



CodeCamp JavaScript

Algorithmics Brasov-Smart Kids

Ziua 4

**Culori, canvas și
proiect Mini Paint**

Introducere

Obiectivele zilei:

- Înțelegerea modului de lucru cu **culorile în HTML, CSS și JS**
- Crearea de **interacțiuni cu utilizatorul** folosind evenimente mouse și input-uri
- Introducerea desenului pe **canvas** și a conceptului de **interactivitate vizuală**

Desen pe Canvas – Cercuri, Culori, Animații Simple

Ce este canvas-ul?

- <canvas> este un element HTML care permite **desenarea de grafică 2D în browser**.
- Poate fi folosit pentru **desen liber, animații, jocuri sau aplicații interactive**.
- Pentru a desena, avem nevoie de **contextul 2D**:

Cod:

```
const canvas = document.getElementById('myCanvas');  
const ctx = canvas.getContext('2d');
```

Desen pe Canvas – Cercuri, Culori, Animații Simple

Desenarea cercurilor

- Cercurile sunt desenate cu metoda `ctx.arc(x, y, radius, 0, Math.PI*2)`
- `x` și `y` – coordonatele centrului cercului
- `radius` – raza cercului
- `ctx.fillStyle` – stabilește culoarea cercului
- `ctx.fill()` – umple cercul cu culoarea selectată

Cod:

```
ctx.beginPath();  
ctx.arc(100, 100, 30, 0, Math.PI * 2);  
ctx.fillStyle = '#ff0000';  
ctx.fill();
```

Desen pe Canvas – Cercuri, Culori, Animații Simple

Culori dinamice

- Culorile pot fi alese de utilizator prin input de tip **color picker**:

JavaScript:

```
const color = document.getElementById('colorPicker').value;  
ctx.fillStyle = color;
```

HTML:

```
<label for="colorPicker">Alege culoarea:</label>  
<input type="color" id="colorPicker" value="#ff0000">
```

Desen pe Canvas – Cercuri, Culori, Animații Simple

Interactivitate – desen cu mouse

- Evenimente mouse importante: mousedown, mousemove, mouseup, mouseleave
- Conceptul: desenul se face doar când mouse-ul este apăsat

Cod:

```
canvas.addEventListener('mousemove', (e) => {  
  if(drawing) drawCircle(e);  
});
```

Desen pe Canvas – Cercuri, Culori, Animații Simple

Animații simple

Desenele pot fi actualizate continuu pentru efecte vizuale
Ex: schimbarea culorii sau a poziției în timp real

Metode utile:

- `ctx.clearRect()` – curățarea canvas-ului
- `requestAnimationFrame()` – creare animații fluide

Aplicabil pentru efecte dinamice sau mini-jocuri

Mini Paint Interactiv

Mini Paint Interactiv

Alege culoarea: Dimensiune cerc: Mic - detalii ▼



Sterge tot

Mini Paint Interactiv

Alege culoarea: Dimensiune cerc: Mic - detalii ▼



Sterge tot

Mini Paint Interactiv

1. Obiectiv

Crearea unei aplicații web pentru desen liber pe canvas, cu culori și dimensiuni diferite ale cercurilor.

2. Descriere

- Utilizatorul poate **desena pe canvas** folosind cercuri de diferite dimensiuni și culori.
- Include cerc mic pentru detalii și cercuri mai mari pentru desen general.
- Canvas-ul poate fi resetat.

3. Funcționalități

- Alegerea culorii cercurilor (color picker).
- Selectarea dimensiunii cercurilor (mic, detalii, mediu, mare).
- Desen liber pe canvas cu mouse-ul.
- Ștergerea întregului desen cu un buton.

4. Tehnologii

- HTML, CSS, JavaScript.
- <canvas> 2D pentru desen.
- Evenimente mouse: mousedown, mousemove, mouseup, mouseleave.
- Compatibil cu browsere moderne.

HTML: Structura paginii

Conține:

- `<h1>` – titlul paginii.
- `<label>` și `<input type="color">` – picker pentru alegerea culorii.
- `<select>` – selectarea dimensiunii cercului (pensule diferite).
- `<canvas>` – zona pe care desenăm.
- `<button>` – resetarea canvas-ului.

