**CENTRO PAULA SOUZA**

**ETEC PROFESSORA MARIA CRISTINA MEDEIROS**

**Técnico em Informática para Internet M-TEC**

**Angelo Ivon Domingues Tenorio De Almeida**

**Fundamentos do JavaScript: lógica básica e manipulação do DOM**

**Ribeirão Pires**

**2025**

**SUMÁRIO**

[1. Breve histórico do JavaScript. 2](#_Toc199336474)

[2. Principais diferenças entre var, let e const. 3](#_Toc199336475)

[2.1. Qualquer coisa nova 3](#_Toc199336476)

[3. O que é o DOM (Document Object Model). 5](#_Toc199336477)

[4. Como funciona o getElementById. 7](#_Toc199336478)

[5. Explicação e exemplos de eventos básicos 10](#_Toc199336479)

[5.1. OnClick. 10](#_Toc199336480)

[5.2. OnBlur. 10](#_Toc199336481)

[5.3. OnKeyUp. 10](#_Toc199336482)

[6. Trecho de código com formulário simples com tratamento de evento. 12](#_Toc199336483)

# Breve histórico do JavaScript.

O JavaScript nasceu em 1995, criado por Brendan Eich em apenas 10 dias, enquanto trabalhava na Netscape. A ideia era simples: dar mais interatividade às páginas da web, que até então eram estáticas. Inicialmente chamado de Mocha, depois LiveScript, acabou recebendo o nome JavaScript por estratégia de marketing — embora não tenha quase nada a ver com Java.

Nos anos seguintes, a linguagem evoluiu aos poucos, ganhando um padrão oficial chamado ECMAScript. Por muito tempo, o JavaScript foi subestimado, usado apenas para pequenos efeitos visuais ou validação de formulários.

**Figura 1 - Aniversário de 30 anos em 2025 do JavaScript**



Fonte: DENO. A brief history of JavaScript. Deno Blog, 22 maio 2025. Disponível em: <https://deno.com/blog/history-of-javascript> . Acesso em: 28 maio 2025.

Tudo mudou em 2005, com a chegada do Ajax e de aplicativos como o Gmail, que mostraram ao mundo que páginas web podiam ser rápidas e dinâmicas. A partir daí, surgiram bibliotecas como jQuery, depois frameworks como Angular, React e Vue, que profissionalizaram o uso da linguagem.

Hoje, o JavaScript está em toda parte: nos navegadores, em servidores (com Node.js), em aplicativos móveis e até em dispositivos IoT. Ele se tornou uma das linguagens mais importantes e versáteis do mundo da programação.

# Principais diferenças entre var, let e const.

var: Variável com escopo de função ou global, permite redeclaração e reatribuição, sofre hoisting (a declaração é movida para o topo do escopo). Pode causar problemas por não respeitar escopo de bloco.

let: Variável com escopo de bloco, permite reatribuição mas não permite redeclaração no mesmo escopo. Também sofre hoisting, porém não pode ser usada antes da declaração (zona temporalmente morta).

const: Variável com escopo de bloco, não permite redeclaração nem reatribuição. Deve ser inicializada no momento da declaração. Para objetos e arrays, o conteúdo interno pode ser alterado, mas o identificador permanece constante.

# O que é o DOM (Document Object Model).

Durante o desenvolvimento web, um dos conceitos fundamentais para a criação de páginas dinâmicas e interativas é o DOM (Document Object Model). Esta pesquisa tem como objetivo compreender a estrutura, função e importância do DOM no contexto do uso do JavaScript como linguagem de manipulação de elementos da interface.

O DOM, sigla para Document Object Model, é uma interface de programação que os navegadores utilizam para representar documentos HTML e XML em forma de uma estrutura hierárquica de objetos, semelhante a uma árvore. Cada elemento, atributo ou conteúdo do HTML torna-se um nó dentro dessa árvore, que pode ser acessado, manipulado ou modificado por linguagens como o JavaScript.

O JavaScript é a principal linguagem utilizada para interagir com o DOM. Ele permite que os desenvolvedores acessem os elementos da página e realizem alterações em tempo real, com base nas ações do usuário, sem a necessidade de recarregar a página.

A manipulação do DOM é uma das bases da programação front-end moderna. Ela permite o desenvolvimento de interfaces dinâmicas e ricas, possibilitando a construção de aplicações web completas (como redes sociais, e-commerces e sistemas interativos) diretamente no navegador.

Além disso, frameworks modernos como React, Vue e Angular utilizam o DOM (ou versões otimizadas dele, como o DOM Virtual) para melhorar o desempenho e a organização do código.

