INSTITUTO FEDERAL
Espírito Santo
Campus de Alegre

Apostila — Material de apoio

Dando Vida à Página: Introdução ao JavaScript

Da Teoria Fundamental à Prática de Mercado

Desenvolvimento Front-End I

Período: TADS EaD - Semana 9

Autor: Prof. Cleziel Franzoni da Costa

Instituto Federal do Espírito Santo — Campus de Alegre

2025

Sumário

1 Apresentação da Semana: O Despertar da Interatividade

Bem-vindo à semana que transformará suas páginas estáticas em experiências dinâmicas. Até agora, você se tornou um arquiteto de estruturas com **HTML** e um designer de interfaces com **CSS**. Na semana passada, construímos formulários, que são as portas de entrada de dados de qualquer aplicação web. Contudo, eles ainda são como prédios sem eletricidade: funcionais, mas inertes.

Nesta semana, vamos instalar o "sistema nervoso"de nossas páginas com o **JavaScript (JS)**. Ele é o terceiro e mais dinâmico pilar do desenvolvimento Front-End, responsável por todo o **comportamento, lógica e interatividade**.

Nosso foco será dar vida aos formulários, implementando **validações em tempo real**, exibindo mensagens de feedback inteligentes e controlando o fluxo de dados antes mesmo que eles sejam enviados a um servidor. Ao final desta jornada, você não apenas entenderá a sintaxe do JavaScript, mas também a sua filosofia: ouvir o usuário e responder a ele de forma elegante e eficiente.

2 Objetivos de Aprendizagem

Ao concluir esta semana de estudos, você será capaz de:

- **Dominar** o papel do JavaScript na tríade do Front-End (HTML, CSS, JS) e sua execução no navegador.
- **Estruturar** o código JavaScript de forma limpa e organizada, utilizando arquivos externos.
- **Manipular** dados de forma eficaz utilizando variáveis, constantes, tipos de dados e operadores.
- **Selecionar e modificar** qualquer elemento da página com precisão, através da manipulação do DOM.
- **Capturar e reagir** a interações do usuário, como cliques e submissão de formulários, utilizando eventos.
- **Construir validações** de formulários robustas, aprimorando a experiência do usuário (UX).

3 Introdução ao JavaScript: O Cérebro da Operação

3.1 O que é o JavaScript?

O **JavaScript (JS)** é uma linguagem de programação de alto nível, interpretada, que roda no lado do cliente (*client-side*). Em termos simples, é o código que permite que suas páginas web "pensem"e "ajam".

- **HTML** é o esqueleto: a estrutura semântica.
- **CSS** é a pele e as roupas: a camada de apresentação visual.
- **JavaScript** é o cérebro e os músculos: a camada de comportamento e interatividade.

3.2 Onde o JavaScript é executado?

O JS é executado pelo **motor JavaScript** (*JavaScript Engine*) embutido em todos os navegadores modernos (como o V8 no Google Chrome e Edge, ou o SpiderMonkey no Firefox). Isso significa que o código é processado diretamente na máquina do usuário, permitindo respostas instantâneas na interface.

3.3 Como inserir JavaScript em uma página

3.3.1 Script Interno

O código é inserido diretamente no arquivo HTML, dentro da tag `<script>`. Útil para testes rápidos.

```
<script>
  console.log("Este código roda de dentro do HTML.");
</script>
```

3.3.2 Script Externo

Esta é a abordagem profissional. O código é escrito em um arquivo separado com a extensão `.js` e vinculado ao HTML.

```
<script src="js/script.js"></script>
```

Boa prática: Separação de Responsabilidades: Sempre coloque seu JS em arquivos externos. Isso torna o código mais legível, reutilizável e fácil de manter.

Posicionamento da Tag `<script>`: Coloque-a imediatamente antes do fechamento da tag `</body>`. O navegador lê o HTML de cima para baixo. Ao fazer isso, você garante que toda a estrutura (DOM) esteja montada e pronta para ser manipulada.

4 Fundamentos da Linguagem: A Caixa de Ferramentas

4.1 Variáveis e Constantes

- `let`: Declara uma variável cujo valor pode ser **reatribuído**.
- `const`: Declara uma constante, cujo valor é **fixo** e não pode ser alterado.

```
let contadorCliques = 0;
contadorCliques = 1; // Valido

const URL_API = "https://api.meusite.com";
// URL_API = "outro-valor"; // Geraria um erro!
```

Atenção: Evite usar `var`. Trata-se de uma sintaxe legada do JavaScript que possui um comportamento de "escopo de função"que pode levar a bugs difíceis de rastrear. `let` e `const` possuem "escopo de bloco", que é mais previsível e seguro.

4.2 Tipos de Dados Primitivos e Operadores

Tipo	Descrição	Exemplo
`String`	Sequência de caracteres (texto)	`"Olá"`, `'Front-End'`
`Number`	Valores numéricos, inteiros ou decimais	`42`, `9.5`, `-100`
`Boolean`	Valor lógico, `true` ou `false`	`true`

4.3 Concatenação e Template Strings

Para construir strings complexas, as **Template Strings** (usando crases ` `) são superiores.

```
let nomeUsuario = "Maria";

// Concatenação (legado, menos legível)
console.log("Bem-vinda, " + nomeUsuario + "!");

// Template String (moderno e limpo)
console.log(`Bem-vinda, ${nomeUsuario}!`);
```

5 O DOM (Document Object Model): O Mapa da Página

O **DOM** é a representação em memória do seu arquivo HTML. O navegador transforma a estrutura de tags em uma **árvore de objetos**, onde cada elemento, atributo e texto é um "nó".

