Professor Fausto Cintra

Apostila de JavaScript (ES6+)

Nível básico

Prof. Fausto G. Cintra
(professor@faustocintra.com.br)

02/07/2021

Sumário

Capítulo 1

COMEÇANDO COM JAVASCRIPT

1.1 O JavaScript e as páginas Web

As páginas Web são formadas pela combinação de três tecnologias:

- **HTML** (*Hypertext Markup Language*, isto é, linguagem de marcação de hipertexto): é responsável por estruturar o conteúdo da página, ou seja, **o que** você vê nela.
- **CSS** (*Cascading Style Sheets*, folhas de estilo em cascata): cuida da aparência da página, como cores, fontes e alinhamento dos elementos. Controla **como** o conteúdo é exibido na página.
- **JavaScript**: é uma linguagem de **programação** que pode ser adicionada às páginas Web, conferindo-lhes interatividade para com o usuário. O JavaScript é capaz, por exemplo, de verificar em um formulário se o usuário digitou algo diferente do esperado ou de fazer coisas se movimentarem pela página.

A linguagem JavaScript será o objeto do nosso estudo. No entanto, como iremos utilizá-la **dentro** de uma página Web, é necessário aprender o básico de HTML.

1.2 A estrutura de uma página HTML

Uma página Web (também chamada de página HTML) é um arquivo de texto simples com extensão `.html` ou `.htm`. A Listagem 1.1 seguir apresenta a estrutura básica de um arquivo HTML.

IMPORTANTE

Os números que aparecem à esquerda do código **não** fazem parte dele. Servem apenas para que possamos nos referir a diferentes partes do código usando o número da linha.

Listagem 1.1 Estrutura básica de um arquivo HTML

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Minha primeira página HTML</title>
6  </head>
7  <body>
8      <p>Olá, mundo!</p>
9  </body>
10 </html>
```

Como podemos ver, o HTML é formado por elementos delimitados pelos caracteres < e >, os quais são chamados **tags**.

Muitas *tags* vêm em pares, formando **seções**. Podemos observar, por exemplo, que a tag `<body>`, chamada *tag* de abertura, tem a correspondente *tag* de fechamento `</body>` (note a presença da / antes do nome da *tag*).

Agora, vamos analisar cada uma das partes desse código.

- `<!DOCTYPE html>` (linha 1): essa *tag* serve para indicar ao navegador Web que irá exibir a página qual a versão da linguagem HTML está sendo usada. No caso, esse *doctype* indica que se trata da versão 5 do HTML, a mais recente.
- Seção **html** (linhas 2 a 10): a maior parte do código da página fica nessa grande seção. Dentro dessa grande seção, temos as seções **head** e **body**.
- Seção **head** (linhas 3 a 6): aqui colocadas *tags* de configuração da página, como a `<meta charset="UTF-8">` (linha 4), para garantir que os caracteres acentuados sejam exibidos corretamente. Os elementos dessa seção, normalmente, não têm um efeito visível para o usuário. Uma exceção é a *tag* `<title>` (linha 5), cujo conteúdo aparece na aba do navegador onde a página estiver sendo exibida.
- Seção **body** (linhas 7 a 9): todo o conteúdo da página que será visível para o usuário é colocado nessa seção. No código de exemplo, temos um parágrafo (`<p>`, linha 8) contendo um texto a ser exibido.

Quando o arquivo HTML contendo este código for exibido em um navegador Web, veremos um resultado semelhante ao da Figura 1.1:

1.3 Adicionando JavaScript a uma página HTML

Para utilizar JavaScript em uma página HTML, precisamos criar uma seção `<script></script>`, normalmente dentro da seção **head**, e adicionar o código JavaScript dentro dela.

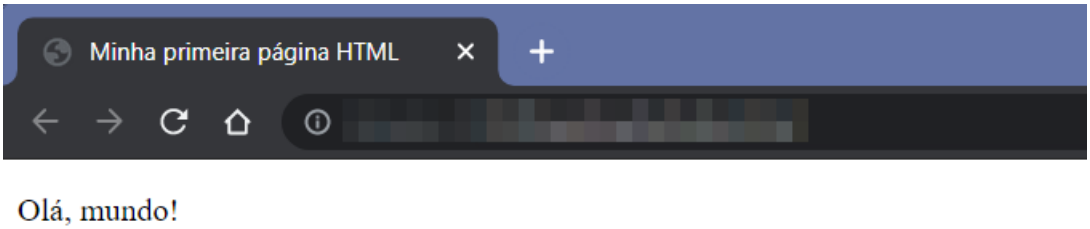

Figura 1.1: Exibição de uma página HTML

A Listagem 1.2 apresenta uma nova versão da mensagem "Olá, mundo!" mas, desta vez, quem dará o recado é a linguagem JavaScript.

Listagem 1.2 Código JavaScript em uma página HTML

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Minha primeira página HTML</title>
6      <script>
7          alert('Olá, mundo!')
8      </script>
9  </head>
10 <body>
11
12 </body>
13 </html>
```

