INSTITUTO FEDERAL
Paraná
Campus Paranaguá

Ministério da Educação

TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO
Disciplina de Desenvolvimento Web II
Aula 01: HTML DOM / Eventos / JavaScript

Gil Eduardo de Andrade

MANIPULANDO DOM - JAVASCRIPT

(https://www.w3schools.com/jsref/dom_obj_document.asp)

O DOM (Document Object Model) ou Modelo de Objeto de Documento, trata-se de uma representação hierárquica dos elementos HTML que compõem uma página web. Tal abordagem permite que os elementos HTML, e seus atributos, sejam acessados, modificados e atualizados através de linguagens de programação, tais como o JavaScript.

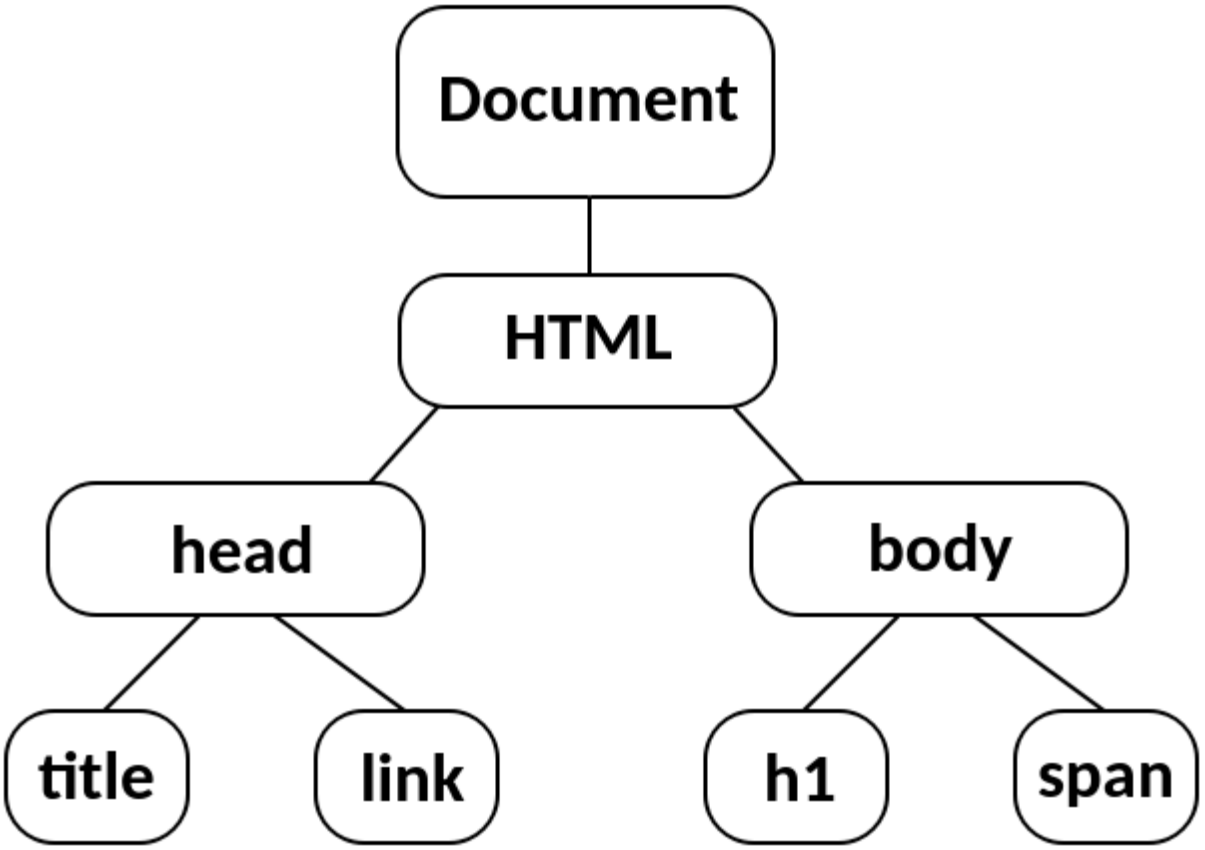


Figura 01: Representação gráfica - DOM

EVENTOS DOM - JAVASCRIPT

A linguagem JavaScript permite, como uma de suas principais funcionalidades, que o DOM (Document Object Model) seja manipulado, possibilitando que a interação com os elementos da página ocorra, tendo como base para tal os eventos.
Através destes eventos, torna-se possível adicionar interatividade e dinamismo às páginas web que estão sendo criadas. Em outras palavras, ao manipular eventos, através da linguagem JavaScript, é possível especificar as ações que deverão ser executadas para cada um dos eventos que estão sendo monitorados (clicar num botão, passar o mouse sobre uma imagem, etc).
A seguir é apresentado uma tabela que mapeia as propriedades CSS, dos elementos HTML, que podem ser acessadas e modificadas, através de linguagem JavaScript, quando um ou mais eventos ocorrem.

Tabela 01: Mapeamento das propriedades CSS no JavaScript.

Propriedade – CSS	Referência – JavaScript (style)
background	background
background-attachment	backgroundAttachment
background-color	backgroundColor
background-image	backgroundImage
background-position	backgroundPosition
background-repeat	backgroundRepeat
border	border
border-bottom	borderBottom
border-bottom-color	borderBottomColor
border-bottom-style	borderBottomStyle
border-bottom-width	borderBottomWidth
border-color	borderColor
border-left	borderLeft
border-left-color	borderLeftColor
border-left-style	borderLeftStyle
border-left-width	borderLeftWidth
border-right	borderRight
border-right-color	borderRightColor
border-right-style	borderRightStyle
border-right-width	borderRightWidth
border-style	borderStyle
border-top	borderTop
border-top-color	borderTopColor