5.1 Principais Métodos de Seleção

- `document.getElementById("id")`: O método mais rápido. Seleciona o **único** elemento com o `id` especificado.
- `document.querySelector("seletor-css")`: Versátil. Retorna o **primeiro** elemento que corresponde a um seletor CSS válido.

5.2 Alterando Conteúdo e Estilo

```
// HTML: <h1 id="titulo-principal">Título Original</h1>

// 1. Seleciona o elemento
const titulo = document.getElementById("titulo-principal");

// 2. Altera o conteúdo de texto
titulo.innerText = "Bem-vindo ao Mundo do JavaScript!";

// 3. Altera o estilo CSS
titulo.style.color = "darkblue";
titulo.style.backgroundColor = "#f0f8ff";
```

6 Eventos: A Reação às Ações do Usuário

Eventos são sinais que a página emite em resposta a uma interação do usuário. O JavaScript pode "ouvir"(*listen*) esses eventos e executar uma função.

6.1 Exemplo com `onclick`

O atributo `onclick` no HTML é uma forma direta de associar um clique a uma função JavaScript.

```
<h2 id="titulo">Formulário simples</h2>
<input type="text" id="nome">
```

```
<button onclick="mostrarMensagem()">Enviar</button>
<p id="mensagem"></p>

<script>
  function mostrarMensagem() {
    const campoNome = document.getElementById("nome");
    const pMensagem = document.getElementById("mensagem");
    const nomeDigitado = campoNome.value;

    if (nomeDigitado === "") {
      pMensagem.innerText = "Por favor, preencha seu nome.";
      pMensagem.style.color = "red";
    } else {
      pMensagem.innerText = `Olá, ${nomeDigitado}!`;
      pMensagem.style.color = "green";
    }
  }
</script>
```

7 Evoluindo: Manipulação Profissional de Formulários

A prática de mercado moderna prioriza a separação total entre HTML e JS. Para isso, usamos o método `addEventListener`.

7.1 O Evento `submit` e o `addEventListener()`

O evento `submit` é disparado no elemento `<form>` quando o usuário tenta enviá-lo. O `addEventListener` "escuta"por um evento e executa uma função (*callback*) quando ele ocorre.

```
<form id="form-contato">
  <label for="nome">Nome:</label>
  <input type="text" id="nome">
  <label for="email">E-mail:</label>
  <input type="email" id="email">
  <button type="submit">Enviar</button>
</form>
<p id="mensagem-feedback"></p>
```

index.html

```
const form = document.getElementById("form-contato");
```

```
const pMensagem = document.getElementById("mensagem-feedback");

form.addEventListener("submit", function(event) {
  // Impede o comportamento padrão do formulário
  event.preventDefault();

  const campoNome = document.getElementById("nome");
  const nome = campoNome.value.trim();
  // ... (restante da lógica de validação)

  if (nome === "") {
    pMensagem.innerText = "Erro: campo nome éobrigatório.";
    pMensagem.style.color = "crimson";
  } else {
    pMensagem.innerText = `Obrigado, ${nome}!`;
    pMensagem.style.color = "darkgreen";
    form.reset(); // Limpa o formulário
  }
});
```

js/script.js

Atenção: A validação no front-end é focada na **experiência do usuário (UX)**. Ela **nunca** deve ser a única camada de segurança. A validação de segurança **obrigatória** deve sempre ser feita também no **back-end**.

8 Boas Práticas e Padrões de Mercado

8.1 Organização de Arquivos

Uma estrutura de projeto limpa é o primeiro sinal de profissionalismo.

```
/meu-projeto
  index.html
  /css
      style.css
  /js
       script.js
```

8.2 Código Limpo (*Clean Code*)

- **Indentação Consistente**: Use 2 ou 4 espaços para indentar blocos de código.
- **Nomenclatura (`camelCase`)**: Use nomes descritivos para variáveis e funções.

- **Comentários**: Comente o **"porquê"**, não o "o quê".

9 Exercícios de Fixação

1. **Botão Mágico**: Crie uma página com um `<h1>` e um botão. Ao clicar no botão, o texto do `<h1>` deve ser alterado para "A Mágica Aconteceu!"e sua cor de fundo deve mudar.

2. **Formulário Personalizado**: Monte um formulário com campos "Nome"e "E-mail". Use `addEventListener` para capturar o evento de `submit`. Impede o recarregamento da página e exiba uma mensagem de boas-vindas.

3. **Validação Avançada**: Melhore o exercício anterior. Adicione uma validação que verifica se o campo de e-mail contém o caractere `@`. Se não contiver, exiba uma mensagem de erro específica.

4. **Desafio (Opcional)**: Crie um botão "Mostrar Hora Atual". Ao ser clicado, ele deve exibir a data e hora formatadas para o padrão brasileiro. Dica: `new Date().toLocaleString('pt-BR');`.

10 Recursos Complementares e Próximos Passos

- **MDN Web Docs – Introdução ao JavaScript**: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript
- **W3Schools – JavaScript DOM**: https://www.w3schools.com/js/js_htmldom.asp
- **Curso em Vídeo – JavaScript para Iniciantes**: https://www.youtube.com/playlist?list=PLHz_AreHm4dlsK3Nr9GVvXCbpQyHQl1o1