Observe com atenção as linhas 6 a 8. Aberto no navegador, um arquivo HTML com este código produz o seguinte resultado (Figura 1.2):

Portanto, agora você já sabe. Toda vez que formos usar JavaScript em uma página Web, devemos:

1. Criar um arquivo com extensão `.html` ou `.htm`.
2. Colocar dentro desse arquivo a estrutura básica de um arquivo HTML. Editores de código, como o Visual Studio Code ou o Gitpod possuem recursos que geram

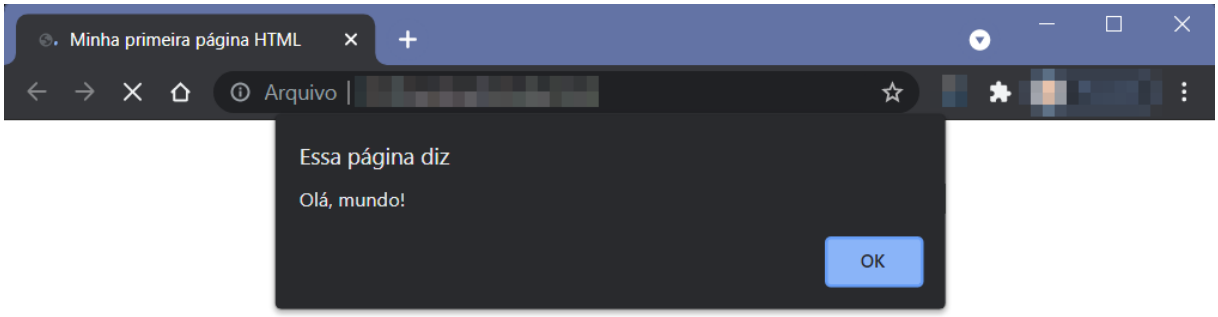

Figura 1.2: Mensagem exibida usando JavaScript

automaticamente este código.

3. Adicionar uma seção `<script></script>` na seção **head** e colocar as instruções JavaScript dentro dela.

1.4 Usando o console JavaScript

Todos os navegadores mais utilizados atualmente tem uma parte "secreta", desconhecida da maioria dos usuários. Essa parte é chamada de Ferramentas de Desevolvedor e pode ser acessada ao pressionar a tecla `F12`. Será aberto um painel, no lado direito ou inferior da tela, conforme mostrado na Figura 1.3.

> **DICA**: usando o menu de opções (2), é possível mudar o posicionamento das Ferramentas de Desenvolvedor na tela.

Na aba Console, é possível digitar instruções JavaScript, incluindo operações aritméticas, e ver imediatamente seu resultado. Veja alguns exemplos na Figura 1.4.

No próximo capítulo, vamos aprender sobre variáveis e usaremos o console para fazer alguns testes.

