**OP-Art – Programmierung**

Ein einfacher Würfel mit Schatteneffekt. Programiert in Javasript (js) mit einem p5.js Editor.

**Erklärung:**

function setup() {

createCanvas(400, 400, WEBGL);

}

function draw() {

background(220);

// Kamera-Steuerung mit Maus

orbitControl();

// Licht hinzufügen

directionalLight(255, 255, 255, 0, 0, -1);

ambientLight(100);

// Würfel-Styling

fill(135, 206, 250); // Hellblauer Farbton

noStroke();

// Rotation

rotateX(frameCount \* 0.01);

rotateY(frameCount \* 0.01);

// Würfel zeichnen

box(100);

}

1. **WEBGL-Modus:**
   * Ermöglicht 3D-Operationen
   * Koordinatensystem ist im Zentrum der Leinwand (0,0,0)
2. **Lichtberechnung:**
   * directionalLight() erzeugt Schattenwurf
   * ambientLight() verhindert komplett schwarze Schatten
   * Lichtparameter können animiert werden (z.B. mit sin()-Funktion)
3. **Rotationsgeschwindigkeit:**
   * frameCount erhöht sich automatisch um 1 pro Frame
   * 0.01 ist der Geschwindigkeitsmultiplikator
   * Beispiel: Bei 60 FPS = 60 \* 0.01 = 0.6 Radiant/Sekunde
4. **Größenanpassung:**
   * Aktueller Wert: 250 Pixel Kantenlänge
   * Kann durch Ändern des Wertes in box() angepasst werden
   * Wertebereich typisch: 50-400 Pixel (abhängig von Leinwandgröße)

Versuche mal den Würfel selber zu bewegen ;-)

**Ein weitere Versuch eine optische Täuschung zu zeigen:**

function setup() {

createCanvas(1000, 1000, WEBGL);

angleMode(DEGREES);

}

function draw() {

background(0);

orbitControl();

// Paradoxe Beleuchtung

directionalLight(255, 0, 0, -1, 1, -1); // Rotes Licht

directionalLight(0, 255, 0, 1, -1, 1); // Grünes Licht (gegensätzliche Richtung)

// Hypnotische Rotation

rotateX(frameCount \* 0.7);

rotateY(frameCount \* 0.5);

rotateZ(frameCount \* 0.3);

// Interferenzmuster mit schwebenden Quadern

noFill();

strokeWeight(2);

// Erzeuge 3 ineinandergeschachtelte Würfel mit Farbwechsel

for(let i = 0; i < 3; i++) {

push();

stroke(

map(sin(frameCount\*2 + i\*50), -1, 1, 0, 255),

map(cos(frameCount\*2 + i\*50), -1, 1, 0, 255),

map(sin(frameCount\*2 + i\*50 + 90), -1, 1, 0, 255)

);

// Pulsierende Skalierung

let scaleFactor = 1 + sin(frameCount\*0.5 + i\*30)\*0.2;

scale(scaleFactor);

// Würfel mit leicht versetzter Rotation

box(250 + i\*30);

pop();

}

}