Canvas Advanced

Adding Images

```
const img = new Image();
img.src = 'image.jpg';
img.onload = () ⇒ ctx.drawImage(img, 50, 50, 100, 100);
```

- `drawImage(image, x, y, width, height)` draws an image.

Gradients

```
const gradient = ctx.createLinearGradient(0, 0, 200, 0);
gradient.addColorStop(0, 'red');
gradient.addColorStop(1, 'blue');
ctx.fillStyle = gradient;
ctx.fillRect(50, 50, 200, 100);
```

- `createLinearGradient(x0, y0, x1, y1)` creates a gradient.
- `addColorStop(offset, color)` adds colors to the gradient.

Patterns

```
const pattern = ctx.createPattern(img, 'repeat');
ctx.fillStyle = pattern;
ctx.fillRect(0, 0, 300, 300);
```

- `createPattern(image, repetition)` creates a pattern.

Transformations

```
ctx.translate(100, 100);
ctx.rotate(Math.PI / 4);
ctx.fillRect(0, 0, 50, 50);
```

- `translate(x, y)` moves the canvas origin.
- `rotate(angle)` rotates the canvas.

Saving and Restoring State

```
ctx.save();
ctx.fillStyle = 'green';
ctx.fillRect(50, 50, 100, 100);
ctx.restore();
```

- `save()` saves the current state.
- `restore()` restores the last saved state.

Animations

```
let x = 0;
function animate() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
    ctx.fillRect(x, 50, 50, 50);
    x += 2;
    requestAnimationFrame(animate);
}
animate();
```

- Use `requestAnimationFrame` for smooth animations.

Event Handling

```
canvas.addEventListener('click', (e) => {
    const rect = canvas.getBoundingClientRect();
    const x = e.clientX - rect.left;
    const y = e.clientY - rect.top;
    console.log(`Clicked at: ${x}, ${y}`);
});
```

- Use event listeners to interact with the canvas.

Final Project: Interactive Drawing App

Features

- Draw lines with the mouse.
- Change colors and line width.
- Clear the canvas.

Final Project: HTML Structure

```
<canvas id="drawingCanvas" width="800" height="600"></canvas>
<div>
    <button id="clearCanvas">Clear</button>
    <input type="color" id="colorPicker" />
    <input type="range" id="lineWidth" min="1" max="10" />
</div>
```

Final Project: JavaScript Setup

```
const canvas = document.getElementById('drawingCanvas');
const ctx = canvas.getContext('2d');
let drawing = false;

canvas.addEventListener('mousedown', () ⇒ drawing = true);
canvas.addEventListener('mouseup', () ⇒ drawing = false);
canvas.addEventListener('mousemove', draw);
```

Final Project: Drawing Logic

```
function draw(e) {
    if (!drawing) return;
    const rect = canvas.getBoundingClientRect();
    const x = e.clientX - rect.left;
    const y = e.clientY - rect.top;
    ctx.lineTo(x, y);
    ctx.stroke();
    ctx.beginPath();
    ctx.moveTo(x, y);
}
```

Final Project: Color and Line Width

```
const colorPicker = document.getElementById('colorPicker');
const lineWidth = document.getElementById('lineWidth');

colorPicker.addEventListener('input', (e) => {
    ctx.strokeStyle = e.target.value;
});

lineWidth.addEventListener('input', (e) => {
    ctx.lineWidth = e.target.value;
});
```

Final Project: Clear Canvas

```
document.getElementById('clearCanvas').addEventListener('click', () => {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
});
```