border-top-style	borderTopStyle
border-top-width	borderTopWidth
border-width	borderWidth
clear	clear
clip	clip
color	color
cursor	cursor
display	display
filter	filter
font	font
font-family	fontFamily
font-size	fontSize
font-variant	fontVariant
font-weight	fontWeight
height	height
left	left
letter-spacing	letterSpacing
line-height	lineHeight
list-style	listStyle
list-style-image	listStyleImage
list-style-position	listStylePosition
list-style-type	listStyleType
margin	margin
margin-bottom	marginBottom
margin-left	marginLeft
margin-right	marginRight
margin-top	marginTop
overflow	overflow
padding	padding
padding-bottom	paddingBottom
padding-left	paddingLeft
padding-right	paddingRight
padding-top	paddingTop
page-break-after	pageBreakAfter
page-break-before	pageBreakBefore
position	position
float	styleFloat
text-align	textAlign
text-decoration	textDecoration
text-decoration: blink	textDecorationBlink

text-decoration: line-through	textDecorationLineThrough
text-decoration: none	textDecorationNone
text-decoration: overline	textDecorationOverline
text-decoration: underline	textDecorationUnderline
text-indent	textIndent
text-transform	textTransform
top	top
vertical-align	verticalAlign
visibility	visibility
width	width
z-index	zIndex

A seguir são apresentados alguns exemplos de eventos DOM, manipulados através da linguagem JavaScript, que permitem alterar as propriedades CSS dos elementos HTML, inserindo interatividade e dinamismos nas páginas web.

Arquivo: ./javascript/DOM e Eventos/exemplo_inicial.html

JavaScript

```
<div id="manipula">Eu sou a frase do exemplo. Serei alterada
quando você clicar.</div>

<hr>

<h2>Altere a frase acima clicando nos botões</h2>
<button onclick="mudaCor()">Muda cor da fonte</button>

<input type="button" value="Muda tamanho da fonte"
onclick="mudaTamanhoFonte()"/>

<input type="button" value="Muda background"
onclick="mudaBackground()"/>

<br><br>

<input type="button" value="Deixa em italico"
onclick="mudaItalico()"/>
```

```
<input type="button" value="Coloca borda"
onclick="colocaBorder()"/>

<input type="button" value="Muda a cor da borda"
onclick="corBorder()"/>

<br><br>

<input type="button" value="Exclui borda"
onclick="excluiBorder()"/>

<input type="button" value="Esconde a div"
onclick="escondeDiv()"/>

<input type="button" value="Reaparece com a div"
onclick="reapareceDiv()"/>
```

