*const* playBoard = document.querySelector(".play-board");

*const* scoreElement = document.querySelector(".score");

*const* highScoreElement = document.querySelector(".high-score");

*const* controls = document.querySelectorAll(".controls i");

*let* gameOver = false;

*let* foodX, foodY;

*let* snakeX = 5, snakeY = 5;

*let* velocityX = 0, velocityY = 0;

*let* snakeBody = [];

*let* setIntervalId;

*let* score = 0;

// Get high score from local storage

*let* highScore = localStorage.getItem("high-score") || 0;

highScoreElement.innerText = `High Score: ${highScore}`;

// Pass a random between 1 and 30 as food position

*const* updateFoodPosition = () *=>* {

    foodX = Math.floor(Math.random() \* 29) + 1;

    foodY = Math.floor(Math.random() \* 29) + 1;

}

*const* handleGameOver = () *=>* {

    clearInterval(setIntervalId);

    alert("Game Over! Press OK to replay...");

    location.reload();

}

// Change velocity value based on key press

*const* changeDirection = *e* *=>* {

    if (*e*.key === "ArrowUp" && velocityY != 1) {

        velocityX = 0;

        velocityY = -1;

    } else if (*e*.key === "ArrowDown" && velocityY != -1) {

        velocityX = 0;

        velocityY = 1;

    } else if (*e*.key === "ArrowLeft" && velocityX != 1) {

        velocityX = -1;

        velocityY = 0;

    } else if (*e*.key === "ArrowRight" && velocityX != -1) {

        velocityX = 1;

        velocityY = 0;

    }

}

// Change Direction on each key click

controls.forEach(*button* *=>* *button*.addEventListener("click", () *=>* changeDirection({ key: *button*.dataset.key })));

*const* initGame = () *=>* {

    if (gameOver) {

        audioDed.play();

        return handleGameOver()

    }

*let* html = `<div class="food" style="grid-area: ${foodY} / ${foodX}"></div>`;

    // When snake eat food

    if (snakeX === foodX && snakeY === foodY) {

        updateFoodPosition();

        snakeBody.push([foodY, foodX]); //Add food to snake body array

        score++;

        audioEat.play();

        highScore = score >= highScore ? score : highScore; // if score > high score => high score = score

        localStorage.setItem("high-score", highScore);

        scoreElement.innerText = `Score: ${score}`;

        highScoreElement.innerText = `High Score: ${highScore}`;

    }

    // Update Snake Head

    snakeX += velocityX;

    snakeY += velocityY;

    // Shifthing forward values of elements in snake body by one

    for (*let* i = snakeBody.length - 1; i > 0; i--) {

        snakeBody[i] = snakeBody[i - 1];

    }

    snakeBody[0] = [snakeX, snakeY];

    // Check snake body is out of wall or no

    if (snakeX <= 0 || snakeX > 30 || snakeY <= 0 || snakeY > 30) {

        return gameOver = true;

    }

    // Add div for each part of snake body

    for (*let* i = 0; i < snakeBody.length; i++) {

        html += `<div class="head" style="grid-area: ${snakeBody[i][1]} / ${snakeBody[i][0]}"></div>`;

        // Check snake head hit body or no

        if (i !== 0 && snakeBody[0][1] === snakeBody[i][1] && snakeBody[0][0] === snakeBody[i][0]) {

            gameOver = true;

        }

    }

    playBoard.innerHTML = html;

}

updateFoodPosition();

setIntervalId = setInterval(initGame, 100);

document.addEventListener("keyup", changeDirection);

*var* audioEat = new *Audio*('eat\_sound.mp3');

*var* audioDed = new *Audio*('ded.mp3');

if(score>2)

{

createSprite(Math.random(0,420),Math.random(o,380),3,5)

}