# JavaScript

Sumário

[JavaScript 1](#_Toc118321615)

[Módulo A 1](#_Toc118321616)

[O que o JavaScript faz? 1](#_Toc118321617)

[Módulo B 1](#_Toc118321618)

[Variáveis e tipos primitivos 1](#_Toc118321619)

[Tratamento de dados 2](#_Toc118321620)

[Operadores 4](#_Toc118321621)

[Módulo C 6](#_Toc118321622)

[DOM – Document Object Model 6](#_Toc118321623)

[Eventos DOM 8](#_Toc118321624)

[Módulo D 8](#_Toc118321625)

## Módulo A

### O que o JavaScript faz?

Permite que o usuário interaja com o site

EcmaScript é a versão padronizada do JavaScript

Para criar um Script você deve abrir uma tag <script> dentro do <body> do documento HTML

Abrir uma Janela de alerta

window.alert('Minha primeira mensagem!');

Abre uma janela que pede uma confirmação

window.confirm('Está gostando de Java Script?')

Abre uma janela que você consegue escrever algo

Ab window.prompt('Qual é o seu nome');

## Módulo B

### Variáveis e tipos primitivos

Comentários // ou /\* \*/

É necessário criar uma varíavel, usando let ou var (só precisa criar na hora de declarar, após isso pode chamar a variável apenas pelo identificador)

Nome da variável = identificador = a1, r3, r4, valor, sexo etc

* Começam com letra $ ou \_
* NÃO PODE COMEÇAR COM MÚMEROS
* Pode ser escrita com letras e números
* Pode usar acentos e símbolos
* Não pode conter espaços, pode ser substituído por \_
* Não pode ter palavras reservadas

Var a1 =9 -> varíavel a1 recebe valor 9

A1 = 7 -> varíavel a1 deixa de possuir o valor 9 e passa a conter o valor sete

Para que uma varíavel que já tem um conteúdo fique vazia ela deve receber null

A1 = null

String – frase entre aspas – podem ser duplas “”, simples ‘’ ou estar entre crase``

Tipos de variáveis

* Number – 5, 98, -33, -.5, 3.2, -18.9, 8.0
* String - cadeia de caracteres – “curso em video”, “5”, “9 92900-7658”
* Boolean – true, false

Typeof – usado para saber o tipo da variável

### Tratamento de dados

var nome = window.prompt('Qual é o seu nome?');

        window.alert('É um grande prazer te conhecer, ' + nome);

concatenação - +

O mais pode ter o sentido de concatenação ou adição. Se as variáveis forem números irá somar, se forem strings irá concatenar

OBS: No caso de coletar dados usando window.prompt, o dado recebido será em string, logo é necessário tornar essa string um number

Conversão de tipo

* Number.parseInt() – converte um número em um número inteiro
* Number.paseFloat() - converte um número em um número decimal
* Number(n.value) – coloca em número e serve tanto para inteiro quanto real, o js decide de acordo com o valor recebido, nos outros os valores são forçados. N é a varíavel, e o value quer dizer para pegar o valor de n
* String() – de número para string
* n.toString() – de número para string, n é a variável

OBS: as letras maiúsculas são indispensáveis para que o código seja compreendido

Template string – caso você queira colocar as variáveis em um trecho sem usar a concatenação você pode colocas a variável assim ${n}(em vez de aspas simples tem que ser aberto dentro de crases ``), n é a variável, sem a necessidade de fechar as aspas e colocar o mais (placeholder)

Formatação de strings

* n.length – informa quantos caracteres a string tem, n é a variável
* n.toUpperCase()– coloca letras maiúsculas na String, n é a variável
* s.toLowerCase() – coloca letras minúsculas na String
* var nome = window.prompt('Qual é o seu nome?')
* document.write (`Olá ${nome}! Seu nome tem ${nome.length} letras, somente em minúsculas ele fica ${nome.toLowerCase()} e em maiúsculas fica ${nome.toLocaleUpperCase()}`)

Na tela

Olá Isabela! Seu nome tem 7 letras, somente em minúsculas fica isabela e em maiúsculas fica ISABELA

document.write – escreve na tela e não em uma janela

document.lnwrite – quebra uma linha

Dentro do document.write você pode colocar tags html

Formatação de números

* n.to.Fixed(número de casas decimais) – regula as casas decimais, n é a variável
* > var n1 = 1545.5
* undefined
* > n1.toFixed(2)
* '1545.50'

> n1 = 45.8794

45.8794

> n1

45.8794

> n1.toFixed(2)

'45.88' - > arredondou pra mais

* n.to.Fixed(número de casas decimais).replace(“.”, “,”) – regula as casas decimais e troca o ponto pela vítgula, n é a variável
* > n1 = 45.8794
* 45.8794
* > n1
* 45.8794
* > n1.toFixed(2).replace('.', ',')
* '45,88'
* n.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'}) – valor monetário, n é a variável
* n1.toLocaleString('pt-BR', {style: 'currency', currency: 'BRL'})
* 'R$ 54.986,46'
* USD = dólar
* EUR = euro

