

# **Introdução: JavaScript na página web II**

**Aula 48**

# Pensamento Computacional

Série: 1ª EM

Introdução: JavaScript na página web II

Aula 48

# O que vamos aprender?



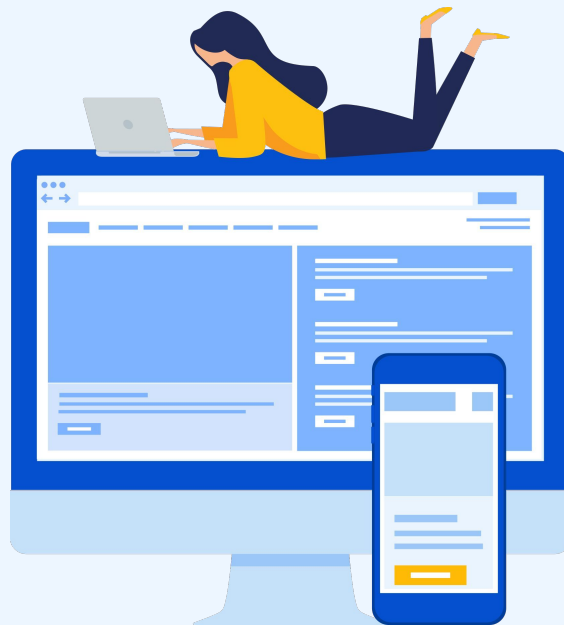
Criar comportamentos dinâmicos no site.



Utilizar o método `querySelector()`.



Aprender boas práticas em JavaScript.



[Acompanhe o vídeo da aula.](#)



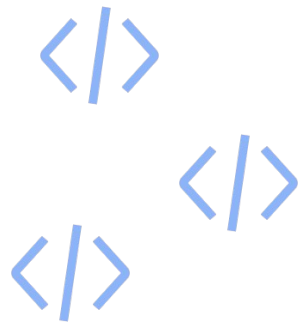
# Compartilhar no GitHub

Durante o desenvolvimento do projeto, você deve salvar no GitHub todas as etapas realizadas na aula e atualizar o seu repositório.

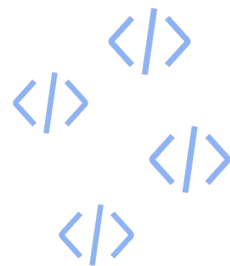
# O método `querySelector`

Agora que já conhecemos a tag `<script>` (onde estará todo o conteúdo de JavaScript) e temos o domínio de duas funções do JavaScript, o `alert` e `console.log`, vamos começar a trabalhar com a interatividade dos dados. Atualmente, no site da Aparecida Nutrição, temos as informações sobre o nome, o peso, a altura, a gordura corporal e o IMC (índice de massa corporal). Na planilha, a coluna do IMC ainda está vazia, sem dados. O primeiro comportamento dinâmico é o de calcular o IMC automaticamente. Faremos isso inserindo um comportamento que fará com que a página fique mais moderna. Também será possível adicionar um novo paciente sem precisar alterar o HTML.

# Aprendendo algumas terminologias da programação



Quando usamos a terminologia “imprimir” no JavaScript , isso se refere ao que irá aparecer na página, para o usuário. Quando se menciona, por exemplo, “imprimir uma função”, isso significa que o resultado daquela função será o que o usuário verá ao navegar por nosso site.



# O método querySelector()

Por meio do JavaScript, podemos transformar um site com comportamento estático – tal como está a página da Aparecida Nutrição – em um site com funcionalidades dinâmicas.

## Meus pacientes

Nome	Peso(kg)	Altura(m)	Gordura Corporal(%)	IMC
Paulo	90	2.00	10	0
João	80	1.72	40	0
Erica	54	1.60	14	0
Douglas	85	1.73	24	0
Tatiana	46	1.55	19	0

# O método `querySelector()`

Estamos trabalhando com uma página desenvolvida em HTML. Todos os códigos contidos na tag

`<script>` serão interpretados como JavaScript. Já os códigos que estiverem fora da tag

`<script>` serão interpretados como HTML.

