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

const context = canvas.getContext('2d');

// Set Canvas Width and Height to Full Screen

canvas.width = window.innerWidth;

canvas.height = window.innerHeight;

let w, h, balls = [];

let mouse = {

    x: undefined,

    y: undefined

}

function init() {

    resizeReset();

    animationLoop();

}

function resizeReset() {

    w = canvas.width = window.innerWidth;

    h = canvas.height = window.innerHeight;

}

function animationLoop() {

    context.clearRect(0, 0, w, h);

    //

    if (mouse.x !== undefined && mouse.y !== undefined) {

        balls.push(new Ball());

    }

    drawBalls();

    requestAnimationFrame(animationLoop);

}

//we call the functions update and draw that are defined in the class Ball

function drawBalls() {

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

        balls[i].update();

        balls[i].draw();

    }

}

// e is an event that occurs when the mouse is moved

function mousemove(e) {

    mouse.x = e.x;

    mouse.y = e.y;

}

// when mouse is not hovering, the x and y positions are no defined

function mouseout() {

    mouse.x = undefined;

    mouse.y = undefined;

}

//using constructor inside the class - it sets up initial information of the object

class Ball {

    constructor() {

        this.x = mouse.x;

        this.y = mouse.y;

        this.size = 20;

        this.style = "rgba(255,255,255,0.5)";

    }

    draw() {

        context.fillStyle = this.style; //filling the background with the syle defined in constuctor

        context.beginPath(); //to start a new path on the canvas

        context.arc(this.x, this.y, this.size, 0, Math.PI \* 2); // to draw circles, 0 is rightmost part of circle, pi x 2 is for a full circle

        context.closePath(); //calling function closePath

        context.fill();

    }

    update() {

    }

}

window.addEventListener("DOMContentLoaded", init);

window.addEventListener("reSize", resizeReset);

window.addEventListener("mousemove", mousemove);

window.addEventListener("mouseout", mouseout);

;

A screen shot of a video game

Description automatically generated