Notă: Codul HTML este simplu, doar body, deoarece folosim CodePen.

HTML

```
1 <h1>Mini Paint Interactiv</h1>
2
3 <label for="colorPicker">Alege culoarea:</label>
4 <input type="color" id="colorPicker" value="#ff0000">
5
6 <label for="sizePicker">Dimensiune cerc:</label>
7 <select id="sizePicker">
8   <option value="10">Mic - detalii</option>
9   <option value="20">Mic</option>
10  <option value="40" selected>Mediu</option>
11  <option value="60">Mare</option>
12 </select>
13
14 <canvas id="myCanvas" width="800" height="500"></canvas>
15 <button id="clearBtn">Șterge tot</button>
```

CSS: Stilizarea paginii

Conține:

- font-family și text-align – definesc aspectul textului și alinierea elementelor pe pagină, pentru a oferi un stil uniform și lizibil.
- canvas – are o margine vizibilă pentru a delimita zona de desen, un spațiu de sus (margin) pentru separare față de celelalte elemente și un cursor de tip crosshair, astfel încât utilizatorul să știe exact unde va desena.
- input, select, button – au margin și padding aplicate pentru ca elementele să fie spațiate corect și ușor de folosit, oferind o interfață prietenoasă și intuitivă.

* CSS

```
1 * body {  
2   font-family: Arial, sans-serif;  
3   text-align: center;  
4   margin: 20px;  
5 }  
6  
7 * canvas {  
8   border: 2px solid black;  
9   margin-top: 20px;  
10  cursor: crosshair;  
11 }  
12  
13 * input, select, button {  
14   margin: 10px;  
15   padding: 5px;  
16   font-size: 14px;  
17 }
```

JavaScript: Selectarea elementelor

- `document.getElementById('myCanvas')` – obținem referința la elementul `<canvas>` din HTML, ca să putem desena pe el cu JavaScript.
- `canvas.getContext('2d')` – creăm un **context 2D** pentru canvas; acesta ne permite să desenăm forme, linii, cercuri și alte elemente grafice.
- `document.getElementById('colorPicker')` – legăm input-ul de culoare, ca să putem afla ce culoare a ales utilizatorul.
- `document.getElementById('sizePicker')` – legăm select-ul de dimensiune, pentru a afla cât de mare să fie cercul desenat.
- `document.getElementById('clearBtn')` – legăm butonul de reset, pentru a șterge tot ce e desenat pe canvas când este apăsat.

JS

```
1 const canvas = document.getElementById('myCanvas');  
2 const ctx = canvas.getContext('2d');  
3 const colorPicker = document.getElementById('colorPicker');  
4 const sizePicker = document.getElementById('sizePicker');  
5 const clearBtn = document.getElementById('clearBtn');
```

Idee-cheie: aici transformăm elementele vizuale din HTML în **variabile JavaScript** pentru a le putea controla programatic.

JavaScript: Gestionarea desenului

1.let drawing = false; – variabilă care ține evidența dacă utilizatorul ține apăsat mouse-ul pe canvas.

2.mousedown – eveniment care apare când apăsăm butonul mouse-ului pe canvas:

- Setăm drawing = true ca să știm că desenarea a început.
- Apelăm drawCircle(e) pentru a desena primul punct.

3.mousemove – eveniment care apare când mutăm mouse-ul:

- Dacă drawing = true, desenăm un cerc la fiecare mișcare, astfel încât să creăm linii continue.

4.mouseup și mouseleave – evenimente care apar când eliberăm mouse-ul sau cursorul părăsește canvas-ul:

- Setăm drawing = false pentru a opri desenul.