Nosso objetivo é aprender a modificar alguns elementos do HTML utilizando o JavaScript.

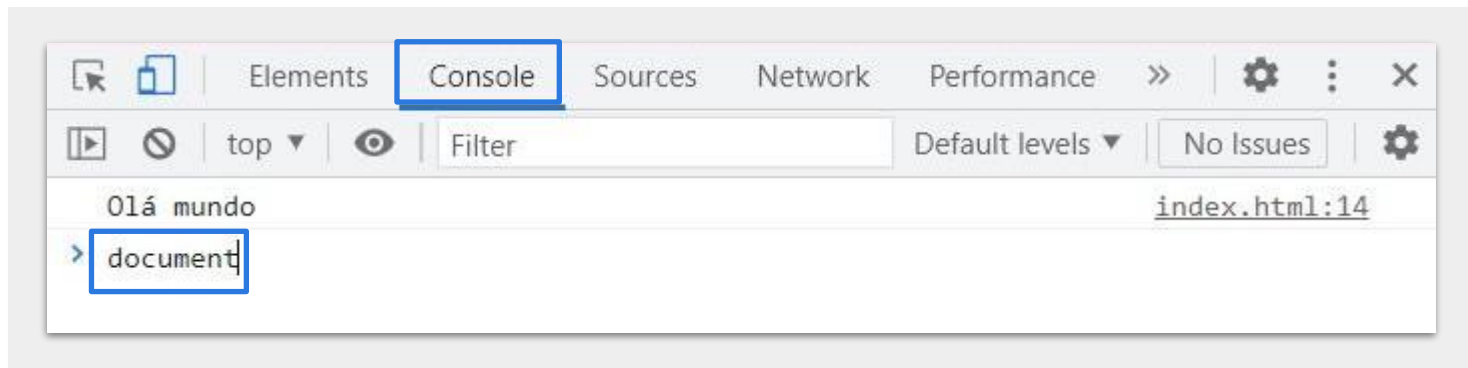
Faremos isso com o título “Meus pacientes”, por exemplo.

Para isso, devemos conhecer o **DOM** (Document Object Model ou, em português, modelo de objeto de documento), que consiste de uma interface de programação para alguns documentos incluindo o HTML. Ele é acessado por meio de uma variável do JavaScript chamada `document`.



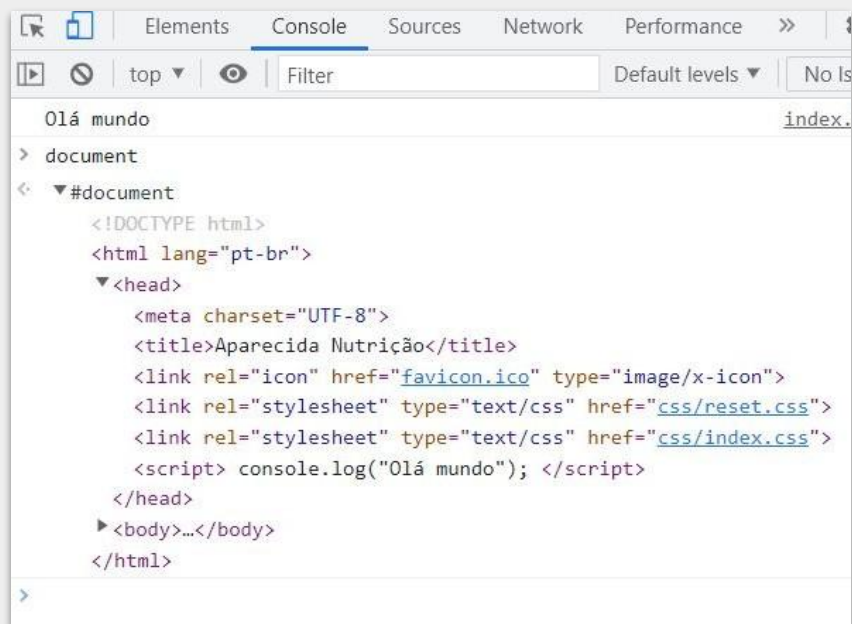
# O método querySelector()

Com o site da Aparecida Nutrição aberto no navegador, para acessar o DOM, devemos ir em console do desenvolvedor – que fica nas ferramentas do desenvolvedor acessado pela tecla F12. Em seguida, clicar em *Console* e, ao lado do sinal de maior na cor azul, digitar `document`.