JavaScript

```
#manipula {
      height:50px;
      line-height:50px;
      color: yellow;
      font-size:10pt;
      background-color: red;
      margin-bottom:15px;
}
```

JavaScript

```
function mudaCor() {
      document.getElementById("manipula").style.color =
"#ffffff";
```

```
}

function mudaTamanhoFonte() {
      document.getElementById("manipula").style.fontSize =
"20pt";
}

function mudaBackground() {
      document.getElementById("manipula").style.backgroundColor
= "green";
}

function mudaItalico() {
      document.getElementById("manipula").style.fontStyle =
"italic";
}

function escondeDiv() {
      document.getElementById("manipula").style.display =
"none";
}

function reapareceDiv() {
      document.getElementById("manipula").style.display =
"block";
}

function colocaBorder() {
      document.getElementById("manipula").style.border = "2px
solid black";
}

function corBorder() {
      document.getElementById("manipula").style.border = "2px
solid #0000ff";
}
```

```
function excluiBorder() {
      document.getElementById("manipula").style.border = "0px";
}
```

Os conceitos apresentados anteriormente (eventos DOM, propriedades CSS, JavaScript) serão abordados, a partir de agora, de maneira mais detalhada neste documento.

MANIPULAÇÃO DO HTML

Utilizando Método: “document.querySelector()”

Selecionar Elemento

JavaScript

```
document.querySelector("tag");          // Pela <tag> HTML
document.querySelector("#id");          // Pelo ID do elemento HTML
document.querySelector(".class");       // Pela classe CSS
```

OBS.: o comando “***querySelector()***” seleciona apenas o primeiro elemento encontrado para <tag> HTML, identificador (id) do elemento HTML e classe CSS.

OBS.: para selecionar todos os elementos encontrados para <tag> HTML, identificador (id) do elemento HTML e classe CSS utilizamos o comando "***querySelectorAll()***".

Outros Métodos Similares - "querySelector()"

Selecionar Elemento

```javascript
document.getElementsByTagName("tag");    // Pela <tag> HTML
document.getElementById("id");           // Pelo ID do elemento HTML
document.getElementsByClassName("class"); // Pela classe CSS
```

OBS.: sempre que possível utilize o comando "***querySelector()***", os comandos acima foram apresentados com intuito de ampliar os conhecimentos sobre o tema, porém, sendo anteriores ao surgimento do "***querySelector()***".

Utilizando Propriedade: "textContent"

Alterar Conteúdo do Elemento

```javascript
let titulo = document.querySelector("h1");
titulo.textContent = "Novo Título";   // Altera o Conteúdo
```

Outra Propriedade Similar - "textContent"

Alterar Conteúdo do Elemento

```javascript
let titulo = document.querySelector("h1");
titulo.innerHTML = "Novo Título";     // Altera o Conteúdo
```

OBS.: sempre que possível utilize a propriedade "***textContent***", a propriedade acima foi apresentada com intuito de ampliar os conhecimentos sobre o tema, porém, é anterior ao surgimento da "***textContent***".

Utilizando o Método: "setAttribute"

Configurar Atributos HTML

```javascript
let imagem = document.querySelector("#logo");
imagem.setAttribute("src", "./img/logo.png");   // Altera o Atributo
imagem.setAttribute("width", "250px");          // Altera o Atributo
```

MANIPULAÇÃO DO CSS

Utilizando Propriedade: "style"

Aplicar Estilo ao Elemento

```javascript
let titulo = document.querySelector("h1");
titulo.style.color = "red";                 // Altera o Estilo
titulo.style.backgroundColor = "#AAA";      // Altera o Estilo
titulo.style.border = "1px solid red";      // Altera o Estilo
titulo.style.borderRadius = "10px";         // Altera o Estilo
```

OBS.: a abordagem apresentada anteriormente, utilizando a propriedade "***style***", é indicada para situações onde apenas uma ou duas propriedades de estilo serão configuradas. A partir do momento em que é necessário um conjunto maior, torna-se mais adequado utilizar a abordagem apresentada a seguir, utilizando o método "***setAttribute()***" para alterar a propriedade "*class*".

Utilizando Propriedade: "setAttribute('class', '...')"

