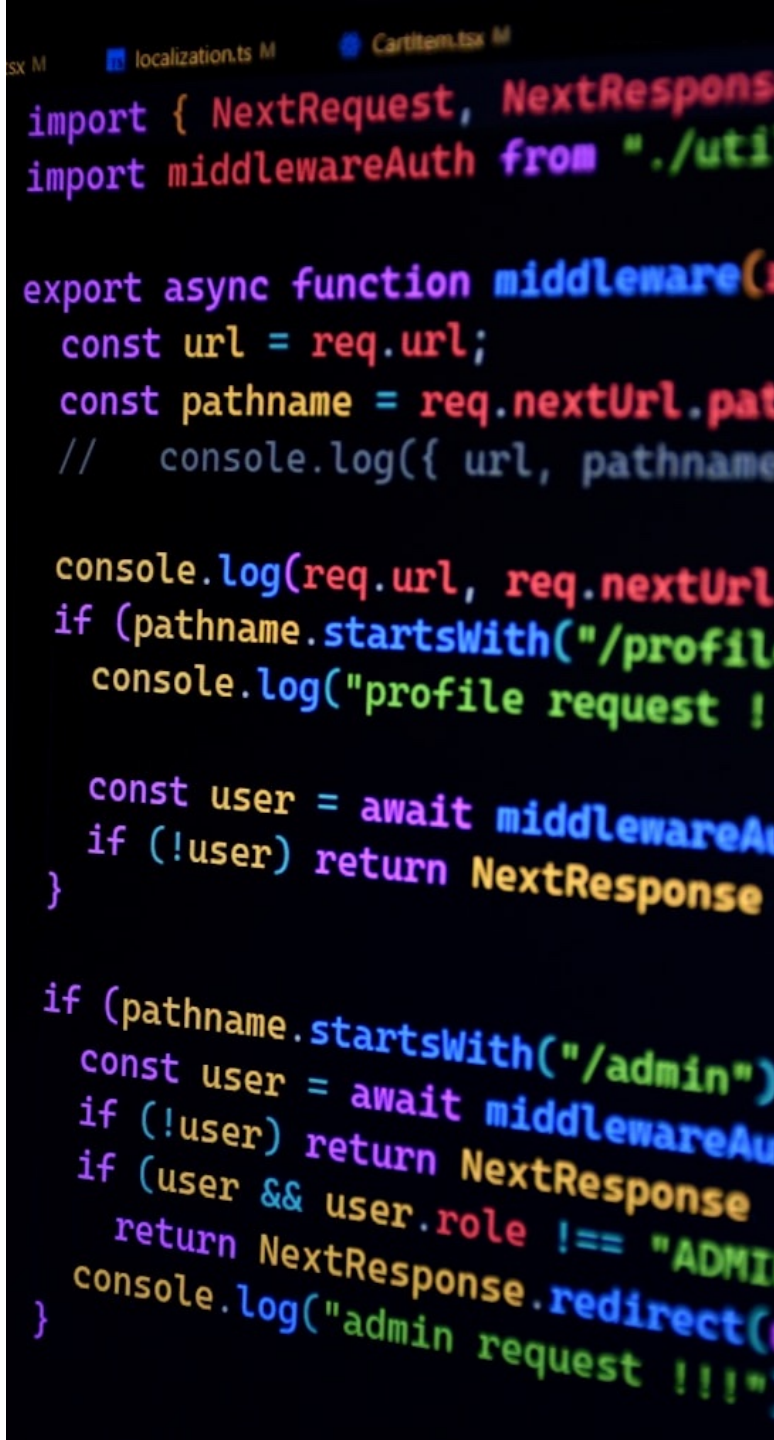
5954024 - Introdução ao Desenvolvimento Web

JavaScript - Teoria e Prática - Parte 2

Prof. Dr. Denis M. L. Martins
DCM | FFCLRP | USP

Objetivos de Aprendizagem

- Entender o que é o DOM
- Manipular elementos HTML com JavaScript
- Compreender o ciclo de vida de páginas web
- Capturar e tratar eventos
- Criar interações com o usuário



Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Document Object Model

- Document Object Model (DOM) é uma representação em árvore da estrutura HTML/CSS.
- **Nós (Nodes)**: Cada elemento, atributo ou texto é um nó na árvore hierárquica.
  - Cada tag HTML no documento corresponde a um há um objeto **Element** de JavaScript correspondente.
  - Assim também para um bloco de texto, que tem seu correspondende em um objeto **Text** correspondente.
- Representação estrutural do documento a partir de uma única API consistente.

