JAVASCRIPT - MÓDULO 1 - AULA 7

1. VETORES EM JAVASCRIPT

Em JavaScript, um vetor (ou array) é uma estrutura de dados que permite armazenar e organizar uma coleção de valores. Os vetores são usados para armazenar múltiplos elementos em uma única variável. Cada elemento em um vetor tem um índice associado, que começa em 0 para o primeiro elemento.

1.1. Declarando e inicializando um vetor

Vetor vazio:

```
var meuVetor = []; // Cria um vetor vazio
```

Vetor com elementos:

```
var frutas = ["Maçã", "Banana", "Laranja"]
```

1.2. Acessando elementos de um vetor

Os elementos de um vetor podem ser acessados usando seus índices. Lembre-se de que os índices começam em 0. Por exemplo:

```
var primeiraFruta = frutas[0]; // Acessa o primeiro elemento (Maçã)
var segundaFruta = frutas[1];  // Acessa o segundo elemento (Banana)
```

1.3. Alterando elementos de um vetor

Você pode alterar um elemento existente em um vetor atribuindo um novo valor ao seu índice:

```
frutas[2] = "Limão"; // Altera o terceiro elemento para "Limão"
```

1.4. Adicionando elementos a um vetor

Você pode adicionar elementos a um vetor usando o método push():

```
frutas.push("Abacaxi"); // Adiciona "Abacaxi" ao final do vetor
```

1.5. Removendo elementos de um vetor

Você pode remover elementos de um vetor usando o método pop(), que remove o último elemento, ou o método splice(), que permite especificar o índice do elemento a ser removido:

```
frutas.pop(); // Remove o último elemento (no caso, "Abacaxi")
```

```
frutas.splice(1, 1); // Remove o segundo elemento (índice 1, no caso, "Banana")
```

1.6. Iterando sobre um vetor

Você pode percorrer os elementos de um vetor usando um loop for:

```
for (var i = 0; i < frutas.length; i++) {
    console.log(frutas[i]);
}
```

1.7. Exemplo completo HTML, CSS e JavaScript

Aqui está um exemplo completo de como usar vetores em JavaScript junto com HTML e CSS para exibir uma lista de frutas:

```
<!DOCTYPE html>
<html>
<head>
    <title>Vetores em JavaScript</title>
    <style>
        ul {
            list-style-type: none;
            padding: 0;
        }

        li {
            margin: 5px;
            padding: 5px;
```

```
            border: 1px solid #ccc;
            background-color: #f0f0f0;
        }
    </style>
</head>
<body>
    <h1>Minha Lista de Frutas</h1>
    <ul id="frutas-lista"></ul>

    <script>
        var frutas = ["Maçã", "Banana", "Laranja",
"Limão", "Abacaxi"];

        // Acessa o elemento UL onde as frutas serão
exibidas
                                var listaFrutas =
document.getElementById("frutas-lista");

        // Itera sobre o vetor de frutas e cria uma
lista
        for (var i = 0; i < frutas.length; i++) {
            var li = document.createElement("li");
            li.textContent = frutas[i];
            listaFrutas.appendChild(li);
        }
    </script>
</body>
</html>
```

Neste exemplo, criamos uma lista de frutas usando um vetor e a exibimos em uma página HTML. Cada fruta é representada por um elemento de lista <li> e estilizada com CSS.

2. DOCUMENT.CREATEELEMENT()

O método document.createElement() é usado para criar um novo elemento HTML no documento. Você pode criar elementos como , <p>, , <ul>, <li>, entre outros. Este método não insere automaticamente o elemento na página; você precisa anexá-lo a um elemento existente usando appendChild ou outros métodos.

Exemplo:

```
// Cria um novo elemento de parágrafo <p>
    var paragrafo = document.createElement("p");

// Você pode definir atributos para o elemento, se necessário
paragrafo.setAttribute("id", "paragrafo1");

// Agora você tem um elemento <p> criado, mas ele não está na página ainda
```

3. TEXTCONTENT

A propriedade textContent é usada para definir ou obter o texto de um elemento HTML. Você pode usá-la para definir o conteúdo de um elemento ou para acessar o conteúdo existente. É uma maneira conveniente de manipular o texto dentro de elementos HTML.