# O método `querySelector()`

Quando digitamos `document` e pressionamos “enter”, aparecerá todo o código da página HTML, como as tags `<head>` e `<body>`, por exemplo, e todos os seus conteúdos. O `document` será a “ponte” entre o JavaScript e o HTML. Qualquer alteração em `document` será exibida para o usuário.



The screenshot shows a web browser's developer console with the 'Console' tab selected. At the top, there's a toolbar with icons for opening the console, pausing, and a filter input. Below the toolbar, the console displays the text 'Olá mundo' followed by a link to 'index.'. Below this, the 'document' object is expanded, showing the '#document' node. The '#document' node is further expanded, revealing the HTML document structure: `<!DOCTYPE html>`, `<html lang="pt-br">`, `<head>`, `<meta charset="UTF-8">`, `<title>Aparecida Nutrição</title>`, `<link rel="icon" href="favicon.ico" type="image/x-icon">`, `<link rel="stylesheet" type="text/css" href="css/reset.css">`, `<link rel="stylesheet" type="text/css" href="css/index.css">`, `<script> console.log("Olá mundo"); </script>`, `</head>`, `<body>...</body>`, and `</html>`.

# O método `querySelector()`

Podemos adicionar o `document` dentro da tag `<script>` no arquivo `index.html`. Porém, sem o uso de aspas, como fizemos em “Olá mundo”. O `document` funciona como uma variável. As aspas são usadas quando trabalhamos com uma *string* (uma sequência de caracteres) – palavra ou frase que queremos imprimir .

```
<script>
  console.log("Olá mundo");
  console.log(document);
</script>
```

# O método `querySelector()`

Por meio do `document`, conseguimos verificar todo o código HTML da página, porém não conseguimos selecionar uma parte única do arquivo. Se, durante o desenvolvimento do site, precisamos localizar uma parte específica do nosso documento que está em HTML, por exemplo, o título “Aparecida Nutrição”, devemos utilizar o método `querySelector()`.

No arquivo HTML que está sendo trabalhado no editor de texto, vemos que o título “Aparecida Nutrição” é uma tag `<h1>`. Vamos descobrir como localizar o texto dessa tag utilizando o `document`.

```
<header>
  <div class="container">
    <h1>Aparecida Nutrição</h1>
  </div>
</header>
```

# O método `querySelector()`

Retornando à página da Aparecida Nutrição que está aberta no navegador, devemos acessar o console novamente e digitar:

```
document.querySelector(" ");
```

Entre as aspas, devemos especificar o que desejamos localizar no código. Por exemplo, o título do site “Aparecida Nutrição” está em `<h1>`. Ao digitarmos `document.querySelector("h1")`; e pressionarmos a tecla enter, veremos em destaque o conteúdo da página correspondente ao `<h1>`, ou seja, o título.

# O método querySelector()

Aparecida Nutrição

## Meus pacientes

Nome	Peso(kg)
Paulo	100
João	80
Erica	54
Douglas	85
Tatiana	46

```
Olá mundo
> document
< ▼ #document
  <!DOCTYPE html>
  <html lang="pt-br">
    ▶ <head>...</head>
    ▶ <body>...</body>
  </html>
> document.querySelector("h1");
```

Após digitarmos o código e pressionar enter, aparecerá o texto contido na tag

```
<h1>
Olá mundo
> document
< ▼ #document
  <!DOCTYPE html>
  <html lang="pt-br">
    ▶ <head>...</head>
    ▶ <body>...</body>
  </html>
> document.querySelector("h1");
< <h1>Aparecida Nutrição</h1>
>
```