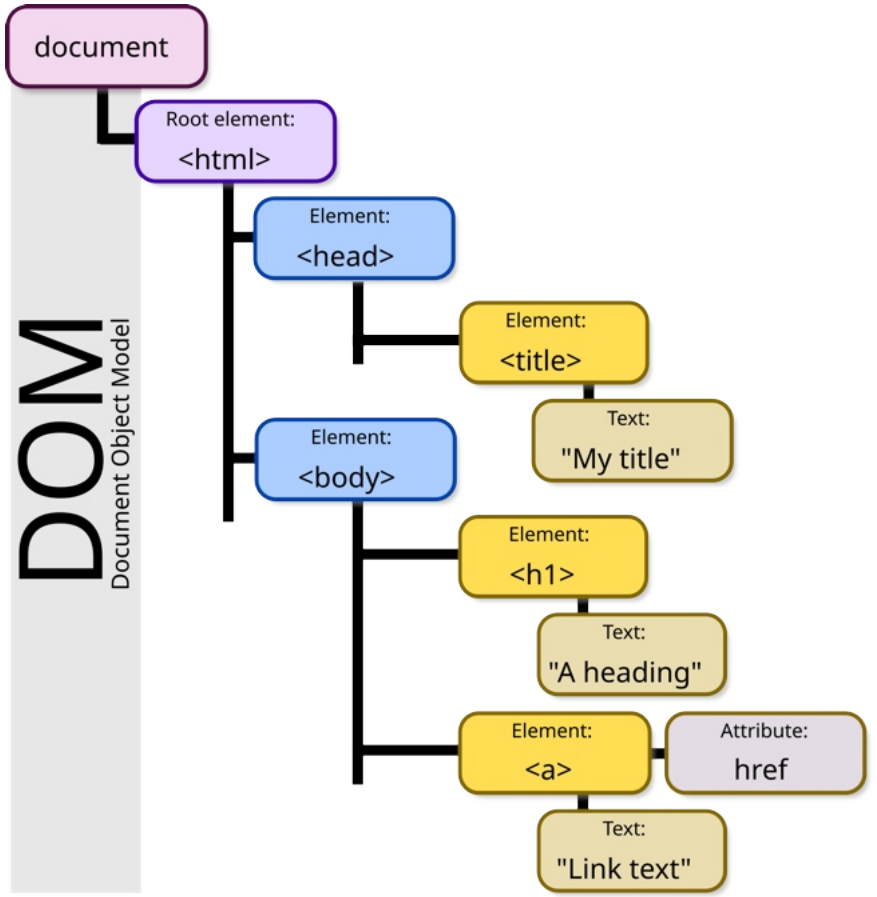


Fig. .1: Estrutura geral do DOM. Fonte: Wikipedia.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo: Estrutura Hierárquica do DOM

Crie um documento **.html** com o código abaixo.

```
<!DOCTYPE HTML>
<html>
<head>
    <title>Navegando Pelo DOM</title>
</head>
<body>
    <h1>Olá, Mundo!</h1>
    <p>Este exemplo é bem simples.</p>
</body>
</html>
```