O DOM é uma ferramenta essencial na construção de páginas web interativas, e seu uso por meio do JavaScript transformou a maneira como os sites funcionam. A partir do entendimento e domínio do DOM, desenvolvedores podem criar experiências mais ricas, responsivas e personalizadas para os usuários, tornando a web mais dinâmica e funcional.

# Como funciona o getElementById.

getElementById() é um método nativo do objeto document em JavaScript, responsável por retornar um único elemento HTML que possua um atributo id específico. O atributo id é único em um documento HTML, o que garante que o método sempre retorne no máximo um elemento correspondente. Caso o identificador não exista, o retorno será null.

Esse método permite que o JavaScript localize rapidamente elementos específicos dentro da árvore do DOM, facilitando a execução de tarefas como alterações de conteúdo, estilos e atributos, além da interação com eventos e ações do usuário.

A função getElementById() possui diversas aplicações práticas no desenvolvimento de páginas web. Abaixo, estão destacadas suas principais utilidades:

1. Acesso direto a elementos HTML

Permite ao desenvolvedor identificar e manipular rapidamente qualquer elemento da interface que possua um id, sem a necessidade de realizar buscas complexas.

2. Alteração de conteúdo

Com o uso de getElementById(), é possível modificar dinamicamente o texto exibido, substituindo informações de maneira imediata com base em ações do usuário ou condições do sistema.

3. Mudança de estilos visuais

Através do método, é possível aplicar novas regras de estilo a elementos específicos, como cores, tamanhos, fontes ou visibilidade, tornando a página visualmente responsiva e interativa.

4. Reação a eventos

Ao obter a referência de um elemento com getElementById(), é possível associar funções a eventos como cliques, digitação ou movimentação do mouse. Isso é fundamental para promover uma experiência de navegação dinâmica.

5. Leitura e manipulação de dados

A função também é amplamente utilizada para acessar campos de formulários, coletar valores inseridos pelos usuários e realizar validações antes do envio de informações.

6. Simplicidade e desempenho

Por ser um método direto e sem necessidade de seletores complexos, getElementById() é altamente eficiente e possui ótimo desempenho, sendo indicado especialmente para elementos únicos e bem identificados.

A função getElementById() representa uma das formas mais simples, porém poderosas, de interação entre o JavaScript e a estrutura HTML de uma página. Por meio dela, é possível acessar, modificar e controlar praticamente qualquer elemento da interface, o que a torna uma ferramenta indispensável no desenvolvimento web.

Mesmo com o surgimento de métodos mais modernos e versáteis, como querySelector, o getElementById() continua sendo amplamente utilizado por sua clareza, desempenho e compatibilidade universal entre os navegadores. Seu uso é recomendado sempre que se deseja manipular diretamente um elemento único de forma rápida e eficiente.

# Explicação e exemplos de eventos básicos

## OnClick.

O evento onclick é acionado sempre que o usuário clica com o botão esquerdo do mouse sobre um elemento HTML. É um dos eventos mais comuns e versáteis da linguagem JavaScript, pois permite a execução imediata de funções com base na interação direta do usuário com a interface.

Principais aplicações:

Execução de ações ao clicar em botões(ex: enviar formulário, abrir menu);

Interação com imagens, links ou outros elementos interativos;

Alteração de conteúdo ou estilos visuais após o clique.

## OnBlur.

O evento onblur é acionado quando um elemento perde o foco, ou seja, quando o usuário interage com um campo (como uma caixa de texto) e em seguida clica fora dele ou navega para outro campo da página. Trata-se de um evento comumente utilizado para validação de dados em formulários, visto que permite verificar o conteúdo inserido logo após o preenchimento, sem a necessidade de submeter o formulário.

Principais aplicações:

Validação de campos de formulário (ex: e-mail, telefone, CPF);

Exibição de mensagens de erro ou confirmação;

Aplicação de estilos visuais indicativos de erro ou sucesso.

## OnKeyUp.

O evento onkeyup ocorre quando o usuário solta uma tecla do teclado após pressioná-la. É amplamente utilizado para monitoramento de entrada de dados em tempo real, permitindo que o sistema reaja enquanto o usuário digita.

Principais aplicações:

Exibição de sugestões ou auto-completar durante a digitação;

Verificação da força de senhas em tempo real;

Filtragem de listas ou tabelas com base em termos digitados pelo usuário.

# Trecho de código com formulário simples com tratamento de evento.

<!DOCTYPE html>

<html>

<head>

<title>formulario </title>

</head>

<body>

<form id="Form">

<input type="text" id="nome" name="nome" placeholder="Nome" required>

<input type="email" id="email" name="email" placeholder="Email" required>

<button type="submit">Enviar</button>

</form>

<script>

const form = document.getElementById('Form');

form.addEventListener('submit', function(event) {

event.preventDefault();

alert('Formulario recebido!');

});

</script>

</body>

</html>