DESENHO TÉCNICO E COMPUTAÇÃO GRÁFICA

Ms Márcio J Morais

GRANTIETÊ FACULDADE

Engenharia da Computação

FACULDADE GRAN TIETÊ

Av. 15 de Novembro, 125 - Centro - Barra Bonita/SP - CEP: 17340-047

(14) 3642-3219

contato@grantiete.com.br

https://grantiete.com.br/

Professor Mestre Márcio Jesus de Morais

marciojmorais@gmail.com

Bacharelado em Engenharia Agronômica

Desenho Técnico e Expressão Gráfica

1.º Bimestre – 04/08 a 09/04

2.º Bimestre – 10/04 a 27/06

Turmas/Termos

Eng. Comp.: 1.º, 2.º, 3.º; 4.º, 5.º, 6.º, 8.º, 9.º e 10.º

Eng. Civil: 4.º, 5.º e 6.º

Terças-feiras - 19h00 – 22h30 - Sala Allan Turing

AULA 02 - Criando seu Primeiro Grid

1. Objetivos da Aula

- Aprender a desenhar linhas no Canvas
- Compreender os comandos básicos de desenho
- Dominar loops para desenho repetitivo
- Criar um grid estático com valores fixos

Comandos Básicos de Desenho

O Processo de Desenhar uma Linha:

```javascript
const ctx = canvas.getContext('2d');

// 1. Começar um novo caminho
ctx.beginPath();

// 2. Mover para ponto inicial (sem desenhar)
ctx.moveTo(x1, y1);

// 3. Desenhar linha até ponto final
ctx.lineTo(x2, y2);

// 4. Aplicar o desenho na tela
ctx.stroke();
```

Analogia com Desenho Real:

beginPath() = Preparar uma nova linha

moveTo() = Posicionar a caneta sem desenhar

lineTo() = Desenhar linha até o destino

stroke() = "Finalizar" o traço na tela

Comandos Detalhados

1. beginPath()

O que faz: Inicia um novo caminho de desenho

Quando usar: Antes de começar qualquer nova forma

Por que importante: Separa desenhos diferentes

```javascript
ctx.beginPath(); // "Vou começar a desenhar algo novo"
```

2. moveTo(x, y)

O que faz: Move o "cursor" para uma posição sem desenhar

Parâmetros: x (horizontal), y (vertical)

Analogia: Levantar a caneta e posicionar em outro lugar

```javascript
ctx.moveTo(100, 50); // Vai para posição (100, 50)
```

3. lineTo(x, y)

O que faz: Desenha linha da posição atual até o ponto especificado

Parâmetros: x (destino horizontal), y (destino vertical)

Analogia: Desenhar com a caneta pressionada

```javascript
ctx.lineTo(200, 150); // Desenha linha até (200, 150)
```

4. stroke()

O que faz: Aplica/renderiza o desenho na tela

Importante: Sem isso, nada aparece!

Analogia: "Revelar" o desenho

```javascript
ctx.stroke(); // Mostra tudo que foi desenhado
```

Loops para Desenho Repetitivo

Grids são formados por muitas linhas. Ao invés de escrever código para cada linha, usamos loops:

Loop For Básico:

```javascript
for (let i = 0; i < 5; i++) {
    console.log("Linha número:", i);
}
```

Loop para Desenhar Linhas Verticais:

```javascript
for (let x = 0; x <= canvas.width; x += 50) {
    ctx.beginPath();
    ctx.moveTo(x, 0);           // Topo da tela
    ctx.lineTo(x, canvas.height); // Base da tela
    ctx.stroke();
}
```

Como Funciona:

x = 0: Primeira linha na posição x=0

x += 50: Próxima linha 50 pixels à direita

x <= canvas.width: Para quando passar da largura

Criando um Grid Completo

Grid com Espaçamento Fixo de 50 pixels:

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

// Configurar cor da linha
ctx.strokeStyle = '#cccccc'; // Cinza claro
ctx.lineWidth = 1;           // Espessura 1 pixel

// LINHAS VERTICAIS
for (let x = 0; x <= canvas.width; x += 50) {
    ctx.beginPath();
    ctx.moveTo(x, 0);
    ctx.lineTo(x, canvas.height);
    ctx.stroke();
}

// LINHAS HORIZONTAIS
for (let y = 0; y <= canvas.height; y += 50) {
    ctx.beginPath();
    ctx.moveTo(0, y);
    ctx.lineTo(canvas.width, y);
    ctx.stroke();
}
```

Exemplo Completo