Use o objeto **document** para navegar pelo DOM:

```
document.childNodes[1]
```

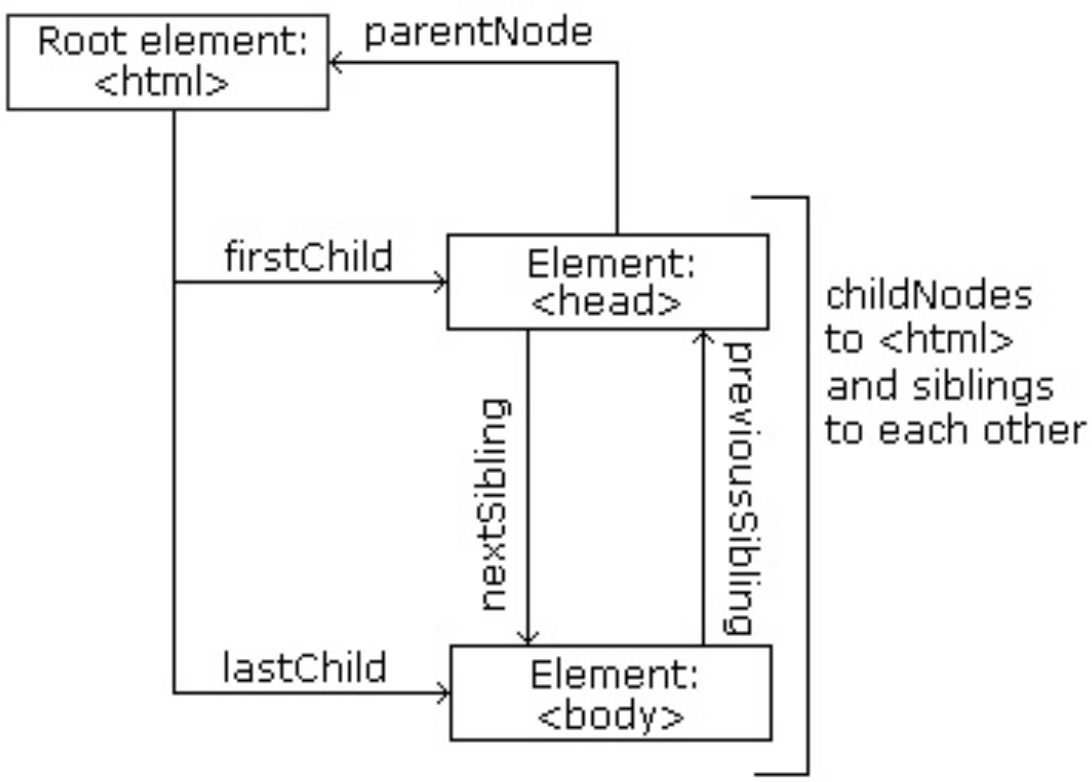


Fig. .1: Hierarquia do DOM. Fonte: W3Schools.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

API DOM: Métodos de Seleção de Elementos

No lado do cliente, JavaScript geralmente é utilizado para manipular o conteúdo de páginas Web. O acesso aos elementos do DOM é realizado via seletores:

1. **getElementById(id)**: Acesso direto por identificador único.
2. **querySelector(selector)**: Primeiro elemento encontrado pelo CSS Selector.
3. **querySelectorAll(selector)**: Retorna uma NodeList (coleção) de todos os elementos.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo

```
<!DOCTYPE HTML>
<html>
<head>
    <title>Navegando Pelo DOM</title>
</head>
<body>
    <div class="container">
    <h1 id="titulo-principal">Meu Título Principal</h1>
    <p class="destaque">Este é um parágrafo destacado
        que será manipulado pelo JavaScript.</p>
    <img id="foto-usuario" src="" alt="Foto do Usuário">
    <div class="botao-group">
        <button class="botao">Botão 1</button>
        <button class="botao">Botão 2</button>
        <button class="botao">Botão 3</button>
    </div>
</body>
</html>
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo (cont.)

```
// Exemplo: Seleção e Manipulação de Atributos
const elementById = document.getElementById('titulo-principal');
const elementByClass = document.querySelector('.destaque');
const allButtons = document.querySelectorAll('button');

// Manipulação de atributo 'src' em uma imagem
const imgElement = document.getElementById('foto-usuario');
if (imgElement) {
  imgElement.src = 'assets/nova-imagem.jpg';
}

// Manipulação de estilo inline via propriedade 'style'
elementByClass.style.color = '#2ecc71'; // Muda cor do texto para verde
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

querySelector

```
const p = document.querySelector("p");
```

Criando Elementos e Inserindo no DOM

```
const novoParagrafo = document.createElement("p");
novoParagrafo.innerText = "Novo parágrafo";
document.body.appendChild(novoParagrafo);
```

Removendo Elementos

```
novoParagrafo.remove();
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Eventos

Evento: ocorrência ação detectada pelo navegador (ex.: clique, teclado, carregamento).

Categoria	Exemplos	Uso típico
Mouse	`click`, `dblclick`, `mouseenter`, `mouseleave`, `contextmenu`	Interação com botões, menus
Keyboard	`keydown`, `keyup`, `keypress`	Formulários, jogos
Form	`submit`, `change`, `input`	Validação em tempo real
Window/ Document	`load`, `DOMContentLoaded`, `resize`, `scroll`	Inicialização e layout dinâmico
Custom	`myEvent` (via `new CustomEvent`)	Comunicação entre componentes

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo de Evento

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>Altera cor da tela</title>
  </head>
  <body>
       <button>Muda cor 1</button>
       <script>
       var btn = document.querySelector('button');
       function random(num) {
         return Math.floor(Math.random()*(num+1));
       }
       btn.onclick = function() {
         var rndCol = 'rgb(' + random(255) + ',' + random(255) + ',' + random(255) + ')';
         document.body.style.backgroundColor = rndCol;
       }
       </script>
  </body>
</html>
```

> Retirado do material do Prof. Evandro Ruiz

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Eventos (cont.)

Cada evento possui uma função associada *event handler* ou *event listener* que é executada cada vez que o evento for acionado.

```
<form id="login-form">
  <input type="text" name="user" placeholder="Usuário">
  <button type="submit">Entrar</button>
</form>
```

```
const form = document.getElementById('login-form');

form.addEventListener('submit', (e) => {
  e.preventDefault(); // impede o envio padrão
  const user = form.user.value;
  console.log(`Usuário: ${user}`);
});

document.querySelector('button').addEventListener('click', () => {
  alert('Botão clicado!');
});
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo de Clique do Mouse

