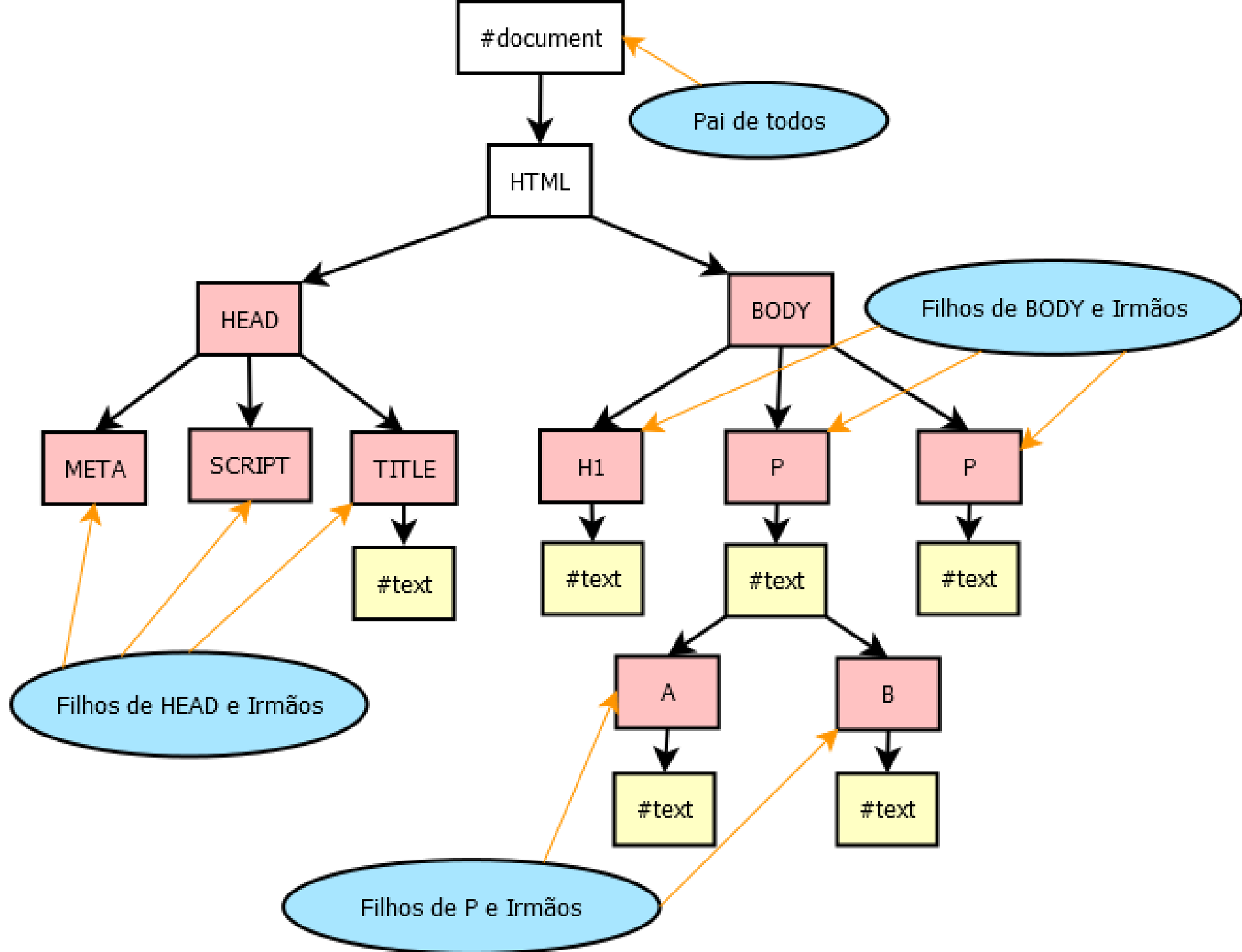


Aula 19

**Criação e Remoção de
Elementos no DOM.**



Slide 1: O Poder de Criar e Destruir! 🏗️

- **Revisão Rápida:** Sabemos selecionar elementos, manipular seu conteúdo/estilo e ouvir eventos.
- **A Missão de Hoje:** Aprender a **adicionar e remover** elementos inteiros da página, tudo com JavaScript.
- **Por que é Importante?** É assim que criamos conteúdo dinâmico! Pense em:
 - Adicionar novos itens a uma lista de tarefas.
 - Carregar e exibir comentários em um post.
 - Inserir produtos em um carrinho de compras.