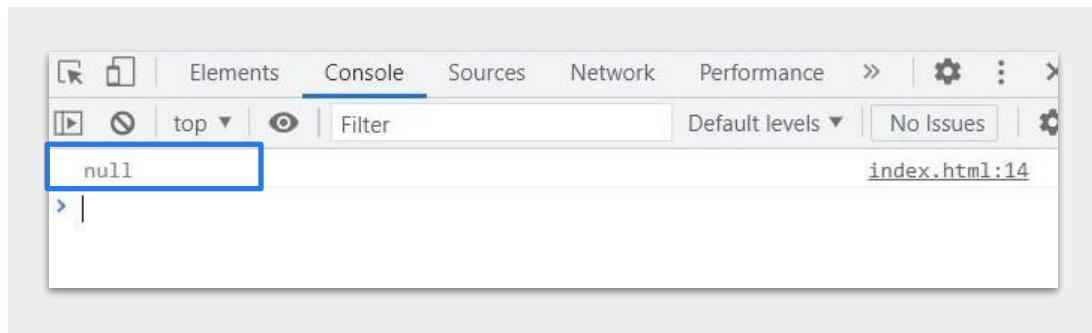
# O método `querySelector()`

Depois de testar o método `querySelector()`, devemos incluí-lo no código do arquivo `index.html`. Na tag `<script>`, devemos adicionar o objeto `console.log` para imprimir o resultado da busca por `<h1>`. Também devemos apagar todo conteúdo da tag `<h1>`, deixando apenas o `console.log` para o `querySelector()`.

```
<script>  
  console.log(document.querySelector("h1"));  
</script>
```

# O método `querySelector()`

Ao salvar o projeto e recarregar o console no navegador, aparecerá o resultado *null* (nulo).



Foi possível executar o mesmo código corretamente direto no console, porém no arquivo não obtivemos o resultado esperado. É possível entender por que isso ocorre analisando a estrutura do arquivo `index.html`.



# O método querySelector()

O navegador, ao carregar a página HTML, lê linha por linha, de cima para baixo. Ao chegar na tag

`<script>`, ele tenta buscar um `<h1>` na página, mas a tag `<h1>` está abaixo do código JavaScript e, por isso, ainda não foi interpretada pelo navegador. Como consequência, o `<h1>` não poderá ser selecionado.

A tag `<script>` termina, ou seja, é fechada antes de iniciar a tag `<body>`, que, por sua vez, contém a tag `<h1>`, com o título “Aparecida Nutrição”.

```
index.html X
index.html > html > body > main > section.container > table > thead > tr
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4 <meta charset="UTF-8">
5 <title>Aparecida Nutrição</title>
6 <link rel="icon" href="favicon.ico" type="image/x-icon">
7 <link rel="stylesheet" type="text/css" href="css/reset.css">
8 <link rel="stylesheet" type="text/css" href="css/index.css">
9
10 </head>
11
12
13 <script>
14   console.log(document.querySelector("h1"));
15 </script>
16
17 <body>
18   <header>
19     <div class="container">
20       <h1>Aparecida Nutrição</h1>
21     </div>
22   </header>
23   <main>
24     <section class="container">
25       <h2>Meus pacientes</h2>
26       <table>
```

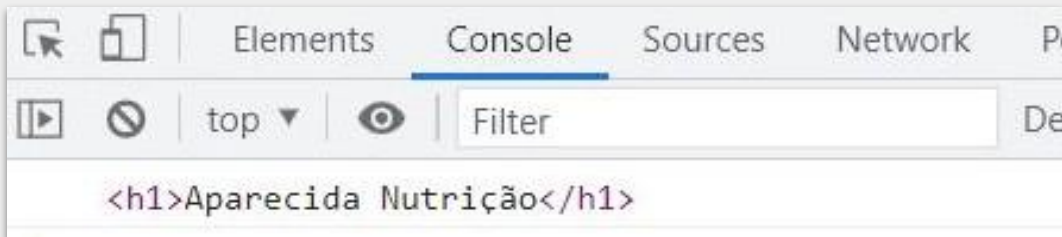
# O método `querySelector()`

Uma boa prática é ter o código em JavaScript em um arquivo distinto dos códigos em HTML. Porém, caso não seja possível, devemos manter o código JavaScript como o último elemento da tag `<body>`.