```
<head>
  </head>
  <style>
   body { height: 200px; background: beige; }
   .dot { height: 8px; width: 8px; border-radius: 4px; background: blue; position: absolute;}
 </style>
 <body>
 <script>
    window.addEventListener("click", event => {
        let dot = document.createElement("div");
        dot.className = "dot";
        dot.style.left = (event.pageX - 4) + "px";
        dot.style.top = (event.pageY - 4) + "px";
        document.body.appendChild(dot);
    });
</script>
</body>
```

> Baseado no material do Prof. Evandro Ruiz

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo de Animação

```
<body style="background-color: black;">
  <img src="https://dcm.ffclrp.usp.br/images/logo-FFCLRP.png"
    width="200" style="position: relative;">
    <script>
        let ball = document.querySelector("img");
        let angle = Math.PI / 2;
        function animate(time, lastTime) {
        if (lastTime != null) {
            angle += (time - lastTime) * 0.001;
        }
        ball.style.top = (100 + Math.sin(angle) * 30) + "px";
        ball.style.left = (110 + Math.cos(angle) * 100) + "px";
        requestAnimationFrame(newTime => animate(newTime, time));
        }
    requestAnimationFrame(animate);
    </script>
</body>
```

> Baseado no material do Prof. Evandro Ruiz

Exercício: Contador Simples

Objetivo: Implementar um contador funcional usando apenas DOM e Eventos.

Requisitos Funcionais

1. **Seleção**: Identificar o elemento `<button>` e o elemento `<span>` de texto.
2. **Estado**: Manter uma variável `contador` no escopo global ou closure.
3. **Interação**: Adicionar listener de `click` ao botão.
4. **Feedback**: Atualizar o texto do `<span>` a cada clique.

> Como impedir que o contador seja incrementado se o usuário clicar muito rápido? Dica: debounce.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Ciclo de Vida da Página Web

Ordem de Execução e Carregamento:

- **Parsing**: O navegador lê o HTML linearmente.
- **DOM Ready**: A estrutura do documento está pronta para manipulação.
- **Load Complete**: Imagens, scripts externos e estilos são totalmente carregados.

Execução JavaScript:

- **Fase 1**: Carrega o documento e o código dos elementos <script> é executado.
  - Executados na ordem em que aparecem no documento
  - A ordem pode mudar pelo uso de atributos **async** e **defer** (aula futura).
- **Fase 2**: O nevegador chama funções de tratamento de evento e outras *callbacks* em resposta aos eventos que ocorrem de forma assíncrona.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Carregamento do DOM e Eventos

Problema comum: JS executa antes do DOM existir.

```
    // ERRO: Tenta acessar elemento que ainda não existe!
    document.querySelector('#meu-botao').addEventListener('click', function() {
    console.log('Botão clicado!');
    });
```

O script roda antes do parser HTML terminar de criar o elemento.

Solução:

```
document.addEventListener("DOMContentLoaded", function() {
    console.log("DOM carregado!");
});
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Ordem de Execução

DOMContentLoaded: dispara quando a árvore DOM está totalmente construída, antes que imagens e estilos sejam carregados.

```
document.addEventListener('DOMContentLoaded', () => {
  console.log('DOM pronto');
});
```

window.onload: dispara após todos os recursos (imagens, CSS, scripts) estarem prontos.

```
window.addEventListener('load', () => {
  console.log('Todos os recursos carregados');
});
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Conclusão

Conceitos-Chave:

- **DOM**: Estrutura em árvore acessível via seletores CSS.
- **Ciclo de Vida**: Prefira `DOMContentLoaded` para inicialização de scripts.
- **Eventos**: Utilize `addEventListener` para uma arquitetura limpa e testável.
- **Interação**: Manipulação de estado (variáveis) combinada com manipulação de atributos (`textContent`, `style`).

Próxima aula: JavaScript assíncrono.

Material adicional

- YouTube: CURSO JAVASCRIPT | Interagindo com a DOM + Manipulando Arrays e Objetos
- FLANAGAN, David. JavaScript: O Guia Definitivo. Disponível em: https://app.minhabiblioteca.com.br/reader/books/9788582607008/.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Lab Prático

Objetivo: criar uma aplicação web que permita ao usuário interagir com tarefas em tempo real, utilizando manipulação básica do DOM, tratamento de eventos e lógica JavaScript. Use o código abaixo como base.

Resultado esperado: página contendo um campo de texto, botão "Adicionar" e uma lista `<ul>` onde cada tarefa é representada por um `<li>` criado dinamicamente; o usuário pode inserir novas tarefas, ver a lista atualizada e interagir com os itens (marcar concluída).

```
<!DOCTYPE html>
<html>
<head>
    <title>Lista de Tarefas</title>
</head>
    <body>
    <h1>Minha Lista de Tarefas</h1>
    <input id="tarefaInput" type="text" placeholder="Digite uma tarefa">
    <button id="addBtn">Adicionar</button>
    <ul id="lista"></ul>
</body>
</html>
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Dúvidas e Discussão

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br