



# **Desenvolvimento Web Completo 2022**


Por Jorge Sant Anna, Programador  
Jamilton Damasceno, Analista de Sistema e Professor




# Seção 4: JavaScript




# Aula 2: Incluindo JavaScript em Páginas HTML5


- 
- Existem 2 (duas) formas de fazer inclusões de códigos JavaScript dentro de arquivos HTML.
  - A primeira forma é incluindo a codificação JavaScript diretamente no arquivo HTML5.
  - Para fazer isso, basta encapsular toda a codificação JavaScript dentro da tag “script” de forma análoga as tags HTML.

- 
- A diferença é que o browser, no momento da interpretação, quando identificará o script, ele vai compreender que aquele conteúdo dentro da tag “script” não se trata de um elemento HTML e, sim, de uma codificação JavaScript a ser interpretada.

# Exemplo

```
<script>  
    //Aqui fica a codificação JavaScript  
</script>
```

- 
- A segunda forma que temos para fazer a inclusão de JavaScript em nossos arquivos HTML5, é a partir de um arquivo externo.
  - Criamos um arquivo, esse arquivo precisa ter a extensão “.js”. O nome é, geralmente, definido com base no que o script faz. Então, o nome do script costuma sugerir para que ele serve.
  - Claro que é interessante que a escrita desse nome respeitem algumas regras, como, por exemplo, não conter espaços ou caracteres especiais para evitar qualquer tipo de erro.


- 
- Para fazer a inclusão de um arquivo externo em JavaScript, basta utilizar a mesma tag “scripts”, porém, definindo o atributo “src” de source, indicando, portanto, qual que é a fonte daquela tag script.





# Exemplo


```
<!-- Segunda forma -->
```

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

- 
- Então, das duas formas podemos incluir scripts dentro do nosso arquivo HTML5.
  - E qual é a melhor forma, então, de fazer inclusões, já que existem duas possibilidades?

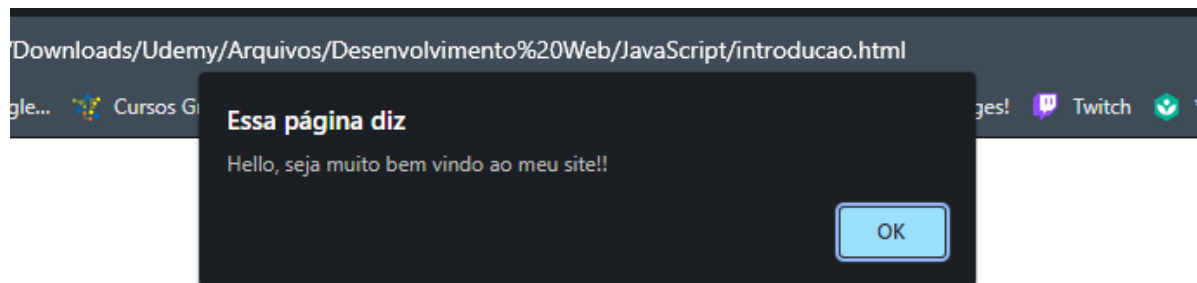
- 
- O resultado dos dois métodos de inclusão de conteúdo JavaScript dentro das nossas páginas é o mesmo. A única dica é o seguinte: caso a sua codificação JavaScript seja muito extensa, vale a pena deslocar o conteúdo para um arquivo à parte. Dessa forma, você consegue compartimentar melhor o seu código, fica mais fácil de ler.
  - Caso você tenha pequenos blocos de JavaScript dentro da sua aplicação, não tem problema fazer a inclusão direta dentro do próprio código HTML.


- 
- Um outro detalhe interessante é que podemos criar blocos de códigos JavaScript tantas vezes quanto forem necessárias. Podemos, inclusive, também, fazer várias inclusões de arquivos externos de JavaScript.
  - Então, precisamos incluir um bloco de código basta utilizar tag “script” que precisam importar, basta definir o atributo “src” e quantas vezes forem necessárias dentro do “head” ou do “body” na nossa página HTML.


