

Manipulação do DOM usando JavaScript e exemplos práticos

1. Introdução

O DOM (Document Object Model, ou Modelo de Objeto de Documento) é uma representação em forma de árvore de todos os elementos de uma página HTML, criada automaticamente pelo navegador ao carregar o documento. Por meio do DOM, o JavaScript é capaz de acessar, modificar, adicionar e remover elementos da página de forma dinâmica, sem a necessidade de recarregar o documento inteiro. Essa capacidade é a base de praticamente toda interatividade presente em aplicações web modernas.

2. Selecionando elementos do DOM

Antes de manipular um elemento, é necessário selecioná-lo dentro da árvore do DOM. O JavaScript oferece diversos métodos para essa finalidade:

Método	Retorno
<code>getElementById(id)</code>	Um único elemento com o ID especificado
<code>querySelector(seletor)</code>	O primeiro elemento que corresponde ao seletor CSS informado
<code>querySelectorAll(seletor)</code>	Uma lista com todos os elementos que correspondem ao seletor
<code>getElementsByClassName(classe)</code>	Uma coleção de elementos com a classe especificada
<code>getElementsByTagName(tag)</code>	Uma coleção de elementos de uma determinada tag

```
const titulo = document.getElementById("titulo-principal");  
const itens = document.querySelectorAll(".item-lista");
```

3. Modificando conteúdo e estilo

Uma vez selecionado, um elemento pode ter seu conteúdo, atributos e estilos alterados dinamicamente por meio de propriedades específicas:

- **.textContent** — altera o texto exibido dentro de um elemento, sem interpretar código HTML.
- **.innerHTML** — altera o conteúdo interno de um elemento, interpretando marcações HTML inseridas.
- **.style** — permite alterar propriedades CSS diretamente pelo JavaScript, como cor e tamanho.

- **.classList** — permite adicionar, remover ou alternar classes CSS de um elemento (métodos "add", "remove" e "toggle").
- **.setAttribute() / .getAttribute()** — definem ou obtêm o valor de um atributo HTML de um elemento.

```
titulo.textContent = "Bem-vindo(a) de volta!";  
titulo.style.color = "#0f3460";  
titulo.classList.add("destaque");
```

4. Criando e removendo elementos

Além de modificar elementos já existentes, o JavaScript permite criar novos elementos dinamicamente e inseri-los na página, ou remover elementos indesejados:

```
const novoItem = document.createElement("li");  
novoItem.textContent = "Novo item da lista";  
  
const lista = document.querySelector("#minhaLista");  
lista.appendChild(novoItem);  
  
// Removendo um elemento  
novoItem.remove();
```

Nesse exemplo, o método "createElement" cria um novo elemento "", que é então adicionado à lista existente por meio do método "appendChild". O método ".remove()" permite excluir um elemento diretamente da página.

5. Exemplos de aplicação

A manipulação do DOM está presente em praticamente todas as interações dinâmicas de uma página web:

- **Listas de tarefas (to-do lists)** — novos itens são adicionados à lista por meio de "createElement" e "appendChild" à medida que o usuário os cadastra, e removidos com ".remove()" ao serem concluídos.
- **Modo escuro (dark mode)** — o método ".classList.toggle()" é utilizado para alternar uma classe CSS no elemento "<body>", alterando o esquema de cores de toda a página instantaneamente.
- **Atualização de contadores em tempo real** — o conteúdo de elementos, como o número de itens em um carrinho de compras, é atualizado dinamicamente com ".textContent" sempre que um produto é adicionado ou removido.
- **Exibição condicional de mensagens** — elementos de alerta ou confirmação são exibidos ou ocultados alterando a propriedade ".style.display" com base em ações do usuário.

- **Carregamento dinâmico de conteúdo** — dados obtidos de uma API são inseridos na página por meio de `innerHTML` ou da criação de elementos, sem a necessidade de recarregar a página inteira.
- **Galerias de imagens interativas** — clicar em uma miniatura altera dinamicamente o atributo `src` de uma imagem principal exibida em destaque, utilizando `setAttribute()` ou acesso direto à propriedade `src`.

6. Considerações finais

A manipulação do DOM é uma das habilidades mais fundamentais no desenvolvimento web com JavaScript, permitindo transformar páginas estáticas em experiências dinâmicas e interativas. Compreender como selecionar, modificar, criar e remover elementos é essencial para construir desde pequenas funcionalidades, como um contador simples, até aplicações complexas que atualizam seu conteúdo em tempo real com base nas ações do usuário.