Mudando o HTML por meio do DOM

A principal arma do DOM

Por meio do DOM, é possível que mudemos o conteúdo do HTML. O que isso significa? Significa que podemos criar, deletar e mudar elementos a fim de que o comportamento desses elementos sejam de acordo com o que a gente espera. Aqui, então, conseguimos citar mais dois conceitos: os **atributos** e as **propriedades**.

O que são os atributos?

Imagine que um elemento HTML como uma `<div>` ou `<img>` é como uma caixa. Essa caixa pode ter várias etiquetas coladas nela, que descrevem as características dessa caixa. Por exemplo, uma etiqueta pode dizer `"cor: azul"` ou `"altura: 100px"`. Essas etiquetas são como os atributos em HTML. Podem ser acessados e modificados usando métodos como **getAttribute** e **setAttribute**.

O que são as propriedades?

Ainda seguindo essa linha de raciocínio da caixa, quando o navegador lê o HTML, ele cria uma representação dessa "caixa" em uma forma que o JavaScript pode usar e entender. Essa representação é chamada de **objeto** no DOM.

O JavaScript não vê apenas as etiquetas (atributos) originais, mas também pode ver outras informações e características dessa caixa, que podem mudar ou ser manipuladas com o JavaScript.

Essas características que o JavaScript vê são chamadas de propriedades desse objeto. Por exemplo, a "caixa" (elemento HTML) poderia ter uma propriedade de cor que podemos mudar com JavaScript para "vermelho" mesmo que a etiqueta original dissesse "azul".

Principais propriedades e atributos

Lista dos principais atributos

- **`src` (em `<img>`, `<script>`, `<iframe>`, `<video>`, `<audio>`):** Especifica a URL do recurso a ser usado pelo elemento. Por exemplo, a URL de uma imagem para `<img>` ou a URL de um script para `<script>`.

- **href (em <a>, <link>):** Define o URL de destino para o link. No caso de `<a>`, é o endereço para onde o usuário será direcionado ao clicar no link. No <link>, geralmente se refere ao caminho de uma folha de estilo CSS.
- **alt (em <img>):** Fornece uma descrição alternativa para uma imagem, que é usada em leitores de tela e exibida se a imagem não puder ser carregada.
- **title (em quase todos os elementos HTML):** Fornece um texto adicional "tooltip" quando o cursor do mouse é posicionado sobre o elemento. Também pode ser usado para adicionar informações adicionais sobre o elemento.
- **id (em quase todos os elementos HTML):** Define um identificador único para o elemento, que pode ser usado para referenciar o elemento através de CSS ou JavaScript.
- class (em quase todos os elementos HTML): Especifica uma ou mais classes para o elemento, que podem ser usadas para definir estilos em CSS ou para selecionar e manipular o elemento com JavaScript.
- **type (em <input>, <button>, <script>, <style>):** Define o tipo do elemento. Por exemplo, para `<input>` pode ser text, radio, checkbox, etc.
- value (em `<input>`, `<option>`): Define o valor inicial do elemento. Para um <input> tipo texto, é o texto inicial no campo. Para um `<input>` tipo checkbox, é o valor enviado quando o checkbox está marcado.

- **name (em <input>, <form>, <fieldset>):** Especifica o nome do elemento, que é usado para identificar os dados do formulário que são enviados ao servidor.
- **placeholder (em <input>, <textarea>):** Fornece uma breve dica sobre o que o usuário deve inserir no campo. Esta dica é exibida no campo antes da entrada do usuário e desaparece quando o usuário começa a digitar.
- **disabled (em <input>, <button>, <select>, <textarea>):** Indica que o elemento está desabilitado, ou seja, não pode ser interagido pelo usuário.
- **readonly (em <input>, <textarea>):** Indica que o campo é somente leitura, ou seja, o usuário pode visualizar o valor, mas não pode modificá-lo.

Lista das principais propriedades

- **innerHTML:** Permite obter ou definir o conteúdo HTML de um elemento como uma string. Modificar essa propriedade substitui o conteúdo interno do elemento.
- **textContent:** Permite obter ou definir o conteúdo de texto de um elemento e de seus descendentes. Diferentemente de `innerHTML`, ele trata todo o conteúdo como texto bruto.