Idee-cheie: desenarea se face doar când mouse-ul este apăsat, iar mutarea mouse-ului creează cercuri succesive pentru a simula o pensulă.

```
let drawing = false;
8
9 // Începe desenarea la mouse down
10 canvas.addEventListener('mousedown', (e) => {
11   drawing = true;
12   drawCircle(e);
13 });
14
15 // Continuă desenarea când mouse-ul se mișcă
16 canvas.addEventListener('mousemove', (e) => {
17   if (drawing) drawCircle(e);
18 });
19
20 // Oprește desenarea la mouse up sau când mouse-ul părăsește canvas-ul
21 canvas.addEventListener('mouseup', () => drawing = false);
22 canvas.addEventListener('mouseleave', () => drawing = false);
```

JavaScript: Funcția drawCircle()

- **canvas.getBoundingClientRect()** – aflăm poziția canvas-ului pe pagină, ca să știm coordonatele exacte relative la acesta.
- **e.clientX - rect.left** și **e.clientY - rect.top** – calculăm **poziția mouse-ului în interiorul canvas-ului**, nu doar în fereastra browser-ului.
- **radius = parseInt(sizePicker.value)** – luăm dimensiunea cercului selectată de utilizator și o transformăm în număr.
- **color = colorPicker.value** – luăm culoarea selectată de utilizator.
- **ctx.beginPath()** – începem desenarea unui nou obiect (cercul).
- **ctx.arc(x, y, radius, 0, Math.PI * 2)** – desenăm cercul la coordonatele calculate, cu raza selectată.
- **ctx.fillStyle = color** – setăm culoarea cercului.
- **ctx.fill()** – umplem cercul cu culoarea aleasă.

```
// Funcție desenare cerc
25 * function drawCircle(e) {
26     const rect = canvas.getBoundingClientRect();
27     const x = e.clientX - rect.left;
28     const y = e.clientY - rect.top;
29     const radius = parseInt(sizePicker.value);
30     const color = colorPicker.value;
31
32     ctx.beginPath();
33     ctx.arc(x, y, radius, 0, Math.PI * 2);
34     ctx.fillStyle = color;
35     ctx.fill();
36 }
```

JavaScript: Butonul „Șterge tot”

- **`addEventListener('click', ...)`** – adăugăm o acțiune care se declanșează când utilizatorul apasă butonul.
- **`ctx.clearRect(0, 0, canvas.width, canvas.height)`** – șterge tot conținutul canvas-ului: de la coordonata (0,0) până la dimensiunea totală (`canvas.width` x `canvas.height`).

```
38 // Buton pentru ștergere canvas
39 * clearBtn.addEventListener('click', () => {
40     ctx.clearRect(0, 0, canvas.width, canvas.height);
41 });
42
```


Îmbunătățiri posibile

👉 Idei:

1. Adaugă o **gumă de șters** 🩹
2. Pune un buton „**Salvează desenul**” 💾

Adaugă o gumă de șters



Cum facem?

- Folosim aceeași funcție de desenare
- Culoarea = alb (ca fundalul)

```
15 |  
16 * <button id="eraserBtn">Gumă de șters</button>  
17
```

```
33  
34 const eraserBtn = document.getElementById('eraserBtn');  
35 * eraserBtn.addEventListener('click', () => {  
36   colorPicker.value = "#ffffff"; // alb = fundal  
37 });  
38
```

Pune un buton „Salvează desenul”

💡 Cum facem?

- Canvas-ul poate fi transformat în imagine (PNG)
- Descarcăm fișierul

```
18  
19  
20 <button id="saveBtn">Salvează desenul</button>  
21
```

```
38  
39 const saveBtn = document.getElementById('saveBtn');  
40 saveBtn.addEventListener('click', () => {  
41   const link = document.createElement('a');  
42   link.download = 'desenul_meu.png';  
43   link.href = canvas.toDataURL();  
44   link.click();  
45 });  
46
```

Ce am învățat azi?

Recapitulare



- ❑ **HTML & CSS:** canvas, input, select, button; stilizare, cursor crosshair
- ❑ **JavaScript:** variabile, funcții, getElementById, evenimente mouse
- ❑ **Canvas 2D:** arc(), fillStyle, fill(), coordonate relative
- ❑ **Interactivitate:** desen doar când mouse-ul e apăsat, schimbare culori/dimensiuni
- ❑ **Mini-proiect:** desen liber, cerc mic pentru detalii, reset canvas