContent = '';

e.target.classList.remove('preview');

}

// Cell click logic

function handleCellClick(e) {

const idx = e.target.dataset.index;

if (!gameActive || gameState[idx]) return;

gameState[idx] = currentPlayer;

cells[idx].textContent = currentPlayer;

cells[idx].classList.add('filled');

const winner = checkWinner();

if (winner) {

showWinner(winner);

} else if (!gameState.includes('')) {

showDraw();

} else {

currentPlayer = currentPlayer === 'X' ? 'O' : 'X';

statusDiv.textContent = `${currentPlayer}'s turn`;

}

}

// Check for winner

function checkWinner() {

for (let combo of winCombos) {

const [a, b, c] = combo;

if (

gameState[a] &&

gameState[a] === gameState[b] &&

gameState[b] === gameState[c]

) {

return { player: gameState[a], combo };

}

}

return null;

}

// Show winner

function showWinner({ player, combo }) {

combo.forEach(i => cells[i].classList.add('win'));

statusDiv.textContent = `${player} wins!`;

scores[player]++;

updateScores();

gameActive = false;

}

// Show draw

function showDraw() {

statusDiv.textContent = "It's a draw!";

scores.Draw++;

updateScores();

gameActive = false;

}

// Update score display

function updateScores() {

scoreX.textContent = `X: ${scores.X}`;

scoreO.textContent = `O: ${scores.O}`;

scoreDraw.textContent = `Draw: ${scores.Draw}`;

}

// Restart game

function restartGame() {

gameState = Array(9).fill('');

cells.forEach(cell => {

cell.textContent = '';

cell.classList.remove('win', 'filled', 'preview');

});

currentPlayer = 'X';

statusDiv.textContent = "X's turn";

gameActive = true;

}

// Initialize

createBoard();

updateScores();

statusDiv.textContent = "X's turn";

restartBtn.addEventListener('click', restartGame);