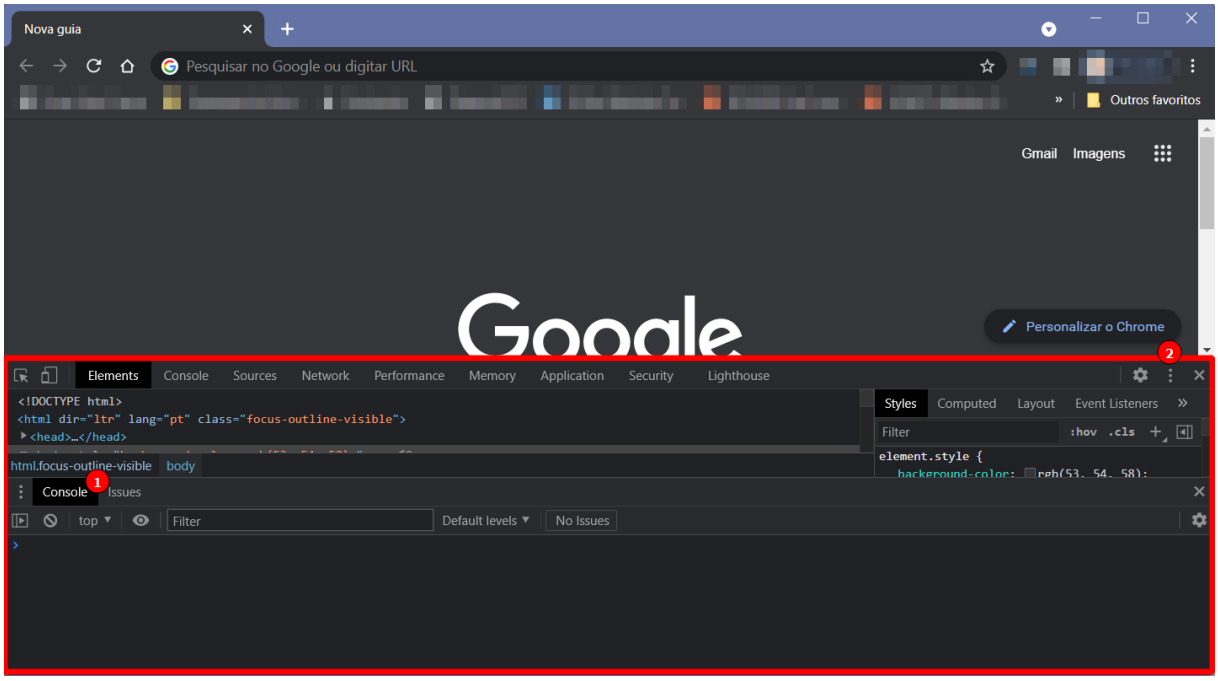

Figura 1.3: Navegador Web exibindo as Ferramentas de Desenvolvedor (no destaque). Note (1) a aba Console e (2) o menu de opções

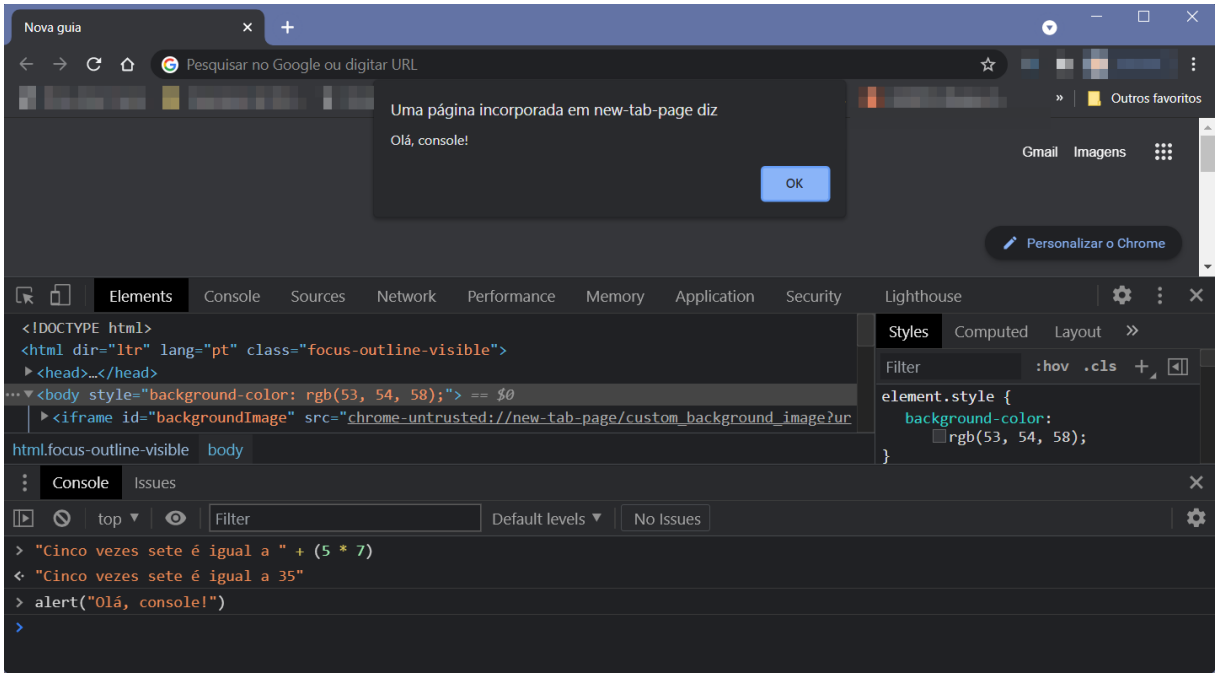

Figura 1.4: Resultado da execução de alguns comandos no console JavaScript do navegador Web

Capítulo 2

VARIÁVEIS E TIPOS DE DADOS

2.1 Declaração de variáveis

O Javascript é uma linguagem de tipagem dinâmica. Isso significa que os tipos de dados de suas variáveis não são determinados no momento em que são declaradas. Em vez disso, eles são deduzidos a partir dos **valores** atribuídos a elas.