- **`style:`** Permite acessar e modificar o estilo CSS inline de um elemento. Por exemplo, `elemento.style.backgroundColor = "red"` muda a cor de fundo do elemento para vermelho.
- **`className:`** Corresponde ao atributo class HTML do elemento. Permite obter ou definir a lista de classes CSS do elemento como uma string.
- **`classList:`** Fornece métodos mais convenientes para adicionar, remover e verificar classes em um elemento. Por exemplo, `elemento.classList.add("nova-classe")`.
- **`id:`** Corresponde ao atributo id HTML do elemento. Permite obter ou definir o identificador do elemento.
- **`attributes:`** Uma coleção de todos os atributos de um elemento. Permite acessar o nome e o valor de cada atributo.
- **`children:`** Uma coleção HTML ao vivo dos elementos filhos de um elemento. Exclui nós de texto e só inclui elementos HTML.
- **`childNodes:`** Uma coleção de todos os nós filhos de um elemento, incluindo nós de texto, elementos, comentários e outros.
- **`parentNode:`** Referência o nó pai de um nó específico na árvore do DOM.
- **`nextSibling e previousSibling:`** Referenciam, respectivamente, o próximo e o anterior irmão de um nó na árvore do DOM, o que pode ser outro elemento ou um nó de texto.
- **`firstChild e lastChild:`** Referenciam, respectivamente, o primeiro e o último filho de um nó.
- **`offsetWidth e offsetHeight:`** Proporcionam as dimensões de layout (largura e altura) de um elemento, incluindo bordas, preenchimento e barras de rolagem, mas não incluindo margens.
- **`clientWidth e clientHeight:`** Similar ao `offsetWidth` e `offsetHeight`, mas não incluem bordas e barras de rolagem.
- **`scrollWidth e scrollHeight:`** Medem a largura e a altura do conteúdo de um elemento, incluindo o conteúdo que não é visível devido à rolagem.
- **`scrollTop e scrollLeft:`** Controlam ou retornam a posição atual da rolagem vertical e horizontal de um elemento.

Exemplos práticos

propriedade: style

```
var caixa = document.getElementById('minhaCaixa');  // Imagine que 'minhaCaixa' é uma div no HTML.
console.log(caixa.style.color);  // Isto iria imprimir a cor atual da 'caixa', que é uma propriedade do objeto.
caixa.style.color = 'vermelho';  // Aqui, estamos mudando a propriedade 'cor' do objeto 'caixa' para 'vermelho'.
```

atributo: src

```
<img src="imagem.jpg" alt="Descrição da imagem" title="Título da
imagem">
```

Este elemento de imagem possui os atributos como **src**, **alt**, e **title**. No JavaScript, você pode acessar e modificar esses atributos. Aqui está como você pode obter o valor do atributo src de uma imagem:

```
var imagem = document.querySelector('img');
var src = imagem.getAttribute('src');
console.log(src); // Imprime: imagem.jpg
```

Exercícios de Fixação

1. **Qual seletor do DOM é usado para selecionar todos os elementos que têm uma determinada classe CSS?**
   a) `document.getElementById()`
   b) `document.getElementsByTagName()`
   c) `document.getElementsByClassName()`
   d) `document.querySelector()`

2. **Se você deseja alterar o texto de todos os elementos `<button>` na página para "Clicado", qual seletor e método seria mais adequado?**
   a) `document.querySelector('button').textContent = "Clicado";`
   b) `document.querySelectorAll('button').forEach(btn => btn.textContent = "Clicado");`

   c) `document.getElementsByTagName('button').textContent = "Clicado";`
   d) `document.getElementById('button').textContent = "Clicado";`

3. **Qual método você usaria para selecionar o primeiro `<p>` dentro de um elemento com a classe container?**
   a) `document.querySelector('container p')`
   b) `document.querySelectorAll('.container p')[0]`
   c) `document.querySelector('.container p')`
   d)`document.getElementsByClassName('container')[0].getElementsByTagName('p')[0]`

Gabarito dos exercícios

1. c
2. b
3. c

Links

Element.attributes
Attributes and Properties (DOM)