Vamos recortar toda a tag `<script>` e seu conteúdo e colar no final da tag `</main>`. Com isso, o `querySelector()` já será identificado no console.

```
</main>
<script>
    console.log(document.querySelector("h1"));
</script>

</body>
```



# O método `querySelector()`

Aprendemos a selecionar o título Aparecida Nutrição utilizando o método `querySelector()`. Mas, além de selecionar, como poderíamos alterar o seu conteúdo? Como faríamos, por exemplo, para que o texto seja Aparecida Nutricionista?

Para isso, devemos criar a variável `var titulo`, na qual salvaremos o `<h1>`. No JavaScript, as variáveis são declaradas com `var`, `let` ou `const`. Nós utilizaremos a declaração `var`. Para essa etapa, a tag

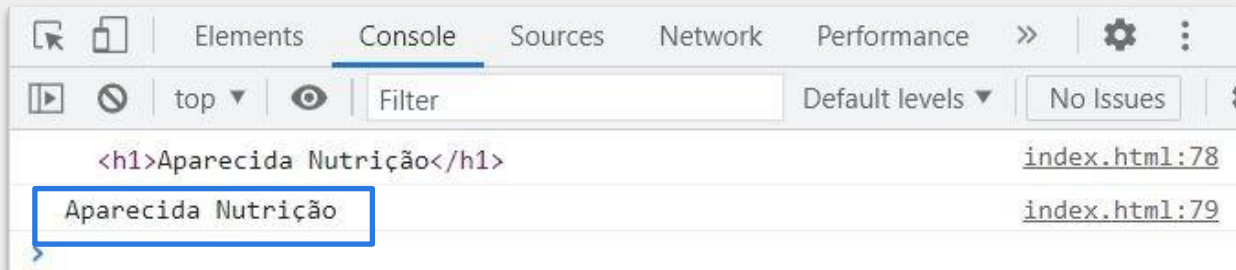
`<script>` deverá ficar da seguinte maneira:

```
<script>
  var titulo = document.querySelector("h1");
  console.log(titulo);
</script>
```

# O método `querySelector()`

Com a variável `titulo` já criada, vamos aprender a selecionar o texto “Aparecida Nutrição” do título. Como o conteúdo está em texto, o JavaScript possui uma propriedade que permite acessá-lo: `textContent`.

```
<script>
  var titulo = document.querySelector("h1");
  console.log(titulo);
  console.log(titulo.textContent);
</script>
```



# O método `querySelector()`

Para alterar o conteúdo da variável `titulo`, devemos modificar a propriedade `textContent` (conteúdo de texto). O que fizemos foi: obter uma variável que corresponde ao elemento `titulo`, que permite buscá-la por meio do `querySelector()`. A partir disso, alteramos a propriedade na variável `titulo.textContent`, que, por consequência, alterou o título Aparecida Nutrição para Aparecida Nutricionista.

```
<script>
  var titulo = document.querySelector("h1");
  titulo.textContent = "Aparecida Nutricionista";
</script>
```

Aparecida Nutricionista

Meus pacientes

# Boas práticas em JavaScript

Até agora, aprendemos:

- a usar o comando `querySelector()` para selecionar e exibir o texto escrito no título.
- a criar uma variável `var` para discriminar o que está sendo pesquisado, `var titulo`.
- a alterar o texto do título com o atributo `titulo.textContent`.



Além desse aprendizado sobre comandos, propriedades e funções de uma linguagem de programação, aprenderemos **boas práticas** em JavaScript.

# Boas práticas em JavaScript

Se, por exemplo, alterarmos o título `<h1>` (Aparecida Nutrição) no HTML, não será mais possível executar os comandos da tag `<script>`, pois nenhum `<h1>` será localizado.