Veja alguns exemplos na Listagem 2.1:

Listagem 2.1 Declarações de variáveis em JavaScript

```
1 let x
2 var preco
3 const meuNome = 'Fausto'
```

> **NOTE BEM**: para declarar uma variável em JavaScript, basta uma das palavras-chave reservadas (**`let`** ou **`var`**) seguida do nome da variável, nada mais. **`const`** exige também que um valor seja atribuído à variável no momento da declaração.

Há três palavras-chave utilizadas para declarar variáveis em JavaScript:

- **`let`**: atualmente, é o método recomendado de criação de variáveis. Uma das vantagens de sua utilização é a impossibilidade de se declarar mais de uma variável com o mesmo nome, o que ajuda a evitar erros de lógica no código. Possui também outros benefícios e características que serão explicadas ao longo desta apostila. É uma adição relativamente recente à linguagem (foi introduzida na versão ES6, de 2015).
- **`var`**: é a palavra-chave originalmente disponível para a declaração de variáveis, desde a primeira versão do JavaScript. Seu uso apresenta alguns problemas, como a possibilidade de redeclarar uma variável já existente, o que pode induzir a erros

de lógica. Evite utilizá-la, até compreender completamente as consequências de seu emprego.

- **`const`**: em algumas situações, é necessário representar valores que não devem mais ser alterados posteriormente. São as chamadas **constantes**. Variáveis declaradas com **`const`** devem receber um valor quando declaradas e não aceitam que este valor seja modificado depois.

2.1.1 Nomeando variáveis

Em JavaScript, nomes de variáveis devem começar com uma letra; os caracteres `$` e `_` também são aceitos primeira posição. Dígitos (`0` a `9`) podem ser utilizados a partir da segunda posição.

Você também não pode usar para nomear variáveis nenhuma das palavras reservadas pelo JavaScript para o seu próprio uso.

É importante frisar que JavaScript é uma **linguagem sensível à diferença entre letras maiúsculas e minúsculas** (*case sensitive*, em inglês), isto é, ela trata letras maiúsculas e minúsculas como coisas diferentes. Assim, uma variável de nome `num` é diferente de outra variável de nome `NUM`, e ambas são diferentes de uma terceira variável nomeada como `Num`.

Apesar de as especificações da linguagem JavaScript assim permitirem, não é recomendável declarar variáveis que contenham caracteres acentuados.

IMPORTANTE

Os exemplos da Listagem 2.2 contêm comentários iniciados com `//`, chamados **comentários de linha**. Esta é uma das formas de se fazer comentários em JavaScript.

Listagem 2.2 Exemplos de nomeação de variáveis

```
let x       // OK!
let primeiroNome    // OK!
let 1nome   // INVÁLIDO: começa com um dígito
let $valor  // OK, mas pouco usual
let _num    // OK, mas pouco usual
let %resultado  // Caractere inicial INVÁLIDO
let área    // OK, mas acentos não são recomendados
```

2.1.2 Declarações múltiplas

Várias variáveis podem ser declaradas simultaneamente (Listagem 2.3):

Listagem 2.3 Declarações múltiplas de variáveis

```
let quantidade, precoUnitario, precoTotal
let nome, email, telefone, celular
```

2.1.3 Convenções de nomeação de variáveis

Quando muitos desenvolvedores trabalham num mesmo projeto, é comum que surja, mais cedo ou mais tarde, alguma discórdia sobre a forma de nomear as variáveis.

Por isso, as comunidades de cada linguagem acabam adotando convenções, que não são regras definidas na própria linguagem, mas sim uma espécie de "combinado" entre seus membros.

A convenção mais comum entre os desenvolvedores JavaScript é a seguinte:

1. **Sempre** iniciar o nome das variáveis com uma letra **minúscula**; e
2. Se o nome da variável for composto por mais de uma palavra, é utilizada **inicial maiúscula a partir da segunda** palavra.