Aplicar Estilo ao Elemento

```javascript
let titulo = document.querySelector("h1");
titulo.setAttribute("class", "dark");  // Altera o Atributo
```

```
.dark {
      color: red;
      background-color: #AAA;
      border: 1px solid red;
      border-radius: 10px;
}
```

OBS.: a abordagem apresentada, se comparada a anterior, permite efetuar a configuração do estilo através de uma única linha, em contrapartida as quatro utilizadas anteriormente. Obviamente, para tal, seria necessário criar a classe **"*dark*"** contendo todo o estilo que deve ser aplicado.

Utilizando Propriedade: "removeAttribute()"

Remover Estilo do Elemento

```javascript
let titulo = document.querySelector("h1");
titulo.removeAttribute("class");      // Remove o Atributo
```

OBS.: é importante salientar que o método "removeAttribute()", para propriedade "class", remove todas as classes de estilo que tenham sido aplicadas ao elemento HTML em questão. A seguir serão apresentados métodos que permitem especificar quais classes devem ser removidas ou adicionadas ao estilo de uma determinado elemento HTML.

Utilizando Métodos: "classList.add()" e "classList.remove()"

Remover Estilo do Elemento

Arquivo: ./javascript/DOM e Eventos/setAtributte

Código HMTL

```JavaScript
<button id="btn_main">
  Main Mode
</button>
<button class="btn_dark" id="btn_dark">
  Dark Mode
</button>
<button class="btn_light" id="btn_light">
  Light Mode
</button>

<h1 class="init" id="titulo">
  GIL EDUARDO DE ANDRADE
</h1>
```

Código CSS

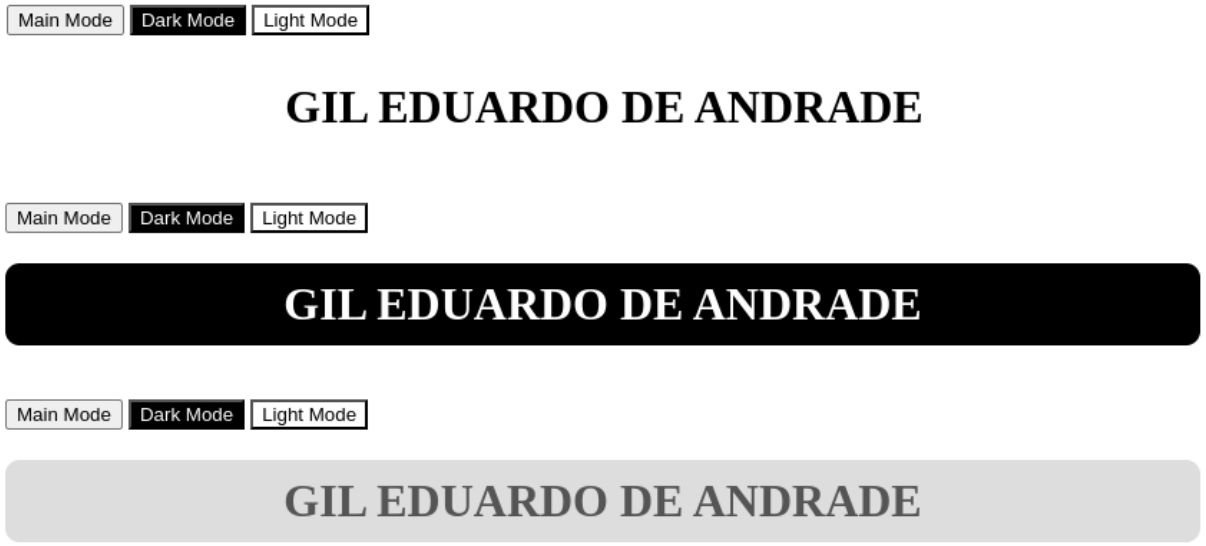
```JavaScript
.btn_dark {
  color: white;
  background-color: black;
}

.btn_light {
  color: black;
  background-color: white;
}

.init {
  display: flex;
  border-radius: 10px;
  padding: 10px 10px;
  justify-content: center;
}

.dark {
  color: white;
```

```
  background-color: black;
}

.light {
  color: #555;
  background-color: #DDD;
}
```

