Manipulació directa de pixels (ImageData)

Aquest és un tema avançat i molt potent dins de Canvas: la **manipulació directa de píxels**. Aquesta tècnica permet realitzar filtres d'imatge, efectes visuals complexos i anàlisi de colors amb una gran precisió i rendiment. La clau per accedir a les dades pures d'una imatge o d'una àrea del Canvas és l'objecte `ImageData`.

L'Objecte `ImageData`

És un objecte simple que conté les dades brutes de la imatge d'un rectangle específic del Canvas. Les seves propietats són:

- `amplada`: L'amplada del rectangle en píxels.
- `alçada`: L'alçada del rectangle en píxels.
- `data`: Una matriu unidimensional anomenada `Uint8ClampedArray` que conté els valors dels píxels.

L'Array `data` i el Model RGBA

Cada píxel es representa per **quatre valors consecutius** (bytes) en la matriu `data`: **Vermell** (R), **Verd** (G), **Blau** (B) i **Alfa** (A).

- **Valors:** Cada component (R, G, B, A) pren un valor entre **0 i 255**.
  - **0:** Representa l'absència (per als colors) o total transparència (per a l'Alfa).
  - **255:** Representa la màxima intensitat de color o la total opacitat (per a l'Alfa).
- Mida de la Matriu: La llargada total de l'array data és igual a:

$$\text{Amplada X Alçada X 4}$$

- Accés a un Píxel: Per accedir a la component Vermella del píxel a la posició $ \mathbf{(x, y)}$, cal calcular l'índex:

$$\text{Índex} = (y \text{ X Amplada } + x) \text{ X } 4$$

Mètodes

Mètode	Descripció	Ús
`ctx.getImageData(x, y, a, h)`	**Llegir dades.** Copia i retorna un objecte `ImageData` de l'àrea especificada del Canvas.	Per aplicar filtres a una imatge ja dibuixada.
`ctx.createImageData(a, h)`	**Crear dades.** Retorna un objecte `ImageData` buit (tots els valors RGBA a 0, és a dir, negre transparent).	Per crear imatges des de zero, com un patró o un gràfic de dades.

Mètode	Descripció	Ús
`ctx.putImageData(imgData, x, y)`	**Dibuixar dades.** Dibuixa el contingut de l'`imgData` al Canvas a la posició $\mathbf{(x, y)}$.	Per mostrar les dades de píxels modificades.

Exemple: Creació quadre monocrom

Aquest exemple millora el teu codi, utilitzant variables en català i l'índex correcte de la matriu.

HTML

```
<canvas width="250" height="120" id="llencolPixels" style="border: 1px solid #ddd;"></
canvas>

<script>
    const llencol = document.getElementById("llencolPixels");
    const ctx = llencol.getContext("2d");

    if (ctx) {
        const ampladaQuadre = 100;
        const alçadaQuadre = 100;

        // 1. Crear un objecte ImageData en blanc (negre transparent)
        const dadesImatge = ctx.createImageData(ampladaQuadre, alçadaQuadre);
        const arrayPixels = dadesImatge.data;

        // 2. Modificar el valor dels components RGBA per a cada píxel (Omplir de
Vermell)
        // Recorrem la matriu de 4 en 4 (una iteració per a cada píxel)
        for (let i = 0; i < arrayPixels.length; i += 4) {
            // [i+0] = Vermell
            arrayPixels[i + 0] = 255;

            // [i+1] = Verd
            arrayPixels[i + 1] = 0;

            // [i+2] = Blau
            arrayPixels[i + 2] = 0;

            // [i+3] = Alfa (Opacitat)
            arrayPixels[i + 3] = 255;
        }

        // 3. Col·locar la nova imatge al canvas
        const posX = 75;
```

```
        const posY = 10;
        ctx.putImageData(dadesImatge, posX, posY);
    }
</script>
```

Exemple: Filtre escala de grisos (sepia)

Aquest és l'ús més comú i interessant: llegir una imatge, aplicar una transformació complexa a nivell de píxel i tornar-la a dibuixar. La lògica del Filtre: Per convertir un píxel a escala de grisos, es calcula la mitjana ponderada (o mitjana simple) dels seus components de color:

$$\text{Nou Color} = (\text{R} + \text{G} + \text{B})/3$$

HTML

```
<canvas id="filtreCanvas" width="400" height="200" style="border: 1px solid #ddd;"></
canvas>

<script>
    const canvasFiltre = document.getElementById("filtreCanvas");
    const ctxFiltre = canvasFiltre.getContext("2d");

    // NOTA: La imatge ha de ser del mateix domini (Same-Origin Policy)!
    const imatgeOriginal = new Image();
    imatgeOriginal.src = "https://mdn.dev/patterns-canvas.png"; // Substitueix per una
imatge real i segura

    imatgeOriginal.onload = function() {
        // 1. Dibuixa la imatge original per poder-ne extreure les dades
        ctxFiltre.drawImage(imatgeOriginal, 0, 0, 200, 200);

        // 2. LLEGIR les dades del Canvas (Quadre de 0,0 a 200x200)
        const dadesOriginals = ctxFiltre.getImageData(0, 0, 200, 200);
        const arrayPixels = dadesOriginals.data;

        // 3. APLICAR el Filtre (Escala de Grisos)
        for (let i = 0; i < arrayPixels.length; i += 4) {
            const R = arrayPixels[i + 0];
            const G = arrayPixels[i + 1];
            const B = arrayPixels[i + 2];

            // FÓRMULA: Mitjana simple dels tres components de color
```

```
            const mitjana = (R + G + B) / 3;

            // Assignem la nova mitjana a R, G, i B
            arrayPixels[i + 0] = mitjana; // Nou Vermell
            arrayPixels[i + 1] = mitjana; // Nou Verd
            arrayPixels[i + 2] = mitjana; // Nou Blau
            // L'Alpha (arrayPixels[i+3]) es manté
        }

        // 4. DIBUIXAR la nova imatge transformada (a la dreta)
        ctxFiltre.putImageData(dadesOriginals, 200, 0);

        ctxFiltre.font = '14px Arial';
        ctxFiltre.fillText("Original", 60, 15);
        ctxFiltre.fillText("Escala de Grisos", 240, 15);
    };
</script>
```

Aquest mètode és la base per a funcions molt més complexes com la detecció de vores (amb algorismes com Sobel), la inversió de colors o la manipulació d'imatges en temps real.

Revisió núm. 1
★Admin l’ha creat 2025-11-19 11:10:03 UTC
Admin l’ha actualitzat 2025-11-19 11:14:51 UTC