Caso a pessoa desenvolvedora decida fazer alterações no HTML, o código deixará de funcionar. Isso é um problema quando se utiliza a busca de uma tag por meio de `document`. Por isso, não buscar uma tag HTML específica é uma boa prática. Em vez disso, podemos usar outras opções que a função `querySelector()` disponibiliza.

# Boas práticas em JavaScript

Além das tags do HTML, o `querySelector()` permite buscar elementos de **classe** e de **id** em todos os elementos dos códigos que estão nos arquivos de HTML e CSS.

Uma boa solução para alterar o `<h1>` é transformá-lo em `<h2>` e também criar uma classe. Podemos chamar essa classe de `titulo`.

```
<h2 class="titulo">Aparecida Nutrição</h2>
```



# Boas práticas em JavaScript

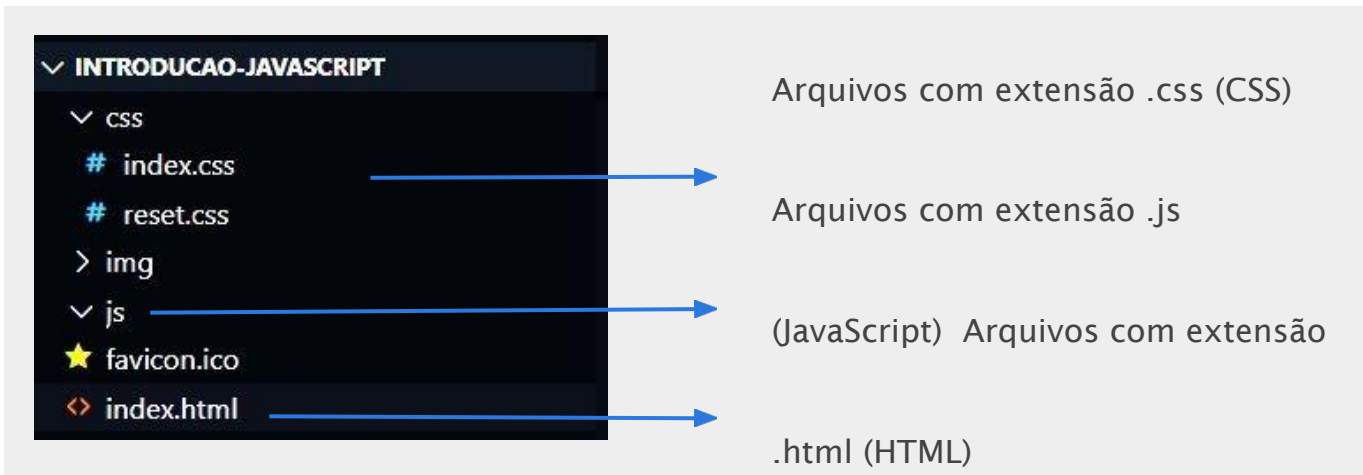
Consequentemente, também será preciso ajustar a tag `<script>` para incluir a classe `titulo`, e não mais a tag `<h1>`.

```
<script>
var titulo = document.querySelector(".titulo");
titulo.textContent = "Aparecida Nutricionista";
</script>
```

Desse modo, se outra pessoa desenvolvedora modificar a tag `<h1>` no código HTML, ela não será prejudicada, e o JavaScript continuará sendo executado corretamente.