Código JavaScript

```JavaScript
let btnM = document.querySelector("#btn_main")
let btnD = document.querySelector("#btn_dark")
let btnL = document.querySelector("#btn_light")
let tit = document.querySelector("#titulo")

btnD.addEventListener("click", darkMode)
btnL.addEventListener("click", lightMode)
btnM.addEventListener("click", mainMode)

function mainMode() {
  tit.classList.remove("dark")
  tit.classList.remove("light")
}

function darkMode() {
  tit.classList.remove("light")
  tit.classList.add("dark")
}

function lightMode() {
  tit.classList.remove("dark")
  tit.classList.add("light")
}
```

OBS.: os conteúdos de funções e eventos JavaScript, utilizados no exemplo acima - ***function()*** | ***addEventListener()*** - serão abordados em detalhes na sequência.



Resultado - Codificação Anterior

FUNÇÕES JAVASCRIPT - BÁSICO

(https://www.w3schools.com/js/js_functions.asp)

Funções / Funções Anônimas / Arrow Functions

Diferentes formas de declaração

Uma função pode ser definida como um conjunto de instruções, ou bloco de código, criado para realizar uma uma tarefa específica. No JavaScript, uma função é executada quando um determinado trecho de código a invoca.

Funções Anônimas (Anonymous Functions)
Uma função anônima, em JavaScript, é uma função que não possui um nome. Este tipo de função pode ser atribuída a uma variável ou passada como parâmetro para outras funções. As funções anônimas possibilitam que a escrita do código seja feita de forma clara e eficiente, necessitando de uma quantidade menor de linhas.

```JavaScript
// Não possui um nome definido após a palavra-chave function
// Atribuímos a função a variável para que seja possível invocá-la
const show = function() {
      return "Anonymous Function";
```

```
}
console.log(show());
```

Funções Seta (Arrow Functions)
As *arrow functions* possuem uma sintaxe mais simples (reduzida), clara e moderna para escrita de funções em JavaScript - são assim chamadas devido à seta (=>) usada na sua sintaxe.

JavaScript
```
// Atribuímos a função a variável para que seja possível invocá-la
const show = () => "Arrow Function";
console.log(show());
```

Utilizando: Anonymous e Arrow Functions

Contador Simples

Arquivo: ./javascript/DOM e Eventos/functions

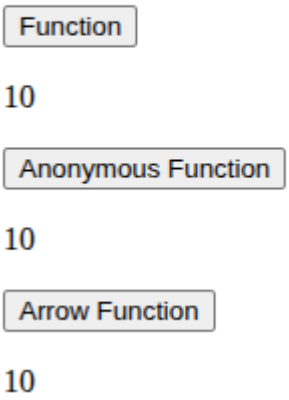
JavaScript
```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width,
initial-scale=1.0">
        <title>Funções - JavaScript</title>
    </head>
    <body onload="setValues()">
        <h1 class="init" id="titulo">FUNÇÕES JAVASCRIPT</h1>

        <button id="btn_fun"> Function </button>
        <p id="v1"></p>

        <button id="btn_ano"> Anonymous Function </button>
        <p id="v2"></p>
```

```
        <button id="btn_arr"> Arrow Function </button>
        <p id="v3"></p>
    </body>
    <script type="text/javascript">

        let val = 0;
        let fun = document.querySelector('#btn_fun');
        let ano = document.querySelector('#btn_ano');
        let arr = document.querySelector('#btn_arr');

        function setValues(num=0) {
            val+=num;
            document.querySelector("#v1").textContent = val;
            document.querySelector("#v2").textContent = val;
            document.querySelector("#v3").textContent = val;
        }

        // FUNCTION
        function func() { setValues(1); }

        // ANONYMOUS FUNCTION
        const anon = function () { setValues(2); }

        // ARROW FUNCTION
        const arro = () =>  setValues(3);

        // CLICK EVENTS
        fun.addEventListener("click", func);
        ano.addEventListener("click", anon);
        arr.addEventListener("click", arro);
    </script>
</html>
```

INSTITUTO FEDERAL
Paraná
Campus Paranaguá

Ministério da Educação

FUNÇÕES JAVASCRIPT



EVENTOS JAVASCRIPT

(https://www.w3schools.com/js/js_events.asp)