### Operadores

Aritméticos

* + - soma
* - - subtração
* \* - multiplicação
* / - divisão (real)
* % - módulo
* \*\* - elevado (potência)

Ordem de precedência

1. ()
2. \*\*
3. \*, /, % - caso tenha mais de um na mesma expressão eles são feitos da esquerda para a direita
4. +, -

Auto atribuição

* n = n+4 – a variável n vai receber a soma do seu valor mais o valor solicitado que no caso é 4:

var n = 5

n = n+4

n == 9

OBS: Dá para fazer essa auto atribuição com todos os operadores aritméticos

Forma da auto atribuição encurtada:

* n +=1
* n-= 6
* n\*=7
* n/=9

Obs dá para fazer o mesmo com % e \*\*

Incremento

* x = x +1 == x++
* x = x -1 == x—

Relacionais

* > - maior que
* < - menor que
* >= - maior ou igual
* <= - menor ou igual
* == - igual
* != - diferente
* === - idêntico ou igualdade restrita, mesmo valor e mesmo tipo
* !== - desigual restrito

OBS: os resultados dessas operações sempre serão valores booleanos, logo serão verdadeiro ou falso

Lógicos

* ! – negação
* && - conjução ou “e” -> todo tem que ser verdadeiros, para o resultado ser verdadeiro
* || - disjunção ou “ou” -> basta que um seja verdadeiro para que o resultado seja verdadeiro, logo o resultado só será falso se todos forem falsos

Os operadores lógicos seguem essa ordem de precedência: !, &&, ||

OBS: Primeiro você faz as operações aritméticas e depois as relacionais e por fim os lógicos

Ternário

* Teste?true:false – a depender do resultado do teste ele vai realizar uma ação se o resultado for verdadeiro e outra se for falso

## Módulo C

### DOM – Document Object Model

Ferramenta para o tratamento de componentes visuais do JS, o seu significado é documento de objetos para documentos, conjunto de objetos dentro do seu navegador, queda acesso aos componentes internos do seu site

Árvore DOM – elementos da arvore DOM baseado no exemplo do ex001 da aula 9

* window – tudo que está dentro do JS está dentro do window (janela)
  + location – qual é a localização do seu site (url), qual é a página atual, a página anterior etc.
  + document – documento atual
    - html
      * head
        + meta
        + title

OBS: e outras tags do dogumento

* + - * body
        + h1
        + p
        + p

strong

* + - * + div
  + history – guarda de onde você veio e para onde você vai, facilita a navegação dentro do site

OBS: Quem está dentro é child, e quem está fora é parente, logo document é parente de HTM e head é child de html

Algumas configurações da página usando o DOM:

* window.document.charset = mostra o set de caracteres, que no caso é o UTF-8
* window.navigator.appName = retorna o nome do navegador atual
* window.document.URL = qual é a url que você está acessando

Selecionando elementos da árvore DOM

* Por marca – getElementsByTagName(), com ele você consegue selecionar mais de um objeto, pois existem elementos com a mesma tag, no caso da árvore anterior nos temos o <p> por exemplo
* var p1 = window.document.getElementsByTagName('p')

A variável p1 vai receber o conteúdo dos parágrafos, que se encontram dentro do document que estão dentro da window

Se você quiser selecionar um parágrafo específico basta colocar [] com o número do parágrafo que você deseja após a declaração da tag, lembrando que o primeiro parágrafo é [0], o segundo [1] e assim sucetivamente

OBS: O window poderia ser apagado que o efeito seria o mesmo

var p1 = window.document.getElementsByTagName('p')[1]

        window.document.write(p1.innerText)

innerText vai pegar o texto que está no primeiro parágrafo, sem as tags filhas, no caso sem a formatação HTML, que no caso desse parágrafo teria o negrito, mas se você colocar no lugar de innerText você colocar ineerHTML o Strong irá aparecer, pois irá levar em consideração a formatação HTML e as tags filhas

p1.style.color = "blue"

O primeiro parágrafo terá a letra da fonte azul, nessa parte da para colocar os códigos CSS após o style

* Por ID – getElementsById() – pega pela propriedade id do HTML

No HTML:

<div id = "msg">Clique em mim</div>

No Script

* var d = document.getElementById('msg')
* d.style.background = 'green'
* Por Nome – getElementsByName() – pega pela propriedade name do HTML

No HTML:

<div name = "msg">Clique em mim</div>

No Script

* var d = document.getElementsByName('msg')[0] -> nesse caso tem que colocar colocar a qual elemento você está se referindo, assim como na parte de tags, já que são elements
* d.style.background = 'green'
* Por classe – getElementsByClassName () – pega pela propriedade class do HTML

No HTML:

