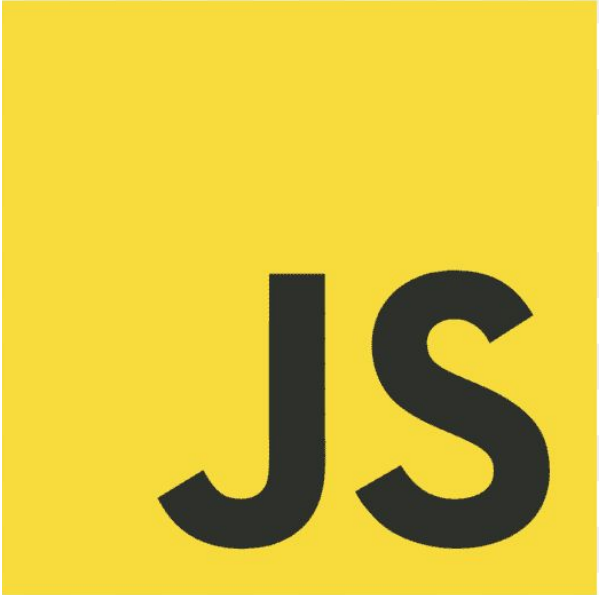
JavaScript Turbo

Dominando os seletores para código rápido e eficiente

JavaScript na Prática

Dominando os Seletores para Código Rápido e Eficiente

JavaScript é uma das linguagens mais utilizadas no desenvolvimento web, e um dos primeiros conceitos essenciais a serem dominados é a manipulação do DOM (Document Object Model). Para isso, os seletores JavaScript são ferramentas fundamentais para interagir com os elementos de uma página HTML. Este guia detalha os principais seletores e ensina como utilizá-los de maneira eficiente para acelerar seu aprendizado e otimizar seu código.



01

Selecionando elementos por ID

O `getElementById` é um dos métodos mais rápidos e eficientes para selecionar um único elemento, pois IDs são únicos dentro de um documento HTML. Esse método é amplamente utilizado quando há necessidade de acessar um elemento específico de forma direta e sem ambiguidades.

Selecionando Elementos por ID

getElementById

Como funciona:

Ao utilizar getElementById, você passa o ID do elemento como parâmetro e recebe o próprio elemento como retorno.

```
// Selecionando um elemento pelo ID e alterando sua cor
const titulo = document.getElementById("meuTitulo");
titulo.style.color = "blue";
```

Exemplo real:

Imagine que você está desenvolvendo um site de notícias e deseja modificar o título principal da página sem afetar outros elementos.

```
<h1 id="meuTitulo">Últimas Notícias</h1>
<button onclick="destacarTitulo()">Destacar</button>

<script>
  function destacarTitulo() {
    const titulo = document.getElementById("meuTitulo");
    titulo.style.color = "red";
    titulo.style.fontSize = "2em";
  }
</script>
```

Sempre que o botão for clicado, a função destacarTitulo() será acionada, modificando a cor e o tamanho do título, tornando-o mais chamativo para o usuário.

02

Selecionando elementos por classe

O método getElementsByClassName retorna uma coleção de elementos que possuem a mesma classe, permitindo manipular vários elementos simultaneamente.

Selecionando Elementos por Classe

getElementsByClassName

Como funciona:

Esse método retorna uma HTMLCollection, que se comporta como um array, mas não é exatamente um. Para iterar sobre os elementos, é necessário convertê-lo em um array ou utilizar loops como for ou forEach.

```
// Selecionando todos os elementos de uma classe e aplicando um estilo
const elementos = document.getElementsByClassName("destaque");
for (let i = 0; i < elementos.length; i++) {
    elementos[i].style.backgroundColor = "yellow";
}
```

Exemplo real:

Suponha que você esteja criando um sistema de listagem de produtos e deseja destacar todos os itens em promoção:

```
<ul>
    <li class="destaque">Produto 1 - Em Promoção!</li>
    <li>Produto 2</li>
    <li class="destaque">Produto 3 - Em Promoção!</li>
</ul>
<button onclick="destacarPromocoes()">Destacar Promoções</button>

<script>
  function destacarPromocoes() {
    const produtos = document.getElementsByClassName("destaque");
    for (let produto of produtos) {
      produto.style.border = "2px solid red";
    }
  }
</script>
```

Isso facilita a personalização em massa de elementos semelhantes sem precisar selecionar cada um individualmente.

03

Selecionando elementos por tag

O método getElementsByTagName permite selecionar todos os elementos de um determinado tipo (como <p>, ,), retornando uma coleção de elementos que podem ser manipulados.

Selecionando Elementos por Tag

getElementsByTagName

Como funciona:

Ele retorna todos os elementos da tag especificada dentro do documento ou de um elemento pai.

```
// Alterando a cor do texto de todos os parágrafos
const paragrafos = document.getElementsByTagName("p");
for (let p of paragrafos) {
    p.style.color = "gray";
}
```

Exemplo real:

Digamos que você esteja desenvolvendo um blog e queira alterar a cor de todos os parágrafos para melhorar a legibilidade.

```
<p>Primeiro parágrafo</p>
<p>Segundo parágrafo</p>
<p>Terceiro parágrafo</p>
<button onclick="alterarTexto()">Mudar Cor do Texto</button>

<script>
  function alterarTexto() {
    const paragrafos = document.getElementsByTagName("p");
    for (let p of paragrafos) {
      p.style.color = "darkblue";
    }
  }
</script>
```

Com apenas um clique, todos os parágrafos terão sua cor alterada, demonstrando o poder desse seletor.

04

Selecionando elementos com query selector

O querySelector é um dos métodos mais versáteis, permitindo selecionar qualquer elemento utilizando seletores CSS. Ele retorna apenas o primeiro elemento que corresponde ao seletor.

Selecionando Elementos com query

querySelector

Como funciona:

Ele retorna todos os elementos especificados dentro do documento.

```
// Selecionando um elemento pelo seletor CSS
const primeiroItem = document.querySelector(".lista li");
primeiroItem.style.fontWeight = "bold";
```

Exemplo real:

Se você quiser destacar o primeiro item de uma lista de tarefas

```
<ul class="lista">
    <li>Comprar pão</li>
    <li>Ir à academia</li>
    <li>Estudar JavaScript</li>
</ul>
<button onclick="destacarPrimeiro()">Destacar Primeiro</button>

<script>
  function destacarPrimeiro() {
    const primeiroItem = document.querySelector(".lista li");
    primeiroItem.style.backgroundColor = "lightgreen";
  }
</script>
```

Este método é ideal para estilizar ou modificar rapidamente um elemento específico sem precisar percorrer toda a estrutura.

AGRADECIMENTOS

OBRIGADO POR LER ATÉ AQUI

Esse Ebook foi gerado por IA, e diagramado por humano.
O passo a passo se encontra no meu Github

Esse conteúdo foi gerado com fins didáticos de construção,
não foi realizado uma validação cuidadosa humana no conteúdo e
pode conter erros gerados por uma IA.

GitHub

https://github.com/wesleysword/ebook-criado-com-prompts

Autor: Wesley Oliveira

Entre em contato: Pelo linkedin: 🔗

www.linkedin.com/in/wesley-oliveira-santos