Observe os exemplos da Listagem 2.4:

Listagem 2.4 Uso da convenção 'camel case' na nomeação de variáveis

```
    // Uma palavra, inicial minúscula
    let area
    // Duas palavras, a segunda com inicial maiúscula
    let areaTerreno
    // Três palavras, iniciais maúsculas a partir da segunda
    let areaTerrenoPadrao
```

> **CURIOSIDADE**: esse tipo de convenção é chamado, em inglês, de *camel case* (*camel* significa "camelo"). O motivo é que as letras maúsculas no meio do nome das variáveis acabam se parecendo com as corcovas do animal.

2.2 Atribuindo valores a variáveis

Para atribuir valor a uma variável, é usado o operador = em JavaScript. A Listagem 2.5 exemplifica os diferentes tipos de valores que podem ser atribuídos a variáveis na linguagem.

Agora, vamos ver em detalhes os **tipos de dados** disponíveis na linguagem JavaScript.

2.3 Tipos de dados

Usaremos, como referência e exemplo, a Listagem 2.5.

Listagem 2.5 Exemplos de atribuição de valores a variáveis

```
1   let nome, sobrenome, naturalidade, idade, altura, peso, casado
2   let conjuge, ocupacao, filhos, nomeCompleto
3   nome = "Afrânio"                                    // string
4   sobrenome = 'Azeredo'                               // string
5   naturalidade = `Morro Alto de Cima (MG)`            // string
6   idade = 44                                          // number
7   altura = 1.77                                       // number
8   peso = undefined                                    // undefined
9   casado = true                                       // boolean
10  conjuge = { nome: 'Jeruza', sobrenome: 'Jordão' }   // object
11  ocupacao = null                                     // object
12  filhos = ['Zózimo', 'Zuleica']                      // object
13  nomeCompleto = function(nome, sobrenome) {          // function
14      return nome + " " + sobrenome
15  }
```

- **string** (linhas 3 a 5): representa uma sequência de caracteres, ou seja, um texto. Em JavaScript, *strings* podem ser delimitadas com aspas duplas (`"`), aspas simples (`'`) ou acentos graves (‘), muitas vezes chamados também de crases. Aspas simples e aspas duplas são totalmente equivalentes entre si, e a escolha por uma ou por outra acaba sendo decisão do programador. Já as *strings* delimitadas por acentos graves têm significado e funções especiais, que serão explicadas mais à frente.
- **number** (linhas 6 e 7): ao contrário de outras linguagens de programação, o JavaScript não faz distinção entre números inteiros e números com parte fracionária (também chamados de números de ponto flutuante), colocando-os todos sob um mesmo tipo. Para separar a parte inteira da parte fracionária, quando esta exista, é sempre usado o ponto (`.`), embora, na língua portuguesa, usemos a vírgula para esse fim. Existem várias outras formas de representar valores numéricos:
  - valores hexadecimais (base 16) podem ser representados usando-se o prefixo `0x`. Por exemplo `0x1A` representa o valor hexadecimal `1A` (equivalente a 26 em decimal).
  - valores octais (base 8) são representados usando-se um `0` no início. **CUIDADO!** Para o JavaScript, `045` não é um 45 decimal com um inútil zero à esquerda, e sim o valor octal `45` (que, convertido em decimais, equivale a 37).
  - Números muito grandes ou muito pequenos podem usar a chamada notação científica. Assim, `4e12` é o mesmo que `4` vezes `1` seguido de doze zeros, ou seja `4000000000000`.
  - Para números *realmente* grandes, foi introduzida na versão 2020 do EcmaScript mais um tipo de dados, o **bigint**. Você pode saber detalhes dessa novidade consultado a documentação da Mozilla Developer Network.

- **undefined** (linha 8): é um tipo especial, usado para indicar que uma variável não tem qualquer valor atribuído a ela. A propósito, toda variável declarada e que ainda não recebeu um valor é considerada **`undefined`**.
- **boolean** (linha 9): como indicado pelo próprio nome, representa valores booleanos. Os únicos valores possíveis são **`true`** (verdadeiro) e **`false`** (falso), sempre escritos totalmente em minúsculas.
- **object** (linhas 10 a 12): é o tipo de dados mais versátil da linguagem, usado, normalmente, para armazenar vários valores em uma única variável. Os vetores (linha 12) e objetos em sentido estrito (linha 10) fazem parte desta categoria. **`null`** é um valor especial utilizado para representar um objeto inexistente.
- **function** (linhas 13 a 15): em JavaScript, funções podem ser atribuídas a variáveis. Essa característica é importante para usos de nível intermediário e avançado da linguagem.

Licença

Créditos

Imagem da capa: Technology vector created by vectorjuice - www.freepik.com