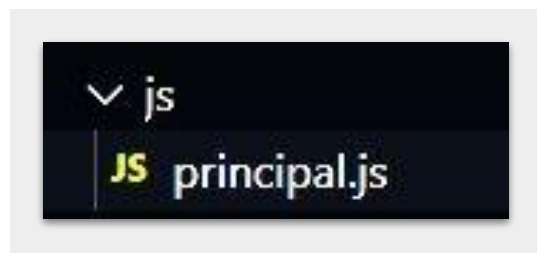
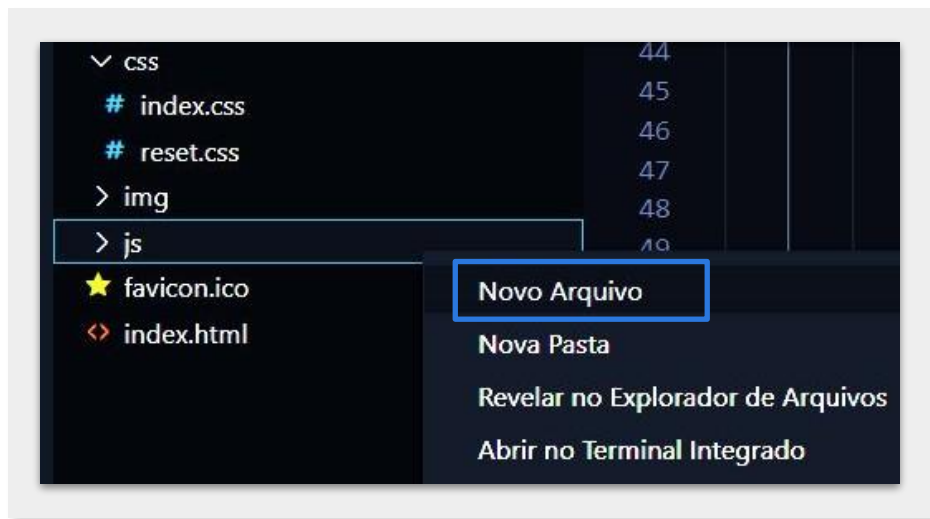
# Boas práticas em JavaScript

Outra boa prática é desvincular o código JavaScript do arquivo HTML (index.html). Aprendemos no curso de HTML/CSS que cada um dos códigos deve estar em arquivos distintos, de acordo com a sua extensão (.css, .html, .js, etc.). No editor de texto, por exemplo, há pastas específicas para cada uma das extensões.



# Boas práticas em JavaScript

Na pasta de JavaScript (js), devemos criar um novo arquivo que irá conter todo nosso código. Faremos isso com a opção *Novo Arquivo* ou *New File*. Essa pasta deve ser nomeada como “principal.js” e será referenciada no código HTML como um **arquivo externo**. Aprenderemos a fazer isso em seguida.

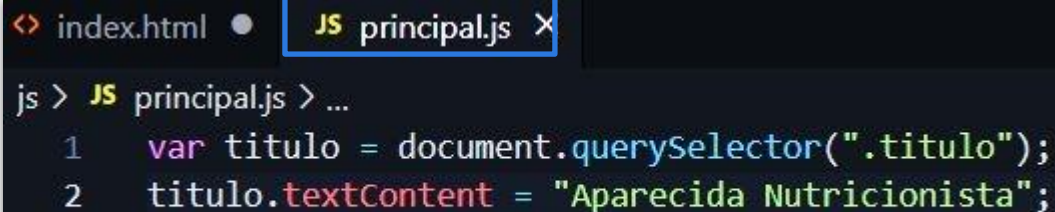


# Boas práticas em JavaScript

Todo conteúdo da tag `<script>` será recortado e colado no arquivo `principal.js`. No HTML, a tag

`<script>` deverá ficar vazia por enquanto.

O conteúdo do arquivo `principal.js` deverá permanecer com os mesmos parâmetros.



```
<> index.html • JS principal.js X
js > JS principal.js > ...
1  var titulo = document.querySelector(".titulo");
2  titulo.textContent = "Aparecida Nutricionista";
```

# Boas práticas em JavaScript

A tag `<script>` continuará na página `index.html`, mas ela não ficará vazia. O atributo `src` (referente ao termo *source*, que significa “fonte”, em inglês) apontará para o arquivo JavaScript externo que criamos. Como o arquivo `principal.js` está na pasta `js`, iremos referenciar onde o conteúdo de JavaScript será encontrado.

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

# Desafio

Testamos muitas funcionalidades do nosso código utilizando o console, disponível nas ferramentas do desenvolvedor. A função `querySelector()` pode ser executada no console com a variável `document`, e, a partir dela, buscarmos uma parte específica do código HTML. Para praticar mais essa possibilidade, experimente analisar o código em seu editor de texto e utilizar a função `querySelector()` para localizar um elemento específico no console.

