<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>world.execute(me); - Mili</title>

<style>

body {

margin: 0;

overflow: hidden;

background: #000;

color: #0f0;

font-family: 'Courier New', monospace;

}

#terminal {

position: absolute;

top: 20px;

left: 20px;

z-index: 100;

white-space: pre;

line-height: 1.5;

text-shadow: 0 0 5px #0f0;

pointer-events: none;

}

#cursor {

display: inline-block;

width: 10px;

height: 20px;

background: #0f0;

animation: blink 1s infinite;

}

@keyframes blink {

0%, 100% { opacity: 1; }

50% { opacity: 0; }

}

#audio-controls {

position: absolute;

bottom: 20px;

left: 20px;

z-index: 100;

}

button {

background: #000;

color: #0f0;

border: 1px solid #0f0;

padding: 5px 10px;

font-family: 'Courier New', monospace;

cursor: pointer;

}

button:hover {

background: #0f0;

color: #000;

}

canvas {

display: block;

}

</style>

</head>

<body>

<!-- 3D 画布 -->

<canvas id="three-canvas"></canvas>

<!-- 终端文字 -->

<div id="terminal">

> Initializing world.execute(me);<br>

> Loading Mili...<br>

<span id="text"></span><span id="cursor">\_</span>

</div>

<!-- 音频控制 -->

<div id="audio-controls">

<button id="play-btn">PLAY</button>

<audio id="audio" src="https://music.jsbaidu.com/upload/320/2021/03/07/1083998.mp3"></audio>

</div>

<!-- Three.js + 音频可视化 -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/three.js/r128/three.min.js"></script>

<script>

// ===== 1. 初始化 Three.js =====

const scene = new THREE.Scene();

const camera = new THREE.PerspectiveCamera(75, window.innerWidth / window.innerHeight, 0.1, 1000);

const renderer = new THREE.WebGLRenderer({ canvas: document.getElementById("three-canvas"), antialias: true });

renderer.setSize(window.innerWidth, window.innerHeight);

renderer.setClearColor(0x000000);

// 相机位置

camera.position.z = 5;

// ===== 2. 创建 3D 几何体 =====

const geometries = [

new THREE.BoxGeometry(),

new THREE.SphereGeometry(1, 32, 32),

new THREE.TorusGeometry(1, 0.4, 16, 32),

new THREE.ConeGeometry(1, 2, 32)

];

const materials = [

new THREE.MeshBasicMaterial({ color: 0x00ff00, wireframe: true }),

new THREE.MeshBasicMaterial({ color: 0xff00ff, wireframe: true }),

new THREE.MeshBasicMaterial({ color: 0x00ffff, wireframe: true }),

new THREE.MeshBasicMaterial({ color: 0xffff00, wireframe: true })

];

const meshes = [];

geometries.forEach((geo, i) => {

const mesh = new THREE.Mesh(geo, materials[i]);

mesh.position.x = (i - 1.5) \* 2.5;

scene.add(mesh);

meshes.push(mesh);

});

// ===== 3. 音频可视化 =====

const audio = document.getElementById("audio");

const playBtn = document.getElementById("play-btn");

let audioContext, analyser, dataArray;

playBtn.addEventListener("click", () => {

if (audio.paused) {

audio.play();

playBtn.textContent = "PAUSE";

initAudioAnalyser();

} else {

audio.pause();

playBtn.textContent = "PLAY";

}

});

function initAudioAnalyser() {

audioContext = new (window.AudioContext || window.webkitAudioContext)();

const source = audioContext.createMediaElementSource(audio);

analyser = audioContext.createAnalyser();

analyser.fftSize = 256;

source.connect(analyser);

analyser.connect(audioContext.destination);

dataArray = new Uint8Array(analyser.frequencyBinCount);

}

// ===== 4. 歌词逐行打印 =====

const lyrics = [

"If I'm a set of points...",

"I will give you my dimension.",

"If I'm a circle...",

"I will give you my circumference.",

"world.execute(me);",

"// Click to change geometry."

];

const terminal = document.getElementById("text");

let line = 0;

let char = 0;

function typeWriter() {

if (line < lyrics.length) {

if (char < lyrics[line].length) {

terminal.innerHTML += lyrics[line].charAt(char);

char++;

setTimeout(typeWriter, Math.random() \* 100 + 50);

} else {

terminal.innerHTML += "<br>> ";

line++;

char = 0;

setTimeout(typeWriter, 500);

}

}

}

setTimeout(typeWriter, 1000);

// ===== 5. 点击交互（切换几何体动画） =====

window.addEventListener("click", () => {

meshes.forEach(mesh => {

mesh.rotation.x += Math.random() \* 0.2;

mesh.rotation.y += Math.random() \* 0.2;

});

});

// ===== 6. 动画循环 =====

function animate() {

requestAnimationFrame(animate);

// 旋转几何体

meshes.forEach(mesh => {

mesh.rotation.x += 0.01;

mesh.rotation.y += 0.01;

});

// 音频可视化影响几何体

if (analyser) {

analyser.getByteFrequencyData(dataArray);

const bass = dataArray[0] / 255;

meshes.forEach(mesh => {

mesh.scale.set(1 + bass, 1 + bass, 1 + bass);

});

}

renderer.render(scene, camera);

}

animate();

// 窗口大小调整

window.addEventListener("resize", () => {

camera.aspect = window.innerWidth / window.innerHeight;

camera.updateProjectionMatrix();

renderer.setSize(window.innerWidth, window.innerHeight);

});

</script>

</body>

</html>