HTML

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="icon" href="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQQXH2zF9qLFA3fR7NjT1l2Lz\_nOJi8h9QK8yY16cOv\_rFfh4BnLHvoOpOpt-MtbkPMUgk&usqp=CAU" type="image/png">

<title>Game</title>

<link rel="stylesheet" href="style.css">

<script src="script.js" defer></script>

</head>

<body>

<div class="background"></div>

<img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQQXH2zF9qLFA3fR7NjT1l2Lz\_nOJi8h9QK8yY16cOv\_rFfh4BnLHvoOpOpt-MtbkPMUgk&usqp=CAU alt="character-img" class="character" id="character-1">

<div class="message">

Enter To Start Game <p><span style="color: red;">&uarr;</span> ArrowUp to Control</p>

</div>

<div class="score">

<span class="score\_title"></span>

<span class="score\_val"></span>

</div>

</body>

</html>

CSS

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: Arial, Helvetica, sans-serif;

}

.background {

height: 100vh;

width: 100vw;

background: url('https://img.freepik.com/free-vector/arcade-game-world-pixel-scene\_24640-45908.jpg') no-repeat center center fixed;

-webkit-background-size: cover;

-moz-background-size: cover;

-o-background-size: cover;

background-size: cover;

}

.character {

height: 100px;

width: 130px;

position: fixed;

top: 40vh;

left: 30vw;

z-index: 100;

}

.pipe\_sprite {

position: fixed;

top: 40vh;

left: 100vw;

height: 70vh;

width: 6vw;

background:radial-gradient(lightgreen 50%, green);

border: 5px solid black;

}

.message {

position: absolute;

z-index: 10;

color: black;

top: 30%;

left: 50%;

font-size: 4em;

transform: translate(-50%, -50%);

text-align: center;

}

.messageStyle{

background: white;

padding: 30px;

box-shadow: rgba(0, 0, 0, 0.24) 0px 3px 8px;

border-radius: 5%;

}

.score {

position: fixed;

z-index: 10;

height: 10vh;

font-size: 10vh;

font-weight: 100;

color: white;

-webkit-text-stroke-width: 2px;

-webkit-text-stroke-color: black;

top: 0;

left: 0;

margin: 10px;

font-family: Arial, Helvetica, sans-serif;

}

.score\_val {

color: gold;

font-weight: bold;

}

@media only screen and (max-width: 1080px) {

.message{

font-size: 50px;

top: 50%;

white-space: nowrap;

}

.score{

font-size: 8vh;

}

.bird{

width: 120px;

height: 90px;

}

.pipe\_sprite{

width: 14vw;

}

}

JSS

// Wait for the DOM to fully load before executing any JavaScript code

document.addEventListener('DOMContentLoaded', () => {

// Get important elements from the HTML

const character = document.getElementById('character-1'); // Character (player)

const message = document.querySelector('.message'); // The message element to show instructions/game over

const scoreVal = document.querySelector('.score\_val'); // Where we display the score

let isJumping = false; // Track whether the character is currently jumping

let isGameRunning = false; // Track whether the game has started

let score = 0; // Player's score

// Listen for key presses

document.addEventListener('keydown', (e) => {

// Start the game when Enter key is pressed

if (e.key === 'Enter' && !isGameRunning) {

message.style.display = 'none'; // Hide the start message

startGame(); // Call startGame to begin the game

}

// Make the character jump when the up arrow (↑) key is pressed

if (e.key === 'ArrowUp' && isGameRunning && !isJumping) {

jump(); // Call the jump function

}

});

// Function to make the character jump

function jump() {

if (isJumping) return; // Prevent jumping if already in the middle of a jump

isJumping = true; // Set jumping to true

let jumpHeight = 0; // Track how high the character jumps

const jumpUp = setInterval(() => {

if (jumpHeight >= 150) { // If the character reaches the maximum jump height

clearInterval(jumpUp); // Stop the upward movement

// Start the falling motion after reaching the peak

const fallDown = setInterval(() => {

if (jumpHeight <= 0) { // If character has reached the ground

clearInterval(fallDown); // Stop falling

isJumping = false; // Allow jumping again

} else {

jumpHeight -= 5; // Gradually decrease jumpHeight to simulate falling

character.style.top = `${40 - jumpHeight / 10}vh`; // Adjust character's position

}

}, 20);

} else {

jumpHeight += 5; // Increase jumpHeight to simulate upward movement

character.style.top = `${40 - jumpHeight / 10}vh`; // Adjust character's position

}

}, 20);

}

// Function to start the game

function startGame() {

isGameRunning = true; // Mark the game as running

score = 0; // Reset the score to 0

scoreVal.innerText = score; // Update the score display

// Start spawning pipes (obstacles) every 2 seconds

const pipeInterval = setInterval(() => {

const pipe = document.createElement('div'); // Create a new pipe element

pipe.classList.add('pipe\_sprite'); // Add the 'pipe\_sprite' class to the pipe

document.body.appendChild(pipe); // Add the pipe to the body

let pipeLeft = window.innerWidth; // Start the pipe at the far right of the screen

// Move the pipe from right to left

const movePipe = setInterval(() => {

if (!isGameRunning) { // If the game has ended, stop moving the pipe

clearInterval(movePipe);

return;

}

pipeLeft -= 5; // Move the pipe 5px to the left each frame

pipe.style.left = `${pipeLeft}px`; // Update the pipe's position

// Check for collision: if the pipe is near the character and the character is too low

if (pipeLeft < 300 && pipeLeft > 150) {

const characterTop = parseInt(window.getComputedStyle(character).top); // Get character's current vertical position

if (characterTop > 45 \* window.innerHeight / 100) { // If the character is too low (collides with pipe)

gameOver(pipeInterval); // Trigger game over

}

}

// If the pipe has moved off the screen, remove it and increase the score

if (pipeLeft < -50) {

clearInterval(movePipe);

pipe.remove();

score++; // Increase the score for successfully avoiding the pipe

scoreVal.innerText = score; // Update the score display

}

}, 20); // Update pipe movement every 20ms

}, 2000); // Spawn a new pipe every 2 seconds

}

// Function to handle the game over scenario

function gameOver(pipeInterval) {

isGameRunning = false; // Stop the game

clearInterval(pipeInterval); // Stop spawning new pipes

message.innerHTML = 'Game Over<br>Press Enter to Restart'; // Show Game Over message

message.classList.add('messageStyle'); // Add styling for the message

message.style.display = 'block'; // Show the message

// Remove all existing pipes from the screen

document.querySelectorAll('.pipe\_sprite').forEach(pipe => pipe.remove());

// Reset the character's position

character.style.top = '40vh'; // Reset character to starting position

}

});