Principais Eventos DOM | JavaScript

Manipulando Eventos

Eventos HTML são “ações” que ocorrem com os elementos HTML, que podem ser detectados pela linguagem JavaScript. Um evento HTML pode ser uma ação que o navegador faz, ou uma ação que o usuário faz, como por exemplo:

- Uma página da web HTML terminou de carregar
- Um campo de entrada HTML foi alterado
- Um botão HTML foi clicado

Neste contexto, quando os eventos ocorrem, pode ser necessário reagir a eles, ou seja, através da linguagem JavaScript é possível executar rotinas específicas para cada um deles.
Para tal, o HTML possibilita que atributos do manipulador de eventos, com código JavaScript, sejam adicionados a elementos HTML.

Tabela 02: Principais eventos HTML.

Evento	Descrição
onclick	Elemento é clicado
ondblclick	Elemento recebe duplo click
onmouseover	Mouse é posicionado sob um elemento

onmouseenter	Mouse entra na área compreendida por um elemento
onmouseout	Mouse sai da área compreendida por um elemento
onkeypress	Tecla é pressionada
onkeydown	Tecla é pressionada (antes do onkeypress)
onsubmit	Formulário é enviado / submetido
onscroll	Uma página é rolada, via scroll
onfocus	Elemento obtém foco
onblur	Elemento perde foco
onload	Elemento, ou toda a página, é carregado
onresize	Janela do navegador é redimensionada
onchange	Valor de um campo de entrada é alterado

A tabela acima apresenta apenas os principais eventos HTML que podem ser manipulados via JavaScript. Cada um dos eventos apresentados possui diferentes propriedades e comportamentos. Para mais detalhes verifique a documentação oficial.

Eventos: “onload” e “onscroll”

Carregar Título* | *Efetuar Contagem

Arquivo: ./javascript/DOM e Eventos/events/load_scroll.html

JavaScript

```
<html>
      <body onload="setTitle()" onscroll="add()" style="height:
1000px;">
            <h1 id="titulo"></h1>
            <h2 id="subtitulo"></h2>
            <h3 id="contador"></h3>
      </body>

</html>

<script type="text/javascript">
    let val = 0;

    function setTitle() {
```

```
        document.querySelector("#titulo").textContent =
"Events DOM - JavaScript";
        document.querySelector("#subtitulo").textContent =
"onload() | onscroll()";
        document.querySelector("#contador").textContent = val;
    }

    function add() {
        val++;
        document.querySelector("#contador").textContent = val;
    }

</script>
```

Events DOM - JavaScript

onload() | onscroll()

63

Eventos: “mouseover” e “keypress”

Aumentar Imagem | Exibir e Esconder

Arquivo: ./javascript/DOM e Eventos/events/mouseover_keypress.html

JavaScript

```
<html>
    <head>
        <style>
            .shadow {
                box-shadow: 10px 10px 7px darkgray;
                transition: box-shadow 0.5s ease-in-out;
            }
        </style>
    </head>
```

```
      <body onload="setTitle()">
              <h1 id="titulo"></h1>
              <h2 id="subtitulo"></h2>
              <img
                  id="img_fogo"

src="https://gileduardo.com.br/ifpr/dw/img/fundo-display.png"
                  width="280px"
                  height="120px"
                  onmouseenter="addShadow()"
                  onmouseout="removeShadow()"
              >
              <br>
          </body>
</html>

<script type="text/javascript">

    document.onkeypress = function (e) {

        let img = document.querySelector("#img_fogo")

        // Letra e/E
        if(e.keyCode == 101 || e.keyCode == 69) {
            img.style.visibility = "hidden"
        }
        // Letra m/M
        else if(e.keyCode == 109 || e.keyCode == 77) {
            img.style.visibility = "visible"
        }
    };

    function setTitle() {
        document.querySelector("#titulo").textContent =
"Events DOM - JavaScript";
        document.querySelector("#subtitulo").textContent =
"mouseover() | mouseOut() | keypress()";
```

```
    }

    function addShadow() {
        let img = document.querySelector("#img_fogo")
        img.classList.add("shadow")
    }

    function removeShadow() {
        let img = document.querySelector("#img_fogo")
        img.classList.remove("shadow")
    }

</script>
```