<div class = "msg">Clique em mim</div>

No Script

* var d = document.getElementByClassName('msg')
* d.style.background = 'green'
* Por seletor – querySelector() ou querySelectorAll()
* NoHTML
* <div id = "msg">Clique em mim</div>
* No Script
* var d = document.querySelector('div#msg')

Escreve como se fossem os seletores do CSS

Para pegar o valor do objeto tem que colocar .value após você solicitar o elemento usando get element

### Eventos DOM

O evento é tudo o que pode acontecer com a div

Alguns eventos:

* mouseenter – ocorre quando o mouse encosta no elemento
* mousemove - ocorre enquanto o mouse se move dentro do elemento
* mousedown - ocorre quando você clica no elemento
* mouseup - ocorre quando você solta o botão de clicar após clicar no elemento
* click - ocorre quando você aperta e solta rapidamente
* mouseout – ocorre quando o mouse sai do elemeto

OBS: Para encontrar mais você pode procurar por JavaScript DOM events list

Função no JS

É um conjunto de códigos (bloco) que são executadas só quando um evento ocorre

function ação(parâmetros){

bloco

}

Os eventos podem tanto ser configuradas na parte do HTML quanto na parte do JS

OBS: Você pode criar uma variável para cada function ou colocar ela do lado de fora que irá servir para todas

Disparando eventos no HTML

<body>

    <div id="area" onclick="clicar()" onmouseenter="entrar()" onmouseout="sair()"> <!--quando clicarem dentro da div o evento "clicar" irá acontecer, a chamada para a function precisa ter o ()-->

        Interaja..

    </div>

    <script>

        var a = document.querySelector('div#area')

        function clicar(){

            a.innerText = 'Clicou!'

        }

        function entrar(){

            a.style.background = 'red'

        }

        function sair(){

            a.style.background = 'green'

            a.innerText = 'Saiu!'

        }

    </script>

</body>

Disparando eventos no Script

<body>

    <h1>Chamando eventos pelo Script</h1>

    <div id="area">

        Interaja..

    </div>

    <script>

        var a = document.querySelector('div#area')

        //prestando atenção em um determinado enveto

        a.addEventListener('click', clicar)

        a.addEventListener('mouseenter', entrar)

        a.addEventListener('mouseout', sair)

        function clicar(){

            a.innerText = 'Clicou!'

        }

        function entrar(){

            a.style.background = 'red'

        }

        function sair(){

            a.style.background = 'green'

            a.innerText = 'Saiu!'

        }

    </script>

</body>

Na parte do JS não mostra quais são os seus erros, mas se você entrar na parte de inspecionar do seu site os erros irão aparecer

Evento pegando valores

<body>

    <h1>Somando valores</h1>

    <input type="number" name="txtn1" id="txtn1"> +

    <input type="number" name="txtn2" id="txtn2">

    <input type="button" value="Somar" onclick="somar()">

    <div id="res">Resultado: </div>

    <script>

        function somar(){

            var tn1 = document.getElementById('txtn1')

            var tn2 = document.querySelector('input#txtn2')

            //como eles vem em uma caixa de texto eles estão como string

            var soma = Number(tn1.value) + Number(tn2.value)

            var res = document.getElementById('res')

            res.innerHTML = `Resultado: ${soma}`

        }

    </script>

</body>

## Módulo D

### Condições

Para executar algo no node.js basta colocar o arquivo como .js e depois clicar em f8 para abrir o terminal, não são todos os comandos que funcionam no JS do HTML que irão funcionar no node.js, um exemplo é o document.write(), em vez disso você deve usar o console.log()

Tipos de condições

* Simples – se a condição for verdadeira algo vai acontecer, senão não ocorrerá nada

if (condição) {

true

}

* Composta - Se a condição for verdadeira vai executar a parte do true, senão vai executar o que está na parte false

if (condição) {

true

}else{

false

}

* Condições aninhadas

if (condição) {

true

}else{

if (condição) {

true

}else{

false

}

}

Para pegar a hora atual

var agora = new Date() // pega as informações de hove

var hora = agora.getHours() // pega a hora que está dentro das informações que foram recebidas na variável agora

* Condição múltipla – valores definidos

Switch(expressão){

case valor 1:

bloco 1

break

case valor 2:

bloco 2

break

case valor 3:

bloco 3

break

case valor 4:

bloco 4

break

default:

bloco 5

break

}

OBS: o uso do break é obrigatório

caixa\_img.src = 'imagens/tarde.png'

para trocar a imagem no JS tem que colocar a variável e em seguida .src, por fim é só colocar o link da imagem

<body onload="carregar()">

Quando a página carregar, a function carregar() vai iniciar

Criando imagem em js

var img = document.createElement('img') //vai criar uma tag img

img.setAttribute('id', 'img') //para criar um id = 'img' para a imagem dentro da variável img

img.setAttribute('src', 'imagens/mcriança.png') // vai alterar o src para que aparecça a imagem desejada

res.appendChild(img) //diz que a variável img que nos criamos com a foto será uma filha da variável foto, que contem o elemento da div com o id foto