- 
- A diferença entre criar a tag “script” dentro do “head” e dentro do “body”, é que as instruções da tag “head” são processadas antes do carregamento do “body”. Isso pode causar um erro de precedência de interpretação.

# Exemplo


```
<script>  
    alert('Hello, seja muito bem vindo ao meu site!!');  
</script>
```





- 
- Essa é a dinâmica de uma linguagem interpretada. Podemos fazer modificações sem passar por um processo de construção de um executável, porque o navegador já incorpora um interpretador do JavaScript e faz toda a tratativa para nós.
  - Quando estamos trabalhando com linguagens de programação interpretadas, ou seja, aquelas linguagens que são processadas em tempo de execução. Nesse caso, JavaScript será interpretado pelo browser, é muito importante ter em mente essa ideia de precedência de execução do código.

- 
- Isso porque, durante a interpretação do código, principalmente quando estamos aprendendo a lidar com o JavaScript, é comum tentar apontar para algum recurso ou algum elemento HTML que ainda não exista, ou seja, que ainda não foi indenizado pelo browser. Quando isso acontece, naturalmente a nossa lógica acaba quebrando. Então, é muito importante compreender essa dinâmica.



- 
- Temos o browser responsável por renderizar os elementos HTML, e por trás, temos a linguagem de programação JavaScript que, muito frequentemente, vai utilizar esses elementos HTML.
  - Mas, para utilizar os elementos HTML, é importante ter em mente que esses elementos precisam estar criados.

- 
- Em geral, utilizamos o JavaScript para interagir com elementos HTML. Esses elementos HTML são renderizados com base em uma árvore de elementos que é conhecida como Document Object Model (DOM) ou Modelo de Objeto de Documento, mais comumente chamado pelo acrônimo, em inglês, DOM.
  - Então, todos os elementos HTML ficam disponíveis no DOM e podem ser, portanto, selecionados a partir da linguagem JavaScript para sofrerem, ou não, algum tipo de ação.


- 
- Por exemplo, estamos com o script incluído dentro do “head” no nosso documento HTML. Isso significa que o nosso JavaScript vai ser processado antes mesmo do nosso “body”.
  - Se, por ventura, quisermos fazer algum tipo de interação entre o JavaScript e o mais novo elemento HTML, como, por exemplo, o “input”, teremos um problema, porque o nosso código JavaScript vai ser executado antes mesmo desse elemento estar montado no DOM e, conseqüentemente, renderizado na página.

# Exemplo

```
<input type="text" placeholder="Teste de Precedência"  
id="nome" disabled="disabled">
```

## Curso de JavaScript

Teste de Precedência

- 
- Nada acontece. Mas porque nada acontece? Simples: porque o nosso script, como dito anteriormente, ele está sendo executado antes mesmo da criação do elemento dentro da nossa página.
  - No momento que ele executa a instrução, ele não encontra esse elemento. Portanto, temos um problema de precedência. O nosso código está sendo executado antes mesmo do elemento ser criado. Isso gera para nós um erro, inclusive, no console do navegador.


# Exemplo

```
<head>
  <meta charset="utf-8">
  <title>Introdução a JavaScript</title>

  <!--Primeira forma -->

  <!-- <script>
    alert('Hello, seja muito bem vindo ao meu site!!');
  </script> -->

  <!-- Segunda forma -->
  <script src="js/introducao.js"></script>
</head>
```

- 
- Para corrigir isso, a precedência de execução do nosso código, em vez de executar o nosso script antes do elemento na página, podemos executar depois. Isso serve tanto para os scripts dentro do nosso documento HTML quanto os scripts importados de arquivos externos.


# Exemplo

```
<body>
  <h1>Curso de JavaScript com Jorge Sant Anna</h1>

  <input type="text" placeholder="Teste de Precedência"
  id="nome" disabled="disabled">


  <!-- Segunda forma -->
  <script src="js/introducao.js"></script>
</body>
```





# Curso de JavaScript com Jorge Sant Anna

Olá

- 
- Repare que agora, no momento da execução do nosso script, o elemento existe no navegador e, portanto, esse elemento pode ser selecionado e o seu valor pode ser modificado.