

JavaScript

Tecnologias Web
2022/2023

1

JavaScript

- *Client-side scripting language*
 - Interpretado directamente pelo *browser (on the fly)*, não necessita de ser compilado



2

Inserção de Scripts

embedded / ficheiro externo

3

Inserção de Scripts

■ *Embedded script:*

- Diretamente no ficheiro HTML entre as tags `<script> ... </script>`

```
<style>
  button{background-color: orange;
          border:2px solid darkorange;
          color:white; font-size: 1.3em;
          width:100px;height:50px;}
</style>

</head>
<body>
  <button id="evt1" onclick="change()">Alert!</button>

  <script>
    function change(){
      alert ('First Script!')
    }
  </script>
</body>
```

Alert!

onclick

This page says
First Script!

OK

■ *Ficheiro Externo *.js*

- `<script src="exemplo.js" type="text/javascript"></script>`

4

Eventos

5

Eventos JS

■ Evento

- Ação que pode ser detectada pelo *JavaScript* e que provoca uma execução específica:
 - Chamada de uma função, em que a função só é executada após a ocorrência do respectivo evento.
 - *event handlers* (utilizados na ficha prática nº10)

Mouse Events

Property	Description
→ onclick	The event occurs when the user clicks on an element
ondblclick	The event occurs when the user double-clicks on an element
onmousedown	The event occurs when a user presses a mouse button over an element
onmousemove	The event occurs when the pointer is moving while it is over an element
→ onmouseover	The event occurs when the pointer is moved onto an element
→ onmouseout	The event occurs when a user moves the mouse pointer out of an element
onmouseup	The event occurs when a user releases a mouse button over an element

6

Eventos JS

- Duas formas distintas de declarar um evento:

- *HTML Event Handlers*

` ... `

```
<body>
  <button id="evt1" onclick="change()">Alert!</button>

  <script>
    function change(){
      alert ('First Script!')
    }
  </script>
</body>
```

Alert!

onclick

This page says
First Script!

OK

7

Eventos JS

- Duas formas distintas de declarar um evento:

- *DOM Event Listeners*

Alert!

click

This page says
First Script!

OK

`element.addEventListener('event',functionName,[Boolean]);`

```
<body>
  <button id="evt1">Alert!</button>

  <script>
    function change(){
      alert('First Script!');
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

8

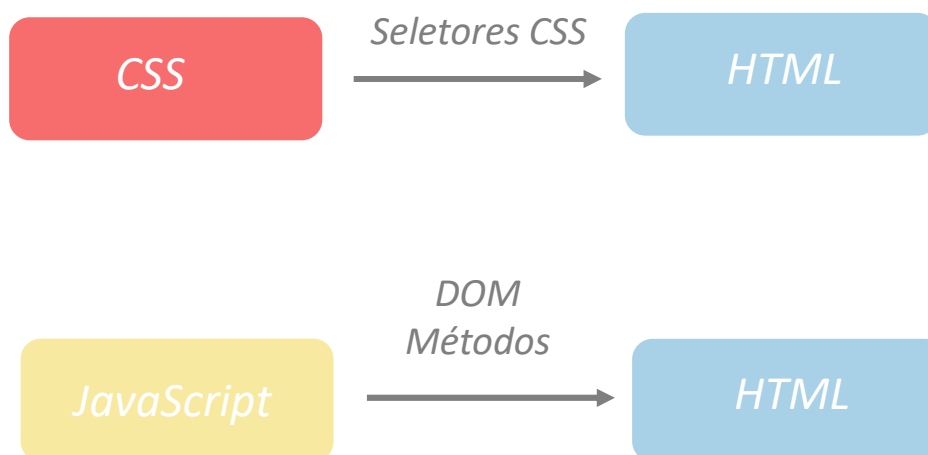
DOM

(Métodos para Aceder a Elementos HTML)

9

Document Object Model

- Acesso a Elementos HTML



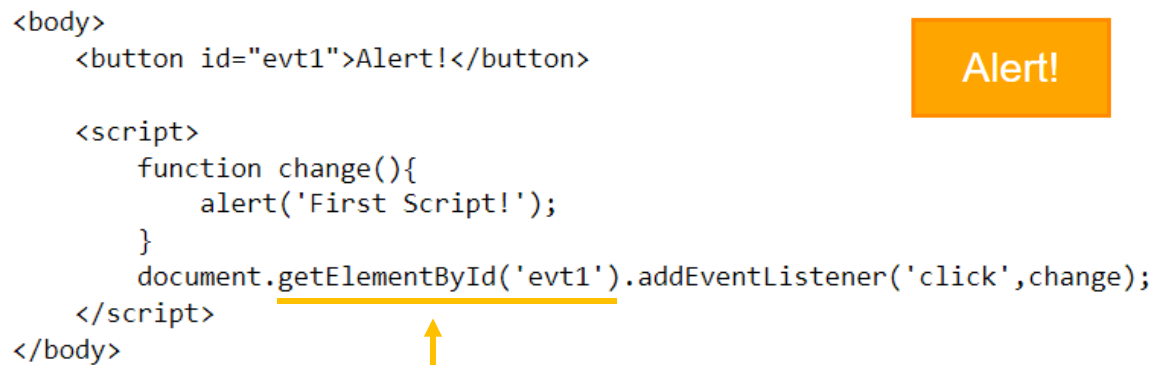
10

■ Métodos para acceder a elementos HTML

Método (<i>DOM Queries</i>)	Descrição
document.getElementById()	retorna o elemento que tem o id com o valor especificado
document.querySelector('.xpto')	retorna o 1º elemento que tem a class xpto
document.querySelector('#xpto')	retorna o elemento que tem o id xpto