```html
<!DOCTYPE html>
<html>
<head>
    <title>Grid Estático - Aula 2</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        canvas {
            border: 2px solid #333;
            background-color: #f9f9f9;
        }
    </style>
</head>
<body>
    <h1>Computação Gráfica - Aula 2: Grid Estático</h1>

    <canvas id="areaDesenho" width="800" height="600"></canvas>

    <div>
        <p><strong>Especificações do Grid:</strong></p>
        <ul>
            <li>Espaçamento: 50 pixels</li>
            <li>Cor: Cinza claro (#cccccc)</li>
            <li>Espessura: 1 pixel</li>
        </ul>
    </div>

    <script>
        // Obter canvas e contexto
        const canvas = document.getElementById('areaDesenho');
        const ctx = canvas.getContext('2d');

        // Função para desenhar o grid
        function desenharGrid() {
            // Limpar canvas
            ctx.clearRect(0, 0, canvas.width, canvas.height);

            // Configurar estilo das linhas
            ctx.strokeStyle = '#cccccc';
            ctx.lineWidth = 1;

            // Espaçamento fixo
            const espacamento = 50;

            // Desenhar linhas verticais
            for (let x = 0; x <= canvas.width; x += espacamento) {
                ctx.beginPath();
                ctx.moveTo(x, 0);
                ctx.lineTo(x, canvas.height);
                ctx.stroke();
            }

            // Desenhar linhas horizontais
            for (let y = 0; y <= canvas.height; y += espacamento) {
                ctx.beginPath();
                ctx.moveTo(0, y);
                ctx.lineTo(canvas.width, y);
                ctx.stroke();
            }

            console.log('Grid desenhado com espaçamento de', espacamento, 'pixels');
        }

        // Desenhar o grid
        desenharGrid();
    </script>
</body>
</html>
```

Conceitos Importantes

1. Por que clearRect()?

```javascript
ctx.clearRect(0, 0, canvas.width, canvas.height);
```

Função: Limpa uma área retangular do canvas

Uso: Remover desenhos anteriores

Parâmetros: x, y, largura, altura da área a limpar

2. Propriedades de Estilo

```javascript
ctx.strokeStyle = '#cccccc'; // Cor das linhas
ctx.lineWidth = 1;           // Espessura das linhas
```

3. Operador +=

```javascript
x += 50  // É o mesmo que: x = x + 50
```

4. Loops e Performance

- 16 linhas verticais (800 ÷ 50 = 16)
- 13 linhas horizontais (600 ÷ 50 = 12 + 1)
- Total: 29 linhas com apenas 2 loops!

Matemática do Grid

Calculando Quantidade de Linhas:

```javascript
const largura = 800;
const altura = 600;
const espacamento = 50;

const linhasVerticais = Math.floor(largura / espacamento) + 1;
const linhasHorizontais = Math.floor(altura / espacamento) + 1;

console.log('Linhas verticais:', linhasVerticais);   // 17
console.log('Linhas horizontais:', linhasHorizontais); // 13
```

Exercícios Práticos

Exercício 1: Grid Mais Denso

Modifique o código para criar um grid com espaçamento de 25 pixels.

Exercício 2: Grid Colorido

Crie um grid com linhas azuis (#0066cc).

Exercício 3: Linhas Mais Grossas

Faça um grid com espessura de linha 2 pixels.

Exercício 4: Grid Duplo

Crie um grid com linhas principais (espaçamento 100px, cor escura) e secundárias (espaçamento 25px, cor clara).

Resumo da Aula

O que aprendemos:

- Comandos básicos: beginPath(), moveTo(), lineTo(), stroke()
- Loops for para desenho repetitivo
- Propriedades de estilo: strokeStyle, lineWidth
- Como limpar o canvas com clearRect()
- Matemática básica para cálculo de grids

Sequência de desenho de uma linha:

beginPath() → 2. moveTo() → 3. lineTo() → 4. stroke()

GLOSSÁRIO

beginPath(): Inicia um novo caminho de desenho

moveTo(): Move cursor sem desenhar

lineTo(): Desenha linha até destino

stroke(): Renderiza o caminho na tela

clearRect(): Limpa área do canvas

strokeStyle: Cor das linhas

lineWidth: Espessura das linhas

Loop For: Estrutura para repetição controlada