Exemplo:

```
// Cria um novo elemento de parágrafo <p>
    var paragrafo = document.createElement("p");

// Define o conteúdo de texto do elemento
    paragrafo.textContent = "Isso é um parágrafo de exemplo.";

// O elemento agora contém o texto "Isso é um parágrafo de exemplo."
```

4. APPENDCHILD

O método appendChild é usado para anexar um elemento filho a um elemento pai. Ele adiciona o elemento filho ao final da lista de filhos do elemento pai.

Exemplo:

```
// Cria um novo elemento de parágrafo <p>
    var paragrafo = document.createElement("p");

// Define o conteúdo de texto do elemento
paragrafo.textContent = "Isso é um parágrafo de exemplo.";

// Obtém um elemento existente onde você deseja anexar o parágrafo
var divConteudo = document.getElementById("conteudo");

// Anexa o parágrafo à div como filho
divConteudo.appendChild(paragrafo);

// Agora, o parágrafo foi adicionado como filho da div "conteudo"
```

Neste exemplo, criamos um elemento de parágrafo, definimos seu conteúdo de texto e, em seguida, o anexamos como filho a um elemento div existente na página usando appendChild.

Essas três técnicas são frequentemente usadas em conjunto para criar e manipular elementos HTML dinamicamente em JavaScript. Por exemplo, você pode criar elementos com createElement, definir seu conteúdo com textContent e, em seguida, anexá-los a elementos existentes com appendChild para construir interfaces de usuário dinâmicas e interativas.

5. SETATTRIBUTE

setAttribute é um método em JavaScript que permite definir ou modificar atributos em elementos HTML de uma página web. Ele é amplamente usado para manipular elementos HTML dinamicamente. Podemos ver abaixo como ele funciona.

5.1. Sintaxe do setAttribute:

```
elemento.setAttribute(nomeDoAtributo, valorDoAtributo);
```

- **elemento:** É o elemento HTML ao qual você deseja adicionar ou modificar um atributo.
- **nomeDoAtributo:** É uma string que representa o nome do atributo que você deseja adicionar ou modificar.
- **valorDoAtributo:** É uma string que representa o valor que você deseja atribuir ao atributo.

5.2. Adicionando um Atributo

Suponha que você queira adicionar um atributo class a um elemento div:

```
var meuDiv = document.getElementById("minha-div"); // Obtém o elemento <div> por seu ID
meuDiv.setAttribute("class", "destaque"); // Define o atributo "class" com o valor
"destaque"
```

Este código define o atributo class da com o ID "minha-div" como "destaque".

5.3. Modificando um Atributo Existente

Agora, suponha que você queira modificar o atributo src de uma imagem:

```
var minhaImagem = document.getElementById("minha-imagem"); // Obtém o elemento <img> por
seu ID
minhaImagem.setAttribute("src", "nova-imagem.jpg"); // Modifica o atributo "src" com um
novo valor
```

EXEMPLO PRÁTICO UTILIZANDO TUDO QUE APRENDEMOS

```
<!DOCTYPE html>
<html>
<head>
    <title>Lista de Frutas</title>
</head>
<body>
    <h1>Lista de Frutas</h1>
                <input    type="text"    id="frutaInput"
placeholder="Digite uma fruta">
                                                <button
onclick="adicionarFruta()">Adicionar</button>

    <h2>Frutas Adicionadas:</h2>
    <ul id="frutasLista"></ul>

    <script>
```

```
        var frutas = []; // Vetor para armazenar as
frutas

        function adicionarFruta() {
                                    var   frutaInput   =
document.getElementById("frutaInput");
                var fruta = frutaInput.value.trim(); //
Remove espaços em branco no início e no fim

            if (fruta !== "") {
                 frutas.push(fruta); // Adiciona a fruta
ao vetor
                 frutaInput.value = ""; // Limpa a caixa
de texto

                // Atualiza a lista de frutas na página
                atualizarListaDeFrutas();
            }
        }

        function atualizarListaDeFrutas() {
                                  var   frutasLista   =
document.getElementById("frutasLista");
                frutasLista.innerHTML = ""; // Limpa a
lista antes de atualizar

              // Itera sobre o vetor de frutas e cria
elementos de lista para cada uma
            for (var i = 0; i < frutas.length; i++) {
                var li = document.createElement("li");
                li.textContent = frutas[i];
                frutasLista.appendChild(li);
            }
        }
```

```
    </script>
</body>
</html>
```