Events DOM - JavaScript

mouseover() | mouseOut() | keypress()

"M - Mostrar" / "E - Esconder" Imagem

Eventos: “click” e “focus”

Visibilidade da Imagem | Exibir e Esconder

Arquivo: ./javascript/DOM e Eventos/events/click_focus.html

JavaScript

```
<html>
      <head>
```

```
            <style>
                  .hide {
                        opacity: 0.2;
                        transition: opacity 2s ease-in-out;
                  }
                  .show {
                        opacity: 1;
                        transition: opacity 2s ease-in-out;
                  }
            </style>
      </head>
      <body onload="setTitle()">
             <h1 id="titulo"></h1>
              <h2 id="subtitulo"></h2>
              <img
                  id="img_fogo"

src="https://gileduardo.com.br/ifpr/dw/img/fundo-display.png"
                  width="280px"
                  height="120px"
                  onclick="setOpacityHide()"
              >
              <p ondblclick="setOpacityShow()">Visibilidade -
Duplo Click</p>
              <input id="texto" type="text" onfocus="imgHide()"
onblur="imgShow()" onkeypress="msgBox()">
      </body>
</html>
<script type="text/javascript">

    function setTitle() {
        document.querySelector("#titulo").textContent =
"Events DOM - JavaScript";
        document.querySelector("#subtitulo").textContent =
"click() | dblclick() | focus() | blur()";
    }
```

```
    function setOpacityHide() {
        let img = document.querySelector("#img_fogo")
        img.classList.remove("show")
        img.classList.add("hide")
    }

    function setOpacityShow() {
        let img = document.querySelector("#img_fogo")
        img.classList.remove("hide")
        img.classList.add("show")
    }

    function imgHide() {
        let img = document.querySelector("#img_fogo")
        img.style.visibility = "hidden"
    }

    function imgShow() {
        let img = document.querySelector("#img_fogo")
        img.style.visibility = "visible"
    }

    function msgBox() {
        let t = document.querySelector("#texto")
        alert(t.value)
    }
</script>
```

ATUALIZAÇÕES PERIÓDICAS

(https://www.w3schools.com/jsref/met_win_setinterval.asp)

setInterval() / clearInterval()

Função

Em JavaScript, a função setInterval() possibilita executar um bloco de código (uma função) repetidamente, com um intervalo de tempo predefinido. Tal funcionalidade mostra-se útil para criação de animações, atualizações de dados em tempo real, ou qualquer outra ação que precisa ser executada periodicamente.

Funcionamento

O método ***setInterval()*** invoca uma função (bloco de código) dentro de um intervalo de tempo pré-definido, em milissegundos. O setInterval() continua invocando a função até que o método ***clearInterval()*** seja chamado ou a janela seja fechada.

- *1 segundo = 1000 milissegundos.*

Arquivo: ./javascript/DOM e Eventos/setinterval/index.html

JavaScript

```
<!DOCTYPE html>
<html lang="pt-br">
```

```
    <head>
        <meta charset="UTF-8">
            <meta  name="viewport"  content="width=device-width,
initial-scale=1.0">
        <title>setInterval - JavaScript</title>
    </head>

    <body onload="startCounter()">
        <h1 class="init" id="titulo">setInerval JAVASCRIPT</h1>
        <h3 id="v">0</h3>
        <button id="btn_stop">PARAR</button>
        <button id="btn_continue">CONTINUAR</button>
    </body>

    <script type="text/javascript">
        let val = 0;
        let btn_s = document.querySelector('#btn_stop');
        let btn_c = document.querySelector('#btn_continue');
        let id;

        // FUNCTION
        function add() {
            val++;
            document.querySelector("#v").textContent = val;
        }
        // SET INTERVAL
        function startCounter() {
            id = setInterval(add, 1000)
        }
         // CLEAR INTERVAL
        function stopCounter() {
            clearInterval(id);
        }
        // CLICK EVENTS
        btn_s.addEventListener("click", stopCounter);
        btn_c.addEventListener("click", startCounter);
    </script>
```

```
</html>
```

setInerval JAVASCRIPT

79

PARAR CONTINUAR