Slide 2: A Fábrica de Elementos: `document.createElement()`

- **Passo 1: Construir o Elemento:** Antes de colocar algo na página, precisamos "fabricá-lo".
- **O Comando:** `document.createElement('tag')`
 - Exemplo: `const novaDiv = document.createElement('div');`
- **O Resultado:** Ele cria um elemento HTML novinho em folha, mas ele existe apenas na memória do JavaScript. Ele está "desconectado" da página, pronto para ser customizado (adicionar texto, classes, etc.).

Slide 3: Colocando Elementos na Página 📌

- **Passo 2: Anexar o Elemento ao DOM:** Depois de criar e customizar, precisamos dizer ao navegador onde colocá-lo.
- `elementoPai.appendChild(novoElemento)` : O método mais comum. Ele adiciona o `novoElemento` como o **último filho** do `elementoPai`.
- **O Processo:**
 1. Selecione o "pai" onde o novo elemento vai entrar (ex: uma `` ou `<div>`).
 2. Crie e configure o novo elemento "filho".
 3. Use `appendChild()` para fazer a conexão.

Slide 4: Removendo Elementos da Página 🗑️

- **Passo 3: Apagar o que não é mais necessário.** Existem duas formas principais.
- `elementoPai.removeChild(elementoFilho)` : A forma "clássica". Você precisa de uma referência tanto do pai quanto do filho que quer remover.
- `elementoARemover.remove()` : A forma **moderna e mais simples**. Você só precisa do elemento que quer apagar e chama o método `.remove()` diretamente nele. É a mais recomendada hoje em dia.

Prática

<> index.html X

JS script.js

style.css

<> index.html > html > body > ul#container

```
1  <!DOCTYPE html>
2  <html lang="pt-BR">
3  <head>
4      <meta charset="UTF-8">
5      <title>Aula 16: Criar e Remover Elementos</title>
6      <link rel="stylesheet" href="style.css">
7  </head>
8  <body>
9      <h1>Aula 16: Criar e Remover Elementos no DOM</h1>
10
11      <button id="btn-adicionar">Adicionar Novo Item</button>
12      <button id="btn-remover-ultimo">Remover Último Item</button>
13
14      <ul id="container">
15      </ul>
16
17      <script src="script.js"></script>
18  </body>
19 </html>
```


index.html

JS script.js

style.css



style.css > .item-criado

1

2 #container { border: 2px solid #ccc; padding: 15px; margin-top: 10px; }

3 .item-criado {

4 background-color: #e7f3fe;


5 border-left: 5px solid #2196F3;

6 padding: 10px;

7 margin-bottom: 5px;

8 list-style-type: none; /* Remove a bolinha da lista */

9 }

JS script.js >  btnRemoverUltimo.addEventListener('click') callback

```
1 // AULA 16: Criar e Remover Elementos - Demonstração dos Conceitos
2 console.log("--- JS: aula16-dom-criacao-remocao.js (Demonstração) ---");
3
4 // --- Selecionando os elementos de controle ---
5 const btnAdicionar = document.getElementById('btn-adicionar');
6 const btnRemoverUltimo = document.getElementById('btn-remover-ultimo');
7 const container = document.getElementById('container');
8
```

```
10 // --- EXEMPLO 1: Criando e Adicionando com createElement e appendChild ---
11 btnAdicionar.addEventListener('click', () => {
12     // 1. Criamos o novo elemento <li> na memória.
13     const novoItem = document.createElement('li');
14
15     // 2. Customizamos o elemento.
16     novoItem.textContent = `Novo Item`;
17     novoItem.classList.add('item-criado'); // Adicionamos a classe do CSS
18
19     // 3. Anexamos o elemento criado como último filho do container (<ul>).
20     container.appendChild(novoItem);
21 });
22
```

```
23
24 // --- EXEMPLO 2: Removendo com .remove() (Moderno) ---
25 btnRemoverUltimo.addEventListener('click', () => {
26     // Verificamos se o container tem algum filho para remover.
27     const ultimoItem = container.lastElementChild;
28     // Chamamos o método .remove() diretamente no elemento que queremos apagar.
29     // É mais simples e direto!
30     ultimoItem.remove();
31
32 });
```

Atividade

Crie uma lista de tarefas, onde posso adicionar tarefas e remover, com a opção de colocar tarefas em amarelo que são de alta prioridade.