Aula 18

Laços de Repetição com while (Enquanto)

Aula 8: Laços de Repetição com `while` (Enquanto)

- **Objetivo de Hoje:**
  - Sair do fluxo sequencial e aprender a fazer o programa **repetir** um bloco de código várias vezes.
- **O que é um Laço (Loop)?**
  - É uma estrutura que permite executar o mesmo código repetidamente, até que uma condição de parada seja atingida.
- **O Laço `while` (O "ENQUANTO" do VisualG):**
  - É a forma mais fundamental de repetição.
  - A ideia é simples: **ENQUANTO** uma condição for verdadeira, **FAÇA ALGO**.

As 3 Partes Essenciais de um `while` Controlado

- **Sintaxe:**

JavaScript

```
// 1. INICIALIZAÇÃO (feita ANTES do laço)
let contador = 0;

// 2. CONDIÇÃO (verificada ANTES de cada repetição)
while (contador < 5) {
  // Bloco de código a ser repetido...
  console.log(contador);

  // 3. ATUALIZAÇÃO (feita DENTRO do laço)
  contador++;
}
```

- **O Fluxo:** O laço verifica a condição, executa o código e atualiza a variável, repetindo o processo.

Cuidado! A Armadilha Mais Comum dos Laços

- **O que é um Loop Infinito?**
  - É um laço cuja condição de parada **NUNCA** é atingida. A condição é sempre `true`.
- **Como Acontece?**
  - Geralmente, por esquecer a **etapa de atualização** (o `contador++`, por exemplo).
  - Se a variável da condição nunca muda, o laço nunca termina.
- **Consequência:**
  - O programa trava, consumindo todos os recursos do processador. No navegador, a aba irá congelar.
- **Regra de Ouro:**
  - Ao criar um laço `while`, sempre se pergunte: "Como e quando esta condição se tornará `false`?"

Prática

index.html

index.html > html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Laço While(Enquanto)</h1>
    <script src="script.js"></script>
</body>
</html>
```

JS script.js > ...

```
1
2   // ============================================================
3   // 1. A SINTAXE BÁSICA DO `while`
4   // ============================================================
5   // Um laço `while` precisa de 3 coisas para funcionar corretamente em uma contagem:
6
7   console.log('--- Exemplo 1: Contando de 1 a 5 ---')
8   // 1. INICIALIZAÇÃO: Uma variável para controlar o laço, criada ANTES dele.
9   let contador = 1
10
11  // 2. CONDIÇÃO: A expressão booleana que é verificada ANTES de cada repetição.
12  while (contador <= 5) {
13    console.log('Contagem:', contador)
14
15    // 3. ATUALIZAÇÃO (ou incremento/decremento): A mudança na variável de controle
16    //    que garantirá que o laço eventualmente termine.
17    contador++ // contador = contador + 1
18  }
19  console.log('Laço finalizado!')
20
```

```
36  // ============================================================
37  // 3. EXEMPLO PRÁTICO: CONTADORES E ACUMULADORES
38  // ============================================================
39  // Podemos usar um laço para somar uma sequência de números.
40
41  console.log('\n--- Exemplo 3: Somando os números de 1 a 10 ---')
42  let numeroAtual = 1 // Nosso contador
43  let soma = 0 // Nosso acumulador, começa em zero.
44
45  while (numeroAtual <= 10) {
46    soma = soma + numeroAtual // Acumula o valor do número atual
47    console.log('Somando', numeroAtual, '-> Soma parcial:', soma)
48    numeroAtual++ // Incrementa o contador para ir para o próximo número
49  }
50  console.log('A soma total de 1 a 10 é:', soma)
51
```

```
53  // 4. USANDO `while` PARA VALIDAR ENTRADA DO USUÁRIO
54  // ================================================================
55  // Este é um uso muito comum: repetir uma pergunta até que o usuário
56  // digite o que esperamos. Aqui o número de repetições é desconhecido.
57
58  console.log('\n--- Exemplo 4: Jogo de adivinhar a senha ---')
59  const senhaCorreta = 'javascript123'
60  let tentativa = prompt('Digite a senha para continuar:')
61
62  while (tentativa !== senhaCorreta) {
63    alert('Senha incorreta! Tente novamente.')
64    // Pergunta de novo, atualizando a variável `tentativa`
65    tentativa = prompt('Digite a senha para continuar:')
66  }
67
68  // O código só chega nesta linha quando o laço termina, ou seja,
69  // quando o usuário acerta a senha.
70  console.log('Senha correta!')
71  alert('Bem-vindo ao sistema!')
72
73  console.log(
74    'Fim da Aula 8! Agora você pode criar programas que fazem tarefas repetitivas!'
75  )
76  // Próxima aula: `do...while`, uma pequena variação do `while` que garante a execução pelo
    menos uma vez.
```

ATIVIDADE

Faça cada exercício em um JS diferente

1. **Contagem de Pares:**

   Escreva um `while` loop que exiba no console todos os números pares de 2 até 10 (inclusive).

Faça cada exercício em um JS diferente

2. **Contagem Regressiva Simples:**

   Crie um `while` loop que faça uma contagem regressiva de 10 até 0. A cada número, exiba a contagem no console. No final, exiba "Fogo!".

Faça cada exercício em um JS diferente

3. **Repetidor de "Olá, Mundo!":**

   Peça ao usuário um número. Use um `while` loop para exibir "Olá, Mundo!" no console o número de vezes que o usuário inseriu.

Faça cada exercício em um JS diferente

4. **Adivinhe a Palavra:**

   Defina uma `const palavraSecreta = "abacaxi";`. Peça ao usuário para tentar adivinhar a palavra. Enquanto ele não acertar, continue pedindo com `prompt`. Quando ele acertar, mostre um `alert` de "Parabéns!".

Faça cada exercício em um JS diferente

Faça um jogo de adivinhação com o while, algo como roleta também é válido