```
<body>
  <button id="evt1">Alert!</button>

  <script>
    function change(){
      alert('First Script!');
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

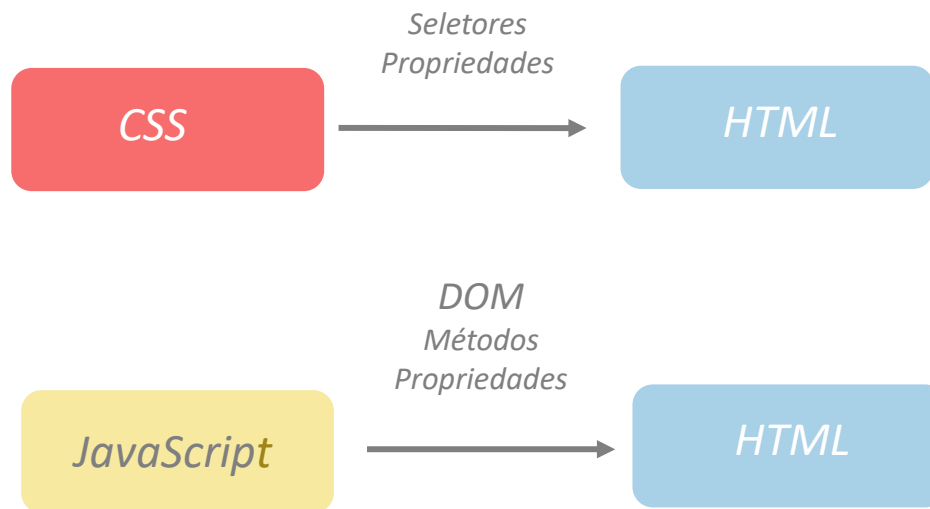


DOM

Propriedades

Document Object Model

■ Alteração de Elementos HTML



13

DOM

■ DOM Properties

<i>Propriedades</i>	<i>Descrição</i>
<i>innerHTML</i>	<i>acesso/alteração do conteúdo HTML de um elemento</i>
<i>textContent</i>	<i>acesso/alteração de texto</i>

```
<body>
  <button id="evt1">Alert!</button>

  <script>
    function change(){
      document.getElementById('evt1').textContent="DOM properties!";
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

Alert!

click

DOM properties!

14

DOM

■ DOM Properties

<i>Propriedades</i>	<i>Descrição</i>
<i>classList</i>	<i>Lista (DOMTokenList) dos valores do atributo class de um elemento</i>

Alert!

click

This page says
v1 v2 v3

OK

```
<body>
  <button id="evt1" class="v1 v2 v3">Alert!</button>

  <script>
    function change(){
      alert(document.getElementById('evt1').classList);
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

15

DOM

■ Três métodos importantes da DOMTokenList gerada pela **classList**:

- contains()
- add()
- remove()

Alert!

This page says
true

OK

This page says
false

OK

```
<body>
  <button id="evt1" class="v1 v2 v3">Alert!</button>

  <script>
    function change(){
      alert(document.getElementById('evt1').classList.contains('v1'));
      document.getElementById('evt1').classList.remove('v1');
      alert(document.getElementById('evt1').classList.contains('v1'));
    }

    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

16

DOM

Métodos para Alteração de Atributos

17

DOM

■ Alteração de atributos.

■ *setAttribute()*

- permite múltiplas alterações: imagem (src); destino (href); estilo (style); ...

```
<body>
  <button id="evt1" class="v11 v12 v13">Alert!</button>

  <script>
    function change(){
      document.getElementById('evt1').setAttribute('style','background-color:lightblue');
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```

Alert!

click

Alert!


18

■ Alteração de atributos.

■ Propriedade **style**

```
<body>
  <button id="evt1" class="v11 v12 v13">Alert!</button>

  <script>
    function change(){
      document.getElementById('evt1').style.backgroundColor='lightblue';
    }
    document.getElementById('evt1').addEventListener('click',change);
  </script>
</body>
```



The diagram illustrates the state of the button before and after a click event. On the left, an orange button labeled "Alert!" represents the initial state. In the center, the word "click" indicates the event that triggers the change. On the right, a light blue button labeled "Alert!" represents the state after the event, where the background color has been updated via the DOM's style property.

Sintaxe

Sintaxe

- *case sensitive*.
- `//` símbolo do comentário
 - `/*` comentário para múltiplas linhas `*/`
- Declaração de variáveis

- *loosely typed*

- `var`
- `let`

```
<body>
  <button id="evt1" class="v11 v12 v13">Alert!</button>

  <script>
    let x=document.getElementById('evt1');

    function change(){
      x.style.backgroundColor='lightblue';
    }
    x.addEventListener('click',change);
  </script>
</body>
```

21

Sintaxe

- Os *code blocks* (conjuntos de instruções) são delimitados por `{ ... }`
- Declaração de uma função (uma das formas possíveis):

```
function calculateSum (a,b = 1){
  return a+b;
}
```

- Estrutura de seleção:

```
if(condição)
  {...}
else
  {...}
```

22